An Improved Pelican Optimization Algorithm Based on Chaos Mapping Factor

Yutong Li, Yu Liu*, Chunli Lin, Jiayao Wen, Pengguo Yan and Yukun Wang

Abstract—**The Pelican Optimization Algorithm (POA) is a meta-heuristic algorithm that emulates the behavior and hunting strategy of pelicans. This paper presents three modifications that enhance the algorithmic traversal of POA by incorporating ten chaotic mapping factors. These modifications aim to improve the efficiency and accuracy of the traversal process. Firstly, the pelican optimization algorithm with chaotic mapping factors added to population generation but random prey generation (COPOA); secondly, the pelican optimization algorithm with chaotic mapping factors applied to prey generation but random population generation (OCPOA); finally, the pelican optimization algorithm with the same chaotic mapping factor, different sequences generated by different random initial values are mapped to the population and prey generation (CCPOA). The results of simulations on 12 benchmark test functions demonstrate that the algorithm (COPOA) adding the chaotic mapping factor to the population generation alone has higher generalizability, convergence speed and finding accuracy compared with the standard POA algorithm. Among these modifications, the Sine-COPOA method performed the best in ten chaotic mapping factor comparison tests. Furthermore, a comparison is made between the Sine-COPOA and typical intelligent optimization algorithms to assess its competence in optimization. The results show that Sine-COPOA performs better in most of the 12 benchmark test functions. Finally, the engineering design problems (pressure vessel problem and 10-bar truss design) are optimized. The experimental results demonstrate that Sine-COPOA effectively solves both function optimization and engineering optimization problems.**

Index Terms—**Pelican Optimization Algorithm, Chaotic mapping factor, Metaheuristic Algorithm, Traversal**

Manuscript received April 20, 2023; revised September 7, 2023.

This work was supported in part by the Scientific Research Project of Liaoning Provincial Department of Education (Grant no. LJKZ0320) for providing financial support for the study.

Yu-Tong Li is a postgraduate student at School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (e-mail: 13942871038@163.com).

Yu Liu is a supervisor of postgraduate of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan,114051, P. R. China (Corresponding author, phone: 86-0412-5929747; fax:86-0412-5929747; lnasacl@126.com)

Chun-Li Lin is a supervisor of postgraduate of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (e-mail: 356408163@qq.com).

Jia-Yao Wen is a postgraduate student of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (e-mail: m17641243544@163.com).

Peng-Guo Yan is an undergraduate student of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (e-mail: yan407319226@163.com).

Yu-kun Wang is a supervisor of postgraduate of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan,114051, P. R. China (e-mail: wyk410@163.com)

I. Introduction

The swarm intelligence optimization algorithm is an intelligent technology that has emerged in recent years to achieve an optimal meta-heuristic algorithm for solving complex problems by simulating natural population laws. The iterative process of the optimization algorithm [1] aims to facilitate the search for the optimal solution by solving the optimal value of the optimization problem. The Bionic Algorithm is a popular heuristic algorithm utilized in optimization to solve a variety of problems. The Bionic Algorithm is applied to various types of optimization problems, such as Numerical Optimization [2]-[3], where the decision variables typically involve continuous functions within specific domains to determine the optimal value. Combinatorial Optimization [4]-[6], on the other hand, deals with arranging or combining discrete variables in accordance with specific constraints and rules. Engineering design optimization problems [7] involve both continuous and discrete decision variables, depending on the specific engineering problem. These problems typically have constraints, including equation or inequality constraints, which vary depending on the problem at hand. Bionic algorithms, such as the Ant Colony Algorithm [8], simulate the foraging behavior of ants, while the Particle Swarm Optimization Algorithm [9] mimics the flight patterns of birds. The field of intelligent optimization algorithms is continuously evolving, with numerous new algorithms emerging in recent years. Examples of such include the Squirrel Search Algorithm [10], the Firefly Optimization Algorithm [11], the Crow Search Algorithm [12], the Salp Swarm Algorithm [13], and Bat Algorithm[14].

Previously, several metaheuristic optimization algorithms have been enhanced by incorporating chaotic mapping factors. These algorithms include Genetic Algorithms [15], Particle Swarm Optimization Algorithm [9], Harmony Search [16], Ant Colony Algorithm [8], Bee Colony Optimization [17], Imperialistic Competitive Algorithms [18], Firefly Algorithms [11], and Simulated Annealing [19]-[20]. The utilization of suitable chaotic mappings in combination with metaheuristic algorithms demonstrates promising prospects. Heidari proposed integrating chaotic patterns into the stochastic process of the Water Cycle Algorithm [21]. Zhao et al. proposed a hybrid Bacterial Foraging Algorithm that combines its adaptive strategy with chaotic search to enhance the efficiency of solving complex optimization problems [22]. Gupta proposed the OCS-GWO algorithm, a chaotic Grey Wolf Optimization Algorithm based on opposition, to enhance the performance of the Grey Wolf Algorithm [23]. Tang et al. optimized the Ant Colony Algorithm through parameter optimization, simulating both the chaotic behavior of individual ants and the self-organizing behavior of ant colonies [24]. Gandomi et al. enhanced the global search capability of the variant Particle Swarm Optimization by introducing chaos [25]. Jordehi incorporated chaotic mappings into Artificial Immunity Algorithms to address the issue of premature convergence [26]. Chuang et al. enhanced the performance of the Catfish Particle Swarm Algorithm by substituting the stochastic parameters with chaotic sequences [27]. Priyanka Anand et al. proposed the Chaotic Self-interested Group Optimization Algorithm to enhance the global convergence speed and performance [28]. Long et al. incorporated the Sine chaotic mapping into the Honey Badger Algorithm to initialize the population [29]. Han et al. proposed a chaotic system-based perceptron neural network image encryption algorithm[30].

The Pelican Optimization Algorithm (POA) was proposed by Pavel Trojovský and Mohammad Dehghani in 2022 [31]. The algorithm emulates the hunting behavior of pelicans. Section 2 presents ten commonly used chaotic mapping factors. The formula for generating negative values has been optimized to expand the range of random values within the interval (0, 1). Section 3 incorporates ten commonly used chaotic mapping factors in the POA algorithm to facilitate population and prey generation. These factors are implemented using three different approaches. Section 4 compares the convergence speed and finding accuracy of adding chaotic mapping factors at different locations, using 12 benchmark test functions. Furthermore, the comparison includes the Sine-COPOA algorithm with both classic and new intelligent optimization algorithms, conducting experiments using 12 benchmark test functions.

II. Pelican Optimization Algorithm

The Pelican Optimization Algorithm emulates the behavior and strategies of pelicans during attacking and hunting to update candidate solutions. The hunting process of the algorithm involves two phases: approaching the prey (exploration phase) and flying over water (exploitation phase).

A. Initialization phase

The mathematical description of initializing pelican populations is defined as follows.

$$X_{i,j} = l_j + rand*(u_j - i_j) \quad i = 1,2,...,N, j = 1,2,...,m \tag{1}$$

Where X_{ij} is the jth variable value of the ith candidate solution, N is the number of pelican populations, m is the problem solving dimension, $rand$ is a random number in interval (0,1), l_j is the lower bound of the jth problem variables, and u_j is the upper bound of the jth problem variables.

In POA, the pelican population matrix is shown in (2).

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} X_{1,1} & \cdots & X_{i,j} & \cdots & X_{1,m} \\ \vdots & \ddots & \vdots & \cdots & \vdots \\ X_{i,1} & \cdots & X_{i,j} & \cdots & X_{i,m} \\ \vdots & \cdots & \vdots & \ddots & \vdots \\ X_{N,1} & \cdots & X_{N,j} & \cdots & X_{N,m} \end{bmatrix}_{N\times m} \tag{2}$$

Where X is the pelican population matrix, and X_i is the ith pelican.

Each pelican represents a candidate solution for the given problem. The objective function is calculated for each candidate solution in order to solve the problem. The value of the objective function is determined using the objective function vector, as shown in (3).

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1} \tag{3}$$

where F is the objective function vector, F_i is the objective function value of the ith candidate solution.

B. Phase 1: Moving towards Prey (Exploration Phase)

In the first phase, the pelican determines the location of the prey and moves towards this area. By modelling the pelican's strategy for approaching prey, the POA algorithm can scan the search space and perform different exploration capabilities in different search areas. The above concept is mathematically defined as follows.

$$X_{i,j}^{P_1} = \begin{cases} X_{i,j} + \text{rand}\cdot(P_j - I\cdot X_{i,j}), & F_p < F_i \\ X_{i,j} + \text{rand}\cdot(X_{i,j} - P_j) \end{cases} \tag{4}$$

Where $X_{i,j}^{P_1}$ is the new state of the ith pelican in jth dimension under phase 1, I is a random number equal to 1 or 2, P_j is the position of the jth dimension of the prey, and F_P is its objective function value.

In POA, the new position of the pelican is accepted if the value of the objective function is optimized at that position, which is called effective update. In the effective update, the algorithm is prevented from moving to a non-optimal region. The process is shown in (5).

$$X_i = \begin{cases} X_i^{P_1}, F_i^{P_1} < F_i \\ X_i, else \end{cases} \tag{5}$$

Where $X_i^{P_1}$ is the new state of the ith pelican and $F_i^{P_1}$ is its objective function value based on phase 1.

C. Phase 2: Winging on the Water Surface (Exploitation Phase)

During the second phase, modeling the pelican's hunting strategy can enhance the algorithm's abilities for local search and development. Mathematically, the algorithm is required to assess points in close proximity to the pelican's position in order to converge towards an improved solution. Mathematical definitions of pelican hunting strategies are as follows.

$$X_{i,j}^{P_2} = X_{i,j} + R\cdot(1-\frac{t}{T})\cdot(2\cdot\text{rand}-1)\cdot X_{i,j} \tag{6}$$

where $X_{i,j}^{P_2}$ is the new state of the ith pelican in jth dimension under phase 2, R is a constant equal to 0.2, $R\cdot(1-t/T)$ is the neighborhood radius of $X_{i,j}$. t is the number of iterations, and T is the maximum number of iterations.

During this phase, the effective updating is shown in (7).

$$X_i = \begin{cases} X_i^{P_2}, F_i^{P_2} < F_i \\ X_i, else \end{cases} \quad (7)$$

Where $X_i^{P_2}$ is the new state of the ith pelican and $F_i^{P_2}$ is its objective function value based on phase 2.

III. Chaotic Maps

In swarm intelligence algorithms, the use of random numbers to create populations can lead to slow convergence, limited search range, and a lack of population diversity. In a study conducted by researchers [32], it was demonstrated that the initial population directly affects the accuracy and speed of algorithm convergence, as well as the overall performance in cases where the population has poor diversity. Chaotic mapping factors are commonly employed in initialization of populations for optimization algorithms. Chaotic sequences possess properties such as traversal, randomness, and regularity within a specific range. Compared to random search, chaotic sequences have a higher probability of thoroughly exploring the search space while maintaining population diversity. Numerous types of chaotic mappings exist in the field of optimization, and Table I presents ten standard chaotic mapping formulas along with their respective parameters.

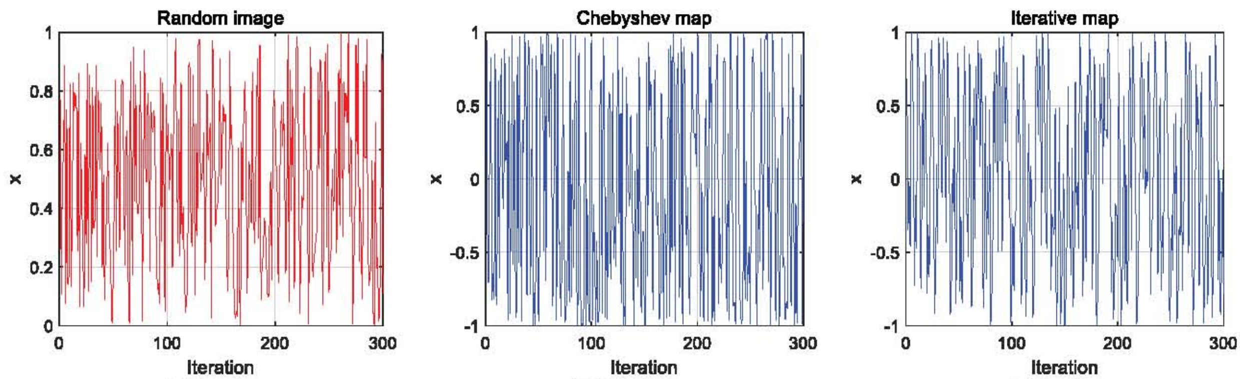
rand(N,dim) generates random numbers from a uniform distribution between (0,1). Fig.1 (a) illustrate the distribution plots of these random numbers and Fig.2 (a) illustrate the frequency distribution plots of these random numbers. The chaos mapping factor replaces the use of *rand(N,dim)* to generate random numbers within the range of 0 and 1. The Chebyshev map and Iterative map formulas produce negative values within the range of -1 and 1, as displayed in Fig.1 (b) and Fig.2 (b), Fig.1 (c) and Fig.2 (c). As displayed in Table II, the two aforementioned chaos mapping formulas were modified by incorporating absolute values to enhance the diversity of values within the interval (0,1). The visualized images and frequency plots for the numerical distribution of the ten improved chaos mapping factors are depicted in Fig.3. The results above indicate that, by combining the initial populations with chaotic mapping factors, the population diversity can be enhanced due to the random nature and effective traversal of chaotic sequences and achieve better convergence and optimization accuracy under test functions.

TABLE I
CHAOS MAPPING FORMULATION

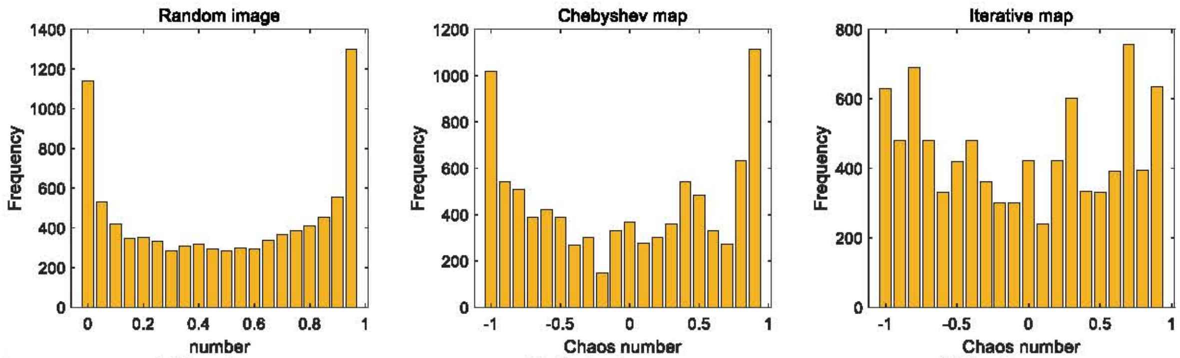
Formula Name	Formula	Parameter Value
Sine map	$X_{k+1}=\frac{\alpha}{4}\sin(\pi X_k), \alpha\in(0,4]$	$\alpha = 3.96$
Iterative map	$X_{k+1}=\sin\left(\frac{\alpha\pi}{X_k}\right), \alpha\in(0,1)$	$\alpha=0.7$
Chebyshev map	$X_{k+1}=\cos\left(k\cos^{-1}(X_k)\right)$	
Singer map	$X_{k+1}=\mu(7.86X_k-23.31X_k^2+28.75X_k^3-13.302875X_k^4$ $\mu\in(0.9,1.08)$	$\mu=1.07$
Circle map	$X_{k+1}=\mathrm{mod}\left(X_k+b-\frac{\alpha}{2\pi}\sin(2\pi X_k),1\right)$	$\alpha=0.5$ $b=0.2$
Cubic map	$X_{k+1}=\rho(1-X_k^2)$	$\rho=2.595$
Tent map	$X_{k+1}=\begin{cases}\frac{X_k}{\alpha}, & X_k<\alpha \\ \frac{(1-X_k)}{(1-\alpha)}, & X_k\ge\alpha\end{cases}$ $\alpha\in(0,1)$	$\alpha=0.7$
Piecewise map	$X_{k+1}=\begin{cases}\frac{X_k}{P}, & 0\le X_k<P \\ \frac{X_k-P}{0.5-P}, & P\le X_k\le 0.5 \\ \frac{1-P-X_k}{0.5-P}, & 0.5\le X_k<1-P \\ \frac{1-X_k}{P}, & 1-P\le X_k<1\end{cases}$	$P=0.4$
Sinusoidal map	$X_{k+1}=\alpha X_k^2\sin(\pi X_k)$	$\alpha=2.35$
Logistic map	$X_{k+1}=\alpha X_k(1-X_k)$	$\alpha=4$

TABLE II
IMPROVED CHAOTIC MAPPING FORMULATION

Formula Name	Formula	Parameter Value
Chebyshev map	$X_{k+1}=\left\|\cos\left(k\cos^{-1}(X_k)\right)\right\|$	
Iterative map	$X_{k+1}=\left\|\sin\left(\frac{\alpha\pi}{X_k}\right)\right\|, \alpha\in(0,1)$	$\alpha=0.7$

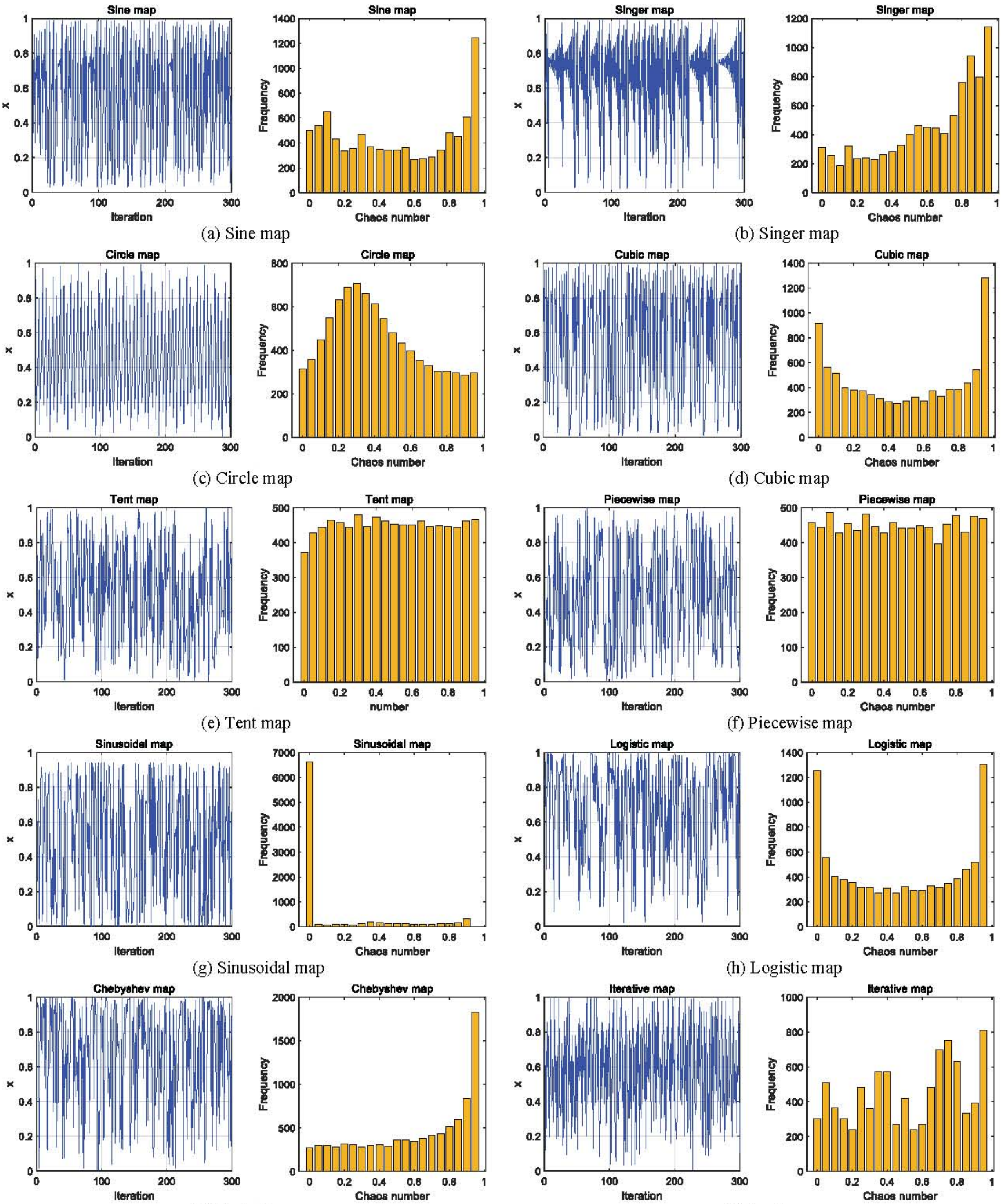


(a) Random (b) Chebyshev map (c) Iterative map
Fig.1. *rand (N, dim)*, Chebyshev and Iterative map generate distribution figures of random numbers between (0,1)



(a) Random (b) Chebyshev map (c) Iterative map

Fig.2. *rand* (*N, dim*), Chebyshev and Iterative map generate frequency distribution figures of random numbers between (0,1)



(a) Sine map (b) Singer map

(c) Circle map (d) Cubic map

(e) Tent map (f) Piecewise map

(g) Sinusoidal map (h) Logistic map

(i) Chebyshev map (j) Iterative map

Fig.3. Improved visualization of random numbers between (0,1) generated by ten chaotic mappings

This paper introduces the incorporation of the chaos map formulation into the POA algorithm and presents three comparative ideas. The first one is incorporating the chaotic map into the generation of pelican populations while randomly generating prey. Refer to (8)-(11) using the Sine map formulation as an example.

$$Z_{pelican_0} = rand \tag{8}$$

$$Z_{\mathrm{pelican}_{k+1}} = \frac{\alpha}{4}\sin(\pi Z_{\mathrm{pelican}_k}), \alpha \in [0,4) \tag{9}$$

$$X_{pelican_{i,j}} = l_j + Z_{pelican} \cdot (u_j - l_j) \tag{10}$$

$$i = 1,2,\ldots,N,\ j = 1,2,\ldots,m$$

$$X_{prey_{i,j}} = l_i + rand \cdot (u_j - l_j) \tag{11}$$

$$i = 1,2,\ldots,N,\ j = 1,2,\ldots,m$$

Where $Z_{pelican_0}$ is the randomly generated initial value, $Z_{prlican_{k+1}}$ is the random number generated by the pelican chaos sequence, $X_{pelican_{i,j}}$ is the value of the jth variable specified by the ith candidate solution, $X_{prey_{i,j}}$ is the value of the jth variable specified by the ith prey.

