Applied Intelligence (2022) 52:7922–7964
https://doi.org/10.1007/s10489-021-02776-7

Rank-driven salp swarm algorithm with orthogonal opposition-based learning for global optimization

Zongshan Wang[1] · Hongwei Ding[1] · Zhijun Yang[1,2] · Bo Li[1] · Zheng Guan[1] · Liyong Bao[1]

Accepted: 16 August 2021 / Published online: 15 October 2021

Abstract

Salp swarm algorithm (SSA) is a relatively new and straightforward swarm-based meta-heuristic optimization algorithm, which is inspired by the flocking behavior of salps when foraging and navigating in oceans. Although SSA is very competitive, it suffers from some limitations including unbalanced exploration and exploitation operation, slow convergence. Therefore, this study presents an improved version of SSA, called OOSSA, to enhance the comprehensive performance of the basic method. In preference, a new opposition-based learning strategy based on optical lens imaging principle is proposed, and combined with the orthogonal experimental design, an orthogonal lens opposition-based learning technique is designed to help the population jump out of a local optimum. Next, the scheme of adaptively adjusting the number of leaders is embraced to boost the global exploration capability and improve the convergence speed. Also, a dynamic learning strategy is applied to the canonical methodology to improve the exploitation capability. To confirm the efficacy of the proposed OOSSA, this paper uses 26 standard mathematical optimization functions with various features to test the method. Alongside, the performance of the proposed methodology is validated by Wilcoxon signed-rank and Friedman statistical tests. Additionally, three well-known engineering optimization problems and unknown parameters extraction issue of photovoltaic model are applied to check the ability of the OOSA algorithm to obtain solutions to intractable real-world problems. The experimental results reveal that the developed OOSSA is significantly superior to the standard SSA, currently popular SSA-based algorithms, and other state-of-the-artmeta-heuristic algorithms for solving numerical optimization, real-world engineering optimization, and photovoltaic model parameter extraction problems. Finally, an OOSSA-based path planning approach is developed for creating the shortest obstacle-free route for autonomous mobile robots. Our introduced method is compared with several successful swarm-based metaheuristic techniques in five maps, and the comparative results indicate that the suggested approach can generate the shortest collision-free trajectory as compared to other peers.

Keywords Salp swarm algorithm · Lens opposition-based learning · Orthogonal experiment design · Dynamic learning · Global optimization · Engineering design optimization · Photovoltaic models · Parameter extraction · Robot path planning

1 Introduction

Optimization problems are common in the field of engineering and need to be solved. The optimization method uses mathematical methods to achieve a variety of search strategies to optimize the objective function. However, for complex engineering problems, due to the limitations of computing time, algorithm conditions, and other factors, it is challenging to obtain the optimal solution. The heuristic optimization method designs various search strategies based on the intuitive or prior knowledge of the problem, which is of practical significance for solving complex engineering optimization problems. In recent years, numerous meta-heuristic techniques have been developed and applied to realize real-world problems. Because of its simplicity, efficiency, flexibility and low

✉ Zongshan Wang
wzs_ynu@163.com

Hongwei Ding
dhw694513739@sina.com

Zhijun Yang
wzs375888295@163.com

[1] School of Information Science and Engineering, Yunnan University, Kunming 650500, China

[2] Yunnan Education Department, Kunming 650500, China

computational cost, meta-heuristic algorithms for solving tricky problems have become a hot research topic [1–3]. In [4], a genetic-moth swarm algorithm was proposed for determining the suitable locations and capacities of distributed generations in distributed system and successfully reduced electrical power loss without violating security constrains. In [5], a PSOGWO algorithm was introduced by combining the grey wolf algorithm and particle swarm algorithm for parameter estimation of photovoltaic model. Compared with some well-performance methods, the experimental results demonstrated that the developed algorithm outperform all competitors. Moreover, metaheuristic techniques have been used extensively in PV model parameter estimation issues for a long time and have provided accurate results that have contributed to the development of this field [6–10]. Meanwhile, metaheuristic techniques have become increasingly prevalent in the field of Wireless Sensor Networks (WSNs). For example, in [11], a modified version of artificial bee colony algorithm was developed for optimal clustering in WSNs. The proposed approach effectively handles the delicate balancing of the network load and extends the network lifetime, the promising performance proves that the proposed method is an effective clustering tool for wireless sensor networks. In [12], sine-cosine algorithm and harris hawks optimization were integrated to form a better performing algorithm for low and high-dimensional feature selection. Compared with other recognized hybrid algorithms and state-of-the-art feature selection approaches, the suggested method has competitive performance and is useful for real-world applications. Furthermore, in recently, many metaheuristic algorithm-based feature selection approaches have been developed and provide superior performance [13–16]. In [17], an improved fractional-order cuckoo search algorithm was proposed for COVID-19 X-ray images classification. The outcomes verify that the involved method performs well in terms of classification accuracy. In addition, many metaheuristic algorithms contributed to battle against the coronavirus disease, which greatly alleviated the work pressure of doctors while gaining more treatment time for patients [18–21]. In [22], a novel particle swarm optimization algorithm was developed for the path planning and obstacle avoidance problems in autonomous mobile robots. The experimental results demonstrate that the proposed approach can generate an optimal collision-free route for mobile robots. Besides, a variety of nature-inspired swarm-based techniques have been employed to establish a safe and reasonable route for mobile robots, such as artificial bee colony [23], chemical reaction optimization [24], bat algorithm [25], ant colony optimization [26], and cuckoo search algorithm [27]. Due to the powerful ability to solve real-world problems, metaheuristic techniques are growing in popularity in many research areas, including image processing [28], economic dispatch [29], design PID controller [30], cloud computing [31], and environmental engineering [32] . According to “no free lunch” (NFL) theorem [33], there is no one method that can handle all optimization problems well. In other words, an optimization algorithm may achieve satisfactory results on some optimization problems, but it may perform poorly on other different problems. Therefore, the study of meta-heuristic algorithm has a strong practical significance.

Nature-inspired meta-heuristic algorithms solve optimization problems by building biological or physical phenomena as mathematical models, which can be divided into four classes: evolution-based, physics-based, human-based and population-based methods. As the most popular meta-heuristic algorithms, the population-based method simulates the social behavior of fauna in nature. The most representative algorithm is Particle Swarm Optimization (PSO) presented by Kennedy and Eberhart [34], which is inspired by the flocking behavior of birds. PSO uses several particles to travel around in the search space to find the optimal solution. Another popular population-based method is Firefly Algorithm (FA), first proposed by Yang [35], which is inspired by the swarming behavior of fireflies in the nature. In fact, the collective intelligence of fireflies in finding food through specific ways of communication is the primary insight for this optimizer. It uses several fireflies to move in the search space to find the optimal solution. In addition, the Artificial Bee Colony (ABC) algorithm [36] was proposed by Karaboga et al. inspired by the cooperative foraging behavior of bees in the colony, and has been successfully applied in many disciplines. In recent years, many novel swarm intelligence based algorithms have been presented, such as: Krill Herd (KH) [37], Bat Algorithm (BA) [38], Coyote Optimization Algorithm (COA) [39], Bacterial Foraging Optimization (BFO) Algorithm [40], Grey Wolf Optimization (GWO) [41], Fruit Fly Optimization Algorithm (FOA) [42], Spotted Hyena Optimizer (SHO) [43], Poor and Rich Optimization Algorithm (PRO) [44], Barnacles Mating Optimizer (BMO) [45], Multi-Verse Optimizer (MVO) [46], Animal Migration Optimization (AMO) [47], Dragonfly Algorithm (DA) [48], Whale Optimization Algorithm (WOA) [49], Flower Pollination Algorithm (FPA) [50], Galactic Swarm Optimization (GSO) [51], Meerkat-Inspired Algorithm (MIA) [52], Farmland Fertility Algorithm (FFA) [53], Pathfinder Algorithm (PFA) [54], Falcon Optimization Algorithm (FOA) [55], Harris Hawks Optimizer (HHO) [56], Owl Optimization Algorithm (OOA) [57], Manta Ray Foraging Optimization (MRFO) [58], Monarch Butterfly Optimization (MBO) [59], Pity Beetle Algorithm (PBA) [60], Earthworm optimization algorithm (EOA) [61] and Salp Swarm Algorithm (SSA) [62]. Among these algorithms, SSA has been widely studied in recent years because of its distinctive perspective.

The SSA algorithm is a novel swarm-based stochastic optimization algorithm proposed by Mirjalili et al. in 2017,

which is inspired by the phenomenon that the salps follow each other to form a chain when they are navigating and foraging in the ocean. Through the process of leading and following, an efficient optimization scheme is constructed. The SSA algorithm has been applied to various optimization problems because of its simple mechanism, dynamic nature and strong global search ability. Such as extracting the parameters of photovoltaic system cell [63, 64], optimization of software-defined networks [65], train neural networks [66, 67], optimize parameters of soil water retention [68], tariff optimization in electrical systems [69], image segmentation [70,71], target localization [72], optimal power flow problem [73], estimation of optimal parameters of polymer exchange membrane fuel cells [74], feature selection [75, 76, 77, 78, 79, 80] and others. Although the SSA algorithm is very competitive, it still suffers from some limitations such as poor convergence, unbalanced exploration and exploitation capacities, which may lead to local optimum stagnation when solving some intractable optimization problems. To address these drawbacks, many scholars try to adjust the exploration or exploitation strategies to enhance the comprehensive performance of the canonical SSA.

Qais et al. [81] modified the control parameters of SSA and introduced a random parameter into the follower position update formula to improve the convergence performance. Gupta et al. [82] proposed a modified version of SSA. The concept of opposition-based learning (OBL) is introduced to improve the ability of canonical SSA to escape local optimal, and the levy flight strategy is introduced to help explore the search space. The comprehensive effect of OBL and levy flight mechanisms strengthens the exploration-exploitation balance during the search process. Zhang et al. [83] proposed an enhanced SSA, which promotes the overall performance of the algorithm by embedding multiple strategies into original SSA. Among all the introduced strategies, OBL helps to enrich the diversity of the initial population, orthogonal learning helps to promote the probability of avoiding local optimal, and the concept of quadratic interpolation can improve the convergence accuracy. Ren et al. [84] proposed an improved version of SSA, which introduces adaptive weight and levy flight strategy into the traditional SSA to promote the comprehensive performance of the algorithm. Among them, the adaptive weight expands the exploration range, while the levy flight strategy strengthens the exploitation of the solution space. Gholami et al. [85] introduced a mutation mechanism to help the algorithm jump out of the local optimum, which improved the problem of insufficient accuracy of the algorithm. Sayed et al. [86] proposed a chaos SSA, that introduces ten different chaotic maps when updating the control parameters of SSA, so as to solve the problem that the algorithm is easy to sink into local optimum. Wu et al. [87] proposed a new version of SSA, which reduced the probability of local optimal stagnation by introducing dynamic weight factor and adaptive mutation strategy. Ibrahim et al. [88] proposed an improved SSA, called SSAPSO, which improves the flexibility of the algorithm in the exploration phase by hybridizing SSA and PSO. Singh et al. [89] proposed a novel method that mixes SSA and Sine Cosine Algorithm (SCA) [90], which enhances the exploratory ability of the algorithm. Li et al. combined Gravitational Search Algorithm (GSA) [91] with SSA to promote the probability of escaping the local optimum. Among them, GSA and SSA have a clear division of labor, the former is responsible for global exploration, the latter is responsible for local exploitation. Yang et al. [92] proposed a memetic SSA. The convergence rate and convergence accuracy of the algorithm are enhanced by using memetic behavior. Syed et al. [93] proposed a weighted SSA, which improves the ability to balance exploitation and exploration during the search process by introducing the concept of optimal location weighted sum in the position update method. Bairathi et al. [94] proposed an OBL-based SSA variant, which enriches the diversity of the initial salp swarm by introducing the concept of OBL during the population initialization stage, thus promoting the convergence speed and enhancing the exploitation ability. Chen et al. [95] proposed an improved SSA. By introducing a non-linear attenuation factor that controls the search range, and the local search ability was improved, and a dynamic learning strategy was introduced to enhance the assistance of elite individuals to the leader. Although each of the SSA variants discussed above improves the performance of the original SSA to some extent, there are still some drawbacks, such as premature convergence, unbalanced global search and local development capabilities. In addition, the above-mentioned improved algorithms are only applicable to specific optimization problems, and none of them can solve all optimization problems well. Therefore, it is of great practical significance to propose more effective algorithms. Motivated by these two viewpoints, this study presents a modified SSA variant, called OOSSA, which improves the overall performance of the canonical SSA. The main contributions of this investigation are summarized as follows:

- An improved variant of SSA is presented and does not affect the structure of basic SSA.
- An adaptive technique is presented for the number of leaders in order to improve the global search ability and to promote the convergence performance.
- The lens opposition-based learning mechanism is proposed, and combined with orthogonal experimental design, an orthogonal opposition-based learning technique is designed to overcome the local optima stagnation.
- A ranking-based dynamic learning strategy is presented to enhance the local exploitation capability.

- The balance between global search and local development of the proposed methodology is held more stably.
- The efficiency of the proposed OOSSA is evaluated on a comprehensive set of 26 well-known benchmark functions with diverse difficulty levels, and compared with a variety of competitive metaheuristic techniques, including the canonical SSA, SSA variants, and other cutting-edge swarm-based methods.
- Three real-world engineering optimization problems and parameter extraction problem of PV model are used to verify the efficiency and practicality of the proposed algorithm.

An OOSSA-based path planning approach is developed for solving the path planning and collision avoidance problem in autonomous mobile robots.

The remainder of this article is structured as follows: Section 2 illustrates the preliminary knowledge. The proposed SSA-based algorithm is specified in detail in Section 3. The experimentation and verification of the proposed method on benchmark functions are performed in Section 4. In Section 5, the proposed method is used to solve three real-life engineering problems and determine the parameters of PV models. In Section 6, the proposed OOSSA-based path planning and obstacle avoidance approach is described and simulation and comparative studies are discussed. Finally, Section 7 concludes this investigation. The detailed flow of the article is shown in Fig. 1.

2 Preliminary knowledge

2.1 Salp swarm algorithm (SSA)

Salps are translucent, colloidal marine organisms that resemble jellyfish and move by inhaling and expelling seawater. Researchers mathematicized the chain structure of salps and proposed SSA. SSA divides salps into two types: leader and followers. The leader is at the front of the salp chain, while the followers are at the back, as shown in Fig. 2.

In SSA, the food source is the foraging target of the salp chain, which guides the leader to update the position. The mathematical model for updating the position of leader is as follows:

$$X_j^1 = \begin{cases} F_j + c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) & c_3 \geq 0.5 \\ F_j - c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) & c_3 < 0.5 \end{cases} \quad (1)$$

where $X1\,j$ and F_j indicate the positions of the leaders and food source in the jth dimension, respectively. ub_j and lb_j correspond to the lower and upper boundaries of the jth dimension in the search space, c_2 and c_3 are random variables evenly distributed in the interval [0,1], which determine the step size and moving direction respectively.

c_1 is an important parameter in SSA, which is called distance control factor. The c_1 parameter is defined as shown in Eq. (2).

$$c_1 = 2e^{-\left(\frac{4l}{T}\right)^2} \quad (2)$$

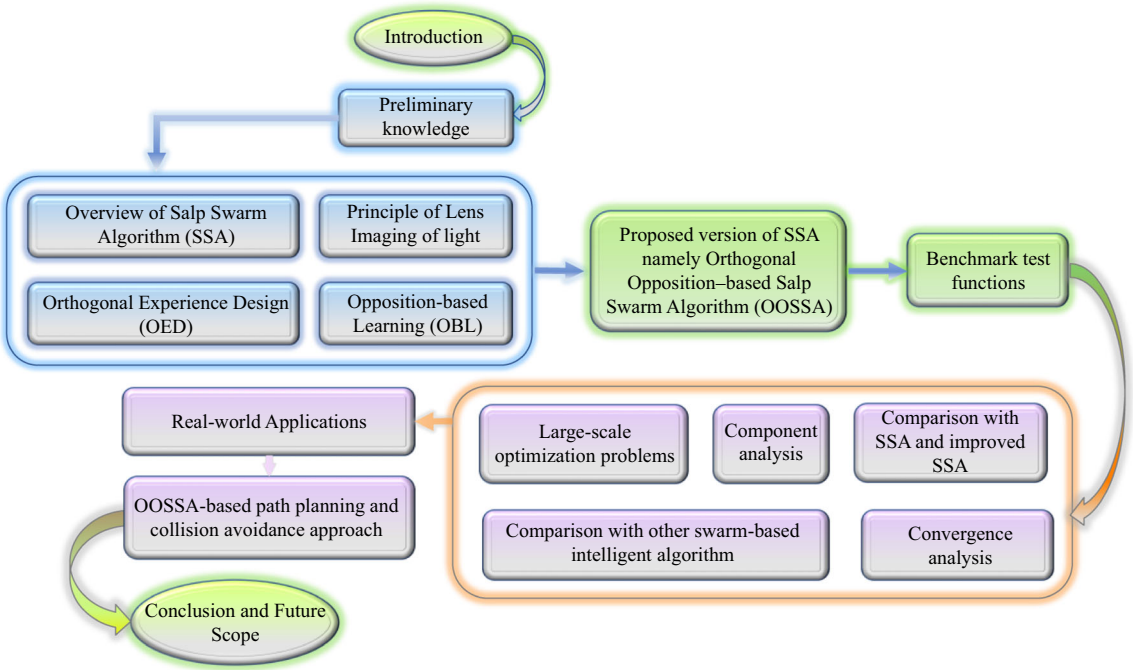


Fig. 1 Outline of the paper

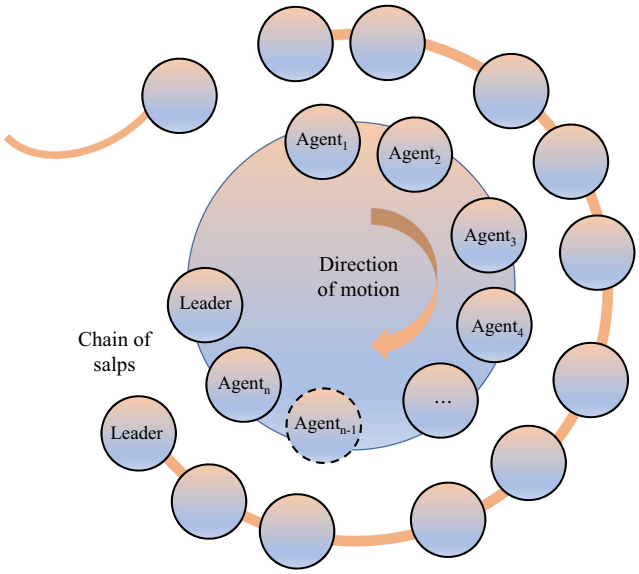


Fig. 2 An illustration of salp chain

where t is the current iteration and T is the maximum number of iterations.

To update the states of the followers, Eq. (3) is applied:

$$X_j^i = \frac{1}{2}\left(X_j^i + X_j^{i-1}\right) \tag{3}$$

where $i \geq 2$ and $Xi\,j$ shows the position of ith follower salp in the jth dimension search area.

The search process of SSA algorithm includes two stages: global exploration and local exploitation. During the initialization stage, the randomly generated initial population searches randomly in the search space to help the algorithm lock the optimal solution region. Then, the algorithm enters the exploitation stage and makes an accurate search in the limited area determined in the previous stage to improve the convergence accuracy. It should be pointed out that the distance control parameter c_1 has an important influence on the search process. In the early stage of evolution, the value of c_1 is large, which can help the algorithm to explore the whole solution space, parameter c_1 is decreased adaptively over the course of iterations, and the small value of c_1 can help the algorithm to carry out accurate exploitation in a specific search area. Since there is no prior knowledge to know the position of the food source, the global optimal solution obtained in each iteration is set as the current food source position.

Fig. 3 demonstrates the flowchart of SSA.

2.2 Principle of lens imaging of light

Convex lens imaging (LI) law is a kind of optical law, which means that an object is placed outside the focus and an inverted real image is formed on the other side of the convex lens [96], as shown in Fig. 4.

The mathematical model of LI can be obtained from Fig. 4 as follows:

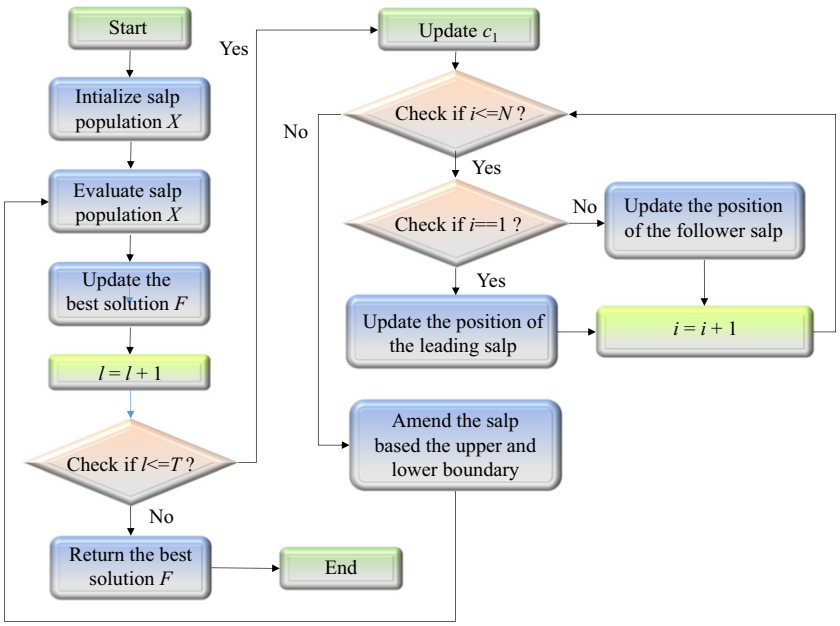
$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f} \tag{4}$$

where u and v are the object distance and image distance, respectively, and f is the focal length.

2.3 Opposition-based learning

Opposition-based learning (OBL) is an effective tool that can be used to improve the performance of stochastic search

Fig. 3 The flowchart of SSA



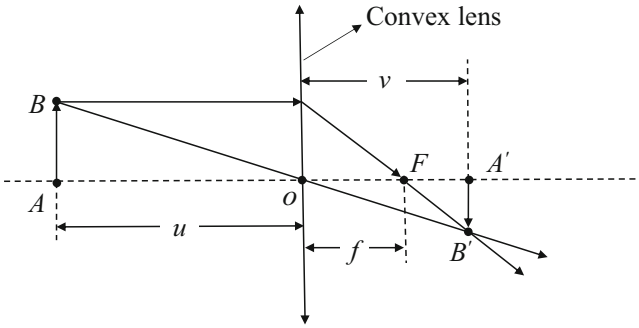


Fig. 4 The convex lens image of light

algorithms, first proposed by Tizhoosh [97] and has been app lied to many intelligent optimization techniques. Its main idea is to evaluate both the current feasible solution and the reverse solution, and choose the better one to use. Reference [98] shows that the reverse solution has a greater probability of approaching the global optimum than the current solution. Therefore, the OBL technique can effectively improve the comprehensive performance of the stochastic algorithm. The concept of OBL is defined as follows:

Definition 1 (Opposite number) [99] Let $x \in [a,b]$ be a real number and to calculate its opposite number $\tilde{x}$, Eq. (5) is presented.

$$\tilde{x} = a + b - x \tag{5}$$

Extend the concept of opposite number to high dimensional space and give the definition of opposite point as follows:

Definition 2 (Opposite point) [99] Let $X=(x_1, x_2, \ldots, x_D)$ be a point in d-dimensional space, $x_1, x_2, \ldots, x_D \in \mathbf{R}$ and $x_i \in [a_i, b_i]$, $i=1,2,\ldots,D$. To calculate its opposite point $\tilde{X} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_D)$, Eq. (6) is used.

$$\tilde{x}_i = a_i + b_i - x_i \tag{6}$$

2.4 Orthogonal experimental design (OED)

OED is an efficient tool for finding the optimal combination of multi-factor and multi-level experiments through a small number of experiments [100]. For example, for an experiment with 2-level-7-factor, 128 tests are required to discover the optimal combination. If the orthogonal test design is used, according to the orthogonal table $L_8(2^7)$, such as Eq. (7), the optimal or better combination can be found through only 8 tests, which greatly improves the test efficiency. Since there is no guarantee that the optimal combination is in the orthogonal table [101], when using the orthogonal table, it is usually necessary to perform factor analysis to find the theoretical optimal combination of the experiment, and combine all the combinations in the orthogonal table to determine the best solution of the experiment. For that reason, for the above-mentioned experiments with 2 levels and 7 factors, eight groups of candidate combinations can be obtained first according to orthogonal table $L_8(2^7)$, then factor analysis is carried out to find a set of theoretical optimal combinations. Finally, nine combinations are evaluated to find the best combinations for the experiment.

