Available online at www.sciencedirect.com
ScienceDirect
Mathematics and Computers in Simulation 213 (2023) 394–417
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Original articles

Special Forces Algorithm: A novel meta-heuristic method for global optimization

Wei Zhang*, Ke Pan*, Shigang Li, Yagang Wang

Shanghai Key Lab of Modern Optical System, and Engineering Research Center of Optical Instrument and System, Ministry of Education, School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, No. 516 Jun Gong Road, Shanghai 200093, China

Received 27 September 2022; received in revised form 29 May 2023; accepted 17 June 2023
Available online 24 June 2023

Abstract

This paper proposes a novel swarm intelligence optimizer inspired by human behavior called the Special Forces Algorithm (SFA). SFA is inspired by the missions of modern Special Forces in real life. First of all, in the initial stage of algorithm iteration, the speed of light mechanism is proposed for simulating the UAV-assisted search of special forces, and the loss probability mechanism is proposed for simulating the loss of contact due to force majeure during the search of special forces. The algorithm can more fully explore the solution space and enhance the ability of an algorithm to jump out of a local optimum. Secondly, by simulating the combat tactics of special forces, the algorithm uses command parameters to control the three stages of large-scale search, transitional stage search and small-scale precise search, making the balance between the exploration stage and the exploitation stage more reasonable. Finally, SFA inspired by special forces operations are presented. To verify the effectiveness of the proposed algorithm, SFA is compared with 6 classical algorithms in 23 benchmark functions. In addition, the algorithm is applied to practical engineering problems. Experimental results show that SFA has shown great potential and competitive results. SFA can achieve good search performance and optimization accuracy based on a better balance of exploration and exploitation capabilities.

Keywords: Meta-heuristic; Optimization; Swarm intelligence; Special Forces Algorithm; Benchmark function

1. Introduction

Many practical optimization problems are difficult to obtain certain mathematical results through traditional methods. For example, in today's hot field of machine learning and artificial intelligence, a large number of problems are constrained, multimodal, discrete, and non-differentiable, making it very difficult to solve. In traditional mathematical methods, it is increasingly difficult to solve optimization problems relying on mathematical features such as conjugate gradient method and quasi-Newton method. Traditional mathematical methods that rely on mathematical features such as conjugate gradient and quasi-Newton methods are not applicable to many practical optimization problems such as engineering problems and path planning problems. In some engineering, economic or

* Corresponding authors.
E-mail addresses: wzhang@usst.edu.cn (W. Zhang), 192550427@st.usst.edu.cn (K. Pan).

https://doi.org/10.1016/j.matcom.2023.06.015

commercial aspects, the traditional optimization method is also unreasonable in considering the reduction of time, cost and risk or the improvement of profit, quality and efficiency [22]. Therefore, metaheuristic algorithms have been designed and used to solve these similar problems. Metaheuristic algorithms are simple and easy to implement. They do not assume mathematical features of the underlying cost function (such as gradients) and hence they are suitable for almost any kind of problem. But strictly speaking, they are approximation algorithms, which also means that they do not necessarily get the real global optimum, but only approximate values.

Metaheuristic algorithms have attracted the attention of researchers due to four main factors [18,20]. Firstly, the first advantage of metaheuristic algorithms is their simplicity, since these methods are often inspired by natural phenomena and concepts. Simplicity helps computer scientists improve metaheuristic algorithms and solve complex problems quickly. Secondly, the flexibility and generality of metaheuristic algorithms allow them to solve a variety of problems without changing the algorithms' structure. Thirdly, most metaheuristic algorithms do not require derivations and strict mathematical forms. Metaheuristic algorithms start by generating random solutions to the optimization problem and search by iteration, mathematical characteristics of the problem are not needed. Finally, compared to traditional optimization methods, metaheuristic algorithms can avoid getting stuck in local minima to a certain extent.

Metaheuristic algorithm is a branch of computer science and applied mathematical optimization that deals with complex computational theories and algorithms. It also covers other fields such as artificial intelligence, computational intelligence, soft computing, mathematical programming, and operations research. In practice, metaheuristic algorithms are very effective in solving complex real-world problems, and also play an important role in reducing costs and rapidly expanding in various fields [1].

There are many taxonomy of metaheuristic algorithms in the related literature. In general, according to the source of inspiration, meta-heuristics are divided into four types. They are Evolutionary-based (also called Evolutionary Algorithm), Physics-based, Animal-based, and Human-based. Metaheuristic algorithms are also called swarm intelligence algorithms because they are based on population iteration. In view of this, swarm intelligence is not included separately in the above classification.

As an early meta-heuristic, Genetic Algorithm (GA) [24] also pioneered evolutionary algorithms. It is based on the theory of evolution and employs some operations in optimization, including recombination, mutation, and selection. Over the years, through the efforts of many researchers, many improved versions have been developed [25]. There are some other popular evolutionary algorithms, such as Differential Evolution (DE) [10], Genetic Programming (GP) [34], and Biogeography-Based Optimizer (BBO) [53].

Physics-based algorithms are a type of algorithms based on natural rules or physical laws. One of the most famous Physics-based algorithms is Gravitational Search Algorithm (GSA) [48], which treated search agents as planets and utilized Newton's law of universal gravitation by interacting with each other's mass. Other examples of these algorithms are Inclined Planes System Optimization (IPO) [42,43,51], Central Force Optimization (CFO) [16] and Arithmetic Optimization Algorithm [4].