The second approach involves incorporating a chaotic mapping solely in the prey generation phase, while the population of pelicans is randomly generated. The principle of the formula is the same as the first one.

The third one incorporates a chaotic mapping into the generation of the pelican population and prey, employing two different random initial values Z_0 to produce two distinct chaotic sequences within the range of (0,1). Refer to (12)-(17) using the Sine map formulation as an example.

$$Z_{pelican_0} = rand \tag{12}$$

$$Z_{prey_0} = rand \tag{13}$$

$$Z_{\mathrm{pelican}_{k+1}} = \frac{\alpha}{4}\sin(\pi Z_{\mathrm{pelican}_k}), \alpha \in [0,4) \tag{14}$$

$$Z_{\mathrm{prey}_{k+1}} = \frac{\alpha}{4}\sin(\pi Z_{\mathrm{prey}_k}), \alpha \in [0,4) \tag{15}$$

$$X_{pelican_{i,j}} = l_j + Z_{pelican} \cdot (u_j - l_j) \tag{16}$$

$$i = 1,2,\ldots,N,\ j = 1,2,\ldots,m$$

$$X_{prey_{i,j}} = l_i + rand \cdot (u_j - l_j) \tag{17}$$

$$i = 1,2,\ldots,N,\ j = 1,2,\ldots,m$$

Where $Z_{pelican_0}$ and Z_{prey_0} is the randomly generated initial value, $Z_{prlican_{k+1}}$ is the random number generated by the pelican chaos sequence, $Z_{prey_{k+1}}$ is the random number generated by the prey chaos sequence.

To clearly show the structure of the proposed three algorithm, the COPOA pseudo-code and flowchart are shown in Algorithm 1 and Fig.4, the OCPOA pseudo-code and flowchart are shown in Algorithm 2 and Fig.5, the CCPOA pseudo-code and flowchart are shown in Algorithm 3 and Fig.6. The pseudo-codes and flowcharts can intuitively help understand the program framework.

Algorithm 1. Pseudo-code of COPOA.

Start COPOA.
1. Input information about the optimization problem.
2. Determine the pelican population size (N) and the algorithm number of iterations (T).
3. Randomly generated initial values $Z_{pelican_0} = rand$.
4. Generate chaotic sequences ($Z_{pelican} = (N, \mathrm{dim})$)in the range of (0,1) by chaotic mapping formula using (9).
5. Initialization of pelican population location by $X_{pelican_{i,j}} = (N, \mathrm{dim})$ using (10) and calculate the objective function.
6. For t = 1:T
7. Generate the locations of the prey population at random.
8. For I = 1:N
9. Phase 1: Moving towards prey (exploration phase).
10. For j = 1:m
11. Calculate new state of pelican at the jth dimension using (4).
12. End.
13. Update the state of the ith pelican population member using (5).
14. Phase 2: Winging on the water surface (exploitation phase).
15. For j = 1:m.
16. Calculate new state of the pelican at the jth dimension using (6).
17. End.
18. Update the state of ith pelican population member using (7).
19. End.
20. Update the best candidate solution.
21. End.
22. Output the best candidate solution obtained by COPOA.

End COPOA.

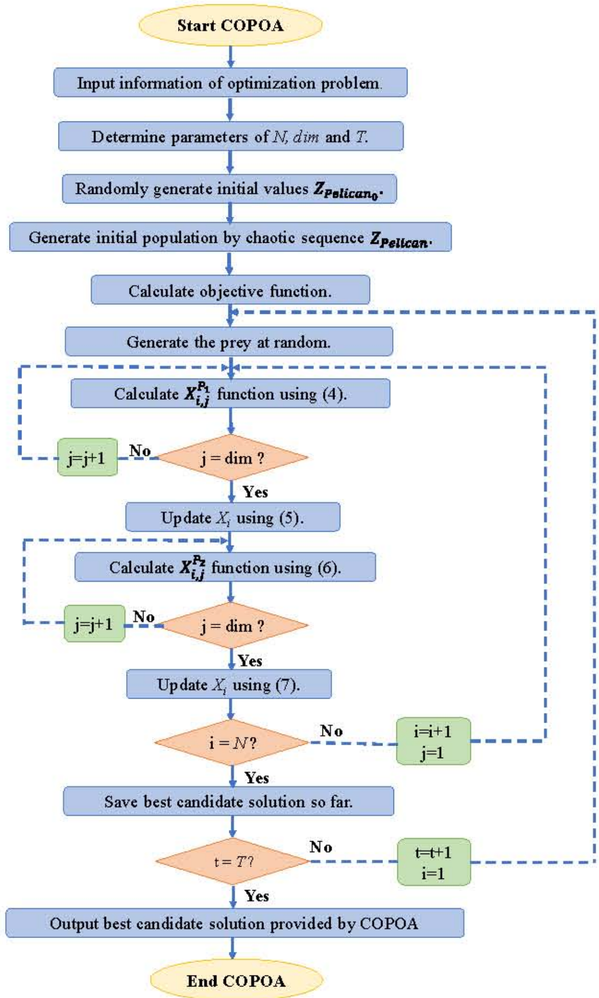


Fig.4. Flowchart of COPOA

Algorithm 2. Pseudo-code of OCPOA.

Start OCPOA.

1. Input information about the optimization problem.
2. Determine the pelican population size (N) and the algorithm number of iterations (T).
3. Randomly generated initial value $Z_{prey_0} = rand$.
4. Generate chaotic sequences ($Z_{prey} = (N, \dim)$) in the range of (0,1) by chaotic mapping formula and using (15).
5. Initialization of pelican population location at random and calculate the objective function.
6. For t = 1:T
7. Generate the locations of the prey population by $X_{prey} = (N, \dim)$ using (17).
8. For I = 1:N
9. Phase 1: Moving towards prey (exploration phase).
10. For j = 1:m
11. Calculate new state of pelican at the jth dimension using (4).
12. End.
13. Update the state of the ith pelican population member using (5).
14. Phase 2: Winging on the water surface (exploitation phase).
15. For j = 1:m.
16. Calculate new state of the pelican at the jth dimension using (6).
17. End.
18. Update the state of ith pelican population member using (7).
19. End.
20. Update the best candidate solution.
21. End.
22. Output the best candidate solution obtained by OCPOA.

End OCPOA.

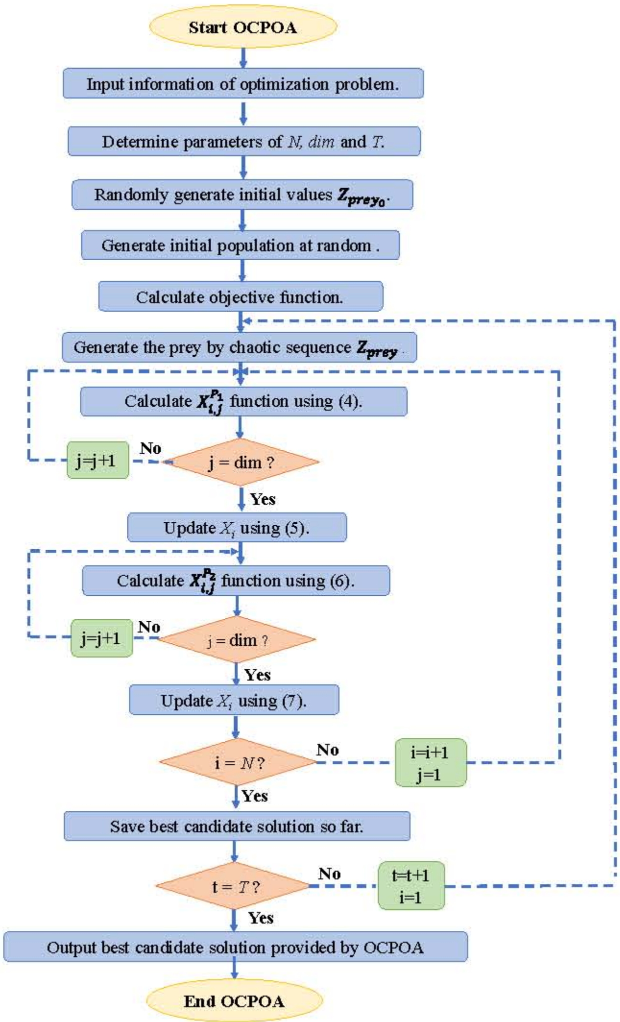


Fig.5. Flowchart of OCPOA

Algorithm 3. Pseudo-code of CCPOA.

Start CCPOA.

1. Input information about the optimization problem.
2. Determine the pelican population size (N) and the algorithm number of iterations (T).
3. Randomly generated initial values $Z_{pelican_0} = rand$, $Z_{prey_0} = rand$.
4. Generate chaotic sequences ($Z_{pelican} = (N, \dim)$, $Z_{prey} = (N, \dim)$) in the range of (0,1) by chaotic mapping formula using (14) and using (15).
5. Initialization of pelican population location by $Z_{pelican} = (N, \dim)$ using (16). and calculate the objective function.
6. For t = 1:T
7. Generate the locations of the prey population by $Z_{prey} = (N, \dim)$ using (17).
8. For I = 1:N
9. Phase 1: Moving towards prey (exploration phase).
10. For j = 1:m
11. Calculate new state of pelican at the jth dimension using (4).
12. End.
13. Update the state of the ith pelican population member using (5).
14. Phase 2: Winging on the water surface (exploitation phase).
15. For j = 1:m.
16. Calculate new state of the pelican at the jth dimension using (6).
17. End.
18. Update the state of ith pelican population member using (7).
19. End.
20. Update the best candidate solution.
21. End.
22. Output the best candidate solution obtained by CCPOA.

End CCPOA.

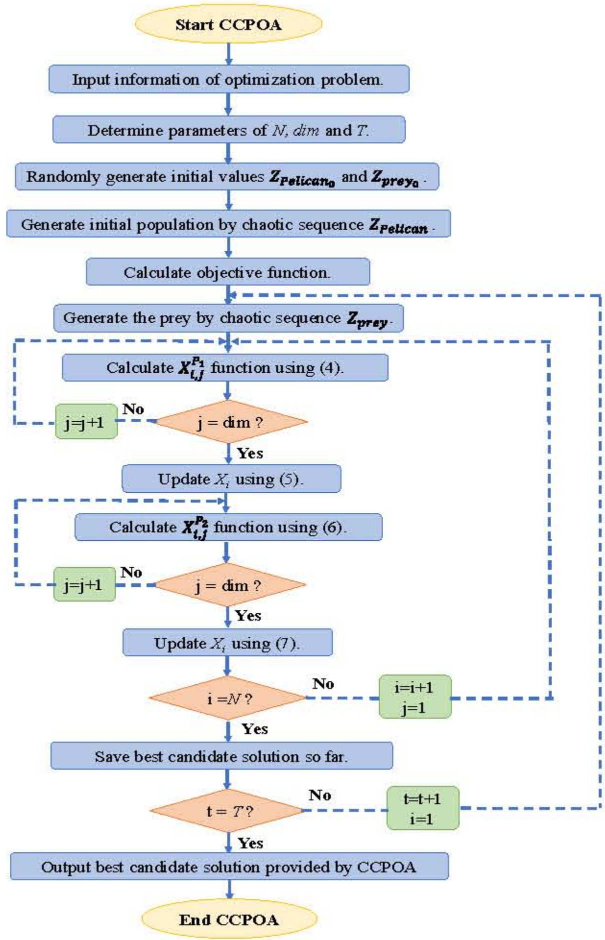


Fig.6. Flowchart of CCPOA

IV. Numerical Simulations and Results

To evaluate the convergence speed and accuracy of the aforementioned three combination methods, each algorithm independently executed 30 times on 12 widely used benchmark test functions from Table III. This helped minimize the impact of randomness on the function solutions. The average and optimal values of the test functions were recorded for comparative analysis. The common parameters for all algorithms were set as follows: populations size N is set to 100, maximum iterations T is set to 1000.

This section shows the convergence graphs of f_1, f_3, f_6 under unimodal functions, f_9, f_{10}, f_{12} under multimodal functions among the twelve test functions.

Based on the experimental results presented in Table IV and Fig.7, for functions f_1 and f_2, convergence speed when the POA is combined with each chaotic map formula is faster and the optimization accuracy is higher compared to the original algorithm. Additionally, when the POA is combined with each chaotic map formula achieves an optimal value of 0 for function f_1. The Sine map is incorporated into the POA. The Sine-CCPOA achieves higher accuracy for function f_2, while the original POA performs better for function f_7 function. The Sine-COPOA demonstrates the highest optimization accuracy with the rest of the functions, and it exhibits fast convergence speed for functions f_3, f_6, $f_9 \sim f_{12}$. The Singer map has been integrated into the POA. Singer-COPOA demonstrates favorable performance in functions f_3, f_4, f_5, f_9, f_{10}, f_{11}, f_{12}, with faster convergence rates observed for functions f_3, $f_9 \sim f_{12}$ in comparison to the other two methods. Moreover, Singer-CCPOA exhibits higher optimization accuracy in functions f_2, f_6, and f_{13}. However, none of the three methods show improvements in terms of both convergence speed and accuracy in f_7. The Circle map is incorporated into the POA, and the Circle-COPOA optimization demonstrates strong performance for functions f_5, f_6, f_9, f_{10}, f_{11}, f_{12}, and f_{13}, with fast convergence for functions $f_9 \sim f_{12}$. On the other hand, the Circle-OCPOA optimization shows superior performance for f_2, while none of the three methods enhance the performance of the original algorithm for functions f_3 and f_4. When the cubic map is incorporated into the POA, the Cubic-COPOA demonstrates high optimization accuracy in functions f_2, f_5, f_6, f_9, f_{10}, f_{11}, f_{12}, f_{13}, Additionally, it exhibits faster convergence speed in functions f_6, f_9, f_{10}, f_{11}, and f_{12}. However, the inclusion of the three methods does not improve the optimization of the original algorithm in functions f_3, f_4, and f_7. When the Tent map is added to the POA, Tent-COPOA perform well for functions f_3, f_5, f_6, f_9, f_{10}, f_{11}, f_{12}, and f_{13}, and the convergence speed is faster under the same functions. Tent-OCPOA outperformed the other methods under f_2, while the other three methods did not optimize the original POA under functions f_4, f_7. With the inclusion of the piecewise map in the POA. The Piecewise-COPOA demonstrates strong performance under the functions f_4, f_5, f_6, f_9, f_{10}, f_{11}, f_{12} and f_{13}. It also converges at a faster rate under the functions f_6, f_{10}, f_9, f_{11}, f_{12}. On the other hand, the Piecewise-CCPOA shows superior performance under the functions f_2. The other three methods fail to optimize the original POA algorithm under the functions f_3 and f_7. By incorporating the Sinusoidal map into the POA, Sinusoidal-COPOA demonstrates significant optimization effect in functions f_3, f_6, f_9, f_{10}, f_{11}, f_{12}, and achieves faster convergence compared to the original functions. Sinusoidal-OCPOA exhibits a noteworthy optimization effect in the f_2 and f_{13}, while Sinusoidal-CCPOA shows superior optimization for function f_5. Nevertheless, the three methods do not exhibit any optimization effect on the original POA in functions f_4 and f_7. The addition of the Logistic map to the POA algorithm results in improved Logistic-COPOA optimization under functions f_3, f_5, f_6, f_9, f_{10}, f_{11}, f_{12}, and f_{13}. Additionally, it leads to faster convergence under functions f_9, f_{10}, f_{11}, and f_{12}. Logistic-OCPOA outperforms the other three methods when applied to function f_2. However, none of the methods optimize the original POA algorithm for functions f_3 and f_7. With the addition of the Chebyshev map, the Chebyshev-CCPOA algorithm demonstrates a good finding effect for f_5, f_6, f_7, f_{12}, f_{13}, and the convergence speed is faster under the same functions. The Chebyshev-COPOA algorithm demonstrates a positive optimization effect for f_3, f_4, f_9, f_{10}, f_{11}, and shows faster convergence for f_3, f_9, f_{10}, and f_{11}. The Chebyshev-OCPOA algorithm performs well for function f_2. With the addition of Iterative map into POA, Iterative-CCPOA exhibits effective optimization results for f_2, f_5, f_6, f_7, f_{12}, f_{13}, and achieves faster convergence for functions f_6, f_7, f_{12} and f_{13}. Iterative-COPOA demonstrates effective optimization outcomes for f_3, f_9, f_{10}, f_{11}, while also exhibiting faster convergence under the same functions. None of the three methods optimize the standard POA for function f_4.

The convergence curves obtained by combining the ten chaotic mapping factors listed above with the standard POA show that, under the COPOA, the convergence curves of the Sine, Chebyshev, Circle, Cubic, Singer, Tent, Piecewise, Sinusoidal, and Logistic maps are faster and more accurate when compared to OCPOA and CCPOA. Among the 12 benchmark test functions analyzed, Sine-COPOA demonstrated superior performance in ten of them. However, when using CCPOA, combining Iterative map with POA yields better results in terms of convergence and accuracy compared to COPOA and OCPOA.

