Applied Soft Computing 147 (2023) 110827

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Review article

An efficient Optimization State-based Coyote Optimization Algorithm and its applications

Qingke Zhang [a,*], Xianglong Bu [a], Zhi-Hui Zhan [b], Junqing Li [a], Huaxiang Zhang [a]

[a] School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China
[b] School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

ARTICLE INFO

Article history:
Received 15 November 2022
Received in revised form 26 August 2023
Accepted 31 August 2023
Available online 9 September 2023

Keywords:
Meta-heuristics algorithms
Coyote Optimization Algorithm
Population state estimation
Multi-thresholding image segmentation
Deployment problems of wireless sensor networks

ABSTRACT

Coyote Optimization Algorithm (COA) has demonstrated efficient performance by utilizing the multiple pack (subpopulation) mechanism. However, the fixed number of packs and a relatively singular evolutionary strategy limit its comprehensive optimization performance. Thus, this paper proposes a COA variant, referred to as the Optimization State-based Coyote Optimization Algorithm (OSCOA). In the OSCOA algorithm, a Population Optimization State Estimation Mechanism is employed for estimating the current population optimization state. Then, the estimation result is used to guide the algorithm in setting the number of packs appropriately as well as selecting appropriate evolutionary strategies to refine search directions, thereby avoiding blind exploration. Additionally, the estimation result assists each pack in selecting suitable parents to generate pups, further improving the global search efficiency of the algorithm. To validate the effectiveness of the proposed algorithm, the OSCOA algorithm is subjected to comprehensive testing and analysis along with seven efficient optimizers on 71 benchmark functions derived from the CEC2014, CEC2017, and CEC2022 benchmark suites. The results of these extensive experiments indicate the competitive performance of OSCOA. Furthermore, to further assess the capability of the OSCOA algorithm in addressing real-world problems, two practical applications is considered: wireless sensor network deployment and image segmentation. The outcomes of these applications further confirm the efficacy and stability of the OSCOA algorithm in tackling real-world scenarios.

Contents

* Corresponding author.
E-mail addresses: tsingke@sdnu.edu.cn (Q. Zhang), bullong@foxmail.com (X. Bu), zhanapollo@163.com (Z.-H. Zhan), lijunqing@lcu-cs.com (J. Li), huaxzhang@sdnu.edu.cn (H. Zhang).

https://doi.org/10.1016/j.asoc.2023.110827
1568-4946/

1. Introduction

In recent decades, with the advancement of science and societal progress, numerous optimization problems across various fields have been evolving towards high-dimensionality, nonlinearity, and an abundance of local optima. As a result, the computational complexity of these problems has surpassed the capacities of mathematics-based optimization algorithms.

To tackle these complex optimization problems, researchers proposed Meta-Heuristic Algorithms (MHAs) inspired by biological behaviors or physical phenomena in nature [1]. Within MHAs, Swarm Intelligence Optimization Algorithms (SIOAs) have gained significant prominence, with researchers introducing a plethora of SIOAs [2], including the classical Artificial Bee Colony (ABC) [3], the Social Spider Optimization (SSO) [4], the Whale Optimization Algorithm (WOA) [5], and the recently developed Salp Swarm Algorithm (SSA) [6], among others. Due to their simplicity, efficiency, and versatility, SIOAs have found wide-ranging applications in various fields. These applications include emotion recognition [7], image segmentation [8], control systems [9], wind power optimization [10], cloud computing [11], and more.

Coyote Optimization Algorithm (COA) [12], proposed by Pierezan et al. in 2018, is a novel SIOA. The algorithm models the unique search behavior of coyotes by simulating their living habits and social interactions. While the COA has demonstrated remarkable performance on certain global optimization problems, it still exhibits shortcomings, such as lower search efficiency and relatively slow convergence speed. To overcome these challenges, numerous enhancements and extensions have been proposed to enhance the optimization ability of COA algorithm for specific problems. An overview of several COA variants, detailing the modifications that researchers have implemented to address diverse problems, is presented in Table 1.

The DCOA [14] and LCOA [17] algorithms introduce the Levy flight mechanism into COA, which improves the population's ability to escape local optima. In addition, chaotic maps are repeatedly introduced into COA. For example, the DCOA algorithm [14] uses the sinusoidal chaotic map to solve the problem of premature convergence, the CCOA [16] uses ten different chaotic maps to improve the quality of population initialization and individual evolution processes, and the MCOA algorithm [15] introduces the tinkerbell chaotic map to help the population escape local optima.

To improve the convergence capability of the COA algorithm, some algorithms redesign evolution operators or introduce excellent strategies from other algorithms. For instance, the ECOA algorithm [19] replaces the cultural trend with the current optimal solution to improve the convergence speed. The HWOA [20] and HCOAG [21] algorithms incorporate excellent mechanisms from the Grey Wolf Optimizer [23] to enhance COA's search efficiency. The ICOA algorithm [22] redefines the cultural trend to improve the population's exploitation ability, and the CCOA algorithm [13] borrows the idea of a cultural algorithm to enhance the population's exploitation of dominant areas.

Moreover, some algorithms modify COA at the parameter level. For example, the FICOA algorithm [18] gradually shifts parameters of evolution operations from exploration to exploitation. The HCOAG algorithm [21] adjusts the number of coyotes to adjust the algorithm's performance, and the MCOA algorithm [15] introduces a parameter adaptive mechanism to assist in setting appropriate parameters for the algorithm.

From the perspective of exploration and exploitation, modifications to COA can be broadly classified into three categories.

Firstly, when dealing with complex or high-dimensional problems, it is necessary to enhance the algorithm's exploratory capabilities to prevent the population from getting trapped in local optima and experiencing premature stagnation. For instance, this can be achieved by introducing chaotic map [15] and Levy flight [17] mechanisms into the COA algorithm.

Secondly, when addressing relatively straightforward problems, variants incorporate strategies aimed at enhancing exploitation. For instance, one approach involves utilizing the current optimal solution to guide the population's search process [19].

Finally, the third modification involves introducing adaptive mechanisms into the algorithm, which can be further classified as either active or passive. The active adaptive mechanism facilitates a gradual transition from exploration to exploitation as the population undergoes iterations, while the passive adaptive mechanism adjusts the optimization strategy based on information feedback from the population.

The first two of the three categories can effectively enhance the exploration and exploitation of the COA algorithm when applied to problems with known objective function characteristics.

Table 1
COA variants overview.

Modification	Year	Problems	Description
CCOA [13]	2019	CEC2017;Heavy duty gas turbine operation problems	Combining normative space techniques in cultural algorithms with COA.
DCOA [14]	2020	Proton exchange membrane fuel cell problems	Introduction of Lévy Flight helps populations move out of local optimum; Use of sinusoidal chaotic map to avoid premature population maturation.
MCOA [15]	2021	Truss optimization problems	Introducing the Tinkerbell chaotic map to assist the population in escaping from local optima; Designing a parameter adaptive mechanism to improve algorithm performance.
CCOA [16]	2021	CEC2005	10 chaotic maps are used for population initialization and population upgrading.
LCOA [17]	2021	CEC2005;Optimal power flow problems	Adding Lévy Flight to the process of producing pups.
FICOA [18]	2021	Fuzzy multilevel image thresholding problems	Designing the parameters in the update operator to decrease gradually helps the population transition gradually from exploration to exploitation; Fuzzy median aggregation with local neighborhood information is introduced to COA.
ECOA [19]	2021	Radial distribution networks	The global optimal coyote replaces cultural trends and novel pup generation mechanisms are constructed, both of which are modifications to enhance population exploitation.
HWOA [20]	2021	Sliding mode direct torque control problems	Individuals choose upgrade methods for GWO or COA based on probability.
HCOAG [21]	2022	CEC2017;Clustering optimization	A Gaussian global-best growing operator is designed to improve convergence speed; A dynamic adjustment scheme of coyote number in each group is proposed to enhance the search ability and operability; The combination of simplified GWO and COA proposes a new upgrade operator to improve algorithm performance.
ICOA [22]	2022	CEC2017;Optimal load forecasting models	Using Sobol sequence to initialize the population provides the initial population with coverage ability; Reconstructing culture trends to improve the population convergence rate

However, these strategies may not be directly applicable to black-box optimization problems, where the objective function remains poorly understood.

The active adaptive mechanism is commonly used in algorithm design to guide the selection of evolutionary strategies during different stages of evolution. Notable examples of such algorithms are the Equilibrium Optimizer (EO) [24], Harris Hawks Optimizer (HHO) [25], and Artificial Electric Field Algorithm (AEFA) [26]. However, this mechanism has limitations. For instance, in relatively straightforward problems, excessive computational resources might be allocated to the early exploration phase, resulting in slow convergence of the population. Conversely, for complex problems, the early exploration capacity might not be sufficient, potentially causing the population to become trapped in local optima.

The passive adaptation mechanism, unlike the active adaptive mechanism that solely relies on the number of population iterations, incorporates auxiliary information to guide the selection of appropriate evolutionary strategies. Common auxiliary information includes historical parameters, fitness values, and population diversity, among others. However, these typical sources of auxiliary information are derived solely from either the solution space or the objective space, potentially leading to an incomplete representation of the population's true states. To address these shortcomings, the population optimization state estimation methods have emerged. These methods aim to provide a more accurate estimation of the population optimization states by integrating information from both the objective space and the solution space.

Zhan et al. proposed a method of estimating the population optimization states in the APSO [27], which calculates the "evolutionary factor" f to guide the evolution process of PSO. However, the method necessitates calculating distances among all individuals, resulting in significant computational demand. To address this computational intensity, Zhan et al. developed the ADDE algorithm [28], which only calculates the distance between the optimal individual and the median individual to estimate the population states. Yu et al. achieved a population optimization state estimation method by calculating the distance ranking of individuals from the optimal individual in the objective and solution spaces in ADE algorithm [29]. Li et al. designed the DEET algorithm [30] using Fitness-Distance Correlation (FDC) to estimate the population state.

However, the aforementioned algorithms share two prevailing limitations. Firstly, they hinge on fixed thresholds for partitioning population optimization states, thereby presenting difficulties in ascertaining suitable threshold values. Inaccurate threshold configuration can profoundly misdirect the algorithm and prompt the adoption of unsuitable strategies. Secondly, these algorithms exclusively depend on the current population optimization state to deduce all evolutionary strategies, consequently curbing their ability of fault tolerance.

Addressing these aforementioned limitations, this paper introduces an advanced optimization algorithm called the Optimization State-based Coyote Optimization Algorithm (OSCOA), which integrates the Indicator of Optimization State (IOS) [29] into the COA framework. The OSCOA algorithm employs a probabilistic-guided approach for the selection of various optimization strategies, thus alleviating the complexity associated with threshold setting. Additionally, the independent selection of diverse strategies enhances the fault tolerance of the IOS technique and significantly diversifies the algorithm's search patterns. And the main contributions of this paper focus on four aspects:

- A IOS-guided dynamic pack management mechanism is proposed.
- A IOS-guided multiple growth strategy mechanism is designed.
- A IOS-guided parent selection mechanism for pups is proposed.
- The strategy selection of the above three mechanisms is designed to be independent of each other, which enriches the search model of the algorithm.

To evaluate the performance of the OSCOA algorithm, this paper compares it with seven outstanding SIOAs. The comprehensive performance of OSCOA is assessed using well-established benchmark suites : CEC2014 and CEC2017. Furthermore, the latest benchmark suite, CEC2022, is employed to evaluate OSCOA's performance on the most recent optimization problems. Experimental results confirm the robust comprehensive optimization capability of OSCOA.

Moreover, this paper presents two real-world applications to assess the proposed algorithm's performance: wireless sensor network deployment and multi-thresholding image segmentation. Simulation experiments demonstrate that OSCOA exhibits remarkable proficiency in solving real-world problems. It should be noted that the descriptions of coyote behavior in the following paper all occur on cth coyote within the pack p and at moment t, and the variable *rand* represents a random real number uniformly distributed in the range [0, 1].

2. COA algorithm

The COA is a SIOA proposed by simulating the social behavior of coyotes. In COA algorithm, coyotes are randomly divided into N_p packs, each containing N_c coyotes, resulting in a total of $N = N_p \times N_c$ coyotes.

The evolutionary process of coyotes can be broadly divided into four phases: Coyote initialization, Coyote growth, Pup birth and death, and Coyote migration between packs. The pseudocode for COA is presented in Algorithm 1.

2.1. Coyote initialization

The behavior of coyotes is affected by both intrinsic and extrinsic factors. Intrinsic factors include sex, social status, and pack membership, while extrinsic factors include snow depth, snowpack hardness, and temperature and so on. Collectively, these factors can be referred to as social conditions, denoted as $\boldsymbol{soc}$. Therefore, the COA model represents coyotes using their social conditions, and the cth coyote in pack p is expressed as shown in Eq. (1).

$$\boldsymbol{soc_c^{p,t}} = \left[soc_{c,1}^{p,t}, soc_{c,2}^{p,t}, \ldots, soc_{c,D}^{p,t}\right] \tag{1}$$

Then, the social conditions of the coyotes are randomly initialized inside the search region according to Eq. (2).

$$soc_{c,j}^{p,t} = lb_j + rand \cdot \left(ub_j - lb_j\right) \qquad j = [1, 2, \ldots, D] \tag{2}$$

where ub_j and lb_j denote the lower and upper bounds of the jth dimension of the search space, respectively.

$$fit_c^{p,t} = f\left(\boldsymbol{soc_c^{p,t}}\right) \tag{3}$$

After the population is initialized, the fitness of the coyotes is evaluated using Eq. (3).

2.2. Coyote growth

During the coyote growth phase, each coyote is influenced by two factors. First, each coyote is guided by the best coyote $\boldsymbol{alpha}$ within the same pack, and this process is illustrated by Eq. (4).

$$\boldsymbol{\delta_1} = \boldsymbol{alpha^{p,t}} - \boldsymbol{soc_{cr_1}^{p,t}} \tag{4}$$

where $\boldsymbol{soc_{cr_1}^{p,t}}$ denotes a randomly selected coyote from pack p. Second, the cultural tendency $\boldsymbol{cult}$ within the same pack also affects the coyote growth. The $\boldsymbol{cult}$ is the median social conditions of coyotes, which can be calculated by Eq. (5).

$$cult_j^{p,t} = \begin{cases} O_{\frac{N_c+1}{2},j}^{p,t} & N_c \text{ is odd} \\ \dfrac{O_{\frac{N_c}{2},j}^{p,t} + O_{\frac{N_c}{2}+1,j}^{p,t}}{2} & N_c \text{ is even} \end{cases} \tag{5}$$

where $O_{(N_c+1)/2,j}^{p,t}$ denotes the $(N_c+1)/2$th variable of an ordered sequence of jth social condition in pack p. Therefore, the second factor affecting coyote growth is shown in Eq. (6).

$$\boldsymbol{\delta_2} = \boldsymbol{cult^{p,t}} - \boldsymbol{soc_{cr_2}^{p,t}} \tag{6}$$

where $\boldsymbol{soc_{cr_2}^{p,t}}$ is a coyote randomly selected from pack p.

By combining the Eqs. (4) and (6), the complete growth operator is shown in Eq. (7).

$$\boldsymbol{new\_soc_c^{p,t}} = \boldsymbol{soc_c^{p,t}} + rand \cdot \boldsymbol{\delta_1} + rand \cdot \boldsymbol{\delta_2} \tag{7}$$

Subsequently, the fitness of the new coyotes is calculated using Eq. (8).

$$new\_fit_c^{p,t} = f(\boldsymbol{new\_soc_c^{p,t}}) \tag{8}$$

Then the greedy selection is performed using Eq. (9) to allow the better coyotes to enter the next generation.

$$\boldsymbol{soc_c^{p,t+1}} = \begin{cases} \boldsymbol{new\_soc_c^{p,t}} & new\_fit_c^{p,t} < fit_c^{p,t} \\ \boldsymbol{soc_c^{p,t}} & otherwise \end{cases} \tag{9}$$

Additionally, each coyote possesses an age attribute, and following its growth, it updates its age using Eq. (10).

$$age_c^{p,t+1} = age_c^{p,t} + 1 \tag{10}$$

2.3. Pup birth and death

In each pack, the COA algorithm randomly selects two coyotes, and their social conditions are employed to generate a pup. The pup generation process is determined by Eq. (11).

$$pup_j^{p,t} = \begin{cases} soc_{cr_3,j}^{p,t} & r_j < P_s \text{ or } j = j_1 \\ soc_{cr_4,j}^{p,t} & r_j \geq P_s + P_a \text{ or } j = j_2 \\ R_j & otherwise \end{cases} \tag{11}$$

where r_j denotes a random real number uniformly distributed in the range of [0, 1], $\boldsymbol{soc_{cr_3}^{p,t}}$ and $\boldsymbol{soc_{cr_4}^{p,t}}$ are two randomly selected coyotes within the pack p, and j_1 and j_2 are two randomly chosen dimensions to ensure that the pup inherit social conditions from the parent coyotes, R_j denotes the randomly generated variables in jth dimension, P_s and P_a denote the scatter and association probability, respectively, and are defined by Eqs. (12) and (13).

$$P_s = \frac{1}{D} \tag{12}$$

Algorithm 1: COA

Input: pack size N_c; the number of packs N_p; maximum number of iterations MaxIter; lower bound $\boldsymbol{lb}$; upper bound $\boldsymbol{ub}$; dimensions of search space D
Output: the best solution

```
Randomly initialize the population of size N using Eq. (2);
Initialize the age = 0 and Iter = 1 ;
Calculate the fitness of initial population using Eq. (3) ;
while Iter ≤ MaxIter do
    for i = 1 : N_p do
        Locate the best coyote alpha in the current packs ;
        for j = 1 : N_c do
            Update the coyotes using Eq. (7) ;
            Perform out-of-bounds detection ;
            Calculate the fitness of new coyote using Eq. (8) ;
            Perform greedy selection using Eq. (9) ;
        end
        Generate a pup using two randomly selected coyotes using Eq. (11) ;
        Determine whether to let pup join the new population ;
    end
    age = age + 1 ;
    A coyote conducts pack migration between packs ;
end
return the best solution ;
```

$$P_a = \frac{1 - P_s}{2} \tag{13}$$

After a pup is born, the COA algorithm compares it with the coyotes within the same pack. If the pup cannot outperform any of the compared coyotes, it is discarded. However, if there are coyotes worse than the pup, it replaces the oldest among them.

2.4. Coyote migration between packs

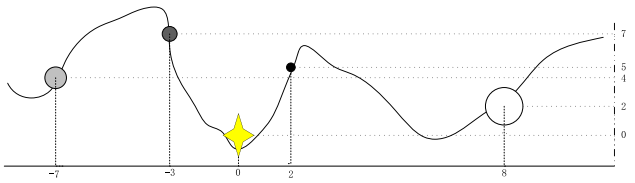
Initially, coyotes are randomly assigned to packs, but sometimes they may occasionally migrate from their current pack to another. This migration behavior facilitates information sharing among packs, enhancing their collective learning process. The occurrence of migration depends on the pack size N_c, and its probability is denoted as P_e.

$$P_e = 0.005 \cdot N_c^2 \tag{14}$$

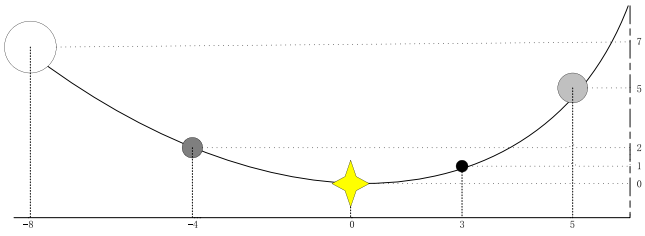
3. OSCOA algorithm

In this paper, a new version of the coyote optimization algorithm, OSCOA, is introduced, which incorporates a population optimization state estimation method. The estimation result aids in dynamically adjusting the number of packs and guides the selection of suitable search strategies. Moreover, the estimation result also influences the selection of suitable parents for generating pups.

Consequently, the key components of the OSCOA algorithm can be summarized into four aspects: population optimization state estimation, dynamic pack management mechanism, multiple growth strategy mechanism, and parent selection mechanism. The ensuing subsections furnish an exhaustive delineation of the OSCOA algorithm encompassing these four aspects.



(a) A chaotic population state



(b) An orderly population state

Fig. 1. States of population distributions.

3.1. Population optimization state estimation

For SIOAs, the population optimization states can assist the algorithm in adjusting search strategies promptly, reducing blind search, and consequently improving the search efficiency. Therefore, an effective population optimization state estimation method, Indicator of the Optimization State (IOS) [29], is introduced into the OSCOA algorithm for estimating the population optimization state. The value of IOS (denoted by c) is calculated in Algorithm 2.

Algorithm 2: IOS

Input: Coyote population and fitness values
Output: c

1 Rank coyotes in the entire population according to their fitness (from strongest to weakest) and the ranking of the i^{th} coyote is recorded as f_i;
2 Calculate the Euclidean distance between each coyote and the optimal coyote, and then rank the distances (from near to far), and the distance ranking of the i^{th} coyote is recorded as d_i;
3 Calculate the IOS according to the following equation

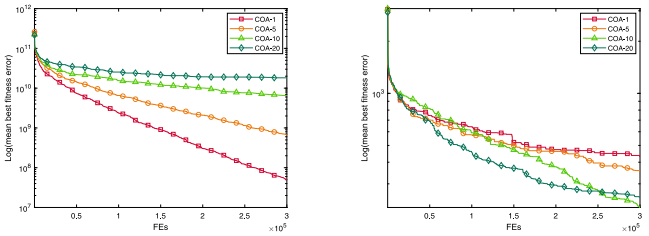
$$IOS = \sum_{i=1}^{N} |f_i - d_i| \tag{15}$$

4 IOS_{min} is the minimum value 0 when f_i and d_i are equal for each coyote, and then the maximum IOS value is calculated by the following equation.

$$IOS_{max} = \begin{cases} \frac{N^2}{2} & N \text{ is even} \\ \frac{(N+1)(N-1)}{2} & N \text{ is odd} \end{cases} \tag{16}$$

5 Normalize the IOS to ensure that it is between 0 and 1, using c to denote the value.

$$c = \frac{IOS - IOS_{min}}{IOS_{max} - IOS_{min}} \tag{17}$$



(a) unimodal funnction (b) hybrid function

Fig. 2. Convergence curves for COA with different N_p.

To facilitate the understanding of IOS, Fig. 1 provides two optimization states of the population. In these figures, the yellow star represents the current best individual, the gray circle denotes the other individuals in the population. The size of the gray circle indicates its distance from the best individual. In Fig. 1(a) the population distribution is chaotic, whereas in Fig. 1(b) the populations are clustered around the optimal solution, showing an ordered state. The values of c for Figs. 1(a) and 1(b) are 0.6667 and 0, respectively. Thus, a higher c value signifies a relatively chaotic state of the population in the search space, indicating a tendency towards exploration. Conversely, a lower c value indicates a more ordered state, suggesting an inclination towards exploitation. Based on this analysis, it can be concluded that IOS possesses the capability to estimate the population optimization state.