Animal-based algorithms mimic the wisdom and collective behavior of various biological groups in the natural world. The earliest birth was the proposal of Particle Swarm Optimization (PSO) [12] that was inspired by the birds flocking behaviors. Some of the other popular algorithms in this class are Artificial Bee Colony (ABC) [26], Firefly Algorithm (FA) [56], Bat Algorithm (BA) [57], Prey–predator algorithm [54], Spider monkey optimization algorithm [7], Farmland Fertility Algorithm (FFA) [52], Grey Wolf Optimizer (GWO) [40], Whale Optimization Algorithm (WOA) [38], Crow Search Algorithm (CSA) [6], and Harris Hawk Optimization (HHO) [23].

The last type is Human-based algorithms. In fact, they have a lot of similarity with the Animal-based, but human behavior or social collective strategies are the main inspiration of them. Several examples of this type are Tabu Search (TS) [21], Socio Evolution and Learning Optimization (SELO) [36], Teaching Learning Based Optimization (TLBO) [47] and Queuing search algorithm [59]. The TS algorithm [21] introduces the concept of tabu list. When the TS is initialized, an initial solution is randomly generated in the search space. The tabu list is left blank. The current solution is denoted as the historical optimal solution. In each iteration, starting from the current solution, the neighborhood of the solution is constructed under the constraints of the current tabu list. Selecting the solution with the best fitness value to replace the solution of the previous generation and updating the tabu table at the same time. After the replacement, if the quality of the solution is improved, then the historically optimal solution will be replaced. Repeating these operations until the optimal solution is found or the algorithm ends when the maximum number of iterations is met. SELO [36] mimics the socio-evolution and learning of parents

and children constituting a family. Individuals organized as family groups (parents and children) interact with one another and other distinct families to attain some individual goals. In the process, these family individuals learn from one another as well as from individuals from other families in the society. This helps them to evolve, improve their intelligence and collectively achieve shared goals. The proposed optimization algorithm models this decentralized learning which may result in the overall improvement of each individual's behavior and associated goals and ultimately the entire societal system. The Queuing search algorithm [59] is inspired by human queuing activities. The Queuing search algorithm considers three general phenomena. Firstly, customers actively follow the queue that provides fast service. Secondly, each customer service is mainly influenced by the staff or the customers themselves. Finally, when the queue order is not strictly maintained, customers may be influenced by others during the service process. According to these three behaviors, the author sets different search mechanisms and realizes the final optimization. In conclusion, all these human-based metaheuristic algorithms construct mathematical searching models by simulating human behaviors.

Despite differences in inspiration and search mechanics, they strive to balance exploitation and exploration. At the beginning of generation, metaheuristic algorithms benefit from the exploration process to generate new solutions. After the exploration phase, they gradually transition to the development phase. During the development phase, metaheuristic algorithms focus on improving the accuracy of the candidate solutions obtained from the exploration phase. The exploitation process results in new solutions based on the best solutions available to the population. Therefore, metaheuristic algorithms use two basic exploration and exploitation processes to avoid falling into local traps and converge to the goal [2,19,41,45].

It is worth mentioning that we only reviewed a few of the most important and basic meta-heuristic algorithms. There are also other meta-heuristic optimization algorithms such as: Teaching Learning Based Optimization (TLBO) [46], Cuckoo Search (CS) [17], Simulated Annealing (SA) algorithm [33], Vibrating Particles System (VPS) [30]; Galaxy-based Search Algorithm (GbSA) [50], Bird Mating Optimizer (BMO) [5]; Big-Bang Big-Crunch (BB-BC) algorithm [13], Social Spider Optimization (SSO) [9], Intelligent Water Drops (IWD) algorithm [49], Colliding Bodies Optimization (CBO) [31]; Charged System Search (CSS) algorithm [32], Water Evaporation Optimization (WEO) algorithm [28]; Glowworm Swarm Optimization (GSO) [35], League Championship Algorithm (LCA) [27], Dolphin echolocation optimization (DEO) [29], and Water Cycle Algorithm (WCA) [14].

Though there are several human-based metaheuristic algorithms, the essential inspirations are modeling the processes of the human's behaviors. Since the special forces' tactics of modern society is mature and advanced, this work proposes a new metaheuristic optimization algorithm by modeling the decentralized procedures of hostage rescuing task carried by special forces. Some of the highlights of this paper are as follows:

- An index *Instruction* with random factor during the optimization which used to change exploration and exploitation phases is created.
- A unique search method based on the probability of losing contact p and the search vector A defined in Section 3 is proposed.
- An independent mechanism to increase the ability of obtaining information around known locations is established.
- Three searching mechanisms are proposed. They are combined in different stages, which can allow them to take advantage of jumping and randomness.
- Thresholds have been designed during the optimization operation, which has a significant impact on the exploration and exploitation potential of SFA.

The main contributions of this paper are as follows. First of all, at the beginning of the algorithm iteration, the light speed mechanism is proposed to simulate the unmanned aerial vehicle assisted search of special forces, and the loss probability mechanism is proposed to simulate the loss of contact caused by force majeure during the search of special forces. The algorithm can fully explore the solution space, and enhance the ability of the algorithm to jump out of the local optimum. Secondly, the algorithm makes the balance between the exploration and exploitation stages more reasonable through three stages of large-scale search, transition stage search and small-scale precise search. As a result, this paper proposes a special force algorithm inspired by the operation of special forces. Finally, experiments are carried out on 23 classic standard test functions, and the proposed algorithm is applied to actual engineering optimization problems to verify the effectiveness of the proposed algorithm. Experimental results show that the proposed algorithm has good optimization performance.

In the remainder of the paper, Section 2 introduces the inspiration of the proposed SFA. Section 3 presents the specific mathematical formula, algorithm flow chart and pseudo code of SFA. The performances and efficiency of the proposed SFA are tested in Section 4. All the results were discussed and analyzed. Section 5 summarizes the conclusions and prospects of this paper.