TABLE III
Benchmark Functions

Benchmark functions	Dimension	Scope	Optimum value
$f_1(x)=\sum_{i=1}^{n} x_i^2$	30	[-100,100]	0
$f_2(x)=\sum_{i=1}^{n}\lvert X_i\rvert+\prod_{i=1}^{n}\lvert X_i\rvert$	30	[-10,10]	0

TABLE III
BENCHMARK FUNCTIONS

Benchmark functions	Dimension	Scope	Optimum value
$f_3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$	30	[-100,100]	0
$f_4(x)=\max_i \lvert x_i \rvert,\quad 1\le i\le n$	30	[-100,100]	0
$f_5(x)=\sum_{i=1}^{n-1}\left[100(x_{i+1}-x_i^2)^2+(x_i-1)^2\right]$	30	[-30,30]	0
$f_6(x)=\sum_{i=1}^{n}([x_i+0.5])^2$	30	[-100,100]	0
$f_7(x)=\sum_{i=1}^{n}ix_i^4+\text{random}[0,1)$	30	[-1.28,1.28]	0
$f_9(x)=\sum_{i=1}^{n}\left(x_i^2-10\cos(2\pi x_i)+10\right)$	30	[-5.12,5.12]	0
$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right)+20+e$	30	[-32,32]	0
$f_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30	[-600,600]	0
$f_{12}(x)=\frac{\pi}{n}\left(10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]+(y_n-1)^2\right)+\sum_{i=1}^{n}u(x_i,10,100,4)$ $y_i=1+\frac{1}{4}(x_i+1)$ $u(x_i,\alpha,k,m)=\begin{cases}K(x_i-\alpha)^m, x_i>\alpha,\\ 0,-\alpha\le x_i\le\alpha,\\ K(-x_i-\alpha)^m, x_i<-\alpha,\end{cases}$	30	[-50,50]	0
$f_{13}(x)=0.1\left(\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}\left((x_i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right]\right)+(x_n-1)\left[1+\sin^2(2\pi x_n)\right]\right)+\sum_{i=1}^{n}u(x_i,5,100,4)$	30	[-50,50]	0

TABLE IV
TEST FUNCTION EXPERIMENT RESULTS

Chaotic map	Bunchmark functions	Statistics	POA	COPOA	OCPOA	CCPOA
Sine map	f_1	ave	3.1233E-212	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	1.5103E-233	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	1.0274E-107	8.0276E-208	5.5560E-207	**2.5882E-208**
		bsf	6.2189E-118	**1.3262E-211**	1.6199E-211	6.3522E-211
		std	2.6626E-213	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	3.5099E-209	**1.2701E-209**	5.0090E-81	1.1557E-74
		bsf	**1.3680E-229**	1.7094E-229	4.0686E-136	4.6773E-166
		std	0.0000E+00	0.0000E+00	7.2288E-160	3.8736E-147
	f_4	ave	5.3747E-104	**1.6285E-104**	3.9346E-08	2.3811E-08
		bsf	7.8035E-115	**5.5424E-115**	6.9777E-11	1.9824E-10
		std	8.3744E-206	7.6831E-207	1.4485E-14	1.3804E-15
	f_5	ave	2.7108E+01	**2.6049E+01**	2.8030E+01	2.7827E+01
		bsf	2.5173E+01	2.4591E+01	**2.3762E+01**	2.5055E+01
		std	1.3416E+00	7.6040E-01	1.9134E+00	1.8100E+00
	f_6	ave	2.2098E+00	**1.2306E+00**	2.0333E+00	2.1784E+00
		bsf	7.3800E-01	**2.4930E-01**	2.5320E-01	4.3703E-07
		std	4.0660E-01	4.2990E-01	1.9532E+00	1.8273E+00

TABLE IV
TEST FUNCTION EXPERIMENT RESULTS

Chaotic map	Bunchmark functions	Statistics	POA	COPOA	OCPOA	CCPOA
Sine map	f_7	ave	**5.2575E-04**	1.2000E-03	4.6000E-03	4.9000E-03
		bsf	**3.3612E-05**	6.3316E-05	3.1000E-03	2.0000E-03
		std	8.9217E-08	3.9430E-07	8.5234E-07	2.2419E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.4448E+02	1.3770E+02
		bsf	**0.0000E+00**	**0.0000E+00**	4.5335E+01	3.6480E+00
		std	0.0000E+00	0.0000E+00	1.3894E+03	2.5398E+03
	f_{10}	ave	3.2567E-15	**8.8818E-16**	2.4551E+00	6.8258E+00
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	2.8048E-30	0.0000E+00	2.2286E+00	1.8097E+01
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	3.5000E-03	9.1000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	2.4966E-05	1.4144E-04
	f_{12}	ave	1.2570E-01	**4.4000E-02**	3.4784E+00	3.4285E+00
		bsf	4.7200E-02	**4.3112E-08**	7.5250E-01	7.4770E-01
		std	4.3000E-03	1.3000E-03	2.9494E+00	4.3323E+00
	f_{13}	ave	2.5567E+00	**2.2793E+00**	3.0244E+00	3.3133E+00
		bsf	1.5412E+00	**1.0236E+00**	1.4319E+00	1.3730E+00
		std	2.4240E-01	3.9200E-01	4.7910E-01	1.5750E+00
Singer map	f_1	ave	1.0358E-211	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	2.4834E-236	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	3.2469E-105	4.0034E-207	1.6520E-207	**1.1794E-207**
		bsf	4.3827E-116	1.3011E-210	1.2502E-210	**8.4501E-211**
		std	2.4464E-208	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	5.3356E-204	**2.5059E-204**	6.3044E-80	7.8377E-83
		bsf	**9.5826E-238**	1.2150E-237	1.3362E-141	1.8886E-202
		std	0.0000E+00	0.0000E+00	1.1243E-157	1.7815E-163
	f_4	ave	1.9763E-105	**1.9496E-105**	2.8643E-08	1.4735E-07
		bsf	**7.7724E-116**	2.2520E-115	3.3972E-10	1.5608E-10
		std	9.8032E-209	9.7784E-209	3.3147E-15	3.1854E-13
	f_5	ave	2.7382E+01	**2.5871E+01**	2.8256E+01	2.9388E+01
		bsf	2.5295E+01	2.3984E+01	2.4760E+01	**2.1432E+01**
		std	1.3551E+00	1.6400E+00	1.0471E+00	1.7544E+02
	f_6	ave	2.3248E+00	1.0738E+00	3.6890E-01	**2.7680E-01**
		bsf	1.2578E+00	2.4300E-01	2.4046E-07	**2.3457E-07**
		std	2.6500E-01	4.7560E-01	3.1910E-01	2.0750E-01
	f_7	ave	**5.9753E-04**	1.1000E-03	5.4000E-03	5.1000E-03
		bsf	**8.0934E-05**	2.7340E-04	2.1000E-03	2.4000E-03
		std	1.2615E-07	3.9844E-07	2.1913E-06	2.0549E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.2831E+02	1.3793E+02
		bsf	**0.0000E+00**	**0.0000E+00**	5.1021E+01	6.6880E-01
		std	0.0000E+00	0.0000E+00	1.5184E+03	1.8033E+03
	f_{10}	ave	3.2567E-15	**8.8818E-16**	6.3034E+00	4.5356E+00
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	2.8048E-30	0.0000E+00	3.1901E+01	1.7412E+01
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	8.2000E-03	3.5000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	5.3146E-05	2.1249E-05
	f_{12}	ave	1.0990E-01	**4.6200E-02**	5.8285E+00	5.0013E+00
		bsf	3.3400E-02	**1.2316E-07**	3.0105E+00	1.0920E-01
		std	2.4000E-03	1.1000E-03	2.3116E+00	6.4747E+00
	f_{13}	ave	2.2693E+00	7.3570E-01	2.8102E+00	**6.7910E-01**
		bsf	1.3794E+00	1.1871E-06	9.1350E-01	**6.0875E-07**
		std	3.0450E-01	3.5860E-01	9.0650E-01	2.0070E-01
Circle map	f_1	ave	2.1188E-210	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	1.7955E-235	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	1.2332E-106	1.4211E-207	**9.7459E-208**	2.7269E-207
		bsf	2.3313E-115	1.0981E-211	**5.6356E-212**	2.6426E-211
		std	3.8151E-211	0.0000E+00	0.0000E+00	0.0000E+00

TABLE IV
TEST FUNCTION EXPERIMENT RESULTS

Chaotic map	Bunchmark functions	Statistics	POA	COPOA	OCPOA	CCPOA
Circle map	f_3	ave	**1.5334E-210**	1.7730E-209	3.6317E-79	5.1496E-78
		bsf	**1.1839E-232**	2.6222E-232	7.5640E-156	1.0544E-197
		std	0.0000E+00	0.0000E+00	3.6423E-156	4.8741E-154
	f_4	ave	**8.4104E-107**	9.0178E-107	2.7183E-08	4.0330E-08
		bsf	5.5179E-115	**4.0956E-115**	1.9719E-10	5.1223E-11
		std	1.3608E-211	1.1569E-211	2.2735E-15	7.1427E-15
	f_5	ave	2.7729E+01	**2.6730E+01**	2.8168E+01	2.8521E+01
		bsf	2.5264E+01	**2.5142E+01**	2.5575E+01	2.6102E+01
		std	1.1828E+00	1.5238E+00	9.3400E-01	6.4940E-01
	f_6	ave	2.3384E+00	**1.2905E+00**	2.8033E+00	1.8651E+00
		bsf	1.2756E+00	2.5200E-01	**2.3750E-01**	2.5510E-01
		std	2.5510E-01	4.0730E-01	1.1203E+00	7.0930E-01
	f_7	ave	**5.8811E-04**	1.2000E-03	4.6000E-03	5.2000E-03
		bsf	1.1596E-04	**1.8723E-05**	2.6000E-03	3.6000E-03
		std	1.2711E-07	6.1350E-07	1.8004E-06	1.6140E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	9.1654E+01	8.3947E+01
		bsf	**0.0000E+00**	**0.0000E+00**	2.7000E+00	1.7569E+01
		std	0.0000E+00	0.0000E+00	1.7290E+03	1.5953E+03
	f_{10}	ave	3.6119E-15	**8.8818E-16**	2.0791E+00	2.8726E+00
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	2.2579E-30	0.0000E+00	1.6756E+00	1.8432E+00
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	4.1000E-03	5.9000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	3.1028E-05	3.7348E-05
	f_{12}	ave	1.1280E-01	**3.9200E-02**	3.3428E+00	2.5825E+00
		bsf	4.0100E-02	**6.7000E-03**	2.1620E-01	5.1200E-02
		std	2.7000E-03	8.2280E-04	3.6331E+00	2.8569E+00
	f_{13}	ave	2.4161E+00	**1.9729E+00**	2.5060E+00	3.8790E+00
		bsf	1.1155E+00	**5.8550E-01**	6.0730E-01	7.4360E-01
		std	3.2220E-01	5.4480E-01	8.5680E-01	7.7751E+00
Cubic map	f_1	ave	7.1219E-215	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	4.1396E-235	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	1.5804E-103	**4.4357E-208**	7.4894E-208	1.6172E-207
		bsf	2.5589E-116	8.6883E-211	**9.5648E-212**	1.3904E-210
		std	7.2394E-205	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	**7.8507E-210**	9.9856E-210	7.4898E-83	2.7475E-76
		bsf	1.5189E-231	**5.9914E-232**	2.7222E-151	6.8378E-165
		std	0.0000E+00	0.0000E+00	1.6268E-163	1.9005E-150
	f_4	ave	**2.8244E-105**	6.5234E-105	3.7935E-08	2.4403E-08
		bsf	5.8591E-116	**2.8524E-116**	8.4846E-11	7.8225E-10
		std	2.1357E-208	1.1523E-207	5.7989E-15	9.0620E-16
	f_5	ave	2.7497E+01	**2.6465E+01**	2.8153E+01	2.8252E+01
		bsf	2.6072E+01	**2.4874E+01**	2.5045E+01	2.5414E+01
		std	8.2360E-01	1.3738E+00	1.1262E+00	8.0350E-01
	f_6	ave	2.2159E+00	**1.1968E+00**	2.1996E+00	2.2871E+00
		bsf	1.2772E+00	**3.9385E-07**	5.0520E-01	2.5340E-01
		std	2.1180E-01	4.5080E-01	1.4286E+00	1.9985E+00
	f_7	ave	**6.1114E-04**	1.2000E-03	5.0000E-03	5.6000E-03
		bsf	**1.8737E-04**	2.3051E-04	2.7000E-03	2.8000E-03
		std	8.8430E-08	2.8013E-07	1.8928E-06	5.2436E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.3695E+02	1.3964E+02
		bsf	**0.0000E+00**	**0.0000E+00**	1.1386E+01	5.1224E+01
		std	0.0000E+00	0.0000E+00	1.5189E+03	1.2976E+03
	f_{10}	ave	3.1382E-15	**8.8818E-16**	7.4683E+00	7.1137E+00
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	8.8818E-16	0.0000E+00	4.9650E+01	1.5523E+01
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	6.6000E-03	4.6000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	1.3426E-04	4.1015E-05

TABLE IV
TEST FUNCTION EXPERIMENT RESULTS

Chaotic map	Bunchmark functions	Statistics	POA	COPOA	OCPOA	CCPOA
Cubic map	f_{12}	ave	1.2970E-01	**4.7700E-02**	3.8028E+00	2.7355E+00
		bsf	5.3700E-02	**6.5000E-03**	2.7200E-02	3.7520E-01
		std	3.8000E-03	3.0000E-03	4.6539E+00	3.1255E+00
	f_{13}	ave	2.5024E+00	**2.1133E+00**	2.8895E+00	3.0955E+00
		bsf	1.5667E+00	**8.0450E-01**	1.9011E+00	1.7971E+00
		std	2.8480E-01	4.9460E-01	4.4470E-01	6.8590E-01
Tent map	f_1	ave	3.8616E-213	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	2.1694E-236	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	9.5622E-108	1.5572E-207	**1.2163E-207**	3.0465E-207
		bsf	4.3111E-118	**1.1794E-211**	2.5883E-211	1.2790E-210
		std	2.1822E-213	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	2.5454E-209	**1.4036E-209**	1.1580E-75	3.6904E-80
		bsf	**1.1015E-230**	8.9898E-230	2.2634E-134	5.0631E-219
		std	0.0000E+00	0.0000E+00	3.7193E-149	3.9496E-158
	f_4	ave	**1.4202E-105**	3.3400E-105	2.7743E-08	1.1935E-07
		bsf	**2.4274E-116**	3.7405E-116	2.9041E-10	1.3983E-09
		std	5.6121E-209	3.1670E-208	1.4697E-15	7.7070E-14
	f_5	ave	2.7642E+01	**2.6180E+01**	2.8364E+01	2.8705E+01
		bsf	2.5273E+01	2.4054E+01	2.5004E+01	**2.0784E+01**
		std	1.0260E+00	1.5523E+00	7.3000E-01	8.5401E+01
	f_6	ave	2.1247E+00	**9.6020E-01**	3.3777E+00	5.6361E+00
		bsf	9.0640E-01	**3.2681E-07**	5.0580E-01	1.5008E+00
		std	2.8610E-01	4.9280E-01	8.4520E-01	9.9110E-01
	f_7	ave	**4.9125E-04**	1.2000E-03	4.9000E-03	5.4000E-03
		bsf	**1.2078E-04**	1.9874E-04	1.6000E-03	2.8000E-03
		std	7.9659E-08	5.2160E-07	2.2749E-06	1.9983E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.2312E+02	7.0206E+01
		bsf	**0.0000E+00**	**0.0000E+00**	1.3853E+00	5.6427E+00
		std	0.0000E+00	0.0000E+00	2.5632E+03	1.8510E+03
	f_{10}	ave	3.8488E-15	**8.8818E-16**	1.9405E+01	1.9962E+01
		bsf	**8.8818E-16**	**8.8818E-16**	3.2420E+00	1.9959E+01
		std	1.7530E-30	0.0000E+00	9.0081E+00	3.2280E-06
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	4.1000E-03	5.0000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	4.5431E-05	3.8003E-05
	f_{12}	ave	1.2250E-01	**4.7300E-02**	4.6015E+00	4.4105E+00
		bsf	4.8800E-02	**1.3300E-02**	2.0515E+00	4.2440E-01
		std	2.7000E-03	1.7000E-03	2.5599E+00	5.5303E+00
	f_{13}	ave	2.3938E+00	**1.2607E+00**	2.7852E+00	1.3207E+00
		bsf	1.2093E+00	3.4050E-01	**2.6140E-01**	3.5940E-01
		std	3.4430E-01	3.4960E-01	9.1990E-01	3.5150E-01
Piecewise map	f_1	ave	3.7759E-211	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	5.6416E-232	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	2.4367E-106	7.8724E-208	2.5654E-207	**3.9989E-208**
		bsf	7.5836E-118	1.8086E-211	2.9044E-211	**1.9931E-212**
		std	8.2493E-211	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	**2.1753E-213**	4.2532E-213	7.5845E-77	2.7282E-84
		bsf	**2.1042E-233**	4.3199E-233	2.1844E-143	1.0697E-166
		std	0.0000E+00	0.0000E+00	1.6682E-151	2.1155E-166
	f_4	ave	9.4547E-105	**5.2843E-105**	3.0747E-08	3.6672E-08
		bsf	3.8327E-117	**2.1337E-117**	3.2028E-10	7.0713E-11
		std	2.3602E-207	6.4746E-208	1.8286E-15	9.3317E-15
	f_5	ave	2.7208E+01	**2.6367E+01**	2.8178E+01	2.8376E+01
		bsf	2.5278E+01	**2.4548E+01**	2.5538E+01	2.6104E+01
		std	1.5446E+00	1.5392E+00	9.7490E-01	5.3600E-01
	f_6	ave	2.1143E+00	**9.0730E-01**	2.2259E+00	1.8525E+00
		bsf	5.8300E-01	**3.0335E-07**	2.5210E-01	4.8346E-07
		std	4.0800E-01	3.7640E-01	1.5772E+00	1.5916E+00