3.2. Dynamic pack management mechanism

Conventional COA primarily steers the population evolution through ***alphas***. However, each pack contains only one ***alpha***, resulting in the total number of ***alphas*** being equal to N_p.

In general, when more individuals participate in guiding the population evolution, the exploration area of the population expands. However, this also leads to a decrease in convergence speed. Conversely, fewer guiding individuals accelerate population convergence speed but elevate the risk of the algorithm becoming trapped in local optima [31]. To validate this assertion, Fig. 2 displays the convergence curves of the COA algorithm with varying N_p. Figs. 2(a) and 2(b) showcase the results for the unimodal function F1 and the hybrid function F20 from CEC2017, respectively. Evidently, algorithms with smaller N_p exhibit superior performance on simpler problems. On the other hand, for complex multimodal problems, algorithms with larger N_p excel. Thus, the appropriate choice of N_p varies depending on different problems [32].

In a bid to select a more suitable N_p, OSCOA dynamically adjusts N_c based on feedback from the IOS. The calculation procedure for N_c is presented in Eq. (18).

$$N_c^{t+1} = \begin{cases} N_c^t - 1 & r_1 < c \quad and \quad N_c^t > LN_c \\ N_c^t + 1 & r_1 \geq c \quad and \quad N_c^t < UN_c \\ N_c^t & otherwise \end{cases} \tag{18}$$

where r_1 denotes a random real number uniformly distributed in the range of [0,1], while UN_c and LU_c are the upper and lower bounds of N_c, respectively.

Since N_c changes with population iteration, thus, the OSCOA algorithm replaces coyote migration with random grouping in each iteration, the exchange of population information can be effectively accomplished across packs. Algorithm 3 outlines the precise process for grouping populations.

Algorithm 3: Population grouping

Input: Population size N; The packs size N_c;
Output: Packs
$N_p = floor(N/N_c)$;
$rem = N\%N_c$;
if $rem == 0$ **then**
  Divide the population into N_p packs of equal size ;
else
  Divide $(N - rem)$ individuals into N_p packs ;
  Randomly select rem packs;
  Each selected packs receives one remaining individual;
end

3.3. Multiple growth strategy mechanism

The OSCOA algorithm provides different growth strategies for different population optimization states. For the exploration states, OSCOA adds weights for δ_1 and δ_2, which are presented in Eq. (19).

$$\boldsymbol{new\_soc_c^{p,t}} = \boldsymbol{soc_c^{p,t}} + w_1 \cdot rand \cdot \boldsymbol{\delta_1} + w_2 \cdot rand \cdot \boldsymbol{\delta_2} \tag{19}$$

$$w_1 = rand \cdot 4 - 2 \tag{20}$$

$$w_2 = rand \cdot 4 - 2 \tag{21}$$

Since larger step sizes are an effective means to enhance algorithm exploration, therefore, w_1 and w_2 are set as random real numbers on the interval $[-2, 2]$. For the exploitation state, the coyotes growth strategy is shown in Eq. (22).

$$\boldsymbol{new\_soc_c^{p,t}} = \boldsymbol{soc_c^{p,t}} + w_3 \cdot rand \cdot \boldsymbol{\delta_1} + w_4 \cdot rand \cdot \boldsymbol{\delta_2} \tag{22}$$

$$w_3 = 0.9 - 0.6 \cdot \frac{FEs}{MaxFEs} \tag{23}$$

$$w_4 = 0.9 - 0.6 \cdot \frac{r_{FEs}}{MaxFEs} \tag{24}$$

where FEs denotes the number of function estimations, MaxFEs denotes the maximum number of function estimations, r_{FEs} denotes a random integer between FEs and MaxFEs. The w_3 gradually decreases with population iterations, thus enhancing the population's exploitation ability over time. On the other hand, w_4 is a smaller and random value compared to w_3, aiming to reduce the impact of ***cult*** and facilitating precise exploitation of the population.

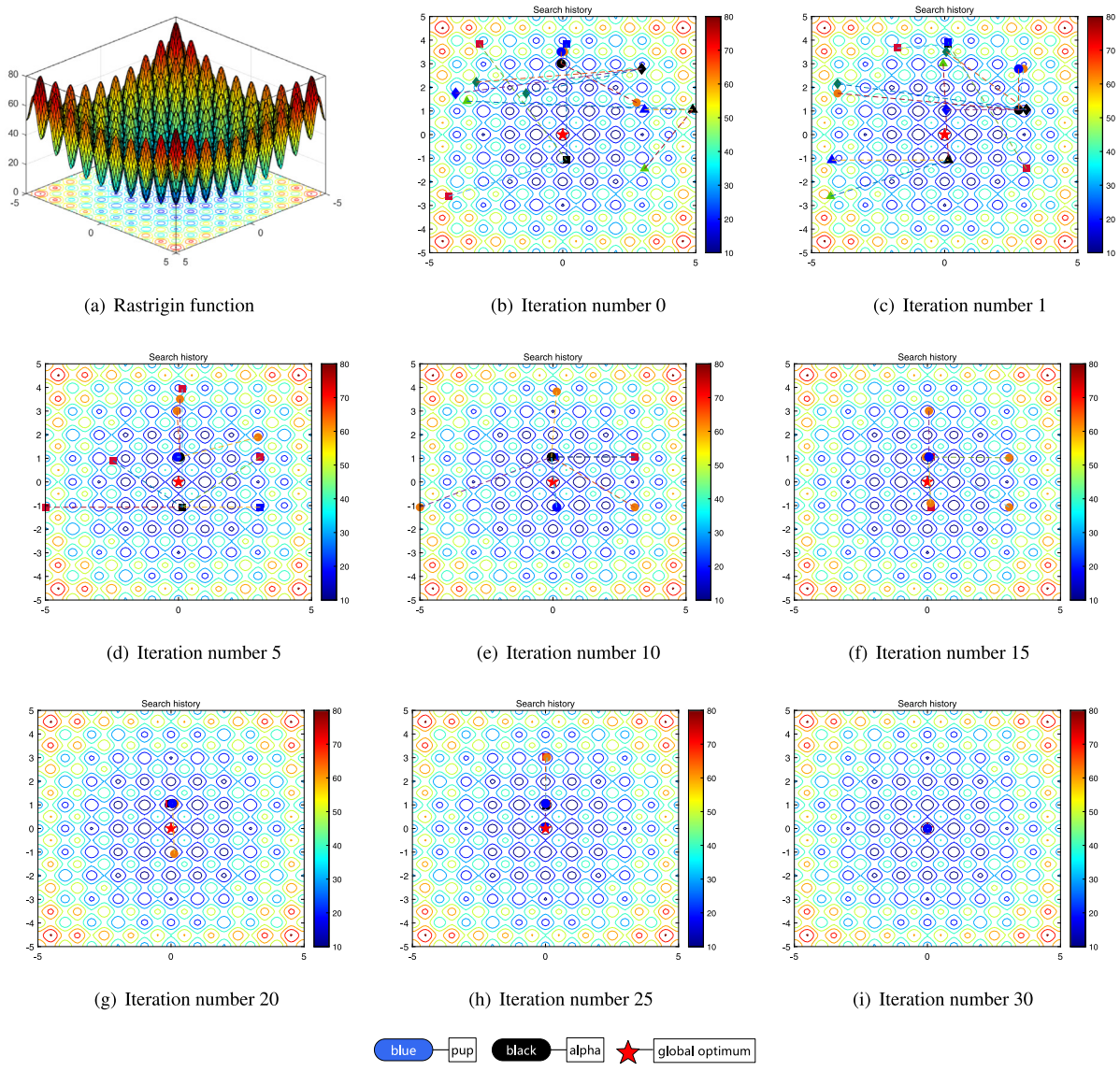
The growth strategy of coyotes is selected at this stage by Eq. (25).

$$coyote\ operator = \begin{cases} Eq.\ (19) & rand < c \\ Eq.\ (22) & otherwise \end{cases} \tag{25}$$

To prevent the population from exploring invalid spaces, each growing coyote must adhere to the following boundary constraint.

$$new\_soc_{c,j}^{p,t} = \min(ub_j, \max(new\_soc_{c,j}^{p,t}, lb_j)) \tag{26}$$

After updating the coyotes, coyotes with better fitness among the new and old coyotes are selected to enter the next generation population using Eq. (9).



(a) Rastrigin function (b) Iteration number 0 (c) Iteration number 1

(d) Iteration number 5 (e) Iteration number 10 (f) Iteration number 15

(g) Iteration number 20 (h) Iteration number 25 (i) Iteration number 30

Fig. 3. Position of coyotes in iteration.

3.4. Parent selection mechanism

In the conventional COA algorithm, the scatter probability P_s is inversely proportional to the dimensionality. As a result, in high-dimensional scenarios, the pup generation is primarily influenced by R_j, emphasizing exploratory potential but significantly compromising population convergence. To mitigate this concern, OSCOA introduces a revamped pup generation approach tailored to this stage.

Firstly, the OSCOA algorithm randomly selects a coyote from the current pack, excluding the $\boldsymbol{alpha}^{p,t}$, as the first parent denoted as $\boldsymbol{parent}_{\boldsymbol{1}}^{\boldsymbol{p,t}}$. Secondly, the determination of the second parent follows Eq. (27).

$$\boldsymbol{parent}_{\boldsymbol{2}}^{\boldsymbol{p,t}} = \begin{cases} \boldsymbol{soc}_{\boldsymbol{cr_5}}^{\boldsymbol{p,t}} & rand < c \\ \boldsymbol{alpha}^{\boldsymbol{p,t}} & otherwise \end{cases} \tag{27}$$

where $\boldsymbol{soc}_{\boldsymbol{cr_5}}^{\boldsymbol{p,t}}$ represents a random coyote different from the first parent in pack p. The pup are generated using Eq. (28).

$$pup_j^{p,t} = \begin{cases} parent_{1,j}^{p,t} & r_j < P_a \\ parent_{2,j}^{p,t} & r_j > 1 - P_a \\ R_j & otherwise \end{cases} \tag{28}$$

where R_j is a randomly generated variable within the j dimension of the search space, r_j represents a randomly generated number with a uniform distribution in the range of [0,1]. In this stage, the conventional COA algorithm's approach is also used to determine whether the pup can be retained in the population.

3.5. An example of optimization by OSCOA

This section provides optimization of a highly multimodal function called the Rastrigin function to demonstrate the operation process of the OSCOA population and the topology of packs. In this experiment, the population size N is set to 12 to clearly illustrate the positions of each coyote, and LU_c and UN_c are set to 3 and 6, respectively.

Fig. 3 displays the population distribution of the algorithm at different iteration numbers. In these figures, shapes of the same type represent members of the same pack, the black shape represents the ***alpha***, and the blue shape represents the newly born pup. To illustrate the pack topology, each coyote is connected to the ***alpha*** of its pack with a dotted line.

Firstly, it can be observed that in the initial stages of the algorithm, the coyotes are scattered randomly across the search

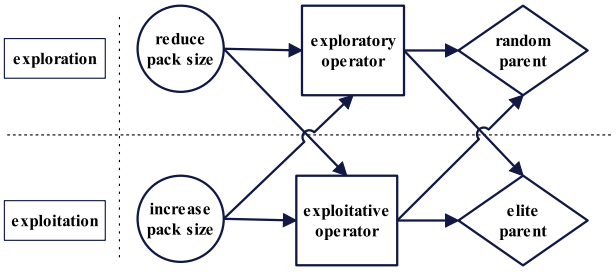


Fig. 4. Strategies combination model in OSCOA.

space and are evenly distributed into four packs. This facilitates extensive exploration of the search space by the population. During the 0th iteration, the square pack occupies a dominant region at [0, −1]. Moving to the 1st iteration, the diamond pack establishes another dominant region at [0, 1] through the birth of pups. As the iterations progress, around the 5th iteration, the population begins to explore both dominant regions. This tendency continues, and by the 10th and 15th iterations, the population notably converges towards the region [0, 1]. Subsequently, around the 20th iteration, a substantial portion of the coyotes clusters around [0, 1]. However, by the 25th iteration, OSCOA effectively breaks free from the local optimum, leveraging its robust exploration strategy, and successfully converges closer to the global optimum with the assistance of pups. Finally, in the 30th iteration, the entire population effectively converges near the global optimum.

3.6. Analysis of OSCOA

Overall, the OSCOA algorithm dynamically adjusts the pack size N_c, selects growth strategies, and determines pup parents based on the c value obtained from IOS. The essence of IOS is to estimate the complexity of the population coverage space. However, relying solely on the estimation from a limited number of individuals might lead to inaccuracies, especially in high-dimensional spaces [30].

To prevent the population from swiftly converging to local optima due to potentially erroneous IOS estimations, the OSCOA algorithm approaches the setting of Eq. (22) weights, namely w_3 and w_4, cautiously. Thus, while the Eq. (22) is more exploitation-oriented compared to the Eq. (19), the Eq. (22) still has a strong exploratory capacity in the early stages of the algorithm. And the OSCOA algorithm does not grant IOS exclusive control, the selection of strategies is governed by probabilities. The IOS only guides the strategy choice. This methodology also eliminates the need for cumbersome threshold settings.

In addition, the OSCOA algorithm introduces an innovative mechanism for strategy selection. In OSCOA algorithm, the selection of strategies in the three mechanisms is independent of each other. This approach significantly diversifies the search pattern of the population by combining different strategies. And the composition model of various strategies within OSCOA is illustrated in Fig. 4.

The Algorithm 4 gives the complete running steps of the OSCOA algorithm.

Algorithm 4: OSCOA algorithm

Input: the number of coyotes N; maximum number of function estimations MaxFEs;lower bound ***lb***; upper bound ***ub***;dimensions of search space D
Output: the best solution

```
Randomly initialize the initial population using Eq. (2);
Initialize the age, FEs and Np ;
Calculate the fitness of initial population using Eq. (3) ;
while FEs ≤ MaxFEs do
    Calculate c by Algorithm 2 ;
    Calculate Nc by Eq. (18) ;
    Population grouping and calculate Np using Algorithm 3;
    for i = 1 : Np do
        Calculate the capacity of the current pack Ncp ;
        Locate the best coyote alpha in the current packs ;
        Calculate the cultural tendency cult ;
        for j = 1 : Ncp do
            Upgrade coyotes by Eq. (25);
            Perform out-of-bounds detection ;
            Calculate the fitness of new coyote using Eq. (8) ;
            Perform greedy selection using Eq. (9) ;
        end
        Select parents of coyote using Eq. (27) ;
        Generate a pup using Eq. (28) ;
        Determine whether to let pup join the population ;
    end
    age = age + 1 ;
end
return the best solution ;
```

4. Complexity analysis

The complexity of an algorithm includes both time complexity and space complexity. Time complexity is a metric for measuring the execution time of an algorithm and directly affects its optimization efficiency. The space complexity refers to the amount of memory space required by an algorithm during its execution and affects the allocation of system resources.

Before analyzing the complexity, it is assumed that the population size is N, the dimensionality of the search space is D, and the population undergoes a maximum of G iterations.

4.1. Time complexity of COA

The main steps of the COA are listed below.

- The time complexity of initializing the population and estimating the fitness of initial population are $O(N \cdot D)$ and $O(N \cdot f(D))$, respectively.
- The time complexity of locating the best individual in the current pack is $O(N)$.
- The time complexity of the population evolution is $O(N \cdot D)$.
- The time complexity of estimating the fitness of new population is $O(N \cdot f(D))$.
- The time complexity of generating pups and estimating pups is $O(N_p \cdot D)$ and $O(N_p \cdot f(D))$.
- The time complexity of adding pups to the population is $O(N)$.
- The time complexity of updating the age of individuals is $O(N)$.
- The time complexity of individual migration is $O(1)$.

The time complexity of running the G generations through the above analysis is presented in Eq. (29)

$$\begin{aligned} &O(N \cdot (D + f(D)) + G \cdot (N + N \cdot D + N \cdot f(D) + N_p \cdot D \\ &+ N_p \cdot f(D) + 2N + 1)) \approx O(N \cdot (D + f(D)) + G \cdot (N(1 \\ &+ D + f(D) + 2) + N_p \cdot (D + f(D)))) \approx O(N \cdot (D + f(D)) \\ &+ G \cdot (N + N_p) \cdot (D + f(D))) \approx O(G \cdot N \cdot (D + f(D))) \end{aligned} \tag{29}$$

Table 2
The time complexity of algorithms in different dimensions.

Time	COA				OSCOA			
	10D	30D	50D	100D	10D	30D	50D	100D
$T0$	0.0169	0.0196	0.0183	0.0195	0.0169	0.0196	0.0183	0.0195
$T1$	0.3968	0.6863	0.9608	2.6701	0.3968	0.6863	0.9608	2.6701
$\widehat{T2}$	2.2192	2.6270	3.0510	6.1520	1.4555	1.8822	2.4245	4.5220
T	107.8343	99.0153	114.2186	178.5590	62.6450	61.0153	79.9836	94.9692

4.2. Time complexity of OSCOA

The main steps of the OSCOA are listed below.

- The time complexity of initializing the population and estimating the fitness of initial population are $O(N \cdot D)$ and $O(N \cdot f(D))$, respectively.
- The time complexity of calculating the c is $O(N \cdot (D + log_2 N)) \approx O(N \cdot log_2 N)$.
- The time complexity of locating the best individual in the current pack is $O(N)$.
- The time complexity of the population upgrade is $O(N \cdot D)$.
- The time complexity of estimating the fitness of new population is $O(N \cdot f(D))$.
- The time complexity of generating pups and estimating pups is $O(N_p \cdot D)$ and $O(N_p \cdot f(D))$.
- The time complexity of adding pups to the population is $O(N)$.
- The time complexity of updating the age of individuals is $O(N)$.

From the analysis presented above, it is evident that the alteration in time complexity between OSCOA and COA solely takes place in the calculation of IOS. Hence, the time complexity of OSCOA can be succinctly expressed using Eq. (30).

$$\begin{aligned} &O(G \cdot N \cdot (D + f(D) + D + log_2 N)) \\ &\approx O(G \cdot N \cdot (D + f(D) + log_2 N)) \end{aligned} \tag{30}$$

4.3. Time complexity testing

To visually validate the time complexity of the OSCOA algorithm, the time complexity testing method proposed in the CEC2017 competition is employed. Firstly, the **Program A** written in the programming language used for the algorithm is executed to obtain the execution time $T0$.

Program A:

$x = 0.55$
for $i = 1 : 1000000$
$x = x + x; x = x/2; x = x * x; x = sqrt(x);$
$x = \log(x); x = \exp(x); x = x/(x + 2);$
end

The execution time $T1$ for running F18 from CEC2017 200,000 times is calculated; The complete execution time $T2$ of algorithms is obtained by evaluating fitness on F18 for 200,000 times, and the process is independently repeated five times to calculate the mean value of $T2$, denoted as $\widehat{T2} = \text{Mean}(T2)$. For detailed procedures, please refer to [33]. Finally, the time complexity of the algorithm is evaluated using Eq. (31).

$$T = \frac{\widehat{T2} - T1}{T0} \tag{31}$$

Table 2 presents the calculation results of the time complexity for COA and OSCOA. From the table, it can be observed that OSCOA outperforms the conventional COA algorithm in terms of T value. This is attributed to the utilization of matrix computations in the code of the OSCOA algorithm, which enhances the efficiency of the algorithm's execution.

4.4. Space complexity

The space complexity is correlated with the number of population N and the problem dimension D, thus, it primarily is influenced by two factors Firstly, the storage space required for the population is $N \cdot D$. Secondly, the storage space required for the fitness values is N. Therefore, the space complexity of OSCOA is $O(N \cdot D + N) \approx O(N \cdot D)$.

5. Experiment and analysis

The experimental environment is as follows: 64-bit Windows operating system; programming software MATLAB 2020b; Inter(R) Core(TM) i7-10700 CPU, 2.90 GHz main frequency, 16G memory.

In the subsequent experiments, the metrics employed to evaluate the global search performance include the mean (*mean*) and standard deviation (*std*). The *mean* assesses convergence accuracy, while the *std* evaluates algorithm stability. A smaller value of *mean* signifies superior algorithmic solution performance. Similarly, a reduced value of *std* implies enhanced algorithm stability. In the tables presenting the experimental results, the *mean* and *std* are displayed in the form of *mean* ± *std*. In addition, two non-parametric tests are introduced into the experiments.

The first test is the Wilcoxon rank-sum test [34], used to determine significant differences between two sets of results. Specifically the Wilcoxon test assign a p-rank to each compared algorithm at a confidence level of 0.05, signifying its relative difference concerning the OSCOA algorithm. The p-rank has three states: win (w), loss (l) and tie (t). 'w' indicates that the algorithm's optimization result is significantly better than OSCOA, marked by '+'; 'l' indicates a significantly worse result, marked by '−'; 't' indicates similarity or no statistical difference compared to OSCOA, marked by '='.

The second test is the Friedman test [35], a non-parametric method for two-way analysis of variance. The Friedman test ranks optimization results for the same function, expressed using f-rank.

The notation 'w/l/t' represents the counts of '+', '−', and '=' obtained by algorithms. The 'overall f-rank' is the sum of f-ranks across all test functions.

To be fair, each algorithm is run 30 times independently on each test function, and the MaxFEs is generated by Eq. (32).

$$MaxFEs = D \cdot 10^4 \tag{32}$$

The paper employs three sets of real-parameter single-objective optimization problems presented by the IEEE Congress on Evolutionary Computation (IEEE CEC) in 2014, 2017, and 2022, respectively, as its test functions. The CEC2014 and CEC2017, being classical benchmark suites, have been extensively used for algorithm evaluation and analysis. Concurrently, CEC2022 is the most recent benchmark suite used to test the performance of OSCOA algorithm on the latest optimization problems.

In the following experiments, the two parameters of OSCOA, LN_c and UN_c, take the values of 5 and 20, respectively, and the population size N is 100, which will be analyzed in detail in Section 7.

Table 3
The comparison results of OSCOA with COA and incomplete algorithms on CEC2017.

Item	COA	OSCOA-1	OSCOA-2	OSCOA-3	OSCOA
Overall f-rank	139.2	110.2	84.6	62.6	37.8
w/l/t	0/27/2	1/27/1	0/27/2	0/18/11	

Table 4
Parameter settings for different algorithms.