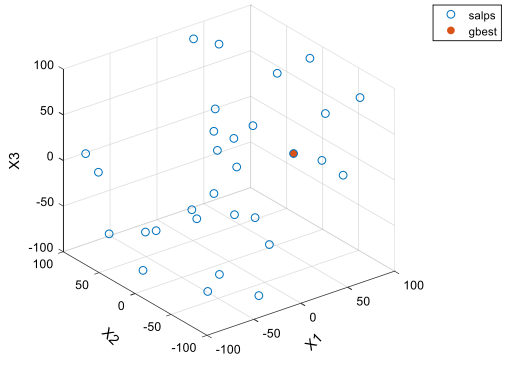
$$L_8(2^7) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 2 & 2 & 2 \\ 1 & 2 & 2 & 1 & 1 & 2 & 2 \\ 1 & 2 & 2 & 2 & 2 & 1 & 1 \\ 2 & 1 & 2 & 1 & 2 & 1 & 2 \\ 2 & 1 & 2 & 2 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 & 2 & 2 & 1 \\ 2 & 2 & 1 & 2 & 1 & 1 & 2 \end{bmatrix}. \tag{7}$$

3 Proposed OOSSA approach

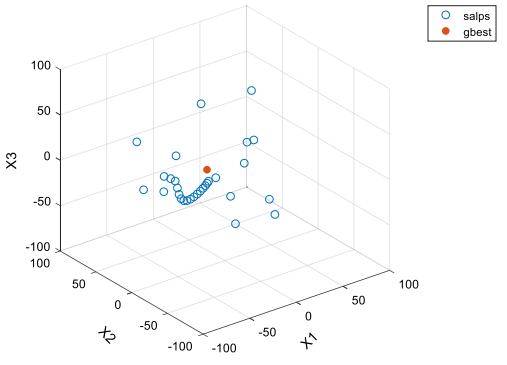
3.1 Motivation

According to the previous section, the leader is constantly moving in the direction of the food source, and the followers follow closely behind the leader. In this case, the salp chain successfully completed the foraging process under the leadership of the leading salp. As can be known from Eqs. (1) and (3), the current optimal solution, that is, the position of the food source, only has a direct impact on the leading salp, while the impact on the followers is indirect, and the force is relatively weak. The food source provides the best guidance when the salp swarm is foraging in the ocean, so it is very important that the food source should be located in the global optimal position. However, due to the lack of prior knowledge, it is difficult to determine whether the current food source is in a global optimal state. If the food source is in the local optimal region, the salp chain will gather in the local optimum region, resulting in the loss of population diversity, and finally make the algorithm converge to the local optimal. Fig. 5 shows the 30 salps location of the Sphere problem with three dimensions in the interval [-100, 100] observed at various stages of SSA.

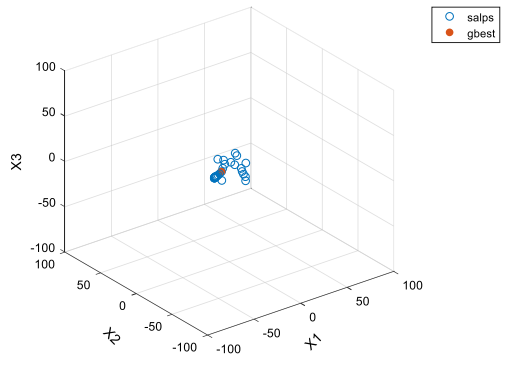
From Fig. 5, at the early stage of SSA (Fig. 5(a)), 30 individuals are scattered in the search space, and the rich population diversity makes the algorithm have a good global exploration ability. As the number of evolutions increases (Fig. 5(b)), the leading salp continues to approach the food source, and followers follow each other. All salps form a chain and move around the food source. In the middle stage of the SSA (Fig. 5(c)), all individuals continue to shrink to the food source position, and the search range continues to narrow, until the end of optimization process (Fig. 5(d)), all individuals gather near the current optimum



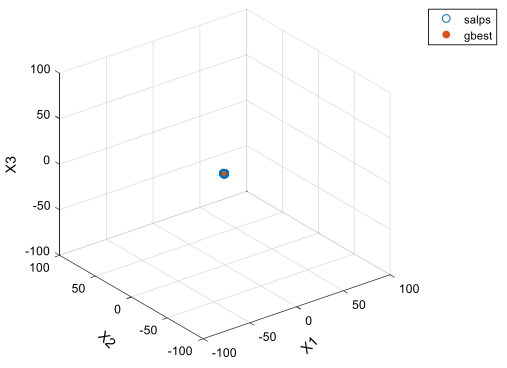
(a) Iteration=1



(b) Iteration=200



(c) Iteration=300



(d) Iteration=400

Fig. 5 Population distribution observed at various stages in SSA for solving Sphere function ($D = 3$)

solution, and the population diversity is lost. If the food source is located in a local optimal, then the leader will lead the salp chain to gradually move to the local optimum area, eventually causing SSA to fall into a local optimum. Therefore, it is necessary to adjust the leader's position update strategy to improve the diversity of the SSA algorithm. The basic SSA is prone to search stagnation, so it is difficult to accomplish the goal of global search-fast convergence balance. In addition to the reasons analyzed above, the unreasonable number of leaders and followers is also an important reason for this tragedy. In the canonical SSA algorithm, the leader is responsible for global exploration, and the follower is responsible for local exploitation. That is to say, the leader first locates a rough approximation of the global optimum, and then the followers accurately search in this area to improve solution accuracy. There is always only one leader in the standard SSA, which means that in the early stage of the evaluation, the global search performed by a single leader is insufficient, too many followers will lead to excess local exploitation, which leads to premature convergence. This situation also exists in the later iteration. To solve this problem, we propose an adaptive mechanism for the number of leaders, i.e., the number of leaders adaptively changes over the course of iterations, trying to achieve the goal of exploration-exploitation balance. Additionally, the basic SSA has a poor performance in terms of convergence accuracy. To settle this problem, we adjust the follower position update mechanism and propose a ranking-based dynamic learning strategy.

The points discussed above have prompted the authors to estimate that there are some limitations in basic SSA, so the above point of view is the motivation behind our proposal of an improved version of SSA. The new SSA-based structure aims to solve the drawbacks of SSA through different modifications, so as to improve the overall performance of basic SSA. Detailed discussion on each of the introduced operators are provided in the following subsections.

3.2 The adaptive mechanism for the number of leaders

In the basic SSA algorithm, the leader is the search agent at the front of the salp chain, and the other salps are followers. The working nature of the salp is determined by its own role. The leader updates its own state based on the position information of the food source, and the followers update their position by following each other. With the increase of the number of evolutions, the salp chain continues to move closer to the food source under the leadership of the leading salp. However, it can be seen from Eq. (3) that when the follower updates the

position, it is only affected by its own state and the position of the individual in front of it, while the global optimal solution, that is, the food source position, only directly affects the leader's position update. Therefore, if there is only one leader in the salp chain, the global optimum solution provides too little help to the salp swarm during the search process. Once the leader falls into the local optima, the followers must follow to the local optimal region, resulting in the premature convergence. To solve this problem, we made some improvements to the number of leaders and followers. First, increase the number of leaders, which can help to improve the global search ability and accelerate the convergence speed of the algorithm. Then, an adaptive technique is presented for the number of leaders. This can help to enhance the ability of exploration-exploitation balance, and improve the solution accuracy. Fig. 6 shows that how the number of leaders and followers change adaptively over the course of iterations. To calculate the number of leaders and followers, Eq. (8) is utilized.

$$\begin{cases} leaderno = ceil\left(N \cdot b \cdot \left(\tan\left(\dfrac{-\pi(l-1)}{4T} + \dfrac{\pi}{4}\right)\right)\right) \\ followerno = N - leaderno \end{cases} \tag{8}$$

where *leader no* and *follower no* show the number of leaders and followers, respectively, l is the current iteration, N is the population size, T is the maximum number of iterations, b is a parameter to adjust the number of leaders. After a large number of experiments, the value of b is set to 0.55.

Equation (8) shows that the number of leaders decreases adaptively over the course of iterations, while the number of followers increases accordingly. In the early iteration, a sufficient number of leaders perform global exploration and a suitable number of followers perform local exploitation. In this case, multiple leaders effectively explore unknown areas, thus improving the algorithm's ability to jump out of local optimum. At the same time, an appropriate number of followers can ensure that the algorithm has a strong exploitation capacity. As the number of iterations increases, the number of leaders decreases adaptively and the number of followers increases accordingly. In this case, a sufficient number of followers can accurately search within the global optimum area determined at the early iteration, so as to improve the convergence accuracy.

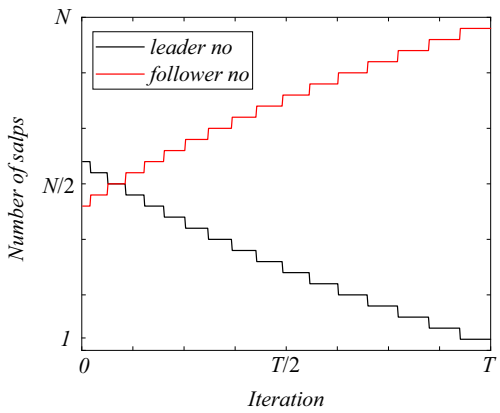


Fig. 6 The number of leaders and followers changes adaptively over the course of iterations

3.3 Orthogonal lens opposition-based learning strategy

In the canonical SSA, at the end of search process, the salp chain tends to move in a small area near the food source. If the food source is at the local optima, then the population will converge to the local optima, resulting in the search stagnation. Therefore, when dealing with intractable multi-modal problems, SSA is prone to premature convergence. Consequently, the ability to escape the local optimum has become the most urgent problem for SSA to solve. Therefore, this study presents a new Orthogonal Lens Opposition-Based Learning (OLOBL) mechanism to help the leading salps migrate to a potentially more promising region.

Firstly, a Lens Opposition-Based Learning (LOBL) strategy is presented as a technique to generate reverse solution in OLOBL strategy. The essence of LOBL is a dynamic opposition-based learning strategy designed by combining OBL and optical LI principle, and its performance is better than that of OBL. Next, the definition of LOBL is illustrated in detail:

Definition 3 (Cardinal point) [102] Let $o_1,o_2,\ldots,o_m$ be several points in D-dimensional space. The opposite of point $X=(x_1,x_2,\ldots,x_D)$ is $\widetilde{X} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_D)$, and the Euclidean distances of their two to point o_i (i=1,2,…,m) are d_i and d^*i, respectively. Let $k=d_i/d^*i$, and k= 1,2,...,n, then o_i is designated the cardinal point of X and $\widetilde{X}$ when $k=i$.

Consider one-dimensional search space as an example, suppose there is an object M with a height of h at x on the coordinate axis, and $x\in[a,b]$, and a lens with focal length r is fixed at the cardinal position o, it should be pointed out that, this paper takes the midpoint of the interval [a, b] as the cardinal point. Based on the LI principle, an image M^* with height h^* can be procured. The one-dimensional spatial LOBL process for the leading salp (x) is illustrated in Fig. 7.

In Fig. 7, x takes o as the cardinal point to get the corresponding opposite point $\tilde{x}$, the geometric relationships can be specified as follows:

$$\frac{\frac{(a+b)}{2} - x}{\tilde{x} - \frac{(a+b)}{2}} = \frac{h}{h^*} \tag{9}$$

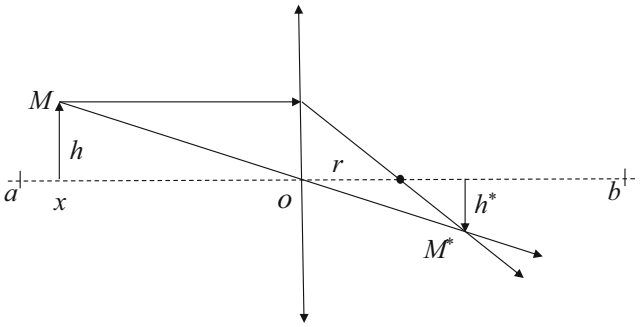


Fig. 7 Lens opposition-based learning

Let $h/h^{*}=k$, Eq. (9) can be modified as:

$$\widetilde{x} = \frac{(a+b)}{2} + \frac{(a+b)}{2k} - \frac{x}{k} \tag{10}$$

Let k=1, the Eq. (10) can be clarified to

$$\widetilde{x} = a + b - x \tag{11}$$

where the Eq. (11) is the OBL technique equation in [103].

Equations (10) and (11) show that the reverse individual obtained from the OBL strategy is fixed, and that the reverse individual obtained from LOBL is dynamic depending on the k-value.

Extending Eq. (10) to the n-dimensional space and Eq. (12) is presented.

$$\widetilde{x_i} = \frac{(a_i+b_i)}{2} + \frac{(a_i+b_i)}{2k} - \frac{x_i}{k} \tag{12}$$

where x_i and $\widetilde{x_i}$ are the i-dimensional components of x and $\widetilde{x}$, respectively, and a_i is the upper boundary, b_i is the lower boundary.

OLOBL is a technique produced by combining OED and LOBL. Compared with OBL, LOBL can further enrich the diversification of the population, thus enhancing the global exploration capability of the algorithm. Therefore, LOBL is used to generate reverse solution in OLOBL mechanism. However, according to the research of Park et al. [104], for a solution, the reverse solution is only better than the current solution in some dimensions. Therefore, taking the reverse value of all the dimensions of the individual will cause dimension degradation, that is, some dimensions are far away from the global optimal solution. To solve this problem, combined with OED and LOBL, an orthogonal lens opposition-based learning (OLOBL) strategy is designed, which fully considers each dimensional component of the current individual and the reverse individual, and combines their dominant dimensions to produce a partial inverse solution.

The OLOBL strategy is embedded in the canonical SSA algorithm, and the dimension D of the problem corresponds to the factors of the OED, and the individual and its opposite individual are the two levels of the OED. The specific process of constructing the partial inverse solution is as follows: a 2-level-D-factor orthogonal experiment is designed for the current individual and its opposite individual, and M partial reverse solution is generated, M is calculated according to Eq. (13). When the partial reverse solution is generated according to the orthogonal table, if the element in the orthogonal table is 1, the partial reverse solution takes the value of the current individual in the corresponding dimension, if the element in the orthogonal table is 2, the partial reverse solution takes the value of the opposite individual in the corresponding dimension. Taking a 7-dimensional problem as an example, the process of producing a partial inverse solution by using a 2-level-7-factor orthogonal experiment is illustrated, as shown in Fig. 8.

$$M = 2^{\lceil \log_2(D+1) \rceil} \tag{13}$$

According to the characteristics of the OED, the elements in the first row of the orthogonal table are all 1, which means

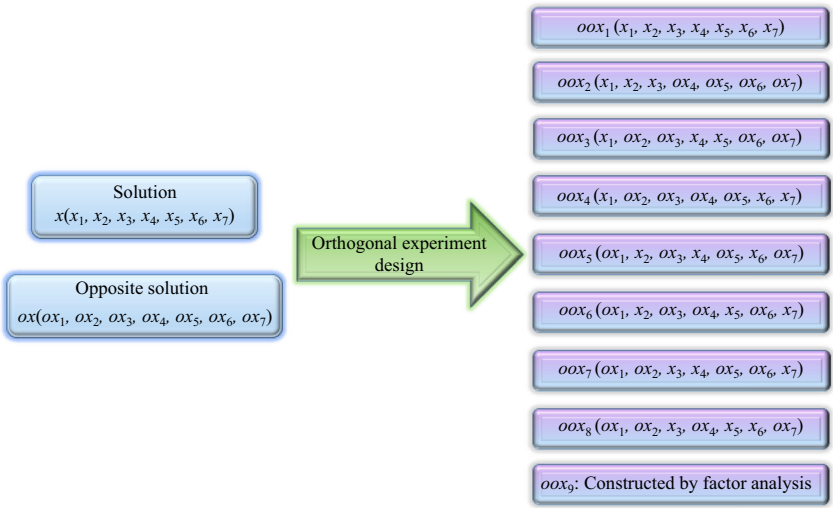


Fig. 8 Construct experimental solution

that the first set of experimental solutions is the current individual itself and does not need to be evaluated. The other M-1 sets of experimental solutions are the combination of the dominant dimensions of the current individual and its reverse individual, that is, partial reverse solutions, which need to be evaluated. According to the content mentioned above, when using the OED, it is necessary to carry out factor analysis to find out a group of theoretical optimal combination that is not in the orthogonal table, which need to be evaluated. Therefore, the number of function evaluations (FEs) required for each execution of OLOBL is M times. To achieve a good balance between enhancing the exploration ability of the algorithm and reducing the number of FEs, only one leader is randomly selected to perform the OLOBL strategy in each iteration, and the other leaders only perform the LOBL tactic, and choose the better one from the leader and its reverse individual or partial reverse individual to enter the later iteration.

3.4 Ranking-based dynamic learning strategy

According to Eq. (3), in the basic SSA, followers learn the previous individual while retaining their own characteristics, and complete the location update. The location update mechanism is relatively simple, once the leader falls into the local optimum, the followers must follow to the local optimum area. To enhance the flexibility of the follower's position update mechanism, this paper proposes a ranking-based dynamic learning strategy. Firstly, the ranking of the current search agent and the previous search agent in the salp swarm is evaluated based on the fitness value, and then the influence weight is designed according to the ranking, and it is applied to the corresponding individual.

After introducing the ranking-based dynamic learning strategy, Eq. (3) can be modified as:

$$X_j^i = \frac{1}{2}\left(\frac{rank_{i-1}}{rank_i + rank_{i-1}} X_j^i + \frac{rank_i}{rank_i + rank_{i-1}} X_j^{i-1}\right) \quad (14)$$

where $rank_i$ and $rank_{i\text{-}1}$ represent the ranking of the two individuals respectively.

The ingenuity of Eq. (14) is that for the current individual i and the previous individual i-1, the ranking of the individuals with better fitness is correspondingly higher, but the ranking value is smaller, so the ranking value of the individual with poor fitness is taken as the coefficient of the better individual, so that the better individual has a larger influence weight, and vice versa.

3.5 Proposed OOSSA

To enhance the comprehensive performance of SSA, we analyze the drawbacks of the method and propose three adjustment mechanisms, and then embedded the improved search mechanism into the basic SSA to present a novel and efficient SSA-based algorithm called OOSSA. The detailed process of OOSSA is as follows:

Step 1 Initialize parameters of the OOSSA method including population size N, maximal number of FEs F_{max}, problem dimension D, the upper and lower boundaries of the i-th dimensional space, lb_i, and ub_i. Randomly generate N individuals in the search space.
Step 2 Evaluate the initial population based on the fitness value, and the position of the search agent with the best fitness is set as the current food source.
Step 3 Calculate the number of leaders and followers according to Eq. (8), and randomly select a leader and mark it as OLOBL-Leader.
Step 4 Judge the role of salps. If the salp is a leader and is not OLOBL-Leader, enter Step5; if the salp is OLOBL-Leader, enter Step6; if the salp is a follower, enter Step7.
Step 5 Use Eq. (1) to amend the state of the leading salp to generate candidate solution 1. The current leader executes the LOBL strategy according to Eq. (12) to generate candidate solution 2, and the search agent with better fitness value is chosen as the new solution among candidate solution 1 and candidate solution 2.
Step 6 Leader OLOBL-Leader executes the OLOBL strategy.
Step 7 Update the state of the follower by Eq. (14).
Step 8 The one with the better fitness value among the food source and the current optimal individual is set as the food source.
Step 9 If the number of FEs does not exceed F_{max}, return to Step 3. Otherwise, output the food source position.

The flowchart of OOSSA is illustrated in Fig. 9.

While keeping the basic framework and overall flow of the original SSA algorithm unchanged, OOSSA introduces the LOBL strategy in the leader position update phase, randomly selects an individual to implement the OLOBL strategy, and modifies the follower position update equation. Suppose the dimension of problem is D, population size is N, and maximum number of iterations is T. The computational complexity of population position updating is $O(TND)$, and the computational complexity of the leader executing LOBL or OLOBL is $O(TD^2)$. Therefore, the eventual computational complexity of OOSSA is $O(TN^2)$, which is the same as that of the original SSA. This shows that the algorithm is modified without increasing the computational overhead.

3.6 Justification of OOSSA

In this subsection, the proposed OOSSA algorithm is justified in a metaphor-free way. A proportion of individuals in the

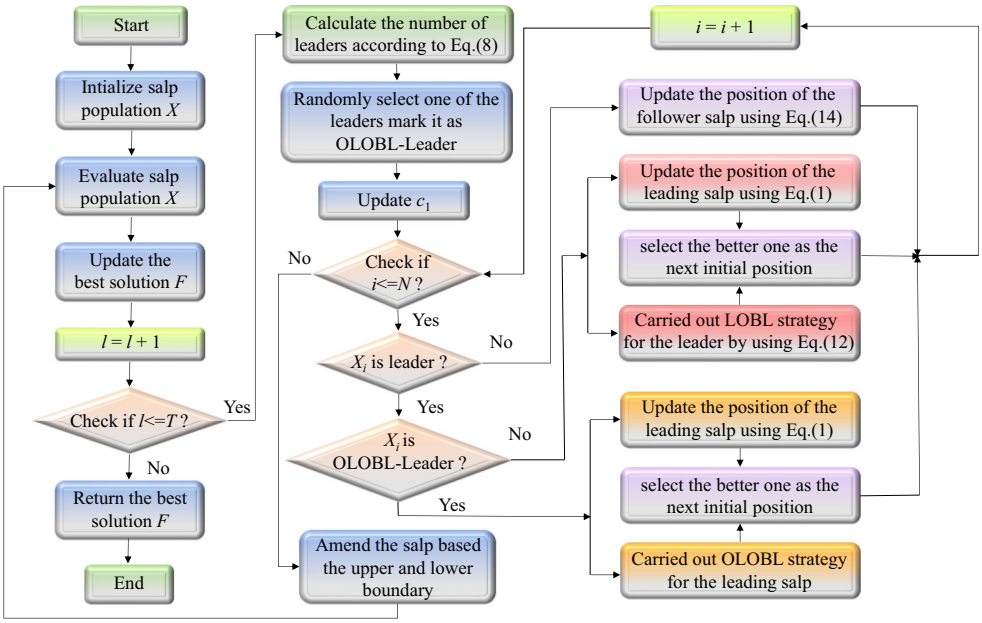


Fig. 9 The flowchart of OOSSA

population search the solution space according to Eq. (1), and this search equation can be rewritten as

$$X_j^1 = \begin{cases} F_j + V_j & c_3 \geq 0.5 \\ F_j - V_j & c_3 < 0.5 \end{cases} \tag{15}$$

To calculate V_j, the following equation is employed

$$V_j = c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) \tag{16}$$

where c_2 is a random number between [0,1], so $c_1((ub_j\text{-}lb_j)c_2+lb_j)$ can be rewritten as $c_1((ub_j\text{-}lb_j)rand(0,1)+lb_j)$. This equation is the same as the one used to allocate initial positions of individuals, which means that the outcome obtained from this equation is a randomly generated position in the jth dimensional search space, which can also be interpreted as a step size. c_1 is an important parameter that decreases adaptively during the iterations and is used to control the value of the step size. Therefore, according to Eq. (16), V_j can be defined as a step size that decreases adaptively over the course of iterations and has a stochastic property.

In Eq. (15), F_j is the current optimal solution, which searches the solution space by moving gradually in steps V_j in the hope of finding more promising regions. The parameter c_3 determines the movement direction of the current optimal solution. Since c_3 is a random number between [0,1], the current optimal solution searches towards positive or negative infinity with equal probability, which ensures that the whole solution space is adequately searched.

Based on the above analysis, from Eq. (15), for the jth dimensional search space, F_j is the current optimal position and V_j is the step size. In the early iteration, larger c_1 values generate larger step sizes V_j, which can help the current optimal solution to discover new promising regions in the search space by moving significantly, thus identifying the region where the global optimal solution is located. In the later iteration, smaller c_1 values produce smaller step sizes V_j, which can help the current optimal individual to discover the global optimal solution by exploiting the promising regions identified in the early iteration. In summary, Eq. (15) is responsible for thoroughly searching the entire solution space in the early evolution to pinpoint the potentially global optimal region and finely exploiting this region in the later evolution to find the optimal solution.

After an individual updates its position according to Eq. (15), it jumps to an OLOBL individual based on the OLOBL strategy, and this operator is analyzed below.