TABLE IV
TEST FUNCTION EXPERIMENT RESULTS

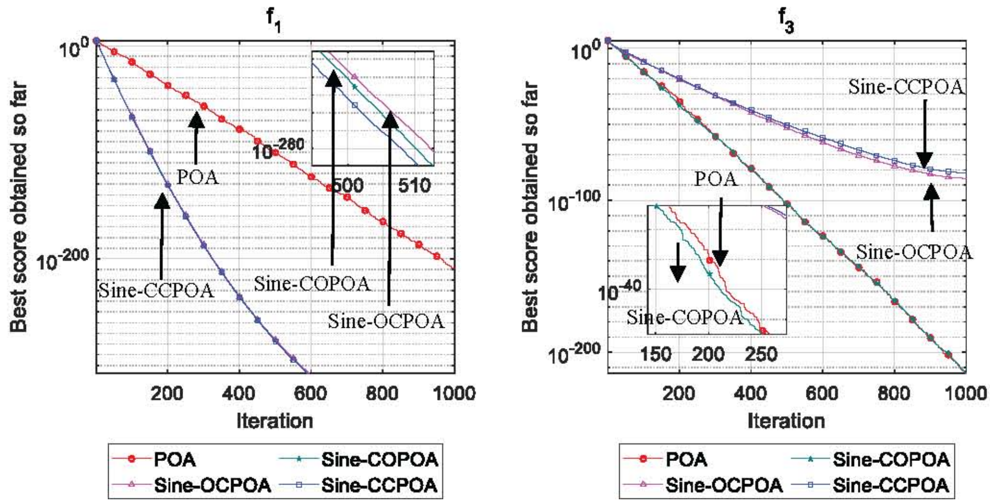
Chaotic map	Bunchmark functions	Statistics	POA	COPOA	OCPOA	CCPOA
Piecewise map	f_7	ave	**5.7392E-04**	1.2000E-03	5.2000E-03	5.1000E-03
		bsf	**1.0887E-04**	1.4531E-04	2.3000E-03	3.3000E-03
		std	6.5479E-08	6.9551E-07	3.7606E-06	1.8216E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.1961E+02	1.3449E+02
		bsf	**0.0000E+00**	**0.0000E+00**	3.7016E+01	2.3906E+01
		std	0.0000E+00	0.0000E+00	1.9102E+03	2.2142E+03
	f_{10}	ave	3.0198E-15	**8.8818E-16**	3.7555E+00	3.2874E+00
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	3.0292E-30	0.0000E+00	4.0431E+00	4.2428E+00
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	4.2000E-03	6.4000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	2.8373E-05	7.7860E-05
	f_{12}	ave	1.1270E-01	**4.2500E-02**	2.2662E+00	3.2698E+00
		bsf	3.4200E-02	**3.1474E-08**	5.1350E-01	5.4990E-01
		std	2.5000E-03	1.2000E-03	2.3029E+00	3.0508E+00
	f_{13}	ave	2.4746E+00	**1.9107E+00**	3.1388E+00	2.5955E+00
		bsf	9.8360E-01	**6.4290E-01**	1.9965E+00	1.3357E+00
		std	3.5560E-01	6.1930E-01	7.6690E-01	6.5120E-01
Sinusoidal map	f_1	ave	3.7189E-216	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	3.6227E-237	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	3.6023E-109	1.1440E-207	**5.5359E-208**	9.5881E-208
		bsf	1.0338E-119	1.6032E-210	**1.8379E-211**	2.8049E-210
		std	1.7682E-216	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	1.8555E-207	**2.4219E-208**	1.9086E-73	1.4635E-99
		bsf	1.2348E-231	**0.0000E+00**	7.1650E-158	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	1.0564E-144	5.4060E-197
	f_4	ave	**1.5490E-104**	2.2152E-104	5.6067E-08	3.1368E-08
		bsf	1.1039E-115	**9.4999E-116**	3.3469E-10	4.9772E-11
		std	5.7747E-207	1.1793E-206	2.4237E-14	4.3162E-15
	f_5	ave	2.7542E+01	2.6430E+01	1.3310E+01	**1.1068E+01**
		bsf	2.5194E+01	2.4762E+01	**8.0728E-04**	8.3750E-04
		std	1.0476E+00	1.3123E+00	1.8265E+02	1.8444E+02
	f_6	ave	2.2962E+00	**4.6180E-01**	2.8106E+00	1.3900E+00
		bsf	1.2598E+00	**2.4069E-07**	4.3941E-07	3.8364E-07
		std	2.9030E-01	3.6750E-01	2.3279E+00	1.8595E+00
	f_7	ave	**4.8894E-04**	1.3000E-03	5.1000E-03	4.1000E-03
		bsf	**2.3602E-05**	1.3516E-04	6.4591E-04	5.7144E-04
		std	1.2137E-07	6.3066E-07	1.7410E-06	3.5882E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.1014E+02	5.6396E+01
		bsf	**0.0000E+00**	**0.0000E+00**	3.7308E+00	4.6201E+00
		std	0.0000E+00	0.0000E+00	2.6593E+03	1.8008E+03
	f_{10}	ave	3.2567E-15	**8.8818E-16**	2.2639E+00	3.2741E+00
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	2.8048E-30	0.0000E+00	3.9056E+00	9.2581E+00
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	8.2000E-03	5.1000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	6.6836E-05	4.3652E-05
	f_{12}	ave	1.1450E-01	**3.3300E-02**	3.4820E+00	1.9084E+00
		bsf	4.4100E-02	**3.0471E-08**	2.6090E-01	1.3400E-02
		std	1.6000E-03	8.6552E-04	2.1774E+00	2.4738E+00
	f_{13}	ave	2.5144E+00	1.8801E+00	**2.4990E-01**	3.1010E-01
		bsf	1.4810E+00	6.5590E-01	7.1130E-07	**3.9923E-07**
		std	2.5300E-01	4.5890E-01	6.2800E-02	8.6200E-02
Logistic map	f_1	ave	8.1798E-210	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	6.6684E-235	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	4.2908E-107	4.9163E-207	**5.2053E-208**	7.5875E-208
		bsf	2.1836E-115	4.0668E-211	1.4896E-210	**2.8781E-211**
		std	2.4516E-212	0.0000E+00	0.0000E+00	0.0000E+00

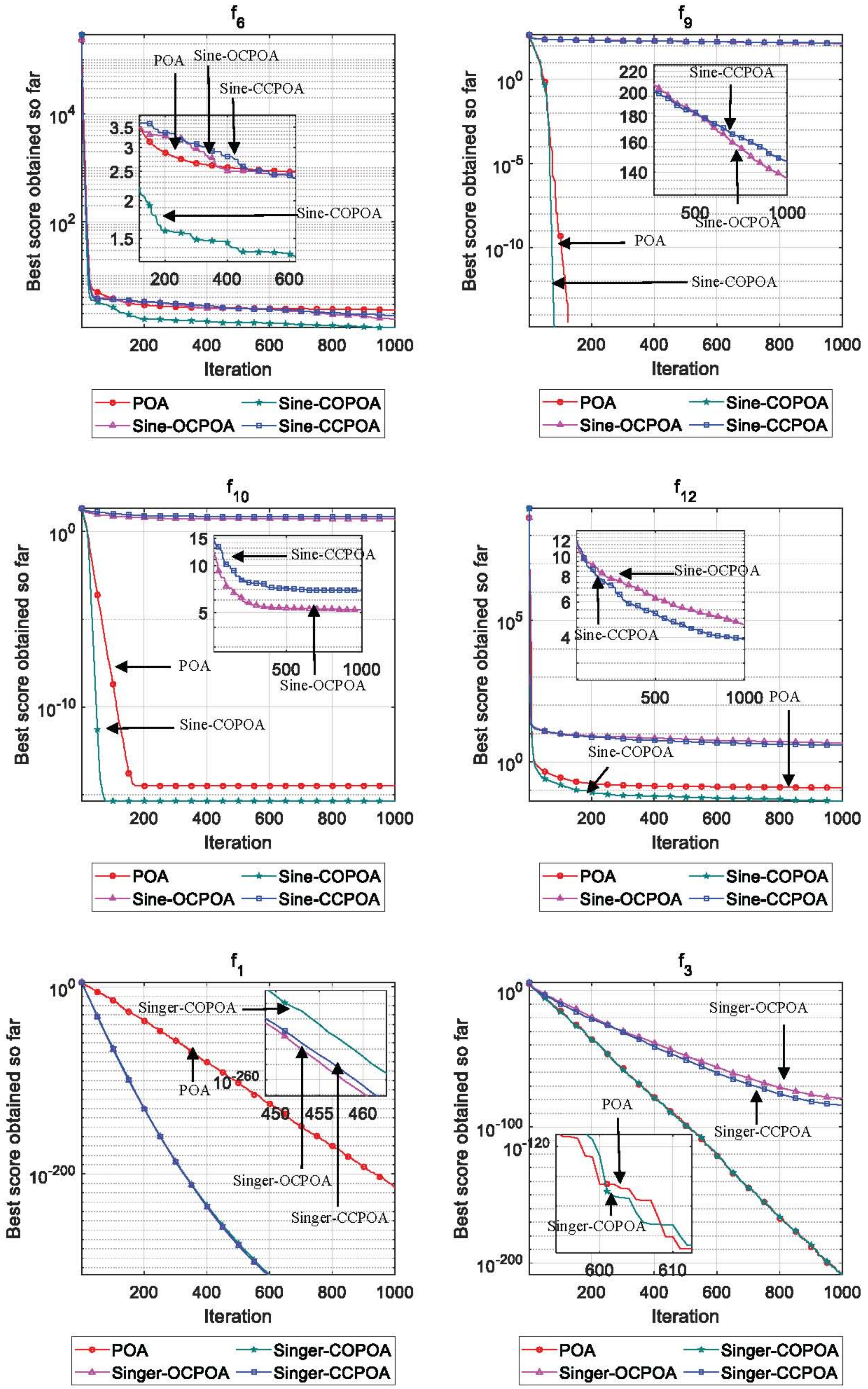
TABLE IV
TEST FUNCTION EXPERIMENT RESULTS

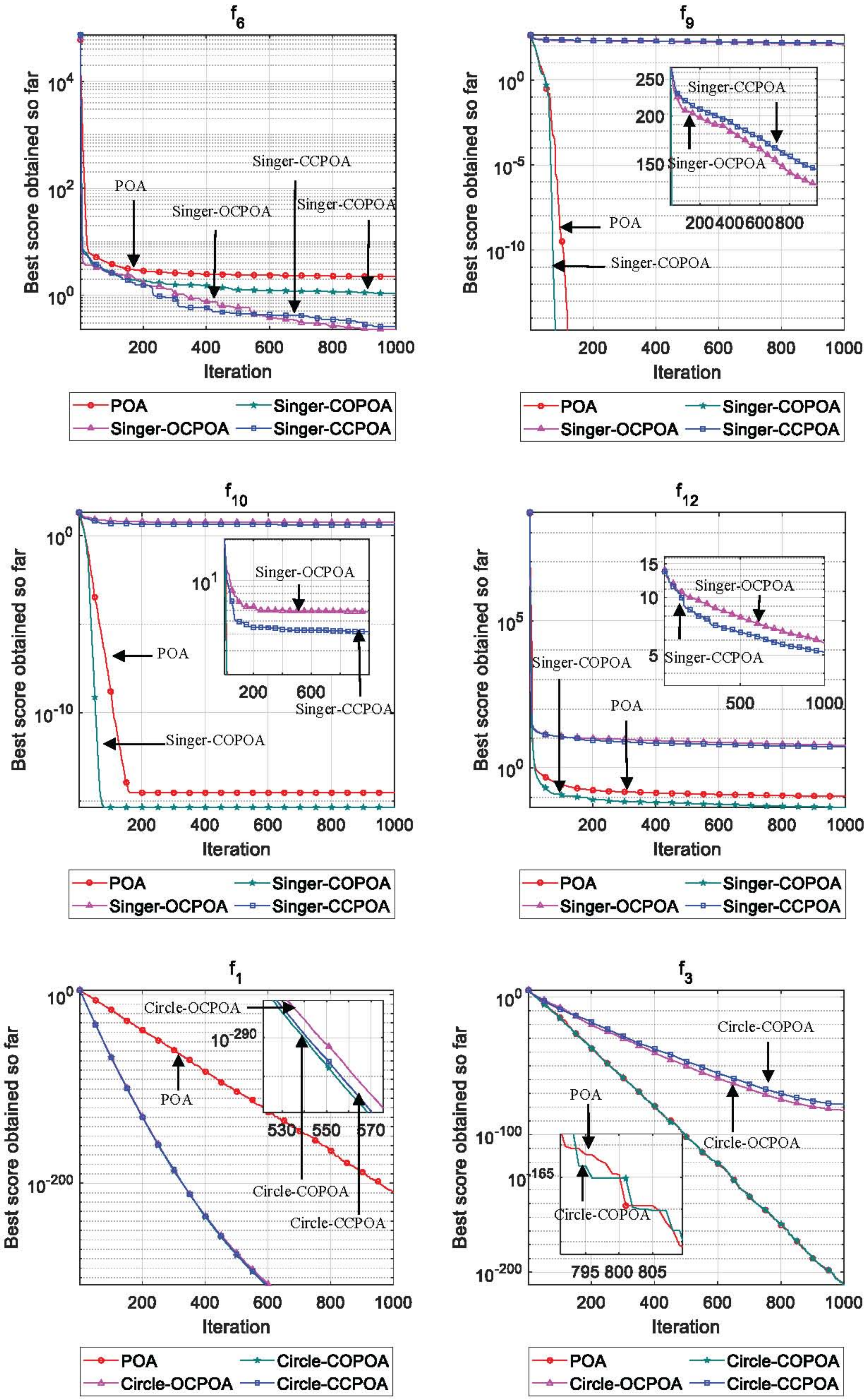
Chaotic map	Bunchmark functions	Statistics	POA	COPOA	OCPOA	CCPOA
Logistic map	f_3	ave	1.1100E-207	**4.2854E-208**	5.0954E-80	1.6731E-74
		bsf	5.1620E-232	**3.3933E-232**	3.6750E-143	2.2477E-148
		std	0.0000E+00	0.0000E+00	7.5257E-158	8.0859E-147
	f_4	ave	**1.6128E-106**	2.9126E-106	3.2748E-08	1.1141E-07
		bsf	**2.7238E-116**	5.6668E-116	1.9933E-10	3.4191E-10
		std	5.8679E-211	2.0103E-210	3.7478E-15	4.9401E-14
	f_5	ave	2.7421E+01	**2.6669E+01**	2.8506E+01	2.8249E+01
		bsf	2.6005E+01	2.5204E+01	2.6123E+01	**2.5125E+01**
		std	8.5070E-01	9.5900E-01	6.0080E-01	1.2136E+00
	f_6	ave	2.4157E+00	**1.0757E+00**	2.2513E+00	2.3420E+00
		bsf	1.5196E+00	**3.0484E-07**	5.0390E-01	5.8754E-07
		std	1.7420E-01	4.0120E-01	1.6059E+00	1.8276E+00
	f_7	ave	**6.4674E-04**	1.3000E-03	4.9000E-03	5.3000E-03
		bsf	**1.3044E-04**	4.1238E-04	2.1000E-03	3.1000E-03
		std	1.7912E-07	3.6511E-07	1.6140E-06	2.9842E-06
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.5010E+02	1.6770E+02
		bsf	**0.0000E+00**	**0.0000E+00**	8.0067E+01	1.6053E+01
		std	0.0000E+00	0.0000E+00	1.1127E+03	2.1084E+03
	f_{10}	ave	3.3751E-15	**8.8818E-16**	6.8767E+00	9.5703E+00
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	2.3393E+00
		std	2.6506E-30	0.0000E+00	3.8922E+01	1.5913E+01
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	8.6000E-03	6.1000E-03
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	5.3875E-05	4.2543E-05
	f_{12}	ave	1.1900E-01	**4.6700E-02**	3.8660E+00	3.7331E+00
		bsf	4.7100E-02	**5.5723E-08**	1.1413E+00	6.6110E-01
		std	2.2000E-03	1.3000E-03	2.9459E+00	3.8694E+00
	f_{13}	ave	2.3866E+00	**2.0053E+00**	2.9397E+00	3.1067E+00
		bsf	1.3581E+00	**1.0745E+00**	1.5444E+00	1.1011E+00
		std	2.4280E-01	3.4830E-01	7.5870E-01	1.1636E+00
Chebyshev map	f_1	ave	1.9191E-212	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	5.4970E-235	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	3.6072E-107	**6.4460E-208**	1.0560E-207	1.9843E-207
		bsf	1.9657E-116	**1.9211E-211**	2.8306E-211	4.0607E-211
		std	1.4676E-212	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	1.1852E-207	**5.3159E-208**	1.9379E-81	1.0565E-85
		bsf	2.1337E-230	**0.0000E+00**	1.5767E-130	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	1.0867E-160	3.2370E-169
	f_4	ave	4.3171E-104	**3.7547E-104**	3.9880E-07	1.9069E-04
		bsf	1.2995E-113	**7.1604E-114**	5.5124E-12	9.9210E-06
		std	5.3895E-206	4.0811E-206	3.9735E-12	2.0870E-08
	f_5	ave	2.7434E+01	2.1995E+01	6.3400E-02	**4.4367E-05**
		bsf	2.5277E+01	1.0300E+00	4.9468E-06	**3.7458E-09**
		std	1.5647E+00	7.4202E+01	3.4800E-02	1.9975E-08
	f_6	ave	2.2084E+00	6.2260E-01	3.9767E-07	**3.0114E-07**
		bsf	1.2687E+00	2.3044E-07	2.0934E-07	**2.2169E-09**
		std	3.0030E-01	4.3790E-01	1.2208E-14	3.2917E-14
	f_7	ave	7.7462E-04	1.1000E-03	7.3162E-04	**2.5344E-04**
		bsf	1.4598E-04	2.5981E-04	**7.2592E-06**	8.4789E-06
		std	3.8248E-07	4.3897E-07	2.2251E-07	4.8535E-08
	f_9	ave	**0.0000E+00**	**0.0000E+00**	2.3000E-03	4.7070E-07
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	4.3295E-05	2.6720E-12
	f_{10}	ave	3.3751E-15	**8.8818E-16**	6.9000E-03	2.1625E-04
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	2.6506E-30	0.0000E+00	1.2963E-04	2.2456E-07
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	3.5000E-03	4.8416E-06
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	2.2453E-05	1.2647E-10

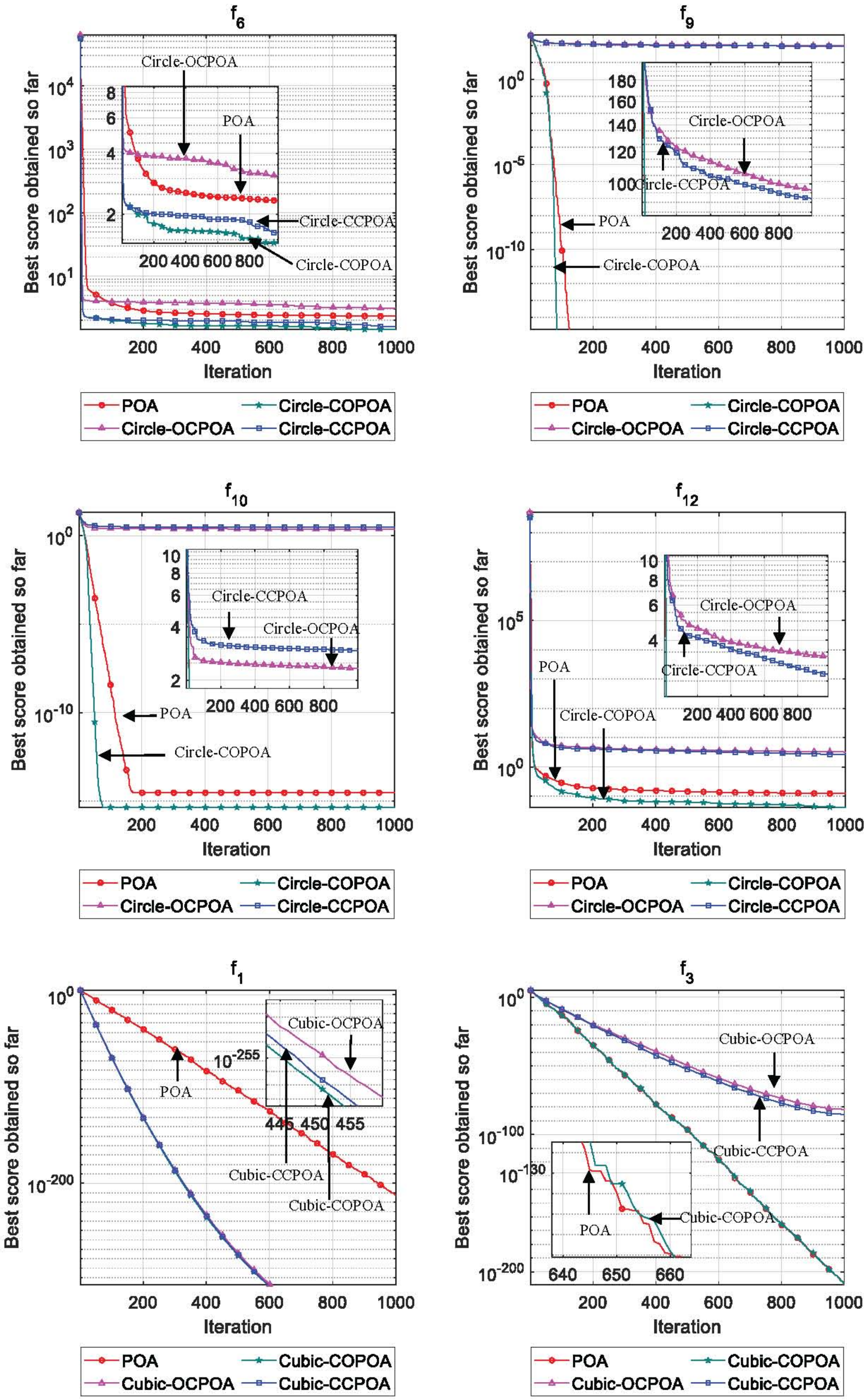
TABLE IV
TEST FUNCTION EXPERIMENT RESULTS