Fig. 1. SWAT.

2. Inspiration

Special forces are armies responsible for raiding important political, economic or military targets of the enemy and performing other special tasks. According to the prior division of labor, these teams carry out surprise attacks, assassinations, or destruction on their respective targets. Today, special forces have evolved into two forms. One is the special forces in the army, and the other is SWAT as shown in Fig. 1.

The special forces represented by SWAT are the expansion of the police to the army, and they are equipped with sophisticated individual weapons. Special forces can perform a variety of tasks, including anti-terrorism, hostage rescue, anti-riot and armed patrols. For example, when carrying out anti-terrorism patrol missions, they divide many teams to search for potential targets (terrorists or hostages). The team is deployed in a decentralized manner, which also requires team members to undertake various tasks such as peripheral reconnaissance, sniping and lurking. During this period, they went through patrol, ambush and anti-ambush until they were cleared at a fixed point.

Integrated emergency medical services by special operation teams during hostage rescue requires an understanding of the unique attributes of the tactical environment. The hostages first need to be searched before medical services can be administered. Usually, technology such as satellite radar is used to assist members of special forces to conduct range searches. In areas where terrorists are hiding and fleeing, it is usually necessary to search in a large area to determine the approximate area where terrorists are located. After the general area is determined, the team members gradually lock the area of the terrorist's location through communication equipment such as walkie-talkies. Finally, a dragnet-type precise search and rescue were carried out. In the rescue process, drones also play a vital role in urban hostage rescue. Based on these rescue scenarios, this paper proposes a new algorithm called SFA, which is a swarm intelligence algorithm based on the behaviors and strategies of special forces in performing hostage rescuing missions.

3. Special forces algorithm

The characteristics of special forces mainly include the following four points. First of all, special forces members will use satellite communications, drones and other equipment to assist in the search and rescue of hostages. During the search and rescue process, team members may encounter terrorists, or there may be problems with the signal of communication equipment, which may cause team members to lose contact. Therefore, the loss probability mechanism and the speed of light random unmanned search mechanism are introduced. Secondly, in the exploration phase, special forces will conduct a large-scale search. When the information has not been grasped in the early stage, the team members can only act separately and conduct a dragnet search. Thirdly, in the transition stage, after the team members determine the approximate range of the hostages, they conduct a mid-range search. Finally, after determining the specific location of the hostages, the development stage carried out a quick rescue.

As a type of common algorithms in the field of optimization, metaheuristic algorithms usually consist of three stages. The first is the initialization phase, in which the algorithm completes the initialization of the candidate solutions so that they are scattered in the problem solution space as much as possible. Secondly, different algorithms

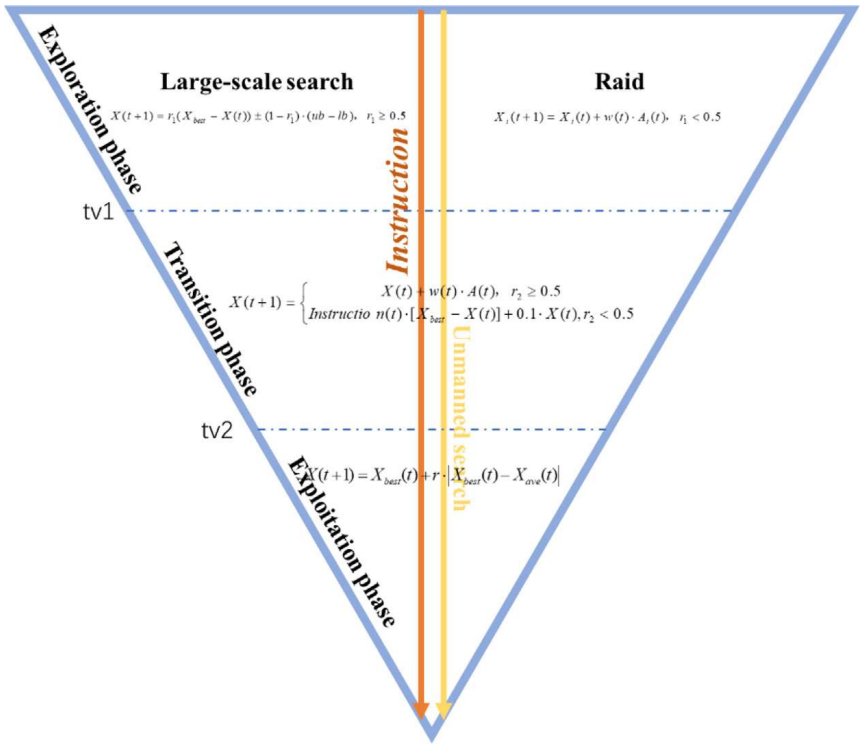


Fig. 2. Searching processes of the proposed SFA.

use specific strategies to explore the solution space, and mainly complete the full exploration of the solution space in the exploration stage. Finally, in the exploitation phase, the algorithm conducts more accurate exploration around the previously explored global optimal solution to improve the accuracy of the solution.

From the description of the characteristics of special forces' hostage rescuing processes and the searching strategies of metaheuristic algorithms, it can be seen that the searching processes are similar to the searching strategies of metaheuristic algorithms. Inspired by this, this paper proposed a new metaheuristic algorithm (called SFA) by modeling the searching processes of hostage rescuing mission carried by special forces. Compared with the existing metaheuristic algorithms, the proposed SFA has two main features and differences. Firstly, the distinguishing scheme between exploration stage and exploitation phase is different from other metaheuristic algorithms. An index *Instruction*(t) with random factor during the optimization is proposed to change the exploration and exploitation phases. Secondly, the loss-of-connection probability mechanism and the random light speed mechanism are proposed to improve the ability of avoiding falling into local optimum. The proposed SFA has three stages including exploration phase, transition phase and exploitation phase. Fig. 2 gives the searching processes of the proposed SFA. Details will be described in the next section.