Algorithms	Parameter settings
COA	$N_p = 10$; $N_c = 10$
BTLBO	$\alpha = \frac{FEs}{MaxFEs}$; $MaxnF = \frac{MaxFEs}{20\times N}\exp\left(\left(\frac{1}{MaxFEs}\right)^{\left(\frac{N}{100}\right)^2}\right)$
ADE	$c_{CR} = 0.05$; $c_F = 0.1$; population is divided into 10 groups
DEET	$\mu = 0.85$; $p_{min} = 0.1$; $p = 0.05$; $c_1 = 0.1$; $Q = 5D$
CCOA	$\gamma = 0.3$; $\psi = 0.3$; $P_n = 0.8$; $N_p = 10$; $N_c = 10$
FDB-SOS	$w = 0.5$; $BF_1, BF_2 = round(1 + rand)$
SHADE	$p_{min} = 2/N$, $H = 100$
OSCOA	$UN_c = 20$; $LN_c = 5$

5.1. Comparison of three mechanisms

In order to test the influence and contribution of three mechanisms to OSCOA algorithm, each mechanism is tested separately and independently. For labeling convenience, the algorithms corresponding to the "dynamic pack management mechanism", "multiple growth strategy mechanism", and "parent selection mechanism" are named as OSCOA-1, OSCOA-2, and OSCOA-3, respectively.

To ensure the fairness and validity of the experimental comparisons, each of the algorithms is executed independently 30 times on the 30D CEC2017 benchmark suite. The statistical outcomes are presented in Table 3.

It can be observed in Table 3 that the overall f-rank scores for COA, OSCOA-1, OSCOA-2, and OSCOA-3 are 139.2, 110.2, 84.6, and 62.6, respectively. This suggests that the "parent selection mechanism" employed in OSCOA-3 has the most significant impact on the overall performance enhancement of the OSCOA algorithm. It is followed by the "multiple growth strategy mechanism" and the "dynamic pack management mechanism" in terms of their contributions.

The w/l/t indicates that OSCOA surpasses COA, OSCOA-1, OSCOA-2, and OSCOA-3 on 27, 27, 27, and 18 functions, respectively. Impressively, OSCOA achieves the highest overall f-rank score of 37.8. These results firmly validate the efficacy of the incorporated strategies within the OSCOA algorithm, as highlighted through the aforementioned analysis.

5.2. Algorithm comparison experiments

To evaluate the optimization performance of OSCOA, a comparison is conducted with several outstanding algorithms, including Cultural Coyote Optimization Algorithm (CCOA) [13], Balanced Teaching-Learning-Based Optimization (BTLBO) [36], Adaptive Differential Evolution (ADE) [29], Differential Evolution with an Evolutionary State Estimation and a Two-level Selection Mechanism (DEET) [30], Fitness-Distance Balance Symbiotic Organism Search (FDB-SOS), and Success-History based Adaptive Differential Evolution (SHADE) [37], in addition to the conventional COA.

These algorithms are chosen for comparison experiments for the following reasons: the CCOA is a variant of the same COA, BTLBO is a recently proposed efficient variant of Teaching-Learning-Based Optimization (TLBO) [38] which is committed to a balance of exploration and exploitation, both the ADE and the DEET introduce the population optimization state estimation mechanism, and the FDB-SOS incorporates a recently proposed efficient individual selection technique Fitness-Distance Balance(FDB), and SHADE algorithm is well known as an efficient DE variant.

The parameter settings of algorithms involved in the experiments follow original papers and are presented in Table 4.

5.2.1. CEC2014 benchmark suite

The CEC2014 benchmark suite includes 30 optimization functions. The functions are divided into four categories including three unimodal functions (F1–F3), thirteen simple multimodal functions (F4–F16), six hybrid functions (F17–F22) and eight composition functions (F23–F30).

Table 5 shows the test results for the 30D CEC2014 benchmark suite. Regarding the unimodal functions, OSCOA's f-rank ranks second only to DEET and SHADE on F1. Then, on F2 and F3, OSCOA's performance is not as strong as that of BTLBO, DEET, and SHADE. Nevertheless, its *mean* and *std* values indicate that its optimization results are very close to the optimal solutions. Moving on to the simple multimodal functions, although OSCOA performed overall worse than SHADE and DEET, it still achieved first place on F12. Next, for the hybrid functions, OSCOA takes the lead in the Friedman test on four out of the six hybrid functions and secures the second place on F21. Finally, for composition functions, OSCOA performs well on F25, F26 and F30.

Analyzing the statistical results, OSCOA's overall f-rank scored 86.1 points. From w/l/t, it can be seen that OSCOA significantly outperforms COA on 28 test functions, BTLBO on 18 test functions, ADE on 20 test functions, DEET on 7 test functions, CCOA on 23 test functions, FDB-SOS on 19 test functions and SHADE on 6 test functions.

To analyze the performance of OSCOA in a high-dimensional search space, Table 6 shows the experimental results on 100D CEC2014. From the statistical results it can be observed that OSCOA wins the second place in the Friedman test with 89.5. Additionally, the Wilcoxon rank-sum test indicates that OSCOA significantly outperforms COA, BTLBO, ADE, DEET, CCOA, FDB-SOS and SHADE on 29, 12, 19, 11, 25, 19 and 9 test functions, respectively. In summary OSCOA can still be highly competitive on the higher dimensional CEC2014 benchmark suite.

5.2.2. CEC2017 benchmark suite

The CEC2017 benchmark suite also encompasses four categories of test functions: unimodal functions (F1, F3), simple multimodal functions (F4–F10), hybrid functions (F11–F20), and composition functions (F21–F30).

Compared to CEC2014, the count of simple multimodal functions decreases from 16 to 7, while the number of hybrid functions increases from 6 to 10, and the quantity of composition functions also rises from 8 to 10. Consequently, overall, CEC2017 poses a more challenging optimization environment than CEC2014. Table 7 shows the test results of the algorithms on the 30D CEC2017 benchmark suite.

In the assessment of the unimodal functions, it can be observed that OSCOA performs better in optimizing F3 compared to F1. Moving on to the analysis of the simple multimodal functions, OSCOA secures the second position on F6, F9, and F10. Then the OSCOA achieves the first place in the Friedman test for seven out of ten hybrid functions, specifically F11–F15 and F18–F19. Regarding the ten composition functions, OSCOA emerges as the leader in the Friedman test for F25, F27, and F30, it also secures the third place on F21, F23, F24, F26, and F29.

The statistical outcomes highlight that OSCOA secures the second position in the Friedman test with a score of 79.3. Furthermore, as indicated by the Wilcoxon rank-sum test, the OSCOA

Table 5
The comparison results of OSCOA with other algorithms on CEC2014 with 30 Dimensions.

Function	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
F1	1.15E+08 ± 1.88E+07 8.0 −	8.24E+04 ± 9.58E+04 4.6 −	6.58E+05 ± 1.89E+06 4.4 −	2.90E+03 ± 3.04E+03 2.1 =	7.64E+06 ± 3.41E+06 7.0 −	7.74E+05 ± 5.72E+05 5.8 −	1.80E+03 ± 2.85E+03 1.6 +	6.89E+03 ± 5.44E+03 2.5
F2	1.96E+10 ± 2.13E+09 8.0 −	3.05E−12 ± 5.48E−12 3.6 +	4.01E+04 ± 1.08E+05 3.7 =	0.00E+00 ± 0.00E+00 1.0 +	6.83E+06 ± 5.16E+06 7.0 −	1.70E+01 ± 1.64E+01 5.7 −	2.84E−14 ± 0.00E+00 2.3 +	1.45E−04 ± 2.11E−04 4.5
F3	3.18E+04 ± 3.75E+03 8.0 −	3.35E−10 ± 1.02E−09 2.2 +	2.37E+03 ± 4.52E+03 5.8 −	1.08E−04 ± 1.27E−04 3.5 −	3.37E+03 ± 6.90E+02 6.8 −	5.76E+01 ± 4.10E+01 5.4 −	5.12E−14 ± 1.80E−14 1.2 +	1.13E−08 ± 7.78E−09 3.2
F4	2.10E+03 ± 3.06E+02 8.0 −	2.07E+01 ± 3.21E+01 4.0 =	7.57E+01 ± 5.38E+01 5.3 −	7.96E−14 ± 4.79E−14 1.2 +	1.93E+02 ± 3.19E+01 7.0 −	4.10E+01 ± 3.30E+01 5.0 −	1.93E−13 ± 1.08E−13 1.9 +	1.17E+00 ± 1.23E+00 3.7
F5	2.09E+01 ± 5.84E−02 6.8 −	2.08E+01 ± 6.38E−02 6.6 −	2.06E+01 ± 4.86E−02 4.3 −	2.02E+01 ± 6.67E−02 2.3 =	2.09E+01 ± 3.69E−02 7.6 −	2.06E+01 ± 7.43E−02 4.7 −	2.01E+01 ± 1.05E−02 1.0 +	2.02E+01 ± 5.47E−02 2.7
F6	3.36E+01 ± 5.88E−01 8.0 −	1.11E+01 ± 2.70E+00 4.4 −	2.10E+01 ± 2.32E+00 6.4 −	4.47E+00 ± 2.30E+00 2.4 =	2.27E+01 ± 2.55E+00 6.4 −	7.27E+00 ± 5.39E+00 2.9 =	5.68E+00 ± 1.69E+00 2.4 =	6.85E+00 ± 3.81E+00 2.9
F7	1.88E+02 ± 1.60E+01 8.0 −	5.66E−03 ± 7.78E−03 4.2 =	1.54E−01 ± 3.46E−01 5.5 −	0.00E+00 ± 0.00E+00 1.0 +	1.07E+00 ± 6.68E−02 6.9 −	1.01E−02 ± 1.61E−02 4.2 =	4.43E−03 ± 5.90E−03 2.8 =	2.46E−03 ± 4.03E−03 3.4
F8	2.00E+02 ± 1.81E+01 8.0 −	1.21E+01 ± 4.43E+00 4.8 −	4.63E+00 ± 8.97E+00 4.1 −	0.00E+00 ± 0.00E+00 1.1 +	6.47E+01 ± 4.96E+00 6.3 −	7.29E+01 ± 1.18E+01 6.7 −	1.71E−13 ± 1.23E−13 2.3 =	1.23E−01 ± 3.09E−01 2.8
F9	2.90E+02 ± 1.49E+01 8.0 −	7.97E+01 ± 1.52E+01 4.3 −	9.72E+01 ± 2.65E+01 4.6 −	2.80E+01 ± 5.92E+00 2.4 +	1.75E+02 ± 1.38E+01 6.9 −	1.23E+02 ± 1.67E+01 5.8 −	2.12E+01 ± 3.21E+00 1.2 +	4.65E+01 ± 1.26E+01 2.8
F10	4.12E+03 ± 2.15E+02 8.0 −	2.76E+02 ± 7.21E+01 4.9 −	1.33E+02 ± 8.38E+01 3.9 −	1.48E+00 ± 2.01E+00 2.6 =	9.67E+02 ± 2.39E+02 5.9 −	1.64E+03 ± 1.61E+02 6.9 −	1.25E−02 ± 1.46E−02 1.0 +	2.59E+00 ± 2.08E+00 2.6
F11	6.48E+03 ± 2.67E+02 7.8 −	2.80E+03 ± 8.77E+02 3.7 =	4.62E+03 ± 4.17E+02 6.0 −	1.81E+03 ± 2.33E+02 2.3 +	5.67E+03 ± 7.84E+02 7.1 −	4.17E+03 ± 4.75E+02 4.8 −	1.42E+03 ± 3.69E+02 1.1 +	2.44E+03 ± 3.47E+02 3.5
F12	1.73E+00 ± 3.31E−01 6.9 −	1.74E+00 ± 2.93E−01 6.6 −	1.02E+00 ± 1.37E−01 5.0 −	1.43E−01 ± 5.46E−02 2.5 −	2.06E+00 ± 4.30E−01 7.4 −	6.57E−01 ± 1.18E−01 4.1 −	1.39E−01 ± 2.59E−02 2.2 −	8.64E−02 ± 4.52E−02 1.3
F13	3.41E+00 ± 3.46E−01 8.0 −	3.77E−01 ± 1.16E−01 5.5 −	3.44E−01 ± 1.13E−01 4.9 =	1.16E−01 ± 3.86E−02 1.3 +	3.30E−01 ± 6.32E−02 4.6 =	3.43E−01 ± 4.37E−02 5.2 −	2.48E−01 ± 7.61E−02 3.0 =	2.82E−01 ± 5.43E−02 3.8
F14	5.39E+01 ± 7.55E+00 8.0 −	3.08E−01 ± 1.84E−01 3.5 =	2.77E−01 ± 3.88E−02 4.6 =	3.09E−01 ± 8.45E−02 5.4 =	1.91E−01 ± 3.08E−02 1.4 +	4.13E−01 ± 2.17E−01 5.5 =	2.39E−01 ± 4.65E−02 3.0 =	3.26E−01 ± 1.43E−01 4.7
F15	5.00E+04 ± 1.16E+04 8.0 −	5.05E+00 ± 1.19E+00 3.6 −	1.01E+01 ± 2.05E+00 5.0 −	2.80E+00 ± 5.45E−01 1.7 =	3.73E+01 ± 6.44E+00 7.0 −	1.58E+01 ± 8.63E−01 6.0 −	2.91E+00 ± 5.86E−01 2.1 =	3.54E+00 ± 1.39E+00 2.6
F16	1.28E+01 ± 1.19E−01 8.0 −	1.11E+01 ± 6.65E−01 4.9 −	1.18E+01 ± 1.71E−01 6.6 −	8.71E+00 ± 7.59E−01 1.6 =	1.16E+01 ± 2.41E−01 5.8 −	1.09E+01 ± 5.55E−01 4.7 −	9.28E+00 ± 5.15E−01 1.7 =	9.74E+00 ± 4.80E−01 2.7
F17	1.24E+06 ± 3.82E+05 7.9 −	2.96E+03 ± 1.12E+03 4.9 −	7.72E+02 ± 2.88E+02 2.1 =	9.44E+02 ± 3.81E+02 2.8 −	4.28E+05 ± 1.95E+05 7.1 −	1.75E+05 ± 2.00E+05 6.2 −	1.42E+03 ± 4.36E+02 3.9 −	7.45E+02 ± 2.86E+02 1.4
F18	5.96E+05 ± 3.36E+05 8.0 −	9.99E+02 ± 1.41E+03 5.0 −	2.96E+01 ± 2.15E+01 1.9 =	2.22E+03 ± 6.62E+03 4.1 −	9.06E+02 ± 7.08E+02 5.8 −	5.61E+03 ± 7.27E+03 6.2 −	1.59E+02 ± 6.24E+01 3.7 −	2.78E+01 ± 8.71E+00 1.6
F19	4.63E+01 ± 4.50E+00 7.8 −	1.97E+01 ± 2.46E+01 5.8 −	7.88E+00 ± 1.49E+00 5.1 −	4.37E+00 ± 7.98E−01 2.5 =	1.15E+01 ± 1.99E+00 6.7 −	7.46E+00 ± 1.11E+00 4.5 −	4.92E+00 ± 1.07E+00 2.5 =	4.34E+00 ± 8.77E−01 1.3
F20	9.87E+03 ± 5.23E+03 8.0 −	1.10E+02 ± 5.91E+01 3.6 −	3.51E+01 ± 1.27E+01 2.4 −	1.01E+03 ± 1.03E+03 6.0 −	1.89E+03 ± 1.02E+03 6.6 −	4.80E+02 ± 4.34E+02 5.1 −	7.21E+01 ± 4.68E+01 2.8 −	2.35E+01 ± 1.19E+01 1.3
F21	8.59E+04 ± 3.54E+04 7.3 −	1.28E+03 ± 4.34E+02 4.4 −	2.56E+04 ± 7.91E+04 3.6 =	2.20E+04 ± 5.31E+04 3.4 −	6.17E+04 ± 3.20E+04 6.4 −	7.64E+04 ± 8.10E+04 6.5 −	4.90E+02 ± 1.81E+02 2.3 =	5.32E+02 ± 1.21E+02 2.4

(*continued on next page*)

Table 5 (*continued*).

F22	3.80E+02 ± 1.04E+02 6.2 =	2.90E+02 ± 1.14E+02 4.5 =	4.16E+02 ± 1.02E+02 6.5 =	1.33E+02 ± 5.93E+01 2.2 =	3.11E+02 ± 1.27E+02 5.1 =	2.54E+02 ± 1.08E+02 4.5 =	1.24E+02 ± 9.33E+01 2.0 =	3.32E+02 ± 1.89E+02 5.0
F23	3.73E+02 ± 8.62E+00 8.0 −	2.00E+02 ± 1.92E−13 1.0 +	3.16E+02 ± 1.29E+00 4.3 =	3.15E+02 ± 0.00E+00 3.4 =	3.17E+02 ± 1.35E+00 6.9 −	3.15E+02 ± 3.71E−13 5.1 =	3.15E+02 ± 0.00E+00 3.4 =	3.15E+02 ± 3.18E−13 4.1
F24	3.12E+02 ± 1.01E+01 8.0 −	2.00E+02 ± 2.35E−02 2.0 +	2.34E+02 ± 9.02E+00 5.9 −	2.26E+02 ± 3.90E+00 3.9 =	2.27E+02 ± 1.34E+00 5.1 −	2.00E+02 ± 1.41E−03 1.0 +	2.35E+02 ± 6.46E+00 6.1 −	2.25E+02 ± 1.78E+00 3.7
F25	2.19E+02 ± 2.28E+00 8.0 −	2.00E+02 ± 2.35E−13 1.7 +	2.04E+02 ± 3.43E+00 4.6 −	2.04E+02 ± 1.57E+00 5.2 =	2.05E+02 ± 2.67E+00 5.7 =	2.03E+02 ± 3.61E+00 3.9 =	2.04E+02 ± 8.09E−01 5.2 −	2.03E+02 ± 3.81E−01 1.8
F26	1.03E+02 ± 2.68E−01 7.8 −	1.00E+02 ± 5.21E−02 6.0 −	1.01E+02 ± 6.58E−01 5.9 −	1.00E+02 ± 5.13E−02 1.3 =	1.00E+02 ± 5.84E−02 4.2 =	1.00E+02 ± 4.03E−02 5.1 −	1.20E+02 ± 4.21E+01 4.1 =	1.00E+02 ± 6.48E−02 1.9
F27	7.76E+02 ± 2.16E+02 8.0 −	2.51E+02 ± 6.06E+01 1.4 +	4.65E+02 ± 5.69E+01 6.5 −	3.17E+02 ± 3.62E+01 2.1 +	4.18E+02 ± 1.03E+01 5.7 =	4.14E+02 ± 5.06E+01 4.9 =	4.16E+02 ± 5.23E+01 4.0 =	4.07E+02 ± 4.74E+01 3.2
F28	1.06E+03 ± 2.69E+01 7.2 −	2.00E+02 ± 2.04E−01 1.1 +	1.02E+03 ± 1.03E+02 6.6 −	8.01E+02 ± 1.39E+01 2.6 +	1.05E+03 ± 4.22E+01 6.9 −	8.67E+02 ± 4.89E+01 3.8 =	9.55E+02 ± 2.79E+02 4.9 =	8.49E+02 ± 3.34E+01 3.0
F29	7.87E+04 ± 1.54E+04 7.4 +	2.52E+06 ± 4.06E+06 5.7 −	1.28E+05 ± 2.70E+05 4.5 =	1.13E+03 ± 6.74E+02 2.8 +	2.83E+03 ± 1.24E+03 3.2 +	8.68E+05 ± 2.74E+06 5.8 =	7.33E+02 ± 7.82E+01 2.3 +	8.33E+05 ± 2.22E+04 4.3
F30	1.57E+04 ± 3.37E+03 7.9 −	3.38E+03 ± 1.24E+03 4.8 −	6.85E+03 ± 8.21E+03 5.1 =	3.06E+03 ± 1.10E+03 4.7 −	5.65E+03 ± 3.25E+03 5.9 −	2.34E+03 ± 8.42E+02 2.9 =	2.52E+03 ± 1.36E+03 3.1 =	1.97E+03 ± 8.54E+02 1.5
Overall f-rank w/l/t	233.0 1/28/1	123.2 7/18/5	145.0 0/20/10	81.1 10/7/13	182.3 2/23/5	148.7 1/19/10	80.7 9/6/15	86.1

Table 6
The comparison results of OSCOA with other algorithms on CEC2014 with 100 Dimensions.

