Applied Intelligence (2023) 53:11833–11860
https://doi.org/10.1007/s10489-022-03994-3

Sea-horse optimizer: a novel nature-inspired meta-heuristic for global optimization problems

Shijie Zhao[1,2] · Tianran Zhang[1] · Shilin Ma[1] · Mengchen Wang[1]

Accepted: 12 July 2022 / Published online: 13 September 2022

Abstract

This paper proposes a novel swarm intelligence-based metaheuristic called as sea-horse optimizer (SHO), which is inspired by the movement, predation and breeding behaviors of sea horses in nature. In the first two stages, SHO mimics different movements patterns and the probabilistic predation mechanism of sea horses, respectively. In detail, the movement modes of a sea horse are divided into floating spirally affected by the action of marine vortices or drifting along the current waves. For the predation strategy, it simulates the success or failure of the sea horse for capturing preys with a certain probability. Furthermore, due to the unique characteristic of the male pregnancy, in the third stage, the proposed algorithm is designed to breed offspring while maintaining the positive information of the male parent, which is conducive to increase the population diversity. These three intelligent behaviors are mathematically expressed and constructed to balance the local exploitation and global exploration of SHO. The performance of SHO is evaluated on 23 well-known functions and CEC2014 benchmark functions compared with six state-of-the-art metaheuristic algorithms. Finally, five real-world engineering problems are utilized to test the effectiveness of SHO. The experimental results demonstrate that SHO is a high-performance optimizer and positive adaptability to deal with constraint problems. SHO source code is available from: https://www.mathworks.com/matlabcentral/fileexchange/115945-sea-horse-optimizer

Keywords Sea-horse optimizer · Metaheuristic · Swarm intelligence · Optimization

1 Introduction

An optimization problem refers to seek the best solution and achieve the optimal value of its objective function under a series of constraints. Before the era of heuristic optimization, the canonical optimization algorithms, such as the gradient descent method and the Newton-like method, were the most common methods for solving some mathematical and applied problems. However, these methods tend to trap into local optimum when handling the large-scale, high-dimensional and nonlinear complex problems, or even have difficult to solve them under more complex multi-coupling constraints [1]. Fortunately, a variety of metaheuristic algorithms (MAs) have been created to tackle these challenging issues. In general, MAs are a community of nature-inspired methods with certain stochastic operators [2]. Obviously, randomness is the essential characteristic of MAs [3], which means that the obtained solutions in each iteration are different with probabilities and may be scattered anywhere in the search space. Compared with the traditional optimization techniques, it is not completely guarantee that every iterative feasible solution becomes better, but has a greater capacity to escape local extremum, which is beneficial to explore and obtain the global optimal solution by multi-agent parallel iterations. Besides, MAs do not depend on the gradient information of functions and no strict requirements for their differentiability and convexity. Moreover, MAs are also not sensitive to initial positions of individuals. As a result, benefit from simple heuristic mechanisms and implementation procedures, MAs have become a class of effective substitutions for the canonical optimization methods to solve large-scale, high-dimensional, and multi-objective complex problems. In recent years, MAs have been successfully applied in feature selection [4], berth allocation [5], multi-objective optimization [6], and many other fields.

As shown in the first part of Fig. 1, the traditional metaheuristics use only one search agent for searching in each

✉ Shijie Zhao
zhaoshijie@lntu.edu.cn

[1] Institute of Intelligence Science and Optimization, Liaoning Technical University, Fuxin 123000, China

[2] Institute for Optimization and Decision Analytics, Liaoning Technical University, Fuxin 123000, China

iteration. For example, Simulated Annealing (SA) [7] is inspired by the solid annealing principle. Based on the greedy strategy, it adds stochastic factors to get a new better solution than the current one with iterations. Tabu Search (TS) [8] uses the short-term memory to record and select the optimization process to guide the next search direction. However, these methods are accessible to get local optimums rather than the global optimum, because only single point is employed for searching and mining solutions.

Different from the first-part heuristic methods, most of current metaheuristics are employing multi- agents in the population to perform parallel iterative searching. Considering the difference of heuristic mechanisms, they can be roughly divided into other four categories, as shown in Fig. 1. The first category belongs to evolutionary algorithms (EAs), which is inspired by evolutionary behaviors in nature [9], such as genetic or mutation. EAs simulate the collective learning in a group with multiple individuals, and each represents a feasible solution in the search space. In order to improve fitness values and develop towards the global optimum, EAs are initiated by randomly generating a population and further iteratively evolved by certain evolutionary operators including selection, mutation, or recombination. In each iteration, due to the randomness of evolutionary operators [10], this kind of metaheuristics have the strong local extremum avoidance ability in most cases. And the more popular are Genetic Algorithm (GA) [11] imitating Darwin's natural selection theory and the principles of genetics. In addition, EAs also consist of Differential Evolution (DE) [12], Genetic Programming (GP) [13], and Evolutionary Strategies (ES) [14]. For the second category, the swarm-based intelligence algorithms (SIs) [15] are inspired by intelligent behaviors of biological communities in nature. SIs mainly simulate social behaviors and information exchanges from plants, animals or other living creatures. The outstanding feature of SIs utilizes the swarm intelligence to search cooperatively, which results in finding the optimal solution in the search space. Frequently, the hunting, predation and reproduction are common and familiar social behaviors in animals. Based on these behaviors, many metaheuristics have been proposed continuously, including Particle Swarm Optimization (PSO) [16] Grey Wolf Optimizer (GWO) [17], Ant Lion Optimizer (ALO) [18], Dragonfly Algorithm (DA) [19], Crow Search Algorithm (CSA) [20], Squirrel Search Algorithm (SSA) [21], Seagull Optimization Algorithm (SOA) [22], Harris Hawks Optimization (HHO) [23], Chimp Optimization Algorithm (ChOA) [24], Tunicate Swarm Algorithm (TSA) [25], Marine Predators Algorithm (MPA) [26], Slime Mould Algorithm (SMA) [27] and Golden Eagle Optimizer (GEO) [28]. Moreover, some plant-inspired SIs in this category are also presented, such as Flower Pollination Algorithm (FPA) [29] from the pollination mechanism of flowering plants, and Sun Flower Optimization (SFO) [30] motivated by the light orientation rule of plants. Besides, certain Human-based SIs have been raised in recent years, for instance, Behavior-based Optimization Algorithm (HBBO) [31], Forensic-based Investigation algorithm (FBI) [32], Political Optimizer (PO) [33], Group Teaching Optimization Algorithm (GTOA) [34]. The third category is Physics-based metaheuristics, which contains Multi-verse Optimizer (MVO) [35], Artificial Electric Field Algorithm (AEFA) [36], Henry Gas Solubility Optimization (HGSO) [37], Archimedes Optimization Algorithm (AOA) [38] and Equilibrium Optimizer (EO) [39].

In addition, many researchers have recently proposed MAs based on certain mathematical rules, that is, they take the mathematical rules or formulas as heuristics. MAs are easier to make a more logical explanation for their optimization process and have strong adaptation to address some complex optimizing problems. And more and more MAs based on certain specific mathematical laws are proposed, such as Sine Cosine Algorithm (SCA) [40] drawing lessons from the sine and cosine formulas, self-organizing map (EA-SOM) [41], Gradient-based Optimizer (GBO) [42] and RUNge Kutta optimizer (RUN) [43].

Although existing metaheuristic algorithms can be utilized for solving many challenging problems, their flexibility does not mean that they are cost-free. Wolpert and Macready [44] introduced the *No Free Lunch* (*NFL*) theorem, whose conclusion is that the performance of optimization algorithms is equivalent for the mutual compensation of all possible functions. In other words, the superiority of an optimization algorithm on a particular set of problems does not necessarily

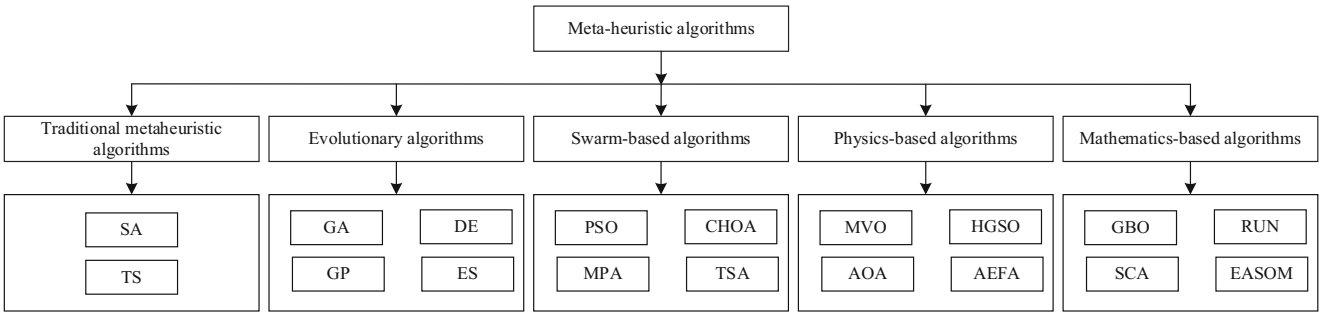


Fig. 1 Classification of metaheuristic algorithms

ensure the same properties on other problems. This is the basis for the development of optimization theory. Up to now, how to strike a proper balance between exploration and exploitation for MAs is still a vital problem to be solved. Hence, in this paper, a new swarm-based intelligence optimization algorithm is proposed: Sea-horse Optimizer (SHO). SHO mimics the moving, predating and breeding behaviors of sea horses in nature. The main contributions of this paper can be shown as bellows:

- Different movement modes, the probabilistic predation mechanism and the unique breeding characteristic of sea horses are constructed and expressed mathematically in detail.
- The effectiveness of the proposed algorithm is evaluated on 23 well-known functions and CEC2014 benchmark functions.
- Statistical analysis, convergence analysis, Wilcoxon test, and Friedman test are used to evaluate optimization performance of SHO, and the experimental results are compared with six state-of-the-art metaheuristic algorithms.
- The constraint programming of SHO is studied for dealing with six common engineering design problems.

The remainder of this paper organizes as follows. Section 2 introduces the motivation and mathematical modeling of the proposed algorithm in detail. Section 3 analyzes the experimental results of SHO compared with six metaheuristics on 53 benchmark functions. In Section 4, SHO is applied in engineering design problems. Finally, Section 5 concludes our works and prospects for future studies.

2 Sea-horse optimizer (SHO)

This section presents a detailed introduction of the SHO algorithm. Firstly, this paper provides the life of sea horses and specifies the motivation of the proposed SHO algorithm. Then, the mathematical modellings are constructed. Finally, the complexity of the algorithm is analyzed and discussed.

2.1 Sea horse

The sea horse (Fig. 2), scientifically known as *hippocampus*, is a general term for several kinds of small fish in warm waters. Sea horses are widely distributed in the tropical, subtropical, and temperate shallow waters [45]. There are about 80 species of sea horses, including some unnamed species [46].

Sea horse gets its name because its head resembles a horse's head. The length of a sea horse varies from 2 cm to 30 cm. For example, the adult size of pygmy sea horse is only about 2 cm, while the adult *hippocampus abdominalis* can be up to 30 cm in length [47]. The snout of a sea horse likes a tubular. Its length affects the rotation of the head and is deemed to be closely related to feeding [48]. Roos et al. [49] investigated that the sea horse has relatively strong predation ability in the larval stage. Sea horses mainly prey on zooplankton and small crustaceans, such as small bran shrimp [50]. When feeding, the tubular kiss extends to the food, puffing out the cheeks and opening the mouth to suck the food. The head of a sea horse generally protrudes from the top, forming a crown. The gills department is uplifted, and the opercular usually has a radial crest. There is only one dorsal fin between trunk and tail for propulsion, and no ventral or caudal fins. Its entire body is completely surrounded by membranous bone plates. Sea horses generally live in clean, low-tide waters that are rich in algal and coral. To rest and escape prey, sea horses change their color with matching that of surroundings.

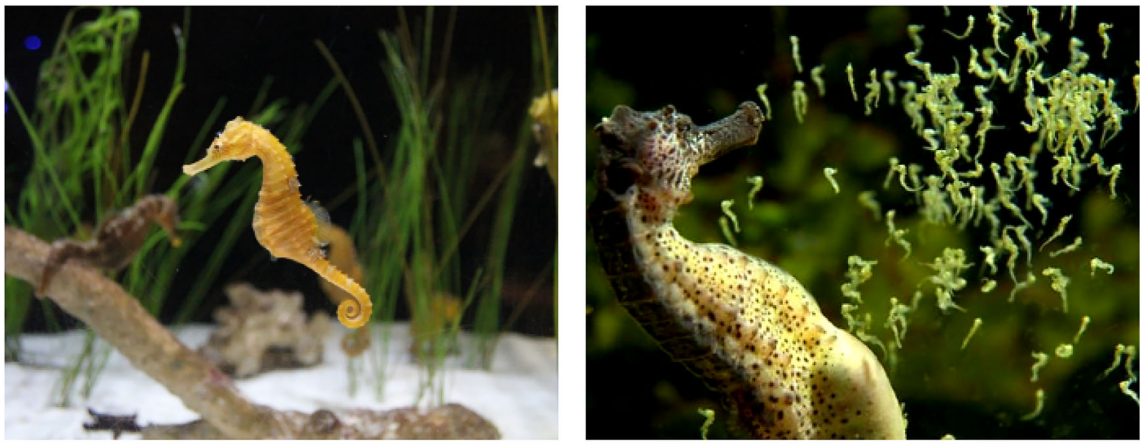
The tail of a sea horse has a structure and function that no other fish has. Studies [51] have shown that this tail' cross-sectional area is square, with each layer of bones consisted of four tightly wound L-shaped bones, which is to protect the middle spines and improve the grasping capacity of the tail for wrapping around the attached objects to rest or carry the body weight upside down. Besides, the sea horse has a special swimming posture. After releasing the algae attached to its tail, a sea horse stands with its head upright in the water and relies entirely on its dorsal and pectoral fins for movement.

Sea horses are the one and only animal on earth that males give birth to offspring, as shown in Fig. 2b. The male sea horse has a brood pouch in front or on the side of its abdomen. During mating, the female sea horse releases ova into the brood pouch, and the male is responsible for fertilizing those ova. Meanwhile, the male sea horses keep the fertilized ova in the brood pouch until they are fully formed and then release them into the seawater.

2.2 Inspiration

Predation, movement and breeding behaviors are particularly critical for the life of the sea horse, which are described below.

- In terms of movement behavior, the sea horse sometimes curls the tail around a stem (or leaf) of algae. Since the stems present spiral floating changes around the roots of algae under the action of marine vortices, the sea horse carries out spiral movement at this time. At other times, Brownian motion occurs when the sea horse hangs upside down from drifting algae or other objects and moves randomly with the waves.
- For predation behavior, sea horses make use of the head's particular shape to sneak up on the prey, and then capture it with up to a 90% chance of success.



(a) (b)

Fig. 2 The behaviors of the sea horse

- For breeding behavior, the male and female sea horses randomly mate to produce new offspring, which contributes to inheriting certain excellent information from their fathers and mothers.

In summary, these three behaviors enable sea horses to adapt to environment and survive better. The proposed SHO algorithm is mainly inspired by the above-mentioned three behaviors. Therefore, these behaviors are our motivation to develop this novel optimizer through mathematical modellings.

2.3 Sea-horse optimizer

The proposed SHO algorithm consists of three crucial components, i.e., movement, predation and breeding. To balance the exploration and exploitation of SHO, the local and global search strategies are designed for the social behaviors of movement and predation, respectively. And the breeding behavior is performed with the completion of the first two behaviors. Their mathematical modellings would be expressed and discussed as follows in detail.

2.3.1 Initialization

Similar to other existing metaheuristics, SHO also starts from the population initialization. Suppose each sea horse represents a candidate solution in the search space of problems, the whole population of sea horses (termed by ***Seahorses***) can be expressed as:

$$Seahorses = \begin{bmatrix} x_1^1 & \cdots & x_1^{Dim} \\ \vdots & \ddots & \vdots \\ x_{pop}^1 & \cdots & x_{pop}^{Dim} \end{bmatrix} \tag{1}$$

where *Dim* denotes the dimension of the variable and *pop* is the population size.