The literature [98] demonstrates that the reverse solution has a higher probability of reaching the global optimal than the current solution, and the literature [104] further demonstrates that the reverse solution outperforms the current solution only in certain dimensions, i.e., opposite learning may cause the dimensional degradation problem. The OED technique can solve this problem by discovering and combining the dominate dimensions of the current and reverse solutions, which is the OOBL strategy mentioned in the paper. The OLOBL strategy proposed in this paper is a generalized version of the OOBL, i.e. OOBL is a special case of OLOBL, so OLOBL has the same properties as OOBL and is more effective. Consequently, an individual can improve the convergence speed of the algorithm by executing the OLOBL strategy to produce an OLOBL-individual that is closer to the global optimum solution. In addition, if an individual is trapped in a local optimum, by executing the OLOBL

strategy, it can jump out of this trap and move to a more favorable area for searching. In summary, OLOBL can improve the convergence speed of the algorithm and enhance its ability to jump out of the local optimum.

The remaining individuals in the population update their position according to Eq. (14), which will be analyzed below to reveal the search properties it implies.

As Eq. (14) is a modification of Eq. (3), we first analyze the search properties implied by Eq. (3). This search equation is derived from Newton's law of motion, based on which the distance moved by an individual is calculated as follows.

$$D(t) = \frac{1}{2}a\Delta t^2 + v_0\Delta ti \geq 2 \tag{17}$$

In optimization, time is iteration, t represents the discrepancy between evolutions, so Δt=1, v_0 is the initial velocity, consider v_0=0, a is the acceleration and calculate according to the following equation

$$a = \frac{v_{final} - v_0}{\Delta t} \tag{18}$$

When the current individual moves to the position of its previous individual, its velocity is calculated as follows.

$$v_{final} = \frac{x_j^{i-1}(t-1) - x_j^i(t-1)}{\Delta t} \tag{19}$$

where x_j^{i-1} and x_j^i represent the position of the current individual and its previous individual in the jth dimension at the previous time step (t-1), respectively. Therefore, the individual moves to the next position according to the following equation

$$\begin{aligned} x_j^i(t) &= x_j^i(t-1) + D(t) \\ &= x_j^i(t-1) + \frac{1}{2}\left(x_j^{i-1}(t-1) - x_j^i(t-1)\right) \end{aligned} \tag{20}$$

The search equation used by the algorithm can be obtained by simplifying Eq. (20), which is Eq. (3) as presented previously.

Based on the above analysis, the individual moves to the next position according to Newton's laws of motion, and although this gradual movement can search the solution space, this pattern is too rigid. Therefore, we improve it by using the fitness value-based ranking of two individuals in the population as a weighting factor to dynamically adjust the movement direction and step size of the current individual so that the next position is closer to the better of the two search agents, which is the proposed search Eq. (14). In the experimental section, we will verify the validity of the proposed search model by rigorous experimentation on benchmark functions.

Next, we will present a theoretical convergence proof of OOSSA.

OOSSA is a swarm intelligence optimization algorithm, so the convergence property of OOSSA is analyzed using the general approach of analyzing the theoretical convergence of population-based techniques.

Theorem 1: If the basic SSA is convergent, the developed OOSSA is also convergent.

Proof of Theorem 1: Let $X(t)$ be the current solution at generation t, which jumps to $\widetilde{X}(t)$ through OLOBL, and their jth dimension values are $X_j(t)$ and $\widetilde{X}_j(t)$, respectively; the global optimal is X^*; the boundary of the search region in jth dimension is $[a_j,b_j]$.

Based on Theorem 1, for any individual $X(t)$ in generation t, it is satisfied as

$$\lim_{t\to\infty} X_j(t) = X_j^* \tag{21}$$

Since $a_j(t)$=min($X_j(t)$), $b_j(t)$=max($x_{i,j}(t)$), it follows that

$$\lim_{t\to\infty} a_j(t) = \lim_{t\to\infty} b_j(t) = X_j^* \tag{22}$$

For the OLOBL-individual$\widetilde{X}(t)$,

$$\widetilde{X}_j(t) = \frac{(a_j(t) + b_j(t))}{2} + \frac{(a_j(t) + b_j(t))}{2k} - \frac{X_j(t)}{k} \tag{23}$$

When $t \to \infty$,

$$\begin{aligned} \lim_{t\to\infty} \widetilde{X}_j(t) &= \lim_{t\to\infty}\left(\frac{(a_j(t) + b_j(t))}{2} + \frac{(a_j(t) + b_j(t))}{(2k)} - \frac{X_j(t)}{k}\right) \\ &= \lim_{t\to\infty} \frac{(a_j(t) + b_j(t))}{2} \\ &\quad + \lim_{t\to\infty} \frac{(a_j(t) + b_j(t))}{(2k)} - \lim_{t\to\infty} \frac{(X_j(t))}{k} \\ &= \frac{\left(X_j^* + X_j^*\right)}{2} + \frac{\left(X_j^* + X_j^*\right)}{2k} - \frac{X_j^*}{k} = X_j^* \end{aligned} \tag{24}$$

From Eq. (24), when $X_j(t)$ converges to X^*, $\widetilde{X}_j(t)$ also converges to X^*. Hence, if the basic SSA algorithm can converge to the global optimal X^*, the OOSSA algorithm is also convergent. It should be noted that the proof does not guarantee that the algorithm converges to the global optimum.

4 Simulation results and discussions

4.1 Benchmark test functions

To authenticate the performance of OOSSA for global optimization problems, a set of 26 widely used benchmark test functions are utilized in the experiment, the details of which

are given in Table 1. These benchmark problems can be characterized three various categories: unimodal, multimodal, and fixed-dimension multimodal. The unimodal problems (f_1~f_9) have no local optima, but only one global optimum, which is used to disclose the local exploitation ability of stochastic algorithms. On the other hand, there are multiple local optima for the multimodal problems (f_{10}~f_{19}), which are suitable for revealing the ability of stochastic algorithms to balance global exploration and local exploitation. The fixed-dimension multimodal problems (f_{20}~f_{26}) face the existence of a large number of local optima and more than one global optimum, which are very appropriate to verify local optimum avoidance and stability of stochastic algorithms. The search space of some benchmark functions are illustrated in Fig. 10. The proposed algorithm is compared with a variety of algorithms including SSA variants and other state-of-the-art swarm intelligence based methods, and all comparison algorithms are given in Table 2.

All algorithms are coded on MATLAB R2014b, and all of the simulation experiments are performed on a computer with Intel(R) Core(TM) i7-7700 CPU(3.60GHz) and 8.00 GB RAM.

4.2 Comparison with SSA and improved SSA

To demonstrate the performance of the proposed OOSSA, we tested it on 26 widely used benchmarks, which are reported in Table 1. The OOSSA algorithm was compared with other thirteen SSA-based methods, including the original SSA algorithm [62], the enhanced SSA algorithm (ESSA) [81], the lifetime scheme-based SSA algorithm (LSSA) [105], the multi-subpopulation-based SSA with Gaussian mutation mechanism (MSNSSA) [106], the chaotic SSA algorithm (CSSA) [80], the self-adaptive SSA algorithm (ASSA) [107], the SSA algorithm based on PSO (SSAPSO) [88], the improved SSA algorithm based on opposition-based learning (ISSA) [108], the Gaussian-SSA algorithm (GSSA) [109], the enhanced opposition-based SSA algorithm (OBSSA) [110], the adaptive SSA algorithm with non-linear coefficient decreasing inertia weight (ASSO) [111], the SSA algorithm with random replacement tactic and double adaptive weighting mechanism (RDSSA) [112], the hybrid enhanced whale optimization SSA algorithm (IWOSSA) [113]. For fair comparisons, the population size N for all algorithms was set to 30. The maximal number of FEs is set to 15000. The dimension is set as 100. The other parameter settings of SSA, ESSA, LSSA, MSNSSA, CSSA, ASSA, SSAPSO, ISSA, GSSA, OBSSA, ASSO, RDSSA, and IWOSSA algorithms are adapted from the original papers. In the proposed OOSSA algorithm, k=10000. Each approach runs 30 times independently on each benchmark problem, and the average and standard deviation of objective function values results found by nine algorithms on these functions as the metrics of performance. At the same time, Friedman rank (f-rank) test [114] was used to test the statistical significance of OOSSA. The simulation results on benchmark functions f_1 to f_{19} with dimensions 100 is shown in Table 3, the statistical results on fixed-dimension problems (f_{20}~f_{26}) is shown in Table 5.

The statistical results from Table 3 show that except for the functions f_6, f_9, f_{11} and f_{14}, the OOSSA algorithm converges to the global optimum on the other 15 test cases. Compared with ESSA, OOSSA finds the similar and better results on six and 13 test functions, respectively. For functions f_{10}, f_{12}, f_{16}, f_{17}, and f_{18}, two algorithms get the theoretical optimum. Compared with MSNSSA, OOSSA provides similar and better results on four and 15 test functions, respectively. OOSSA outperforms SSA, LSSA, CSSA, ASSA, SSAPSO, ISSA, GSSA and IWOSSA significantly in terms of solution accuracy on all test problems. With respect to OBSSA, OOSSA obtains better and similar values for 14 and four benchmarks, respectively. However, OBSSA shows better performance on f_6, but the gap between the two approaches is negligible. Compared to ASSO, OOSSA gets similar and better results on four and 15 test cases, respectively. According to the results of the comparison between OOSSA and RDSSA, the performance of the developed approach is better than its rival on 13 problems. For other six functions, two methods show similar performance on five of them, while the better value is achieved by RDSSA on the remaining one. Additionally, according to the average ranking values of all optimizers achieved from the Friedman test, which are reported at the bottom of Table 3, the OOSSA obtains the top rank, followed by RDSSA, ESSA, OBSSA, ASSO, MSNSSA, GSSA, IWOSSA, ASSA, LSSA, ISSA, CSSA, and SSA. In other words, OOSSA is recommended as the best optimizer among all its peers.

Moreover, Wilcoxon signed rank test (significance level is set to 0.05) [114] is used to verify that the proposed method has significant advantages over other competitors. The p-values calculated in the Wilcoxon signed rank test of OOSSA and other compared algorithms for all benchmark functions with 100 dimensions are given in Table 4. For example, if the optimal algorithm is OOSSA, a comparison is made between OOSSA versus SSA, OOSSA versus ESSA, OOSSA versus LSSA and so on. Among them, N/A represents not available, which means that the corresponding method performs best on this test function, and there is no statistical data to compare with itself. In the statistical table, the symbols "-", "+" and "=" represent the performance of the corresponding approach is worse than, better than, and similar to that of OOSSA, respectively. According to the Wilcoxon's rank sum test, when the p-value is less than 0.05, the zero hypothesis is rejected, that is, it is considered that there is a significant difference between the two methods [114]. It should be noted that when p-values are greater than 0.05, bold is used.

Table 1 26 widely used benchmark test functions

Function name	Function formulation	Search range	f_{min}
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	[−100,100]	0
Schwefel 2.22	$f_2(x) = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	[−10,10]	0
Schwefel 1.2	$f_3(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	[−100,100]	0
Schwefel 2.21	$f_4(x)=\max_i\{\lvert x_i\rvert, 1\le i\le D\}$	[−100,100]	0
Axis paralled hyper-elliposide	$f_5(x) = \sum_{i=2}^{D} i x_i^2$	[−10,10]	0
Quartic	$f_6(x) = \sum_{i=1}^{D} i x_i^4 + \text{random}[0,1)$	[−1.28,1.28]	0
High Conditioned	$f_7(x) = \sum_{i=1}^{D} \left(10^6\right)^{\frac{i-1}{D-1}} x_i^2$	[−100,100]	0
Bent Cigar	$f_8(x) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2$	[−10,10]	0
Generalized Penalized Function	$f_9(x) = \frac{\pi}{D}\left\{ 10\sin^2(\pi y_i) + \sum_{i=1}^{D-1} (y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right] + (y_D-1)^2 \right\} + \sum_{i=1}^{D} u(x_i,10,100,4)$ $y_i = 1+\frac{x_i+1}{4}$ $u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m, x_i > a \\ 0, -a < x_i < a \\ k(-x_i-a)^m, x_i < a \end{cases}$	[−50,50]	0
Rastrigin	$f_{10}(x) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	[−5.12,5.12]	0
Ackley	$f_{11}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$	[−32,32]	0
Griewank	$f_{12}(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[−600,600]	0
Discus	$f_{13}(x) = 10^6 x_1^2 + \sum_{i=2}^{D} x_i^2$	[−10,10]	0
Zakharov	$f_{14}(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5x_i\right)^2 + \left(\sum_{i=1}^{D} 0.5x_i\right)^4$	[−5,10]	0
Schaffer's F7	$f_{15}(x) = \left[\frac{1}{D-1}\sum_{i=1}^{D-1}\left(\sqrt{x_i}\left(\sin\left(50.0x_i^{0.2}\right)+1\right)\right)\right]^2$	[−100,100]	0
Non-continuous Rotated Rastrigin's	$f_{16}(x) = \sum_{i=1}^{D}\left[y_i^2 - 10\cos(2\pi y_i) + 10\right]$ $y_i = \begin{cases} x_i \rightarrow & \lvert x_i\rvert < \frac{1}{2} \\ round(2x_i)/2 \rightarrow & \lvert x_i\rvert \ge \frac{1}{2} \end{cases}$	[−5.12,5.12]	0
Katsuura	$f_{17}(x) = \frac{10}{D^2}\prod_{i=1}^{D}\left(1 + i\sum_{j=1}^{32}\frac{\lvert 2^j x_i - \text{round}\left(2^j x_i\right)\rvert}{2^j}\right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2}$	[−5,5]	0
Inverted cosine wave	$f_{18}(x) = -\sum_{i=1}^{D-1}\left(\begin{array}{c}\exp\left(-\frac{x_i^2 + x_{i+1}^2 + 0.5x_i x_{i+1}}{8}\right) \\ \times\cos\left(4\sqrt{x_i^2 + x_{i+1}^2 + 0.5x_i x_{i+1}}\right)\end{array}\right)$	[−100,100]	-d+1
Powell	$f_{19}(x) = \sum_{i=1}^{D/4}\left[\begin{array}{c}(x_{4i-3}+10x_{4i-2})^2 + 5(x_{4i-1}-x_{4i})^2 + \\ (x_{4i-2}-2x_{4i-1})^4 + 10(x_{4i-3}+x_{4i})^4\end{array}\right]$	[−5,5]	0
Six-Hump Camel-Back Function	$f_{20}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	[−5,5]	−1.0316
Goldstein-Price Function	$f_{21}(x) = \left[1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]\times \left[30+(2x_1-3x_2)^2\times\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]$	[−2,2]	3

Table 1 (continued)

Function name	Function formulation	Search range	f_{min}
Shekel's Family	$f_{22}(x) = -\sum_{i=1}^{5}\left[(X-a_i)(X-a_i)^T + c_i\right]^{-1}$	[0,10]	−10.1532
Shekel's Family	$f_{23}(x) = -\sum_{i=1}^{7}\left[(X-a_i)(X-a_i)^T + c_i\right]^{-1}$	[0,10]	−10.4028
Shekel's Family	$f_{24}(x) = -\sum_{i=1}^{10}\left[(X-a_i)(X-a_i)^T + c_i\right]^{-1}$	[0,10]	−10.5363
Hartman's Function	$f_{25}(x) = -\sum_{i=1}^{4}\alpha_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	[0,1]	−3.86278
Hartman's Function	$f_{26}(x) = -\sum_{i=1}^{4}\alpha_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	[0,1]	−3.32237

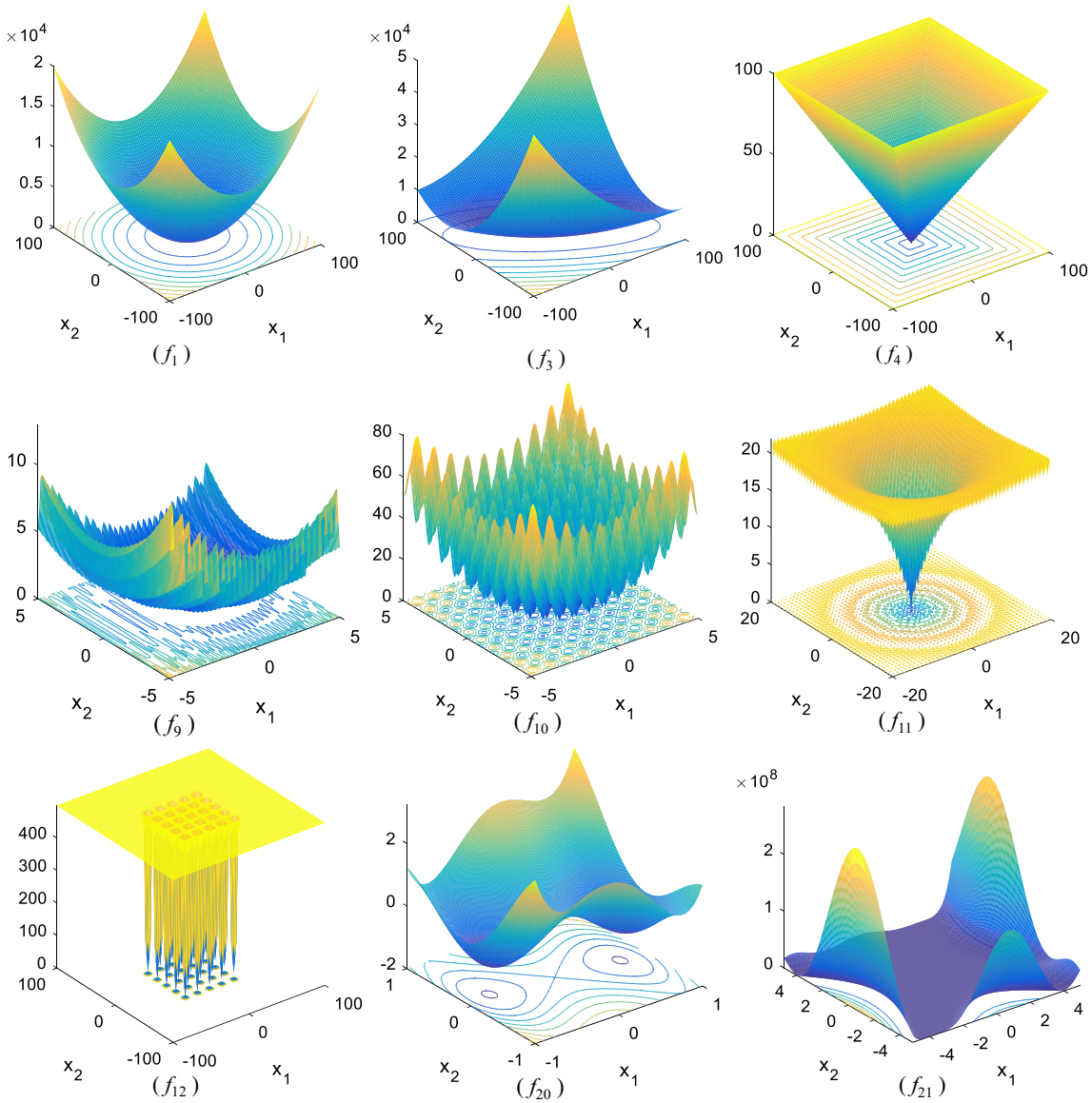


Fig. 10 Search space of some typical benchmark problems

Table 2 Comparison algorithms

Algorithm	Year	Type	Reference
SSA	2017	Basic SSA	[62]
CSSA	2018	SSA variant	[80]
ESSA	2019	SSA variant	[81]
SSAPSO	2018	SSA variant	[88]
LSSA	2020	SSA variant	[105]
MSNSSA	2021	SSA variant	[106]
ASSA	2020	SSA variant	[107]
ISSA	2019	SSA variant	[108]
GSSA	2021	SSA variant	[109]
OBSSA	2021	SSA variant	[110]
ASSO	2021	SSA variant	[111]
RDSSA	2021	SSA variant	[112]
IWOSSA	2021	SSA variant	[113]
MPA	2020	Recent algorithm	[116]
EO	2020	Recent algorithm	[117]
TSA	2020	Recent algorithm	[129]
SOGWO	2020	Recent algorithm	[130]
IGWO	2021	Recent algorithm	[131]
HGS	2019	Recent algorithm	[132]
EESHHO	2021	Recent algorithm	[133]
ArOA	2021	Recent algorithm	[134]
AOA	2020	Recent algorithm	[135]
DMMFO	2021	Recent algorithm	[136]
WEMFO	2021	Recent algorithm	[137]
OGWO	2021	Recent algorithm	[138]
OOSSA		The proposed algorithm	

From Table 4, the p-values are greater than 0.05 in the following cases: OOSSA versus ESSA on f_6, OOSSA versus OBSSA on f_6, and OOSSA versus ASSO on f_6. Except for the three comparisons mentioned above, the p-values obtained in all other cases are less than 0.05. This means that the overall performance of OOSSA is obviously superior to other rivals, that is, the superiority of OOSSA is statistically significant. Based on the above discussion, compared with the basic SSA algorithm and other improved version of SSA algorithms, the overall performance of OOSSA has a strong competitiveness.

Another issue of interest is the performance of OOSSA on fixed-dimension problems. The comparison results between OOSSA and thirteen comparison SSA variants outlines in Table 5 address this concern. Due to the low dimensions of these test cases, the optimal solutions obtained by the fourteen approaches on all functions can basically achieve theoretical optimum. In terms of average values, the results of OOSSA on most functions are highly close to the global optimal. With respect to standard deviation, OOSSA can also obtain satisfactory results on most cases, indicating that the methodology has a good stability. According to the average ranking values of OOSSA and the involved SSA variants on seven test functions provided by Friedman test, we can see that OOSSA ranks second, behind CSSA, and followed by OBSSA, GSSA, IWOSSA, SSAPSO, ISSA, ESSA, ASSA, SSA, RDSSA, MSNSSA, LSSA, and ASSO, which further indicates that OOSSA has a better performance in terms of the capability to balance diversification and intensification, and is also highly competitive in respect of stability.

4.3 The influence of improving mechanisms on SSA

In this section, we investigate the effectiveness of different mechanisms of the proposed OOSSA algorithm. The three components of OOSSA that differ from SSA are: the number of leaders changes adaptively, the orthogonal lens opposition-based learning (OLOBL) strategy, and dynamic learning (DL) mechanism. To verify the validity of the adjustment strategies, OLOBL and DL are embedded into the basic SSA algorithm respectively, and the OLOBL-SSA and DL-SSA algorithms are obtained. In addition, to compare the effectiveness of LOBL and OBL strategies, the orthogonal opposition-based learning using OBL is embedded into the basic SSA to obtain OOBL-SSA. It should be pointed out that OLOBL-SSA, DL-SSA and OOBL-SSA all adopt the multi-leader mechanism. Select all the unimodal and multimodal problems (f_1-f_{19}) with 100 dimensions in Table 1 for simulation experiments to compare OOSSA, OLOBL-SSA, DL-SSA and OOBL-SSA. The parameter settings are the same as those in the previous section. Each algorithm runs 30 times independently on each benchmark problem, and recording the average optimal object function value, standard deviation, and Friedman ranking results, as shown in Table 6.