Function	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
F1	2.17E+09 ± 1.80E+08 8.0 −	2.05E+06 ± 1.03E+06 3.1 +	4.34E+08 ± 5.07E+08 6.5 −	2.67E+05 ± 1.66E+05 1.6 +	9.52E+07 ± 1.00E+07 6.3 −	1.84E+07 ± 1.66E+07 4.8 −	4.53E+05 ± 4.40E+05 2.0 +	3.29E+06 ± 8.03E+05 3.8
F2	2.54E+11 ± 2.12E+10 8.0 −	4.21E+02 ± 8.33E+02 3.5 +	3.86E+04 ± 9.99E+04 4.2 =	2.07E−10 ± 3.34E−10 1.5 +	7.54E+09 ± 2.64E+09 6.9 −	1.01E+04 ± 6.21E+03 5.0 =	2.26E−10 ± 3.86E−10 1.7 +	1.36E+04 ± 1.09E+04 4.9
F3	4.23E+05 ± 2.48E+04 7.9 −	1.14E+01 ± 1.45E+01 2.3 +	3.60E+05 ± 4.97E+04 7.1 −	2.51E+04 ± 2.41E+04 4.2 =	3.72E+04 ± 1.07E+04 5.6 −	3.79E+03 ± 2.08E+03 4.3 −	6.70E−02 ± 1.47E−01 1.3 +	6.67E+02 ± 5.82E+02 3.1
F4	5.32E+04 ± 3.46E+03 8.0 −	2.82E+02 ± 5.14E+01 5.5 =	1.97E+02 ± 5.25E+01 3.6 =	1.30E+02 ± 2.51E+01 1.3 +	1.05E+03 ± 4.95E+02 7.1 −	2.11E+02 ± 3.24E+01 4.0 +	1.57E+02 ± 3.24E+01 2.4 +	2.50E+02 ± 6.27E+01 4.3
F5	2.13E+01 ± 3.88E−02 7.3 −	2.12E+01 ± 2.24E−02 6.2 −	2.10E+01 ± 1.68E−02 4.5 −	2.08E+01 ± 1.19E−01 3.1 −	2.13E+01 ± 2.99E−02 7.5 −	2.10E+01 ± 4.51E−02 4.4 −	2.01E+01 ± 1.59E−02 1.3 =	2.04E+01 ± 6.13E−02 1.6
F6	1.47E+02 ± 1.93E+00 8.0 −	9.02E+01 ± 8.00E+00 4.9 −	1.08E+02 ± 7.68E+00 6.3 −	4.47E+01 ± 8.98E+00 1.0 +	1.11E+02 ± 1.15E+01 6.6 −	7.32E+01 ± 4.37E+00 3.1 −	7.68E+01 ± 4.67E+00 3.7 −	6.33E+01 ± 8.44E+00 2.4
F7	2.24E+03 ± 1.52E+02 8.0 −	1.36E−02 ± 2.23E−02 5.1 =	8.95E−02 ± 2.00E−01 3.8 =	1.73E−03 ± 3.68E−03 1.5 +	6.11E+01 ± 3.31E+01 7.0 −	5.39E−03 ± 1.46E−02 3.6 −	4.43E−03 ± 7.77E−03 2.7 =	3.24E−03 ± 5.46E−03 4.4
F8	1.28E+03 ± 4.16E+01 8.0 −	2.10E+02 ± 3.11E+01 4.9 −	1.39E+02 ± 2.27E+02 3.5 −	1.25E+02 ± 2.07E+01 3.8 −	5.02E+02 ± 2.23E+01 6.9 −	3.97E+02 ± 4.49E+01 5.9 −	1.36E−12 ± 1.20E−13 1.0 +	3.79E−01 ± 5.12E−01 2.0
F9	1.58E+03 ± 5.21E+01 8.0 −	5.27E+02 ± 4.92E+01 4.7 −	4.17E+02 ± 8.44E+01 4.2 −	2.03E+02 ± 2.58E+01 1.1 +	9.27E+02 ± 5.39E+01 7.0 −	7.48E+02 ± 5.95E+01 6.0 −	2.49E+02 ± 3.59E+01 2.0 +	3.02E+02 ± 6.47E+01 3.0
F10	2.52E+04 ± 4.06E+02 8.0 −	5.00E+03 ± 1.70E+03 4.5 −	2.74E+03 ± 9.74E+02 3.2 −	4.30E+03 ± 7.09E+02 4.4 −	8.60E+03 ± 1.03E+03 5.9 −	1.11E+04 ± 6.69E+02 7.0 −	7.50E−03 ± 3.95E−03 1.0 +	6.36E+00 ± 9.33E−01 2.0
F11	2.92E+04 ± 1.86E+02 8.0 −	1.41E+04 ± 1.19E+03 2.7 =	2.19E+04 ± 9.03E+02 5.6 −	1.63E+04 ± 2.17E+03 3.8 −	2.58E+04 ± 2.21E+03 6.9 −	2.16E+04 ± 1.00E+03 5.4 −	1.02E+04 ± 5.29E+02 1.0 +	1.35E+04 ± 1.17E+03 2.6
F12	3.72E+00 ± 1.10E−01 7.1 −	2.88E+00 ± 2.13E−01 6.0 −	1.53E+00 ± 1.49E−01 4.9 −	6.19E−01 ± 1.97E−01 3.1 −	4.02E+00 ± 1.61E−01 7.9 −	1.37E+00 ± 2.49E−01 4.0 −	2.16E−01 ± 2.13E−02 1.6 =	2.07E−01 ± 6.07E−02 1.4
F13	8.13E+00 ± 2.69E−01 8.0 −	6.35E−01 ± 5.49E−02 5.9 −	4.98E−01 ± 5.23E−02 3.2 =	3.77E−01 ± 4.48E−02 1.3 +	6.16E−01 ± 6.63E−02 5.5 −	6.24E−01 ± 9.46E−02 5.4 −	5.24E−01 ± 7.80E−02 3.7 =	4.71E−01 ± 8.21E−02 2.9
F14	6.60E+02 ± 4.68E+01 8.0 −	4.51E−01 ± 2.23E−01 5.0 −	3.24E−01 ± 2.85E−02 3.0 =	4.08E−01 ± 1.66E−01 4.6 =	3.38E−01 ± 3.55E−02 4.2 =	3.83E−01 ± 1.64E−01 4.7 =	3.88E−01 ± 1.80E−01 4.0 =	3.18E−01 ± 2.49E−02 2.3
F15	2.18E+07 ± 4.17E+06 8.0 −	4.92E+01 ± 7.30E+00 2.6 =	1.12E+02 ± 4.27E+01 5.1 −	2.67E+01 ± 5.17E+00 1.1 +	1.46E+03 ± 6.08E+02 7.0 −	1.09E+02 ± 1.53E+01 5.7 −	7.31E+01 ± 1.41E+01 4.1 −	4.42E+01 ± 1.10E+01 2.4
F16	4.67E+01 ± 1.49E−01 8.0 −	4.32E+01 ± 1.13E+00 4.8 −	4.37E+01 ± 3.71E−01 5.4 −	4.18E+01 ± 8.35E−01 2.9 −	4.56E+01 ± 2.97E−01 7.0 −	4.32E+01 ± 7.34E−01 4.6 −	4.01E+01 ± 5.00E−01 1.3 =	4.07E+01 ± 7.66E−01 1.8
F17	1.13E+08 ± 2.27E+07 8.0 −	1.45E+05 ± 7.19E+04 3.6 =	8.49E+06 ± 1.27E+07 5.9 −	5.07E+04 ± 4.10E+04 2.3 =	4.28E+06 ± 1.01E+06 6.5 −	2.57E+06 ± 1.77E+06 5.5 −	2.05E+04 ± 1.32E+04 1.1 +	8.81E+04 ± 5.63E+04 3.1
F18	8.40E+09 ± 1.36E+09 8.0 −	3.88E+03 ± 2.73E+03 5.3 =	1.50E+04 ± 4.32E+04 4.1 =	1.95E+03 ± 2.37E+03 3.8 =	1.20E+03 ± 3.89E+02 4.2 =	2.81E+03 ± 3.11E+03 4.3 =	4.93E+02 ± 1.45E+02 2.1 =	2.36E+03 ± 2.24E+03 4.2
F19	1.39E+03 ± 1.07E+02 8.0 −	1.15E+02 ± 2.03E+01 4.1 −	1.14E+02 ± 2.63E+01 4.2 −	9.73E+01 ± 1.45E+01 2.8 =	2.09E+02 ± 2.88E+01 7.1 −	1.24E+02 ± 2.41E+01 4.4 −	1.03E+02 ± 3.96E+01 4.1 −	8.88E+01 ± 1.91E+01 1.6
F20	4.50E+05 ± 8.72E+04 8.0 −	6.87E+02 ± 1.35E+02 2.8 −	8.47E+04 ± 9.41E+04 6.6 −	2.75E+04 ± 4.22E+04 4.7 −	1.43E+04 ± 6.00E+03 5.7 −	1.80E+03 ± 9.92E+02 4.4 −	8.38E+02 ± 5.89E+02 2.5 =	4.65E+02 ± 1.36E+02 1.3
F21	4.18E+07 ± 3.37E+06 8.0 −	6.13E+04 ± 3.51E+04 3.3 =	4.69E+06 ± 6.97E+06 6.0 −	5.64E+04 ± 1.18E+05 2.3 +	3.04E+06 ± 1.27E+06 6.6 −	9.86E+05 ± 3.81E+05 5.4 −	5.15E+03 ± 3.03E+03 1.5 +	6.64E+04 ± 4.87E+04 3.2
F22	5.28E+03 ± 2.32E+02 8.0 −	1.88E+03 ± 4.70E+02 3.2 =	2.77E+03 ± 5.43E+02 5.9 =	1.73E+03 ± 7.44E+02 2.7 =	2.52E+03 ± 3.63E+02 5.0 =	2.27E+03 ± 6.06E+02 4.7 =	1.50E+03 ± 3.09E+02 2.3 +	2.37E+03 ± 8.28E+02 3.9
F23	1.62E+03 ± 1.11E+02 8.0 −	2.00E+02 ± 1.92E−13 1.0 +	3.48E+02 ± 1.19E−10 3.7 +	3.48E+02 ± 0.00E+00 2.0 +	3.89E+02 ± 4.14E+01 7.6 −	3.48E+02 ± 4.64E−08 3.8 +	3.48E+02 ± 6.82E−03 5.9 −	3.48E+02 ± 1.43E−06 4.1

(*continued on next page*)

Table 6 (*continued*).

F24	1.05E+03 ± 3.53E+01 8.0 −	2.00E+02 ± 6.28E−02 1.2 +	4.04E+02 ± 2.49E+01 5.4 −	4.09E+02 ± 8.50E+00 5.6 −	3.73E+02 ± 1.26E+01 4.0 −	2.00E+02 ± 6.67E−04 1.8 +	4.30E+02 ± 5.45E+00 6.7 −	3.57E+02 ± 1.07E+01 3.3
F25	4.84E+02 ± 1.69E+01 8.0 −	2.00E+02 ± 0.00E+00 1.4 +	3.39E+02 ± 6.99E+01 6.0 −	2.87E+02 ± 1.35E+01 5.5 −	2.32E+02 ± 5.18E+00 3.1 +	2.00E+02 ± 2.20E−13 1.7 +	3.01E+02 ± 1.31E+01 6.1 −	2.70E+02 ± 8.24E+00 4.3
F26	1.57E+02 ± 1.54E+02 2.8 =	2.00E+02 ± 0.00E+00 3.7 =	1.29E+02 ± 4.47E+01 2.9 =	2.00E+02 ± 8.89E−03 5.7 =	2.01E+02 ± 1.97E−01 7.3 =	2.00E+02 ± 1.92E−13 3.9 =	2.00E+02 ± 2.33E−02 5.9 =	2.39E+02 ± 1.80E+02 3.8
F27	4.03E+03 ± 2.27E+01 8.0 −	2.00E+02 ± 2.58E−13 1.1 +	2.92E+03 ± 2.52E+02 6.9 −	1.44E+03 ± 1.12E+02 2.2 +	2.53E+03 ± 2.19E+02 6.1 −	2.31E+03 ± 1.14E+02 4.9 −	2.18E+03 ± 1.86E+02 4.5 −	1.65E+03 ± 1.79E+02 2.6
F28	3.35E+03 ± 3.78E+02 4.3 −	2.00E+02 ± 3.07E−13 1.0 +	3.60E+03 ± 7.38E+02 4.6 −	3.03E+03 ± 5.42E+02 4.1 −	5.48E+03 ± 9.78E+02 7.9 −	4.33E+03 ± 6.20E+02 6.6 −	3.60E+03 ± 5.26E+02 5.3 −	2.44E+03 ± 2.67E+02 2.2
F29	8.63E+07 ± 3.03E+06 6.8 −	1.73E+06 ± 5.48E+06 1.5 +	1.38E+07 ± 2.18E+07 4.4 +	3.70E+07 ± 4.80E+07 4.5 =	1.57E+08 ± 9.85E+07 7.4 −	8.32E+06 ± 2.63E+07 4.1 +	1.42E+03 ± 1.77E+02 2.2 +	8.32E+07 ± 2.05E−07 5.4
F30	4.76E+06 ± 6.81E+05 8.0 −	1.51E+04 ± 2.76E+03 5.3 −	1.32E+04 ± 5.19E+03 4.0 =	1.09E+04 ± 1.75E+03 3.7 −	4.39E+04 ± 1.26E+04 7.0 −	9.19E+03 ± 1.39E+03 2.4 =	1.09E+04 ± 2.35E+03 3.7 −	8.93E+03 ± 6.38E+02 1.6
Overall f-rank w/l/t	228.2 0/29/1	110.1 9/12/9	144.7 2/19/9	92.2 11/11/8	190.8 1/25/4	135.8 5/19/6	88.7 12/9/9	89.5

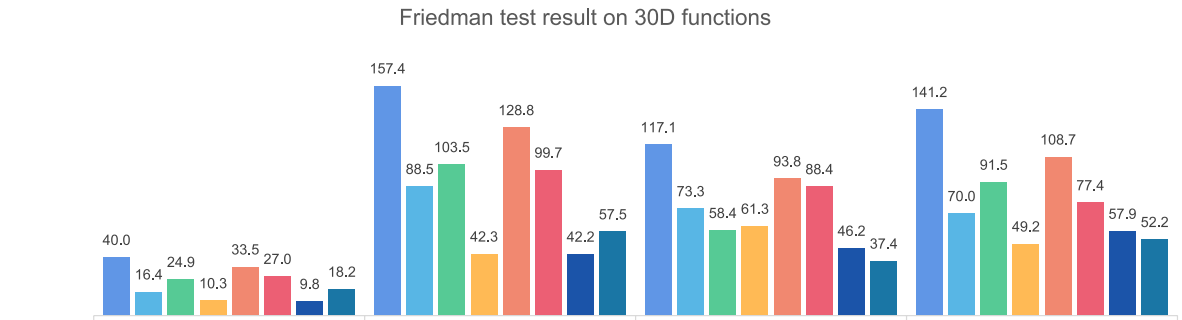
Table 7
The comparison results of OSCOA with other algorithms on CEC2017 with 30 Dimensions.

Function	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
F1	1.76E+10 ± 1.73E+09 8.0 −	4.27E−09 ± 1.35E−08 3.5 +	7.33E+04 ± 1.28E+05 4.1 =	1.42E−15 ± 4.49E−15 1.2 +	3.72E+06 ± 1.60E+06 6.8 −	4.50E+03 ± 6.12E+03 5.6 −	1.99E−14 ± 7.34E−15 2.5 +	1.64E−01 ± 1.40E−01 4.5
F3	1.14E+05 ± 1.68E+04 8.0 −	1.91E−11 ± 3.36E−11 2.5 +	5.86E+04 ± 1.67E+04 6.9 −	6.51E+03 ± 1.06E+04 2.6 =	3.43E+04 ± 4.12E+03 5.9 −	1.96E+02 ± 1.23E+02 4.5 −	7.15E+03 ± 2.26E+04 2.3 −	7.10E−06 ± 5.03E−06 3.5
F4	1.89E+03 ± 4.39E+02 8.0 −	2.22E+01 ± 3.39E+01 2.1 =	8.58E+01 ± 2.02E+01 5.7 −	5.49E+01 ± 1.95E+01 3.2 =	1.36E+02 ± 1.53E+01 7.1 −	5.59E+01 ± 3.27E+01 4.3 =	3.72E+01 ± 3.13E+01 2.6 =	4.59E+01 ± 2.42E+01 3.1
F5	2.88E+02 ± 1.85E+01 8.0 −	8.10E+01 ± 1.90E+01 4.1 −	1.05E+02 ± 3.32E+01 4.9 −	3.14E+01 ± 8.90E+00 1.8 +	1.84E+02 ± 2.55E+01 6.9 −	1.34E+02 ± 2.03E+01 5.8 −	2.13E+01 ± 4.02E+00 1.5 +	5.71E+01 ± 1.26E+01 3.3
F6	4.38E+01 ± 2.43E+00 8.0 −	1.44E−01 ± 3.09E−01 4.3 −	1.48E+00 ± 3.83E+00 5.5 −	4.15E−06 ± 4.95E−06 1.3 +	1.30E+01 ± 3.06E+00 6.9 −	1.86E−02 ± 2.71E−02 3.8 −	2.47E−02 ± 3.54E−02 4.4 −	1.02E−03 ± 1.02E−03 2.1
F7	7.09E+02 ± 8.62E+01 8.0 −	1.20E+02 ± 2.45E+01 4.1 −	1.54E+02 ± 3.84E+01 5.1 −	5.91E+01 ± 6.95E+00 1.9 +	2.87E+02 ± 2.55E+01 7.2 −	2.06E+02 ± 1.49E+01 5.7 −	5.34E+01 ± 4.31E+00 1.2 +	7.81E+01 ± 1.34E+01 2.7
F8	2.70E+02 ± 2.34E+01 8.0 −	7.91E+01 ± 2.15E+01 4.1 −	1.18E+02 ± 2.76E+01 5.4 −	2.90E+01 ± 8.58E+00 2.1 =	1.58E+02 ± 2.62E+01 6.6 −	1.36E+02 ± 1.36E+01 5.8 −	2.15E+01 ± 6.60E+00 1.5 +	4.31E+01 ± 1.70E+01 2.8
F9	6.23E+03 ± 1.02E+03 8.0 −	1.18E+02 ± 1.25E+02 5.4 −	4.41E+02 ± 1.20E+03 5.2 −	3.35E−01 ± 7.24E−01 1.4 =	1.82E+03 ± 3.29E+02 6.8 −	7.96E−01 ± 5.96E−01 2.7 −	9.48E+00 ± 4.60E+00 4.6 −	1.91E−01 ± 2.29E−01 1.6
F10	6.49E+03 ± 1.78E+02 7.9 −	2.97E+03 ± 5.90E+02 3.5 =	4.57E+03 ± 5.11E+02 5.6 −	2.61E+03 ± 4.67E+02 3.0 =	5.98E+03 ± 4.20E+02 7.1 −	4.35E+03 ± 4.91E+02 5.2 −	1.77E+03 ± 2.32E+02 1.0 +	2.57E+03 ± 4.21E+02 2.5
F11	7.87E+02 ± 9.56E+01 8.0 −	1.13E+02 ± 5.65E+01 5.1 −	8.03E+01 ± 2.94E+01 3.6 −	4.69E+01 ± 3.70E+01 2.5 =	1.59E+02 ± 1.50E+01 6.7 −	1.17E+02 ± 2.25E+01 5.4 −	7.93E+01 ± 3.42E+01 3.4 −	1.88E+01 ± 7.91E+00 1.3
F12	5.98E+08 ± 1.37E+08 8.0 −	1.71E+04 ± 1.54E+04 4.1 −	1.33E+04 ± 6.71E+03 4.5 −	2.29E+03 ± 9.24E+02 2.5 =	1.77E+06 ± 6.88E+05 7.0 −	6.25E+04 ± 2.41E+04 6.0 −	2.10E+03 ± 1.05E+03 2.1 =	2.43E+03 ± 3.15E+03 1.9
F13	1.48E+06 ± 3.95E+05 8.0 −	7.31E+02 ± 5.16E+02 3.9 −	8.03E+02 ± 1.31E+03 3.2 −	2.70E+04 ± 2.20E+04 5.5 −	3.51E+03 ± 6.69E+03 4.9 −	1.84E+04 ± 2.08E+04 6.2 −	2.65E+02 ± 3.37E+02 3.0 −	4.71E+01 ± 2.39E+01 1.3
F14	1.80E+02 ± 3.72E+01 5.1 −	2.30E+02 ± 9.98E+01 5.5 −	4.14E+01 ± 8.02E+00 2.0 =	2.99E+03 ± 3.95E+03 4.8 −	1.82E+02 ± 3.78E+01 5.5 −	4.44E+03 ± 4.30E+03 7.7 −	1.06E+02 ± 6.19E+01 3.7 −	3.50E+01 ± 1.31E+01 1.7
F15	1.63E+03 ± 2.11E+02 6.6 −	2.33E+03 ± 5.36E+03 5.1 −	3.33E+01 ± 2.86E+01 1.7 =	5.31E+03 ± 9.71E+03 5.0 −	8.15E+02 ± 2.89E+02 5.5 −	6.98E+03 ± 1.13E+04 6.6 −	1.85E+02 ± 9.45E+01 3.9 −	2.34E+01 ± 1.48E+01 1.3
F16	1.66E+03 ± 1.30E+02 7.9 −	6.96E+02 ± 2.67E+02 4.0 =	8.04E+02 ± 3.26E+02 4.6 =	5.41E+02 ± 2.77E+02 3.1 =	1.39E+03 ± 3.01E+02 6.9 −	5.40E+02 ± 2.91E+02 3.5 =	2.90E+02 ± 1.49E+02 1.6 +	7.17E+02 ± 3.25E+02 4.3
F17	5.35E+02 ± 8.53E+01 8.0 −	2.59E+02 ± 1.81E+02 4.7 =	3.66E+02 ± 6.78E+01 6.4 −	1.54E+02 ± 9.94E+01 3.0 =	1.93E+02 ± 9.62E+01 3.8 =	1.83E+02 ± 1.02E+02 3.9 =	8.56E+01 ± 5.34E+01 2.1 +	2.34E+02 ± 1.12E+02 4.4
F18	3.83E+05 ± 1.71E+05 7.8 −	7.70E+03 ± 1.04E+04 4.6 −	6.87E+01 ± 8.82E+01 1.9 =	9.78E+04 ± 2.70E+05 4.1 −	1.13E+05 ± 5.16E+04 6.3 −	1.28E+05 ± 7.26E+04 6.4 −	1.54E+02 ± 7.12E+01 3.5 −	4.25E+01 ± 1.16E+01 1.2
F19	4.69E+04 ± 2.20E+04 7.8 −	1.03E+02 ± 8.27E+01 3.8 −	2.23E+01 ± 8.65E+00 2.1 −	1.42E+04 ± 2.05E+04 5.2 −	8.70E+02 ± 5.08E+02 5.6 −	1.25E+04 ± 1.48E+04 6.7 −	1.01E+02 ± 4.75E+01 3.9 −	9.24E+00 ± 3.79E+00 1.1
F20	2.06E+02 ± 4.53E+01 4.7 =	2.05E+02 ± 9.75E+01 4.4 =	3.99E+02 ± 1.74E+02 6.9 =	2.07E+02 ± 1.08E+02 4.7 =	2.04E+02 ± 9.72E+01 4.0 =	1.61E+02 ± 8.47E+01 3.1 +	7.62E+01 ± 5.14E+01 1.9 +	3.63E+02 ± 2.02E+02 6.1
F21	4.69E+02 ± 9.16E+00 8.0 −	2.83E+02 ± 2.06E+01 4.4 −	2.89E+02 ± 3.58E+01 4.5 −	2.27E+02 ± 7.56E+00 1.8 +	3.53E+02 ± 2.37E+01 6.6 −	3.31E+02 ± 1.55E+01 6.0 −	2.23E+02 ± 5.69E+00 1.6 +	2.46E+02 ± 1.40E+01 2.8
F22	3.21E+03 ± 1.94E+03 7.4 =	1.01E+02 ± 1.19E+00 3.4 +	1.43E+03 ± 1.80E+03 6.7 =	1.00E+02 ± 7.89E−01 1.5 +	1.16E+02 ± 7.33E+00 5.6 =	1.01E+02 ± 1.19E+00 3.0 +	1.01E+02 ± 1.37E+00 2.4 +	1.67E+03 ± 1.70E+03 5.8
F23	6.40E+02 ± 1.61E+01 8.0 −	4.40E+02 ± 1.93E+01 4.2 −	4.77E+02 ± 4.29E+01 5.5 −	3.90E+02 ± 1.89E+01 2.2 =	5.17E+02 ± 2.68E+01 6.6 −	4.78E+02 ± 2.00E+01 5.6 −	3.72E+02 ± 8.67E+00 1.4 +	4.03E+02 ± 1.87E+01 2.4

(*continued on next page*)

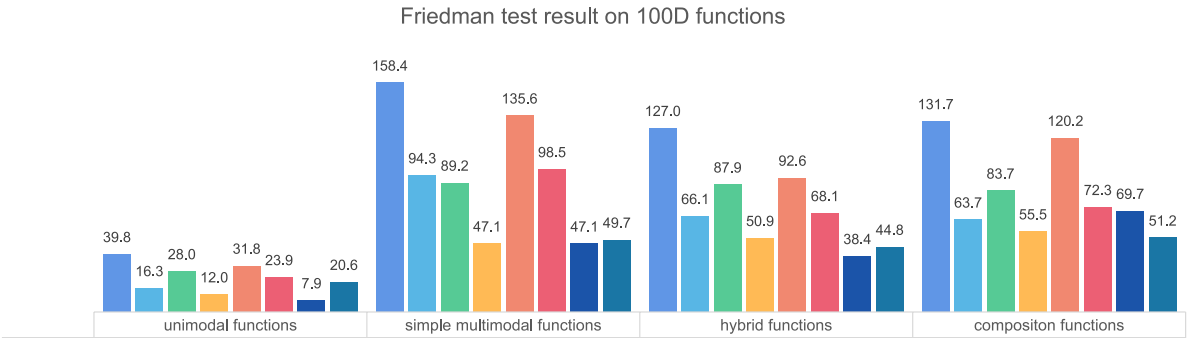
Table 7 (*continued*).