Each solution is randomly generated between the lower bound and upper bound of a specify problem, denoted by *LB* and *UB*, respectively. The expression of the i^{th} individual X_i in search space [*LB*, *UB*] is

$$\begin{aligned} X_i &= \left[x_i^1, \ldots, x_i^{Dim}\right] \\ x_i^j &= rand \times \left(UB^j - LB^j\right) + LB^j \end{aligned} \tag{2}$$

where *rand* denotes the random value in [0, 1]. x_i^j denotes the j^{th} dimension in the i^{th} individual. *i* is a positive integer ranging from 1 to *pop* and *j* is a positive integer in the range [1, *Dim*]. LB^j and UB^j imply the lower bound and the upper bound of the j^{th} variable of the optimized problem.

Taking the minimum optimization problem as an example, the individual with the minimum fitness is regarded as the elite individual, denoted by X_{elite}. X_{elite} can be obtained by Eq. (3).

$$X_{elite} = argmin(f(X_i)) \tag{3}$$

where $f(\cdot)$ represents the objective function value of a given problem.

2.3.2 The movement behavior of sea horses

For the first behavior, the different movement patterns of sea horses approximately follow the normal distribution *randn*(0, 1). In order to trade off the exploration and exploitation performance, we take $r_1 = 0$ as the cut-off point, half for the local mining and the other half for the global search. So movements can be divided into the following two cases.

Case 1 The spiral motion of the sea horse with the vortex in the sea. It mainly realizes the local exploitation of SHO, when the normal random value r_1 is located at the right side of the cut-

off point. Sea horses are moving towards the elite X_{elite} by following the spiral motion. Especially, the Lévy flight [52] is employed to simulate the movement step size of sea horses, which is conducive to the sea horse with the high probability crossing to other positions in early iterations and avoiding the excessive local exploitation of SHO. At the same time, this spiral moving mode of the sea horse changes constantly the rotation angle for expanding the neighborhoods of current local solutions. In this case, it can be expressed mathematically to generate the new position of a sea horse as follows:

$$X^1_{new}(t+1) = X_i(t) + Levy(\lambda)((X_{elite}(t) - X_i(t)) \times x \times y \times z + X_{elite}(t)) \tag{4}$$

where $x = \rho \times cos(\theta)$, $y = \rho \times sin(\theta)$ and $z = \rho \times \theta$ denote three-dimensional components of coordinates (x, y, z) under the spiral movement, respectively, which are helpful to update the positions of search agents. $\rho = u \times e^{\theta v}$ represents the length of the stems defined by the logarithmic spiral constants u and v (Set to $u = 0.05$ and $v = 0.05$). θ is the random value between $[0, 2\pi]$. $\boldsymbol{Levy}(z)$ is the Lévy flight distribution function and is calculated by Eq. (5) [52].

$$\boldsymbol{Levy}(z) = s \times \frac{w \times \sigma}{|k|^{\frac{1}{\lambda}}} \tag{5}$$

In Eq. (5), λ is the random number between $[0, 2]$ (Set to $\lambda = 1.5$ in this paper). s is a fixed constant of 0.01. w and k are random numbers between $[0, 1]$. σ is calculated by using Eq. (6).

$$\sigma = \left(\frac{\Gamma(1+\lambda) \times \sin\left(\frac{\pi\lambda}{2}\right)}{\Gamma\left(\frac{1+\lambda}{2}\right) \times \lambda \times 2^{\left(\frac{\lambda-1}{2}\right)}} \right) \tag{6}$$

Case 2 The Brownian motion of the sea horse with the sea waves. Under the action of drifting, the exploration of SHO is carried out, when r_1 is located at the left side of the cut-off point. In this case, the search operation is important for the local extremum avoidance of SHO. Brownian motion is applied to mimic another moving length of the sea horse for ensuring its better exploration in the search space. Its mathematical expression for this case is

$$X^1_{new}(t+1) = X_i(t) + rand * l * \beta_t * (X_i(t) - \beta_t * X_{elite}) \tag{7}$$

where l is the constant coefficient (Set to $l = 0.05$ in this paper). β_t is the random walk coefficient of Brownian motion, which is a random value obeying the standard normal distribution in essence. And it can be given by using Eq. (8) [53].

$$\beta_t = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{x^2}{2}\right) \tag{8}$$

Totally, these two cases can be formulated to obtain the new position of the sea horse at iteration t as follows.

$$X^1_{new}(t+1) = \begin{cases} X_i(t) + Levy(\lambda)((X_{elite}(t) - X_i(t)) \times x \times y \times z + X_{elite}(t)) & r_1 > 0 \\ X_i(t) + rand * l * \beta_t * (X_i(t) - \beta_t * X_{elite}) & r_1 \le 0 \end{cases} \tag{9}$$

where $r_1 = \boldsymbol{randn}()$ is a normal random number.

Figure 3 illustrates the position updating diagram of the sea horse by following two kinds of different movement modes, i.e., the spiral or Brownian motion, and both are reflected the moving randomness of the sea horse based on the uncertain environment in the sea.

2.3.3 The predation behavior of sea horses

There are two outcomes for the sea horse to prey on zooplankton and small crustaceans: success and failure. Considering that the probability of the sea horse succeed in capturing food is over 90%, the random number r_2 of SHO is designed to distinguish these two results and set to a critical value with 0.1. Since the elite, to a certain, indicates the approximate position of the prey, the predation success emphasizes the exploitation ability of SHO. If $r_2 > 0.1$, it means that the predation of the sea horse is successful, that is, the sea horse sneaks up on the prey (elite), moves faster than the prey and finally captures it. Otherwise, when the predation fails, the response speed of both is opposite to that before, which implies the sea horse trends to explore the search space. The mathematical expression of this predation behavior is:

$$X^2_{new}(t+1) = \begin{cases} \alpha * \left(X_{elite} - rand * X^1_{new}(t)\right) + (1-\alpha) * X_{elite} & if\ r_2 > 0.1 \\ (1-\alpha) * \left(X^1_{new}(t) - rand * X_{elite}\right) + \alpha * X^1_{new}(t) & if\ r_2 \le 0.1 \end{cases} \tag{10}$$

where $X^1_{new}(t)$ denotes the new position of the sea horse after movement at the iteration t. r_2 is the random number between $[0, 1]$. α decreases linearly with iterations to adjust the moving step size of the sea horse for hunting prey, and calculates by Eq. (11).

$$\alpha = \left(1 - \frac{t}{T}\right)^{\frac{2t}{T}} \tag{11}$$

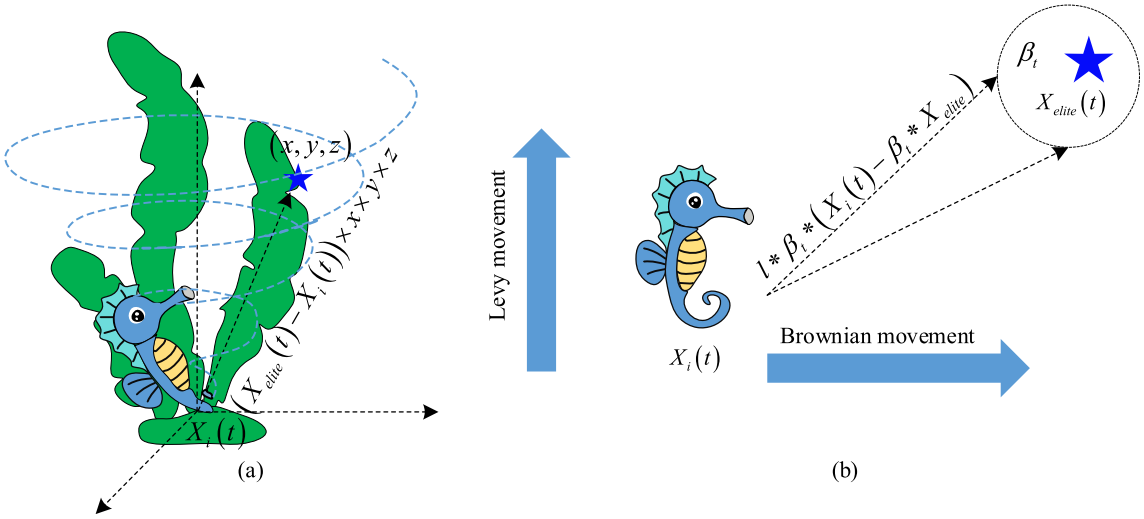


Fig. 3 Different motion patterns of the sea horse in sea

where T denotes the maximum number of iterations.

Figure 4 displays the two possible outcomes of the predation behavior of the sea horse. As shown in Fig. 4, the blue star position indicates the updated position of the sea horse, and the approximate position of the prey is marked with the red dot. It can be seen from Fig. 4a that when the sea horse preys successfully, the sea horse moves to the elite position. Under the control of parameter α, it will gradually converge to the global optimal individual with the iterations increasing. In Fig. 4b, the global search is performed because the prey cannot be captured. The parameter $1 - \alpha$ is applied to the vector between the current individual and the elite, and α is acted on the current updated individual. This is designed to allow sea horses to globally search in the early iterations and avoid over-exploiting in the later iterations.

2.3.4 The breeding behavior of sea horses

The population is categorized into male and female groups according to their fitness values. It is worthwhile emphasizing that, since male sea horses are responsible for breeding, the SHO algorithm takes half of the individuals with best fitness values as fathers and the other half as mothers. This division will facilitate the inheritance of good characteristics between fathers and mothers for producing the next generation, and avoid the over-localization of new solutions. The concise mathematical expression for the role assignment of sea horses is

$$\begin{aligned} fathers &= X^2_{sort}(1 : pop/2) \\ mothers &= X^2_{sort}(pop/2 + 1 : pop) \end{aligned} \tag{12}$$

where X^2_{sort} denotes all X^2_{new} in ascending order of fitness values. ***fathers*** and ***mothers*** indicate from the male and female populations, respectively.

Males and females are randomly mated to produce new offspring. To make the proposed SHO algorithm execute easily, it is assumed that each pair of sea horses only breeds a child. The expression of the i^{th} offspring is as follows.

$$X_i^{\boldsymbol{offspring}} = r_3 X_i^{\boldsymbol{father}} + (1 - r_3) X_i^{\boldsymbol{mother}} \tag{13}$$

where r_3 is the random number between [0, 1]. i is a positive integer in the range of [1, pop/2]. $X_i^{\boldsymbol{father}}$ and $X_i^{\boldsymbol{mother}}$ represent

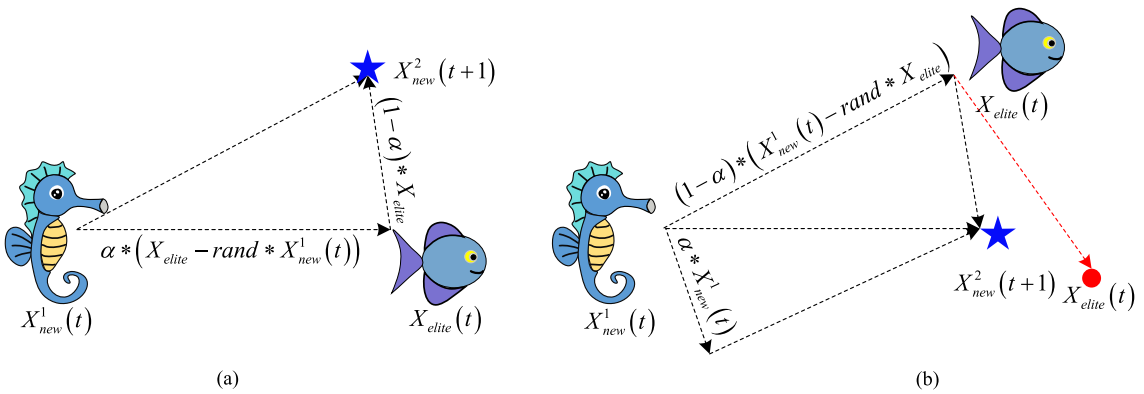


Fig. 4 Predation process of the sea horse

randomly selected individuals from the male and female populations, respectively.

The breeding process of sea horses is shown in Fig. 5. As seen from Fig. 5a, each individual is sorted in ascending order according to their fitness values. Figure 5b shows approximatively the position of a new generated offspring. It is randomly created in the line between parents, which effectively transmits genetic information between two subpopulations.

2.3.5 The implementation process of SHO

The pseudo-code of the proposed SHO algorithm is shown in Algorithm 1. The implementation process of SHO starts the population initialization by creating a set of random solutions. After the sea horse population is updated by Eqs. (9) and Eq. (10), Eq. (11) is employed to breed the offspring. A new population is composed of the offspring and the previous updated sea horses. However, the size of this new population is $1.5Pop$. To prevent the population from expanding without limit, each individual in the new population is estimated. Individuals are sorted in ascending order from top to bottom according to fitness values, and the first pop sea horses are iteratively selected as the new population for the next evolutionary process.

2.4 Computational complexity

The complexity is an important indicator to theoretically measure the optimization performance of algorithms. The time and space complexity of the proposed algorithm are discussed below.

The time complexity analysis is as follows.

(i) It takes $O(n \times Dim)$ time to initialize the population of sea horses, where n denotes the population size and Dim represents the dimension of variables. $O(n)$ is required to calculate the fitness value of each sea horse.
(ii) During iterations, calculating the fitness value of each sea horse is $O(\boldsymbol{Maxiteration} \times n)$ time, where $\boldsymbol{Maxiteration}$ is the maximum number of iterations.
(iii) In SHO, defined the movement behavior, predation behavior and breeding behavior of sea horses need $O(\boldsymbol{Maxiteration} \times n \times Dim)$ time.

Hence, the total computational time complexity of the proposed SHO algorithm is $O(\boldsymbol{Maxiteration} \times n \times Dim)$.

For the space complexity, the maximum space is considered to be occupied during the generation of the offspring in the iterations. Therefore, the space complexity of SHO is $O(n \times Dim)$.

3 Experimental results and discussion

This section is the simulation experiment of SHO. To validate SHO's performance of local exploitation, local extremum avoidance and global exploration, the statistical analysis and convergence analysis were performed on 23 well-known functions and CEC2014 benchmark functions. Then, the high dimensional optimization performance of SHO was also verified. Finally, statistical tests were carried out on these test functions. Six state-of-the-art metaheuristics, namely GA [11], DA [19], SCA [40], ChOA [24], TSA [25] and SFO [30], were compared with our proposed SHO algorithm.

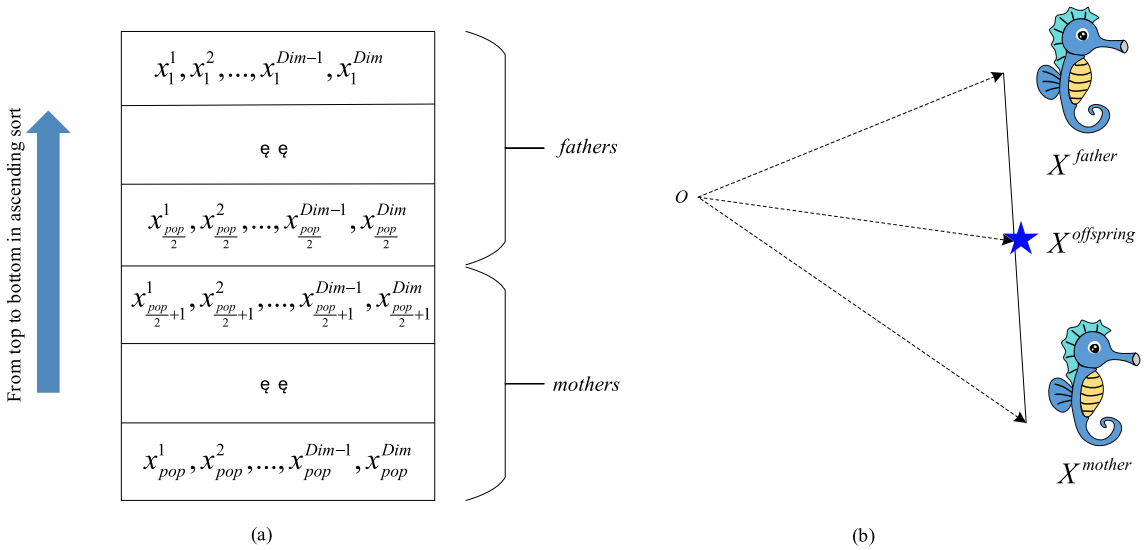


Fig. 5 Breeding process of sea horses

Table 1 shows the parameter settings of all metaheuristics. All experiments have been accomplished on MatlabR2018a version with the operating system of Windows 10(64-bit) and Intel(R) Core(TM) I7-9750H CPU (@ 2.60 GHz).