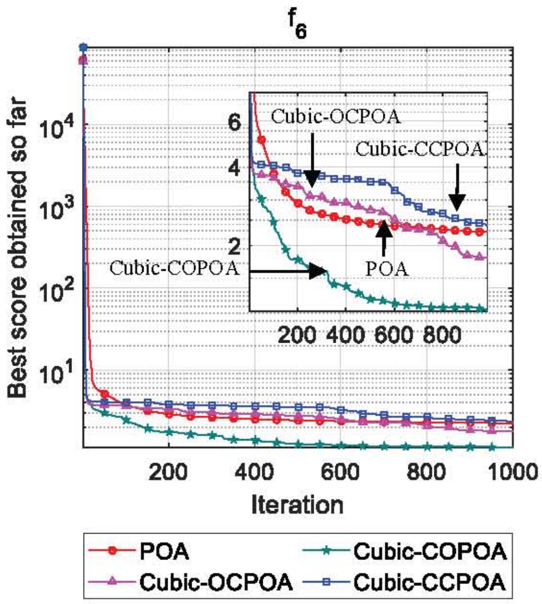
Chaotic map	Bunchmark functions	Statistics	POA	COPOA	OCPOA	CCPOA
Chebyshev map	f_{12}	ave	1.1170E-01	1.2000E-02	1.6900E-05	**1.7690E-08**
		bsf	4.3500E-02	4.6120E-08	1.9451E-08	**1.9990E-11**
		std	2.4000E-03	2.4566E-04	1.3872E-09	5.0174E-16
	f_{13}	ave	2.4683E+00	9.5600E-02	1.5000E-03	**2.4218E-07**
		bsf	1.1498E+00	4.1531E-07	5.5050E-07	**5.3203E-10**
		std	3.1990E-01	3.2100E-02	1.0702E-05	2.5875E-13
Iterative map	f_1	ave	8.8422E-209	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		bsf	5.1049E-233	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	f_2	ave	3.1248E-107	1.6660E-206	1.8966E-207	**9.4411E-208**
		bsf	3.2053E-118	1.8898E-211	5.0240E-211	**1.0491E-212**
		std	7.4033E-213	0.0000E+00	0.0000E+00	0.0000E+00
	f_3	ave	1.9575E-212	**1.5437E-212**	4.4404E-81	7.8598E-85
		bsf	7.8103E-236	**0.0000E+00**	1.8015E-143	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	5.7163E-160	1.7915E-167
	f_4	ave	**1.4162E-106**	2.8421E-106	1.8967E-08	2.4425E-04
		bsf	**5.5812E-113**	6.3287E-113	1.2862E-11	1.0898E-07
		std	3.0411E-211	1.1887E-210	5.0293E-15	8.7762E-08
	f_5	ave	2.7093E+01	1.9277E+01	3.5100E-02	**3.7074E-05**
		bsf	2.4672E+01	4.9490E-01	4.5437E-04	**3.2912E-09**
		std	1.0525E+00	8.3821E+01	5.1000E-03	6.1227E-09
	f_6	ave	2.2205E+00	5.4710E-01	4.7246E-05	**2.6795E-07**
		bsf	1.4346E+00	1.8791E-07	2.1069E-07	**9.5155E-09**
		std	1.8290E-01	4.1980E-01	5.7431E-08	2.7522E-14
	f_7	ave	5.0527E-04	1.2000E-03	5.9301E-04	**2.1601E-04**
		bsf	5.3840E-05	5.4490E-05	1.8639E-04	**1.6285E-06**
		std	1.4325E-07	7.5404E-07	1.1594E-07	2.5495E-08
	f_9	ave	**0.0000E+00**	**0.0000E+00**	1.6000E-03	4.5506E-07
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	2.3551E-05	2.9769E-12
	f_{10}	ave	3.3751E-15	**8.8818E-16**	6.2000E-03	4.2485E-05
		bsf	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**	**8.8818E-16**
		std	2.6506E-30	0.0000E+00	1.6973E-04	2.1000E-03
	f_{11}	ave	**0.0000E+00**	**0.0000E+00**	5.2000E-03	4.6039E-06
		bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
		std	0.0000E+00	0.0000E+00	2.7131E-05	1.2469E-10
	f_{12}	ave	1.1950E-01	2.0600E-02	4.6166E-06	**1.3624E-08**
		bsf	2.9200E-02	3.8810E-08	2.7915E-08	**3.8806E-11**
		std	4.9000E-03	6.4124E-04	6.8306E-11	3.6098E-16
	f_{13}	ave	2.4630E+00	1.9080E-01	7.5741E-04	**4.8351E-07**
		bsf	1.5619E+00	3.0868E-07	2.6702E-07	**4.2302E-14**
		std	2.5400E-01	1.4290E-01	4.4830E-06	5.6539E-13



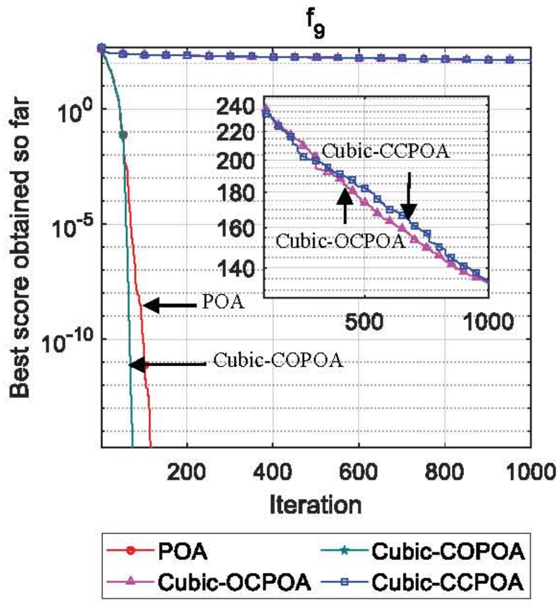

f6
Best score obtained so far
Iteration
POA
Sine-OCPOA
Sine-CCPOA
Sine-COPOA
f9
f10
f12
f1
Singer-COPOA
Singer-OCPOA
Singer-CCPOA
f3


f6
Best score obtained so far
Iteration
POA
Singer-OCPOA
Singer-CCPOA
Singer-COPOA
f9
250
200
150
200400600800
Singer-CCPOA
Singer-OCPOA
POA
Singer-COPOA
f10
Singer-OCPOA
Singer-CCPOA
POA
Singer-COPOA
f12
15
10
5
Singer-OCPOA
Singer-CCPOA
Singer-COPOA
POA
f1
Circle-OCPOA
POA
Circle-COPOA
Circle-CCPOA
f3
Circle-COPOA
Circle-OCPOA
POA
Circle-COPOA
795 800 805
Circle-CCPOA

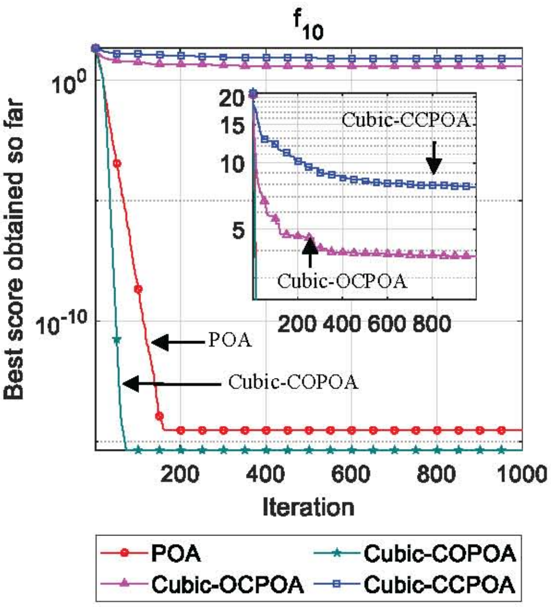

f6
Best score obtained so far
Iteration
Circle-OCPOA
POA
Circle-CCPOA
Circle-COPOA
f9
Circle-OCPOA
Circle-CCPOA
POA
Circle-COPOA
f10
Circle-CCPOA
Circle-OCPOA
POA
Circle-COPOA
f12
Circle-OCPOA
Circle-CCPOA
POA
Circle-COPOA
POA
Circle-COPOA
Circle-OCPOA
Circle-CCPOA
f1
Cubic-OCPOA
POA
Cubic-CCPOA
Cubic-COPOA
f3
Cubic-OCPOA
Cubic-CCPOA
Cubic-COPOA
POA
Cubic-COPOA
Cubic-OCPOA
Cubic-CCPOA



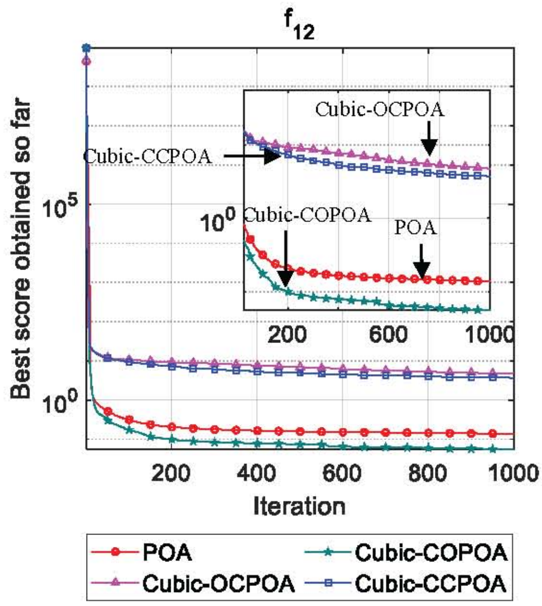
f6
Best score obtained so far
Iteration
Cubic-OCPOA
Cubic-CCPOA
Cubic-COPOA
POA
POA
Cubic-COPOA
Cubic-OCPOA
Cubic-CCPOA



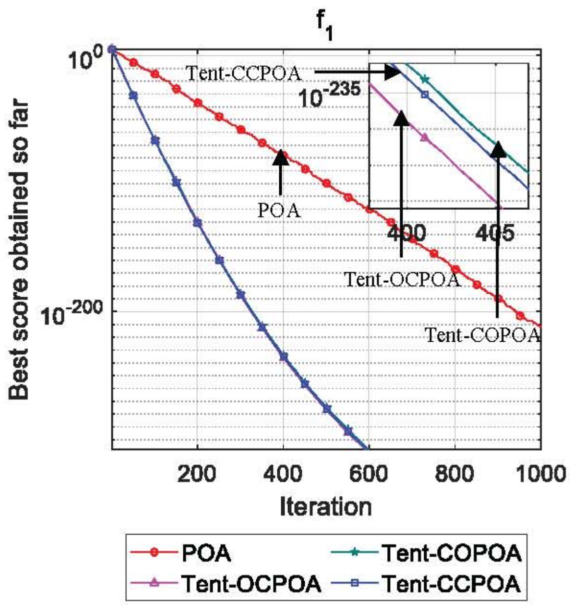
f9
Best score obtained so far
Iteration
Cubic-CCPOA
Cubic-OCPOA
POA
Cubic-COPOA
POA
Cubic-COPOA
Cubic-OCPOA
Cubic-CCPOA



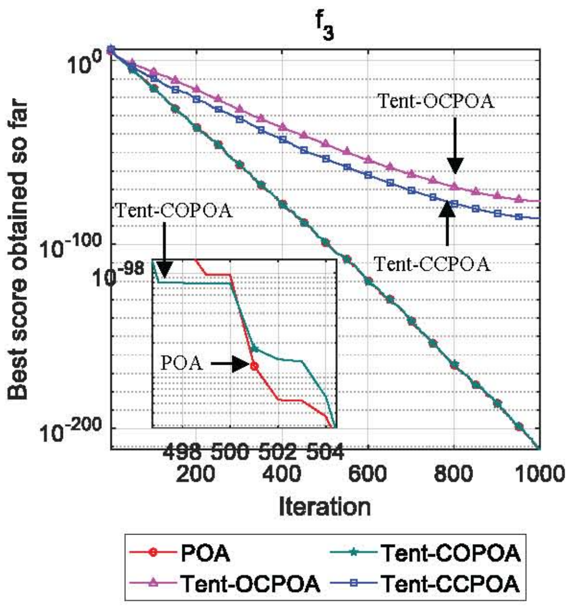
f10
Best score obtained so far
Iteration
Cubic-CCPOA
Cubic-OCPOA
POA
Cubic-COPOA
POA
Cubic-COPOA
Cubic-OCPOA
Cubic-CCPOA



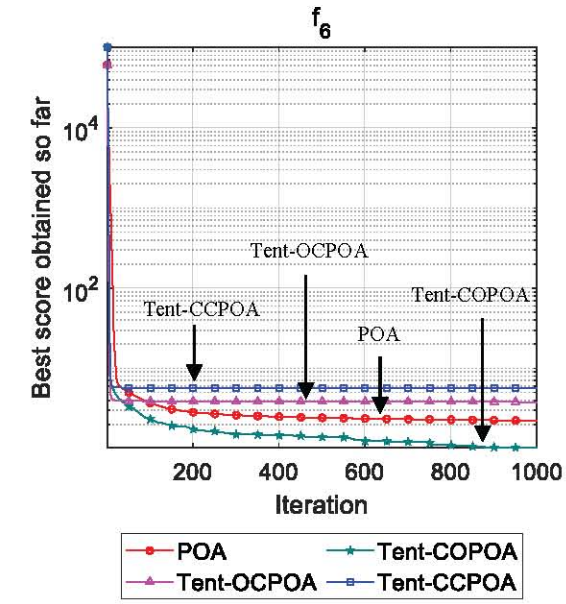
f12
Best score obtained so far
Iteration
Cubic-OCPOA
Cubic-CCPOA
Cubic-COPOA
POA
POA
Cubic-COPOA
Cubic-OCPOA
Cubic-CCPOA

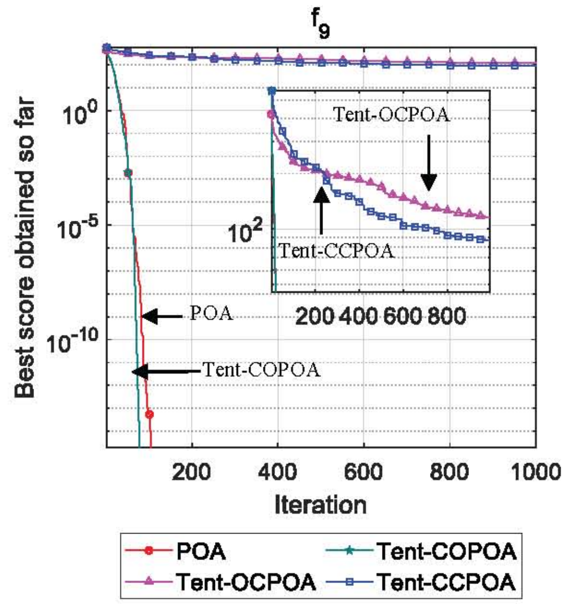


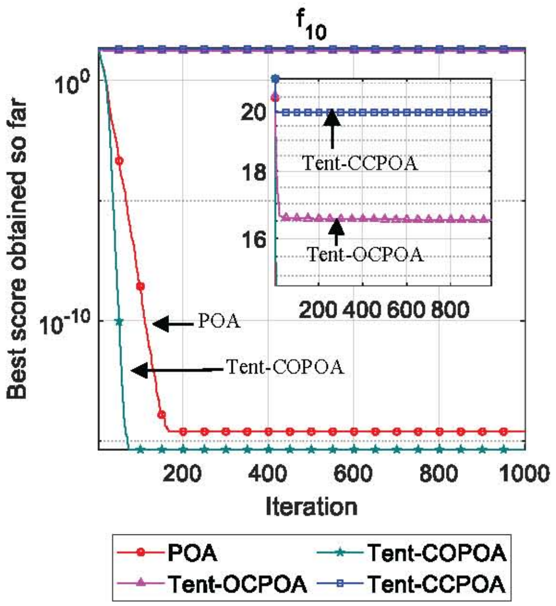
f1
Best score obtained so far
Iteration
Tent-CCPOA
POA
Tent-OCPOA
Tent-COPOA
POA
Tent-COPOA
Tent-OCPOA
Tent-CCPOA

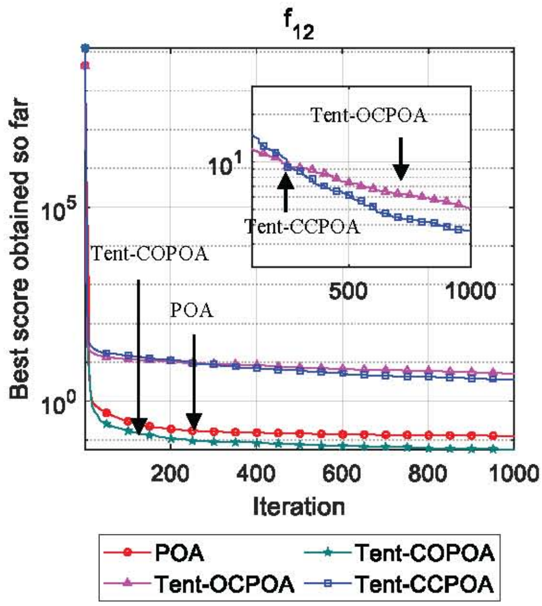


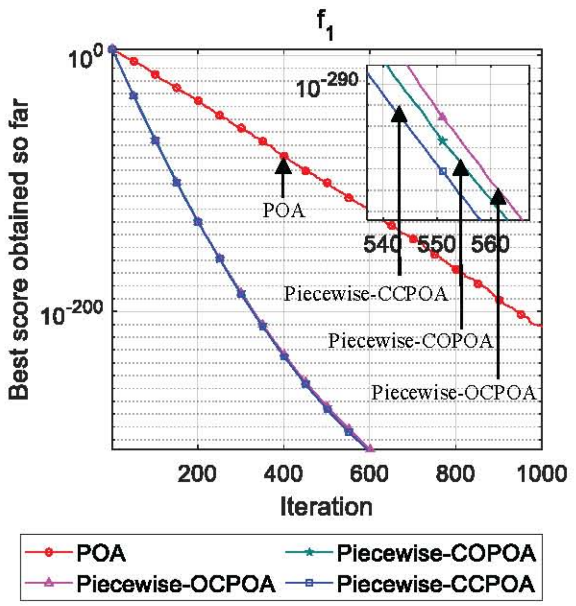
f3
Best score obtained so far
Iteration
Tent-OCPOA
Tent-COPOA
Tent-CCPOA
POA
POA
Tent-COPOA
Tent-OCPOA
Tent-CCPOA

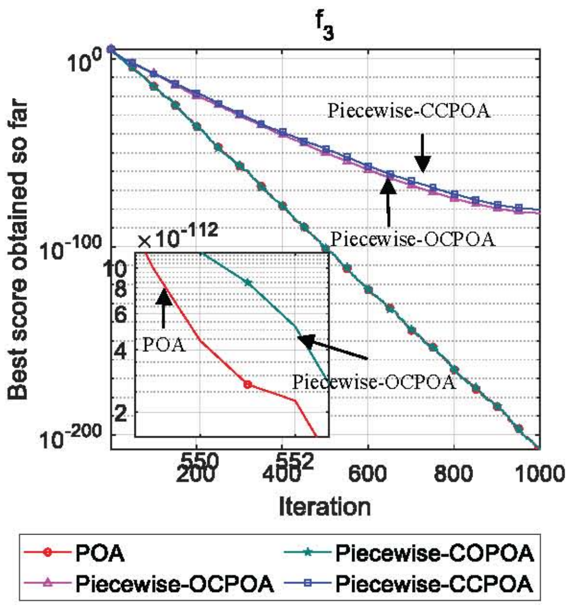

f6
Best score obtained so far
Iteration
Tent-OCPOA
Tent-CCPOA
Tent-COPOA
POA
POA
Tent-COPOA
Tent-OCPOA
Tent-CCPOA