According to the statistical results in Table 6, the convergence accuracy and stability of the OLOBL-SSA, DL-SSA and OOBL-SSA algorithms embedded with a single component are better than the basic SSA algorithm on all functions, which proves the effectiveness of the different mechanisms of the proposed method. Compared with DL-SSA, OOSSA achieved superior results for 16 test functions except f_{10}, f_{12}, and f_{16}, and for functions f_{10}, f_{12}, and f_{16}, the two algorithms achieved similar and satisfactory results. Compared with OOBL-SSA, OOSSA achieved similar and better performance on 4 and 15 test functions. The OLOBL-SSA algorithm, like OOSSA, can find the theoretical optimal value in most functions. But for the test functions f_6, f_9, and f_{14}, the convergence accuracy and optimization stability of OOSSA are obviously better than that of OLOBL-SSA, especially on the function f_9 and f_{14}, OOSSA shows significant advantages. Therefore, the two components have certain effects, and embedding both of them into the basic SSA algorithm can further improve the overall performance of the algorithm. In addition, according to Table 6, compared with OOBL-SSA, OLOBL-

Table 3 Comparisons of fourteen algorithms on 19 test functions with 100 dimensions

Function	Results	SSA	ESSA	LSSA	MSNSSA	CSSA	ASSA	SSAPSO	ISSA	GSSA	OBSSA	ASSO	RDSSA	IWOSSA	OOSSA
f_1	Mean	1.45E+03	1.57E-41	0.0043	1.81E-22	1.5442	0.0430	3.03E+03	89.5894	1.49E-14	5.77E-32	1.94E-26	6.06E-46	1.47E-06	0
	Std	402.7496	8.53E-41	0.0041	5.33E-23	1.8662	0.0106	1.59E+03	17.9876	6.78E-14	5.34E-32	2.67E-27	3.32E-45	1.41E-06	0
	f-rank	13	3	9	6	11	10	14	12	7	4	5	2	8	1
f_2	Mean	47.4933	1.08E-22	0.0963	1.08E-11	19.4505	0.0226	270.7965	10.6351	2.42E-17	1.68E-16	1.14E-13	1.70E-24	0.0927	0
f_3	Std	6.5460	4.46E-22	0.0445	2.06E-12	4.4279	0.0077	59.4548	1.0478	2.96E-17	7.18E-17	6.32E-15	9.29E-24	0.5003	0
	f-rank	13	3	10	7	12	8	14	11	4	5	6	2	9	1
	Mean	4.34E+04	9.16E-41	5.62E+03	8.02E-21	1.91E+04	1.46E+04	9.35E+04	2.42E+03	1.50E+04	4.89E-30	2.28E-25	2.05E-38	1.06E+05	0
	Std	2.39E+04	5.02E-40	4.35E+03	2.46E-20	1.03E+04	7.21E+03	1.55E+04	1.07E+03	7.60E+03	8.82E-30	1.03E-25	1.21E-37	2.25E+04	0
	f-rank	12	2	8	6	11	9	13	7	10	4	5	3	14	1
f_4	Mean	27.0380	5.67E-26	21.6093	3.72E-12	23.7176	10.0239	78.3318	5.7509	23.0698	5.06E-17	3.46E-14	6.44E-31	44.3680	0
	Std	3.4387	2.03E-25	6.7799	9.81E-13	3.1505	2.9390	5.0910	0.7003	2.6272	2.68E-17	4.37E-15	2.39E-30	7.6379	0
	f-rank	12	3	9	6	11	8	14	7	10	4	5	2	13	1
f_5	Mean	887.8972	2.63E-41	0.1874	8.91E-23	0.8847	0.0177	2.57E+03	48.8235	5.61E-16	2.69E-32	9.66E-27	4.52E-44	5.69E-07	0
	Std	237.1191	1.40E-40	0.1433	2.88E-23	0.7483	0.0112	1.06E+03	12.7609	1.58E-15	2.56E-32	1.30E-27	2.47E-43	3.81E-07	0
	f-rank	13	3	10	6	11	9	14	12	7	4	5	2	8	1
f_6	Mean	2.4876	8.25E-05	0.5907	4.16E-04	0.2908	0.0915	3.4171	0.0908	0.1270	8.72E-05	1.07E-04	6.29E-04	0.0852	9.88E-05
	Std	0.5828	8.04E-05	0.2248	3.03E-04	0.0850	0.0162	1.7301	0.0399	0.1452	1.01E-04	1.11E-04	5.77E-04	0.0351	7.97E-05
	f-rank	13	1	12	5	11	9	14	8	10	2	4	6	7	3
f_7	Mean	7.97E+07	1.64E-40	1.3140	1.29E-17	7.89E+05	133.1636	1.05E+08	3.32E+06	1.90E-14	4.47E-27	1.34E-21	1.33E-58	0.0127	0
	Std	3.35E+07	8.67E-40	1.0040	1.97E-18	4.67E+05	69.2331	2.03E+07	1.33E+06	7.06E-14	4.00E-27	4.38E-22	7.31E-58	0.0081	0
	f-rank	13	3	9	6	11	10	14	12	7	4	5	2	8	1
f_8	Mean	1.36E+07	5.58E-41	7.79E+03	1.70E-18	8.74E+03	427.1492	3.05E+07	8.84E+05	6.33E-11	3.23E-28	1.97E-22	1.19E-41	0.0135	0
	Std	4.26E+06	3.04E-40	5.86E+03	5.73E-19	7.34E+03	213.0277	1.15E+07	2.01E+05	2.91E-10	3.35E-28	2.46E-23	6.50E-41	0.0102	0
	f-rank	13	3	10	6	11	9	14	12	7	4	5	2	8	1
f_9	Mean	35.3255	1.0062	9.1727	0.1642	7.7547	1.5685	2.81E+05	0.8693	24.3240	0.1522	0.9300	0.0014	21.8744	0.0339
	Std	11.5573	0.0788	6.1089	0.0241	1.9086	0.9211	4.42E+05	0.1452	9.2930	0.0373	0.0621	8.90E-04	9.7817	0.0064
	f-rank	13	7	10	4	9	8	14	5	12	3	6	1	11	2
f_{10}	Mean	240.3944	0	185.2467	0	183.1611	208.2155	722.6061	63.9110	3.20E-12	0	0	0	253.2697	0
	Std	44.2435	0	96.8821	0	29.2512	58.6184	90.7388	13.4930	1.27E-11	0	0	0	136.0232	0

Table 3 (continued)

Function	Results	SSA	ESSA	LSSA	MSNSSA	CSSA	ASSA	SSAPSO	ISSA	GSSA	OBSSA	ASSO	RDSSA	IWOSSA	OOSSA
	f-rank	12	1	10	1	9	11	14	8	7	1	1	1	13	1
f_{11}	Mean	9.9543	8.88E-16	0.0385	1.57E-12	4.2571	0.0314	19.1737	3.9357	5.15E-09	8.88E-16	1.91E-14	4.09E-15	1.43E-04	8.88E-16
	Std	0.9232	0	0.0406	1.95E-13	0.5701	0.0243	0.3026	0.2136	1.31E-08	0	1.80E-15	1.08E-15	4.46E-05	0
	f-rank	13	1	9	5	12	8	14	11	6	1	4	3	7	1
f_{12}	Mean	13.5680	0	0.0614	0	0.3515	0.0306	28.9351	1.8990	3.54E-15	0	0	0	0.0060	0
	Std	3.5509	0	0.0753	0	0.2524	0.0304	11.9287	0.2278	6.28E-15	0	0	0	0.0103	0
	f-rank	13	1	10	1	11	9	14	12	7	1	1	1	8	1
f_{13}	Mean	1.10E+03	5.23E-36	0.0084	1.10E-21	459.6082	0.0019	354.5348	260.3708	5.60E-18	3.93E-31	2.20E-27	3.79E-44	2.46E-07	0
	Std	664.2424	2.86E-35	0.0081	1.20E-21	203.4279	0.0015	91.1716	251.0718	2.24E-17	5.96E-31	1.47E-27	2.07E-43	2.32E-07	0
	f-rank	14	3	10	6	13	9	12	11	7	4	5	2	8	1
f_{14}	Mean	64.3522	1.17E-43	0.0306	1.51E-24	0.1373	0.0088	49.4894	501.3531	4.56E-09	215.6247	1.68E-28	2.86E-62	0.0101	3.98E-75
	Std	16.8056	6.41E-43	0.0275	4.96E-25	0.1565	0.0051	16.3664	2.34E+03	2.15E-08	27.8799	2.78E-29	1.57E-61	0.0109	1.68E-74
	f-rank	12	3	9	5	10	7	11	14	6	13	4	2	8	1
f_{15}	Mean	5.0064	2.22E-10	0.2244	1.32E-06	4.2167	1.6562	8.3109	2.2796	5.01E-10	4.55E-09	1.29E-07	2.26E-16	0.0370	0
	Std	0.2056	1.21E-09	0.1270	9.12E-08	0.3476	0.5219	0.4031	0.1278	8.14E-10	1.00E-09	6.18E-09	1.24E-15	0.0337	0
	f-rank	13	3	9	7	12	10	14	11	4	5	6	2	8	1
f_{16}	Mean	474.9610	0	342.2456	0	299.7936	315.9863	877.4782	77.6109	4.30E-07	0	0	0	394.5909	0
	Std	82.1482	0	189.7159	0	55.8888	138.5810	95.9362	26.5817	5.24E-07	0	0	0	198.6438	0
	f-rank	13	1	11	1	9	10	14	8	7	1	1	1	12	1
f_{17}	Mean	6.67E-12	0	1.51E-12	2.11E-13	5.67E-13	9.15E-12	2.74E-13	6.58E-12	8.33E-12	6.09E-12	2.05E-16	0	2.04E-13	0
	Std	6.46E-13	0	3.71E-13	3.10E-13	1.61E-13	7.40E-13	4.98E-13	6.23E-13	8.74E-13	1.32E-12	9.23E-18	0	4.71E-13	0
	f-rank	12	1	9	6	8	14	7	11	13	10	4	1	5	1
f_{18}	Mean	−2.9407	−99	−20.4045	−99	−11.0537	−12.5152	−4.8240	−25.3180	−98.9999	−99	−99	−99	−19.1129	−99
	Std	1.0692	0	8.6365	0	2.4518	3.1726	1.2623	4.3480	1.51E-04	0	0	0	12.7715	0
	f-rank	14	1	8	1	12	11	13	7	6	1	1	1	9	1
f_{19}	Mean	89.8927	4.29E-48	0.2526	1.06E-23	18.0394	0.0263	642.5729	4.2621	4.5844	7.13E-33	1.36E-27	2.11E-52	0.0093	0
	Std	44.0807	1.76E-47	0.2171	4.14E-24	8.0846	0.0113	147.2318	1.5388	7.1802	6.90E-33	4.27E-28	9.06E-52	0.0103	0
	f-rank	13	3	9	6	12	8	14	10	11	4	5	2	7	1
	Average f-rank	12.8421	2.4211	9.5263	4.7895	10.8421	9.3158	13.2631	9.9474	7.7895	3.9474	4.1053	2	9	1.1579
	Overall f-rank	13	3	10	6	12	9	14	11	7	4	5	2	8	1

Table 4 Statistical conclusions based on Wilcoxon signed-rank test on 100-dimensional benchmark problems

Function	SSA *p*-value	ESSA *p*-value	LSSA *p*-value	MSNSSA *p*-value	CSSA *p*-value	ASSA *p*-value	SSAPSO *p*-value	ISSA *p*-value	GSSA *p*-value	OBSSA *p*-value	ASSO *p*-value	RDSSA *p*-value	IWOSSA *p*-value
f_1	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_2	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_3	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_4	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_5	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_6	3.0199E-11	**0.3953**	3.0199E-11	1.0277E-06	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	**0.3255**	**0.9117**	1.4294E-08	3.0199E-11
f_7	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_8	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_9	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11
f_{10}	1.2118E-12	N/A	1.2118E-12	N/A	1.2118E-12	N/A	1.2118E-12	1.2118E-12	0.0028	N/A	1.2118E-12	N/A	1.2118E-12
f_{11}	1.2118E-12	N/A	1.2118E-12	1.2068E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	N/A	2.5353E-13	3.9410E-12	1.2118E-12
f_{12}	1.2118E-12	N/A	1.2118E-12	N/A	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.3041E-07	N/A	N/A	N/A	1.2118E-12
f_{13}	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_{14}	3.0199E-11	2.9215E-09	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	4.0772E-11	3.0199E-11
f_{15}	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_{16}	1.2118E-12	N/A	1.2118E-12	N/A	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	4.5736E-12	N/A	N/A	N/A	1.2118E-12
f_{17}	1.2118E-12	N/A	1.2118E-12	1.3056E-07	1.2118E-12	1.2118E-12	1.6572E-11	1.2118E-12	1.2118E-12	4.5736E-12	1.2118E-12	N/A	0.0216
f_{18}	1.2118E-12	N/A	1.2118E-12	N/A	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	N/A	N/A	N/A	1.2118E-12
f_{19}	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
+/=/−	19/0/0	11/8/0	19/0/0	15/4/0	19/0/0	18/1/0	19/0/0	19/0/0	19/0/0	13/6/0	15/4/0	13/5/1	19/0/0

Table 5 Comparisons of fourteen algorithms on 7 fixed-dimension test functions

Function	Results	SSA	ESSA	LSSA	MSNSSA	CSSA	ASSA	SSAPSO	ISSA	GSSA	OBSSA	ASSO	RDSSA	IWOSSA	OOSSA
f_{20}	Mean	−1.0316	−1.0312	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0282	−1.0316	−1.0316	−1.0316
	Std	1.50E-14	0.0014	7.08E-11	5.38E-16	4.43E-15	4.23E-09	5.61E-16	1.73E-14	5.96E-14	5.22E-16	0.0042	1.39E-15	7.58E-14	5.29E-16
	f-rank	7	13	11	3	6	12	4	8	9	1	14	5	10	2
f_{21}	Mean	3.0000	3.3399	3.0010	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000	3.4444	8.4000	3.0000	3.0000
	Std	2.42E-13	1.3276	0.0011	9.95E-15	7.04E-14	1.40E-05	6.57E-15	2.89E-13	3.82E-13	6.31E-15	0.4345	10.9846	1.19E-13	7.54E-15
	f-rank	7	12	11	4	5	10	1	8	9	2	13	14	6	3
f_{22}	Mean	−6.8005	−10.1518	−6.2897	−6.1929	−9.7340	−6.8718	−6.3728	−7.6419	−9.3969	−9.4009	−3.8265	−7.7339	−8.1301	−9.0693
	Std	3.3154	0.0015	3.6414	2.1145	1.6279	3.1046	3.0764	3.2250	1.7561	2.2954	0.8724	2.8814	2.9835	2.5136
	f-rank	10	1	12	13	2	9	11	8	4	3	14	7	6	5
f_{23}	Mean	−7.8943	−10.3993	−7.3109	−5.9735	−9.4618	−8.4876	−7.4805	−8.2487	−9.3054	−8.5845	−3.7410	−6.1033	−8.2453	−9.0311
	Std	3.4093	0.0114	3.2499	2.0147	2.4703	3.0322	3.2666	3.3740	2.0828	3.0715	0.8556	3.2402	2.9169	2.8367
	f-rank	9	1	11	13	2	6	10	7	3	5	14	12	8	4
f_{24}	Mean	−8.2854	−10.5347	−7.4346	−6.2101	−9.0455	−9.6255	−8.0589	−9.2001	−8.4318	−8.9849	−3.9731	−8.5302	−9.0925	−9.3506
	Std	3.2684	0.0026	3.2458	2.2002	3.0620	2.3901	3.3677	2.7425	2.6360	2.8689	0.6478	3.1472	2.6596	2.7320
	f-rank	10	1	12	13	6	2	11	4	9	7	14	8	5	3
f_{25}	Mean	−3.8628	−3.7380	−3.8604	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.7876	−3.8112	−3.8628	−3.8628
	Std	1.72E-10	0.0851	0.0029	2.26E-05	5.36E-15	1.88E-05	3.16E-15	1.80E-11	2.34E-11	1.59E-06	0.0555	0.1961	5.93E-09	2.78E-07
	f-rank	5	14	11	10	2	9	1	3	4	8	13	12	6	7
f_{26}	Mean	−3.2371	−2.7719	−3.1831	−3.2570	−3.2698	−3.2385	−3.2588	−3.2421	−3.3184	−3.2108	−2.5968	−3.2866	−3.2906	−3.2619
	Std	0.0668	0.3502	0.1366	0.0782	0.0660	0.0558	0.0605	0.0625	0.0218	0.0618	0.2051	0.0556	0.0536	0.0615
	f-rank	10	13	12	7	4	9	6	8	1	11	14	3	2	5
	Average f-rank	8.2857	7.8517	11.4286	10.4286	3.8571	8.1429	6.2857	6.5714	5.5714	5.2857	13.7143	8.7143	6.1429	4.1429
	Overall f-rank	10	8	13	12	1	9	6	7	4	3	14	11	5	2

SSA shows significant advantages on 15 benchmark functions except f_{10}, f_{12}, f_{16}, and f_{18}. For the test functions f_{10}, f_{12}, f_{16}, and f_{18}, the two algorithms show the same performance. This shows that LOBL in orthogonal opposition-based learning is more helpful for the algorithm to jump out of the local optimum than OBL. According to Friedman ranking test results, OOSSA ranks first and OLOBL-SSA ranks second, followed by DL-SSA, OBL-SSA and SSA, which further verifies the effectiveness of the components and the performance of OOSSA is obviously better than that of SSA variants with a single component.

4.4 High-dimensional performance analysis

To demonstrate the feasibility of applying OOSSA to solve large-scale problems, 19 test problems listed in Table 1 (f_1~f_{19}) are chosen for simulation experiments, and the function dimension is set to 10000. The parameter settings remain the same as those used in the previous experiment. Table 7 shows the best solution, worst solution, average and standard deviation obtained by OOSSA in 30 independent experiments on 19 large-scale problems. In addition, the success rate (SR%) metric is used to evaluate the efficiency of the proposed methodology in solving large-scale optimization problems. The criterion for judging whether a solution is successful is as follows:

$$\begin{cases} \frac{|f_A - f_T|}{f} < 10^{-5}, f_T \neq 0 \\ |f_A - f_T| < 10^{-5}, f_T = 0 \end{cases} \tag{25}$$

where f_A is the result obtained by the algorithm on the test function, and f_T is the theoretical optimal value of the test function.

According to the statistical results in Table 7, for large-scale numerical optimization problems, OOSSA can converge to the theoretical optimal value on 16 test functions except f_6, f_9, f_{11}, and f_{14}. From the standard deviation, OOSSA has the same superior stability while maintaining high solution accuracy. For functions f_6, f_9, f_{11}, and f_{14}, although OOSSA failed to find the theoretical optimal solution, the results obtained are still satisfactory. Compared to the results obtained on functions with dimensions 100, the result on f_6 is in the same order of magnitude, the results on f_9 is slightly inferior, but the convergence accuracy is still considerable, while on f_{11}, the same results are obtained, and for f_{13}, the similar result is obtained. This fully demonstrates that OOSSA algorithm is strongly robust in dealing with large-scale optimization problems. In terms of success rate, OOSSA can achieve 100% on the other 17 test functions except for 0 on test function f_9 and 20% on test function f_6, which verifies that OOSSA algorithm is an effective tool for solving large-scale optimization problems.

4.5 Comparison with other swarm-based intelligent algorithms

To further testify the overall performance of OOSSA, it is compared with nine state-of-the-art algorithms. They are Marine Predators Algorithm (MPA) [116], Equilibrium Optimizer (EO) [117], Tunicate Swarm Algorithm (TSA) [129], Selective Opposition Based Grey Wolf Optimization (SOGWO) [130], Improved Grey Wolf Algorithm (IGWO) [131], Henry Gas Solubility (HGS) Optimization [132], Memetic Harris Hawk Optimization (EESHHO) [133], Arithmetic Optimization Algorithm (ArOA) [134], Archimedes Optimization Algorithm (AOA) [135], Moth-Flame Optimization Algorithm Based on Diversity and Mutation Mechanism (DMMFO) [136], Moth Flame Optimizer with Double Adaptive Weights (WEMFO) [137], Opposition-based Learning Grey Wolf Optimizer (OGWO) [138]. These are cutting-edge algorithms that have been verified to have good optimization performance and have been successfully applied to solve a variety of optimization problems. Therefore, by comparing with these algorithms, we can effectively authenticate the effectiveness and superiority of OOSSA. In this section, some of the test functions in Table 1 (f_1~f_{19}) are selected for simulation experiment, and the function dimension is set to 100. To ensure the fairness of the comparative experiments, the parameters of the compared algorithms are consistent with the original paper, which can provide assurance that each method to take full advantage of its overall performance. Table 8 shows the average value and the standard deviation obtained from 30 independent experiments by 13 algorithms on 19 test functions, and the Friedman test is also employed to double-check the performance of OOSSA. Apart from these, Wilcoxon's rank sum test at a 0.05 significance level is applied to analyze the performance gap between OOSSA and its peers.

According to the comparison results provided in Table 8 between OOSSA and the applied renowned algorithms, OOSSA outperforms TSA, SOGWO, IGWO, DMMFO, and OGWO on all test cases. With respect to MPA, EO, AOA, and WEMFO, OOSSA provides better results on 15, 17, 16, and 12 problems, respectively, and for the remaining cases they obtain similar performance. Compared with HGS, OOSSA achieve the better and similar values for 12 and five benchmarks, respectively. For other two functions f_9 and f_{14}, HGS yields slightly superior results. With regard to ArOA, OOSSA gets more promising and comparable results on 15 and three problems, respectively, while the better result is given by ArOA on the remaining function f_6. From the comparison results of OOSSA and EESHHO on 19 benchmarks, the proposed methodology gets more potential and resemble results

Table 6 Comparisons of OLOBL-SSA, DL-SSA, OOBL-SSA, and OOSSA on 19 test functions with 100 dimensions

Function	Results	SSA	OLOBL-SSA	DL-SSA	OOBL-SSA	OOSSA
f_1	Mean	1.41E+03	0	4.82E-39	1.62E-22	0
	Std	396.0337	0	2.42E-39	1.90E-22	0
	f-rank	5	1	3	4	1
f_2	Mean	47.4085	0	5.60E-20	7.63E-12	0
	Std	5.8319	0	1.74E-20	3.35E-12	0
	f-rank	5	1	3	4	1
f_3	Mean	5.06E+04	0	1.05E-37	1.66E-22	0
	Std	2.66E+04	0	1.10E-37	1.64E-22	0
	f-rank	5	1	3	4	1
f_4	Mean	28.1304	0	1.88E-20	2.09E-12	0
	Std	4.5227	0	5.98E-21	1.47E-13	0
	f-rank	5	1	3	4	1
f_5	Mean	875.4389	0	2.17E-39	4.66E-23	0
	Std	275.0973	0	1.30E-39	3.16E-23	0
	f-rank	5	1	3	4	1
f_6	Mean	2.6832	8.92E-05	1.04E-04	1.20E-04	4.38E-05
	Std	0.6420	8.18E-05	1.05E-04	1.24E-04	5.34E-05
	f-rank	5	2	3	4	1
f_7	Mean	7.61E+07	0	3.55E-34	3.25E-18	0
	Std	2.95E+07	0	1.75E-34	3.24E-18	0
	f-rank	5	1	3	4	1
f_8	Mean	1.29E+07	0	4.55E-35	1.99E-18	0
	Std	4.16E+06	0	2.49E-35	2.76E-18	0
	f-rank	5	1	3	4	1
f_9	Mean	33.0015	0.1452	0.1709	0.1629	0.0339
	Std	9.8040	0.0345	0.0337	0.0380	0.0062
	f-rank	5	2	4	3	1
f_{10}	Mean	249.9030	0	0	0	0
	Std	44.9377	0	0	0	0
	f-rank	5	1	1	1	1
f_{11}	Mean	10.2687	8.88E-16	4.44E-15	1.61E-12	8.88E-16
	Std	1.0978	0	0	8.88E-13	0
	f-rank	5	1	3	4	1
f_{12}	Mean	12.8754	0	0	0	0
	Std	3.5134	0	0	0	0
	f-rank	5	1	1	1	1
f_{13}	Mean	1.01E+03	0	2.37E-39	1.15E-24	0
	Std	610.0508	0	2.74E-39	1.83E-24	0
	f-rank	5	1	2	3	1
f_{14}	Mean	72.3570	189.1811	5.61E-41	188.6257	1.63E-76
	Std	16.3749	40.3712	3.12E-41	30.0553	5.56E-76
	f-rank	3	5	2	4	1
f_{15}	Mean	5.0618	0	9.03E-11	1.03E-06	0

Table 6 (continued)

Function	Results	SSA	OLOBL-SSA	DL-SSA	OOBL-SSA	OOSSA
	Std	0.2752	0	1.30E-11	2.45E-07	0
	f-rank	5	1	3	4	1
f_{16}	Mean	475.4789	0	0	0	0
	Std	112.1545	0	0	0	0
	f-rank	5	1	1	1	1
f_{17}	Mean	6.91E-12	0	7.89E-12	7.96E-12	0
	Std	4.49E-13	0	7.73E-13	8.37E-13	0
	f-rank	5	1	5	4	1
f_{18}	Mean	−2.6258	−99	−39.3082	−99	−99
	Std	0.9471	0	12.2275	0	0
	f-rank	5	1	4	1	1
f_{19}	Mean	92.2466	0	4.65E-40	4.14E-24	0
	Std	28.3720	0	3.04E-40	4.08E-24	0
	f-rank	5	1	3	4	1
	Average f-rank	4.8947	1.3157	2.7895	3.2632	1
	Overall f-rank	5	2	3	4	1

on seven and 11 test functions, respectively. For the other benchmark function f_9, EESHHO displays better performance. Additionally, according to the ranking of all methods in Friedman test, the OOSSA obtains the top rank, followed by EESHHO, HGS, AOA, MPA, EO, ArOA, WEMFO, OGWO, SOGWO, IGWO, TSA, and DMMFO, which indicates that OOSSA has the most outstanding performance among all competitors.

Table 9 shows the p-values of the Wilcoxon test obtained for the involved thirteen algorithms on 19 classical functions with 100 dimensions. As can be seen from the statistical results, the p-values of the following cases are greater than 0.05: OOSSA versus HGS on f_{17}, OOSSA versus EO on f_9, f_{16}, and f_{17}; OOSSA versus EESHHO on f_7 and f_{19}, OOSSA versus ArOA on f_6, OOSSA versus AOA on f_{16}. All other p-values are less than 0.05, which implies that the overall performance of the OOSSA algorithm has a significant advantage over the other twelve well-performance cutting-edge optimization algorithms.