Input: The population size pop, the maximum number of iterations T and the variable dimension Dim
Output: The optimal search agent X_{best} and its fitness value f_{best}
Initialize sea horses $X_i(i=1,\cdots,N)$
Calculate the fitness value of each sea horse
Determine the best sea horse X_{elite}
while $(t<T)$ **do**
/*Movement behavior*/
if $r_1 = randn > 0$ **do**
Set $u=0.05, v=0.05$ constant parameters
Rotation angle $Rand(-2\pi, 2\pi)$
Generate Levy coefficient by using Eq. (5)
Update positions of the sea horses by using Eq. (4)
else if do
Set $p=0.05$ constant parameter
Update positions of the sea horses by using Eq. (7)
end if
/*predation behavior */
Update positions of the sea horses by using Eq. (10)
Handle variables out of bounds
Calculate the fitness value of each sea horse
/*Breeding behavior */
Select mothers and fathers by using Eq. (12)
Breed offspring by using Eq. (13)
Handle variables out of bounds
Calculate the fitness value of each offspring
Select the next iteration population from the offspring and parents ranked top pop in fitness values
Update elite (X_{elite}) position
$t = t+1$
end while

Table 1 Parameter settings of all algorithms

Algorithm	Parameters	Value
All algorithms	Population size	30
	Dim	30
SHO	r_1	0
	probability of successr_2	0.1
GA	Crossover	0.8
	Mutation	0.2
SCA [40]	Number of elites a	2
DA [19]	Base coefficient b	[0.1−0]
	Neighborhood radius r	[0.25−2.25] ×[ub−lb]
	Separation coefficient s	2×b
	Alignment coefficient$a$	2×b
	Cohesion coefficient$c$	2×b
	Food attraction coefficient$f$	2
	Enemy distraction coefficient$e$	b
ChOA [24]	r_1, r_2	Random
	m	Chaotic
TSA [25]	Parameters $P_{\max}$	4
	Parameters $P_{\min}$	1
SFO [30]	Pollination rate p	0.05
	Mortality rate m	0.1
	survival rate s	0.85

3.1 Benchmark functions

23 well-known functions can be grouped into three categories, namely unimodal, multimodal and fixed-dimension multimodal functions. In detail, the unimodal functions $F1-F7$ (in Table 2) have one and only one global optimal value, which are used to estimate the convergence accuracy and convergence speed. Multimodal functions$F8-F13$ (in Table 3) with multiple local optimums are appropriate for testing the local extreme avoidance and the global exploration performance. The global exploration ability in low-dimensional conditions can be tested by fixed-dimension multimodal functions $F14-F23$ (in Table 4). Moreover, the proposed SHO algorithm is further evaluated in the modern universal test suite CEC2014 benchmark functions, as reported in Table 5.

CEC2014 suite is the general test standard of modern algorithms, which has strong test suitability for all kinds of metaheuristic algorithms. Because of the dynamic and complexity of these benchmark functions, they are more convincing to validate the optimization performance of the proposed SHO. In order to keep the competition fairness and objectivity of each metaheuristic, all algorithms have been independently run for 30 times in each experiment. Mean (Mean) and Standard deviation (Std) results of 30 experiments were employed as statistical indicators to measure their optimization performance.

3.2 Qualitative analysis

This experiment was performed on the first two dimensions of variables in 23 well-known functions, mainly aiming to observe the behavioral optimization ability of SHO. The population size and the maximum number of iterations were set to 30 and 500, respectively. Figure 6 depicts the qualitative measurements of SHO for tackling partial unimodal and multimodal functions. The first column describes the topological structures of test functions in a two-dimensional view of the first two variables. The last 4 columns are the test indictors, respectively the search history, the trajectory of 1st sea horse

Table 2 Unimodal functions

Function name	Expression	Search space	Dim	$f_{\min}$
Sphere	$F_1(x) = \sum_{i=1}^{n} x_i^2$	[−100,100]	30	0
Schwefel 2.22	$F_2(x) = \sum_{i=0}^{n} \lvert x_i \rvert + \prod_{i=0}^{n} \lvert x_i \rvert$	[−10,10]	30	0
Shifted Schwefel's Problem 1.2	$F_3(x) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j \right)^2$	[−100,100]	30	0
Schwefel 2.21	$F_4(x) = max_i\{\lvert x_i \rvert, 1 \le i \le n\}$	[−100,100]	30	0
Rosenbrock	$F_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	[−30,30]	30	0
Step	$F_6(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	[−100,100]	30	0
Quartic	$F_7(x) = \sum_{i=1}^{n} i x_i^4 + \boldsymbol{random}[\boldsymbol{0}, \boldsymbol{1})$	[−1.28,1.28]	30	0

in the first dimension, the average fitness of all sea horses and convergence curve.

The search history can reflect the position distribution of all sea horses during the iterative process, where the red rot represents the global optimal solution obtained by SHO. It can be evidently detected from the second column of Fig. 6 that numerous search agents cluster surrounding global optimum in unimodal functions. Yet for multimodal functions, many search agents scatter mostly near multiple local optima in the whole search space. For unimodal functions, it is beneficial to the local re-exploitation of the proposed SHO algorithm to seek a higher accuracy. In terms of multimodal functions, the decentralized form reveals sea horses' exploration throughout the whole region and the tradeoff situation among several local optimal values.

The first dimensional trajectory of 1st sea horse is devoted to reflect the primary exploratory behavior of the search agent. As shown in the third column of Fig. 6, the curve fluctuates significantly in the initial stage of iterations, while the amplitude of vibration tends to be slow felt in late iterations. The curve changes guarantee that SHO turns from global exploration to local exploitation iterative process by degrees. In unimodal functions, the duration of the oscillation state is short, which indicates that SHO converges quickly. However, the difference is that the oscillation state in multimodal functions usually lasts for long times, even exceeding 60%–70% of the iterative process, which reflects the global exploration capacity of SHO.

The average fitness value represents the average target value of all sea horses in each iteration. It is mainly used to reflect the general tendency of population evolution. In the fourth column of Fig. 6, it can be found intuitively that the entire population progresses from the initial stage to the last. With the constant updating of sea horses, the average fitness values lean to decline. For unimodal functions, the curve drops rapidly merely at the beginning of iteration. After the rapid descent, the tangent slope of the curve fluctuation verges on stable. This embodies that SHO converges to near the optimal value in early iterations and then strengthens the exploitation accuracy. In contrast, multimodal functions have steeper curve

Table 3 Multimodal functions

Function name	Expression	Search space	*Dim*	$f_{\min}$
Schwefel	$F_8(x) = \sum_{i=1}^{n} -x_i sin\left(\sqrt{\lvert x_i \rvert} \right)$	[−500,500]	30	$-418.9829 \times D$
Rastrigin	$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10cos(2\pi x_i + 10) \right]$	[−5.12,5.12]	30	0
Ackley	$F_{10}(x) = -20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \right) + exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i) \right) + 20 + e$	[−32,32]	30	0
Shifted Rotated Griewank's without Bounds	$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	[−600,600]	30	0
Penalized 1	$F_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1}) \right] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{x_i + 1}{4}$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	[−50,50]	30	0
Penalized 2	$F_{13}(x) = 0.1\{\sin^2(3\pi x_i) + \sum_{i=1}^{n}(x_i - 1)^2 \left[1 + \sin^2(3\pi x_i + 1) \right] + (x_n - 1)^2 \left[1 + \sin^2(2\pi x_n) \right]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	[−50,50]	30	0

Table 4 Fixed-dimension Multimodal functions

Function name	Expression	Search space	Dim	$f_{\min}$
Foxholes	$F_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}\left(x_i-a_{ij}\right)}\right)^{-1}$	$[-65.536, 65.536]$	2	1
Kowalik	$F_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1\left(b_i^2+b_ix_2\right)}{b_i^2+b_ix_3+x_4}\right]^2$	$[-5,5]$	4	0.00030
6 Hump Camel Back	$F_{16}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	$[-5,5]$	2	−1.0316
Branin	$F_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)cosx_1+10$	$[-5,10]$ $[0,15]$	2	0.3983
GoldStein Price	$F_{18}(x)=\left[1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2\right)+6x_1x_2+3x_2^2\right]\times\left[30+(2x_1-3x_2)^2\times\left(18-32x_i+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]$	$[-2,2]$	2	3
Hartman3	$F_{19}(x)=-\sum_{i=1}^{4}c_iexp\left(-\sum_{i=1}^{3}a_{ij}\left(x_j-p_{ij}\right)^2\right)$	$[0,1]$	3	−3.86
Hartman6	$F_{20}(x)=-\sum_{i=1}^{4}c_iexp\left(-\sum_{j=1}^{6}a_{ij}\left(x_j-p_{ij}\right)^2\right)$	$[0,1]$	6	−3.32
Shekel5	$F_{21}(x)=-\sum_{i=1}^{5}\left[(X-a_i)(X-a_i)^T+c_i\right]^{-1}$	$[0,10]$	4	−10.1532
Shekel7	$F_{22}(x)=-\sum_{i=1}^{7}\left[(X-a_i)(X-a_i)^T+c_i\right]^{-1}$	$[0,10]$	4	−10.4028
Shekel10	$F_{23}(x)=-\sum_{i=1}^{10}\left[(X-a_i)(X-a_i)^T+c_i\right]^{-1}$	$[0,10]$	4	−10.5363

Table 5 Summaries of CEC-2014 benchmark functions [54]

	No.	Function	Range	Optimal value
Unimodal Functions	$CEC-1$	Rotated High Conditioned Elliptic Function	$[-100,100]$	100
	$CEC-2$	Rotated Bent Cigar Function	$[-100,100]$	200
	$CEC-3$	Rotated Discus Function	$[-100,100]$	300
Simple multimodal Functions	$CEC-4$	Shifted and Rotated Rosenbrock's Function	$[-100,100]$	400
	$CEC-5$	Shifted and Rotated Ackley's Function	$[-100,100]$	500
	$CEC-6$	Shifted and Rotated Weierstrass Function	$[-100,100]$	600
	$CEC-7$	Shifted and Rotated Griewank's Function	$[-100,100]$	700
	$CEC-8$	Shifted Rastrigin's Function	$[-100,100]$	800
	$CEC-9$	Shifted and Rotated Rastrigin's Function	$[-100,100]$	900
	$CEC-10$	Shifted Schwefel's Function	$[-100,100]$	1000
	$CEC-11$	Shifted and Rotated Schwefel's Function	$[-100,100]$	1100
	$CEC-12$	Shifted and Rotated Katsuura Function	$[-100,100]$	1200
	$CEC-13$	Shifted and Rotated HappyCat Function	$[-100,100]$	1300
	$CEC-14$	Shifted and Rotated HGBat Function	$[-100,100]$	1400
	$CEC-15$	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	$[-100,100]$	1500
	$CEC-16$	Shifted and Rotated Expanded Scaffer's F6 Function	$[-100,100]$	1600
Hybrid Functions	$CEC-17$	Hybrid Function 1 (N=3)	$[-100,100]$	1700
	$CEC-18$	Hybrid Function 2 (N=3)	$[-100,100]$	1800
	$CEC-19$	Hybrid Function 3 (N=4)	$[-100,100]$	1900
	$CEC-20$	Hybrid Function 4 (N=4)	$[-100,100]$	2000
	$CEC-21$	Hybrid Function 5 (N=5)	$[-100,100]$	2100
Hybrid Functions	$CEC-22$	Hybrid Function 6 (N=5)	$[-100,100]$	2200
Composition Functions	$CEC-23$	Composition Function 1 (N=5)	$[-100,100]$	2300
	$CEC-24$	Composition Function 2 (N=3)	$[-100,100]$	2400
	$CEC-25$	Composition Function 3 (N=3)	$[-100,100]$	2500
Composition Functions	$CEC-26$	Composition Function 4 (N=5)	$[-100,100]$	2600
	$CEC-27$	Composition Function 5 (N=5)	$[-100,100]$	2700
	$CEC-28$	Composition Function 6 (N=5)	$[-100,100]$	2800
	$CEC-29$	Composition Function 7 (N=3)	$[-100,100]$	2900
	$CEC-30$	Composition Function 8 (N=3)	$[-100,100]$	3000

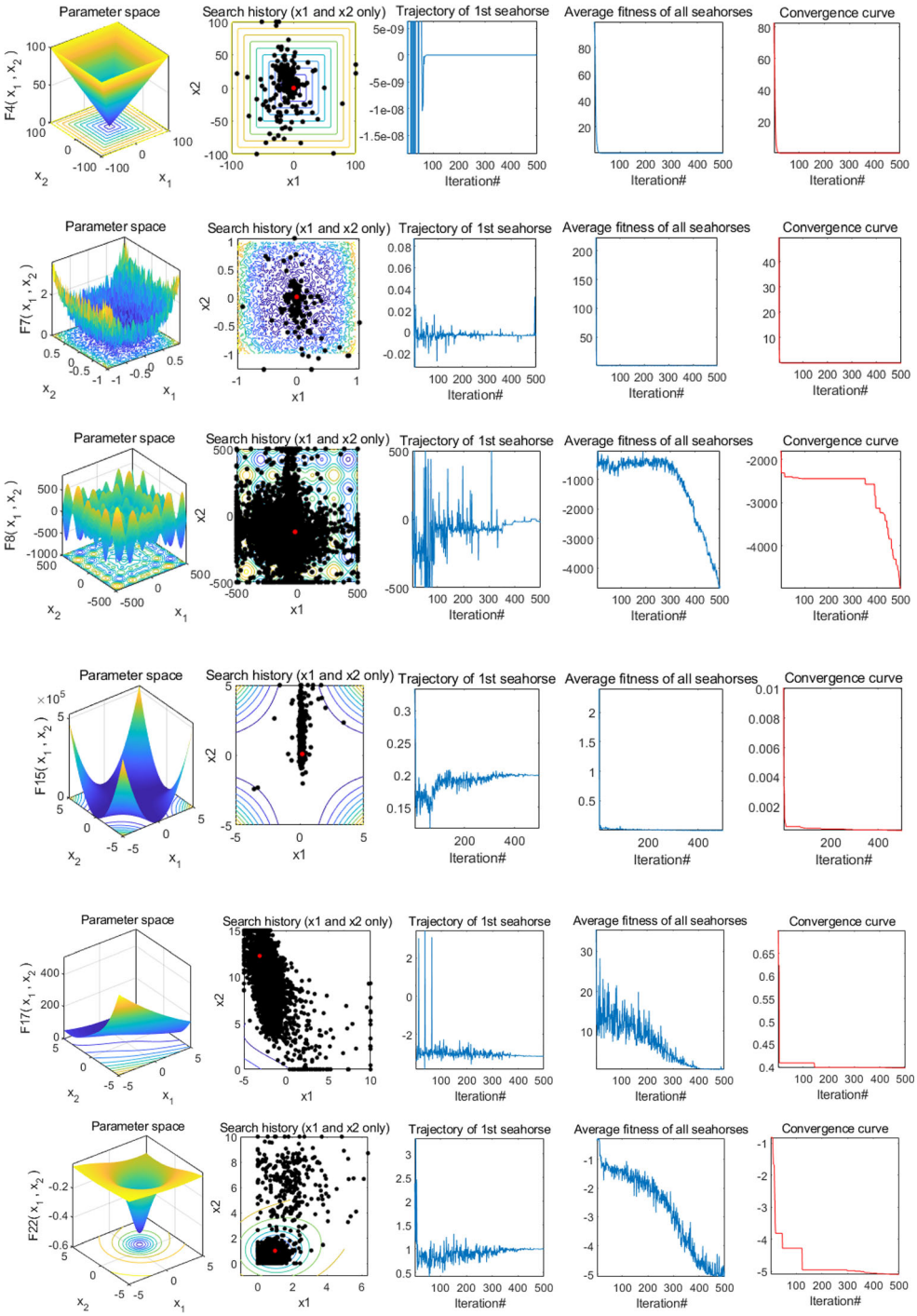


Fig. 6 Quantitative analysis results included topological structure, search history, the trajectory of 1st sea horse, average fitness of all sea horses and convergence curve

rates. The amplitude of the curve decreases with the increase of the number of iterations, which implies the global exploration in the early stage and local exploitation in the later stage of SHO.

The convergence curve represents the behavior of the optimal solution obtained by the sea horse so far. In unimodal functions, the curves drop rapidly after refining solutions. The curves of most multimodal functions descend step by step. This is due to jumping out close a local optimal solution and gradually searching to the global optimal solution. Meanwhile, the switching time between global exploration and local exploitation can also be seen from curves.