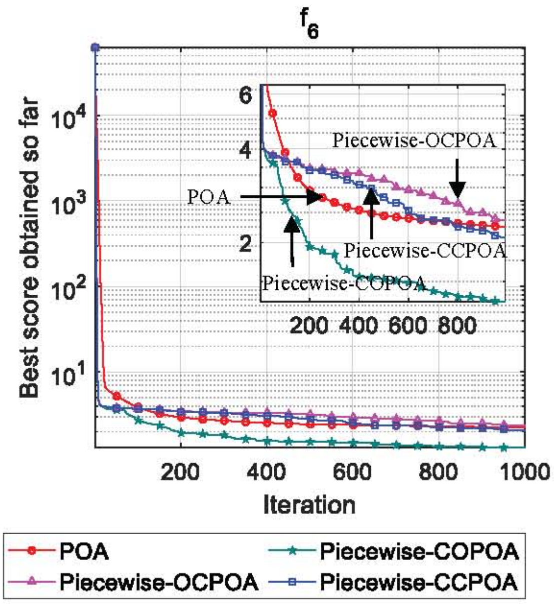

f9
Best score obtained so far
Iteration
Tent-OCPOA
Tent-CCPOA
POA
Tent-COPOA
POA
Tent-COPOA
Tent-OCPOA
Tent-CCPOA

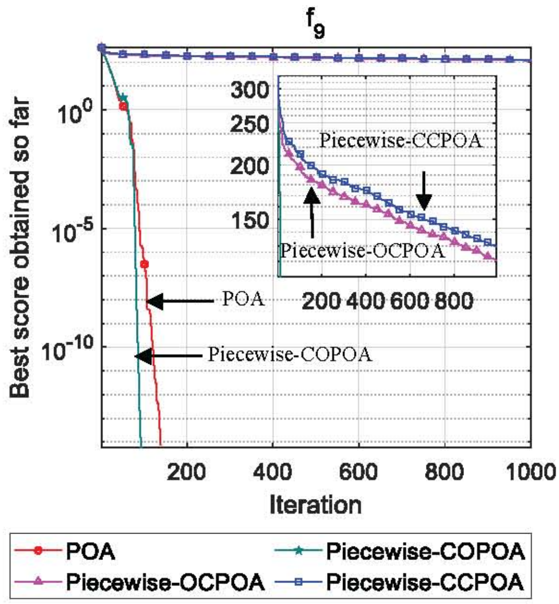

f10
Best score obtained so far
Iteration
Tent-CCPOA
Tent-OCPOA
POA
Tent-COPOA
POA
Tent-COPOA
Tent-OCPOA
Tent-CCPOA

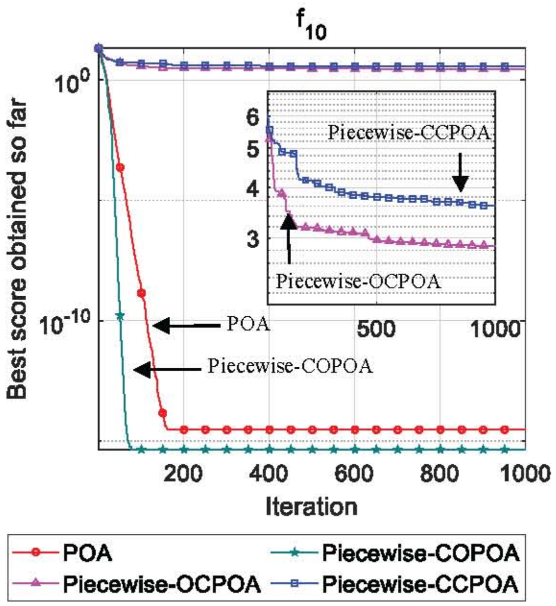

f12
Best score obtained so far
Iteration
Tent-OCPOA
Tent-CCPOA
Tent-COPOA
POA
POA
Tent-COPOA
Tent-OCPOA
Tent-CCPOA

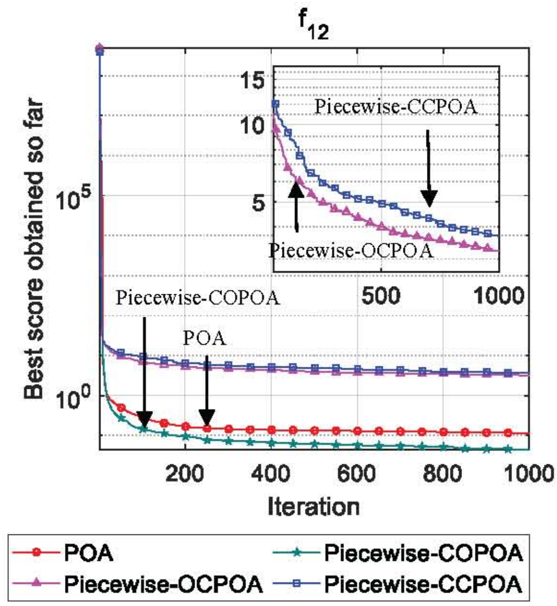

f1
Best score obtained so far
Iteration
POA
Piecewise-CCPOA
Piecewise-COPOA
Piecewise-OCPOA
POA
Piecewise-COPOA
Piecewise-OCPOA
Piecewise-CCPOA

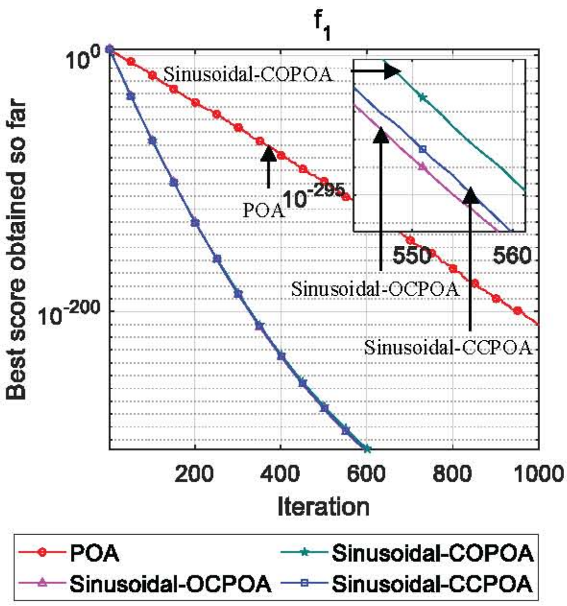

f3
Best score obtained so far
Iteration
Piecewise-CCPOA
Piecewise-OCPOA
POA
Piecewise-OCPOA
POA
Piecewise-COPOA
Piecewise-OCPOA
Piecewise-CCPOA

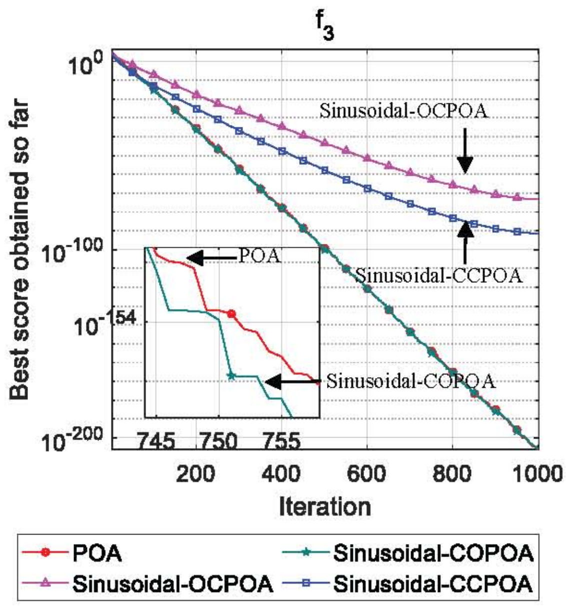

f6
Best score obtained so far
Iteration
Piecewise-OCPOA
POA
Piecewise-CCPOA
Piecewise-COPOA
POA
Piecewise-COPOA
Piecewise-OCPOA
Piecewise-CCPOA

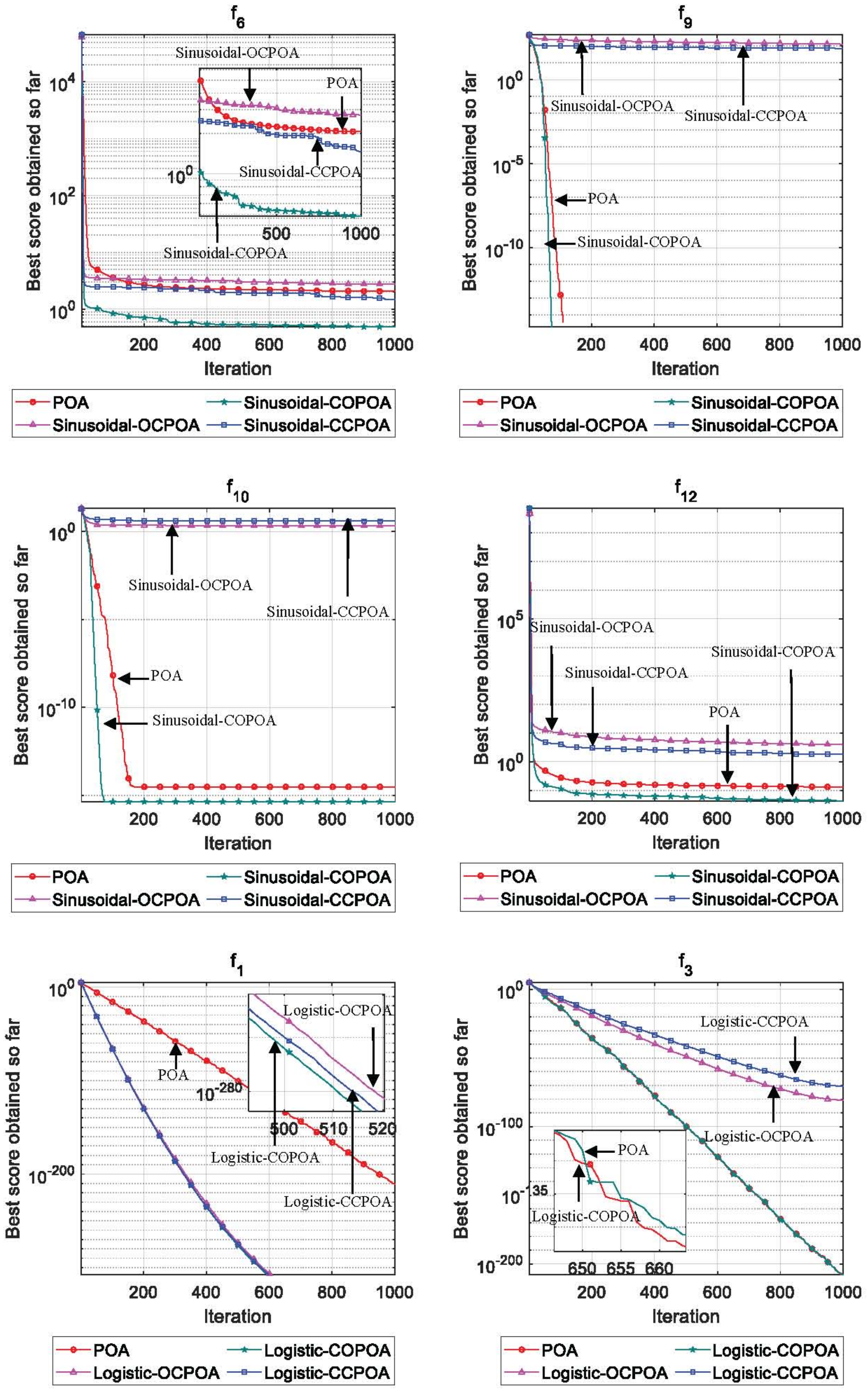

f9
Best score obtained so far
Iteration
Piecewise-CCPOA
Piecewise-OCPOA
POA
Piecewise-COPOA
POA
Piecewise-COPOA
Piecewise-OCPOA
Piecewise-CCPOA


f10
Best score obtained so far
Iteration
Piecewise-CCPOA
Piecewise-OCPOA
POA
Piecewise-COPOA
POA
Piecewise-COPOA
Piecewise-OCPOA
Piecewise-CCPOA

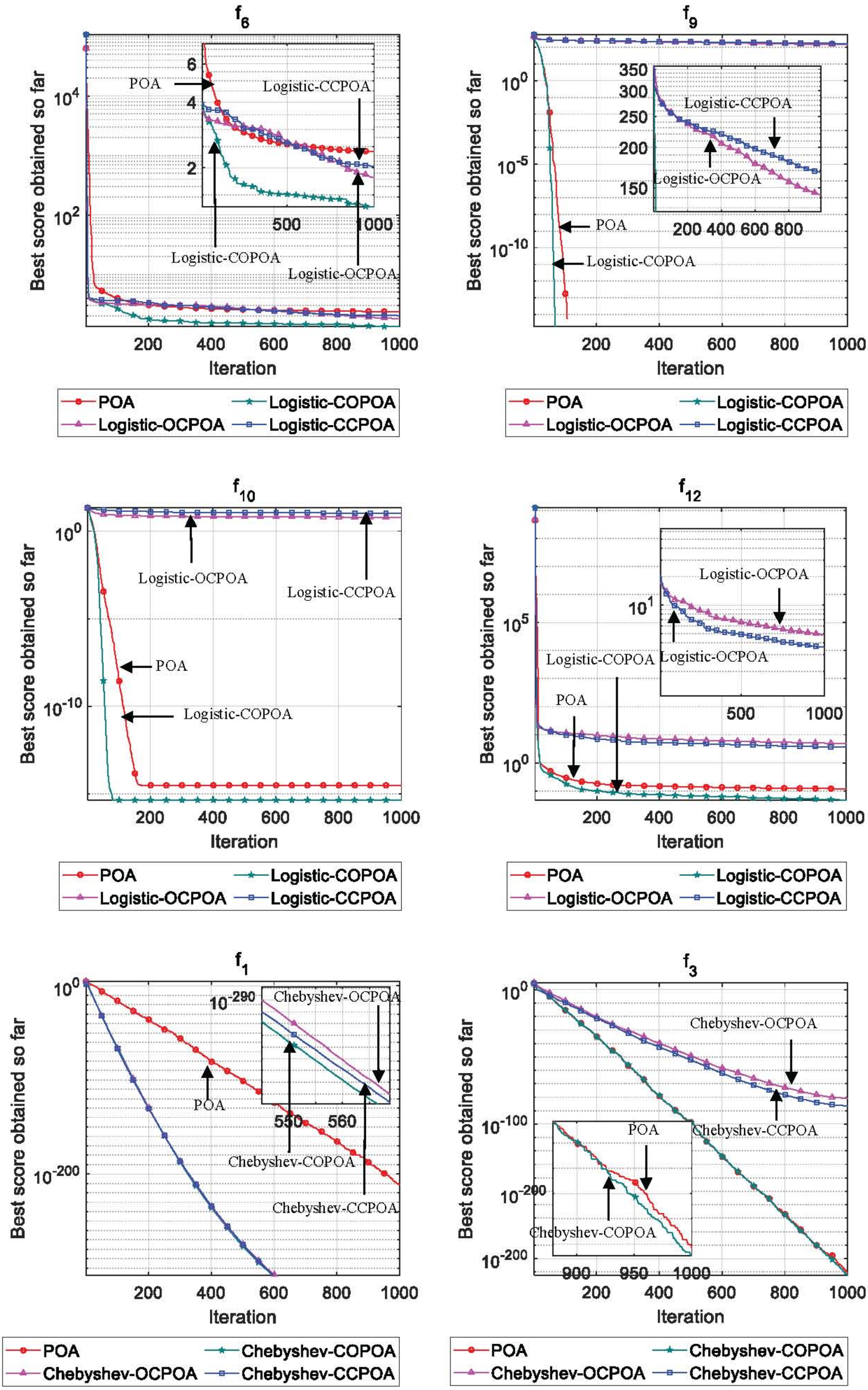

f12
Best score obtained so far
Iteration
Piecewise-CCPOA
Piecewise-OCPOA
Piecewise-COPOA
POA
POA
Piecewise-COPOA
Piecewise-OCPOA
Piecewise-CCPOA

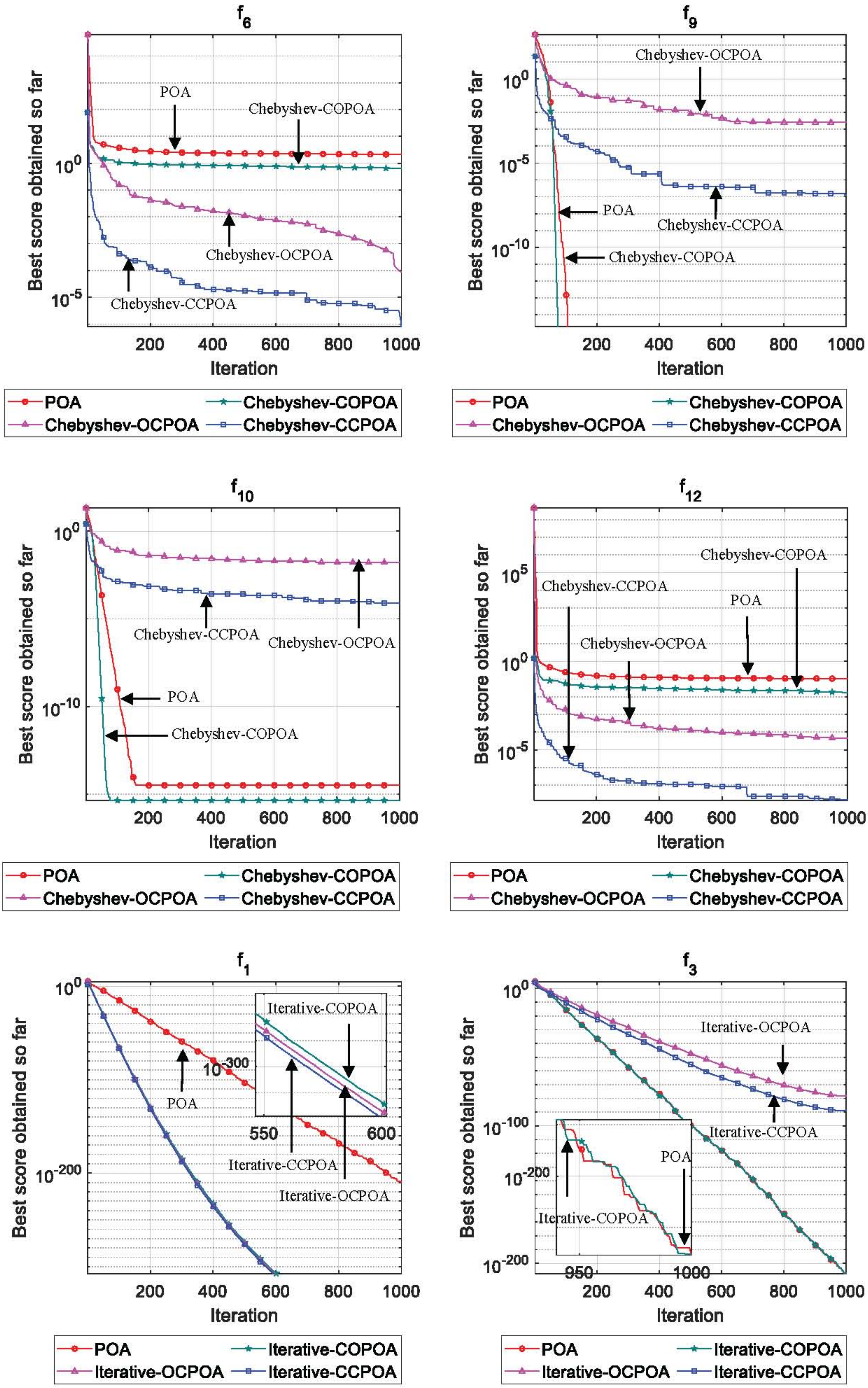

f1
Best score obtained so far
Iteration
Sinusoidal-COPOA
POA
Sinusoidal-OCPOA
Sinusoidal-CCPOA
POA
Sinusoidal-COPOA
Sinusoidal-OCPOA
Sinusoidal-CCPOA


f3
Best score obtained so far
Iteration
Sinusoidal-OCPOA
POA
Sinusoidal-CCPOA
Sinusoidal-COPOA
POA
Sinusoidal-COPOA
Sinusoidal-OCPOA
Sinusoidal-CCPOA