4.6 Convergence analysis

To test the convergence performance of the proposed OOSSA algorithm, we select some representative benchmark functions in Table 1 with 100 dimensions, and show the convergence curves of OOSSA, ESSA, LSSA, MSNSSA, CSSA, ASSA, SSAPSO, ISSA, GSSA, OBSSA, ASSO, RDSSA, IWOSSA and SSA on these test cases in Fig. 11, and the convergence graphs of OOSSA, TSA, SOGWO, MPA, HGS, EO, EESHHO, ArOA, AOA, IGWO, WEMFO, DMMFO, and OGWO on the applied benchmarks are drawn in Fig. 12. The convergence plots can help to analyze the convergence trend of OOSSA in a more intuitive way.

It can be clearly seen from Fig. 11 that OOSSA has a higher solution accuracy and faster convergence speed for all functions compared to the other thirteen SSA-based competitors. It is note that for test function f_{11}, although EESSA obtains the same solution accuracy as OOSSA, it performs significantly worse than OOSSA in terms of convergence speed. On the other hand, from the curves presented in Fig. 12, the convergence trend of OOSSA outperforms all involved peers on most cases. On f_{11}, OOSSA reaches the same solution accuracy as EESHHO and HGS, but the suggested method exhibits a substantial advantage in terms of convergence speed. In addition, the better solution accuracy is achieved by HGS on f_{14}, while OOSSA shows a competitive convergence rate in the early stages, but it falls in to a local optimal in the later iterations. Based on the above assessment, we can assert that OOSSA has an outstanding convergence trend, facilitated by the fact that the developed approach focuses on a delicate balance between exploratory and exploitative inclinations.

5 OOSSA for engineering design problems

To test the effectiveness of OOSSA in solving practical problems, we applied OOSSA to three classical engineering design problems named pressure vessel design, I-beam design, and cantilever beam design. Even though the problems have several constraints, OOSSA still expects to handle these

Table 7 Results obtained by OOSSA on 10,000-dimensional functions

Function	OOSSA				
	Best	Worst	Mean	Std	SR%
f_1	0	0	0	0	100
f_2	0	0	0	0	100
f_3	0	0	0	0	100
f_4	0	0	0	0	100
f_5	0	0	0	0	100
f_6	2.65E-06	2.24E-04	5.45E-05	5.42E-05	20
f_7	0	0	0	0	100
f_8	0	0	0	0	100
f_9	0.2149	0.2770	0.2566	0.0245	0
f_{10}	0	0	0	0	100
f_{11}	8.88E-16	8.88E-16	8.88E-16	0	100
f_{12}	0	0	0	0	100
f_{13}	0	0	0	0	100
f_{14}	9.04E-83	3.09E-75	6.19E-76	1.38E-75	100
f_{15}	0	0	0	0	100
f_{16}	0	0	0	0	100
f_{17}	0	0	0	0	100
f_{18}	−9999	−9999	−9999	0	100
f_{19}	0	0	0	0	100

constraints and obtain the optimal solution. The experimental data of the compared algorithms are referred to the original literature.

5.1 Pressure vessel design

The objective of the pressure vessel design problem is to minimize fabrication cost, which include welding, materials and forming. As shown in Fig. 13, the pressure vessel design problem can be solved by determining the optimal values of four design variables: thickness of head T_h, thickness of shell T_s, inner radius R, and length of cylindrical shell L.

The mathematical formulations of this problem are as follows:

Consider $\overrightarrow{x} = [x_1\ x_2\ x_3\ x_4] = [T_s\ T_h R L]$.

Minimize $f(\overrightarrow{x}) = 0.6224x_1x_2\ \ x_3 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$.

Subject to

$$g_1\left(\overrightarrow{x}\right) = -x_1 + 0.0193x_3 \le 0,$$

$$g_2\left(\overrightarrow{x}\right) = -x_3 + 0.00954x_3 \le 0,$$

$$g_3\left(\overrightarrow{x}\right) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0,$$

$$g_4\left(\overrightarrow{x}\right) = x_4 - 240 \le 0.$$

Variable range $0 \le x_1, x_2 \le 99$, $10 \le x_3, x_4 \le 200$.

OOSSA is used to solve the pressure vessel design problem and ten other algorithms are selected for comparison, and the results obtained are listed in Table 10. It should be noted that the results of the compared algorithms are directly derived from the previous work. From Table 10, it can be seen that for the pressure vessel design problem, OOSSA obtains optimal results and has a greater advantage over the other compared algorithms.

5.2 I-beam design problem

Another practical engineering problem used in this section is the I-beam design problem. The main objective of this problem is to design an I-shaped beam within a minimum vertical deflection. As shown in Fig. 14, this problem includes four structural parameters: length (b), height (h) and two thicknesses (t_w and t_f).

The mathematical model of this problem can be constructed as below:

(Consider) $\overrightarrow{x} = [x_1\ x_2\ x_3\ x_4] = [bht_w\ t_f]$.

(Minimize) $f(\overrightarrow{x}) = \frac{5000}{t_w(h-2t_f)} \frac{3}{12+\frac{bt_f^3}{6}+2bt_f\left(\frac{h-t_f}{2}\right)^2}$.

Subject to $g(\overrightarrow{x}) = 2bt_w + t_w(h-2t_f) \le 0$.

Variable range $10 \le x_1 \le 50$, $10 \le x_2 \le 80$,$0.9 \le x_3 \le 5$, $0.9 \le x_4 \le 5$.

OOSSA was used to solve the I-beam design problem and compared with eight algorithms, and the results obtained are given in Table 11. From the results listed in the table, for the I-beam design problem, OOSSA obtained the same and similar results as the BWOA and SSA, and significantly outperformed the other six compared algorithms. This proves that OOSSA has a strong competitive edge in this problem.

5.3 Cantilever beam design

This section tests the performance of OOSSA using the cantilever beam design problem whose main purpose is to minimize the weight of the cantilever beam. As illustrated in Fig. 15, the cantilever beam contains five hollow cells with square cross sections, each defined by a variable with a constant thickness, and thus contains a total of five structural parameters. The problem can be solved by determining the optimal values of the five structural parameters.

The mathematical expressions of the cantilever beam design problem are as follows:

(Minimize) $f(\overrightarrow{x}) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5)$.

Subject to $g(\overrightarrow{x}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} \le 1$.

Variable range $0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100$.

Table 8 Comparisons of thirteen algorithms on 19 test functions with 100 dimensions

Function	Results	TSA	SOGWO	MPA	HGS	EO	EESHHO	ArOA	AOA	IGWO	WEMFO	DMMFO	OGWO	OOSSA
f_1	Mean	3.54E-10	3.07E-12	2.20E-19	1.42E-143	5.99E-29	0	0.0258	2.43E-82	2.51E-12	2.14E-23	3.29E+04	3.67E-15	0
	Std	3.58E-10	2.72E-12	2.45E-19	7.80E-143	1.70E-28	0	0.0087	1.32E-81	1.97E-12	1.01E-22	7.24E+03	4.52E-15	0
	f-rank	11	10	7	3	5	1	12	4	9	6	13	8	1
f_2	Mean	2.12E-07	3.89E-08	1.44E-11	8.23E-71	1.60E-17	2.20E-181	1.51E-60	2.46E-43	1.52E-08	1.77E-14	224.7894	4.04E-10	0
	Std	1.38E-07	1.24E-08	1.10E-11	4.51E-11	9.44E-18	0	8.25E-60	1.12E-42	6.49E-09	2.42E-14	41.5279	2.77E-10	0
	f-rank	12	11	8	3	6	2	4	5	10	7	13	9	1
f_3	Mean	1.26E-04	1.70E+03	8.8866	7.04E-14	4.6673	1.14E-203	0.6431	3.77E-64	5.89E+03	1.77E-10	2.32E+05	920.5124	0
	Std	7.42E+03	1.47E+03	12.3809	3.86E-13	10.9024	0	0.5472	1.70E-63	2.85E+03	7.10E-10	3.53E+04	928.2126	0
	f-rank	4	11	9	5	8	2	7	3	12	6	13	10	1
f_4	Mean	56.0193	0.8555	2.35E-07	3.45E-68	0.0043	1.70E-139	0.0891	2.43E-38	3.7332	4.44E-10	88.1379	2.2099	0
	Std	12.4441	0.6042	1.29E-07	1.85E-67	0.0105	9.32E-139	0.0119	7.88E-38	2.3007	1.44E-09	2.4502	2.2689	0
	f-rank	12	9	6	3	7	2	8	4	11	5	13	10	1
f_5	Mean	1.73E-10	4.86E-13	6.87E-20	9.97E-158	1.40E-29	0	2.77E-106	3.85E-78	1.14E-12	9.07E-24	1.45E+04	7.378E-16	0
	Std	2.51E-10	3.03E-13	5.04E-20	5.46E-157	1.77E-29	0	1.52E-105	1.82E-77	1.18E-12	2.69E-23	2.68E+03	7.42E-16	0
	f-rank	12	10	8	3	6	1	4	5	11	7	13	9	1
f_6	Mean	0.0489	0.0070	0.0019	0.0012	0.0025	2.58E-04	5.49E-05	6.47E-04	0.0130	0.0020	90.7925	0.0025	6.63E-05
	Std	0.0165	0.0021	9.33E-04	0.0021	0.0014	3.35E-04	6.08E-05	3.43E-04	0.0046	0.0013	31.1422	0.0022	5.24E-05
	f-rank	12	10	6	5	8	3	1	4	11	7	13	9	2
f_7	Mean	6.98E-07	3.83E-09	7.61E-156	3.49E-167	1.21E-25	2.40E-322	51.8390	1.04E-75	2.95E-09	1.28E-18	1.91E+08	5.90E-12	0
	Std	1.14E-06	3.00E-09	4.17E-155	0	1.17E-25	0	57.8071	4.43E-75	2.04E-09	6.67E-18	7.26E+07	9.26E-12	0
	f-rank	11	10	4	3	6	2	12	5	9	7	13	8	1
f_8	Mean	2.41E-06	1.52E-08	2.73E-15	5.34E-144	4.14E-25	0	1.68E-67	1.47E-76	2.62E-08	1.49E-17	3.17E+08	3.39E-11	0
	Std	2.32E-06	1.02E-08	2.50E-15	2.91E-143	4.99E-25	0	9.19E-67	7.66E-76	2.30E-08	8.09E-17	6.60E+07	4.26E-11	0
	f-rank	12	10	8	3	6	1	5	4	11	7	13	9	1
f_9	Mean	11.0946	0.2861	0.0448	6.41E-06	0.0388	4.69E-05	0.9009	1.0893	0.1933	0.4060	7.59E+07	0.2633	0.0365
	Std	4.5679	0.0615	0.0092	1.14E-05	0.0080	2.90E-05	0.0203	0.0553	0.0502	0.0631	3.23E+07	0.0871	0.0071
	f-rank	12	8	5	1	4	2	10	11	6	9	11	7	3
f_{10}	Mean	999.7646	10.8969	0	0	0	0	0	0	130.0050	446.2258	847.8756	1.1783	0
	Std	119.4328	5.8513	0	0	0	0	0	0	52.0586	357.3273	64.0263	2.5308	0

Table 8 (continued)

Function	Results	TSA	SOGWO	MPA	HGS	EO	EESHHO	ArOA	AOA	IGWO	WEMFO	DMMFO	OGWO	OOSSA
	f-rank	13	9	1	1	1	1	1	1	10	11	12	8	1
f_{11}	Mean	3.97E-06	1.31E-07	4.98E-11	8.88E-16	3.45E-14	8.88E-16	6.78E-04	19.9668	1.56E-07	1.3723	19.7511	5.73E-09	8.88E-16
	Std	3.24E-06	4.93E-08	2.56E-11	0	5.73E-15	0	0.0012	2.34E-05	7.04E-08	5.0639	0.1190	2.65E-09	0
	f-rank	9	7	5	1	4	1	10	13	8	11	12	6	1
f_{12}	Mean	0.0134	0.0090	0	0	0	0	608.0626	0	0.0044	0	307.1925	0.0022	0
	Std	0.0164	0.0150	0	0	0	0	174.0302	0	0.0071	0	84.3708	0.0091	0
	f-rank	11	10	1	1	1	1	13	1	9	1	12	8	1
f_{13}	Mean	3.86E-12	2.04E-14	9.30E-21	2.25E-40	9.12E-31	0	9.88E-105	9.80E-79	2.61E-14	5.30E-25	493.9897	7.71E-17	0
	Std	4.26E-12	1.15E-14	1.01E-20	1.23E-39	1.08E-30	0	5.41E-104	3.74E-78	1.87E-14	2.01E-24	112.6501	1.03E-16	0
	f-rank	12	10	8	5	6	1	3	4	11	7	13	9	1
f_{14}	Mean	8.33E-12	2.89E-13	4.25E-19	1.93E-113	5.01E-28	1.43E-08	1.36E+03	1.27E-71	2.36E-12	2.09E-25	518.0831	9.98E-16	7.47E-75
	Std	1.01E-11	2.32E-13	3.59E-19	1.06E-112	7.36E-28	5.95E-08	143.8588	4.62E-71	1.99E-12	5.97E-25	77.7780	9.45E-16	4.08E-74
	f-rank	10	8	6	1	4	11	13	3	9	5	12	7	2
f_{15}	Mean	3.1302	0.0103	6.74E-07	2.37E-39	1.36E-09	0	0.0032	9.00E-22	0.0103	1.36E-08	8.5266	8.27E-05	0
	Std	1.5108	0.0027	6.17E-07	7.80E-39	4.89E-10	0	0.0084	2.03E-21	0.0028	1.46E-08	0.3173	4.18E-05	0
	f-rank	12	11	7	3	5	1	9	4	10	6	13	8	1
f_{16}	Mean	988.0366	27.1090	0	0	0.0333	0	0	22.9166	277.3652	397.1056	846.4746	13.8193	0
	Std	93.6936	50.2783	0	0	0.1826	0	0	125.5196	136.9208	380.1985	69.5645	37.3923	0
	f-rank	13	9	1	1	6	1	1	8	10	11	12	7	1
f_{17}	Mean	1.52E-11	1.45E-11	0	1.65E-15	1.93E-15	0	0	0	1.68E-11	1.1E-12	3.05E-12	1.41E-11	0
	Std	5.91E-13	5.82E-13	0	9.01E-15	1.04E-14	0	0	0	4.12E-13	5.35E-13	7.23E-13	6.81E-13	0
	f-rank	12	11	1	6	7	1	1	1	13	8	9	10	1
f_{18}	Mean	−5.9377	−16.4112	−71.6466	−99	−23.6364	−99	−98.5716	−36.6699	−5.1035	−44.4780	−6.1754	−35.5503	−99
	Std	1.6435	5.1955	33.9875	0	10.3757	0	0.1270	41.5391	4.3060	43.2858	1.1354	10.2110	0
	f-rank	12	10	5	1	9	1	4	7	13	6	11	8	1
f_{19}	Mean	0.0012	6.17E-05	1.12E-19	3.11E-132	3.68E-14	7.72E-310	2.92E-141	6.13E-77	7.92E-04	1.05E-24	9.10E+03	2.93E-05	0
	Std	0.0012	5.05E-05	4.03E-19	1.68E-131	1.81E-13	0	1.60E-140	3.35E-76	6.21E-04	5.12E-24	3.43E+03	2.78E-05	0
	f-rank	12	10	7	4	8	2	3	5	11	6	13	9	1
	Average f-rank	11.2631	9.6842	5.4211	2.7895	5.6316	1.9474	6.3158	4.8947	10.2105	7	12.3158	8.3684	1.2105
	Overall f-rank	12	10	5	3	6	2	7	4	11	8	13	9	1

Table 9 Statistical conclusions based on Wilcoxon signed-rank test on 100-dimensional benchmark problems

Function	TSA *p*-value	SOGWO *p*-value	MPA *p*-value	HGS *p*-value	EO *p*-value	EESHHO *p*-value	ArOA *p*-value	AOA *p*-value	IGWO *p*-value	WEMFO *p*-value	DMMFO *p*-value	OGWO *p*-value
f_1	1.2118E-12	1.2118E-12	1.2118E-12	2.9343E-05	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_2	1.2118E-12	1.2118E-12	1.2118E-12	6.6096E-05	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_3	1.2118E-12	1.2118E-12	1.2118E-12	3.4526E-07	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_4	1.2118E-12	1.2118E-12	1.2118E-12	1.2717E-05	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_5	1.2118E-12	1.2118E-12	1.2118E-12	6.6167E-04	1.2118E-12	N/A	0.0216	5.7720E-11	1.2118E-12	1.2118E-12	1.2118E-12	1.6572E-11
f_6	3.0199E-11	3.0199E-11	3.0199E-11	1.3250E-04	3.0199E-11	0.0061	**0.2643**	6.6955E-11	3.0199E-11	4.5043E-11	3.0199E-11	3.3384E-11
f_7	1.2118E-12	1.2118E-12	1.2118E-12	0.0028	1.2118E-12	**0.3337**	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_8	1.2118E-12	1.2118E-12	1.2118E-12	2.2130E-06	1.2118E-12	N/A	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_9	3.0199E-11	3.0199E-11	0.0012	3.0199E-11	**0.2062**	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11
f_{10}	1.2118E-12	1.2118E-12	N/A	N/A	N/A	N/A	N/A	N/A	1.2118E-12	4.7884E-08	1.2118E-12	1.2118E-12
f_{11}	1.2118E-12	1.2118E-12	1.2118E-12	N/A	9.1148E-13	N/A	6.6096E-05	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_{12}	1.2118E-12	1.2118E-12	N/A	N/A	N/A	N/A	1.2118E-12	N/A	1.2118E-12	N/A	1.2118E-12	4.5664E-12
f_{13}	1.2118E-12	1.2118E-12	1.2118E-12	3.1349E-04	1.2118E-12	N/A	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_{14}	3.0199E-11	3.0199E-11	3.0199E-11	1.9558E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0939E-06	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11
f_{15}	1.2118E-12	1.2118E-12	1.2118E-12	3.1349E-04	3.0199E-11	N/A	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_{16}	1.2118E-12	1.2118E-12	N/A	N/A	**0.3337**	N/A	N/A	**0.3337**	1.2118E-12	1.7016E-08	1.2118E-12	1.2118E-12
f_{17}	1.2118E-12	1.2118E-12	N/A	**0.3337**	**0.1608**	N/A	N/A	N/A	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
f_{18}	1.2118E-12	1.2118E-12	1.2118E-12	N/A	1.2118E-12	N/A	1.2118E-12	4.7884E-08	1.2118E-12	1.6572E-11	1.2118E-12	1.2118E-12
f_{19}	1.2118E-12	1.2118E-12	1.2118E-12	1.2717E-05	1.2118E-12	**0.3337**	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12	1.2118E-12
+/=/−	19/0/0	19/0/0	15/4/0	12/5/2	14/5/0	8/10/1	15/4/0	19/0/0	19/0/0	18/1/0	19/0/0	19/0/0

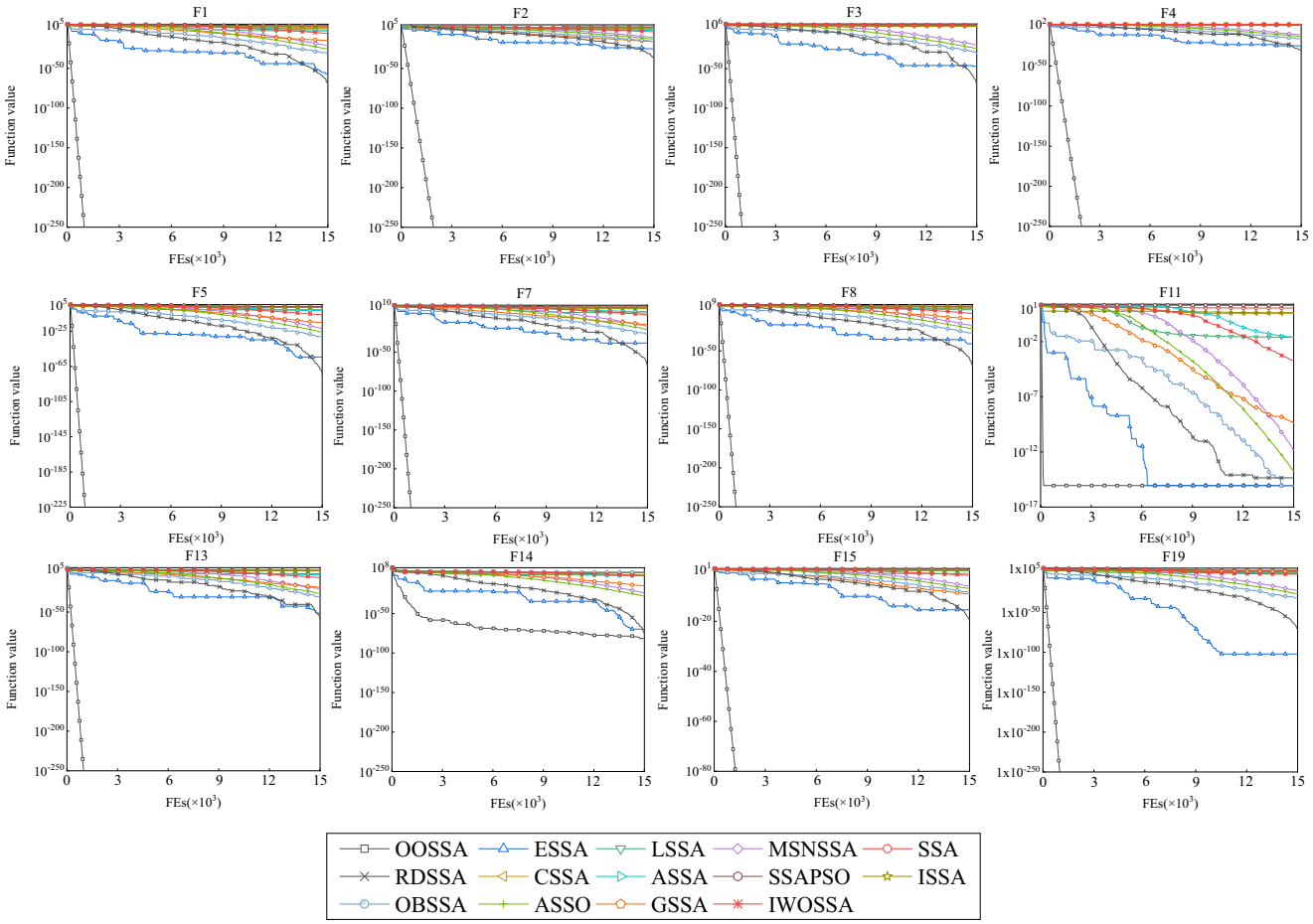


Fig. 11 Convergence curves of SSA-based algorithms for twelve representative test functions

OOSSA is used to solve the cantilever beam design problem and compared with other eleven advanced algorithms, and the obtained results are listed in Table 12. From the results listed in the table, OOSSA performs much better than other compared algorithms for the cantilever beam design problem, which further validates the superior performance of the proposed methodology for the practical engineering optimization problem.

5.4 OOSSA for parameter estimation of photovoltaic model

Solar energy is considered to be an environmentally friendly renewable energy source and has gained a lot of attention in recent years. By accurately predicting solar photovoltaic (PV) characteristics, the performance of PV cell systems can be optimized. Extracting PV parameters is a hot research problem in the field of solar PV systems. Among the models describing solar cell characteristics, the single diode model (SDM) is the most popular and its structure is shown in Fig. 16. To calculate the output current of the SDM, the following mathematical model is used [145].

$$\begin{aligned} I_L &= I_{ph} - I_d - I_{sh} \\ &= I_{ph} - I_{sd} \cdot \left[\exp\left(\frac{q \cdot (V_L + R_S \cdot I_L)}{n \cdot k \cdot T} \right) - 1 \right] - \frac{V_L + R_S + I_L}{R_{sh}} \end{aligned} \quad (26)$$

where I_L represents the output current, I_{ph} represents the photo-generated current, I_d denotes the diode current, and I_{sh} is the shunt resistor current, I_{sd} denotes the reverse saturation current of diode, n is the ideality factor, k denotes the Boltzmann constant, q represents the electron charge, the series and shunt resistances are represented by R_s and R_{sh}, respectively. In the SDM, five parameters (I_{ph}, I_{sd}, R_s, R_{sh}, and n) are needed to be identified.

By introducing root mean square error (RMSE), the PV parameter identification problem can be reformulated as an optimization problem and solved by the metaheuristic algorithm. The objective of the optimization is to minimize the error between the measured data and simulated data. The objective function of the optimization problem is defined as:

$$F(\mathrm{X}) = \sqrt{\frac{1}{N} \sum_{k=1}^{N} f(V_L, I_L, \mathrm{X})^2} \quad (27)$$

where N represents the number of experimental data.