F24	6.89E+02 ± 1.01E+01 8.0 −	5.35E+02 ± 2.61E+01 4.6 −	5.26E+02 ± 2.58E+01 4.2 −	4.57E+02 ± 1.53E+01 2.0 =	5.88E+02 ± 2.62E+01 6.4 −	5.80E+02 ± 1.39E+01 6.1 −	4.39E+02 ± 8.61E+00 1.3 =	4.85E+02 ± 1.22E+01 3.1
F25	1.34E+03 ± 1.02E+02 8.0 −	3.89E+02 ± 3.01E+00 4.9 −	3.89E+02 ± 6.02E+00 3.7 −	3.87E+02 ± 1.33E+00 3.5 −	4.63E+02 ± 1.69E+01 7.0 −	3.87E+02 ± 1.40E+00 3.9 −	3.87E+02 ± 4.32E−01 3.7 −	3.87E+02 ± 1.23E−01 1.5
F26	4.41E+03 ± 1.23E+02 8.0 −	1.99E+03 ± 7.14E+02 4.7 −	1.92E+03 ± 5.80E+02 4.2 −	1.42E+03 ± 9.82E+01 2.3 =	3.06E+03 ± 3.42E+02 7.0 −	2.12E+03 ± 6.66E+02 4.8 −	1.29E+03 ± 1.12E+02 1.7 +	1.58E+03 ± 2.17E+02 3.1
F27	5.48E+02 ± 4.85E+00 7.5 −	5.33E+02 ± 2.11E+01 5.7 −	5.33E+02 ± 1.97E+01 5.7 −	5.10E+02 ± 7.95E+00 3.7 −	5.26E+02 ± 7.51E+00 5.4 −	5.07E+02 ± 1.87E+01 2.7 =	5.09E+02 ± 8.00E+00 3.5 =	5.02E+02 ± 1.01E+01 1.6
F28	1.31E+03 ± 1.14E+02 8.0 −	3.82E+02 ± 7.20E+01 4.3 =	3.88E+02 ± 6.62E+01 4.9 =	3.11E+02 ± 3.60E+01 1.5 +	5.19E+02 ± 3.03E+01 6.9 −	3.33E+02 ± 5.36E+01 3.0 +	3.47E+02 ± 6.24E+01 3.1 =	3.40E+02 ± 6.31E+01 4.3
F29	1.35E+03 ± 1.28E+02 8.0 −	7.50E+02 ± 1.28E+02 5.2 −	7.93E+02 ± 1.17E+02 5.6 −	5.03E+02 ± 6.81E+01 1.7 =	9.90E+02 ± 1.29E+02 6.8 −	5.63E+02 ± 6.63E+01 3.4 =	5.16E+02 ± 5.84E+01 2.5 =	5.54E+02 ± 9.96E+01 2.7
F30	5.30E+05 ± 1.43E+05 8.0 −	3.48E+03 ± 1.93E+03 4.7 −	2.13E+03 ± 2.28E+02 2.9 −	2.90E+03 ± 2.45E+03 2.8 −	2.07E+04 ± 1.50E+04 6.6 −	1.58E+04 ± 8.08E+03 6.3 −	2.20E+03 ± 1.58E+02 3.4 −	1.98E+03 ± 4.47E+01 1.2
Overall f-rank w/l/t	222.7 0/27/2	125.0 3/20/6	133.3 0/21/8	82.0 7/8/14	182.5 0/26/3	143.8 3/21/5	75.4 12/11/6	79.3



	unimodal functions	simple multimodal functions	hybrid functions	compositon functions
COA	40.0	157.4	117.1	141.2
BTLBO	16.4	88.5	73.3	70.0
ADE	24.9	103.5	58.4	91.5
DEET	10.3	42.3	61.3	49.2
CCOA	33.5	128.8	93.8	108.7
FDB-SOS	27.0	99.7	88.4	77.4
SHADE	9.8	42.2	46.2	57.9
OSCOA	18.2	57.5	37.4	52.2

Fig. 5. Friedman test ranking for different types of 30D problems.



	unimodal functions	simple multimodal functions	hybrid functions	compositon functions
COA	39.8	158.4	127.0	131.7
BTLBO	16.3	94.3	66.1	63.7
ADE	28.0	89.2	87.9	83.7
DEET	12.0	47.1	50.9	55.5
CCOA	31.8	135.6	92.6	120.2
FDB-SOS	23.9	98.5	68.1	72.3
SHADE	7.9	47.1	38.4	69.7
OSCOA	20.6	49.7	44.8	51.2

Fig. 6. Friedman test ranking for different types of 100D problems.

significantly outperforms COA, BTLBO, ADE, DEET, CCOA, FDB-SOS, and SHADE in 27, 20, 21, 8, 26, 21, and 11 test functions, respectively. In summary, the OSCOA demonstrates high competitiveness in the CEC2017 benchmark suite.

Similarly to the experiments conducted for CEC2014, the 100D CEC2017 is employed to assess OSCOA's performance in high-dimensional scenarios. The outcomes of these experiments are presented in Table 8. Although the overall f-rank indicates that OSCOA secures the second position after DEET and SHADE with a score of 76.8, the Wilcoxon rank-sum test reveals that OSCOA surpasses DEET and SHADE in nine tested functions each.

5.3. Analysis of the performance on different types of problems

In order to analyze the performance of OSCOA on different types of problems, Tables 9–10 and Figs. 5–6 show the statistical results of the Wilcoxon and the Friedman tests for the different types of test functions, respectively.

The Fig. 5 reveals that OSCOA achieves a second-place ranking on the simple multimodal functions, closely trailing DEET and SHADE. Additionally, OSCOA secures the top position in the hybrid function category. Although it slightly lags behind the DEET algorithm in the composition functions, OSCOA still demonstrates excellent performance in this area.

According to Table 9, it is observable that DEET and SHADE outperform OSCOA on unimodal functions and simple multimodal functions. However, OSCOA demonstrates superior performance on hybrid functions, significantly outperforming comparative algorithms on 14, 12, 7, 9, 13, 12, and 9 test functions. Moreover, OSCOA performs well on composition functions, significantly outperforming comparative algorithms on 16, 11, 13, 4, 13, 7, and 4 functions.

Subsequently, the statistical data of 100 dimensions is analyzed. As depicted in Fig. 6, the Friedman test results exhibit some changes compared to the 30D results, including OSCOA's overall f-rank in the unimodal functions and hybrid functions decreases as the dimension increases, but still has a strong performance, while OSCOA's ranking in the simple multimodal functions and composition functions has improved.

From Table 10, it can be observed that OSCOA shows improved performance on simple multimodal functions, significantly outperforming DEET on 8 functions and SHADE on 4 functions. Additionally, OSCOA maintains a strong competitive advantage on complex hybrid and composition functions. Particularly, in the

Table 8
The comparison results of OSCOA with other algorithms on CEC2017 with 100 Dimensions.

Function	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
F1	2.41E+11 ± 1.22E+10 8.0 −	4.77E+03 ± 7.72E+03 4.7 =	2.59E+02 ± 8.18E+02 3.1 +	7.47E−11 ± 1.62E−10 1.2 +	3.47E+09 ± 1.80E+09 7.0 −	6.88E+03 ± 6.27E+03 5.1 =	2.21E−09 ± 3.58E−09 1.8 +	6.97E+03 ± 1.06E+04 5.1
F3	6.28E+05 ± 5.79E+04 7.9 −	1.83E+01 ± 2.08E+01 2.6 +	3.74E+05 ± 1.32E+05 7.0 −	9.01E+04 ± 1.22E+05 3.4 =	2.70E+05 ± 1.77E+04 5.9 −	9.91E+04 ± 1.26E+04 4.6 −	5.28E−10 ± 4.89E−10 1.0 +	7.99E+03 ± 1.95E+03 3.6
F4	5.04E+04 ± 4.10E+03 8.0 −	2.75E+02 ± 4.83E+01 5.1 =	2.46E+02 ± 1.18E+02 3.9 =	1.21E+02 ± 7.55E+01 1.8 +	1.24E+03 ± 4.71E+02 7.0 −	2.18E+02 ± 3.96E+01 3.4 =	1.53E+02 ± 5.19E+01 2.3 +	2.47E+02 ± 3.67E+01 4.5
F5	1.58E+03 ± 5.12E+01 8.0 −	5.61E+02 ± 9.21E+01 4.5 −	5.87E+02 ± 1.80E+02 4.8 −	2.10E+02 ± 2.44E+01 1.7 +	1.04E+03 ± 5.95E+01 7.0 −	7.60E+02 ± 7.75E+01 5.7 −	2.14E+02 ± 3.16E+01 1.8 +	2.56E+02 ± 3.50E+01 2.5
F6	9.15E+01 ± 4.23E+00 8.0 −	2.12E+01 ± 4.16E+00 5.9 −	9.92E+00 ± 1.85E+01 3.4 −	6.78E−01 ± 2.81E−01 3.0 −	6.37E+01 ± 4.78E+00 7.1 −	1.29E+00 ± 2.67E−01 4.2 −	3.85E+00 ± 4.88E+00 3.4 =	2.90E−01 ± 8.80E−02 1.3
F7	6.05E+03 ± 2.22E+02 8.0 −	7.83E+02 ± 9.88E+01 4.3 −	1.09E+03 ± 4.82E+02 5.1 −	3.16E+02 ± 3.26E+01 1.1 +	1.82E+03 ± 1.60E+02 7.0 −	1.06E+03 ± 6.08E+01 5.8 −	6.42E+02 ± 7.06E+01 3.3 −	4.92E+02 ± 5.65E+01 1.6
F8	1.64E+03 ± 3.52E+01 8.0 −	5.95E+02 ± 9.31E+01 4.6 −	6.40E+02 ± 1.80E+02 5.0 −	1.82E+02 ± 2.89E+01 1.1 +	1.09E+03 ± 6.56E+01 7.1 −	7.42E+02 ± 7.20E+01 5.7 −	2.48E+02 ± 3.11E+01 2.3 +	2.98E+02 ± 5.21E+01 2.5
F9	8.96E+04 ± 2.24E+03 8.0 −	1.00E+04 ± 2.27E+03 4.8 −	1.19E+04 ± 7.17E+03 5.0 −	4.31E+02 ± 4.03E+02 1.1 +	4.81E+04 ± 3.86E+03 7.0 −	1.09E+04 ± 6.24E+03 4.8 −	5.00E+03 ± 2.32E+03 3.3 −	2.30E+03 ± 1.62E+03 2.0
F10	2.89E+04 ± 4.85E+02 8.0 −	1.30E+04 ± 1.58E+03 2.4 =	2.18E+04 ± 5.93E+02 5.8 −	1.53E+04 ± 1.52E+03 4.4 −	2.46E+04 ± 1.84E+03 7.0 −	2.11E+04 ± 5.98E+02 5.2 −	9.76E+03 ± 3.53E+02 1.0 +	1.25E+04 ± 1.01E+03 2.3
F11	1.36E+05 ± 1.61E+04 7.1 −	1.20E+03 ± 1.61E+02 3.7 −	1.64E+05 ± 2.22E+04 7.9 −	3.30E+04 ± 3.01E+04 5.3 −	1.97E+03 ± 2.59E+02 5.1 −	7.94E+02 ± 3.08E+02 2.3 −	5.93E+03 ± 1.48E+04 3.6 −	3.28E+02 ± 6.68E+01 1.0
F12	7.10E+10 ± 3.24E+09 8.0 −	1.16E+06 ± 1.07E+06 4.7 −	3.39E+05 ± 1.18E+05 3.4 =	8.15E+04 ± 6.67E+04 1.7 +	1.07E+08 ± 5.41E+07 7.0 −	5.36E+06 ± 2.59E+06 6.0 −	9.32E+04 ± 1.00E+05 1.9 +	3.13E+05 ± 7.71E+04 3.3
F13	7.86E+09 ± 7.58E+08 8.0 −	1.17E+04 ± 9.26E+03 5.8 −	1.98E+05 ± 6.12E+05 3.7 =	3.99E+03 ± 4.42E+03 3.8 =	9.19E+03 ± 2.51E+03 5.6 −	7.28E+03 ± 1.08E+04 3.5 =	2.93E+03 ± 1.55E+03 3.4 =	2.75E+03 ± 1.36E+03 2.3
F14	6.94E+06 ± 1.29E+06 8.0 −	4.59E+03 ± 5.74E+03 3.9 −	2.63E+04 ± 2.91E+04 4.9 −	8.07E+02 ± 1.45E+02 2.8 −	5.27E+05 ± 1.54E+05 6.7 −	4.32E+05 ± 1.87E+05 6.1 −	5.72E+02 ± 2.12E+02 2.4 −	4.56E+02 ± 1.11E+02 1.2
F15	1.24E+09 ± 1.49E+08 8.0 −	5.34E+03 ± 4.83E+03 5.4 =	1.08E+06 ± 3.36E+06 5.9 =	1.06E+03 ± 1.62E+03 2.6 =	1.03E+03 ± 4.32E+02 3.9 =	2.60E+03 ± 3.10E+03 3.9 =	4.62E+02 ± 1.62E+02 2.2 +	2.13E+03 ± 1.98E+03 4.3
F16	1.03E+04 ± 3.10E+02 8.0 −	3.70E+03 ± 4.90E+02 4.2 =	5.78E+03 ± 1.83E+03 5.8 −	2.60E+03 ± 6.38E+02 2.6 +	6.75E+03 ± 5.67E+02 6.6 −	2.78E+03 ± 8.90E+02 2.8 +	2.42E+03 ± 4.20E+02 1.8 +	3.82E+03 ± 9.81E+02 4.1
F17	8.33E+03 ± 6.97E+02 8.0 −	2.89E+03 ± 3.46E+02 4.8 −	3.85E+03 ± 4.09E+02 6.5 −	2.61E+03 ± 4.94E+02 3.8 =	3.58E+03 ± 7.09E+02 6.0 −	1.98E+03 ± 4.99E+02 2.3 =	2.14E+03 ± 2.54E+02 2.5 =	2.14E+03 ± 6.26E+02 2.4
F18	1.80E+07 ± 2.69E+06 7.9 −	2.82E+04 ± 2.00E+04 3.8 −	3.91E+06 ± 9.12E+06 5.7 −	5.72E+03 ± 4.18E+03 2.0 +	9.41E+05 ± 3.43E+05 6.3 −	9.20E+05 ± 4.19E+05 6.1 −	2.52E+03 ± 1.83E+03 1.5 +	1.49E+04 ± 1.20E+04 3.0
F19	1.12E+09 ± 1.82E+08 8.0 −	1.68E+03 ± 2.27E+03 4.3 −	9.12E+06 ± 2.88E+07 5.5 −	1.78E+03 ± 4.42E+03 3.2 −	2.02E+03 ± 1.84E+03 4.6 −	4.49E+03 ± 4.58E+03 4.2 −	3.33E+03 ± 5.21E+03 3.6 −	1.38E+03 ± 1.15E+03 2.3
F20	4.93E+03 ± 1.21E+02 8.0 −	2.58E+03 ± 5.77E+02 3.3 =	3.85E+03 ± 7.52E+02 6.0 −	2.91E+03 ± 8.21E+02 4.6 −	3.73E+03 ± 6.77E+02 5.8 −	2.23E+03 ± 5.84E+02 2.3 =	1.96E+03 ± 2.72E+02 2.0 +	2.68E+03 ± 5.54E+02 3.7
F21	1.90E+03 ± 4.54E+01 8.0 −	7.93E+02 ± 5.28E+01 4.8 −	6.92E+02 ± 1.11E+02 4.1 −	4.13E+02 ± 2.93E+01 1.5 +	1.13E+03 ± 8.16E+01 6.9 −	9.81E+02 ± 5.41E+01 6.1 −	4.84E+02 ± 4.36E+01 2.4 =	5.02E+02 ± 5.13E+01 2.4
F22	2.95E+04 ± 5.87E+02 7.9 −	1.41E+04 ± 1.04E+03 2.8 =	2.23E+04 ± 6.81E+02 5.6 −	1.66E+04 ± 1.46E+03 3.9 −	2.67E+04 ± 2.07E+03 7.0 −	2.20E+04 ± 6.86E+02 5.5 −	1.10E+04 ± 7.46E+02 1.3 +	1.37E+04 ± 7.98E+02 2.1
F23	1.93E+03 ± 6.84E+01 8.0 −	1.09E+03 ± 1.08E+02 5.3 −	1.03E+03 ± 2.84E+02 4.5 −	7.43E+02 ± 3.42E+01 2.4 −	1.40E+03 ± 7.56E+01 6.7 −	1.04E+03 ± 1.41E+02 4.6 −	8.61E+02 ± 5.05E+01 3.4 −	7.14E+02 ± 2.88E+01 1.3

(*continued on next page*)

Table 8 (*continued*).

F24	2.36E+03 ± 4.55E+01 8.0 −	1.62E+03 ± 1.35E+02 5.0 −	1.58E+03 ± 1.25E+02 4.9 −	1.07E+03 ± 3.26E+01 1.0 +	2.01E+03 ± 1.24E+02 6.9 −	1.66E+03 ± 7.24E+01 5.2 −	1.26E+03 ± 6.19E+01 2.9 −	1.17E+03 ± 4.71E+01 2.1
F25	3.18E+04 ± 3.35E+03 8.0 −	8.45E+02 ± 6.95E+01 4.7 =	8.24E+02 ± 8.40E+01 4.5 =	7.18E+02 ± 4.43E+01 2.4 =	1.75E+03 ± 2.57E+02 6.9 −	7.25E+02 ± 6.99E+01 2.7 =	8.04E+02 ± 6.10E+01 3.5 =	8.04E+02 ± 6.09E+01 3.0
F26	1.96E+04 ± 4.40E+02 7.5 −	1.01E+04 ± 3.51E+03 4.8 −	1.02E+04 ± 2.28E+03 4.2 −	5.42E+03 ± 3.89E+02 1.7 =	1.94E+04 ± 1.46E+03 7.4 −	1.13E+04 ± 5.59E+02 5.2 −	7.38E+03 ± 8.62E+02 2.9 −	6.41E+03 ± 8.23E+02 2.2
F27	1.01E+03 ± 3.82E+01 6.5 −	9.81E+02 ± 6.63E+01 5.9 −	9.38E+02 ± 3.05E+02 4.5 −	7.43E+02 ± 5.93E+01 2.6 −	1.21E+03 ± 1.47E+02 7.8 −	8.44E+02 ± 9.18E+01 4.1 −	8.21E+02 ± 6.48E+01 3.7 −	6.74E+02 ± 2.53E+01 1.1
F28	1.46E+04 ± 5.67E+02 7.9 −	6.25E+02 ± 4.31E+01 4.6 −	3.53E+03 ± 5.10E+03 4.7 =	5.17E+02 ± 1.78E+01 2.2 =	1.85E+03 ± 2.24E+02 6.3 −	5.42E+02 ± 2.28E+01 2.7 =	5.50E+02 ± 2.68E+01 3.4 =	5.81E+02 ± 1.80E+01 3.9
F29	1.25E+04 ± 8.19E+02 8.0 −	3.98E+03 ± 4.95E+02 5.4 −	4.26E+03 ± 6.92E+02 5.6 −	2.53E+03 ± 4.84E+02 2.7 =	6.55E+03 ± 3.74E+02 7.0 −	2.31E+03 ± 4.29E+02 2.2 =	2.35E+03 ± 4.61E+02 2.3 =	2.66E+03 ± 4.44E+02 2.8
F30	2.77E+09 ± 2.79E+08 8.0 −	4.94E+03 ± 1.56E+03 4.3 =	3.75E+03 ± 7.76E+02 3.2 =	2.91E+03 ± 3.23E+02 1.9 =	1.10E+05 ± 6.72E+04 6.9 −	6.40E+03 ± 3.77E+03 4.9 −	6.59E+03 ± 6.12E+03 3.6 =	3.80E+03 ± 9.49E+02 3.0
Overall f-rank w/l/t	228.7 0/29/0	130.3 1/19/9	144.1 1/21/7	73.4 11/9/9	189.4 0/28/1	127.1 1/19/9	74.4 12/9/8	76.8

Table 9
Wilcoxon rank-sum results in different types of 30D problems.

	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE
Unimodal functions	0/5/0	4/1/0	0/3/2	2/1/2	0/5/0	0/5/0	4/1/0
Simple multimodal functions	0/20/0	0/14/6	0/18/2	9/1/10	1/18/1	0/16/4	9/3/8
Hybrid functions	0/14/2	0/12/4	0/7/9	0/9/7	0/13/3	1/12/3	3/9/4
Composition functions	1/16/1	6/11/1	0/13/5	6/4/8	1/13/4	3/7/8	5/4/9

Table 10
Wilcoxon rank-sum results in different types of 100D problems.

	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE
Unimodal functions	0/5/0	4/0/1	1/3/1	3/0/2	0/5/0	0/3/2	5/0/0
Simple multimodal functions	0/20/0	0/14/6	0/15/5	11/8/1	0/19/1	1/17/2	9/4/7
Hybrid functions	0/16/0	0/9/7	0/11/5	4/5/7	0/13/3	1/9/6	8/4/4
Composition functions	0/17/1	6/8/4	2/11/5	4/7/7	1/16/1	4/9/5	2/10/6

Table 11
ADR in different types of 30D problems.

	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
Unimodal functions	0.9976	0.1998	0.9752	0.3845	0.9819	0.5226	0.3814	0.1975
Simple multimodal functions	0.4011	0.1221	0.1544	0.0861	0.2265	0.1651	0.0758	0.0940
Hybrid functions	0.6140	0.3173	0.2236	0.4764	0.4496	0.5560	0.1398	0.1275
Composition functions	0.3705	0.2450	0.2640	0.1813	0.2551	0.2506	0.2132	0.2269

Table 12
ADR in different types of 100D problems.