f6
Best score obtained so far
Iteration
Sinusoidal-OCPOA
POA
Sinusoidal-CCPOA
Sinusoidal-COPOA
POA
Sinusoidal-COPOA
Sinusoidal-OCPOA
Sinusoidal-CCPOA
f9
Best score obtained so far
Iteration
Sinusoidal-OCPOA
Sinusoidal-CCPOA
POA
Sinusoidal-COPOA
POA
Sinusoidal-COPOA
Sinusoidal-OCPOA
Sinusoidal-CCPOA
f10
Best score obtained so far
Iteration
Sinusoidal-OCPOA
Sinusoidal-CCPOA
POA
Sinusoidal-COPOA
POA
Sinusoidal-COPOA
Sinusoidal-OCPOA
Sinusoidal-CCPOA
f12
Best score obtained so far
Iteration
Sinusoidal-OCPOA
Sinusoidal-COPOA
Sinusoidal-CCPOA
POA
POA
Sinusoidal-COPOA
Sinusoidal-OCPOA
Sinusoidal-CCPOA
f1
Best score obtained so far
Iteration
Logistic-OCPOA
POA
Logistic-COPOA
Logistic-CCPOA
POA
Logistic-COPOA
Logistic-OCPOA
Logistic-CCPOA
f3
Best score obtained so far
Iteration
Logistic-CCPOA
Logistic-OCPOA
POA
Logistic-COPOA
POA
Logistic-COPOA
Logistic-OCPOA
Logistic-CCPOA


f6
f9
f10
f12
f1
f3
Best score obtained so far
Iteration
POA
Logistic-COPOA
Logistic-OCPOA
Logistic-CCPOA
Chebyshev-COPOA
Chebyshev-OCPOA
Chebyshev-CCPOA



f6
POA
Chebyshev-COPOA
Chebyshev-OCPOA
Chebyshev-CCPOA
Best score obtained so far
Iteration
POA
Chebyshev-COPOA
Chebyshev-OCPOA
Chebyshev-CCPOA
f9
Chebyshev-OCPOA
POA
Chebyshev-CCPOA
Chebyshev-COPOA
Best score obtained so far
Iteration
POA
Chebyshev-COPOA
Chebyshev-OCPOA
Chebyshev-CCPOA
f10
Chebyshev-CCPOA
Chebyshev-OCPOA
POA
Chebyshev-COPOA
Best score obtained so far
Iteration
POA
Chebyshev-COPOA
Chebyshev-OCPOA
Chebyshev-CCPOA
f12
Chebyshev-COPOA
Chebyshev-CCPOA
POA
Chebyshev-OCPOA
Best score obtained so far
Iteration
POA
Chebyshev-COPOA
Chebyshev-OCPOA
Chebyshev-CCPOA
f1
Iterative-COPOA
POA
Iterative-CCPOA
Iterative-OCPOA
Best score obtained so far
Iteration
POA
Iterative-COPOA
Iterative-OCPOA
Iterative-CCPOA
f3
Iterative-OCPOA
Iterative-CCPOA
POA
Iterative-COPOA
Best score obtained so far
Iteration
POA
Iterative-COPOA
Iterative-OCPOA
Iterative-CCPOA

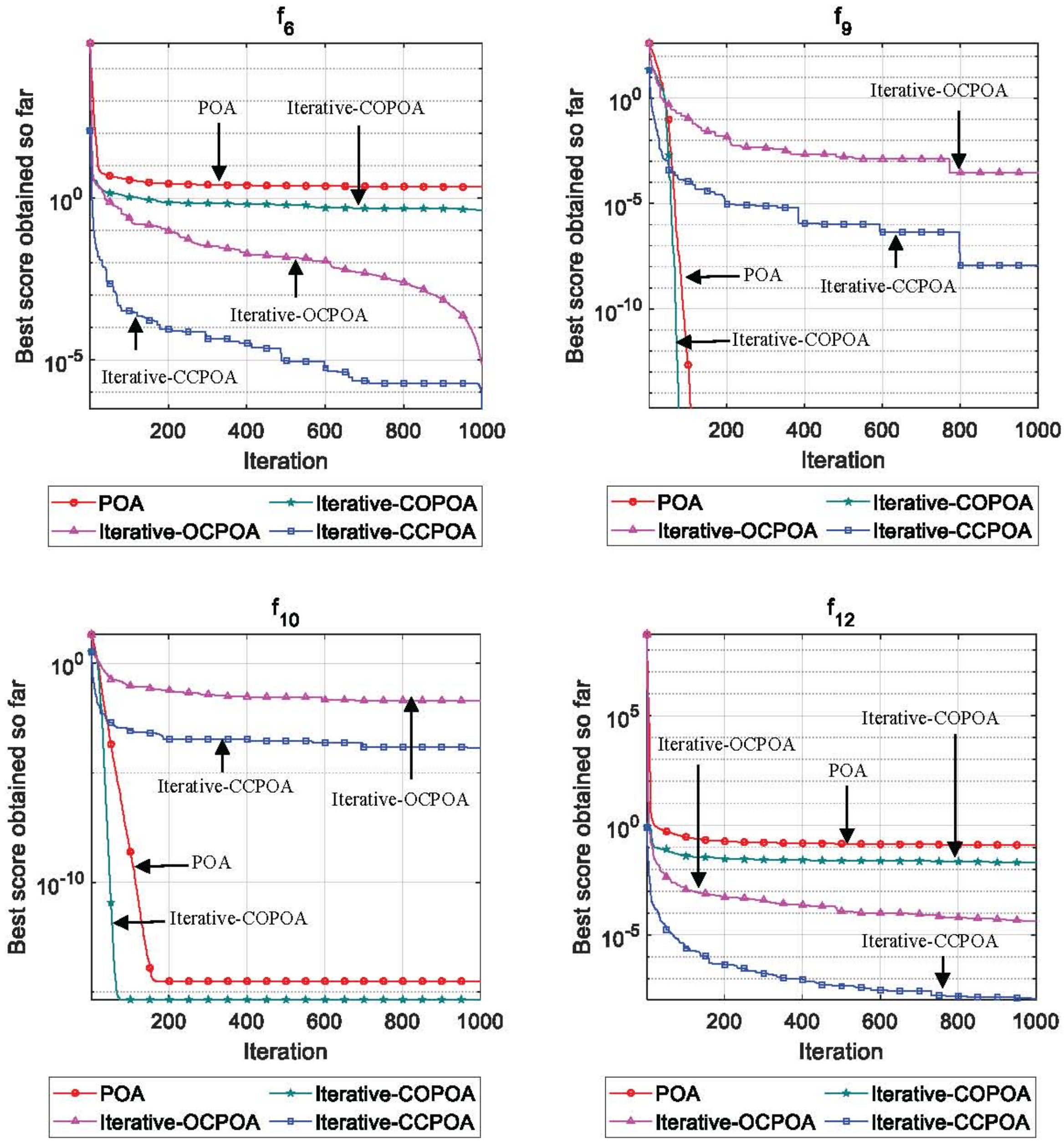


Fig.7. Convergence curve of POA, COPOA, OCPOA, CPOA under the benchmark test function

B. Comparing Sine-COPOA and other algorithms

Finally, the Sine-COPOA is compared with the classic PSO, GWO, and the latest Chimp Optimization Algorithm (ChOA) [33], as well as the Bonobo optimizer (BO) [34] in recent years. To ensure the scientific validity of the algorithm comparison, the number of repeated trials, population size N, and maximum iterations T are kept consistent with those mentioned above. The comparison results are presented in Table V and Fig.8.

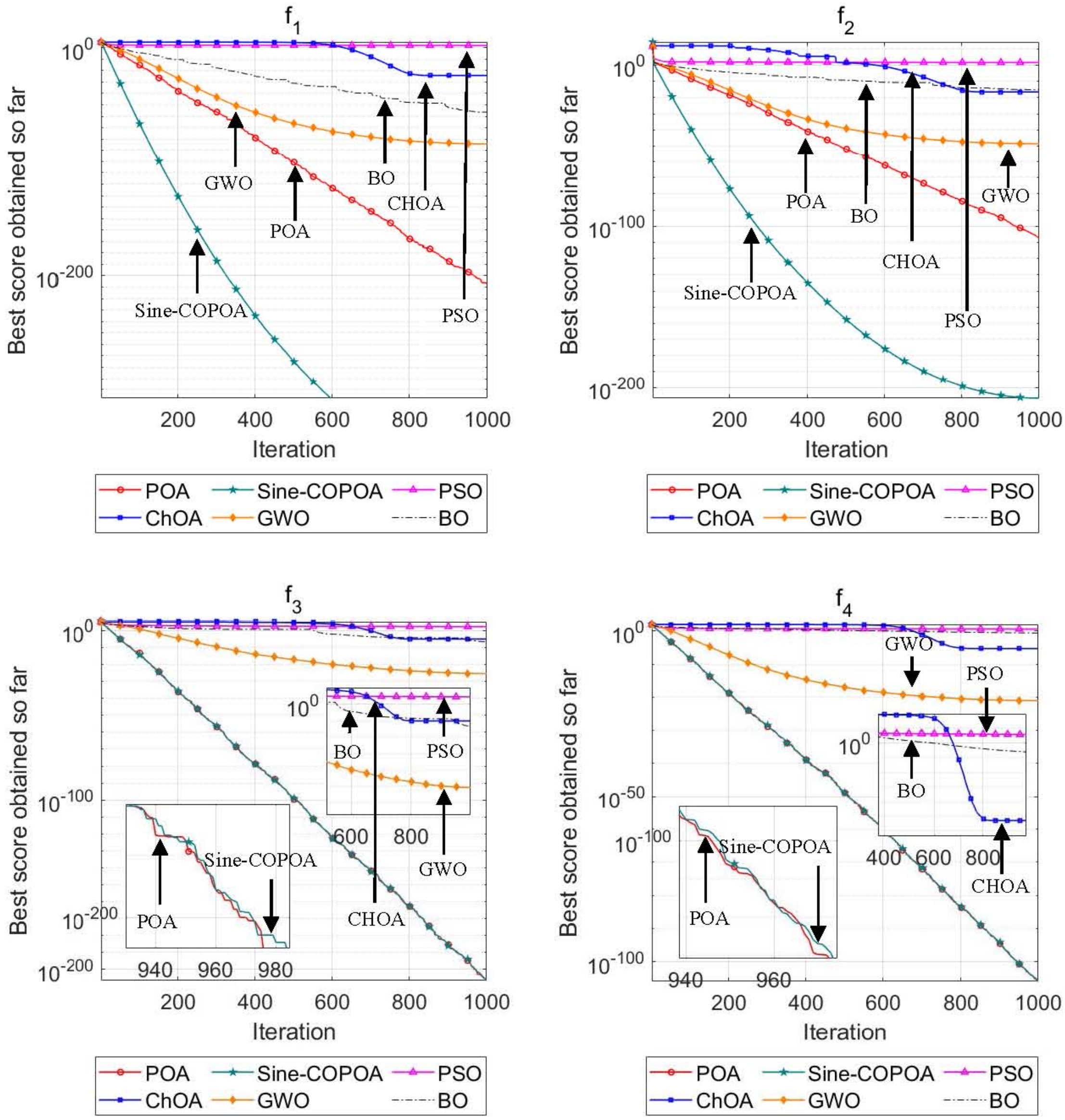
Comparing the six algorithms, it is observed that the Sine-COPOA algorithm exhibits strong performance in terms of f_1, f_2, f_3, f_4, f_9, f_{10}, f_{11}, and also attains faster convergence for the same function. On the other hand, the BO algorithm demonstrates high accuracy in finding solutions for f_5, f_6, f_{12}, f_{13}, and GWO demonstrates high accuracy in finding solutions for f_7. The overall combined outcomes suggest that Sine-COPOA outperforms in terms of both finding accuracy and convergence speed.

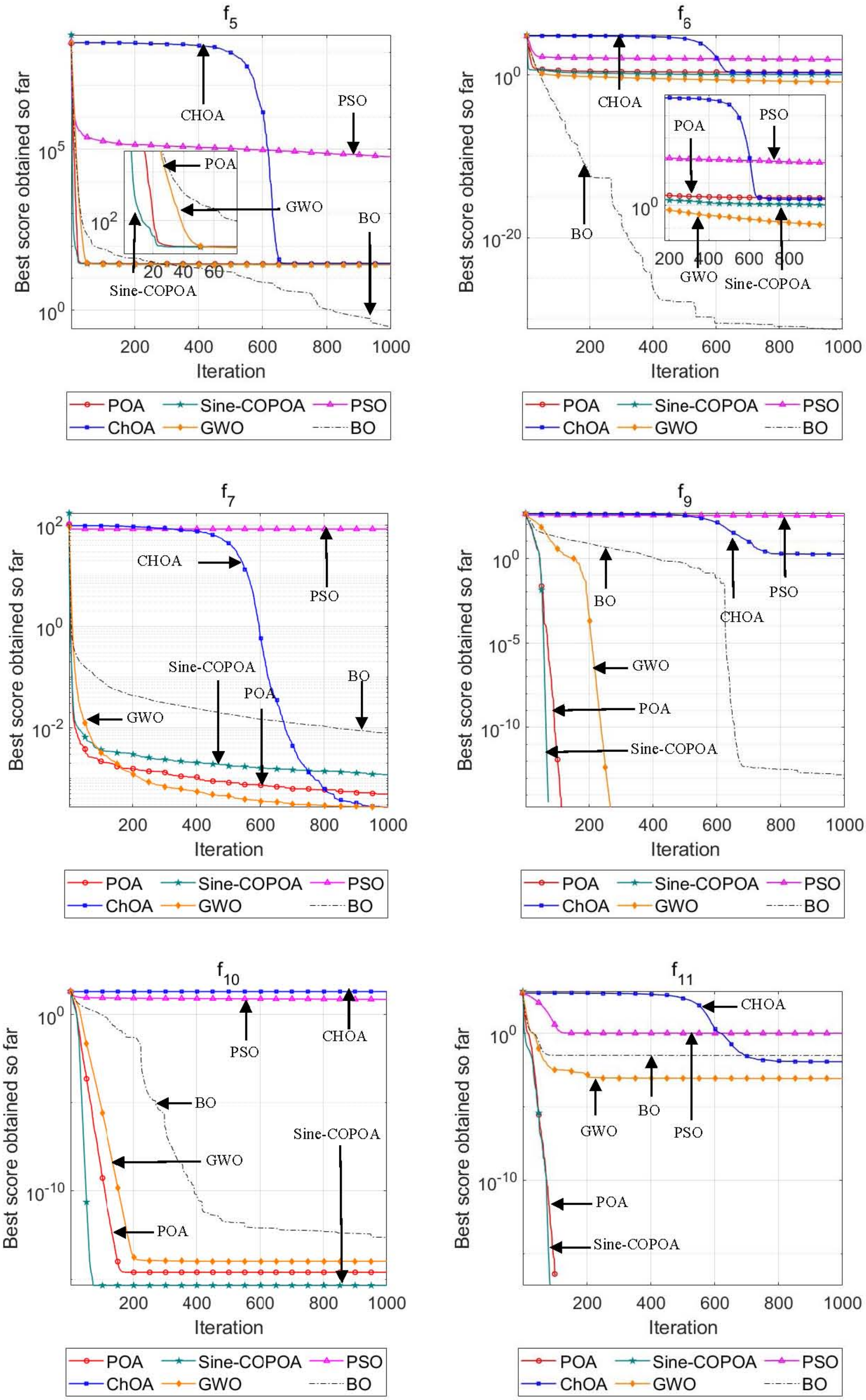
TABLE V
COMPARISON RESULTS OF SIX ALGORITHMS

Bunchmark functions	Statistics	COA	POA	COPOA	PSO	GWO	BO
f1	ave	2.0617E-26	3.3145E-211	**0.0000E+00**	7.6892E+01	1.0928E-85	4.6834E-57
	bsf	7.9893E-43	3.7282E-234	**0.0000E+00**	5.8342E+01	3.0667E-89	1.7564E-78
f2	ave	3.3060E-17	3.5731E-106	**2.1690E-208**	3.9369E+01	2.3127E-49	1.0846E-16
	bsf	2.7943E-24	**1.1803E-118**	2.1690E-208	3.0544E+01	3.7952E-50	1.7472E-21

TABLE V
COMPARISON RESULTS OF SIX ALGORITHMS

Bunchmark functions	Statistics	COA	POA	COPOA	PSO	GWO	BO
f3	ave	2.0709E-06	4.2572E-214	**7.9908E-215**	2.0292E+02	1.5472E-26	6.9262E-08
	bsf	9.0597E-11	**4.2881E-231**	7.9908E-215	1.3032E+02	1.9526E-31	2.6617E-11
f4	ave	7.7730E-06	9.0012E-105	**8.7034E-105**	3.5961E+00	8.6880E-22	2.6530E-01
	bsf	6.0343E-11	**1.3406E-116**	8.7034E-105	3.1390E+00	1.4251E-23	7.3000E-02
f5	ave	2.8748E+01	2.7410E+01	2.6468E+01	5.9011E+04	2.6130E+01	**5.8030E-01**
	bsf	2.8051E+01	2.5211E+01	2.6468E+01	3.6331E+04	2.5115E+01	**3.8008E-13**
f6	ave	1.8741E+00	2.2815E+00	1.1217E+00	7.2264E+01	1.7420E-01	**7.8311E-31**
	bsf	1.1482E+00	1.4728E+00	1.1217E+00	4.2223E+01	4.6056E-06	**0.0000E+00**
f7	ave	2.3820E-04	5.9311E-04	1.3000E-03	8.2038E+01	**2.4291E-04**	7.3000E-03
	bsf	1.7061E-05	1.7678E-04	1.3000E-03	4.2071E+01	**5.5445E-05**	2.9000E-03
f9	ave	1.7443E+00	**0.0000E+00**	**0.0000E+00**	3.3290E+02	3.2890E-01	1.4969E-13
	bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	3.0213E+02	**0.0000E+00**	5.6843E-14
f10	ave	1.9961E+01	3.4935E-15	**8.8818E-16**	7.2095E+00	1.1073E-14	1.7201E-13
	bsf	1.9958E+01	**8.8818E-16**	**8.8818E-16**	6.5068E+00	7.9936E-15	8.6153E-14
f11	ave	1.4400E-02	**0.0000E+00**	**0.0000E+00**	9.9280E-01	2.6020E-04	2.4600E-02
	bsf	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	9.2660E-01	**0.0000E+00**	**0.0000E+00**
f12	ave	2.0210E-01	9.4500E-02	3.8600E-02	3.0562E+00	1.6200E-02	**1.7128E-32**
	bsf	7.4400E-02	3.3800E-02	3.8600E-02	2.0550E+00	4.4042E-07	**1.5705E-32**
f13	ave	2.8812E+00	2.4031E+00	1.9927E+00	1.1911E+01	1.1550E-01	**2.0944E-29**
	bsf	2.6055E+00	1.1999E+00	1.9927E+00	9.1316E+00	4.6575E-06	**1.3498E-32**




f5
Best score obtained so far
Iteration
CHOA
PSO
POA
GWO
BO
Sine-COPOA
POA Sine-COPOA PSO
ChOA GWO BO
f6
CHOA
PSO
POA
BO
GWO
Sine-COPOA
f7
CHOA
PSO
Sine-COPOA
POA
BO
GWO
f9
BO
CHOA
PSO
GWO
POA
Sine-COPOA
f10
PSO
CHOA
BO
GWO
POA
Sine-COPOA
f11
CHOA
GWO
BO
PSO
POA
Sine-COPOA

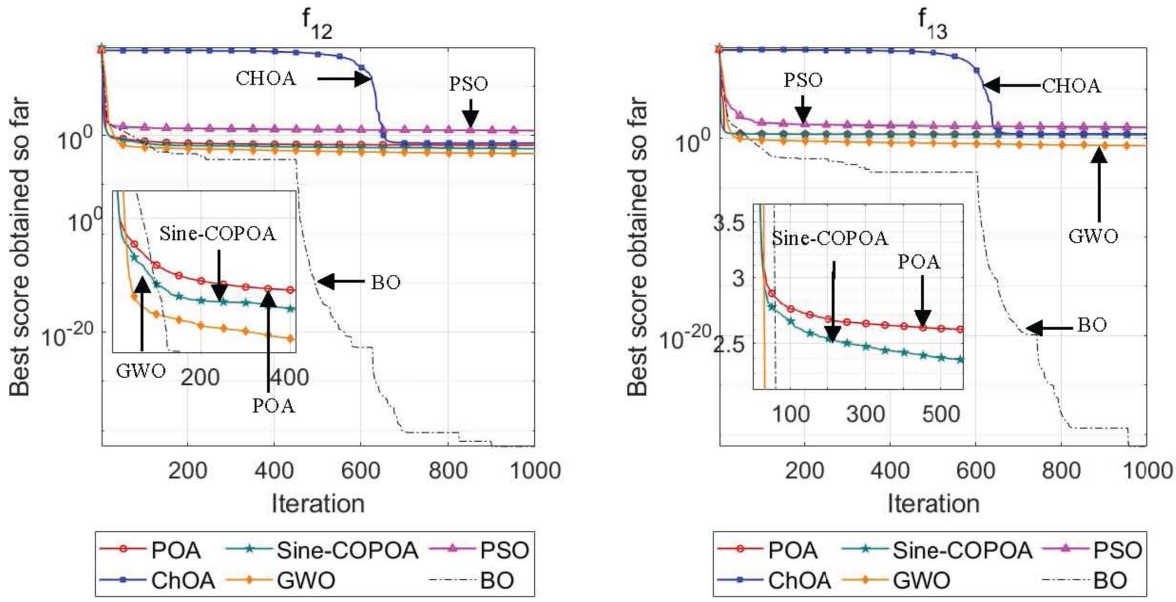


Fig.8. Convergence curve of POA, Sine- COPOA, PSO, ChOA, GWO, BO under the benchmark test functions

V. Engineering Optimization Design Problems

A. *Pressure vessel problem*

The objective of the pressure vessel optimization design problem is to minimize the economic cost while adhering to specific constraints. The economic cost comprises the materials, forming, and welding involved in constructing the vessel. The diagram of the pressure vessel problem model is shown in Fig.9. Four variables are used to calculate the objective function: shell thickness (z_1), head thickness (z_2), inner radius (x_3), and length of the vessel without including the head (x_4) [35]. The objective function and constraints of the pressure vessel optimization design problem are as follows.