3.3 Exploitation analysis

Table 6 shows the statistical results of SHO and other methods evolving 15,000 (30 × 500) times on unimodal functions.

Table 6 Statistical results of unimodal functions

Function		GA	SCA	DA	SFO	TSA	ChOA	SHO
*F*1	Mean	1.4051E+04	1.7454E+01	2.1819E+03	1.3695E+04	2.8824E-21	3.6878E-06	**3.8629E-141**
	Std	4.4683E+03	2.3343E+01	1.3071E+03	2.0724E+03	5.6785E-21	7.0567E-06	**8.5767E-141**
*F*2	Mean	4.3992E+06	2.0590E-02	1.6054E+01	2.4321E+06	1.2417E-13	3.8850E-05	**6.6299E−78**
	Std	1.0818E+07	2.7428E-02	7.4873E+00	6.3112E+06	2.0463E-13	4.7678E-05	**1.6358E-77**
*F*3	Mean	5.0546E+04	8.7862E+03	1.6027E+04	4.7264E+04	4.2578E-04	7.9198E+01	**1.0468E-98**
	Std	1.0318E+04	5.9603E+03	9.3888E+03	1.3149E+04	1.0864E-03	1.3989E+02	**4.2769E-98**
*F*4	Mean	7.0622E+01	3.5448E+01	3.1810E+01	4.5216E+01	3.3187E-01	2.7983E-01	**6.8173E-57**
	Std	4.6237E+00	1.1058E+01	8.1300E+00	3.7794E+00	3.1679E-01	3.1541E-01	**1.0938E-56**
*F*5	Mean	1.8670E+07	2.5994E+04	3.6017E+05	1.0083E+06	2.8276E+01	2.8831E+01	**2.8130E+01**
	Std	8.4628E+06	6.2080E+04	3.9105E+05	3.9273E+05	8.5711E-01	**2.1915E-01**	5.6369E-01
*F*6	Mean	1.3204E+04	2.0646E+01	2.1348E+03	1.3457E+04	3.7791E+00	3.5997E+00	**3.3426E+00**
	Std	4.8985E+03	2.4649E+01	1.0585E+03	2.1479E+03	6.4087E-01	**4.1750E-01**	5.0046E-01
*F*7	Mean	1.0698E+01	1.1626E-01	6.2861E-01	5.4947E-01	1.0674E-02	2.1953E-03	**1.1173E-04**
	Std	5.0069E+00	1.0444E-01	5.5224E-01	2.6805E-01	4.8621E-03	2.0499E-03	**8.7439E-05**

Bold entries highlight the best results of the algorithm on the function

According to Table 6 analysis, SHO achieves best results in 12/14 indicators. For Mean indexes, SHO can outperform other algorithms on seven unimodal functions. The average optimization performance of GA, DA and SFO perform poorly on functions *F*1, *F*2, *F*3, *F*5, and *F*6. For functions *F*1, *F*2, and *F*3, SHO has a greater competitive advantage in optimization accuracy than TSA, which ranked 2. For Std indexes, SHO is optimal on other functions except functions *F*5 and *F*6. Std indicators for functions *F*5 and *F*6, in spite of ChOA is the best followed by SHO, the accuracy is the same order of magnitude as ChOA and the difference is not remarkable. Unimodal test results demonstrate that SHO has the advantages of higher local exploitation capacity, high convergence accuracy, and strong algorithm stability and robustness. Meanwhile, these testing results indicate that the long step search of the spiral motion influenced by vortex and the search to the elite in the predation stage can ensure the exploitation of the searching agents for the global optimal.

3.4 Exploration analysis

Table 7 provides the statistical results of 15,000 (30 × 500) evolutions of each algorithm on multimodal functions. Multimodal functions have several local optimal values, which are applied to test the global optimization ability of the algorithm. Table 7 reveals that SHO performs best in 5/6 of multimodal functions for Mean indicators. For functions *F*9 and *F*10, SHO can obtain the global optimal value, while other algorithms can only search for lower precision (i.e., GA and DA). Likewise, SHO has more outstanding exploration capacity than other algorithms on functions *F*10, *F*12and *F*13. It shows that Brownian motion under the action of drifting plays an important role in these functions. To some extent, SHO can jump out of local extreme values and develop to the global optimal. For function *F*8, GA ranked first in average optimization performance, and TSA ranked second. For functions *F*9, *F*10, *F*11, and *F*12, the standard deviation results of SHO are better than other algorithms, especially the standard deviation results of SHO on functions *F*9 and *F*11 can reach 0. In addition, the Std results of ChOA are optimal on functions *F*8 and *F*13.

Table 8 shows the statistical results of all algorithm evolutions for 15,000 (30 × 500) times on fixed-dimension multimodal functions. It can be seen from Table 8 that SHO has superior Mean indicators on functions *F*15, *F*16, *F*17, and *F*18. For function *F*14, ChOA has the best Mean and Std indicators and DA achieves suboptimal optimization results. SHO obtains the best average fitness and minimal Std indicator on function *F*15. Although SCA, DA, SFO, CHOA and SHO can find global optimal value on function *F*16, SHO has better standard deviation result, which proves that SHO is still competitive. For functions *F*17 and *F*18, DA, SFO and SHO can obtain the global optimal value, as well as the corresponding Std indicators are SFO and SHO minimum respectively. However, SHO's optimization performance on functions *F*19 to *F*23 are relatively general, but SHO is still better than some comparison algorithms such as GA, SCA and CHOA. For function *F*19, DA's Mean and Std results are the best and SFO gets the second-best results. TSA has the best Mean indicator on function *F*20. DA is superior to other algorithms on function *F*21. For functions *F*22 and *F*23, DA and SFO respectively achieve best Mean indicators.

The results of multimodal functions show that SHO has good global optimization performance and can effectively avoid local extremums. Competing with the new or state-of-the-art metaheuristic methods, SHO still has high convergence

Table 7 Statistical results of multimodal functions

Function		GA	SCA	DA	SFO	TSA	ChOA	SHO
*F*8	Mean	**-7.5442E+03**	−3.7403E+03	−5.4263E+03	−3.6413E+03	−6.1500E+03	−5.7246E+03	−5.8448E+03
	Std	7.5106E+02	2.8028E+02	5.5566E+02	3.5203E+02	5.8402E+02	**5.5321E+01**	4.6053E+02
*F*9	Mean	1.7899E+02	3.2522E+01	1.6893E+02	3.3396E+02	1.8346E+02	1.2816E+01	**0.0000E+00**
	Std	3.5110E+01	3.5504E+01	4.0873E+01	2.8430E+01	5.1740E+01	1.1501E+01	**0.0000E+00**
*F*10	Mean	1.6555E+01	1.2922E+01	1.0653E+01	1.9350E+01	1.4812E+00	1.9963E+01	**4.3225E-15**
	Std	8.6338E-01	9.3093E+00	1.5009E+00	8.1813E-01	1.6249E+00	9.8274E-04	**6.4863E-16**
*F*11	Mean	1.2519E+02	1.0597E+00	2.1235E+01	3.4169E+02	6.7253E-03	3.0769E-02	**0.0000E+00**
	Std	7.5817E+01	5.0028E-01	1.2342E+01	6.2929E+01	8.1576E-03	4.7573E-02	**0.0000E+00**
*F*12	Mean	2.8939E+07	6.5470E+04	2.4349E+04	5.6933E+03	7.8642E+00	5.2017E-01	**2.4844E-01**
	Std	3.4110E+07	2.9057E+05	9.3788E+04	1.3911E+04	3.8324E+00	2.2828E-01	**1.1026E-01**
*F*13	Mean	7.1859E+07	3.2716E+05	3.4024E+05	5.2061E+05	3.0523E+00	2.7509E+00	**2.1616E+00**
	Std	3.8832E+07	1.7018E+06	9.1433E+05	4.1173E+05	5.9074E-01	**1.5493E-01**	2.4166E-01

Bold entries highlight the best results of the algorithm on the function

accuracy and strong robustness. The results reflect the global exploration by Brownian motion and the population diversity of generated offspring.

3.5 Convergence analysis

Convergence analysis can clearly comprehend local exploitation and global exploration process of the algorithm. Figure 7 shows the convergence curves of GA, DA, SCA, SFO, TSA, ChOA and SHO for partial test functions. As can be seen from the Fig. 7, SHO has the better parallel optimization capacity. The differences of local exploiting performance between different algorithms are presented on functions *F*3, *F*5 and *F*7, and SHO's optimization ability is the best in these functions. For the reason that SHO is impacted by the adaptive parameter α and the way of the spiral motion, which make SHO converge toward the optimal solution faster than other comparison algorithms, and subsequently re-exploit the optimization precision.

Table 8 Statistical results of fixed-dimension multimodal functions

Function		GA	SCA	DA	SFO	TSA	ChOA	SHO
*F*14	Mean	2.1700E+00	1.7964E+00	1.2623E+00	1.7962E+00	8.4309E+00	**1.0977E+00**	4.4619E+00
	Std	1.3724E+00	9.8476E-01	7.7636E-01	9.4877E-01	5.3333E+00	**3.9968E-01**	3.9493E+00
*F*15	Mean	7.9425E-03	1.0169E-03	3.8415E-03	2.1998E-03	1.2720E-02	1.3024E-03	**4.0466E-04**
	Std	5.2326E-03	3.5916E-04	6.7440E-03	2.1135E-03	2.2234E-02	**4.6318E-05**	2.4564E-04
*F*16	Mean	−9.9737E-01	**−1.0316E+00**	**−1.0316E+00**	**−1.0316E+00**	−1.0285E+00	**−1.0316E+00**	**−1.0316E+00**
	Std	3.2584E-02	6.5333E-05	7.0772E-07	3.9599E-07	9.6512E-03	1.0504E-05	**1.2086E-08**
*F*17	Mean	4.2081E-01	4.0065E-01	**3.9789E-01**	**3.9789E-01**	3.9794E-01	3.9891E-01	**3.9789E-01**
	Std	2.6939E-02	2.0337E-03	7.1726E-06	**4.4403E-07**	4.4015E-05	1.4105E-03	8.9400E-07
*F*18	Mean	3.7709E+00	3.0001E+00	**3.0000E+00**	**3.0000E+00**	1.2005E+01	3.0003E+00	**3.0000E+00**
	Std	9.9571E-01	9.6817E-05	8.8746E-07	3.9408E-05	1.7842E+01	3.5984E-04	**1.8208E-08**
*F*19	Mean	−3.8440E+00	−3.8542E+00	**−3.8626E+00**	−3.8625E+00	−3.8621E+00	−3.8549E+00	−3.8583E+00
	Std	2.0872E-02	4.0331E-03	**6.4762E-04**	1.4041E-03	1.7428E-03	1.9266E-03	3.9642E-03
*F*20	Mean	−2.9827E+00	−2.8417E+00	−3.2427E+00	−3.2350E+00	**−3.2661E+00**	−2.5865E+00	−3.0059E+00
	Std	1.4818E-01	4.0763E-01	9.9403E-02	**6.8387E-02**	6.9168E-02	4.6733E-01	2.7576E-01
*F*21	Mean	−2.1578E+00	−2.4888E+00	**−6.7452E+00**	−6.0621E+00	−6.6850E+00	−2.7528E+00	−5.9756E+00
	Std	**8.9209E-01**	1.8985E+00	2.6455E+00	3.4991E+00	3.2892E+00	2.0898E+00	3.3566E+00
*F*22	Mean	−2.5108E+00	−3.2342E+00	**−7.2770E+00**	−6.9284E+00	−6.5270E+00	−3.8763E+00	−6.3229E+00
	Std	**1.2125E+00**	2.0292E+00	3.0871E+00	3.5954E+00	3.6851E+00	1.8815E+00	2.9223E+00
*F*23	Mean	−2.4954E+00	−3.8475E+00	−7.1016E+00	**−7.5357E+00**	−6.9250E+00	−4.6818E+00	−6.3452E+00
	std	**8.8861E-01**	2.0526E+00	3.1919E+00	3.5948E+00	3.7597E+00	1.2935E+00	2.3590E+00

Bold entries highlight the best results of the algorithm on the function

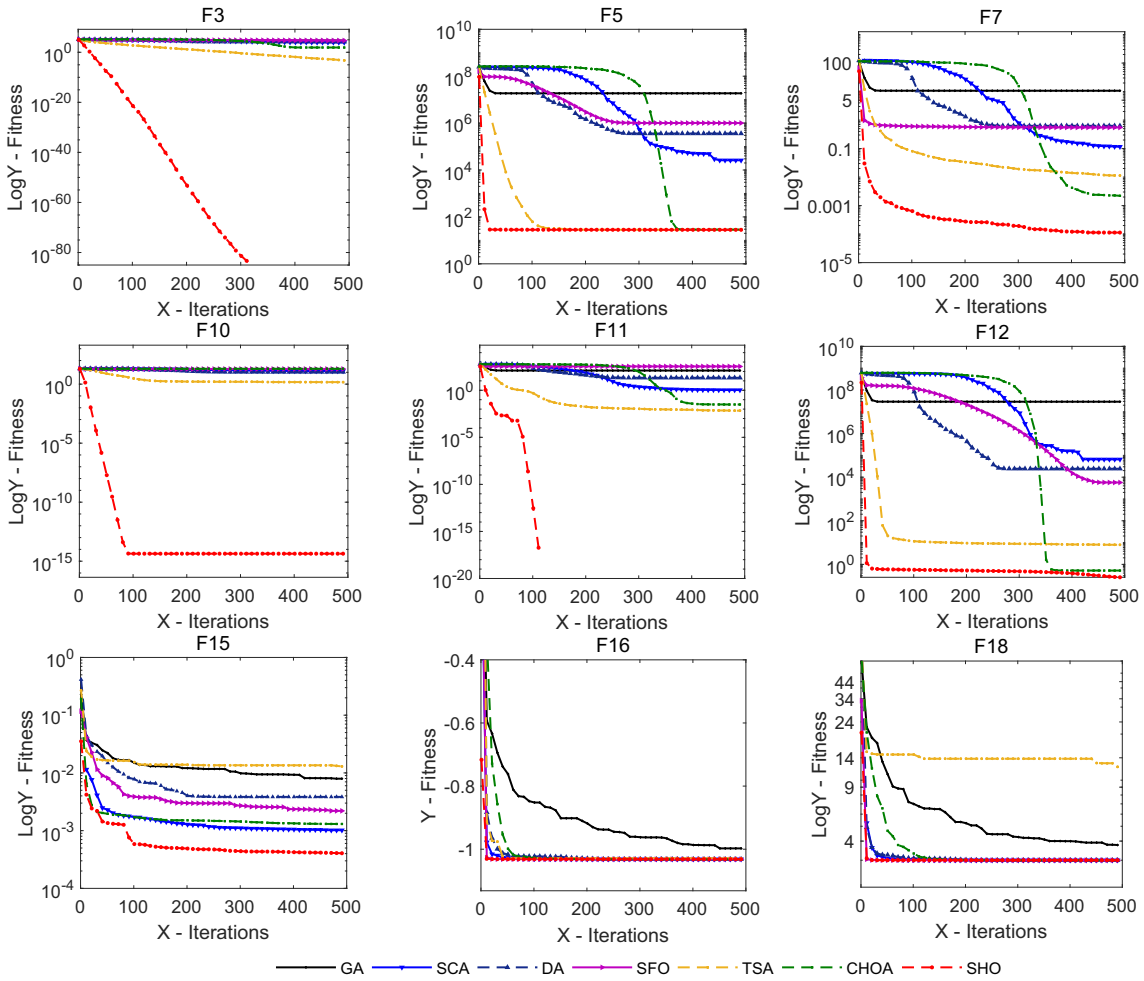


Fig. 7 Convergence analysis of SHO and comparison algorithms

However, other algorithms are defective in obtaining higher convergence accuracy under the same population size and iteration numbers. The convergence curves of multimodal functions show the global optimization potential of different algorithms. For function $F10$, SHO has the most obvious exploration performance. It firstly jumps out of the local optimal value and concentrates accurately near global optimums, while other algorithms converge slowly. It should be noted that on function $F11$, SHO achieves the global optimal value 0 during iterations. Since the figure shows an average fitness value in logarithmic terms, the curve breaks during iterations. For function $F12$, apart from the rapid convergence in early iterations, SHO can still be re-exploited in late iterations. For function $F15$, SHO shows outstanding optimization accuracy. Even if other algorithms, such as DA and SFO, can find the global optimal value, SHO has the fastest convergence speed on functions $F16$ and $F18$. Convergence curves of SHO on multimodal functions show that, sea horses tend to seek optimal solution in the whole search space through Brownian motion of floating action and offspring renewal in early iterations, while sea horses exploit precisely through spiral motion and the successful predation stage in later iterations. Therefore, convergence analysis proves that the proposed SHO is effective.