3.1. Special conditions

Before describing the proposed SFA in detail, some special components of the algorithm are introduced. In the SFA, there is an "instruction", which will be used as a sign to guide all members to perform tasks. The specific task type is distinguished by the instruction and the following threshold, expressed as Eq. (1). The *Instruction* will change with the number of iterations, as seen in Fig. 3.

$$Instruction(t) = (1 - 0.15rand)(1 - \frac{t}{T}) \tag{1}$$

where t is the current iteration number, T is the maximum number of iterations, and *rand* is a random number inside (0,1).

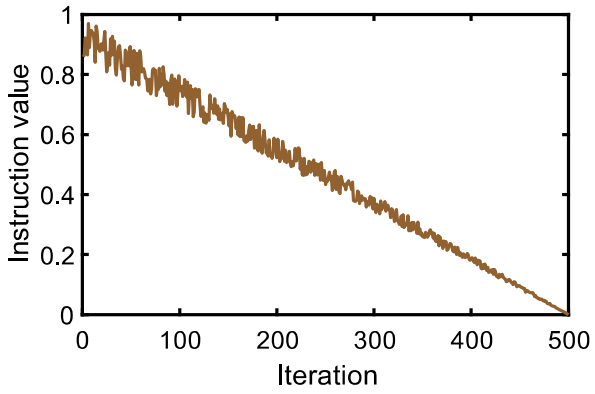


Fig. 3. The curve of Instruction during 500 iterations.

A function $task(Instruction(t)) : [0, 1] \rightarrow T = \{T_1, T_2, T_3\}$ asks that assigns a $task(Instruction(t)) \in T$ to the team at time t is defined in Eq. (2). Two thresholds tv_1 and tv_2 are given for distinguishing different tasks.

$$task(Instruction(t)) = \begin{cases} T_1, & tv_1 \leq Instruction(t) \leq 1 \\ T_2, & tv_2 < Instruction(t) < tv_1 \\ T_3, & 0 \leq Instruction(t) \leq tv_2 \end{cases} \tag{2}$$

where $T_1 = \{\text{Large-scale search, Raid}\}$ corresponding to exploration phase, $T_2 = \{\text{Transition phase}\}$, $T_3 = \{\text{Arrest-rescue}\}$ corresponding to exploitation phase, $0 \leq tv_2 < tv_1 \leq 1$. It should be noted that tv_1 and tv_2 are used to balance the phases of exploration, transition and exploitation. To achieve the best performance, tv_1 and tv_2 should be adjusted according to the problem to be solved. But this is unfavorable for the practical application of the algorithm. From numerous tests on 23 classic standard test functions, tv_1 and tv_2 are chosen as 0.5 and 0.3 for simulation in this paper.

During the execution of the task, members can get the position information of their teammates, but the communication of some members may be interrupted. In the whole process of the algorithm, any member may lose the information of some members, which is described by Eq. (3).

$$p(t) = p_0 \cos(\frac{\pi t}{2T}) \tag{3}$$

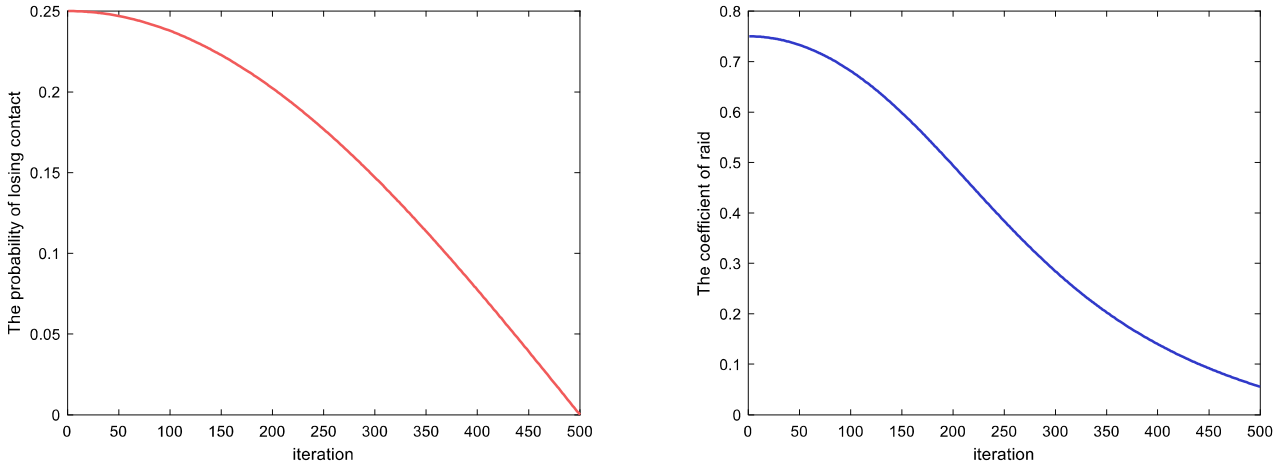
where $p(t)$ is the probability of losing contact at the current iteration t, and p_0 is the initial constant. During the hostage search of special forces, communication equipment such as walkie-talkies may be interfered by terrorists, so there may be loss of contact. However, as the search space gradually shrinks, the probability of team members losing contact will gradually decrease. In a small range, communication can be achieved without the use of communication equipment. Therefore, as the iteration progresses, the probability of losing contact gradually decreases to 0. The cosine function is chosen to simulate the probability of losing contact of special members. In the early stage of iteration, the probability of losing contact is relatively high. In the middle of the iteration, there is still a certain probability of losing contact. The probability gradually decreases to 0 in the later stages of the iteration. Considering the clarity of the situation and information, the probability of losing contact will gradually decrease with the increase of the number of iterations, and its change trend is shown in Fig. 4. (a).