Minimize :

$$f(\overline{x}) = 1.7781 z_2 x_3^2 + 0.6224 z_1 x_3 x_4 + 3.1661 z_1^2 x_4 + 19.84 z_1^2 x_3$$

Subject to :

$$g_1(\overline{x}) = 0.00954 X_3 \le z_2$$

$$g_2(\overline{x}) = 0.0193 X_3 \le z_1$$

$$g_3(\overline{x}) = X_4 \le 240$$

$$g_4(X) = 1296000 - \pi X_3^2 X_4 - \frac{4}{3}\pi X_3^3 \le 0$$

Where, $z_1 = 0.0625 x_1, z_2 = 0.0625 x_2$

With bounds: $x_1 \le 99,\quad 1 \le x_2,\quad x_3 \le 200,\quad 10 \le x_4$

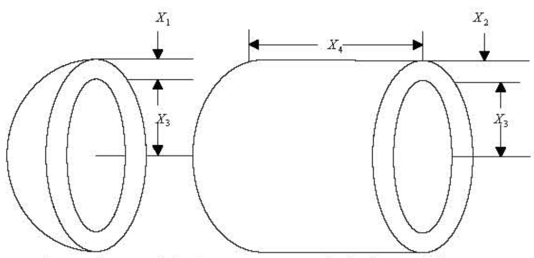


Fig.9. The model of pressure vessel design problem

The experimental performance of Sine-COPOA is compared with Whale Optimization Algorithm (WOA) [36], Harris Hawks Optimization (HHO) [37], African Vultures Optimization Algorithm (AVOA) [38], and Aquila Optimizer (AO) [39] for the pressure vessel design problem, as illustrated in Fig.10. To assess the convergence accuracy of Sine-COPOA and the aforementioned algorithms for the pressure vessel design problem, mathematical statistical tests, encompassing mean values and variances, are conducted and summarized in Table 6. Furthermore, the table compiles the optimal results obtained after running each algorithm for 30 iterations. It is evident from Table VI that AVOA, POA, and Sine-COPOA attain the identical minimum value for the pressure vessel design problem. Nevertheless, Sine-COPOA demonstrates greater stability in optimizing this problem, based on mean values and variances.

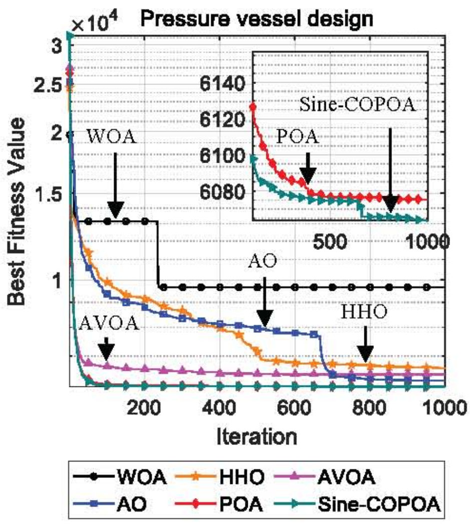


Fig.10. Convergence curve of WOA, HHO, AVOA, AO, POA, Sine-COPOA under the pressure vessel problem

A. *10-bar truss design*

This problem aims to minimize the weight of the truss structure, while ensuring compliance with frequency constraints. The design variable vectors A and x represent the cross-sectional areas of the bars and the nodal coordinates, respectively. The diagram of the 10-bar truss design model is shown in Fig.11 [40]. The mathematical formulation of this problem can be defined as follows.

$$\text{Minimize}: \quad f(\bar{x}) = \sum_{i=1}^{10} L_i(x_i)\rho_i A_i$$

$$\text{Subject to}: \quad g_1(\bar{x}) = \frac{7}{w_1(\bar{x})} - 1 \le 0$$

$$g_2(\bar{x}) = \frac{15}{w_2(\bar{x})} - 1 \le 0$$

$$g_3(\bar{x}) = \frac{20}{w_3(\bar{x})} - 1 \le 0$$

With bounds: $6.45\times10^{-5} \le A_i \le 5\times10^{-3}, \quad i = 1,2,\dots,10$

where, $\bar{x} = \{A_1, A_2, \dots, A_{10}\}, \quad \rho = 2770$

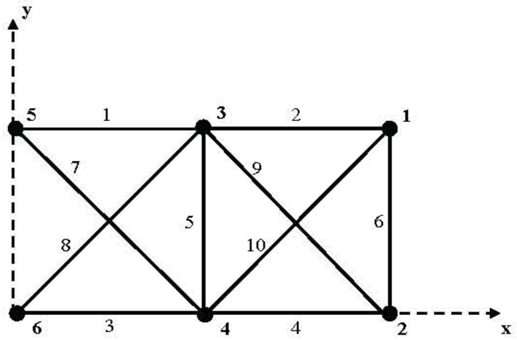


Fig.11. The 10-bar truss design model

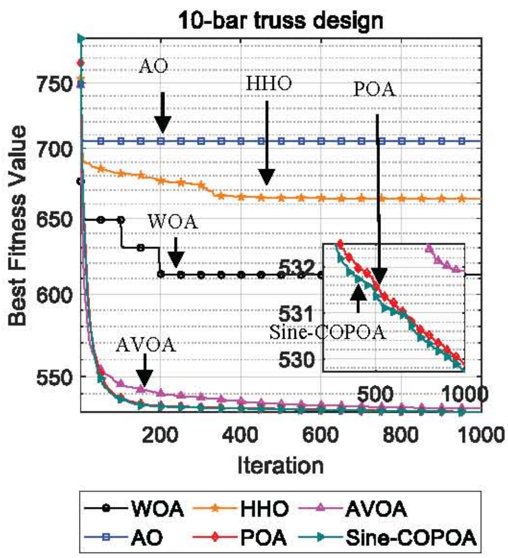


Fig.12. Convergence curve of WOA, HHO, AVOA, AO, POA, Sine-COPOA under the 10-bar truss design

The experimental performance graph from the experiment, comparing Sine-COPOA with WOA, HHO, AVOA, AO, and POA in the problem of designing a 10-bar truss, is depicted in Fig.12. To demonstrate the accuracy of convergence, a statistical test is conducted, as presented in Table VII. The table reveals that Sine-COPOA achieves the lowest minimum value in the 10-bar truss design problem, with a relatively smaller variance, indicating higher stability of the algorithm.

TABLE VI
COMPARISON RESULTS OF SIX ALGORITHMS UNDER THE PRESSURE VESSEL PROBLEM

	WOA	HHO	AVOA	AO	POA	Sine-COPOA
$f(x)$	6285.1881	6061.6692	6059.7143	6061.8690	6059.7143	6059.7143
x_1	13.7359	13.3997	12.8541	12.8831	13.2755	13.2934
x_2	6.7580	7.0999	6.9895	7.1711	6.8597	6.9848
x_3	43.7076	42.0826	42.0985	42.0836	42.0985	42.0985
x_4	157.6671	176.8341	176.6366	176.8357	176.6366	176.6366
ave	7475.0475	6628.1730	6461.0729	6313.2227	6106.1370	**6062.4309**
best	6285.1881	6061.6692	**6059.7143**	6061.8690	**6059.7143**	**6059.7143**
std	510803.3409	127951.0002	219137.5669	158450.2670	11168.3309	41.6361

TABLE VII
COMPARISON RESULTS OF SIX ALGORITHMS UNDER THE 10-BAR TRUSS DESIGN

	WOA	HHO	AVOA	AO	POA	Sine-COPOA
$f(x)$	560.7699	594.6512	525.0565	592.1279	524.6620	524.5705
x_1	0.0038	0.0038	0.0036	0.0031	0.0035	0.0035
x_2	0.0021	0.0016	0.0014	0.0020	0.0015	0.0015
x_3	0.0030	0.0038	0.0035	0.0039	0.0035	0.0035
x_4	0.0021	0.0015	0.0014	0.0015	0.0015	0.0015
x_5	0.0001	0.0002	0.0001	0.0012	0.0001	0.0001
x_6	0.0004	0.0013	0.0005	0.0011	0.0005	0.0005
x_7	0.0025	0.0015	0.0025	0.0022	0.0024	0.0024
x_8	0.0018	0.0038	0.0024	0.0023	0.0023	0.0023
x_9	0.0011	0.0010	0.0011	0.0011	0.0012	0.0012
x_{10}	0.0022	0.0017	0.0013	0.0019	0.0013	0.0013
ave	619.8361	668.3400	532.5948	698.7833	529.3527	**529.1495**
best	560.7699	594.6512	525.0565	592.1279	524.6620	**524.5705**
std	1498.6564	1840.0181	17.7471	2190.7548	7.1602	7.1389

VI. CONCLUSIONS

Chaotic mapping factors are used to adjust parameters in metaheuristic algorithms, and they have emerged as a prominent research topic in recent optimization literature. Three combinations consisting of COPOA, OCPOA, and CCPOA have been proposed to enhance the characteristics of the POA algorithm. The COPOA algorithm, with chaotic mapping factors incorporated solely into the pelican population, exhibits better optimization effects and high generalization among these combinations. Additionally, the CCPOA algorithm demonstrates good optimization findings under the Iterative map by incorporating chaotic mapping factors into both the pelican population and prey generation. When comparing the performance of Sine-COPOA with typical POA, PSO, GWO, BO, and ChOA in 12 benchmark test functions, the results indicate that Sine-COPOA outperforms the other algorithms in terms of convergence speed and finding accuracy in 7 functions. In the context of engineering optimization for pressure vessels and 10-bar truss designs, Sine-COPOA outperforms WOA, HHO, AVOA, AO, and POA in terms of convergence accuracy and stability. Future research should focus on enhancing the algorithm's global and local search capabilities, optimizing and improving it by incorporating methods such as optimal neighborhood perturbation and adaptive weighting, and applying it to tackle practical problems.

REFERENCES

[1] Guo, Meng-wei, et al. "Pseudo-parallel chaotic self-learning antelope migration algorithm based on mobility models." Applied Intelligence, vol.52, no.3, pp.2369-2410, 2022.

[2] Tian, Dongping, Xiaofei Zhao, and Zhongzhi Shi. "Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization." Swarm and Evolutionary Computation, vol.51, pp.100573, 2019.

[3] Zhu, Guopu, and Sam Kwong. "Gbest-guided artificial bee colony algorithm for numerical function optimization." Applied Mathematics and Computation, vol.217, no.7, pp.3166-3173, 2010.

[4] Bengio, Yoshua, Andrea Lodi, and Antoine Prouvost. "Machine learning for combinatorial optimization: a methodological tour d'horizon." European Journal of Operational Research, vol.290, no.2, pp.405-421, 2021.

[5] Kulkarni, Anand J., and Hinna Shabir. "Solving 0–1 knapsack problem using cohort intelligence algorithm." International Journal of Machine Learning and Cybernetics, vol.7, pp.427-441, 2016.

[6] Ouaarab, Aziz, Belaïd Ahiod, and Xin-She Yang. "Discrete cuckoo search algorithm for the travelling salesman problem." Neural Computing and Applications, vol.24, pp.1659-1669, 2014.

[7] Yap, David FW, et al. "A hybrid artificial immune systems for multimodal function optimization and its application in engineering problem." Artificial Intelligence Review, vol.38, pp.291-301, 2012.

[8] Dorigo, Marco, Mauro Birattari, and Thomas Stutzle. "Ant colony optimization." IEEE computational intelligence magazine, vol.1, no.4, pp.28-39, 2006.

[9] Kennedy, James, and Russell Eberhart. "Particle swarm optimization." Proceedings of ICNN'95-International Conference on Neural Networks. Vol. 4. IEEE, 1995.

[10] Jain, Mohit, Vijander Singh, and Asha Rani. "A novel nature-inspired algorithm for optimization: Squirrel search algorithm." Swarm and Evolutionary Computation, vol.44, pp.148-175, 2019.

[11] Yang, Xin-She, and Suash Deb. "Eagle strategy using Lévy walk and firefly algorithms for stochastic optimization." Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), pp.101-111, 2010.

[12] Askarzadeh, Alireza. "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm." Computers & Structures, vol.169, pp.1-12, 2016.

[13] Mirjalili, Seyedali, et al. "Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems." Advances in Engineering Software, vol.114, pp.163-191, 2017.

[14] Gandomi, Amir H., and Xin-She Yang. "Chaotic bat algorithm." Journal of Computational Science, vol.5, no.2, pp.224-232, 2014.

[15] Gharooni-fard, Golnar, et al. "Scheduling of scientific workflows using a chaos-genetic algorithm." Procedia Computer Science, vol.1, no.1, pp.1445-1454, 2010.

[16] Alatas, Bilal. "Chaotic harmony search algorithms." Applied Mathematics and Computation, vol.216, no.9, pp.2687-2699, 2010.

[17] Alatas, Bilal. "Chaotic bee colony algorithms for global numerical optimization." Expert Systems with Applications, vol.37, no.8 pp.5682-5687, 2010.

[18] Talatahari, S., et al. "Imperialist competitive algorithm combined with chaos for global optimization." Communications in Nonlinear Science and Numerical Simulation, vol.17, no.3, pp.1312-1319, 2012.

[19] J. Mingjun, T. Huanwen, Application of chaos in simulated annealing, Chaos, Solitons & Fractals, vol.21, pp.933–941, 2004.

[20] Mafarja, Majdi M., and Seyedali Mirjalili. "Hybrid whale optimization algorithm with simulated annealing for feature selection." Neurocomputing, 260 (2017): 302-312.

[21] Heidari, Ali Asghar, Rahim Ali Abbaspour, and Ahmad Rezaee Jordehi. "An efficient chaotic water cycle algorithm for optimization tasks." Neural Computing and Applications, vol.28, pp.57-85, 2017.

[22] Zhao, Weiguo, and Linming Zhao. "An Improved Bacterial Foraging Optimizer with Adaptive and Chaotic Search." Advanced Science Letters, vol.7, no.1, pp.305-308, 2012.

[23] Gupta, Shubham, and Kusum Deep. "An opposition-based chaotic grey wolf optimizer for global optimisation tasks." Journal of Experimental & Theoretical Artificial Intelligence, vol.31, no.5, pp.751-779, 2019.

[24] Tang, Yinggan, et al. "Parameter identification of time-delay chaotic system using chaotic ant swarm." Chaos, Solitons & Fractals, vol.41, no.4, pp.2097-2102, 2009.

[25] Gandomi, Amir Hossein, et al. "Chaos-enhanced accelerated particle swarm optimization." Communications in Nonlinear Science and Numerical Simulation, vol.18, no.2, pp.327-340, 2013.

[26] Rezaee Jordehi, A. "A chaotic artificial immune system optimisation algorithm for solving global continuous optimisation problems." Neural Computing and Applications, vol.26, pp.827-833, 2015.

[27] Chuang, Li-Yeh, Sheng-Wei Tsai, and Cheng-Hong Yang. "Chaotic catfish particle swarm optimization for solving global numerical optimization problems." Applied Mathematics and Computation, vol.217, no.16, pp.6900-6916, 2011.

[28] Anand, Priyanka, and Sankalap Arora. "A novel chaotic selfish herd optimizer for global optimization and feature selection." Artificial Intelligence Review, vol.53, no.2, pp.1441-1486, 2020.

[29] Hongyu Long, Yuqiang He, Yongsheng He, Chunyan Song, Qian Gao, and Hao Tan, "Application of Improved Honey Badger Algorithm in Multi-objective Reactive Power Optimization," IAENG International Journal of Applied Mathematics, vol. 52, no.4, pp.1105-1122, 2022.

[30] Yilin Han, Ye Tao, Wenyu Zhang, Wenhua Cui, and Tianwei Shi, "Perceptron Neural Network Image Encryption Algorithm Based on Chaotic System," IAENG International Journal of Computer Science, vol. 50, no.1, pp.42-50, 2023.

[31] Trojovský, Pavel, and Mohammad Dehghani. "Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications." Sensors, vol.22, no.3, pp.855, 2022.

[32] Haupt, Randy L., and Sue Ellen Haupt. Practical Genetic Algorithms. John Wiley & Sons, 2004.

[33] Khishe, Mohammad, and Mohammad Reza Mosavi. "Chimp optimization algorithm." Expert Systems with Applications, vol.149, pp.113338, 2020.

[34] Das, Amit Kumar, and Dilip Kumar Pratihar. "Bonobo optimizer (BO): an intelligent heuristic with self-adjusting parameters over continuous spaces and its applications to engineering problems." Applied Intelligence, vol.52, no.3, pp.2942-2974, 2022.

[35] He, Xiaoyu, and Yuren Zhou. "Enhancing the performance of differential evolution with covariance matrix self-adaptation." Applied Soft Computing, vol.64, pp.227-243, 2018.

[36] Mirjalili, Seyedali, and Andrew Lewis. "The whale optimization algorithm." Advances in Engineering Software, vol.95, pp.51-67, 2016.

[37] Heidari, Ali Asghar, et al. "Harris hawks optimization: Algorithm and applications." Future Generation Computer Systems, vol.97, pp.849-872, 2019.

[38] Abdollahzadeh, Benyamin, Farhad Soleimanian Gharehchopogh, and Seyedali Mirjalili. "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems." Computers & Industrial Engineering, vol.158, pp.107408, 2021.

[39] Abualigah, Laith, et al. "Aquila optimizer: a novel meta-heuristic optimization algorithm." Computers & Industrial Engineering, vol. 157, pp.107250, 2021.

[40] Ho-Huu, V., et al. "Optimal design of truss structures with frequency constraints using improved differential evolution algorithm based on an adaptive mutation scheme." Automation in Construction, vol.68 pp.81-94, 2016.