3.6 Performance analysis of SHO on CEC2014 benchmark functions

To further verify SHO's local extremum avoidance, the challenging CEC2014 benchmark functions are used to test compared with six well-known algorithms. The maximum number of iterations for all algorithms is set to 1000. CEC2014 benchmark functions are composed of basic functions via translation, rotation, or combination. They are more difficult to search for optimization in that their functions are fairly dynamic and complex with many local optimal values. Table 9 shows the test results of SHO and six comparison algorithms. It can be seen that SHO obtains better average fitness values on 21/30 functions. In terms of unimodal functions, SHO achieves the best Mean and Std results compared other algorithms on functions $CEC-1$ and $CEC-3$, while the Mean and Std results of DA are the best on functions $CEC-2$ and $CEC-4$. GA and SFO are inferior to that of other algorithms on unimodal functions.

Table 9 Results of the comparative methods on CEC2014 benchmark functions

Function		GA	SCA	DA	SFO	TSA	ChOA	SHO
CEC−1	Mean	7.1541E+08	4.3394E+08	3.2165E+08	1.0933E+09	3.4998E+08	5.9442E+08	**3.1313E+08**
	Std	3.0860E+08	1.1369E+08	1.9252E+08	2.4271E+08	2.2031E+08	1.2287E+08	**1.0601E+08**
CEC−2	Mean	2.5672E+10	2.6377E+10	**6.7457E+09**	5.9410E+10	3.1068E+10	4.4232E+10	2.9947E+10
	Std	8.2687E+09	4.6549E+09	**2.6005E+09**	1.1592E+10	9.0278E+09	6.5469E+09	7.1444E+09
CEC−3	Mean	1.3215E+05	6.0581E+04	1.5502E+05	8.6755E+04	5.1628E+04	8.2252E+04	**4.8301E+04**
	Std	4.7604E+04	1.2896E+04	5.3551E+04	1.0711E+04	1.0866E+04	**6.6033E+03**	8.7387E+03
CEC−4	Mean	4.3739E+03	2.3211E+03	**1.3531E+03**	1.0374E+04	3.6752E+03	3.4980E+03	2.5911E+03
	Std	2.4292E+03	6.0498E+02	**4.5493E+02**	2.5953E+03	2.6388E+03	1.2225E+03	8.0821E+02
CEC−5	Mean	5.2109E+02	5.2103E+02	5.2102E+02	5.2115E+02	5.2104E+02	5.2104E+02	**5.2061E+02**
	Std	1.0002E-01	6.1044E-02	7.7323E-02	5.1403E-02	5.5924E-02	**4.7479E-02**	1.2736E-01
CEC−6	Mean	6.3897E+02	6.3708E+02	6.3678E+02	6.4642E+02	6.3170E+02	6.3759E+02	**6.2975E+02**
	Std	2.2263E+00	2.5213E+00	4.0156E+00	**4.6314E-01**	3.1164E+00	2.0091E+00	2.3985E+00
CEC−7	Mean	9.7441E+02	9.2577E+02	**7.6438E+02**	1.2889E+03	9.7983E+02	1.1427E+03	9.6235E+02
	Std	7.7159E+01	4.5882E+01	**2.6599E+01**	9.3637E+01	7.1296E+01	7.3975E+01	7.3326E+01
CEC−8	Mean	1.0010E+03	1.0740E+03	1.0932E+03	1.2011E+03	1.0633E+03	1.0605E+03	**9.5955E+02**
	Std	3.7720E+01	2.3386E+01	5.2549E+01	**2.8822E+00**	3.9181E+01	1.9146E+01	3.1206E+01
CEC−9	Mean	1.2178E+03	1.2057E+03	1.2119E+03	1.3082E+03	1.2295E+03	1.1859E+03	**1.1159E+03**
	Std	5.3832E+01	**2.3683E+01**	5.4277E+01	9.3803E+00	6.0981E+01	2.5421E+01	2.9693E+01
CEC−10	Mean	5.0373E+03	7.6277E+03	7.1360E+03	8.8227E+03	6.5307E+03	7.5655E+03	**4.1719E+03**
	Std	9.5542E+02	6.4910E+02	7.8063E+02	**4.9624E+01**	7.8288E+02	9.7607E+02	8.1086E+02
CEC−11	Mean	8.6880E+03	8.7373E+03	7.6124E+03	1.0157E+04	7.1699E+03	8.9184E+03	**5.2581E+03**
	Std	7.0025E+02	2.7731E+02	7.3366E+02	2.3078E+02	7.4788E+02	**2.2181E+02**	6.6720E+02
CEC−12	Mean	1.2033E+03	1.2031E+03	1.2025E+03	1.2040E+03	1.2029E+03	1.2032E+03	**1.2009E+03**
	Std	6.8459E-01	3.7186E-01	6.1772E-01	5.3978E-01	3.7290E-01	3.2664E-01	**2.1770E-01**
CEC−13	Mean	1.3042E+03	1.3038E+03	**1.3015E+03**	1.3074E+03	1.3043E+03	1.3048E+03	1.3043E+03
	Std	8.5029E-01	**4.6392E-01**	8.8730E-01	9.3713E-01	1.0935E+00	6.5608E-01	7.6774E-01
CEC−14	Mean	1.4903E+03	1.4709E+03	**1.4226E+03**	1.6182E+03	1.4984E+03	1.5404E+03	1.4959E+03
	Std	3.2765E+01	1.3540E+01	**7.8765E+00**	2.7035E+01	3.6347E+01	3.1471E+01	2.2488E+01
CEC−15	Mean	3.2992E+05	1.8263E+04	**1.0148E+04**	9.0047E+04	2.4395E+04	1.1312E+05	1.9158E+04
	Std	3.4331E+05	1.5433E+04	**1.2038E+04**	3.6836E+04	4.3298E+04	1.0722E+05	1.9308E+04
CEC−16	Mean	1.6135E+03	1.6132E+03	1.6132E+03	1.6137E+03	1.6130E+03	1.6128E+03	**1.6121E+03**
	Std	3.6715E-01	2.6346E-01	3.0512E-01	**1.7746E-01**	3.1301E-01	2.5890E-01	3.6975E-01
CEC−17	Mean	5.2827E+07	1.3049E+07	1.4146E+07	4.2113E+07	1.3408E+07	3.4397E+07	**1.2123E+07**
	Std	3.0184E+07	**6.7566E+06**	1.3652E+07	2.0276E+07	1.1967E+07	1.9203E+07	8.7767E+06
CEC−18	Mean	1.3906E+09	3.2679E+08	**2.5360E+07**	1.2298E+09	5.8538E+08	8.5140E+08	3.8081E+07
	Std	6.0486E+08	1.8013E+08	6.2690E+07	6.5911E+08	1.1114E+09	1.1157E+09	**4.4161E+07**
CEC−19	Mean	2.1875E+03	2.0363E+03	**2.0010E+03**	2.1648E+03	2.0958E+03	2.1825E+03	2.0272E+03
	Std	9.3674E+01	**2.7907E+01**	7.5359E+01	5.8222E+01	1.1935E+02	1.2790E+02	5.0688E+01
CEC−20	Mean	2.8397E+05	4.6391E+04	1.6293E+05	2.3972E+05	5.2572E+04	1.1282E+05	**3.6221E+04**
	Std	3.2828E+05	2.6562E+04	1.4469E+05	1.8076E+05	6.3617E+04	4.9118E+04	**1.5493E+04**
CEC−21	Mean	1.8000E+07	3.6383E+06	4.6345E+06	1.4708E+07	7.0577E+06	1.2736E+07	**1.7666E+06**
	Std	1.1718E+07	2.8485E+06	5.8211E+06	1.0700E+07	1.1104E+07	**2.0673E+06**	2.2875E+06
CEC−22	Mean	3.7917E+03	3.2650E+03	3.3163E+03	4.1374E+03	3.5616E+03	3.1975E+03	**2.9631E+03**
	Std	3.0189E+02	**1.6814E+02**	2.7814E+02	4.6058E+02	8.6894E+02	1.8833E+02	2.5275E+02
CEC−23	Mean	2.8937E+03	2.7168E+03	2.7191E+03	2.9294E+03	2.7265E+03	2.7618E+03	**2.6754E+03**
	Std	1.2223E+02	**2.7187E+01**	4.5852E+01	1.2686E+02	9.3238E+01	5.3314E+01	8.4566E+01
CEC−24	Mean	2.7387E+03	2.6091E+03	2.6624E+03	2.6964E+03	2.6058E+03	2.6001E+03	**2.6000E+03**
	Std	2.7368E+01	9.5131E+00	9.6441E+00	1.6537E+01	1.7586E+01	4.8450E-02	**1.6553E-04**

Table 9 (continued)

Function		GA	SCA	DA	SFO	TSA	ChOA	SHO
CEC−25	Mean	2.7741E+03	2.7425E+03	2.7445E+03	2.7361E+03	2.7272E+03	2.7122E+03	**2.7000E+03**
	Std	1.9907E+01	1.0221E+01	1.6483E+01	9.0656E+00	7.4081E+00	1.3367E+01	**3.0447E-13**
CEC−26	Mean	2.7058E+03	**2.7035E+03**	2.7320E+03	2.7135E+03	2.7804E+03	2.7912E+03	2.7674E+03
	Std	1.6175E+00	**4.8931E-01**	4.6221E+01	5.7684E+00	5.6244E+01	5.3373E+01	4.6964E+01
CEC−27	Mean	3.8630E+03	3.8524E+03	3.7666E+03	3.9141E+03	3.7974E+03	3.9813E+03	**3.5917E+03**
	Std	1.6684E+02	2.5855E+02	3.9186E+02	2.1930E+02	3.1077E+02	**7.7434E+01**	3.0922E+02
CEC−28	Mean	6.4218E+03	**5.4804E+03**	6.6395E+03	8.9325E+03	7.5270E+03	5.7092E+03	5.6777E+03
	Std	1.1194E+03	3.5072E+02	1.0632E+03	6.0699E+02	9.8157E+02	**2.4924E+02**	5.3136E+02
CEC−29	Mean	4.1276E+07	3.0269E+07	8.0442E+07	2.9796E+08	6.3045E+07	5.3284E+07	**1.8706E+07**
	Std	3.9552E+07	**1.2792E+07**	7.5615E+07	9.0464E+07	4.6862E+07	3.5495E+07	2.2772E+07
CEC−30	Mean	6.9770E+05	5.2659E+05	4.7197E+05	2.0541E+06	5.0654E+05	7.9851E+05	**1.6934E+05**
	Std	4.2663E+05	1.6822E+05	3.7758E+05	1.0868E+06	5.2721E+05	1.8848E+05	**1.4944E+05**

Bold entries highlight the best results of the algorithm on the function

For multimodal functions *CEC* − 5, *CEC* − 6, *CEC* − 8, *CEC* − 10, *CEC* − 11, *CEC* − 12, and *CEC* − 15, SHO outperforms other algorithms with stronger exploration performance. Moreover, SHO has the minimum Std result on function *CEC* − 12. SHO also shows stronger optimization performance on most of hybrid functions. For example, SHO achieves the best Mean results on functions *CEC* − 17, *CEC* − 20, *CEC* − 21 and *CEC* − 22, and has the best Std result on function *CEC* − 20. Except functions *CEC* − 26 and *CEC* − 28, SHO's Mean results can obtain the best on the other composition functions, and its Std values are the minimum on functions *CEC* − 24 and *CEC* − 25. The test results on CEC2014 benchmark functions further show that SHO is more competitive than other algorithms.

Figure 8 shows the boxplots for seven algorithms on CEC2014 benchmark functions. If '+' has appeared outside the upper limit, it meant that the algorithm has a poor search accuracy in 30 runs. Otherwise, if '+' was outside the lower limit, it indicated that the algorithm can be fully explored and exploited to search for the superior precision value. It can be seen from the Fig. 8 analysis that SHO displays '+' out of the upper edge for a total of 3 times (functions *CEC* − 6, *CEC* − 10 and *CEC* − 20), which has the least number of occurrences compared with other algorithms. The medians of SHO are lower than that of the other six algorithms. Additionally, SHO has the smallest differences between upper and lower limits of functions *CEC* − 20 and *CEC* − 25. The boxplots verifies that SHO has strong robustness and stability. As described above, SHO algorithm shows superior optimization performance on CEC2014 benchmark functions.

3.7 Scalability analysis of SHO

This subsection evaluates the adaptability of SHO in high dimensional problems. In order to make compares more concise and effective, two groups of experiments were conducted by selecting representative test functions. The first set of functions came from *F*1, *F*2, *F*3, *F*12 and *F*13 of 23 well-known functions. They can test the local exploitation accuracy and global optimization ability of the algorithm, representing the traditional unimodal test functions. For the first group, dimensions were set to 10, 30, 50, 100, and 500. The second experiment was designed for high-dimensional optimization performance on CEC2014 benchmark functions. In the second set, simple multimodal functions *CEC* − 5, hybrid function *CEC* − 22, composition functions *CEC* − 24, *CEC* − 25 and *CEC* − 27 were selected. Because of the complexity of composition functions, they are more difficult to search for optimization in high dimensions. Using these functions evaluate the performance of the algorithm to deal with high-dimensional problems more closely to the difficulty of practical application problems, which make the scalability analysis more sufficient. Since the CEC2014 benchmark functions ran with variable dimensions only 10, 30, 50 and 100, these four dimensions were tested separately. SHO is compared with the above six comparison algorithms. Each algorithm was operated independently for 30 times, and the average fitness value is taken as the statistical indicator. The population size and maximum iteration times of group 1 and group 2 were set to 30 × 500 and 30 × 1000, respectively.

Figure 9 shows the change trends of log-average fitness values for seven algorithms in different dimensions. As can see from Fig. 9 that y values of all algorithms are in positive proportion to the increase of x values. This suggests that higher dimensions lead to problems solving more difficult. In the first set experiment, SHO is able to maintain the optimal average fitness value in the high-dimensional case for six functions, compared with other algorithms. The SHO algorithm is obviously superior to other comparison algorithms in all dimensions on functions *F*1, *F*3, and *F*4. On function

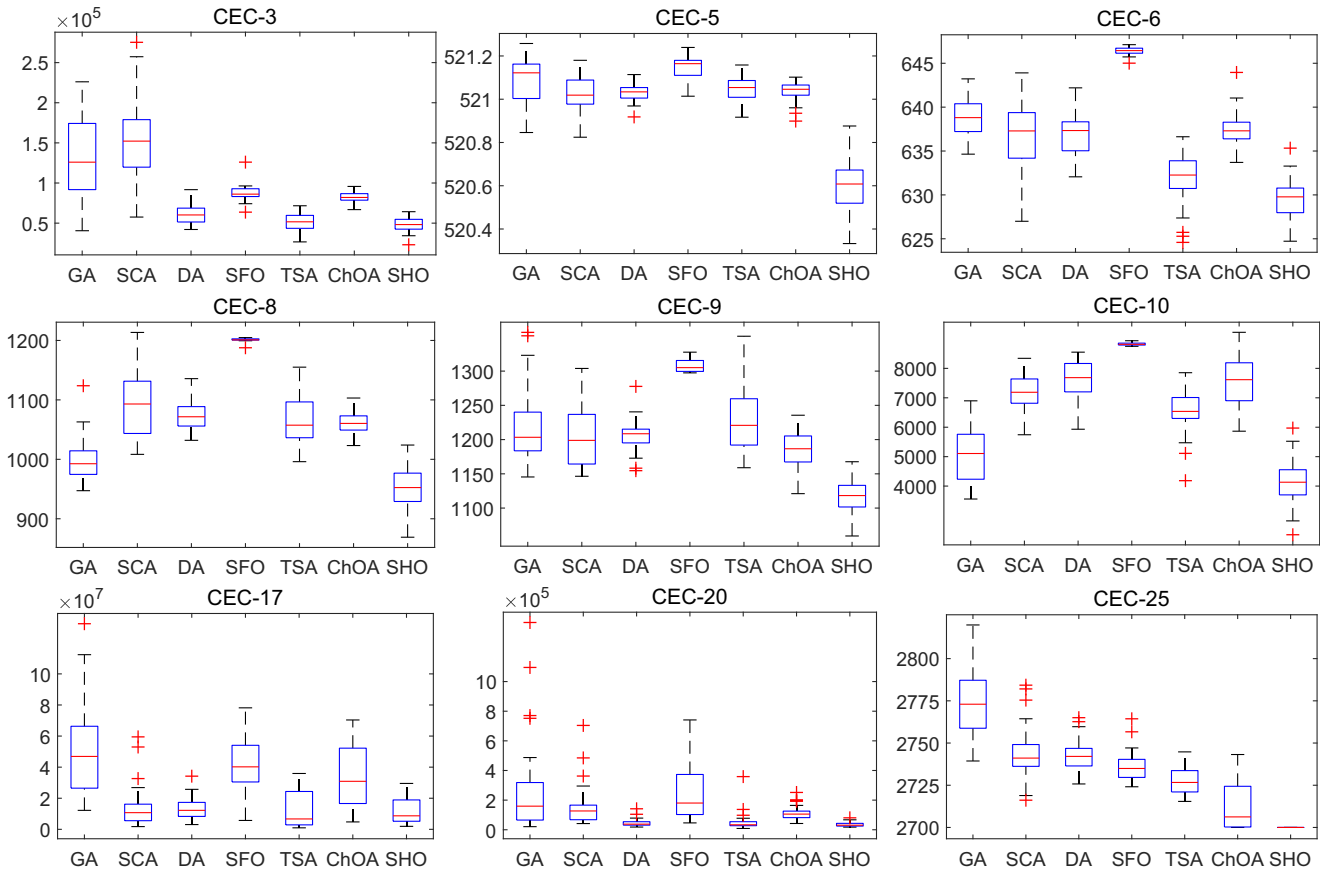


Fig. 8 Box plot of the proposed SHO and six comparison algorithms on CEC2014 benchmark functions