	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
Unimodal functions	0.9998	0.5503	0.9430	0.5969	0.9982	0.9780	0.2000	0.9250
Simple multimodal functions	0.5968	0.3378	0.3323	0.2378	0.4521	0.3583	0.2339	0.2403
Hybrid functions	0.9077	0.6675	0.8468	0.6513	0.7177	0.6842	0.5928	0.5722
Composition functions	0.6205	0.3895	0.4563	0.3850	0.5164	0.4269	0.3841	0.3760

case of composition functions, OSCOA significantly outperforms DEET and SHADE on 7 and 10 functions, respectively.

In order to further analyze the performance of the algorithm on different types of problems, the average deviation rate (*ADR*) between the best solution found by the algorithm and the global optimum is used as a metric of algorithm effectiveness. The *ADR* is defined in Eq. (33)

$$ADR = \left(\sum_{i=1}^{NF} \frac{P_{\text{best}} - P_{\text{optimal}}}{P_{\text{best}}} \right) / NF \tag{33}$$

where P_{best} represents the best solution found by the algorithm and P_{optimal} represents the global optimum solution, NF represents the number of test functions. Eq. (33) maps the optimization results of the algorithm to the range [0,1], indicating that if P_{best} is closer to P_{optimal}, the *ADR* approaches 0. Tables 11 and 12 present the *ADR* of the algorithms on different dimensions and types of problems.

From Table 11, it can be observed that OSCOA achieves the first position with an *ADR* of 0.1975 in the unimodal functions, and achieves a decent 0.0940 in the simple multimodal functions. In the hybrid functions, OSCOA again takes the first position with an *ADR* value of 0.1275. Lastly, in the composition functions, OSCOA achieves an *ADR* of 0.2269. Analyzing Table 12, it can be noted that OSCOA's performance slightly diminishes in unimodal functions with 100 dimensions. However, it still demonstrates commendable performance in multimodal functions. Notably, OSCOA takes the lead in hybrid functions and composition functions.

5.4. Performance of OSCOA on the latest problems

To evaluate the performance of the OSCOA algorithm on the latest optimization problems, the CEC2022 benchmark suite is used as the test functions in this section's experiment. The CEC2022 also includes four types of functions, namely unimodal function (F1), basic functions (F2–F5), hybrid functions (F6–F8), and composition functions (F9–F12).

Table 13 presents the test results on CEC2022. From the table, it can be observed that OSCOA achieves the global optimum on F1, F3, F5, and F11, and obtains the first rank in the Friedman test on F6 and F12. From the statistical data, OSCOA has an overall f-rank value of 39.18, ranking closely behind the DEET algorithm with a slight margin. And the w/l/t results show that OSCOA significantly outperforms the competitors on 5, 7, 8, 4, 8, 5, and 2 functions, respectively.

5.5. Convergence and stability analysis of OSCOA

From the description of the OSCOA algorithm, it can be inferred that OSCOA is a stochastic search algorithm. Solis et al. provided the necessary and sufficient conditions for global convergence of stochastic algorithms [39].

- **Condition 1.** *$f(D(x,\xi)) \leq f(x)$ and if $\xi \in S$, $f(D(x,\xi)) \leq f(\xi)$, where f represents the objective function, D represents the stochastic optimization algorithm, ξ represents the optimal solution searched during the iterations of D, S represents a subset of the solution space, and x represents the best solution in the S.*
- **Condition 2.** *For any (Borel) subsets A of S with $v(A) > 0$, the following holds: $\prod_{k=0}^{\infty}(1 - \mu_k(A)) = 0$, where $\mu_k(A)$ is the probability measure that satisfies the global convergence criterion, and $v(A)$ represents the n-dimensional volume of the set A.*
- **Theorem 1.** *Let f be a measurable function, S be a measurable subset of R^n, and x_k be a sequence of solutions generated by a stochastic search. When Condition 1 and Condition 2 are satisfied, it holds that $\lim_{k\to+\infty} P[x_k \in R_\varepsilon] = 1$, where $P[x_k \in R_\varepsilon]$ represents the probability that the solution $x_k \in R_\varepsilon$*

Table 13
The comparison results of OSCOA with other algorithms on CEC2022.

Function	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
F1	0.00E+00 ± 3.04E−14 3.73 −	0.00E+00 ± 5.88E−14 5.40 −	8.23E+02 ± 1.08E+03 8.00 −	0.00E+00 ± 0.00E+00 2.97 =	2.62E−04 ± 1.71E−04 7.00 −	0.00E+00 ± 0.00E+00 2.97 =	0.00E+00 ± 0.00E+00 2.97 =	0.00E+00 ± 0.00E+00 2.97 =
F2	4.71E+00 ± 2.41E+00 5.13 =	9.94E+00 ± 1.72E+01 5.33 =	3.94E+00 ± 4.57E+00 4.03 =	6.94E+00 ± 2.50E+00 6.60 =	2.34E−01 ± 6.64E−01 1.67 +	6.92E+00 ± 3.24E+00 5.97 −	4.50E+00 ± 3.17E+00 3.80 =	5.30E+00 ± 2.26E+00 3.47 =
F3	8.76E−04 ± 2.36E−03 4.43 −	0.00E+00 ± 3.04E−14 4.43 −	1.72E−02 ± 4.68E−02 7.00 −	0.00E+00 ± 0.00E+00 2.13 −	2.87E−02 ± 2.41E−02 7.87 −	7.70E−08 ± 1.46E−07 4.73 −	1.89E−07 ± 5.00E−07 2.87 =	0.00E+00 ± 1.32E−13 2.53 =
F4	1.65E+01 ± 5.65E+00 6.67 −	1.30E+01 ± 5.90E+00 5.20 =	1.44E+01 ± 3.69E+00 5.93 −	4.64E+00 ± 2.88E+00 2.03 +	8.00E+00 ± 2.67E+00 3.87 =	1.46E+01 ± 5.04E+00 6.57 −	2.92E+00 ± 8.79E−01 1.73 +	1.05E+01 ± 4.65E+00 4.00 =
F5	1.61E+00 ± 3.22E+00 6.40 −	6.65E−02 ± 1.59E−01 4.23 −	2.07E+01 ± 4.98E+01 5.83 −	0.00E+00 ± 0.00E+00 3.07 =	5.59E−02 ± 1.89E−01 6.67 −	0.00E+00 ± 7.44E−10 3.67 −	0.00E+00 ± 0.00E+00 3.07 =	0.00E+00 ± 0.00E+00 3.07 =
F6	9.68E−01 ± 5.44E−01 3.93 =	2.37E+03 ± 2.78E+03 7.17 −	4.05E+00 ± 1.42E+01 2.87 −	7.81E−01 ± 9.11E−01 3.67 =	3.72E+01 ± 1.55E+01 6.40 −	2.00E+03 ± 4.21E+03 7.40 −	4.17E−01 ± 4.45E−01 2.43 =	6.32E−01 ± 4.68E−01 2.13 =
F7	6.82E+00 ± 9.70E+00 4.23 =	9.99E+00 ± 9.74E+00 6.07 −	1.12E+01 ± 5.47E+00 6.73 −	2.87E+00 ± 6.97E+00 2.87 =	2.98E+00 ± 3.94E+00 5.40 =	9.88E−01 ± 1.12E+00 4.27 =	6.17E−02 ± 9.86E−02 2.80 =	5.57E+00 ± 9.01E+00 3.63 =
F8	1.14E+01 ± 9.09E+00 4.67 =	1.63E+01 ± 8.30E+00 5.87 =	1.45E+01 ± 7.66E+00 5.53 =	3.12E+00 ± 6.95E+00 2.40 +	7.60E+00 ± 5.42E+00 4.20 =	1.36E+01 ± 8.08E+00 5.83 =	3.65E+00 ± 7.11E+00 2.83 =	1.27E+01 ± 9.90E+00 4.67 =
F9	2.29E+02 ± 0.00E+00 4.00 =	2.29E+02 ± 0.00E+00 4.00 =	2.29E+02 ± 0.00E+00 4.00 =	2.29E+02 ± 0.00E+00 4.00 =	2.29E+02 ± 7.46E−08 8.00 −	2.29E+02 ± 0.00E+00 4.00 =	2.29E+02 ± 0.00E+00 4.00 =	2.29E+02 ± 0.00E+00 4.00 =
F10	8.70E+01 ± 3.53E+01 5.27 +	1.07E+02 ± 2.77E+01 4.27 =	1.02E+02 ± 2.44E+00 6.80 −	1.07E+02 ± 2.75E+01 2.87 −	1.00E+02 ± 7.07E−02 6.67 −	1.00E+02 ± 3.95E−02 2.57 +	1.07E+02 ± 2.71E+01 4.17 =	1.00E+02 ± 5.07E−02 3.40 =
F11	1.00E+01 ± 3.88E+01 4.50 =	4.67E+01 ± 1.25E+02 5.27 −	1.11E+02 ± 6.04E+01 7.80 −	0.00E+00 ± 0.00E+00 2.37 −	6.37E−03 ± 1.08E−02 6.80 −	0.00E+00 ± 2.11E−13 2.99 =	0.00E+00 ± 1.22E−13 2.63 -	0.00E+00 ± 3.22E−13 3.65 =
F12	1.63E+02 ± 1.60E+00 5.33 −	1.63E+02 ± 1.53E+00 5.83 −	1.61E+02 ± 1.20E+00 3.37 =	1.61E+02 ± 7.57E−01 3.95 −	1.63E+02 ± 1.61E+00 5.27 −	1.61E+02 ± 2.23E+00 2.83 =	1.64E+02 ± 1.39E+00 7.75 −	1.61E+02 ± 1.07E+00 1.67 =
Overall f-rank	58.30	63.07	67.90	38.91	69.80	53.79	41.05	39.18
w/l/t	1/5/6	0/7/5	0/8/4	2/4/6	1/8/3	1/5/6	1/2/9	

generated by the algorithm at step k, and R_ε represents the set of global optimal points.

The proofs that OSCOA satisfies Conditions 1 and 2 are presented in the following.

Proof (*OSCOA Satisfies Condition 1*)**.** In the OSCOA algorithm, the solution sequence can be represented as $\{soc_g^t\}$, where soc_g^t represents the fittest coyote in the tth generation. Therefore, the OSCOA algorithm defines the function D:

$$D\left(x_i^t, soc_g^t\right) = \begin{cases} soc_g^t, f\left(soc_g^t\right) \le f\left(x_i^t\right) \\ x_i^t, f\left(soc_g^t\right) > f\left(x_i^t\right) \end{cases} \tag{34}$$

From the above equation, it can be inferred that $f(soc_g^t)$ preserves convergence in the solution space and continuously approaches the minimum value of the solution space.

Proof (*OSCOA Satisfies Condition 2*)**.** For an algorithm with a population size of N, it can be shown that Condition 2 can be transformed into a proof for $S \subseteq \bigcup_{i=1}^{N} M_i^t$, where M_i^t represents the support set of individual i's sample space in generation t.

In Eq. (28), it is shown that pups consists of three parts, where R_j represents an arbitrary position in the search space of dimension j. Therefore, pups has the potential to appear at any position within the search space, i.e., $M_i^t = S$ for pups. Consequently, the OSCOA algorithm satisfies $S \subseteq \bigcup_{i=1}^{N} M_i^t$.

The analysis above has shown that the OSCOA algorithm satisfies Conditions 1 and 2. Therefore, according to Theorem 1, it can be concluded that the OSCOA algorithm converges to the global optimum with probability 1 as $t \to \infty$ [40].

To provide a clearer representation of the convergence performance of the OSCOA algorithm, Fig. 7 illustrates the four qualitative metrics for the convergence performance, including search history, one-dimensional trajectory, optimization history, and diversity history. The search history displays the positions of the top two dimensions of 10 random individuals in each generation of the population. The one-dimensional trajectory shows the position changes of the first dimension of the first individual. The optimization history represents the fitness value of the best individual. The diversity history displays the diversity changes of the population in the search space, where the diversity is represented by the average Euclidean distance of all individuals. These qualitative metrics are evaluated on several commonly used test functions, including Sphere, Schwefel-102-noise, Quaritic, Griewank, and Ackley.

From the search history, it can be observed that a large number of individuals are distributed around the optimal solution, demonstrating the exploitation capability of OSCOA. At the same time, the appropriate dispersion of individuals also indicates OSCOA's potential exploration capability. The one-dimensional trajectory shows that they have a large fluctuation in the early stage, but as the iteration progresses, the trajectory gradually becomes stable. At the same time, the optimization history curve and the diversity history curve of the population both exhibit a gradual decreasing trend.

The four metrics shown above all demonstrate OSCOA's excellent convergence ability.

Moreover, some average convergence curves of algorithms are shown in Fig. 8 to further demonstrate the convergence ability of the OSCOA algorithm. It can be seen from these figures that OSCOA shows excellent optimization ability in the early stage on CEC2014 F12, F29 and CEC2017 F11, F14. Although OSCOA's optimization ability is insufficient in the early stage in other functions, it has stronger continuous optimization ability.

Next, the algorithm stability is analyzed, some boxplots of algorithms are shown in Fig. 9. The outcomes illustrated in these boxplots confirm that the OSCOA algorithm consistently demonstrates stable optimization performance across various dimensions and test functions. This further reinforces the superiority and reliability of the OSCOA algorithm.

6. Operating characteristics of OSCOA

To provide a clearer demonstration of how IOS guides the strategy adjustments of the OSCOA algorithm, Fig. 10 illustrates four operating characteristics of the OSCOA algorithm across four distinct test functions.

The leftmost figure in Fig. 10 depicts the 3D plot of test functions, followed by four figures illustrating the synchronized trends between four key operating characteristics and the IOS curve during the operation of OSCOA. These four characteristics include the number of packs, the selection rate of the exploitation operator, the selection rate of elite parents, and population diversity.

From Fig. 10, it can be seen that the IOS curve shows synchronous changes with the operating characteristics, indicating the effectiveness of IOS in guiding the algorithm search.

Concurrently, these figures also showcase how OSCOA tailors distinct optimization strategies for different types of optimization problems. For relatively simple problems like CEC2017 F4 and F9, the algorithm quickly transitions from exploration to exploitation in the early stages, helping the population to exploit more advantageous regions. However, for relatively complex problems like CEC2017 F22, OSCOA algorithm selects more exploration strategies. This results in diversity fluctuations within the population, ultimately bolstering its capacity for early-stage exploration. In the case of CEC2014 F10, which encompasses numerous local optima, Figs. 10(q)–10(t) show a stronger early-stage exploration capability. Furthermore, the diversity curve indicates that population diversity remains more sustained. This persistence aids the population in exploring a broader spectrum of areas and evades swift convergence towards local optima.

To clearly compare the differences in population diversity changes of OSCOA on the above four test functions, four population diversity curves are shown in Fig. 10(u). From the figure, it is evident that OSCOA algorithm successfully and adaptively regulates population diversity across different optimization problems. This further proves the rationality and effectiveness of the design of OSCOA algorithm.

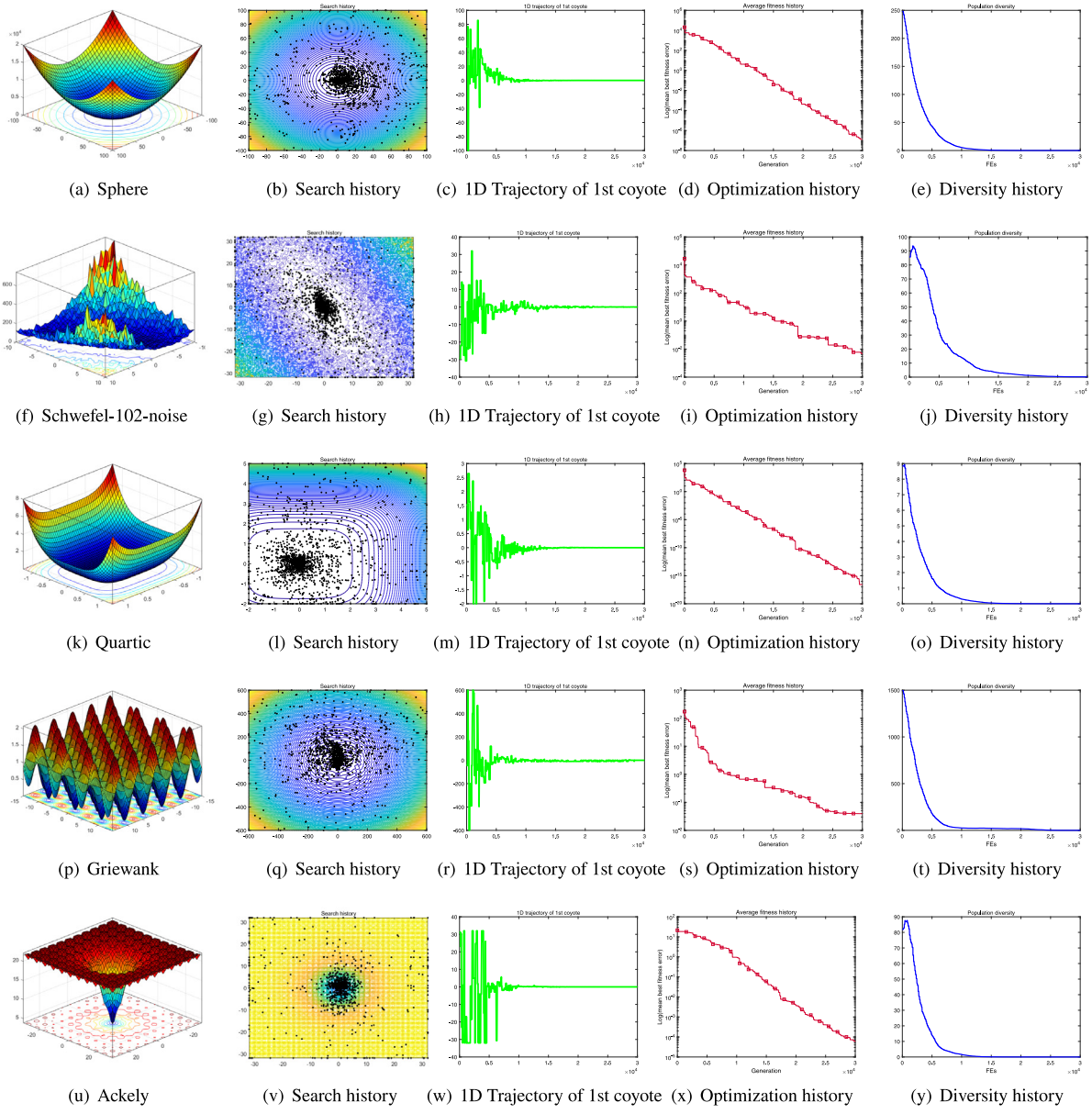
To better demonstrate the characteristics of population diversity in the OSCOA algorithm, Fig. 11 shows the diversity history of OSCOA and SHADE on the some test functions. Observing these figures, it can be observed that OSCOA exhibits a more diverse trend in population diversity compared to SHADE in most of the test functions. This indicates that OSCOA is more effective in regulating population diversity and maintaining a diverse set of solutions throughout the optimization process.

7. Analysis of parameters

This section conducts an analysis on the OSCOA's population size N, upper bound UN_c and lower bound LN_c of pack size.

7.1. Upper and lower bounds of pack size

Firstly, two parameters, UN_c and LN_c, in Eq. (18) are analyzed with the N set to 100 in this experiment. To ensure that each pack contains at least three coyotes, LN_c is set as a minimum of 3, while UN_c is set as a maximum of 100.



(a) Sphere (b) Search history (c) 1D Trajectory of 1st coyote (d) Optimization history (e) Diversity history

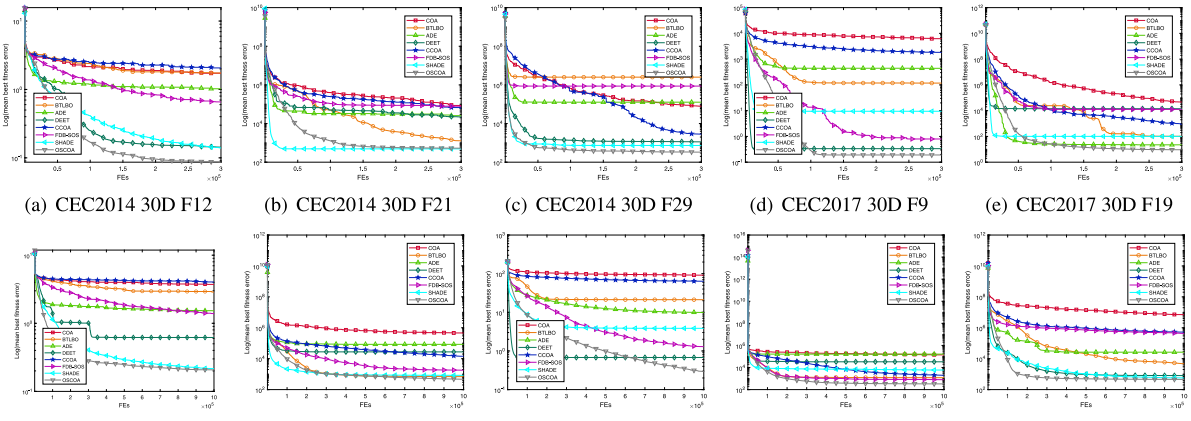
(f) Schwefel-102-noise (g) Search history (h) 1D Trajectory of 1st coyote (i) Optimization history (j) Diversity history

(k) Quartic (l) Search history (m) 1D Trajectory of 1st coyote (n) Optimization history (o) Diversity history

(p) Griewank (q) Search history (r) 1D Trajectory of 1st coyote (s) Optimization history (t) Diversity history

(u) Ackely (v) Search history (w) 1D Trajectory of 1st coyote (x) Optimization history (y) Diversity history

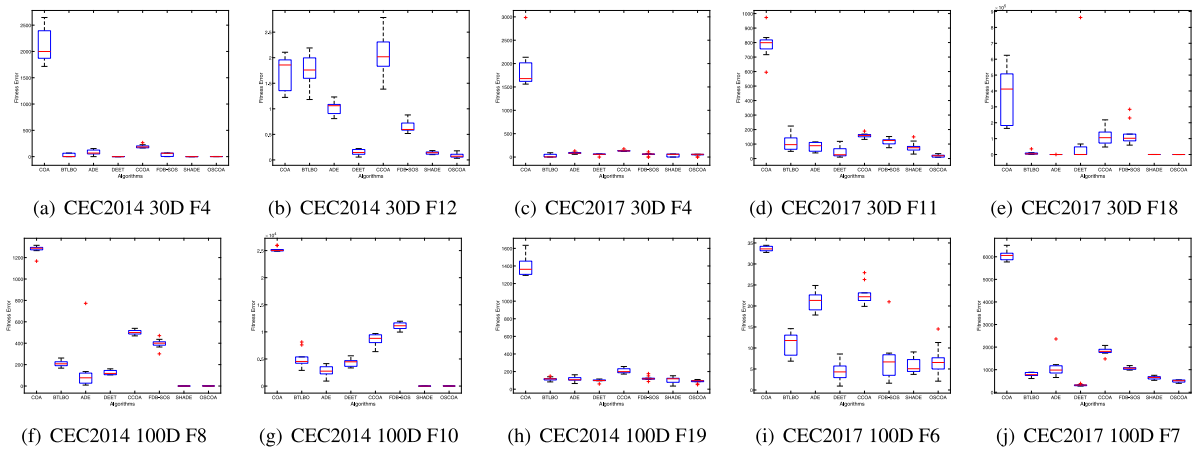
Fig. 7. Qualitative metrics: search history, optimization history, trajectory in 1st dimension and diversity history.