3.2. Exploration phase

The exploration phase is the initial stage after the initialization of the algorithm is completed. The exploration phase of SFA includes two strategies including large-scale search and raid.

(1) Large-scale search

In the exploration phase, special forces mainly perform large-scale search tasks. In large-scale search, the activity area of team members can be very large. They are allowed to randomly search for any potential target anywhere within feasible limits.



(a) the probability of losing contact (b) the coefficient of raid

Fig. 4. The trend of two iterative parameters.

Considering the possibility that the members may perform two tasks in the exploration stage, the algorithm introduces a random number to give the members the randomness of search and the division of tasks, that is, update the location according to Eq. (4).

$$X(t+1) = r_1(X_{best} - X(t)) \pm (1 - r_1) \cdot (ub - lb), r_1 \geq 0.5 \tag{4}$$

where $X(t+1)$ represents the position vector of the member in the next iteration, $X(t)$ is the current position vector of the member, X_{best} is the current best position of all the team, r_1 is the random number taken inside the interval (0,1). ub and lb are the upper and lower bounds of the variables.

(2)Raid

In the process of large-scale search missions by special forces, special forces sometimes may raid possible locations since they have mastered some information about hostages or gangsters. Each time they move, their position will be affected by the direction of the closest best member they know. When the random number r_1 determines to carry out sudden search, the members update their positions according to Eq. (5).

$$X_i(t+1) = X_i(t) + w(t) \cdot A_i(t), r_1 < 0.5 \tag{5}$$

where $A_i(t)$ represents the search vector of any team member in the current process. For any member, the search vector can be given by Eq. (6).

$$A_i(t) = \frac{f_i(t)}{f_i(t) + f_{aim}(t)}[X_{aim}(t) - X_i(t)] \tag{6}$$

where $X_i(t)$ is the current position of member i, $X_{aim}(t)$ is the position of member aim which is the target of the member i as the known best position currently, $f_i(t) = f(X_i(t))$ and $f_{aim}(t) = f(X_{aim}(t))$ are the fitness values of $X_i(t)$ and $X_{aim}(t)$ at time t respectively.

According to Eq. (3), for any member, he may lose some position information of other team members during any iteration. Therefore, the current best position known by each team member may not be the best position of the whole force. $w(t)$ in Eq. (5) is the coefficient of $A_i(t)$, called the coefficient of raid, which will gradually decrease according to the number of iterations until it is close to 0, as shown in Eq. (7).

$$w(t) = 0.75 - 0.55 \arctan[(\frac{t}{T})^{2\pi}] \tag{7}$$

The trend of the coefficient of raid $w(t)$ with the number of iterations is shown in Fig. 4(b).

3.3. Transition phase

The transition phase is a buffer link between the exploration phase and the exploitation phase. At this stage, the members can continue to complete the previous tasks and will gradually change to the exploitation phase. See Eq. (8) for details.

$$X(t+1) = \begin{cases} X(t) + w(t) \cdot A(t), r_2 \geq 0.5 \\ Instruction(t) \cdot [X_{best} - X(t)] + 0.1 \cdot X(t), r_2 < 0.5 \end{cases} \tag{8}$$

where $A(t) = [A_1(t), A_2(t), \ldots, A_N(t)]$, $X(t) = [X_1(t), X_2(t), \ldots, X_N(t)]$, N represents the total number of the whole force members, $r_2 \in (0, 1)$ is a random number satisfying the uniform distribution.

The strategy of raid is a special task form of special forces in the exploration phase and transition phase. It is conducive to different processing of the information when some exact information is known, and the information can be verified through search without missing the possible efficiency of the information.

3.4. Exploitation phase

In the exploitation phase, special forces have already obtained a lot of information on the location of criminals or hostages, and officially enter the moment of capture. Their task is to arrest criminals or rescue hostages. The strategy is called “arrest-rescue”.

In the iterative process, as the information of the situation becomes clearer, it can be known from Eq. (3) that the information loss rate of team members will be close to 0. On this basis, the special force members in the exploitation stage will decisively approach and adopt a centralized siege to attack according to the most likely point known by the whole team (here is the location of the hostage or gangster). At this time, the position update adopted by the team members is represented by Eq. (9).

$$X(t+1) = X_{best}(t) + r \cdot |X_{best}(t) - X_{ave}(t)| \tag{9}$$

where r is the random number row vector taking value inside $(-1,1)$ satisfying uniform distribution, and its dimension is the dimension of search space.X_{ave} is average number of the current position of the whole members, which can be calculated by Eq. (10).

$$X_{ave}(t) = \frac{1}{N} \sum_{i=1}^{N} X_i(t) \tag{10}$$

where N represents the total number of the whole force members.

The arrest-rescue work is the final stage of the special forces. In terms of strategy, the members adopt the method of gradually approaching the goal. They gradually converge and narrow the encirclement.