$F2$, the change amplitude of y value is basically the same when the seven algorithms transition from 10 dimension to 100 dimension. However, from 100 to 500 dimensions, GA and SFO algorithms change more, while other algorithms are more stable. The slope of SHO curve changes more stable than other algorithms, especially on functions $F10$ and $F12$ when the dimension transitions from 100 to 500. The first set of experimental results confirm the dimensional insensitivity of SHO. Simultaneously, SHO can accurately track the difficulty of problems caused by the increase of dimensions, and search for better accuracy. In the second set of the experiment, SHO remains optimal in each dimension of functions $CEC-5$, $CEC-12$ and $CEC-27$. For function $CEC-22$, the variation degree of SHO, DA and SCA have things in common. Even though DA is better than SHO when the dimension is 50, SHO is still optimal in the dimension 100. On function $CEC-24$, SHO and ChOA have similar searching accuracy in each dimension. For function $CEC-25$, seven algorithms do not change significantly when the dimension rises from 10 to 30, however, as the dimension is larger, SHO can still outperform other algorithms while keeping the original parallel optimization accuracy. High-dimensional test results show that SHO is a reliable algorithm that maintains an effective best-finding situation and adaptability in handling more complex high-dimensional problems.

3.8 Statistical analysis

In order to avoid accidental test results, statistical tests are performed by using the Wilcoxon rank-sum test statistical method [55] applying a significance level of 5%. Further this method is used to evaluate the effectiveness and superiority of SHO. SHO was used as the control algorithm, and pairwise comparison was made with the other six algorithms. If the p value generated by the two algorithms is less than 5%, it indicates that there is a significant difference between the two algorithms in statistical significance. Otherwise, the difference between the two algorithms is not obvious. Two groups of tests were conducted. The first group performed mann-Whitney U test of SHO on the first 13 well-known functions. The second group was a more powerful mann-Whitney U test on CEC2014 benchmark functions.

Tables 10 and 11 are the summary of test results, where '+' represents significant difference and '-' represents poor significant difference. In Table 10, SHO conducts 78 (13 × 6) groups of experiments, among which 74 groups of data shows significant differences. 157 of 180 (30 × 6) comparisons are '+' in Table 11. The results of the two groups verify that the optimization performance of the SHO

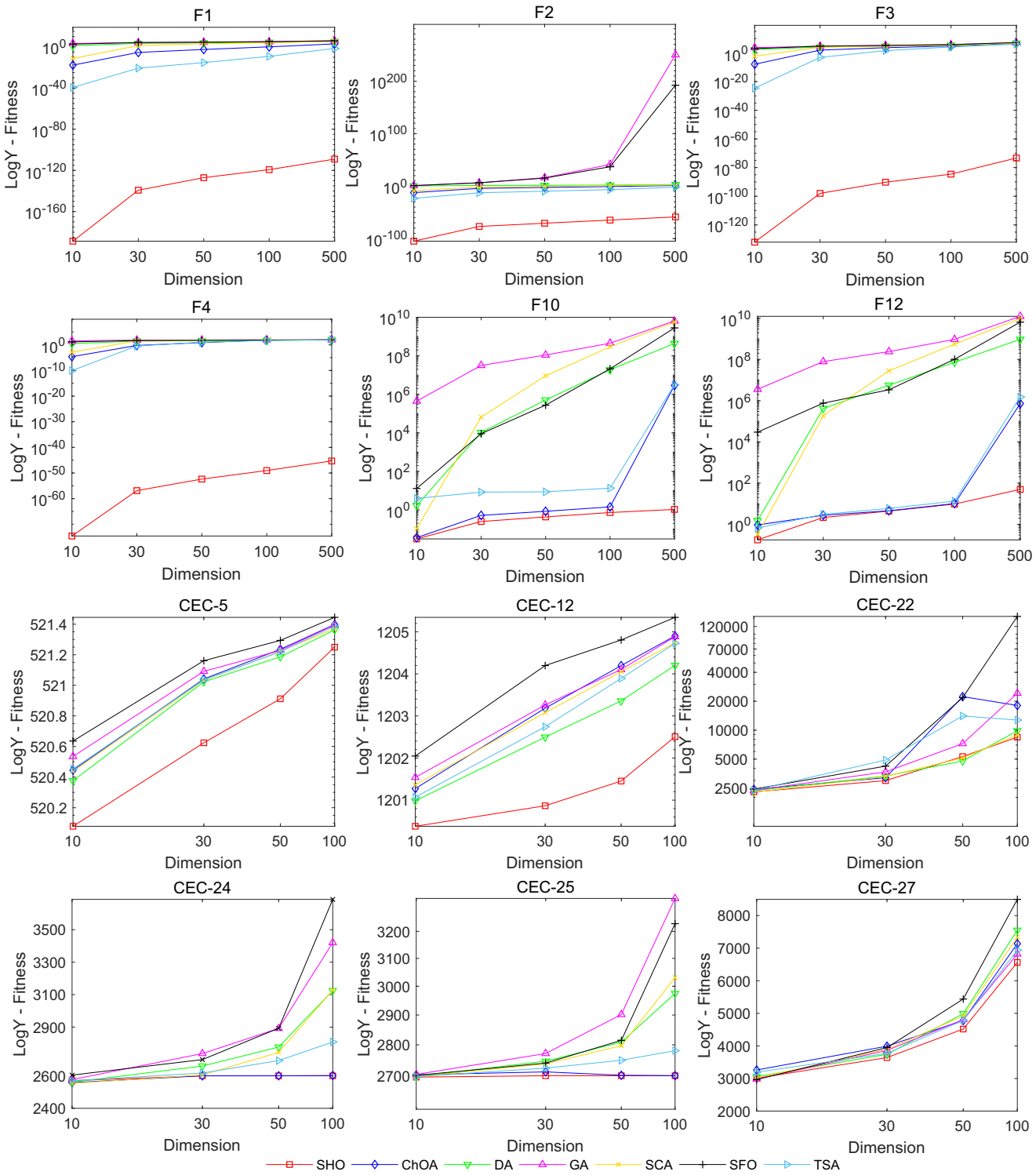


Fig. 9 Scalability analysis of different algorithms

algorithm is better than the other six comparison algorithms in the statistical sense.

In addition, Friedman rank test [56] is a nonparametric method that uses rank to implement significant differences for multiple population distributions. Friedman rank tests of seven algorithms were performed on the most challenging CEC2014 benchmark functions to compare their comprehensive average performance. Table 12 shows Friedman test results of seven algorithms. As is shown in Table 12, SHO ranked the first and is significantly better than the other six comparison algorithms. Thus, Friedman rank tests demonstrate that the proposed SHO is effective and stable.

Table 10 Wilcoxon rank-sum test results for SHO against other algorithms on 13 well-known functions

Function	SHO vs. GA		SHO vs. SCA		SHO vs. DA		SHO vs. SFO		SHO vs. TSA		SHO vs. ChOA	
	p value	win	p value	win	p value	win	p value	win	p value	win	p value	win
*F*1	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
*F*2	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
*F*3	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
*F*4	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
*F*5	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	1.86E-01	–	5.46E-09	+
*F*6	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	2.75E-03	+	6.79E-02	–
*F*7	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	9.92E-11	+
*F*8	1.96E-10	+	3.02E-11	+	1.37E-03	+	3.02E-11	+	2.61E-02	+	8.77E-02	–
*F*9	1.21E-12	+	1.21E-12	+	1.21E-12	+	1.21E-12	+	1.21E-12	+	1.21E-12	+
*F*10	1.72E-12	+	1.72E-12	+	1.72E-12	+	1.72E-12	+	1.72E-12	+	1.72E-12	+
*F*11	1.21E-12	+	1.21E-12	+	1.21E-12	+	1.21E-12	+	2.93E-05	+	1.21E-12	+
*F*12	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
*F*13	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	6.52E-09	–	3.02E-11	+
±.		13/0		13/0		13/0		13/0		11/2		11/2

'+' indicates significant difference and '-'indicates poor significant difference.

4 SHO for engineering design problems

In this section, the proposed SHO is applied to solve five real-world optimization problems, namely tension/compression spring design problem, reducer design problem, pressure vessel design problem, cantilever beam design problem, and welded beam design problem. Meanwhile, SHO is compared with some famous metaheuristic algorithms to evaluate its constraint programmability. Since real-world optimization problems have been constrained by inequalities or equalities, a simple method to deal with constraints was adopted that was the static penalty function. In this method, the solution is punished for violating the constraints, thus a constrained problem is converted into an unconstrained problem. For each real-world optimization problem, SHO was independently run for 30 times, as well as the statistical indicators consisted of maximum value (Worst), minimum value (Best), mean value (Mean) and standard deviation (Std) of 30 runs. The population size and the maximum number of iterations were set to 30 and 500.

4.1 Tension/compression spring design problem

In this case, it can define goals as minimizing the weight of the spring. Its specific design is shown in Fig. 10. There are three decision variables in this problem, which are wire diameter (d), mean coil diameter (D) and the number of active coils (N). The mathematical model is as follows.

$$\begin{aligned} &\vec{x} = [x_1, x_2, x_3] = [d, D, N] \\ &\min Z\left(\vec{x}\right) = (x_3 + 2)x_2{x_1}^2 \\ &s.t.\ g_1\left(\vec{x}\right) = 1 - \frac{x_2^3 x_3}{71785x_1^4} \le 0 \\ &g_2\left(\vec{x}\right) = \frac{4x_2^2 - x_1x_2}{12566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5108x_1^2} \le 0 \\ &g_3\left(\vec{x}\right) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0 \\ &g_4\left(\vec{x}\right) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \end{aligned} \tag{14}$$

In this case, the limits of decision variables are $0.05 \le x_1 \le 2.00$,$0.25 \le x_2 \le 1.30$ and $2.00 \le x_3 \le 15\ .00$. Presently, metaheuristic algorithms that have been successfully applied this problem including GA [57], CA [58], CPSO [59], WOA [38], GEO [28], SCA [60], HS [60], GWO [42], and AOA [38]. The proposed SHO algorithm is compared with these algorithms, and the optimal results obtained by each algorithm are shown in Table 13. As can be seen from Table 13, SHO can reach the minimum optimal value compared with ten algorithms. Table 14 shows the statistical results of all algorithms. Apparently, on the premise of less evolutions, SHO still has lower Mean and Std indicators than other algorithms. The results of the two tables prove that SHO has good applicability in this problem.

4.2 Reducer design problem

Reducer design problem is to design a simple gear box between the engine and propeller of the aircraft, so as to facilitate the normal operation of the propeller and engine. The specific schematic diagram of reducer design problem is shown in Fig. 11. The objective of this problem finds the combination that minimizes the total number of reducers,

Table 11 Wilcoxon rank-sum test results for SHO against other algorithms on CEC2014 benchmark functions

Function	SHO vs. GA		SHO vs. SCA		SHO vs. DA		SHO vs. SFO		SHO vs. TSA		SHO vs. ChOA	
	p value	win	p value	win	p value	win	p value	win	p value	win	p value	win
CEC−1	1.60E-07	+	1.91E-02	+	4.55E-01	–	3.02E-11	+	7.84E-01	–	2.61E-10	+
CEC−2	4.36E-02	+	2.61E-02	+	3.02E-11	+	9.92E-11	+	4.92E-01	–	8.48E-09	+
CEC−3	3.47E-10	+	2.77E-05	+	4.50E-11	+	3.34E-11	+	2.64E-01	–	3.02E-11	+
CEC−4	8.66E-05	+	2.90E-01	–	1.20E-08	+	4.08E-11	+	1.96E-01	–	1.86E-03	+
CEC−5	3.34E-11	+	3.02E-11	+	3.34E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
CEC−6	3.34E-11	+	1.61E-10	+	1.70E-08	+	3.02E-11	+	3.34E-03	+	3.69E-11	+
CEC−7	6.52E-01	–	3.39E-02	+	3.69E-11	+	3.69E-11	+	5.20E-01	–	1.29E-09	+
CEC−8	5.61E-05	+	3.02E-11	+	4.08E-11	+	3.02E-11	+	8.15E-11	+	3.69E-11	+
CEC−9	1.21E-10	+	3.34E-11	+	7.77E-09	+	3.02E-11	+	3.50E-09	+	5.07E-10	+
CEC−10	1.37E-03	+	3.02E-11	+	4.50E-11	+	3.02E-11	+	3.16E-10	+	4.08E-11	+
CEC−11	3.02E-11	+	3.02E-11	+	8.99E-11	+	3.02E-11	+	3.16E-10	+	3.02E-11	+
CEC−12	3.02E-11	+	3.02E-11	+	3.69E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
CEC−13	6.63E-01	–	1.08E-02	+	6.07E-11	+	3.69E-11	+	7.96E-01	–	9.47E-03	+
CEC−14	3.04E-01	–	6.10E-03	+	3.02E-11	+	3.02E-11	+	9.71E-01	–	7.60E-07	+
CEC−15	2.37E-10	+	5.59E-01	–	4.86E-03	+	8.89E-10	+	8.19E-01	–	2.67E-09	+
CEC−16	1.09E-10	+	3.34E-11	+	3.02E-11	+	3.02E-11	+	5.57E-10	+	2.03E-09	+
CEC−17	9.26E-09	+	3.18E-01	–	8.07E-01	–	2.39E-08	+	8.07E-01	–	4.74E-06	+
CEC−18	3.02E-11	+	2.37E-10	+	1.62E-01	–	3.02E-11	+	1.17E-05	+	4.08E-11	+
CEC−19	3.20E-09	+	1.22E-01	–	1.38E-02	+	1.41E-09	+	2.61E-02	+	6.01E-08	+
CEC−20	7.12E-09	+	7.29E-03	+	7.38E-10	+	1.46E-10	+	4.04E-01	–	3.82E-10	+
CEC−21	2.87E-10	+	2.60E-05	+	1.77E-03	+	1.55E-09	+	3.85E-03	+	4.50E-11	+
CEC−22	1.21E-10	+	5.61E-05	+	1.75E-05	+	9.92E-11	+	1.24E-03	+	3.56E-04	+
CEC−23	1.41E-09	+	4.22E-03	+	1.30E-03	+	5.56E-10	+	1.91E-02	+	8.83E-07	+
CEC−24	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+	3.02E-11	+
CEC−25	1.41E-11	+	1.41E-11	+	1.41E-11	+	1.41E-11	+	1.41E-11	+	1.41E-11	+
CEC−26	2.26E-02	+	7.01E-03	+	7.05E-01	–	2.64E-02	+	1.95E-05	+	2.08E-07	+
CEC−27	5.56E-04	+	4.44E-07	+	2.38E-03	+	9.52E-04	+	1.30E-03	+	3.16E-10	+
CEC−28	4.43E-03	+	1.02E-01	–	1.87E-05	+	3.02E-11	+	8.89E-10	+	6.20E-01	–
CEC−29	4.98E-04	+	1.11E-03	+	1.58E-04	+	3.02E-11	+	1.53E-05	+	5.46E-06	+
CEC−30	7.38E-10	+	1.17E-09	+	8.88E-06	+	4.50E-11	+	4.86E-03	+	1.61E-10	+
±		27/3		25/5		26/4		30/0		20/10		29/1

'+' indicates significant difference and '-' indicates poor significant difference.