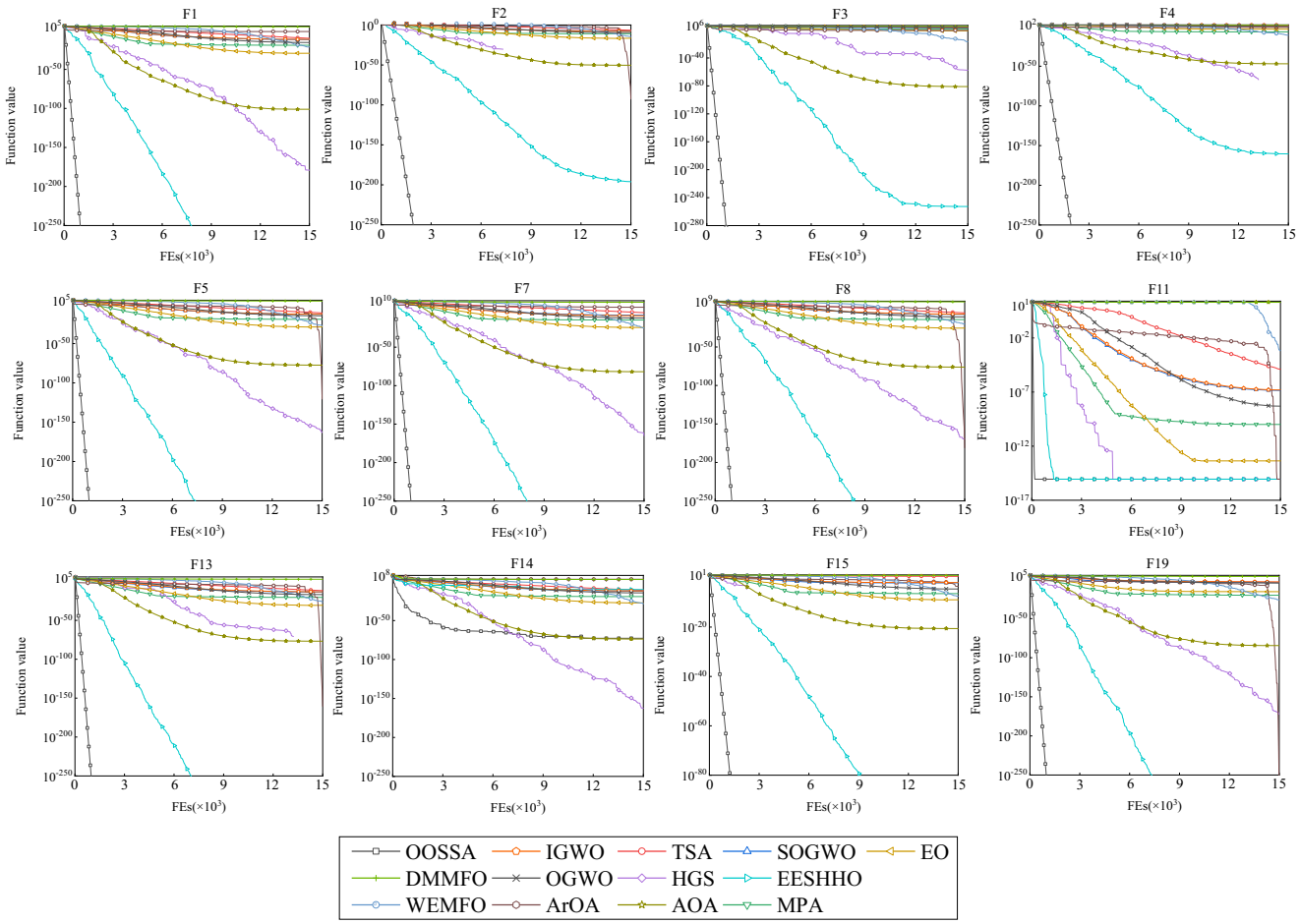


Fig. 12 Convergence curves of OOSSA and twelve cutting-edge algorithms for twelve representative test functions

In Eq. (27), for the SDM:

$$\begin{cases} f(V_L, I_L, \mathrm{X}) = I_{ph} - I_{sd} \cdot \left[\exp\left(\dfrac{q \cdot (V_L + R_S \cdot I_L)}{n \cdot k \cdot T} \right) - 1 \right] \\ \qquad\qquad - \dfrac{V_L + R_S \cdot I_L}{R_{sh}} - I_L \\ \mathrm{X} = \{ I_{ph}, I_{sd}, R_S, R_{sh}, n \} \end{cases} \tag{28}$$

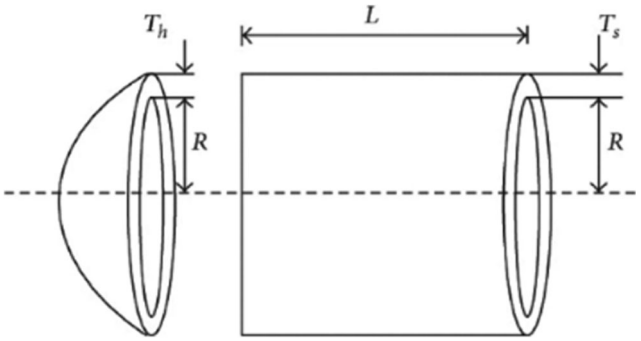


Fig. 13 Pressure vessel design problem

where $0 \le I_{ph} \le 1$, $0 \le I_{sd} \le 1$, $0 \le R_S \le 0.5$, $0 \le R_{sh} \le 100$, and $1 \le n \le 2$.

We applied OOSSA to extract the parameters of the SDM and compared it with the canonical SSA algorithm and the other nine well-known algorithms. All algorithms were performed 30 times independently, each run evaluating the fitness function 30,000 times. The experimental parameters were used as provided in the literature [146]. The irradiance was 1000 W/m^2 and the temperature was 33°C. The experimental results are shown in Table 13, and the I-V and P-V characteristic curves are plotted in Fig. 17.

6 Implementation of OOSSA for mobile robot path planning

With the sustainable advancement of intelligent technology, autonomous mobile robots (AMRs) are applied in an ever increasing number of fields, such as intelligent transport, intelligent handling. Path planning is a fundamental and essential technology in AMRs, the task of which is to generate a shortest obstacle-free trajectory route from a departure point to a target point. Nature-inspired population-based intelligence

Table 10 Comparison of optimization results for pressure vessel design

Algorithm	T_h	T_s	L	R	Optimal cost
OOSSA	0.779522	0.390588	40.38971848	199.026456	5902.93731
SSA [62]	0.790678	0.390834	40.96773875	195.91822	6012.1885
GWO [41]	0.8125	0.4345	42.098181	176.75873	6051.5639
WOA [49]	0.8125	0.4375	42.0982699	176.639	6059.741
MFO [115]	0.8125	0.4375	42.098445	176.636596	6059.7143
MVO [46]	0.8125	0.4375	42.0907382	176.73869	6060.8066
MPA [116]	0.8125	0.4375	42.098445	176.636607	6059.7144
EO [117]	0.8125	0.4375	42.0984456	176.6365958	6059.7143
HHO [56	0.817584	0.407293	42.009174576	176.071964	6000.46259
SCA [90]	0.8125	0.43375	42.04861	177.7078	6076.3651
CSS [122]	0.8125	0.4375	42.103624	176.572656	6059.0888

techniques are enjoying increasing popularity in mobile robot path planning (MRPP), e.g. PSO, GA, and ABC algorithms have been implemented to guide AMRs from one location to another, and safe and less time-consuming tracks have been projected successfully. In this section, we propose a novel OOSSA-based MRPP approach to handle the task of planning shortest collision-free path for AMRs in different workspace.

6.1 Robot path-planning problem description

Topologically, the MRPP project is associated with the shortest route issue of discovering a route between initial and the destination in a diagram. which is generally expressed as an optimization problem and can be solved using swarm intelligence techniques. The task to be achieved by optimization is to find a shortest possible trail from the source to the terminal, while all threating regions need to be avoided. The core of solving this intractable problem is to establish an efficient fitness function, and the developed OOSSA-based MRPP approach manipulates it through continues evaluation to derive the optimal solution, i.e. to generate the safe and shortest path. Based on the above analysis, we designed the fitness function by considering route path length and conflict avoidance to evaluate the quality of the trajectories generated by the OOSSA-based MRPP method, which is calculated as follows:

$$F = L(1 + \varpi \cdot \eta) \tag{29}$$

where ϖ is a control parameter for encouraging safe paths and rejecting routes that collide with threatening areas, L represents the path length and to calculate it, the following equation is employed.

$$L = \sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{30}$$

where (x_i, y_i) denotes the coordinates of the ith interpolation point.

In Eq. (29), η is a flag variable that serves to quantify the safety status of the path. To calculate it, the following equation is presented.

Fig. 14 I-beam design problem

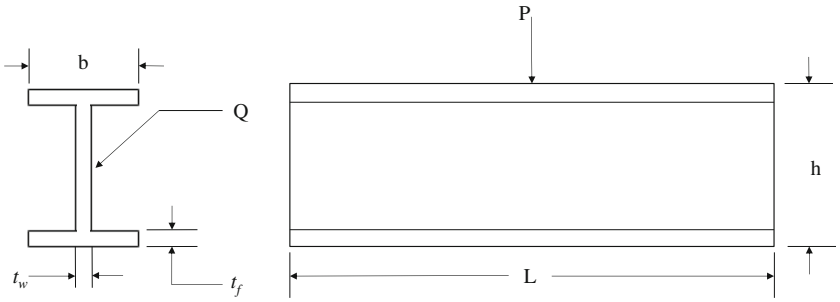


Table 11 Comparison of optimization results for I-beam design problem

Algorithm	b	h	t_w	t_f	Optimum vertical deflection
OOSSA	50	80	1.76470588	5	0.0066259581
SSA [62]	50	80	1.76470587	5.0000	0.0066259581
WOA [49]	49.99799	80	1.7647477	5	0. 00662619
MFO [115]	50	80	1.7647	5	0.0066259
BWOA [118]	50	80	1.76470588	5	0.0066259581
SOS [119]	50	80	0.9	2.32179	0.0130741
ARSM [120]	37.05	80	1.71	2.31	0.0157
IARSM [121]	48.42	79.99	0.9	2.4	0.131
CS [120]	50	80	0.9	2.321675	0.0130747

Fig. 15 Cantilever beam problem

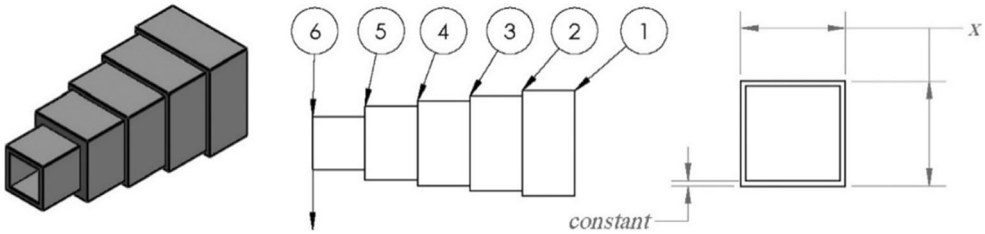


Table 12 Comparison of results on cantilever beam design problem

Algorithm	x_1	x_2	x_3	x_4	x_5	Optimum weight
OOSSA	5.7987206	5.2887823	4.5313474	3.2840148	2.1347847	1.3127493670
SSA [62]	6.0151345	5.3093046	4.4950067	3.5014262	2.1527879	1.33995639
MFO [115]	5.9848717	5.3167269	4.4973325	3.5136164	2.1616202	1.339988085
MVO [46]	6.0239402	5.30601123	4.49501132	3.49602232	2.15272617	1.3399595
SOS [123]	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996
CS [124]	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999
SaISOS [125]	5.929337	5.314156	4.449033	3.473583	2.1616463	1.33515
ALO [126]	6.01812	5.31142	4.48836	3.49751	2.158329	1.33995
MMA [127]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA-I [127]	6.0100	5.30400	4.4900	3.4980	2.1500	1.3400
GCA-II [127]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GOA [128]	6.011674	5.31297	4.48307	3.50279	2.16333	1.33996

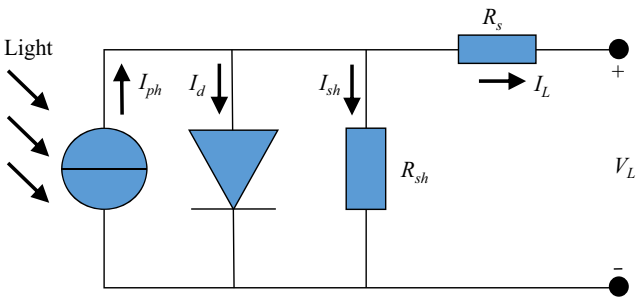


Fig. 16 Equivalent circuit of SDM

$$\eta = \sum_{k=1}^{nobs} \sum_{j=1}^{m} \max\left(1 - \frac{d_{j,k}}{robs_k}, 0\right) \tag{31}$$

where *nobs* represents the number of obstacles in the workspace, m denotes the number of interpolant points in the route, $d_{j,k}$ indicates the distance from the jth interpolant point to the center of the kth obstacle, and $robs_k$ is the radius of the kth obstacle. From Eq. (31), if the generated path does not contain any conflicts, a small value of η is obtained, and vice versa.

Table 13. Comparison results among different algorithms on SDM

Algorithm	I_{ph}(A)	I_{sd}(μA)	R_s(Ω)	R_{sh}(Ω)	n	RMSE
BSA [139]	0.7609	0.37749	0.0358	56.5266	1.4970	1.0398E-03
LBSA [140]	0.7609	0.34618	0.0362	59.0978	1.4881	1.0143E-03
IBSA [141]	0.7607	0.35502	0.0361	58.2012	1.4907	1.0092E-03
BLPSO [140]	0.7607	0.36620	0.0359	60.2845	1.4939	1.0272E-03
HISA [142]	1.0323	2.67736	1.2317	74.8451	47.658	2.0166E-03
PSO-WOA [143]	0.7606	0.34016	0.0361	59.3231	1.4864	1.0710E-03
mGWO [144]	0.7606	0.38534	0.0357	64.6624	1.4991	1.1279E-03
IGWO [144]	0.7627	0.23878	0.0365	30.5388	1.4521	2.3038E-03
WOA [49]	0.7599	0.41702	0.0351	74.4021	1.5072	1.3900E-03
SSA [62]	0.7775	0.14269	0.0068	55.6980	1.9957	1.8300E-02
OOSSA	0.7608	0.33809	0.0361	53.2795	1.4858	9.9966E-04

6.2 Experiment and results

In this subsection, the proposed OOSSA-based MRPP approach is fulfilled and the obstacle-free shortest route for a path planning problem in the AMRs is successfully mimicked on the Matlab 2014b platform. To validate the performance of the suggested methodology more authoritatively, five terrains provided by [147] were utilized for the simulation, and these environment setups have various characteristics and complexity levels that can help to test the method comprehensively. After investigating the suitability of the OOSSA method in MRPP issues, the routes gained by it are compared with those achieved by several classical swarm-based metaheuristic techniques such as PSO, GWO, FA, SSA, and ABC. For an

Table 14 Type of environment

Terrain	No. of obstacles	Initial coordinates	Final coordinates	X axis	Y axis	Obstacle radius
Map 1	3	0, 0	4, 6	[1 1.8 4.5]	[1 5.0 0.9]	[0.8 1.5 1]
Map 2	6	0, 0	10, 10	[1.5 8.5 3.2 6.0 1.2 7.0]	[4.5 6.5 2.5 3.5 1.5 8.0]	[1.5 0.9 0.4 0.6 0.8 0.6]
Map 3	13	3, 3	14, 14	[1.5 4.0 1.2 5.2 9.5 6.5 10.8 5.9 3.4 8.6 11.6 3.3 11.8]	[4.5 3.0 1.5 3.7 10.3 7.3 6.3 9.9 5.6 8.2 8.6 11.5 11.5]	[0.5 0.4 0.4 0.8 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7]
Map 4	30	3, 3	14, 14	[10.1 10.6 11.1 11.6 12.1 11.2 11.7 12.2 12.7 13.2 11.4 11.9 12.4 12.9 13.4 8 8.5 9 9.5 10 9.3 9.8 10.3 10.8 11.3 5.9 6.4 6.9 7.4 7.9]	[8.8 8.8 8.8 8.8 8.8 11.7 11.7 11.7 11.7 11.7 9.3 9.3 9.3 9.3 9.3 5.3 5.3 5.3 5.3 5.3 6.7 6.7 6.7 6.7 6.7 8.4 8.4 8.4 8.4 8.4]	[0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4]
Map 5	45	0, 0	15, 15	[2 2 2 2 2 2 4 4 4 4 4 4 4 4 4 6 6 6 8 8 8 8 8 8 8 8 8 10 10 10 10 10 10 10 10 10 12 12 12 12 12 14 14 14 14]	[8 8.5 9 9.5 10 10.5 3 3.5 4 4.5 5 5.5 6 6.5 7 11 11.5 12 1 1.5 2 2.5 3 3.4 4 4.5 5 6 6.5 7 7.5 8 8.5 9 9.5 10 10 10.5 11 11.5 12 10 10.5 11 11.5]	[0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4]

Table 15 The minimum path length comparison of OOSSA-based MRPP approach and competitors under five terrains

Terrain	PSO	FA	ABC	GWO	SSA	OOSSA
	Path length	Path length	Path length	Path length	Path length	Path length
Map 1	7.7328	7.5281	7.7527	7.7625	8.6017	**7.4573**
Map 2	14.5037	14.4288	14.8418	14.3565	15.2430	**14.3082**
Map 3	17.2219	16.1593	17.4818	17.3517	16.9979	**15.8140**
Map 4	16.1925	16.3031	16.2919	16.3281	16.8157	**15.8613**
Map 5	21.8520	21.9933	21.8792	22.3073	22.1842	**21. 0046**

unbiased comparison, the common parameters are set to the same values and special ones are kept in line with their original literature to help the involved approached can perform at their best. The details of the used terrains are illustrated in Table 14. Each algorithm plans the path for the robot ten times in each environmental map and we record the optimal trajectory. The length of the path produced by respective methods are reported in Table 15, and the corresponding trajectory is drawn in Figs. 18, 19, 20, 21 and 22.

From the optimal trace created by all algorithms in five terrains, as recorded in Table 14, the OOSSA-based MRPP approach planned the collision free shortest paths for all environmental setup compared to the competitors. Next, the optimal routes produced by all methods for each terrain are detailed below.

A comparison of optimal routes developed by all methods for the first terrain is illustrated in Fig. 18(a-f). From the figure, FA, SSA, and OOSSA design the same trajectory to navigate the AMRs from the starting point to the destination, while PSO, ABC, and GWO provide an alternative trajectory route. Clearly, the former are thinking more wisely. The comparison between FA, SSA, and OOSSA shows that the suggested method has the competence to circumvents local optimum, while the other two approaches try to give the best path for the AMRs, but they fall into the local optimal. In general, OOSSA-based MRPP approach provides competitive paths under simple terrain.Figure 19(a-f) depicts the optimal performance of all involved methods for the second terrain. From Fig. (a-e), the involved competitors can produce a threatening area-free safety path to help the AMRs move from the initial to the goal. However, getting stuck in local optimum leads to path redundancy. In contrast, the OOSSA-based MRPP approach builds a most encouraging route, which helps the AMRs to reach the destination safely while consuming less fuel. The comparison results between the OOSSA-based MRPP algorithm and its peers show that the developed approach can serve as an excellent tool for MRPP.

Figure 20(a-f) visualizes the optimal routes planned by all methods for the third landscape with 13 threatening regions of different sizes. From the figure, various trajectories are established while manoeuvring from the starting point to a destination to create an obstacle-free trial. Different from the tortuous paths generated by PSO, FA, GWO, and SSA, OOSSA and ABC forge straightforward trajectories, which can apparently contribute better to the robot's fuel savings. Furthermore, compared to OOSSA, ABC constructs a route trajectory that seems less sensible, as it is clearly worse than OOSSA in terms of path length.

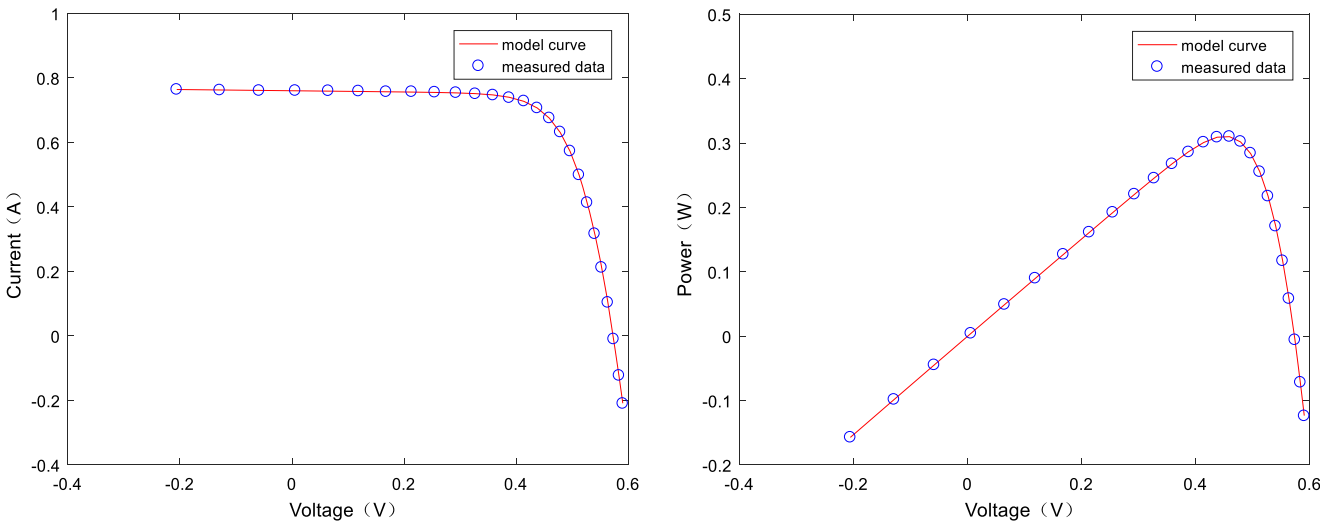


Fig. 17 Comparison between measured data and simulated data obtained by OOSSA for SDM

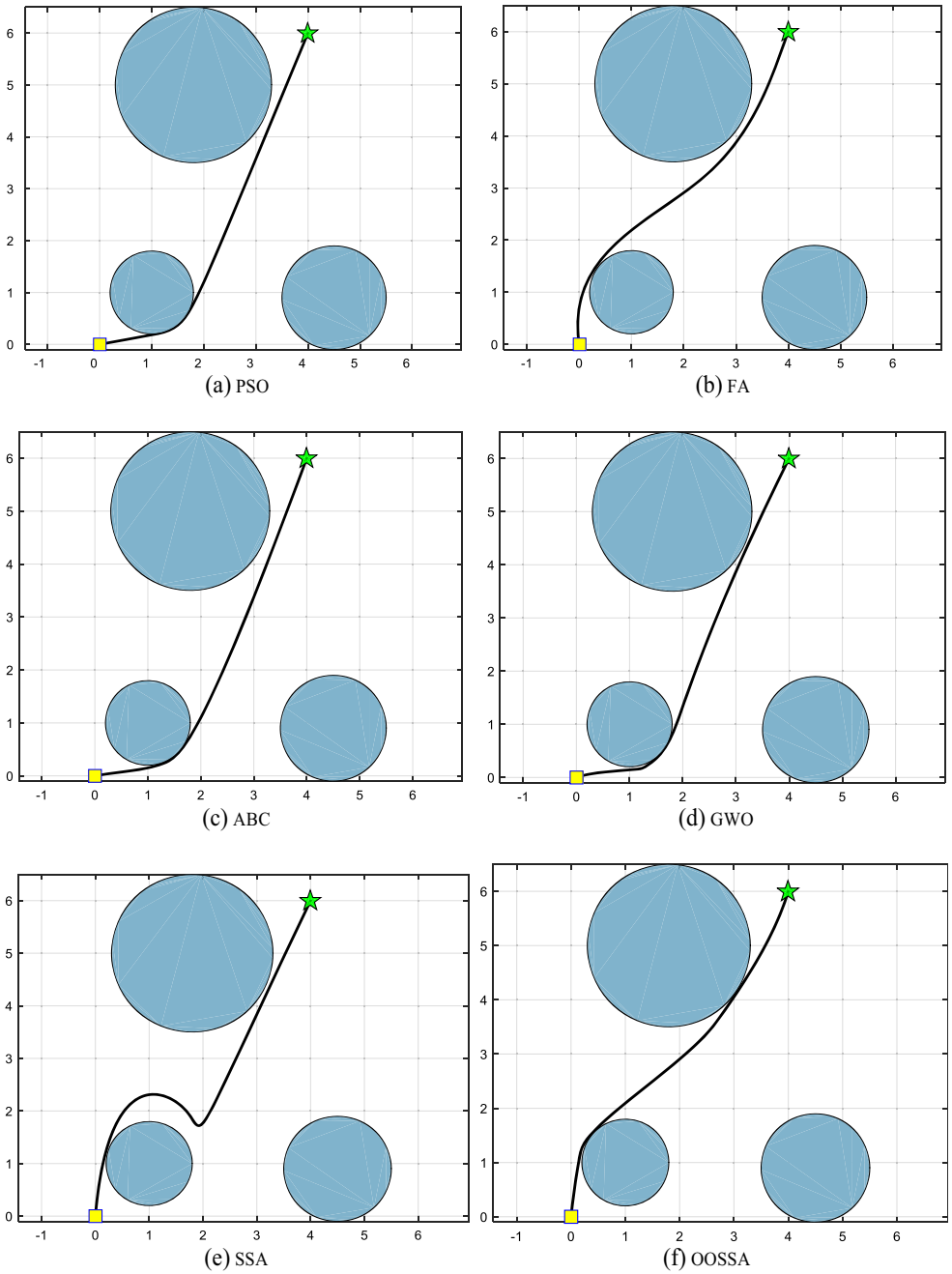


Fig. 18 Map 1 (**a**) PSO, (**b**) FA, (**c**) ABC, (**d**) GWO, (**e**) SSA and (**f**) OBDSSA

Correspondingly, the path lengths counted in Table 15 confirm this conclusion.