3.5. Unmanned search

Unmanned search is a task line independent of the main force, which is undertaken by micro robots or unmanned aerial vehicles. They are like intelligence agents on the battlefield. In each iteration, these machine intelligence agents perform a random search described in Eq. (11) based on the location information of all special force members in the previous iteration. Then, all the information is fed back. At the end of each iteration, the location information of the current team members and unmanned search information are integrated to participate in the update of the current optimal value.

$$\begin{cases} X_u = X + v \\ v_1^2 + v_2^2 + \cdots + v_{\text{dim}}^2 = c^2 \end{cases} \tag{11}$$

Among them, v_{dim} is the component of v in each dimension, which is the next moving distance. c is the maximum movable range, determined by Eq. (12), X_u is the updated position.

$$c = k \cdot (lb + (1 - \frac{t}{T}) \cdot (ub - lb)) \tag{12}$$

where k is a constant, with a value inside the interval $(0, 0.5)$.

In the computer program, we randomly construct v so that the sum of squares of its components is equal to the sum of squares of c.

Unmanned search is a unique information channel. Because its formula is based on the principle of constancy o f light velocity, it is also called the random mechanism of light velocity.

3.6. Algorithm implementation

Concluding the searching strategies of the proposed SFA, Fig. 5 gives the simulated searching processes of SFA in 3D environments. Fig. 6 shows the flowchart of the proposed SFA algorithm.

3.7. Computational complexity analysis

The computational complexity of SFA is mainly composed of the following processes: initialization, fitness value calculation, lost connection information filtering and location update. When the number of special forces is n, the maximum number of iterations is t, and the dimension of the problem is D, the computational complexity of initialization is $O(N)$. The computational complexity of fitness value update calculation is $O(2N \times T)$, the computational complexity of lost contact information filtering is $O(N \times T)$, the computational complexity of unmanned search calculation is $O(N \times T \times D)$, the computational complexity of location update is $O(N \times T \times D)$. Therefore, the total computational complexity is $O(N \times (1 + 3T + 2T \times D))$.

The pseudo-code of the proposed algorithm is shown in the following Algorithm 1.

Algorithm 1: Pseudo-code of SFA

Initialize parameters: tv_1 , tv_2 , p_0 ,the population size N and maximum number of iterations T

Initialize: the positions of special force members $X_i (i = 1,2,...,N)$

While $t \le T$ **do**

Calculate X_u of unmanned search by Eq. (11)

Calculate the fitness values of all members and X_u

Update the best fitness and its location X_{best}

Construct *Instruction* by Eq. (1)

If $Instruction(t) \ge tv_1$ **do**

If $r_1 \ge 0.5$ **do** update the positions by Eq. (4)

Else if $r_1 < 0.5$ **do** update the positions by Eq. (5)

Else if $tv_2 < Insturction(t) < tv_1$ **do**

If $r_2 \ge 0.5$ **do** update the positions of members by Eq. (8)

Else if $r_2 < 0.5$ **do** update the positions by Eq. (8)

Else if $Instruction(t) \le tv_2$ **do**

Update the positions of members by Eq. (9)

End if Update p, w and $t = t + 1$

End While

Return X_{best}

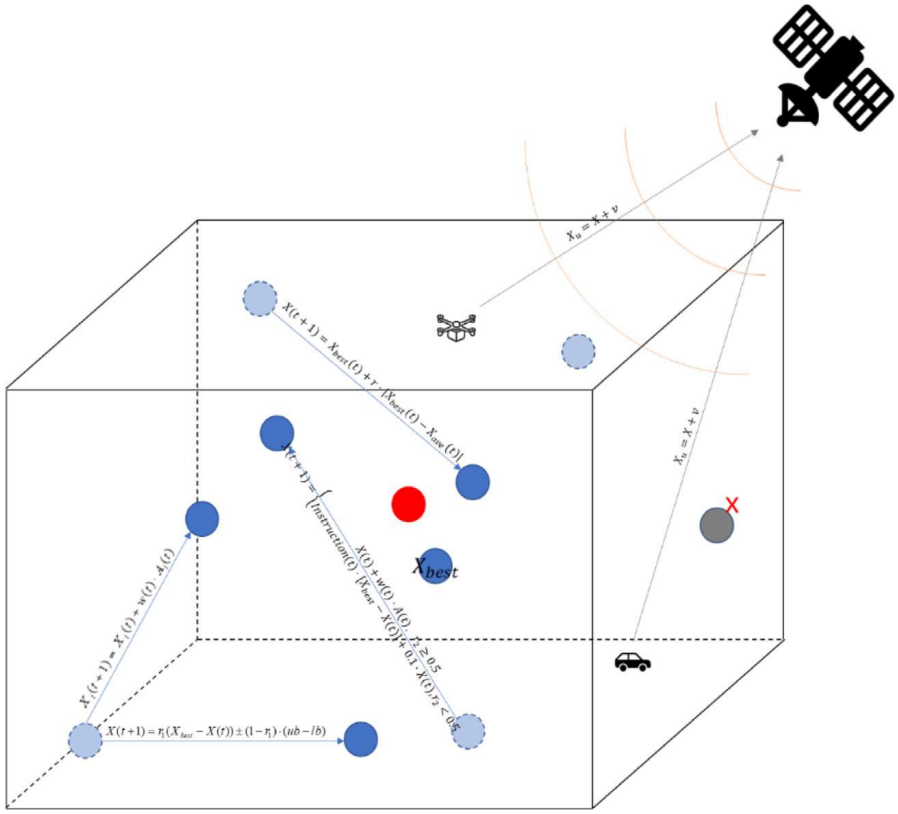


Fig. 5. The flowchart of SFA.

4. Experimental results and analysis

4.1. Benchmark functions and compared algorithms

In order to demonstrate the effectiveness of the proposed optimizer, a well-studied set of diverse benchmark functions are selected from references [11,58] to evaluate the performance of SFA.