(a) CEC2014 30D F12 (b) CEC2014 30D F21 (c) CEC2014 30D F29 (d) CEC2017 30D F9 (e) CEC2017 30D F19

(f) CEC2014 100D F12 (g) CEC2014 100D F20 (h) CEC2017 100D F6 (i) CEC2017 100D F11 (j) CEC2017 100D F14

Fig. 8. The convergence curves of the comparison of OSCOA with other algorithms.



(a) CEC2014 30D F4 (b) CEC2014 30D F12 (c) CEC2017 30D F4 (d) CEC2017 30D F11 (e) CEC2017 30D F18

(f) CEC2014 100D F8 (g) CEC2014 100D F10 (h) CEC2014 100D F19 (i) CEC2017 100D F6 (j) CEC2017 100D F7

Fig. 9. The boxplots of the comparison of OSCOA with other algorithms.

In order to comprehensively analyze the impact of LN_c and UN_c on the algorithm, and find the optimal parameter combination, grid search is conducted. To quickly locate the advantageous parameter region, a coarse-grained grid search is carried out with values of LN_c set as 3, 5, 20, 33, and 50, and values of LN_c set as 5, 20, 25, 33, 50, and 100. These parameter combinations are tested on CEC2017, and the results are presented in the form of overall f-rank in Fig. 12(a).

From Fig. 12(a), it can be seen that as UN_c increases, the overall performance of the algorithm gradually deteriorates. Therefore, a smaller UN_c is more suitable for OSCOA. For the LN_c, the figure shows that when LN_c is greater than 5, smaller values of LN_c result in better algorithm performance. However, when LN_c is reduced to 3, the performance decreases. This above analysis indicates that OSCOA is sensitive to both UN_c and LN_c.

Fig. 12(a) also reveals that OSCOA achieves optimal performance when LN_c and UN_c are taken to be 5 and 20 respectively. To pinpoint the optimal combination of LN_c and UN_c, Fig. 12(b) presents more fine-grained experimental results, confirming that 5 and 20 are the optimal values for LN_c and UN_c, respectively.

7.2. Population size

To analyze the effect of population size N on the performance of the OSCOA, Fig. 13 shows the experimental results of the OSCOA algorithm on CEC2014 for different population sizes. The experiment is performed with UN_c and LN_c of 20, 5 respectively.

The experimental results on unimodal functions show that smaller N favors the algorithm's optimization, this indicates that a smaller population size is more conducive to population exploitation. On the contrary, increasing the population size improves algorithms' exploration capability to some extent [41], which is beneficial for optimizing multimodal problems. This trend is also demonstrated in the three types of multimodal functions shown in Fig. 13. For example, when the population size N is increased from 60 to 100 or 150, the algorithm's performance is improved. However, as N continues to increase, the performance of OSCOA gradually deteriorates. This indicates that the balance between population exploration and exploitation has been disrupted, and excessive population size weakens the algorithm's exploitation capabilities. Therefore, under the current parameter environment, setting N to 100 adequately balances the algorithm's exploration and exploitation capabilities (as shown in Fig. 13).

8. Application I: Multi-thresholding image segmentation problems

Image segmentation is a technique to simplify images. The key of image segmentation is to find the optimal thresholds to divide pixels into several sub-classes and extract useful targets. The segmentation technique can help reduce the complexity of image content [42] and it is essential for image analysis. Image segmentation has been widely used in various fields, such as medical image [43], pattern recognition, document recognition [44] and so on.

As of now, many thresholding segmentation methods have been proposed, encompassing techniques like clustering, mutual information, and maximum entropy. The maximum entropy approach views an image's histogram as a probability distribution and then calculates the maximum entropy to identify optimal thresholds [45]. Kapur et al. introduced a simple yet effective maximum entropy method that yields ideal segmentation outcomes [46]. This method is employed to formulate the image segmentation model. In this section, we utilize the OSCOA algorithm to ascertain the optimal segmentation threshold for some images. Moreover, this real-world application problem serves to evaluate the OSCOA algorithm's competence in addressing practical optimization challenges.

8.1. Kapur's entropy method

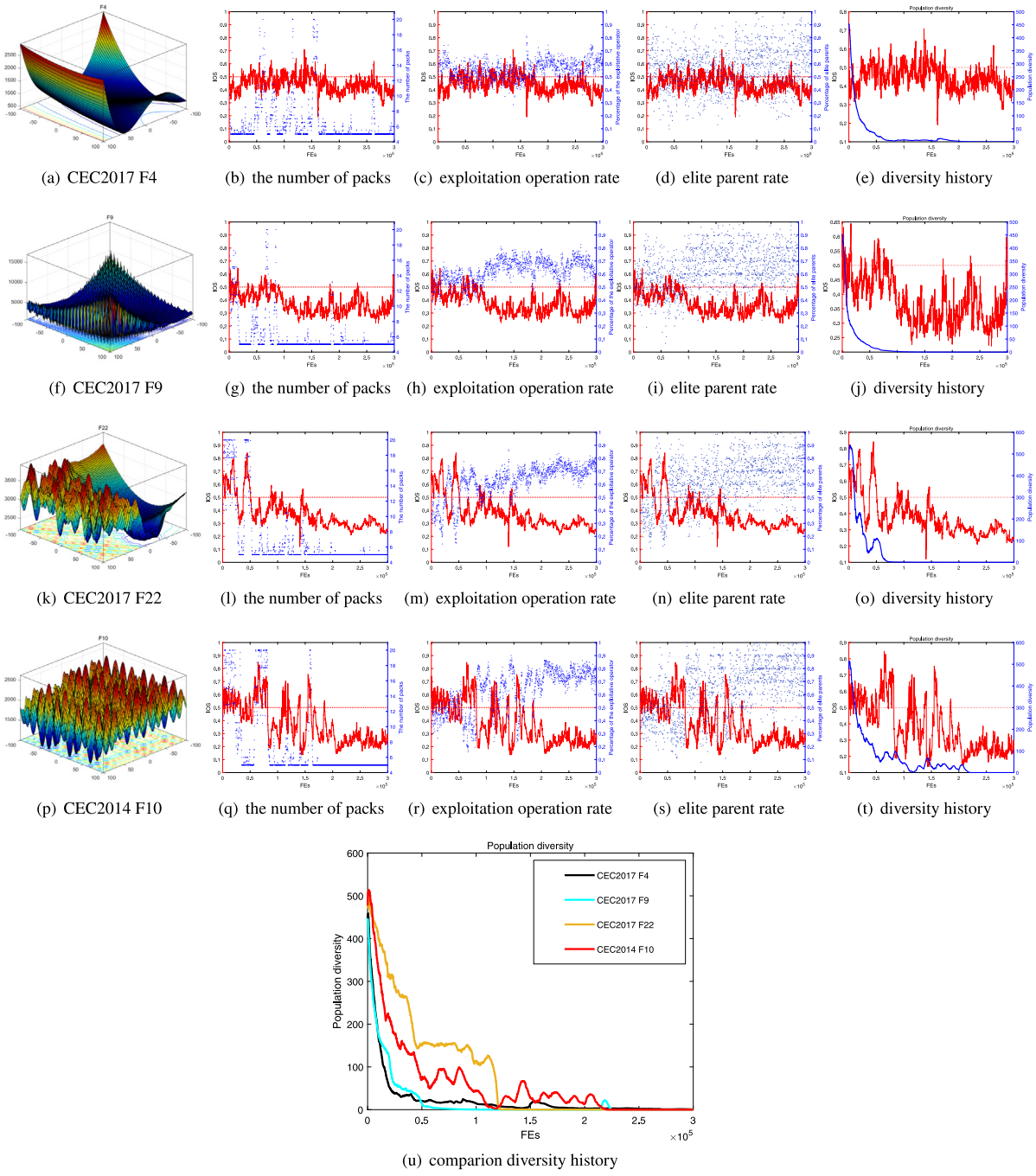
Assume that the gray interval of an image is $[0, 1, \ldots, L-1]$, and the entropy of the modified image is represented by H, and its calculation formula is shown in Eq. (35).

$$H = -\sum_{i=0}^{L-1} \frac{p_i}{P} \cdot \ln \frac{p_i}{P} \tag{35}$$

where p_i is the number of pixels with the value i, P is the total number of pixels. Suppose that an image requires $m+1$ level image segmentation, and m thresholds are required. These thresholds are expressed as $[t_1, t_2, \ldots, t_m]$ which divide the pixel interval of the image into $m+1$ subclasses, and Eqs. (36)–(38) gives the entropy of the nth subclass.

$$H_n = -\sum_{i=t_{n-1}}^{t_n-1} \frac{X_i}{T_n} \cdot \ln \frac{X_i}{T_n}, n \in [1, 2, \ldots, m+1] \tag{36}$$

$$X_i = \frac{p_i}{P} \tag{37}$$



(a) CEC2017 F4 (b) the number of packs (c) exploitation operation rate (d) elite parent rate (e) diversity history

(f) CEC2017 F9 (g) the number of packs (h) exploitation operation rate (i) elite parent rate (j) diversity history

(k) CEC2017 F22 (l) the number of packs (m) exploitation operation rate (n) elite parent rate (o) diversity history

(p) CEC2014 F10 (q) the number of packs (r) exploitation operation rate (s) elite parent rate (t) diversity history

(u) comparion diversity history

Fig. 10. Operating characteristics of OSCOA.

$$T_n = \sum_{i=t_{n-1}}^{t_n-1} X_i \tag{38}$$

where t_0 and t_{m+1} denote the upper and lower bounds of the pixel interval, respectively. Therefore, for image segmentation of M + 1 level, the objective function to be optimized can be expressed as Eq. (39).

$$\max F\,(t_1, t_2, \ldots, t_m) = H_1 + H_2 + \cdots + H_{m+1} \tag{39}$$

8.2. Experiment and analysis

This section uses the proposed OSCOA algorithm to optimize the image segmentation problems.

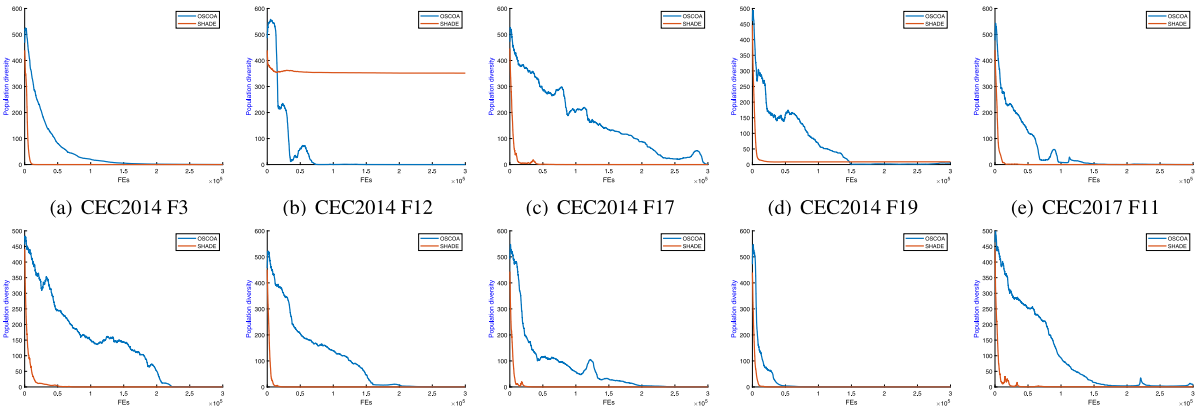
In this experiment, image segmentation of 11 levels are performed on nine grayscale images, the images include Aeroplane, Cameraman, Pepper, Fighter, Woman, Sailboat, Monkey, Tank and House, which are from the Berkeley University Dataset. Each algorithm is run independently 30 times and the MaxFEs is set to 10^5.

This experimental results are shown in Table 14, while the four original images, the segmented images, and the threshold distribution are shown in Fig. 14, respectively.

From the Table 14, it can be seen that the OSCOA algorithm gets the second place of Friedman test on image Monkey, but it wins the first place on all the other eight images. The overall f-rank of OSCOA is the first place with 13.6. The results of the

Table 14
The comparison results of OSCOA with other algorithms on image segmentation problems.

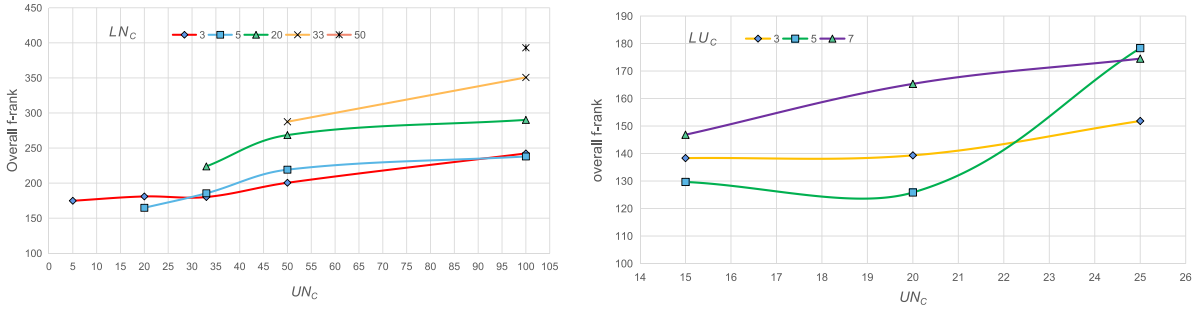
Image	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
Aeroplane	31.23613 ± 0.1021 6.95 −	31.24122 ± 0.1129 5.75 −	31.30795 ± 0.0796 4.55 −	31.41147 ± 0.1244 1.85 =	31.29034 ± 0.0962 4.45 −	31.23729 ± 0.0959 7.95 −	31.42368 ± 0.1171 3.05 =	31.48004 ± 0.1009 1.45
Cameraman	33.08229 ± 0.0752 7.75 −	33.19645 ± 0.0360 3.65 −	33.15503 ± 0.0281 6.05 −	33.25475 ± 0.0189 1.75 =	33.18957 ± 0.0429 5.05 −	33.09906 ± 0.0472 7.15 −	33.25049 ± 0.0198 3.35 =	33.26294 ± 0.0151 1.25
Pepper	33.67699 ± 0.0631 7.93 −	33.78252 ± 0.1367 3.63 −	33.76555 ± 0.0542 5.23 −	33.86234 ± 0.0008 1.83 −	33.74574 ± 0.0686 5.93 −	33.73825 ± 0.0507 7.13 −	33.86083 ± 0.0021 2.93 −	33.86278 ± 0.0002 1.43
Fighter	33.56649 ± 0.0789 7.50 −	33.54974 ± 0.2418 4.70 −	33.59222 ± 0.0747 5.90 −	33.78458 ± 0.0010 1.90 =	33.66256 ± 0.0500 3.80 −	33.55769 ± 0.0790 7.70 −	33.78425 ± 0.0010 2.90 =	33.78517 ± 0.0000 1.60
Woman	32.88941 ± 0.0871 7.81 −	32.75164 ± 0.2699 6.71 −	32.89294 ± 0.0737 5.01 −	33.07496 ± 0.0588 3.01 −	32.88495 ± 0.0838 6.01 −	32.90534 ± 0.0554 4.01 −	33.06315 ± 0.0605 2.01 −	33.11246 ± 0.0254 1.41
Sailboat	33.05568 ± 0.0848 4.16 −	32.93584 ± 0.2174 7.56 =	33.03933 ± 0.0869 4.86 −	33.21384 ± 0.0127 2.06 =	33.05306 ± 0.0611 5.86 −	33.01335 ± 0.0761 6.86 −	33.19648 ± 0.0420 2.86 =	33.22120 ± 0.0078 1.76
Monkey	33.08748 ± 0.0496 7.90 −	33.15680 ± 0.0903 3.70 −	33.11988 ± 0.0525 5.90 −	33.28169 ± 0.0247 2.00 =	33.15187 ± 0.0558 5.10 −	33.09422 ± 0.0589 7.10 −	33.25834 ± 0.0363 2.60 =	33.27847 ± 0.0351 1.70
Tank	29.03807 ± 0.1132 7.84 −	28.84100 ± 0.3586 6.74 −	29.14730 ± 0.0673 5.04 −	29.41391 ± 0.0044 1.94 =	29.20694 ± 0.0630 3.94 −	29.06233 ± 0.0935 6.14 −	29.27443 ± 0.1913 2.84 −	29.41583 ± 0.0035 1.54
House	32.39387 ± 0.1317 7.44 −	32.16673 ± 0.3089 7.84 =	32.45222 ± 0.1330 4.74 −	32.82980 ± 0.0000 1.84 =	32.50827 ± 0.1229 3.94 −	32.41975 ± 0.0852 5.64 −	32.80988 ± 0.0628 3.14 =	32.82980 ± 0.0000 1.44
Overall f-rank w/l/t	65.3 0/9/0	50.3 0/7/2	47.3 0/9/0	18.2 0/2/7	44.1 0/9/0	59.7 0/9/0	25.7 0/3/6	13.6



(a) CEC2014 F3 (b) CEC2014 F12 (c) CEC2014 F17 (d) CEC2014 F19 (e) CEC2017 F11

(f) CEC2017 F19 (g) CEC2014 F25 (h) CEC2014 F30 (i) CEC2017 F27 (j) CEC2017 F30

Fig. 11. Comparison of the population diversity of OSCOA and SHADE.



(a) Results of coarse-grained experiments (b) Results of fine-grained experiments

Fig. 12. Overall f-rank for different LN_c and UN_c combinations.

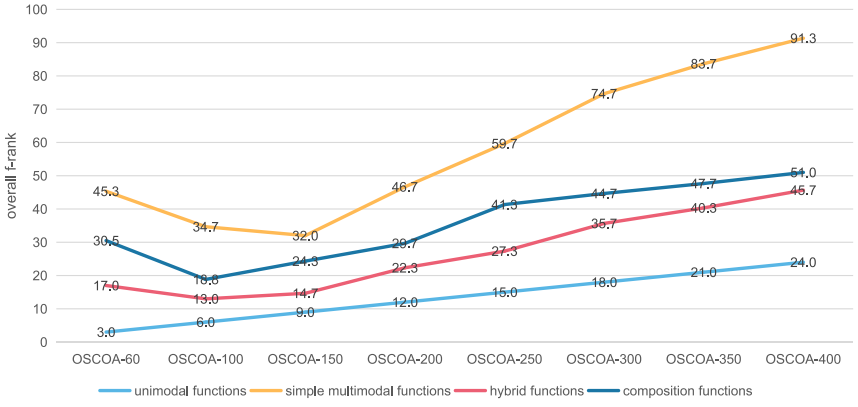


Fig. 13. Overall f-rank of OSCOA with different population sizes.

Wilcoxon rank-sum test show that OSCOA significantly outperforms COA, ADE, CCOA and FDB-SOS on all images tested, and OSCOA outperforms BTLBO, DEET and SHADE on 7, 2 and 3 images, respectively. In summary, the OSCOA algorithm demonstrates advantages in image segmentation problems.

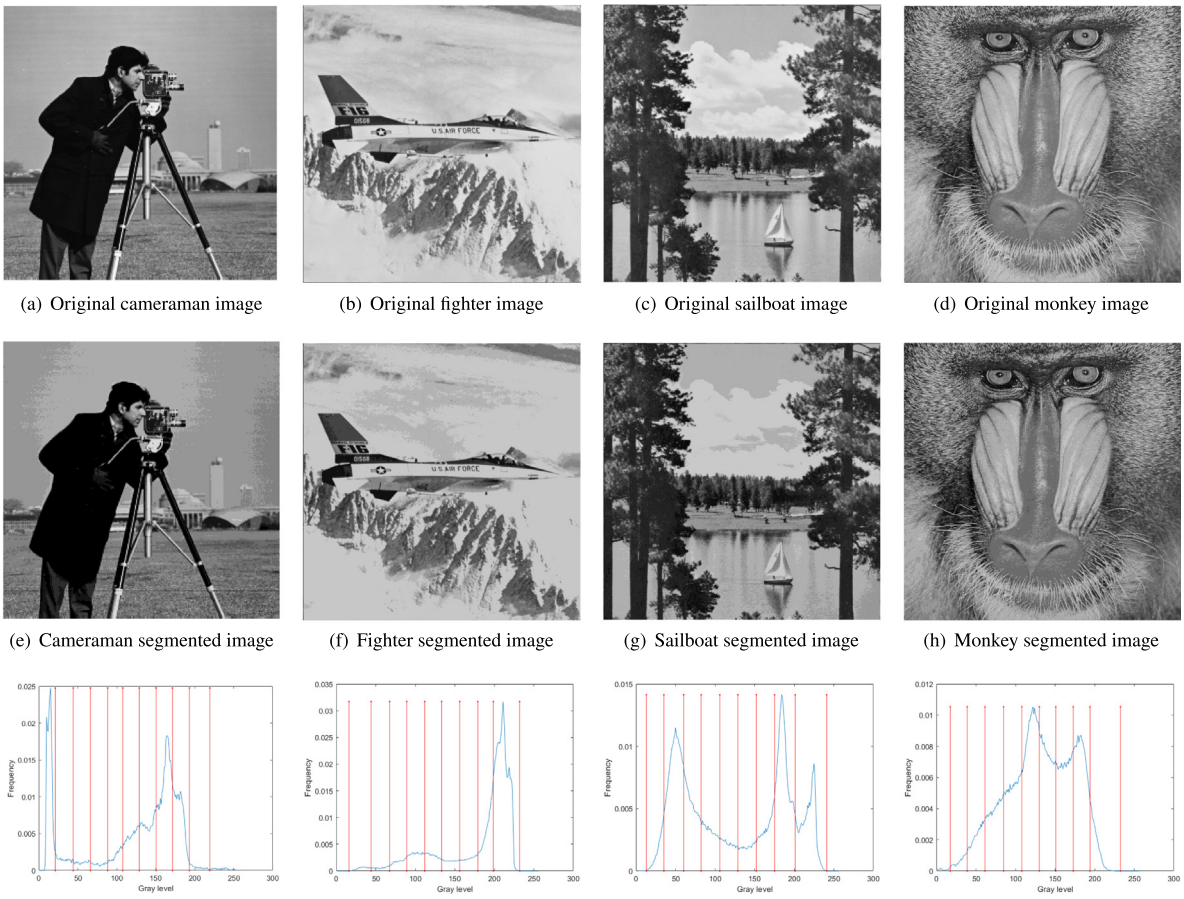
9. Application II: Deployment problems of wireless sensor networks

Wireless sensor networks (WSNs) are composed of homogeneous sensors, can monitor and collect information in the network coverage area in real time. WSNs as an important information gathering tool have been widely used in smart home, disaster prediction, agricultural production and environmental monitoring [47].