Table 12 Friedman rank test results for SHO and other algorithms on CEC2014 benchmark functions

Algorithm	Friedman rank test	Rank
SCA	3.1333	2
GA	5.1000	6
DA	3.1667	3
TSA	3.9333	4
ChOA	4.5667	5
SFO	6.3667	7
SHO	1.7333	1

which involves constraints such as bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shafts. There are seven decision variables to control this problem, namely, face width (x_1), the module of teeth (x_2), number of teeth in the pinion (x_3), length of the first shaft between bearings (x_4), length of the second shaft between bearings (x_5), diameter of first (x_6) and diameter of the second shafts (x_7). The mathematical expression is as follows.

$$
\begin{aligned}
&\min Z = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - \\
&1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right) \\
&s.t.g_1(x) = \frac{27}{\left(x_1x_2^2x_3\right)} - 1 \le 0 \\
&g_2(x) = \frac{397.5}{\left(x_1x_2^2x_3^2\right)} - 1 \le 0 \\
&g_3(x) = \frac{1.93x_4^3}{\left(x_2x_3x_6^4\right)} - 1 \le 0 \\
&g_4(x) = \frac{1.93x_5^3}{\left(x_2x_3x_7^4\right)} - 1 \le 0 \\
&g_5(x) = \frac{1}{\left(110x_6^3\right)} \times \sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6} - 1 \le 0 \\
&g_6(x) = \frac{1}{\left(85x_7^3\right)} \times \sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6} - 1 \le 0 \\
&g_7(x) = \frac{x_2x_3}{40} - 1 \le 0 \\
&g_8(x) = 5\frac{x_2}{x_1} - 1 \le 0 \\
&g_9(x) = \frac{x_1}{12x_2} - 1 \le 0 \\
&g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0 \\
&g_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0
\end{aligned}
\tag{15}
$$

Range of decision variables: $2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28$
$7.3 \le x_4 \le 8.3, 7.3 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5.0 \le x_7 \le 5.5$

The proposed algorithm is compared with SC [61], GA [38], HS [60], SCA [60], GWO [42], WOA [42], HGSO [37] and AOA [38] previously applied, and the optimal results are shown in Table 15. According to Table 15 analysis, SHO is able to obtain the best solution compared with eight algorithms, which is obviously better than GA, SCA, and HS. Table 16 shows the statistical results of all algorithms. SHO can obtain the optimal value for Mean indicator under the precondition of fewer evolution times, but the shortcoming is that Std indicator has not reached the expected result.

Table 13 Optimal results of different algorithms for tension/compression spring design problem

Algorithms	d	D	N	Optimal value
GA [57]	0.051480	0.351661	11.632201	0.01270478
CA [58]	0.050000	0.317395	14.031795	0.012721
CPSO [59]	0.051728	0.357644	11.244543	0.012674
WOA [38]	0.0514	0.3513	11.6284	0.012695
GEO [28]	0.0518499	0.3605987	11.065069	0.0126658
SCA [60]	0.050780	0.334779	12.72269	0.012709667
HS [60]	0.05025	0.316351	15.23960	0.012776352
GWO [42]	0.0514	0.3503	11.6764	0.012669
AOA [38]	0.0508	0.3348	11.7020	0.012681
SHO	0.05194	0.36289	10.9358	**0.01266644**

Bold entries highlight the best result of the algorithm on this problem

4.3 Pressure vessel design

The primary objective of pressure vessel design minimizes the total cost of material forming and welding for cylindrical vessels. Figure 12 shows the design of pressure vessel design and the representation of parameters at the corresponding locations. This problem contains four decision variables, namely shell thickness (T_s), head thickness (T_h), entry radius (R), and length of cylindrical section without considering the head (L). The mathematical expression is described as follows.

$$
\begin{aligned}
&\vec{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L] \\
&minz\left(\vec{x}\right) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + \\
&\qquad 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
&s.t.g_1\left(\vec{x}\right) = -x_1 + 0.0193x_3 \le 0 \\
&\qquad g_2\left(\vec{x}\right) = -x_2 + 0.00954x_3 \le 0 \\
&\qquad g_3\left(\vec{x}\right) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \\
&\qquad g_4\left(\vec{x}\right) = x_4 - 240 \le 0 \\
&0 \le x_1 \le 990 \le x_2 \le 9910 \le x_3 \le 20010 \le x_4 \le 200
\end{aligned}
\tag{16}
$$

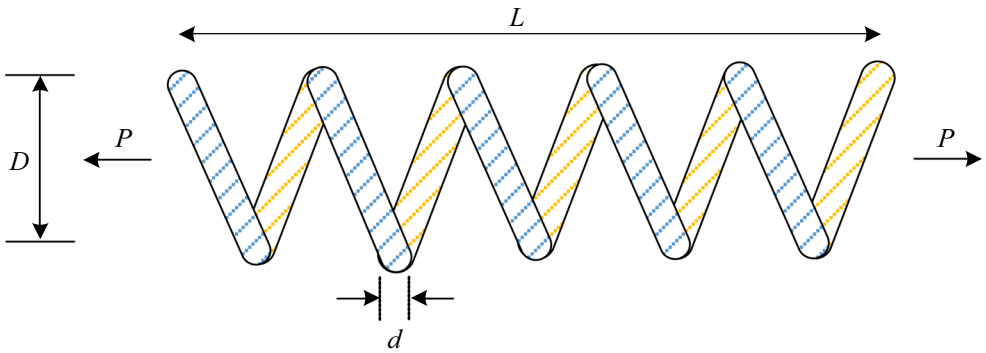


Fig. 10 Tension/compression spring design problem

Table 14 Statistical results of different algorithms for tension/compression spring design problem

Algorithms	Best	Mean	Worst	Std	Eval
GA [57]	0.012704	0.012769	0.012822	3.93E-05	/
CA [58]	0.012721	0.013568	0.0151156	8.4E-04	50,000
CPSO [59]	0.012674	0.012730	0.012924	5.19E-05	200,000
WOA [38]	0.012683	0.014709	0.017211	2.30E-03	30,000
GEO [28]	/	/	/	/	/
SCA [60]	0.012709667	0.012839637	0.012998448	7.8E-05	30,000
HS [60]	0.012776352	0.013069872	0.015214230	0.000375	30,000
GWO [42]	0.012669	0.013037	/	1.254E-03	20,000
AOA [38]	0.012681	0.013369	0.015625	7.44E-04	30,000
SHO	0.01266644	0.01282720	0.014428	3.5556E-04	15,000

'/' indicates that no corresponding work has been done in the literature.

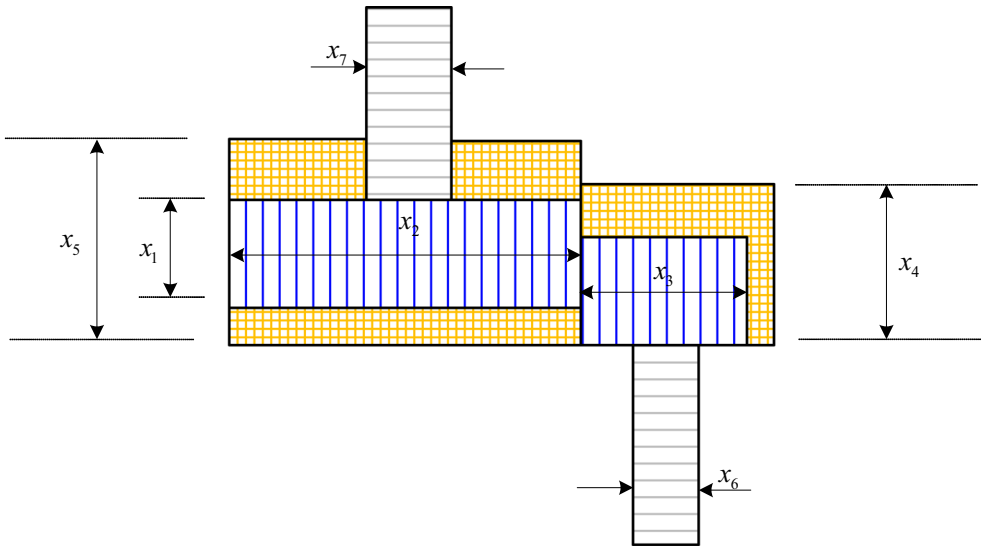


Fig. 11 Speed reducer design problem

Table 15 Optimal results of different algorithms for speed reducer design problem

Algorithms	x_1	x_2	x_3	x_4	x_5	x_6	x_7	Optimal value
SC [61]	3.50000	0.70000	17	7.327602	7.715321	3.350267	5.286655	2994.744241
GA [38]	3.5592	0.7133	19.659	7.9365	8.0197	3.6719	5.3276	3730
HS [60]	3.520124	0.7	17	8.37	7.8	3.366970	5.288719	3029.002
SCA [60]	3.508755	0.7	17	7.3	7.8	3.461020	5.289213	3030.563
GWO [42]	3.5000	0.70	17.00	7.38	7.81	3.3504	5.2867	2998.0976
WOA [42]	3.5000	0.70	17.00	8.03	7.91	3.3600	5.2850	3006.8794
HGSO [37]	3.4970	0.7100	17.020	7.6700	7.8100	3.3600	5.2850	2997.10
AOA [38]	3.50384	0.7	17	7.3	7.72933	3.35649	5.2867	2997.9157
SHO	3.5000	0.7000	17	7.3000	7.7163	3.3502	5.2867	**2994.504**

Bold entries highlight the best result of the algorithm on this problem

Table 16 Statistical results of different algorithms for speed reducer design

Algorithms	Best	Mean	Worst	Std	Eval
SC [61]	2994.7442410	3009.9647360	3001.7582640	4	54,456
GA [38]	3730	8140	17,300	4.15E+03	30,000
HS [60]	3029.002	3295.329	3619.465	5.70235E+01	30,000
SCA [60]	3030.563	3065.917	3104.779	1.80742E+01	30,000
GWO [42]	2998.0976	3003.0686	/	3.1431E+00	20,000
WOA [42]	3006.8794	3032.2744	/	2.6815E+01	20,000
HGSO [37]	2997.1	2996.4	2996.9	4.39E-05	30,000
AOA [38]	3000	3000	3000	1.22E-12	30,000
SHO	2994.504	2998.257	3025.619	5.941579E+00	15,000

Table 17 displays the optimal results of SHO and comparison algorithms such as CPSO [59], SMA [27], GWO [60], WOA [38], HHO [23], AEO [62], SCA [60], MPA [26], EO [39], AOA [38], and AO [63] for solving this problem. It can be detected from the Table 17 that the proposed SHO algorithm is superior to elven algorithms. It is noted that the comparison results are more obvious for MFO, WOA, and CPSO. In addition, it is comparable to new algorithms MPA, AOA, and AO. Table 18 shows the statistical results of these algorithms. SHO achieves the best Mean indicator. This result indicates that SHO can replace some traditional algorithms to solve this problem well after improving the number of evolutions.

4.4 Cantilever beam design problem

The purpose of this problem is to discover the optimal weight of the cantilever arm with setting an upper limit on the vertical displacement of the free end. Figure 13 exhibits a concrete illustration of a cantilever beam. The cantilever beam consists of five elements, each of which is a hollow section with a constant thickness. There are five decision variables and one constraint in the case. The mathematical expression is as follows.

$$\begin{aligned} &\min Z(x) = 0.0624 \times (x_1 + x_2 + x_3 + x_4 + x_5) \\ &s.t. \quad g(x) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0 \\ &0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100 \end{aligned} \tag{17}$$

Table 19 lists the optimal results of SHO, MA [64], GCA_I [64], GCA_II [64], and MFO [65]. SHO outperforms four comparison algorithms. Table 20 shows the statistical results of these algorithms.

4.5 Welded beam design

The design cost of welding beam is minimized under the constraint of weld shear stress (τ), bending stress (σ) in the beam, buckling load (P_c) on the rod and deflection

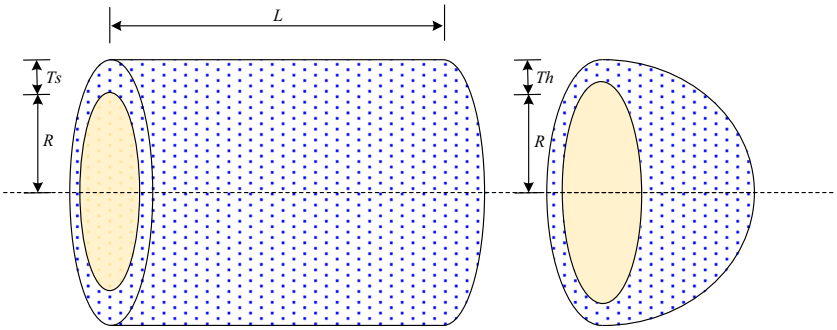


Fig. 12 Pressure vessel design

Table 17 Optimal results of different algorithms for pressure vessel design

Algorithms	T_s	T_h	R	L	Optimal value
CPSO [59]	0.812500	0.437500	42.091266	176.746500	6061.0777
SMA [27]	0.7931	0.3932	40.6711	196.2178	5994.1857
GWO [60]	0.779035	0.384660	40.327793	199.65029	5889.3689
WOA [38]	0.9730	0.6512	50.6804	93.0377	7.11E+03
HHO [23]	0.9833	0.4758	49.9297	98.9036	6.39E+03
AEO [62]	0.8374205	0.413937	43.389597	161.268592	5994.50695
SCA [60]	0.817577	0.417932	41.74939	183.57270	6137.3724
MPA [26]	0.8125	0.4375	42.098445	176.636607	6059.7144
EO [39]	0.8125	0.4375	42.0984456	176.6365958	6059.7143
AOA [38]	0.8303737	0.4162057	42.75127	169.3454	6048.7844
AO [63]	1.0540	0.182806	59.6219	38.8050	5949.2258
SHO	0.7782	0.3847	40.3223	199.9623	**5885.4926**

Bold entries highlight the best result of the algorithm on this problem

Table 18 Statistical results of different algorithms for pressure vessel design

Algorithms	Best	Mean	Worst	Std	Eval
CPSO [59]	6061.0777	6147.1332	6363.8041	8.645E+01	240,000
SMA [27]	5994.1857	/	/	/	/
GWO [60]	5889.3689	5891.5247	5894.6238	1.3910E+01	30,000
WOA [38]	7.11E+03	1.05E+04	1.35E+04	3.23E+03	30,000
HHO [23]	6.39E+03	6.61E+03	6.89E+03	2.54E+02	30,000
AEO [62]	59,945.0695	6136.3019	6820.8007	1.612901E+02	30,000
SCA [60]	6137.3724	6326.7606	6512.3541	1.26609E+02	30,000
MPA [26]	6059.7144	6102.8271	6410.0929	1.0661E+02	25,000
EO [39]	6059.7143	6668.114	7544.4925	5.6624E+02	15,000
AOA [38]	5.90E+03	6.52E+03	6.60E+03	4.31E+02	30,000
AO [63]	5949.2258	/	/	/	25,000
SHO	**5885.4926**	6335.809	7319.365	5.868E+02	15,000

Bold entries highlight the best result of the algorithm on this problem

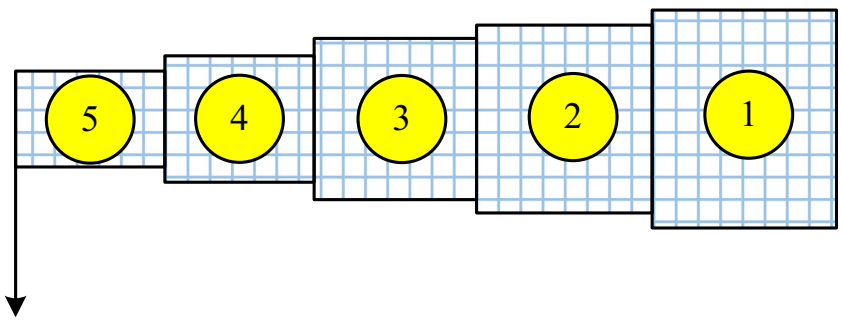


Fig. 13 Cantilever beam design problem

Table 19 Optimal results of different algorithms for cantilever beam design problem