A total of 23 benchmark functions, including unimodal function, multimodal function and fixed dimensional multimodal function, mainly evaluate SFA's ability of exploration and exploitation. The unimodal test function (F1–F7) has only one global best value, which is used to test the exploitation ability of the algorithm, that is, the accuracy of convergence and approximation. And the multimodal function (F8–F13) is used to reveal the exploration performance of the algorithm, which is related to the diversification capabilities and the ability to avoid local optima. The fixed dimensional multimodal function (F14–F23) is specifically used to evaluate the comprehensive performance of the algorithm in the case of fixed dimensions. The mathematical formulas and details of these test functions are shown in Table 1. The proposed SFA is compared with other mature algorithms. These algorithms include PSO [12], AOA [4], GWO [40], WOA [38] and two new high-performance algorithms in recent years: Equilibrium optimizer (EO) [15] and Harris Hawks Optimizer (HHO) [23]. In order to measure the experimental results, this comparison is evaluated by average results (Ave) and standard deviation (Std), and the best results will be displayed in boldface.

4.2. Test platform

All tests have the same running environment: the test software is MATLAB R2018a, the computer operating system is Windows 10 Home, and the CPU is Intel (R) core (TM) i5- 10400@2.9 GHz. In the qualitative analysis part, the population number of SFA is set to 30 and the maximum number of iterations is 200. In quantitative analysis, all algorithms are tested under the same conditions. The population number is set to 50 and the maximum

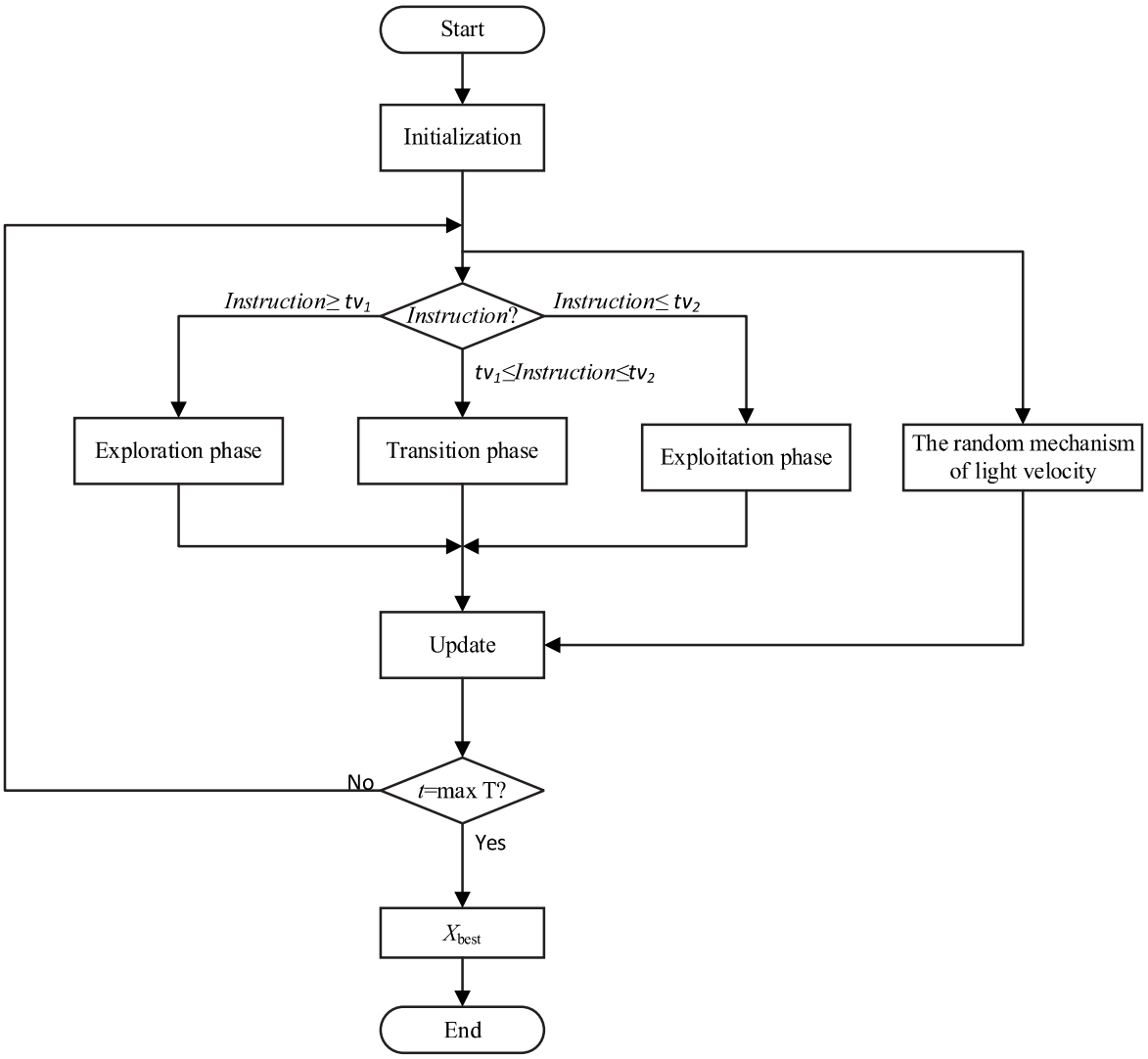


Fig. 6. Simulated searching processes of SFA in 3D space.

number of iterations is set to 500. The parameter settings of all algorithms are shown in Table 2 (including SFA). In order to reduce the impact of random factors in the algorithms on the results, all compared algorithms run 30 times in each function, and take the average value as the final result.

4.3. Qualitative analysis

Fig. 7 shows the qualitative results of SFA for several test functions. There are five graphs in each set of results: the search space, the convergence curve, the trajectory of the first member and the average fitness of all members. In order to clarify all the indicators that change with the number of iterations, the instruction of the special forces is also shown at the end.