The simulation results of the fourth scenario with 30 obstacles shows that two different trajectories are established by applied methods while maoeuvring from the starting location to an ending point, as shown in Fig. 21(a-f). On the one hand, PSO, ABC, GWO, and SSA devises similar tracks, but travelling from the terrain edge is not conductive to assisting the AMRs in saving fuel, although avoiding collision with threatening regions. On the contrary, better routes are built by OOSSA and FA, and two algorithms have the competence to build paths that pass between obstacles, which validates the superiority of the methods. Moreover, compared to the sinuous route taken by FA, OOSSA yields a straightforward trajectory while manoeuvring from the starting point to destination. Overall, OOSSA is recommended as the optimal path planner under the complex terrain compared to its competitors.

Optimal paths generated by six applied approaches for the fifth terrain are plotted in Fig. 22(a-f). From the figure, we can conclude that all approaches can be recognized as reliable path planner. While the involved competitors have achieved

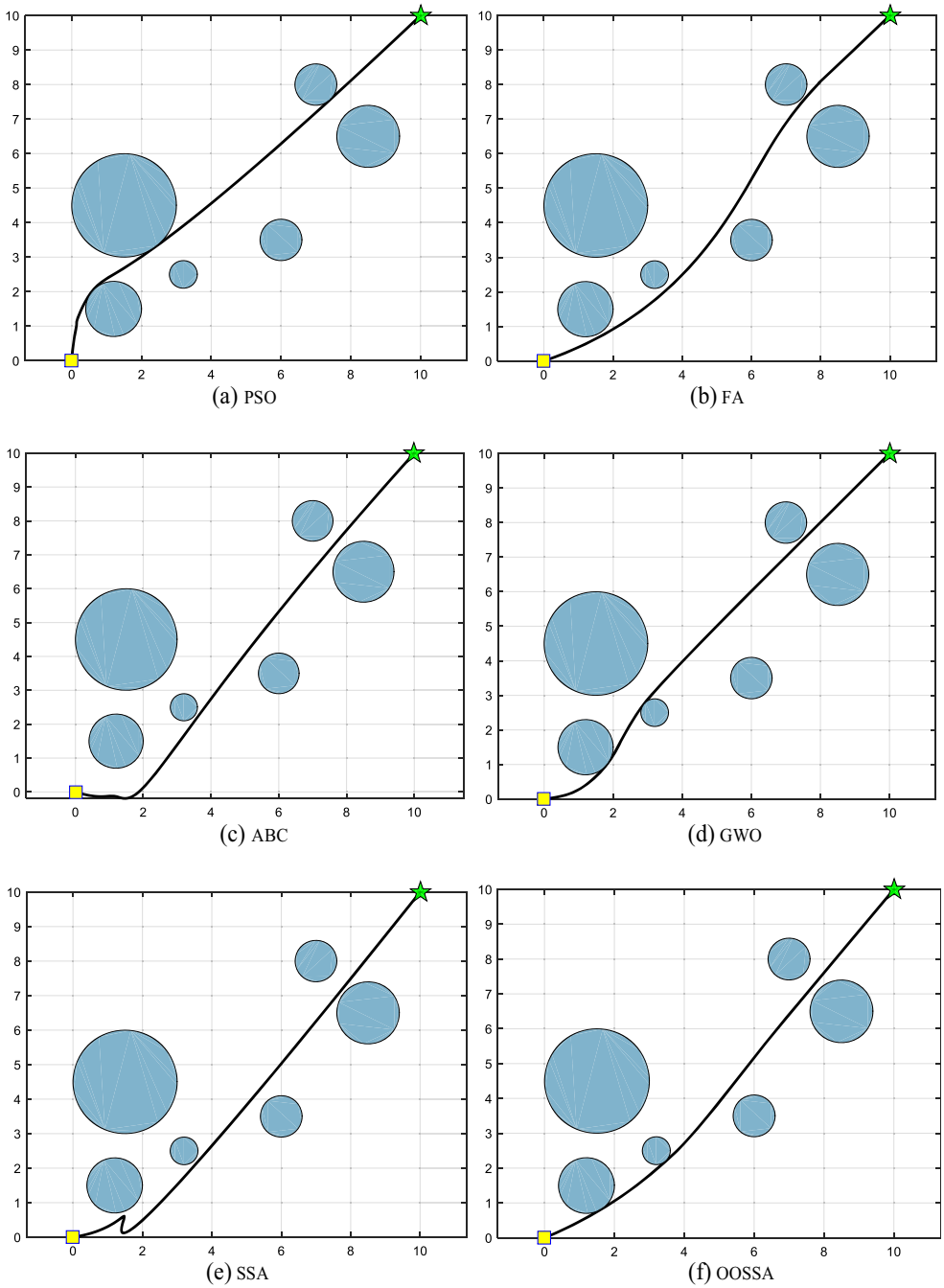


Fig. 19 Map 2 (**a**) PSO, (**b**) FA, (**c**) ABC, (**d**) GWO, (**e**) SSA and (**f**) OBDSSA

satisfactory results, there is still scope for improvement. The statistical results listed in Table 15 also show that the OOSSA-based MRPP approach received the shortest collision-free path, followed by PSO, ABC, FA, SSA, and GWO. The OOSSA method can handle the delicate balance between exploration and exploitation, thus jumping out of the local optimum solution and planning a directly straightforward path to navigate the AMRs from the beginning to the finish. Overall, the superior results allow the recommended approach to be a promising tool for MRPP problems in AMRs.

7 Conclusions

In this study, the limitations of canonical SSA are discussed in detail, including the unreasonable distribution of the number of leaders and followers, the monotonous follower position update mechanism, and the lack of a technique to help the algorithm jump out of the local optimal, which leads to SSA, like existing metaheuristic algorithms, being easy to fall into local optimal, lazy convergence, and unbalanced exploration and exploitation. To relieve these drawbacks in a new manner, a novel version of SSA is developed, called OOSSA,

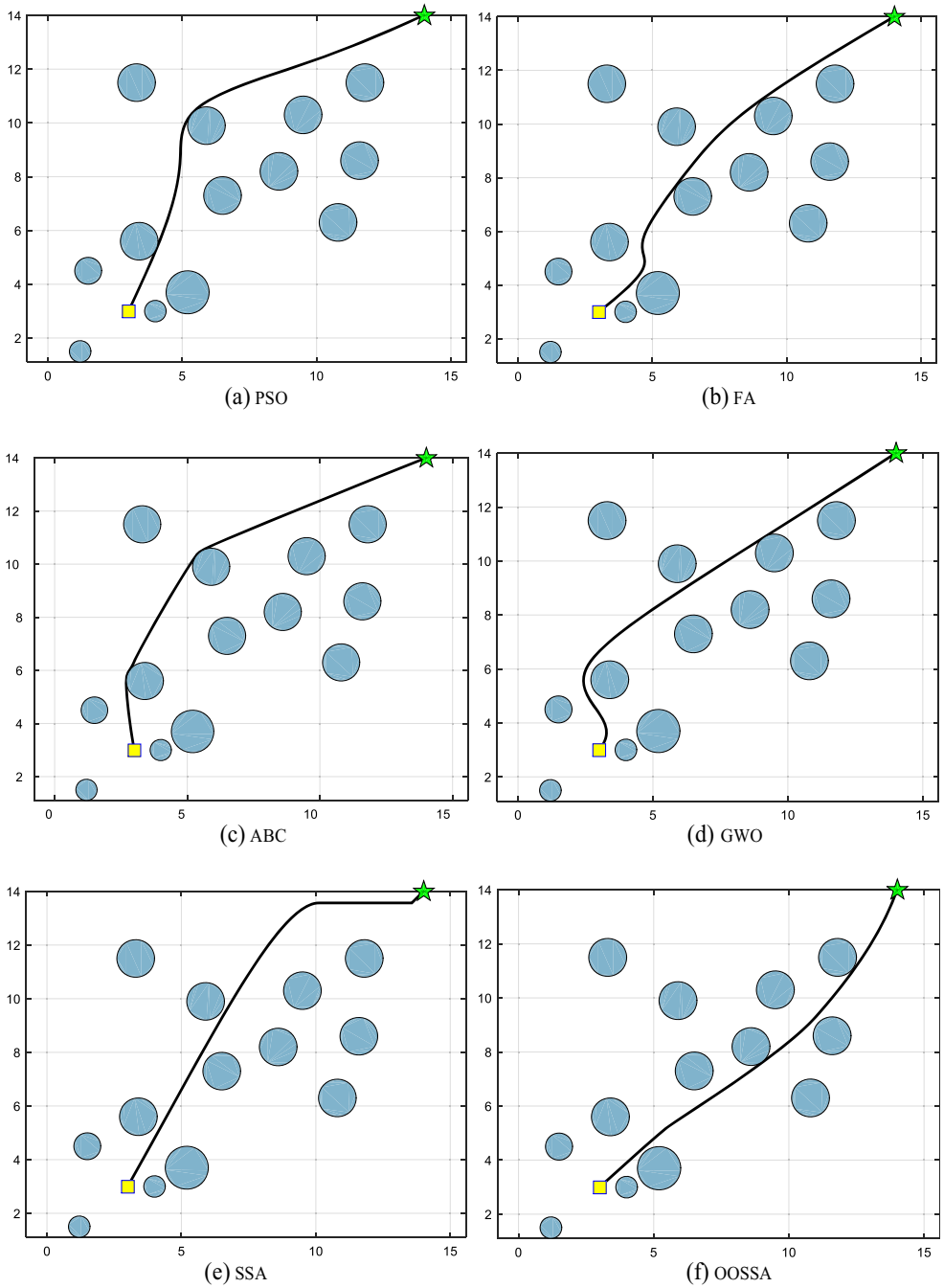


Fig. 20 Map 3 (**a**) PSO, (**b**) FA, (**c**) ABC, (**d**) GWO, (**e**) SSA and (**f**) OBDSSA

which introduces three reliable adjustment strategies into the basic SSA. First, a leader-follower number adaptive mechanism is presented to improve the method's global search ability in the early iterations and local exploitation competence in the later iterations. Second, an enhanced local optimal avoidance ability is obtained by introducing a lens opposition-based learning operator. In addition, an OLOBL strategy is constructed by combining LOBL and OED and embedded in the basic SSA to improve the exploratory ability of the algorithm while handling the dimensional degradation problem posed by OBL. To strike a trade-off between boosting the exploration potential of the method and reducing the number of FEs, only one leader is selected to perform OL OBL in each evolutionary iteration. Finally, a ranking-based dynamic learning strategy is introduced in the follower position update phase, which effectively improves the local exploitation capability of the algorithm.

The performance of the developed OOSSA is verified in a meaningful way on a set of 26 test functions widely used in the literature with various characteristics. The proposed OOSSA

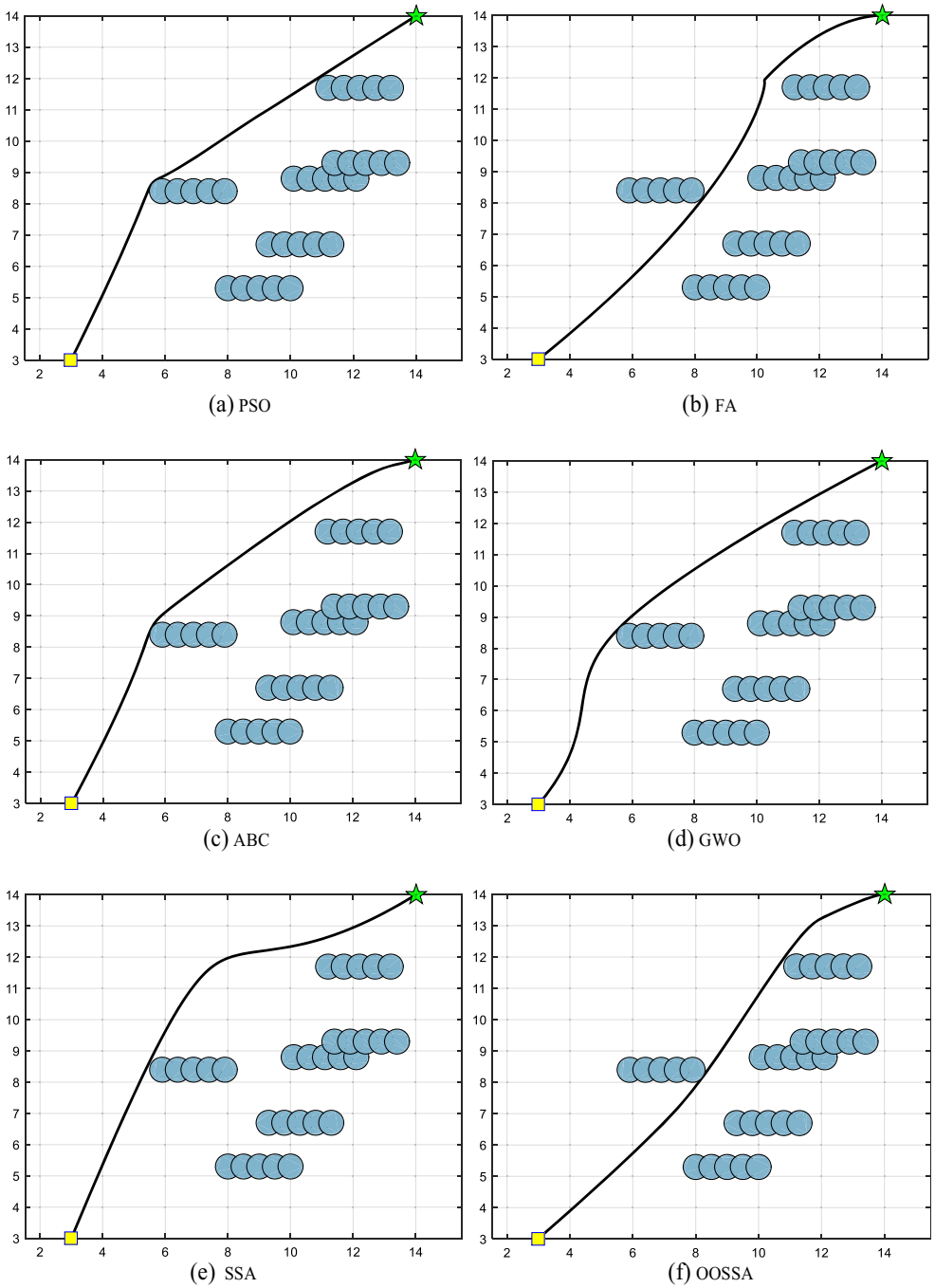


Fig. 21 Map 4 (**a**) PSO, (**b**) FA, (**c**) ABC, (**d**) GWO, (**e**) SSA and (**f**) OBDSSA

method is also compared with a comprehensive set of 13 well-performance SSA variants and 12 cutting-edge swarm-based metaheuristic approaches. The comparison results show that the proposed methodology outperforms other peers in a significant way. Also, OOSSA exhibits competitive performance on practical engineering design problems, including pressure vessel, I-beam, and cantilever beam. Additionally, the suggested approach is successfully applied to estimate the parameters of PV model. The test results indicate that OOSSA can serve as a promising tool for solar PV parameter estimation.

Finally, an OOSSA-based path planning and collision avoidance approach for autonomous mobile robots is presented. The performance of the introduced path planning approach is tested on five environmental maps and the outcomes achieved by this method are compared with those produced from other nature-inspired swarm-based techniques, including PSO, GWO, ABC, FA, and SSA. The comparative study shows that the introduced OOSSA-based path planning approach can provide the shortest collision-free route among all competitors under all the environmental setups.

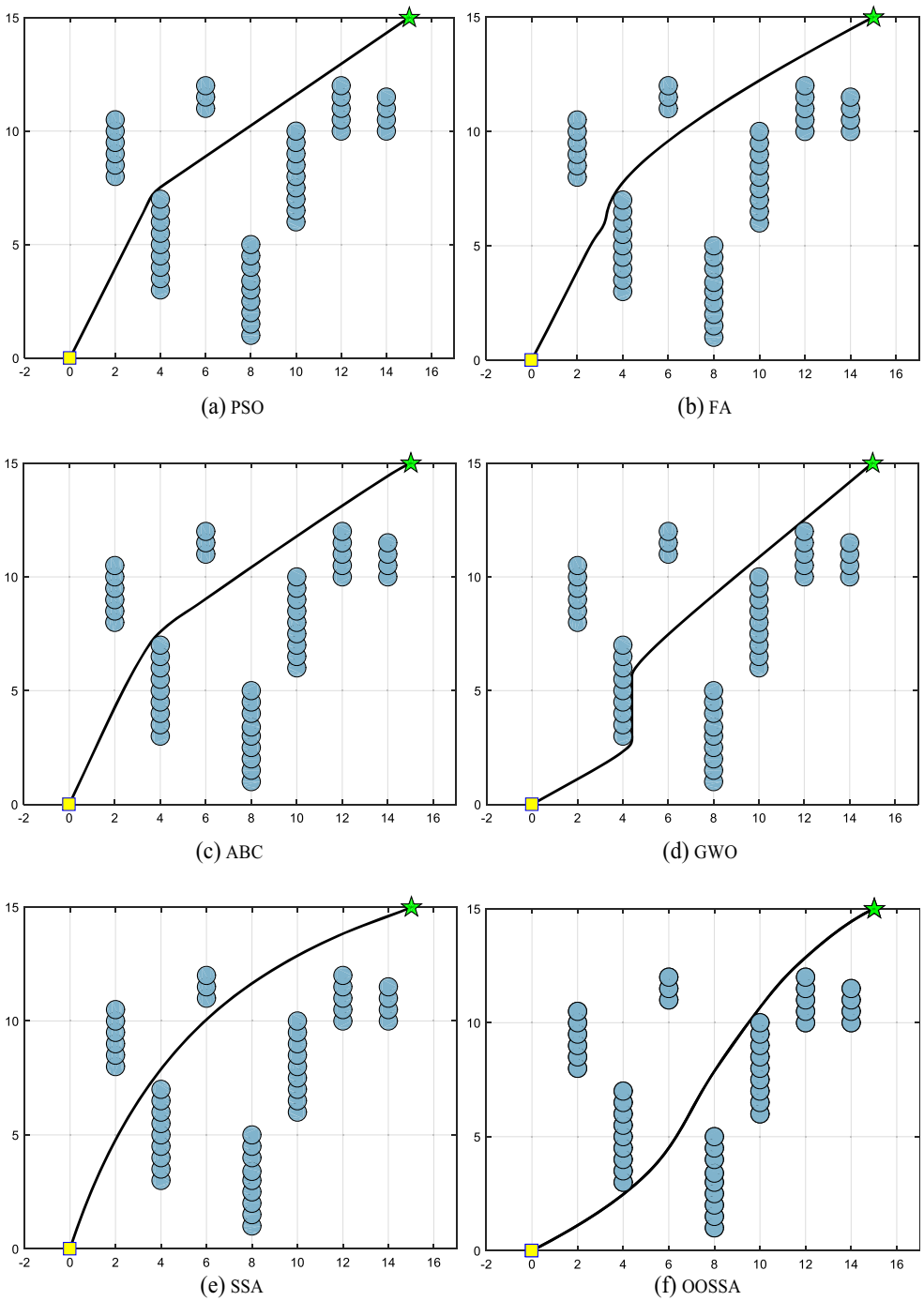


Fig. 22 Map 5 (**a**) PSO, (**b**) FA, (**c**) ABC, (**d**) GWO, (**e**) SSA and (**f**) OBDSSA

Overall, the developed strategies are significant to the comprehensive performance of metaheuristic techniques. Research teams interested in metaheuristic algorithms can integrate our strategies into the used nature-inspired swarm-based techniques to solve their optimization problems, and it is recommended that research groups concerned with solving their optimization tasks using nature-inspired swarm-based techniques use OOSSA for this purpose. In the future, we hope to generalize OOSSA and design effective constraint handling techniques to solve multi-objective optimization problems. To perform well on multi-objective optimization assignments, further balancing the diversity and convergence of OOSSA is a necessary task. Furthermore, we will also focus on measuring the effectiveness of the developed operators on other metaheuristic techniques.

Acknowledgements This research is supported by the National Natural Science Foundation of China (Grant Numbers 61461053, 61461054, and 61072079), Yunnan University of the China Postgraduate Science Foundation (Grant Number 2020306).