9.1. Mathematical model of WSNs coverage rate

For WSNs problem, the deployment of sensor nodes is the core important issue [48,49]. Its goal is to maximize the coverage of the target area with a limited number of sensors. Hence, it is a typical optimization problem.

Suppose that a number of v isomorphic sensors are deployed in the 2-dimensional plane P. These sensors are represented using the set $C = \{c_1, c_2, \ldots, c_v\}$. Each sensor in the set has the same sensing radius R_s and communication radius $R_c = 2R_S$, and the



(a) Original cameraman image (b) Original fighter image (c) Original sailboat image (d) Original monkey image

(e) Cameraman segmented image (f) Fighter segmented image (g) Sailboat segmented image (h) Monkey segmented image

(i) Distribution of thresholds for segmented cameraman image (j) Distribution of thresholds for segmented fighter image (k) Distribution of thresholds for segmented sailboat image (l) Distribution of thresholds for segmented monkey image

Fig. 14. Segmentation results of some images.

coordinate of the ith sensor is denoted as $c_i = (x_i, y_i)$. The detection probability of c_i for any point z can be expressed as follows.

$$P(c_i, z) = \begin{cases} 1 & d(c_i, z) \le R_s - R_e \\ \exp\left(\frac{-\lambda_1 \alpha_1{}^{\beta_1}}{\alpha_2{}^{\beta_2} + \lambda_2}\right) & R_s - R_e < d(c_i, z) < R_s + R_e \\ 0 & otherwise \end{cases} \tag{40}$$

$$\alpha_1 = R_e - R_s + d(c_i, z) \tag{41}$$

$$\alpha_2 = R_e + R_s - d(c_i, z) \tag{42}$$

where $d(c_i, z)$ denotes the Euclidean distance between c_i and z, $R_e (0 < R_e < R_s)$ denotes the reliability parameter, β_1, β_2, λ_1 and λ_2 are the measurement parameter of sensors. The joint detection probability of any point z is represented by Eq. (43)

$$P(C, z) = 1 - \prod_{i=1}^{v} (1 - P(c_i, z)) \tag{43}$$

A plane contains an infinite number of points, thus, it is impossible to calculate the detection probability of all points. Therefore, in this experiment the target areas are gridded into $m \times n$ pixel points, and only the pixel points are detected. Therefore the coverage rate of the target area is expressed as Eq. (44).

$$P_{\text{cov}} = \frac{\sum_{x=1}^{m} \sum_{y=1}^{n} P(C, (x, y))}{m \times n} \tag{44}$$

9.2. Experiment and analysis

This subsection conducts optimization experiments on the deployment of WSNs, and four deployment scenarios are designed for this experiment. The scenarios 1 and 2 are for deploying 15 and 20 sensors in a 30 × 30 area, respectively. And scenarios 3 and 4 are for deploying 30 and 40 sensors in a 100 × 100 area, respectively. Each algorithm is run 30 times independently and MaxFEs is set to 10^5. The experimental results are presented in Table 15.

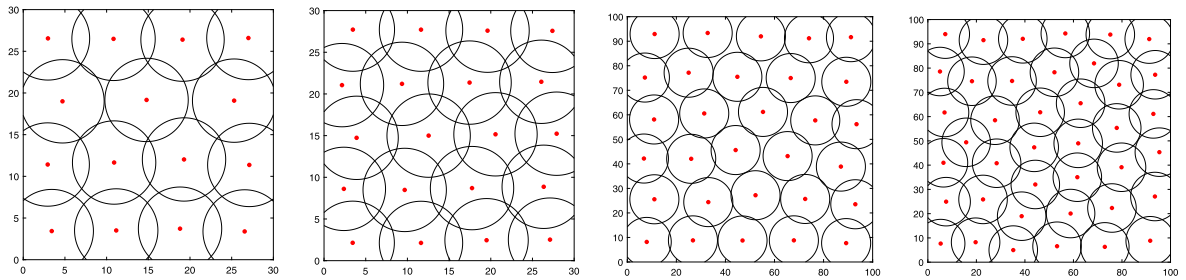
In the Table 15, the OSCOA algorithm significantly outperforms COA, ADE, CCOA and FDB-SOS in all four scenarios. Although the OSCOA algorithm does not exhibit a significant advantage over BTLBO, DEET, and SHADE, it achieves first place in scenario 1 and 4 in the Friedman test. In addition, Fig. 15 illustrates optimal sensor node deployment schemes obtained by OSCOA in four simulation experiments. In summary, OSCOA has a competitive performance for the deployment of WSNs.

10. Conclusion

This paper introduces a novel variant of the COA algorithm, named OSCOA. The OSCOA algorithm incorporates the Indicator of the Optimization State (IOS) to estimate the population optimization state. The IOS is integrated into three key optimization components of COA, resulting in the proposal of three optimization mechanisms: the dynamic pack management mechanism, the multiple growth strategy mechanism, and the parent selection mechanism.

Table 15
The comparison results of OSCOA with other algorithms on deployment problems of wireless sensor networks.

Function	COA	BTLBO	ADE	DEET	CCOA	FDB-SOS	SHADE	OSCOA
Scenario 1	0.84016 ± 0.0045 7.7 −	0.92565 ± 0.0014 3.5 =	0.89849 ± 0.0112 5.7 −	0.92659 ± 0.0011 3.4 =	0.86824 ± 0.0074 6.7 −	0.87739 ± 0.0093 5.3 −	0.92663 ± 0.0011 2.3 =	0.92685 ± 0.0011 1.6
Scenario 2	0.91750 ± 0.0062 8.0 −	0.97749 ± 0.0003 4.4 =	0.95765 ± 0.0064 5.3 −	0.97767 ± 0.0006 3.3 =	0.93816 ± 0.0063 6.4 −	0.94314 ± 0.0069 6.0 −	0.97895 ± 0.0066 1.3 =	0.97772 ± 0.0005 1.6
Scenario 3	0.74131 ± 0.0059 7.5 −	0.87860 ± 0.0015 3.2 =	0.81829 ± 0.0212 4.4 −	0.87974 ± 0.0024 1.7 =	0.76892 ± 0.0100 7.1 −	0.78154 ± 0.0121 6.1 −	0.87937 ± 0.0111 2.3 =	0.87824 ± 0.0001 3.7
Scenario 4	0.84210 ± 0.0029 7.6 −	0.95764 ± 0.0015 2.9 =	0.90445 ± 0.0107 4.7 −	0.95869 ± 0.0032 1.8 =	0.84102 ± 0.0113 7.8 −	0.87999 ± 0.0075 5.8 −	0.91937 ± 0.0040 3.6 =	0.95875 ± 0.0004 1.6
Overall f-rank w/l/t	30.8 0/4/0	14.0 0/0/4	20.1 0/4/0	10.2 0/0/4	28.0 0/4/0	23.2 0/4/0	9.5 0/0/4	8.5



(a) scenario 1(converage rate %92.7) (b) scenario 2(converage rate %97.8) (c) scenario 3(converage rate %87.8) (d) scenario 4(converage rate %95.8)

Fig. 15. Sensor distributions optimized by OSCOA.

The dynamic pack management mechanism utilizes IOS to dynamically adjust the number of packs, thereby preventing the algorithm from getting trapped in local optima. With the multiple growth strategy mechanism, the IOS promptly selects the suitable search strategy (exploration or exploitation) during the evolutionary process, avoiding aimless searching. The parent selection mechanism, guided by IOS, ensures the selection of appropriate parent coyotes for generating pups, thereby significantly enhancing the algorithm's overall global search efficiency.

Strategies within each of the three mechanisms are chosen using probabilities, eliminating the need for threshold settings and increasing fault tolerance. Moreover, the usage of three independent selection modes enriches the algorithm's search patterns.

To validate the effectiveness of the OSCOA algorithm, extensive experiments are conducted on CEC2014, CEC2017, and CEC2022 benchmark suites, comparing OSCOA with seven efficient optimizers. The experimental results demonstrate that the OSCOA algorithm exhibits competitive and even superior performance, particularly in tackling complex problems such as hybrid problems. Furthermore, the results indicate that OSCOA maintains good scalability when addressing high-dimensional problems.

Moreover, two real-world applications including the image segmentation and the deployment of wireless sensor networks are employed to verify the capability of the OSCOA algorithm in addressing real-world problems. Experimental and statistical results reflect that the proposed OSCOA algorithm has a strong convergence performance and robust global search capability in practical scenarios.

The proposed OSCOA algorithm performs high efficiency and has been successfully used to solve two real-world optimization problems. However, the OSCOA algorithm still has some shortcomings and limitations. Although the IOS technique provides fast feedback on the population, its implementation demands additional computational resources. Therefore, further exploration is necessary to precisely ascertain the population optimization state while conserving computational resources.

This study drive the exploration and application of flexible and precise estimation methods of population optimization states. Furthermore, it opens up possibilities for the future application of OSCOA algorithms in multi-objective optimization and other real-world contexts.

CRediT authorship contribution statement

Qingke Zhang: Conceptualization, Software, Resources, Supervision, Writing – original draft, Writing – review, Project administration, Funding acquisition. **Xianglong Bu:** Conceptualization, Methodology, Software, Investigation, Formal analysis, Visualization, Writing – original draft. **Zhi-Hui Zhan:** Resources, Writing – review. **Junqing Li:** Resources, Writing – review. **Huaxiang Zhang:** Resources, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors are unable or have chosen not to specify which data has been used.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (62006144), the major fundamental research project of Shandong, China (No. ZR2019ZD03), the Taishan Scholar Project of Shandong, China (No. ts20190924) and the Natural Science Foundation of Shandong Province, China (NO. ZR2022MF346).

References

[1] I.H. Osman, J.P. Kelly, Meta-heuristics theory and applications, J. Oper. Res. Soc. 48 (6) (1997) 657, http://dx.doi.org/10.1057/palgrave.jors.2600781.

[2] F. Yang, P. Wang, Y. Zhang, L. Zheng, J. Lu, Survey of swarm intelligence optimization algorithms, in: 2017 IEEE International Conference on Unmanned Systems (ICUS), 2017, pp. 544–549, http://dx.doi.org/10.1109/ICUS.2017.8278405.

[3] D. Karaboga, B. Akay, A comparative study of Artificial Bee Colony algorithm, Appl. Math. Comput. 214 (1) (2009) 108–132, http://dx.doi.org/10.1016/j.amc.2009.03.090.

[4] E. Cuevas, M. Cienfuegos, D. Zaldívar, M. Pérez-Cisneros, A swarm optimization algorithm inspired in the behavior of the social-spider, Expert Syst. Appl. 40 (16) (2013) 6374–6384, http://dx.doi.org/10.1016/j.eswa.2013.05.041.

[5] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67, http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

[6] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, S.M. Mirjalili, Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191, http://dx.doi.org/10.1016/j.advengsoft.2017.07.002.

[7] A. Vijaya Lakshmi, P. Mohanaiah, WOA-TLBO: Whale optimization algorithm with Teaching-learning-based optimization for global optimization and facial emotion recognition, Appl. Soft Comput. 110 (2021) 107623, http://dx.doi.org/10.1016/j.asoc.2021.107623.

[8] H. Su, D. Zhao, F. Yu, A.A. Heidari, Y. Zhang, H. Chen, C. Li, J. Pan, S. Quan, Horizontal and vertical search artificial bee colony for image segmentation of COVID-19 X-ray images, Comput. Biol. Med. 142 (2022) 105181, http://dx.doi.org/10.1016/j.compbiomed.2021.105181.

[9] H. Feng, W. Ma, C. Yin, D. Cao, Trajectory control of electro-hydraulic position servo system using improved PSO-PID controller, Autom. Constr. 127 (2021) 103722, http://dx.doi.org/10.1016/j.autcon.2021.103722.

[10] E. Kaymaz, S. Duman, U. Guvenc, Optimal power flow solution with stochastic wind power using the Lévy coyote optimization algorithm, Neural Comput. Appl. 33 (12) (2021) 6775–6804, http://dx.doi.org/10.1007/s00521-020-05455-9.

[11] A. Pradhan, S.K. Bisoy, A. Das, A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment, J. King Saud Univ. - Comput. Inf. Sci. (2021) http://dx.doi.org/10.1016/j.jksuci.2021.01.003.

[12] J. Pierezan, L. Dos Santos Coelho, Coyote optimization algorithm: A new metaheuristic for global optimization problems, in: 2018 IEEE Congress on Evolutionary Computation (CEC), 2018, pp. 1–8, http://dx.doi.org/10.1109/CEC.2018.8477769.
[13] J. Pierezan, G. Maidl, E. Massashi Yamao, L. dos Santos Coelho, V. Cocco Mariani, Cultural coyote optimization algorithm applied to a heavy duty gas turbine operation, Energy Convers. Manage. 199 (2019) 111932, http://dx.doi.org/10.1016/j.enconman.2019.111932.
[14] Z. Yuan, W. Wang, H. Wang, A. Yildizbasi, Developed Coyote Optimization Algorithm and its application to optimal parameters estimation of PEMFC model, Energy Rep. 6 (2020) 1106–1117, http://dx.doi.org/10.1016/j.egyr.2020.04.032.
[15] J. Pierezan, L. dos Santos Coelho, V. Cocco Mariani, E. Hochsteiner de Vasconcelos Segundo, D. Prayogo, Chaotic coyote algorithm applied to truss optimization problems, Comput. Struct. 242 (2021) 106353, http://dx.doi.org/10.1016/j.compstruc.2020.106353.
[16] H. Tong, Y. Zhu, J. Pierezan, Y. Xu, L.d.S. Coelho, Chaotic coyote optimization algorithm, J. Ambient Intell. Humaniz. Comput. 13 (5) (2022) 2807–2827, http://dx.doi.org/10.1007/s12652-021-03234-5.
[17] E. Kaymaz, S. Duman, U. Guvenc, Optimal power flow solution with stochastic wind power using the Lévy coyote optimization algorithm, Neural Comput. Appl. 33 (12) (2021) 6775–6804, http://dx.doi.org/10.1007/s00521-020-05455-9.
[18] L. Li, L. Sun, Y. Xue, S. Li, X. Huang, R.F. Mansour, Fuzzy multilevel image thresholding based on improved coyote optimization algorithm, IEEE Access 9 (2021) 33595–33607, http://dx.doi.org/10.1109/ACCESS.2021.3060749, Conference Name: IEEE Access.
[19] T.D. Pham, T.T. Nguyen, B.H. Dinh, Find optimal capacity and location of distributed generation units in radial distribution networks by using enhanced coyote optimization algorithm, Neural Comput. Appl. 33 (9) (2021) 4343–4371, http://dx.doi.org/10.1007/s00521-020-05239-1.
[20] Z. Jin, X. Sun, G. Lei, Y. Guo, J. Zhu, Sliding mode direct torque control of SPMSMs based on a hybrid wolf optimization algorithm, IEEE Trans. Ind. Electron. 69 (5) (2022) 4534–4544, http://dx.doi.org/10.1109/TIE.2021.3080220, Conference Name: IEEE Transactions on Industrial Electronics.
[21] X. Zhang, Y. Jiang, S. Liu, Hybrid coyote optimization algorithm with grey wolf optimizer and its application to clust-ering optimization, Acta Automat. Sinica 48 (6) (2022) 1–17, http://dx.doi.org/10.16383/j.aas.c190617.
[22] S. Wu, J. Jiang, Y. Yan, W. Bao, Y. Shi, Improved coyote algorithm and application to optimal load forecasting model, Alex. Eng. J. 61 (10) (2022) 7811–7822.
[23] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61, http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.
[24] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, Knowl.-Based Syst. 191 (2020) 105190, http://dx.doi.org/10.1016/j.knosys.2019.105190.
[25] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Gener. Comput. Syst. 97 (2019) 849–872, http://dx.doi.org/10.1016/j.future.2019.02.028.
[26] Anita, A. Yadav, AEFA: Artificial electric field algorithm for global optimization, Swarm Evol. Comput. 48 (2019) 93–108, http://dx.doi.org/10.1016/j.swevo.2019.03.013.
[27] Z.-H. Zhan, J. Zhang, Y. Li, H.S.-H. Chung, Adaptive particle swarm optimization, IEEE Trans. Syst. Man Cybern. B 39 (6) (2009) 1362–1381, http://dx.doi.org/10.1109/TSMCB.2009.2015956.
[28] Z.-H. Zhan, Z.-J. Wang, H. Jin, J. Zhang, Adaptive distributed differential evolution, IEEE Trans. Cybern. 50 (11) (2020) 4633–4647, http://dx.doi.org/10.1109/TCYB.2019.2944873.
[29] W.-J. Yu, M. Shen, W.-N. Chen, Z.-H. Zhan, Y.-J. Gong, Y. Lin, O. Liu, J. Zhang, Differential evolution with two-level parameter adaptation, IEEE Trans. Cybern. 44 (7) (2014) 1080–1099, http://dx.doi.org/10.1109/TCYB.2013.2279211.
[30] Y. Li, G. Li, Differential evolutionary algorithm with an evolutionary state estimation method and a two-level selection mechanism, Soft Comput. 24 (15) (2020) 11561–11581, http://dx.doi.org/10.1007/s00500-019-04621-z.
[31] Q. Yang, J.-Q. Yan, X.-D. Gao, D.-D. Xu, Z.-Y. Lu, J. Zhang, Random neighbor elite guided differential evolution for global numerical optimization, Inform. Sci. 607 (2022) 1408–1438, http://dx.doi.org/10.1016/j.ins.2022.06.029.
[32] Z.-J. Wang, Z.-H. Zhan, S. Kwong, H. Jin, J. Zhang, Adaptive granularity learning distributed particle swarm optimization for large-scale optimization, IEEE Trans. Cybern. 51 (3) (2021) 1175–1188, http://dx.doi.org/10.1109/TCYB.2020.2977956.
[33] N.H. Awad, M.Z. Ali, P.N. Suganthan, J.J. Liang, B.Y. Qu, Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Nanyang Technological University, Singapore and Jordan University of Science and Technology, Jordan and Zhengzhou University, Zhengzhou China, 2016.
[34] F. Wilcoxon, S. Katti, R.A. Wilcox, Critical Values and Probability Levels for the Wilcoxon Rank Sum Test and the Wilcoxon Signed Rank Test, American Cyanamid Company, 1963.
[35] G.G. Tejani, V.J. Savsani, V.K. Patel, S. Mirjalili, Truss optimization with natural frequency bounds using improved symbiotic organisms search, Knowl.-Based Syst. 143 (2018) 162–178, http://dx.doi.org/10.1016/j.knosys.2017.12.012.
[36] A. Taheri, K. RahimiZadeh, R.V. Rao, An efficient Balanced Teaching-Learning-Based optimization algorithm with Individual restarting strategy for solving global optimization problems, Inform. Sci. 576 (2021) 68–104, http://dx.doi.org/10.1016/j.ins.2021.06.064.
[37] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for Differential Evolution, in: 2013 IEEE Congress on Evolutionary Computation, 2013, pp. 71–78, http://dx.doi.org/10.1109/CEC.2013.6557555.
[38] R. Rao, V. Savsani, D. Vakharia, Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Comput. Aided Des. 43 (3) (2011) 303–315, http://dx.doi.org/10.1016/j.cad.2010.12.015.
[39] F.J. Solis, R.J.-B. Wets, Minimization by random search techniques, Math. Oper. Res. 6 (1) (1981) 19–30, http://dx.doi.org/10.1287/moor.6.1.19.
[40] S. Xian, X. Feng, Meerkat optimization algorithm: A new meta-heuristic optimization algorithm for solving constrained engineering problems, Expert Syst. Appl. 231 (2023) 120482, http://dx.doi.org/10.1016/j.eswa.2023.120482.
[41] R. Tanabe, A.S. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in: 2014 IEEE Congress on Evolutionary Computation (CEC), 2014, pp. 1658–1665, http://dx.doi.org/10.1109/CEC.2014.6900380.
[42] J. Kuruvilla, D. Sukumaran, A. Sankar, S.P. Joy, A review on image processing and image segmentation, in: 2016 International Conference on Data Mining and Advanced Computing (SAPIENCE), 2016, pp. 198–203, http://dx.doi.org/10.1109/SAPIENCE.2016.7684170.
[43] M.H. Ryalat, D. Emmens, M. Hulse, D. Bell, Z. Al-Rahamneh, S. Laycock, M. Fisher, Evaluation of particle swarm optimisation for medical image segmentation, in: J. Świątek, J.M. Tomczak (Eds.), Advances in Systems Science, Springer International Publishing, Cham, 2017, pp. 61–72.
[44] Y.-H. Chiu, K.-L. Chung, W.-N. Yang, Y.-H. Huang, C.-H. Liao, Parameter-free based two-stage method for binarizing degraded document images, Pattern Recognit. 45 (12) (2012) 4250–4262, http://dx.doi.org/10.1016/j.patcog.2012.02.023.
[45] T. Pun, Entropic thresholding, a new approach, Comput. Graph. Image Process. 16 (3) (1981) 210–239, http://dx.doi.org/10.1016/0146-664X(81)90038-1.
[46] J. Kapur, P. Sahoo, A. Wong, A new method for gray-level picture thresholding using the entropy of the histogram, Comput. Vis. Graph. Image Process. 29 (3) (1985) 273–285, http://dx.doi.org/10.1016/0734-189X(85)90125-2.
[47] A.M. Khedr, W. Osamy, A. Salim, Distributed coverage hole detection and recovery scheme for heterogeneous wireless sensor networks, Comput. Commun. 124 (2018) 61–75, http://dx.doi.org/10.1016/j.comcom.2018.04.002.
[48] A.M. Khedr, W. Osamy, D.P. Agrawal, Perimeter discovery in wireless sensor networks, J. Parallel Distrib. Comput. 69 (11) (2009) 922–929, http://dx.doi.org/10.1016/j.jpdc.2009.08.002.
[49] L. Rui, M. Yuangbing, J. Chai, Deployment strategy of wireless sensor network based on reformative sparrow search algorithm, Microelectron. Comput. 39 (4) (2022) 65–74.