Algorithms	x_1	x_2	x_3	x_4	x_5	Optimal value
MMA [64]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA_I [64]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA_II [64]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
MFO [65]	5.984871	5.316726	4.497332	3.513616	2.1616200	1.339988
SHO	6.0049	5.3227	4.4737	3.5065	2.16637	**1.339987**

Bold entries highlight the best result of the algorithm on this problem

Table 20 Statistical results of different algorithms for cantilever beam design problem

Algorithms	Best	Mean	Worst	Std	Eval
MMA [64]	1.3400	/	/	/	/
GCA_I [64]	1.3400	/	/	/	/
GCA_II [64]	1.3400	/	/	/	/
MFO [65]	1.339988	/	/	/	/
SHO	1.339987	1.341450	1.348049	1.685306E-03	15,000

of the beam end (δ). There are four variables that determine this problem, namely welding thickness (h), length of bar attachment (l), bar height (t) and bar thickness (b) (Fig. 14).

$$\begin{aligned}
&\overrightarrow{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b]\\
&minZ\left(\overrightarrow{x}\right) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)\\
&s.t.g_1\left(\overrightarrow{x}\right) = \tau\left(\overrightarrow{x}\right) - 13600 \le 0\\
&g_2\left(\overrightarrow{x}\right) = \sigma\left(\overrightarrow{x}\right) - 30000 \le 0\\
&g_3\left(\overrightarrow{x}\right) = \delta\left(\overrightarrow{x}\right) - 0.25 \le 0\\
&g_4\left(\overrightarrow{x}\right) = x_1 - x_4 \le 0\\
&g_5\left(\overrightarrow{x}\right) = P - P_c\left(\overrightarrow{x}\right) \le 0\\
&{}_6g\left(\overrightarrow{x}\right) = 0.125 - x_1 \le 0
\end{aligned} \tag{18}$$

$$g_7\left(\overrightarrow{x}\right) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) \quad -5.0 \le 0$$

$$0.1 \le x_1 \le 20.1 \le x_2 \le 100.1 \le x_3 \le 100.1 \le x_4 \le 2$$

$$\text{where } \tau\left(\overrightarrow{x}\right) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \tau' = \frac{6000}{\sqrt{2}x_1}x_2,$$

$$\tau'' = \frac{MR}{J}, M = 6000\left(14 + \frac{x_2}{2}\right), R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^x}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \sigma\left(\overrightarrow{x}\right) = \frac{504000}{x_4x_3^2}$$

$$\delta\left(\overrightarrow{x}\right) = \frac{65856000}{(30 \times 10^6)x_3^2x_4}$$

$$P_c\left(\overrightarrow{x}\right) = \frac{4.013(30 \times 10^6)\sqrt{\frac{x_3^2x_4^6}{36}}}{196}\left(1 - \frac{x_3}{28}\sqrt{\frac{30 \times 10^6}{4(12 \times 10^6)}}\right)$$

Table 21 shows the optimal solutions of SHO, HHO [23], CPSO [59], GWO [60], SCA [60], WOA [38], GEO [28] and MVO [60] algorithms for tackling this problem. Among the eight algorithms, SHO has the better design cost, which confirms that SHO has higher optimization ability. The statistical results of these algorithms are given in Table 22. It can be seen that SHO has better Mean and Std indicators. Hence, these results prove that SHO can achieve ideal results with other algorithms at a lower computational cost.

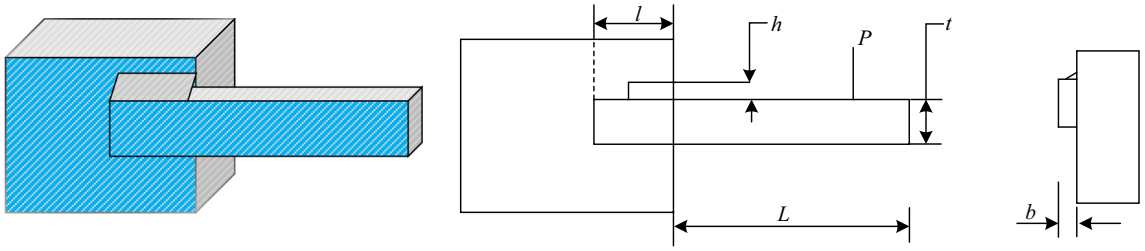


Fig. 14 Welded beam design problem

Table 21 Optimal results of different algorithms for welded beam design

Algorithms	h	l	t	b	Optimal value
HHO [23]	0.2134	3.5601	8.4629	0.2346	1.8561
CPSO [59]	0.202369	3.544214	9.048210	0.205723	1.728024
GWO [60]	0.205678	3.475403	9.036964	0.206229	1.726995
SCA [60]	0.204695	3.536291	9.004290	0.210025	1.759173
WOA [38]	0.329	2.5471	6.8078	0.3789	2.3584
GEO [28]	0.2443688	3.0630204	8.2914827	0.2443689	1.8653598
MVO [60]	0.205611	3.472103	9.040931	0.205709	**1.725472**
SHO	0.20585	3.46946	9.03276	0.20591	1.7259

Table 22 Statistical results of different algorithms for welded beam design problem

Algorithms	Best	Mean	Worst	Std	Eval
HHO [23]	1.8561	1.9302	1.9759	6.47E-02	30,000
CPSO [59]	1.728024	1.748831	1.782143	1.2926E-02	200,000
GWO [60]	1.726995	1.727128	1.727564	1.157E-03	30,000
SCA [60]	1.759173	1.817657	1.873408	2.7543E-02	30,000
WOA [38]	2.3584	2.5685	2.7862	2.14E-01	30,000
GEO [28]	1.8653598	/	/	/	50,000
MVO [60]	**1.725472**	1.729680	1.741651	4.866E-03	30,000
SHO	1.7259	1.7699	2.0029	6.055328E-02	15,000

5 Conclusion

In this paper, a novel sea horse optimizer (SHO) was presented based on natural heuristics. The proposed SHO algorithm simulates three kinds of intelligent behaviors of sea horses, which are feeding, male reproduction, and movement. Firstly, SHO employs the logarithmic helical equation and Levy flight to mathematically express and construct for the spiral floating. The aim is to make sea horses move randomly with large span step size and improve the local exploitation of SHO. Meanwhile, Brownian motion is used to explore the search space more comprehensively. Then, the speed between the sea horse and the prey are set according to the probability to express whether the predation success or failure. In order to regulate constantly the search neighborhoods of the proposed SHO algorithm, adaptive parameter is introduced into the predation behavior. Finally, based on the new generated population by following the first two behaviors, offspring are bred, which inherit the good genetic characteristics from their fathers and increase the diversity of individuals in population.

Qualitative experiments analyzed the influence of different behaviors of sea horses on different stages of the proposed SHO algorithm from the search history, the trajectory of 1st sea horse, average fitness of all sea horses and convergence curve. 23 well-known functions were selected to verify the local exploitation accuracy and global exploration ability of algorithms. Meanwhile, CEC2014 benchmark functions were used to test the local extreme value avoidance of the algorithm and the effectiveness of SHO in the high-dimensional case of the two sets of test functions.

Experimental results show that SHO is significantly superior to six state-of-the-art comparison algorithms on seven unimodal functions and most of the multimodal functions. For CEC2014 benchmark functions, SHO also has stronger local extremum avoidance than the other six algorithms. Simultaneously, SHO has fast convergence speed and good local extremum avoidance proved by convergence analysis. It can still keep high convergence accuracy in higher dimensions. The results of Wilcoxon rank sum test and Friedman rank test prove the superiority of SHO on most test functions. Finally, the application of SHO in several practical engineering problems verify that SHO has high optimization ability and low computational cost, which can replace some traditional metaheuristics or certain new proposed algorithms in recent years.

In the future, SHO can be applied in a wider range of fields, such as the hyper-parameter optimization of extreme learning machines and the intelligent solving of complex optimization problems. In addition, multi-objective and binary versions of SHO can be further developed to address multi-objective and discrete optimization problems.

Acknowledgements This work was supported in part by the Basic Research Foundation of Liaoning Educational Committee (Grant No. LJ2019JL017), the Scientific Research Foundation for Doctors, the China Postdoctoral Science Foundation (Grant No. 2021 M701537), the Scientific Research Foundation for Doctors, Department of Science & Technology of Liaoning Province (Grant No. 2019-BS-118).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Saha C, Das S, Pal K, Mukherjee S (2014) A fuzzy rule-based penalty function approach for constrained evolutionary optimization. IEEE Trans Cybern 46(12):2953–2965
2. Spall JC (2005) Introduction to stochastic search and optimization: estimation, simulation, and control, vol. 65. Wiley, New York
3. Hoos HH, Stützle T (2004) Stochastic local search: foundations and applications. Elsevier, Amsterdam
4. Alweshah M (2021) Solving feature selection problems by combining mutation and crossover operations with the monarch butterfly optimization algorithm. Appl Intell 51(6):4058–4081
5. Prencipe LP, Marinelli M (2021) A novel mathematical formulation for solving the dynamic and discrete berth allocation problem by using the bee Colony optimisation algorithm. Appl Intell 51(7): 4127–4142
6. Goodarzian F, Kumar V, Ghasemi P (2021) A set of efficient heuristics and meta-heuristics to solve a multi-objective pharmaceutical supply chain network. Comput Ind Eng 158:107389
7. Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
8. Glover F (1989) Tabu search—part I. ORSA J Comput 1(3):190–206
9. Back T (1996) Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford University Press, New York
10. Fan Q, Huang H, Li Y, Han Z, Hu Y, Huang D (2021) Beetle antenna strategy based grey wolf optimization. Expert Syst Appl 165:113882
11. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–73
12. Price KV (2013) Differential evolution. In handbook of optimization (pp 187-214). Springer, Berlin, Heidelberg
13. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102
14. Tinkle DW, Wilbur HM, Tilley SG (1970) Evolutionary strategies in lizard reproduction. Evolution 24(1):55–74
15. Kumar A, Rathore PS, Díaz VG, Agrawal R (Eds.) (2020) Swarm intelligence optimization: algorithms and applications. Wiley, New York
16. Kennedy J, Eberhart R (1995) Particle swarm optimization. In proceedings of ICNN'95-international conference on neural networks, vol 4. IEEE, pp 1942–1948
17. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
18. Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98
19. 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
20. Askarzadeh A (2016) A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct 169:1–12
21. Jain M, Singh V, Rani A (2019) A novel nature-inspired algorithm for optimization: squirrel search algorithm. Swarm Evol Comput 44:148–175
22. Dhiman G, Kumar V (2019) Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems. Knowl-Based Syst 165:169–196
23. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Future Gener Comput Syst 97:849–872
24. Khishe M, Mosavi MR (2020) Chimp optimization algorithm. Expert Syst Appl 149:113338
25. Kaur S, Awasthi LK, Sangal AL, Dhiman G (2020) Tunicate swarm algorithm: a new bio-inspired based metaheuristic paradigm for global optimization. Eng Appl Artif Intell 90:103541
26. Faramarzi A, Heidarinejad M, Mirjalili S, Gandomi AH (2020) Marine predators algorithm: a nature-inspired metaheuristic. Expert Syst Appl 152:113377
27. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: a new method for stochastic optimization. Future Gener Comput Syst 111:300–323
28. Mohammadi-Balani A, Nayeri MD, Azar A, Taghizadeh-Yazdi M (2021) Golden eagle optimizer: a nature-inspired metaheuristic algorithm. Comput Ind Eng 152:107050
29. Yang X-S (2012) Flower pollination algorithm for global optimization. In international conference on unconventional computing and natural computation (pp 240-249). Springer, Berlin, Heidelberg
30. Gomes GF, da Cunha SS, Ancelotti AC (2019) A sunflower optimization (SFO) algorithm applied to damage identification on laminated composite plates. Eng Comput 35(2):619–626
31. Ahmadi SA (2017) Human behavior-based optimization: a novel metaheuristic approach to solve complex optimization problems. Neural Comput Appl 28(1):233–244
32. Chou J-S, Nguyen N-M (2020) FBI inspired meta-optimization. Appl Soft Comput 93:106339
33. Askari Q, Younas I, Saeed M (2020) Political optimizer: a novel socio-inspired meta-heuristic for global optimization. Knowl-Based Syst 195:105709
34. Zhang Y, Jin Z (2020) Group teaching optimization algorithm: a novel metaheuristic method for solving global optimization problems. Expert Syst Appl 148:113246
35. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513
36. Yadav A (2019) AEFA: artificial electric field algorithm for global optimization. Swarm Evol Comput 48:93–108
37. Hashim FA, Houssein EH, Mabrouk MS, Al-Atabany W, Mirjalili S (2019) Henry gas solubility optimization: a novel physics-based algorithm. Future Gener Comput Syst 101:646–667
38. Hashim FA, Hussain K, Houssein EH, Mabrouk MS, Al-Atabany W (2021) Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems. Appl Intell 51(3): 1531–1551
39. Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S (2020) Equilibrium optimizer: a novel optimization algorithm. Knowl-Based Syst 191:105190
40. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
41. Cuevas E, Galvez J (2019) An optimization algorithm guided by a machine learning approach. Int J Mach Learn Cyb 10(11):2963–2991
42. Ahmadianfar I, Bozorg-Haddad O, Chu X (2020) Gradient-based optimizer: a new metaheuristic optimization algorithm. Inf Sci 540: 131–159
43. Ahmadianfar I, Heidari AA, Gandomi AH, Chu X, Chen H (2021) RUN beyond the metaphor: an efficient optimization algorithm based on Runge Kutta method. Expert Syst Appl 181:115079
44. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
45. Martin-Smith KM, Vincent AC (2006) Exploitation and trade of Australian seahorses, pipehorses, sea dragons and pipefishes (family Syngnathidae). Oryx 40(2):141–151
46. Kuiter RH (2000) Seahorses, pipefishes and their relatives: a comprehensive guide to Syngnathiformes. TMC Publishing, Chorleywood
47. Kuiter RH (2001) Revision of the Australian seahorses of the genus Hippocampus (Syngnathiformes: Syngnathidae) with descriptions of nine new species. Rec Aust Mus 53(3):293–340
48. Leysen H, Roos G, Adriaens D (2011) Morphological variation in head shape of pipefishes and seahorses in relation to snout length and developmental growth. J Morphol 272(10):1259–1270

49. Roos G, Van Wassenbergh S, Herrel A, Adriaens D, Aerts P (2010) Snout allometry in seahorses: insights on optimisation of pivot feeding performance during ontogeny. J Exp Biol 213(13):2184–2193
50. Kendrick AJ, Hyndes GA (2005) Variations in the dietary compositions of morphologically diverse syngnathid fishes. Environ Biol Fish 72(4):415–427
51. Porter MM, Adriaens D, Hatton RL, Meyers MA, McKittrick J (2015) Why the seahorse tail is square. Sci 349(6243):aaa6683
52. Mantegna RN (1994) Fast, accurate algorithm for numerical simulation of levy stable stochastic processes. Phys Rev E 49(5):4677–4683
53. Einstein A (1956) Investigations on the theory of the Brownian movement. Dover, New York
54. Liang J-J, Qu B-Y, Suganthan PN (2013) Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore 635:490
55. Wilcoxon F (1992) Individual comparisons by ranking methods. In breakthroughs in statistics (pp 196-202). Springer, New York, NY
56. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18
57. Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41(2):113–127
58. Coello Coello CA, Becerra RL (2004) Efficient evolutionary optimization through the use of a cultural algorithm. Eng Optimiz 36(2):219–236
59. He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20(1):89–99
60. Dhiman G, Kumar V (2017) Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 114:48–70
61. Ray T, Liew KM (2003) Society and civilization: an optimization algorithm based on the simulation of social behavior. IEEE Trans Evol Comput 7(4):386–396
62. Zhao W, Wang L, Zhang Z (2020) Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm. Neural Comput Appl 32(13):9383–9425
63. Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-qaness MA, Gandomi AH (2021) Aquila optimizer: a novel meta-heuristic optimization algorithm. Comput Ind Eng 157:107250
64. Chickermane HEMIANT, Gea HC (1996) Structural optimization using a new local approximation method. Int J Numer Meth Eng 39(5):829–846
65. Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst 89:228–249