References

1. Boubezoul A, Paris S (2012) Application of global optimization methods to model and feature selection. Pattern Recogn 45(10): 3676–3686
2. Sebastian N, Suvrit S, Wright SJ (2011) Optimization for machine learning. The MIT Press, Cambridge
3. Pasandideh SHR, Niaki STA, Gharaei A (2015) Optimization of a multiproduct economic production quantity problem with stochastic constraints using sequential quadratic programming 84:98–107
4. Mohammed EA, Mohamed AAA, Mitani Y (2019)Genetic-moth swarm algorithm for optimal placement and capacity of renewable DG sources in distribution systems[J]. Int J Interact Multim Artif Intell 5(7):105–117
5. Rezk H, Arfaoui J, Gomaa MR (2021) Optimal Parameter Estimation of Solar PV Panel Based on Hybrid Particle Swarm and Grey Wolf Optimization Algorithms[J]. Int J Interact Multimed Artificial Intell 6:6
6. Yang X, Gong W (2021)Opposition-based JAYA with population reduction for parameter estimation of photovoltaic solar cells and modules[J]. Appl Soft Comput 104:107218
7. Abd Elaziz M, Thanikanti SB, Ibrahim IA, Lu S, Nastasi B, Alotaibi MA, Hossain MA, Yousri D (2021) Enhanced marine predators algorithm for identifying static and dynamic photovoltaic models parameters[J]. Energy Convers Manag 236:113971
8. Wang J, Yang B, Li D, Zeng C, Chen Y, Guo Z, Zhang X, Tan T, Shu H, Yu T (2021) Photovoltaic cell parameter estimation based on improved equilibrium optimizer algorithm[J]. Energy Convers Manag 236:114051
9. Messaoud RB (2020) Extraction of uncertain parameters of single and double diode model of a photovoltaic panel using Salp swarm algorithm[J]. Measurement 154:107446
10. Gao S, Wang K, Tao S, Jin T, Dai H, Cheng J (2021) A state-of-the-art differential evolution algorithm for parameter estimation of solar photovoltaic models[J]. Energy Convers Manag 230:113784
11. Wang Z, Ding H, Li B, Bao L, Yang Z (2020) An energy efficient routing protocol based on improved artificial bee colony algorithm for wireless sensor networks[J]. IEEE Access 8:133577–133596
12. Hussain K, Neggaz N, Zhu W, Houssein EH (2021) An efficient hybrid sine-cosine Harris hawks optimization for low and high-dimensional feature selection[J]. Expert Syst Appl 176:114778
13. Dhal P, Azad C (2021) A multi-objective feature selection method using Newton's law based PSO with GWO[J]. Appl Soft Comput 107:107394
14. Li AD, Xue B, Zhang M (2021) Improved binary particle swarm optimization for feature selection with new initialization and search space reduction strategies[J]. Appl Soft Comput 106: 107302
15. Pan JS, Tian AQ, Chu SC, et al. (2021) Improved binary pigeon-inspired optimization and its application for feature selection[J]. Appl Intell: 1–19
16. Hammouri AI, Mafarja M, Al-Betar MA et al (2020) An improved dragonfly algorithm for feature selection[J]. Knowl-Based Syst 203:106131
17. Yousri D, Abd Elaziz M, Abualigah L, Oliva D, al-qaness MAA, Ewees AA (2021)COVID-19 X-ray images classification based on enhanced fractional-order cuckoo search optimizer using heavy-tailed distributions[J]. Appl Soft Comput 101:107052
18. Soui M, Mansouri N, Alhamad R, et al. (2021)NSGA-II as feature selection technique and AdaBoost classifier for COVID-19 prediction using patient's symptoms[J]. Nonlinear Dynamics 1–23
19. Too J, Mirjalili S (2021) A hyper learning binary dragonfly algorithm for feature selection: a COVID-19 case study[J]. Knowl-Based Syst 212:106553
20. Sahlol AT, Yousri D, Ewees AA et al (2020)COVID-19 image classification using deep features and fractional-order marine predators algorithm[J]. Sci Rep 10(1):1–15
21. Canayaz M (2021) MH-COVIDNet: diagnosis of COVID-19 using deep neural networks and meta-heuristic-based feature selection on X-ray images[J]. Biomed Signal Process Control 64: 102257
22. Zhang L, Zhang Y, Li Y (2020) Mobile robot path planning based on improved localized particle swarm optimization[J]. IEEE Sensors J 21(5):6962–6972
23. Liang JH, Lee CH (2015) Efficient collision-free path-planning of multiple mobile robots system using efficient artificial bee colony algorithm[J]. Adv Eng Softw 79:47–56
24. Islam MR, Protik P, Das S, Boni PK (2021) Mobile robot path planning with obstacle avoidance using chemical reaction optimization[J]. Soft Comput 25(8):6283–6310
25. Liu L, Luo S, Guo F, Tan S (2020)Multi-point shortest path planning based on an improved discrete bat algorithm[J]. Appl Soft Comput 95:106498
26. Li X, Wang L (2020) Application of improved ant colony optimization in mobile robot trajectory planning[J]. Math Biosci Eng 17(6):6756–6774
27. Mohanty PK, Parhi DR (2016) Optimal path planning for a mobile robot using cuckoo search algorithm[J]. J Exp Theoretical Artificial Intell 28(1–2):35–52
28. Houssein EH, Helmy BE, Oliva D, Elngar AA, Shaban H (2021) A novel black widow optimization algorithm for multilevel thresholding image segmentation[J]. Expert Syst Appl 167: 114159
29. Li LL, Liu ZF, Tseng ML, Zheng SJ, Lim MK (2021) Improved tunicate swarm algorithm: solving the dynamic economic emission dispatch problems[J]. Appl Soft Comput 108:107504
30. Ekinci S, Hekimoğlu B, Izci D (2021) Opposition based Henry gas solubility optimization as a novel algorithm for PID control of DC motor[J]. Eng Sci Technol Int J 24(2):331–342
31. Abed-alguni BH, Alawad NA (2021) Distributed Grey Wolf optimizer for scheduling of workflow applications in cloud environments[J]. Appl Soft Comput 102:107113
32. Venkata Rao R, Singh KH (2018)Multi-team perturbation guiding jaya algorithm for optimization of wind farm layout. Appl Soft Comput 71:800–815
33. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization[J]. IEEE Trans Evol Comput 1(1):67–82
34. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the 1995 IEEE international conference on neural networks. p 1942–8
35. Yang SX (2010) Firefly algorithm, stochastic test functions and design optimization. Int J Bio-Inspired Comput 2(2):78–84
36. D. Karaboga (2005) An Idea Based on Honey Bee Swarm for Numerical Optimization, Erciyes Univ., Kayseri, Turkey, Tech. Rep.-TR06
37. Gandomi AH, Alavi AH (2012) Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17(12):4831–4845
38. Yang X, Gandomi AH (2012) Bat algorithm: a novel approach for global engineering optimization. Eng Comput 29(5):464–483
39. Pierezan J, Coelho LDS (2018) Coyote optimization algorithm: a new metaheuristic for global optimization problems. In: 2018 IEEE congress on evolutionary computation (CEC), pp 1–8
40. Sharma V, Pattnaik SS, Garg T (2012) A review of bacterial foraging optimization and its applications. Procedia - Social Behav Sci 48(1):1294–1303
41. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69(3):46–61

42. Pan WT (2012) A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowl-Based Syst 26(2): 69–74
43. Dhiman G, Kumar V (2017) Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 114:48–70
44. Moosavi SHS, Bardsiri VK (2019) Poor and rich optimization algorithm: a new human-based and multi populations algorithm. Eng Appl Artif Intell 86:165–181
45. Sulaiman MH, Mustaffa Z, Saari MM, Daniyal H (2020) Barnacles mating optimizer: a new bio-inspired algorithm for solving engineering optimization problems. Eng Appl Artif Intell 87:103330
46. Mirjalili S, Mirjalili SM, Hatamlou A (2016)Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Applic 27(2):495–513
47. Li X, Zhang J, Yin M (2014) Animal migration optimization: an optimization algorithm inspired by animal migration behavior. Neural Comput Appl 24(7–8):1867–1877. https://doi.org/10.1007/s00521-013-1433-8
48. Mirjalili S (2016) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073. https://doi.org/10.1007/s00521-015-1920-1
49. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
50. Yang XS (2012) Flower pollination algorithm for global optimization. In: Unconventional computation and natural computation. Springer, pp. 240–249
51. Muthiah-Nakarajan V, Noel MM (2016) Galactic swarm optimization: a new global optimization metaheuristic inspired by galactic motion. Appl Soft Comput 38:771–787. https://doi.org/10.1016/j.asoc.2015.10.034
52. Klein CE, Coelho LDS (2018)Meerkats-inspired algorithm for global optimization problems. The European symposium on artificial neural networks. https://www.elen. ucl.ac.be/proceedings/esann/esannpdf/es2018-35.Pdf
53. Shayanfar H, Gharehchopogh FS (2018) Farmland fertility: a new metaheuristic algorithm for solving continuous optimization problems. Appl Soft Comput 71:728–746. https://doi.org/10.1016/j.asoc.2018.07.033
54. Yapici H, Cetinkaya N (2019) A new meta-heuristic optimizer: pathfinder algorithm. Appl Soft Comput 78:545–568. https://doi.org/10.1016/j.asoc.2019.03.012
55. de Vasconcelos Segundo EH, Mariani VC, dos Santos Coelho L (2019a) Design of heat exchangers using falcon optimization algorithm. Appl Therm Eng 156:119–144. https://doi.org/10.1016/j.applthermaleng.2019.04.038
56. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Futur Gener Comput Syst 97:849–872. https://doi.org/10.1016/j.future.2019.02.028
57. de Vasconcelos Segundo EH, Mariani VC, dos Santos Coelho L (2019b) Metaheuristic inspired on owls behavior applied to heat exchangers design. Thermal Sci Eng Progress 14:100431. https://doi.org/10.1016/j.tsep.2019.100431
58. Zhao W, Zhang Z, Wang L (2020) Manta ray foraging optimization: an effective bio-inspired optimizer for engineering applications. Eng Appl Artificial Intell 87:103300. https://doi.org/10.1016/j.engappai.2019.103300
59. Wang G-G, Deb S, Zhi-Hua C (2019) Monarch butterfly optimization. Neural Comput Appl[J] 31(7):1–20
60. Kallioras NA, Lagaros ND, Avtzis DN (2018) Pity beetle algorithm - a new metaheuristic inspired by the behavior of bark beetles[J]. Adv Eng Softw 121:147–166
61. Wang G-G, Suash D, Santos CLD (2018) Earthworm optimization algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Int J Bio-Inspired Comput[J] 12(1):1–22
62. MirjaliliS GAH, Mirjalili SZ et al (2017) Salp swarm algorithm: a bioinspired optimizer for engineering design problems[J]. Adv Eng Softw 114(6):163–191
63. Abbassi R, Abbassi A, Heidari AA, Mirjalili S (2019) An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models. Energy Convers Manag 179:362–372
64. Abbassi A, Abbassi R, Heidari AA, Oliva D, Chen H, Habib A, Jemli M, Wang M (2020) Parameters identification of photovoltaic cell models using enhanced exploratory salp chains-based approach. Energy 198:117333
65. Ateya AA, Muthanna A, Vybornova A, Algarni AD, Abuarqoub A, Koucheryavy Y, Koucheryavy A (2019) Chaotic salp swarm algorithm for SDN multi-controller networks. Eng Sci Technol Int J 22(4):1001–1012
66. Airathi D, Gopalani D (2019) Salp swarm algorithm (SSA) for training feed-forward neural networks. In: Bansal J, Das K, Nagar A, Deep K, Ojha A (eds) Soft computing for problem solving. Springer, Berlin, pp 521–534
67. Abusnaina AA, Ahmad S , Jarrar R, Mafarja M (2018) Training neural networks using salp swarm algorithm for pattern classification, in: Proceedings of the 2nd International Conference on Future Networks and Distributed Systems, ACM, p. 17
68. Zhang J, Wang Z, Luo X (2018) Parameter estimation for soil water retention curve using the salp swarm algorithm. Water 10(6):815
69. Khalid A, Khan ZA, Javaid N (2018) Game theory based electric price tariffand salp swarm algorithm for demand side management, in: 2018 Fifth HCT Information Technology Trends (ITT), IEEE, pp 99–103
70. Ibrahim A, Ahmed A, Hussein S, Hassanien AE (2018) Fish image segmentation using salp swarm algorithm. In: International Conference on advanced machine learning technologies and applications. Springer, Cham, pp 42–51
71. Xing Z, Jia H (2019) Multilevel color image segmentation based on GLCM and improved salp swarm algorithm. IEEE Access 7: 37672–37690
72. Liu X, Xu H (2018) Application on target localization based on salp swarm algorithm. In 2018 37th Chinese control conference (CCC). IEEE, pp 4542–4545
73. El-Fergany AA, Hasanien HM (2019) Salp swarm optimizer to solve optimal power flow comprising voltage stability analysis. Neural Comput Appl, pp 1–17
74. El-Fergany AA (2018) Extracting optimal parameters of PEM fuel cells using salp swarm optimizer. Renew Energy 119:641–648
75. Ibrahim HT, Mazher WJ, Ucan ON, Bayat O (2017) Feature selection using salp swarm algorithm for real biomedical datasets. Int J Comput Sci Netw Secur 12:13
76. Khamees M, Albakry A, Shaker K (2018) A new approach for features selection based on binary slap swarm algorithm. J Theor Appl Inf Technol 96:1896–1906
77. Faris H, Mafarja MM, Heidari AA, Aljarah I, Al-Zoubi AM, Mirjalili S, Fujita H (2018) An efficient binary Salp swarm algorithm with crossover scheme for feature selection problems. Knowl-Based Syst 154:43–67
78. Zhao H, Huang G, Yan N (2018) Forecasting energy-related CO2 emissions employing a novel ssa-lssvm model: considering structural factors in China. Energies 11:781–801
79. Tubishat M , Ja'Afar S , Alswaitti M, et al. (2020) Dynamic Salp swarm algorithm for feature selection[J]. Expert Syst Appl, 113873
80. Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell 48(10):3462–3481

81. Qais MH, Hasanien HM, Alghuwainem S (2019) Enhanced salp swarm algorithm: application to variable speed wind generators[J]. Eng Appl Artif Intell 80:82–96
82. Gupta S, et al. (2019) Harmonized salp chain-built optimization, Eng Comput
83. Zhang H , Cai Z , Ye X, et al. (2020) A multi-strategy enhanced salp swarm algorithm for global optimization[J]. Eng Comput (1)
84. Ren H, Li J, Chen H, Li CY (2021) Adaptive levy-assisted salp swarm algorithm: analysis and optimization case studies[J]. Math Comput Simul 181:380–409
85. Gholami K, Parvaneh MH (2019) A mutated salp swarm algorithm for optimum allocation of active and reactive power sources in radial distribution systems[J]. Appl Soft Comput 85:105833
86. Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell 48(10):3462–3481
87. Wu J, Nan R, Chen L (2019) Improved salp swarm algorithm based on weight factor and adaptive mutation. J Exp Theoretic Artificial Intell 31(3):493–515. https://doi.org/10.1080/0952813X.2019.1572659
88. Ibrahim RA, Ewees AA, Oliva D, Elaziz MA, Lu S (2018) Improved salp swarm algorithm based on particle swarm optimization for feature selection. J ambient Intell Humaniz Comput, pp 1–15
89. Singh N, Chiclana F, Magnot JP (2019) A new fusion of salp swarm with sine cosine for optimization of non-linear functions. Eng Comput, pp 1–28
90. Mirjalili S (2016) Sca: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
91. Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179:2232–2248
92. Yang B, Zhong L, Zhang X, Shu H, Yu T, Li H, Sun L (2019) Novel bio-inspired memetic salp swarm algorithm and application to MPPT for PV systems considering partial shading condition. J Clean Prod
93. Syed MA, Syed R (2019) Weighted salp swarm algorithm and its applications towards optimal sensor deployment, J. King Saud Univ.-Comput. Inf. Sci., Weighted Salp Swarm Algorithm and its applications towards optimal sensor deployment
94. Bairathi D , Gopalani D (2018) Opposition based salp swarm algorithm for numerical optimization, in: International Conference on Intelligent Systems Design and Applications, Springer, pp. 821–831
95. Lei C, Yue L, Zhi-long K (2020) Improved salp swarm algorithm based on reduction factor and dynamic learning. Control Theory Appl 37(8):1766–1780
96. Born M, Wolf E (1959) Principles of optics. Pergamon Press, New York
97. Tizhoosh HR (2005)Opposition-based learning: a new scheme for machine intelligence[C]. International Conference on Computational Intelligence for Modelling, Control and Automation, Vienna, pp 695–701
98. Rahnamayan S, Tizhoosh HR, Salama M (2008) Opposition versus randomness in soft computing techniques[J]. Appl Soft Comput 8(2):906–918
99. Rahnamayan S, Tizhoosh HR, Salama MMA (2008)Opposition-based differential evolution[J]. IEEE Trans Evol Comput 12(1): 64–79
100. Zhang H, Heidari AA, Wang M, Zhang L, Chen H, Li C (2020) Orthogonal Nelder-Mead moth flame method for parameters identification of photovoltaic modules[J]. Energy Convers Manag 211: 112764
101. Zhihui ZHAN, Jun Z, Yun L et al (2011) Orthogonal learning particle swarm optimization[J]. IEEE Trans Evol Comput 15(6): 832–847. https://doi.org/10.1109/TEVC.2010.2052054
102. Yu F, Li YX, Wei B, Xu X, Zhao ZY (2014) The application of a novel OBL based on lens imaging principle in PSO. Tien Tzu Hsueh Pao/Acta electronica Sinica 42:230–235. https://doi.org/10.3969/j.issn.0372-2112.2014.02.004
103. Mahdavi S, Rahnamayan S, Deb K (2018) Opposition based learning: a literature review. Swarm Evol Comput 39:1–23
104. Park SY, Lee JJ (2016) stochastic opposition-based learning using a beta distribution in differential evolution[J]. IEEE Trans Cybernet 46(10):2184–2194. https://doi.org/10.1109/TCYB.2015.2469722
105. Braik M, Sheta A, Turabieh H et al (2020) A novel lifetime scheme for enhancing the convergence performance of salp swarm algorithm[J]. Soft Comput 25(1):181–206
106. Zhong-Yun C, Da-Min Z, Zi-Yun X (2021)Multi-subpopulation based symbiosis and non-uniform Gaussian mutation salp swarm algorithm. Acta Automatica Sinica, 1–9. https://doi.org/10.16383/j.aas.c190684
107. Salgotra R, Singh U, Singh S et al (2020)Self-adaptive Salp swarm algorithm for engineering optimization problems[J]. Appl Math Model 89:188–207
108. Tubishat M, Idris N, Shuib L et al (2019) Improved Salp swarm algorithm based on opposition based learning and novel local search algorithm for feature selection[J]. Expert Syst Appl 145: 113122
109. Nautiyal B, Prakash R, Vimal V, et al. (2021) Improved Salp swarm algorithm with mutation schemes for solving global optimization and engineering problems[J]. Eng Comput, 1–23
110. Hussien AG (2021) An enhanced opposition-based Salp swarm algorithm for global optimization and engineering problems[J]. J Ambient Intell Humaniz Comput 1–22
111. Ozbay FA, Alatas B (2021) Adaptive Salp swarm optimization algorithms with inertia weights for novel fake news detection model in online social media[J]. Multimed Tools Appl 1–25
112. Ren H, Li J, Chen H, Li CY (2021) Stability of salp swarm algorithm with random replacement and double adaptive weighting[J]. Appl Math Model 95:503–523
113. Saafan MM, El-Gendy EM (2021) IWOSSA: an improved whale optimization salp swarm algorithm for solving optimization problems[J]. Expert Syst Appl 176:114901
114. DERRAC J, CARCIA S, MOLINA D et al (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms[J]. Swarm Evol Comput 1(11):3–18. https://doi.org/10.1016/j.swevo.2011.02.002
115. Mirjalili S (2015)Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm[J]. Knowledge-Based Syst 89(NOV):228–249
116. Faramarzi A, HeidarinejadM, Mirjalili S, et al (2020) Marine Predators Algorithm: A Nature-inspiredMetaheuristic[J]. Expert Systems with Applications 152:113377
117. Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S (2020) Equilibrium optimizer: anovel optimization algorithm[J]. Knowledge-Based Systems 191:105190
118. Chen H, Xu Y, Wang M, Zhao X (2019) A balanced whale optimization algorithm for constrained engineering design problems[J]. Appl Math Model 71:45–59
119. Cheng M-Y, Prayogo D (2014) Symbiotic organisms search: a new metaheuristic optimization algorithm[J]. Comput Struct 139:98–112
120. Wang GG (2003) Adaptive response surface method using inherited latin hypercube design points. J Mech Des 125:210–220
121. Gandomi AH, Yang X-S, Alavi AH (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 29:17–35

122. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search[J]. Acta Mech 213(3/4):267–289. https://doi.org/10.1007/s00707-009-0270-4
123. Cheng M-Y, Prayogo D (2014) Symbiotic organisms search: a new metaheuristic optimization algorithm[J]. Comput Struct 139:98–112
124. Gandomi AH, Yang XS, Alavi AH (2013) Cuckoo search algorithm: a metaheuristicapproach to solve structural optimization problems, Eng. Comput 29:17–35
125. Nama S,Saha AK, Sharma S (2020) A novel improved symbiotic organisms search algorithm.Computational Intelligence 1–31. https://doi.org/10.1111/coin.12290
126. MirjaliliS (2015) The ant lion optimizer[J]. Adv Eng Softw 83:80–98
127. Chickermane H, Gea H (1996) Structural optimization using a new local approximation method[J]. Int J NumerMeth Eng 39: 829–46
128. SaremiS, Mirjalili S, Lewis A (2017) Grasshopper optimisation algorithm: theory andapplication. Adv Eng Softw 105:30–47
129. Kaur S, Awasthi LK, Sangal AL, Dhiman G (2020) Tunicate SwarmAlgorithm: A new bio-inspired based metaheuristic paradigm for globaloptimization. Engineering Applications of Artificial Intelligence 90:103541
130. Dhargupta S, Ghosh M,Mirjalili S, Sarkar R (2020) Selective opposition based grey wolfoptimization. Expert Systems with Applications 151:113389
131. Nadimi-ShahrakiMH, Taghian S, Mirjalili S (2021) An improved grey wolf optimizer for solvingengineering problems[J]. Expert Systems with Applications 166:113917
132. HashimFA, Houssein EH, Mabrouk MS et al (2019) Henry gas solubility optimization: anovel physics-based algorithm. Futur Gener Comput Syst 101:646–667
133. Li C,Li J, Chen H, et al (2021) Memetic Harris Hawks Optimization: Developments andperspectives on project scheduling and QoS-aware web service composition [J].Expert Systems with Applications 171
134. LaithAbualigah, Ali Diabat, Seyedali Mirjalili, Mohamed Abd Elaziz, and Amir HGandomi (2020) The arithmetic optimization algorithm. Computer methods in appliedmechanics and engineering 376:113609
135. Hashim FA, Hussain K, Houssein EH, Mabrouk MS, Al-Atabany W(2020) Archimedes optimizationalgorithm: A new metaheuristic algorithm for solving optimization problems,Appl. Intell 51:1531–1551
136. Ma L,Wang C, Xie N, et al (2021)Moth-flame optimization algorithm based on diversity andmutation strategy[J]. Applied Intelligence 1–37
137. Shan W,Qiao Z, Heidari A A, et al (2021) Double adaptive weights for stabilization of mothflame optimizer: balance analysis, engineering cases, and medical diagnosis[J].Knowledge-Based Systems 214:106728
138. Yu X, Xu WY, Li CL (2021)Opposition-based learning grey wolf optimizer for global optimization[J]. Knowl-Based Syst 226: 107139
139. Civicioglu P (2013) Backtracking search optimization algorithm for number optimization problems. Appl Math Comput 219: 8121–8144
140. Chen D, Zou F, Lu R, Wang P (2017b) Learning backtracking search optimization algorithm and its application. Inf Sci 376:71–94
141. Nama S, Saha AK, Ghosh S (2017) Improved backtracking search algorithm for pseudo dynamic active earth pressure on retaining wall supporting c-f backfill. Appl Soft Comput 52:885–897
142. Kler D, Goswami Y, Rana KPS, Kumar V (2019) A novel approach to parameter estimation of photovoltaic systems using hybridized optimizer. Energy Convers Manag 187:486–511
143. Xiong G, Zhang J, Yuan X, Shi D, He Y, Yao G (2018) Parameter extraction of solar photovoltaic models by means of a hybrid differential evolution with whale optimization algorithm. Sol Energy 176:742–761
144. Long W, Cai S, Jiao J, Xu M, Wu T (2020) A new hybrid algorithm based on grey wolf optimizer and cuckoo search for parameter extraction of solar photovoltaic models. Energy Convers Manag 203:112243
145. Yu K, Liang J, Qu B,Cheng Z, Wang H (2018) Multiple learning backtracking search algorithm forestimating parameters of photovoltaic models. Appl. Energy 226:408–422
146. Easwarakhanthan T, Bottin J, Bouhouch I, Boutrit C (1986)Nonlinear minimization algorithm for determining the solar cell parameters with microcomputers. Int J Sol Energy 4:1–12
147. Agarwal D, Bharti PS (2021) Implementing modified swarm intelligence algorithm based on slime moulds for path planning and obstacle avoidance problem in mobile robots[J]. Appl Soft Comput 107:107372

Zongshan Wang received his M.Sc. degree in Communication and Information System from Yunnan University, China, in 2020. He is currently pursuing a Ph.D. with the School of Information Science and Engineering, Yunnan University. His principle research interests includes warm intelligence optimization algorithms, energy-efficient routing protocol design for wireless sensor networks based on swarm intelligence algorithms, edge computing, robot path planning, and system modeling.

Bo Li received his PhD. degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, China, in 2005. He is currently a Professor in the School of Information Science and Engineering, Yunnan University. He has published many research papers in international conferences and reputed journal in Springer. His main research interests include swarm intelligence optimization algorithms, wireless sensor networks, and edge computing.

Hongwei Ding received his PhD. degree from the School of Information Science and Engineering, Yunnan University, Kunming, China, in 2002. He is currently a Professor and Ph.D. supervisor in the School of Information Science and Engineering, Yunnan University. He has published more than 100 research papers inInternational/ National Conferences and Journals of good repute. His main research interests include wireless sensor networks, swarm intelligence optimization algorithms, communication and information systems, network and communication engineering, polling multiple access communication theory, and random multiple access communication systems.

Zheng Guan received her PhD. degree from the School of Information Science and Engineering, Yunnan University, Kunming, China, in 2012. She is currently an Associate Professor inthe School of Information Science and Engineering, Yunnan University. She has published many research papers ininternational conferences and reputed journal in Springer. Her main research interests include network performance optimization, polling systems and their applications.

Zhijun Yang received the Ph.D. degree from the School of Information Science and Engineering, Yunnan University, Kunming, China. He is currently a Senior Fellow at the Education Department of Yunnan Province. He has published more than 100 research papers in International/ National Conferences and Journals of good repute. His research interests include swarm intelligence optimization algorithms, wireless sensor networks, information systems, and polling multiple access communication systems.

Liyong Bao received his PhD. degree from the School of Information Science and Engineering, Yunnan University, Kunming, China, in 2011. He is currently an Associate Professor inthe School of Information Science and Engineering, Yunnan University. He has published many research papers in international conferences and reputed journal in Springer. His main research interests includes warm intelligence optimization algorithms, wireless sensor networks, information systems, and polling multiple access communication theory.