The first picture of each group of pictures shows the search space of the functions, which depicts the three-dimensional appearance of the corresponding function. The convergence curve shows the change of the best fitness value of all members during the search process. By observing the downward trend of the curve, we can know the convergence speed of the special forces and the time to switch between exploration and exploitation. The trajectory of the first member monitors the first dimensional variable of the first member during the optimization process. The average fitness curve reflects how the average fitness of whole special forces changes during the process. The diagram of instruction reveals the details of this indicator.

Table 1
Details of benchmark functions.

Function	Dim	Range	$f_{\min}$
$f_1(x) = \Sigma_{i=1}^{n} x_i^2$	50	[−100, 100]	0
$f_2(x) = \Sigma_{i=1}^{n} \lvert x_i \rvert + \Pi_{i=1}^{n} \lvert x_i \rvert$	50	[−100,100]	0
$f_3(x) = \Sigma_{i=1}^{n} (\Sigma_{j=1}^{i} x_i)^2$	50	[−100, 100]	0
$f_4(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	50	[−100, 100]	0
$f_5(x) = \Sigma_{i=1}^{n} [100(x_{i+1} + x_i^2)^2 + (x_i - 1)^2]$	50	[−30, 30]	0
$f_6(x) = \Sigma_{i=1}^{n} ([x_i + 0.5])^2$	50	[−100, 100]	0
$f_7(x) = \Sigma_{i=1}^{n} i x_i^4 + random[0, 1]$	50	[−1.28, 1.28]	0
$f_8(x) = \Sigma_{i=1}^{n} - x_i \sin(\sqrt{\lvert x_i \rvert})$	50	[−500, 500]	$-418.98 \times n$
$f_9(x) = \Sigma_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	50	[−5.12, 5.12]	0
$f_{10}(x) = -20\exp(-0.2(\frac{1}{n}\Sigma_{i=1}^{n} x_i^2)^{0.5}) - \exp(\frac{1}{n}\Sigma_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	50	[−32, 32]	0
$f_{11}(x) = \frac{1}{4000}\Sigma_{i=1}^{n} x_i^2 - \Pi_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	50	[−600, 600]	0
$f_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \Sigma_{i=1}^{n}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \Sigma_{i=1}^{n} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{x_i + 1}{4}$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a) x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a) x_i < a \end{cases}$	50	[−50, 50]	0
$f_{13}(x) = 0.1\{\sin^2(3\pi x_1) + \Sigma_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \Sigma_{i=1}^{n} u(x_i, 5, 100, 4)$	50	[−50, 50]	0
$f_{14}(x) = (\frac{1}{500} + \Sigma_{j=1}^{25}\frac{1}{j + \Sigma_{i=1}^{2}(x_i - a_{ij})^6})^{-1}$	2	[−65, 65]	1
$f_{15}(x) = \Sigma_{i=1}^{11}[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]^2$	4	[−5, 5]	0.00030
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[−5, 5]	−1.0316
$f_{17}(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	2	[−5, 5]	0.398
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	[−2, 2]	3
$f_{19}(x) = -\Sigma_{i=1}^{4} c^i \exp(-\Sigma_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	3	[1, 3]	−3.86
$f_{20}(x) = -\Sigma_{i=1}^{4} c^i \exp(-\Sigma_{j=1}^{6} a_{ij}(x_i - p_{ij})^2)$	6	[0, 1]	−3.32
$f_{21}(x) = -\Sigma_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	−10.1532
$f_{22}(x) = -\Sigma_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	−10.4028
$f_{23}(x) = -\Sigma_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	−10.5363

As shown in Fig. 7, the convergence graphs can be steadily decreased, and finally reach a sufficiently accurate optimal value. The convergence trend can fully reflect the random movement and jump of SFA in the exploration phase, and can clearly distinguish the changing moments of several phases. The final value of the convergence curve directly determines the optimization performance of the algorithm. This important data will be evaluated in detail in the subsequent quantitative analysis comparison chart.

The trajectory graph can tell us the movement path of the first member as a representative. In this way, we can judge the situation of the individual in the whole process, such as whether the search method is changed according to the stage and whether it is converging to the current optimal value. Referring to Van Den Bergh and Engelbrecht [55], these activities can guarantee that a meta-heuristic finally convergences to a position and exploit the target region. According to the motion trajectory in Fig. 7, we can see that in the early search activity, the

Table 2
Parameter settings of optimization algorithms.

Algorithm	Parameter	Value
SFA	Thresholds of phases tv_1, tv_2	0.5, 0.3
	Initial number of the probability of losing contact p_0	0.25
PSO [12]	Inertia weight w	Linear reduction from 0.9 to 0.1
	Cognitive and social constant c_1, c_2	2, 2
AOA [4]	MOP	[0, 1]
GWO [40]	Convergence constant a	[2, 0]
WOA [38]	Convergence constant a	[2, 0]
	Spiral factor b	1
EO [15]	a_1, a_2	2, 1
	Generation probability GP	0.5
HHO [23]	default constant in the levy flight β	1.5

amplitude spans in the image are very large, which means that the global search range at this time is very wide. This is also in line with the early exploration tendency. As time goes by, the amplitude will be greatly reduced, but it will remain to a certain extent, because the SFA needs randomness in the transition phase to avoid falling into a local minimum. In the end, the trajectory of the first member will be stable, which also means successful convergence during the exploitation phase.

According to the three types of graphs in Fig. 7, we can clearly find the moments of phase division, and the average fitness curve reveals that the SFA will not converge too early or excessively. Although the average fitness curve is fluctuating, the average fitness value maintains a downward trend. The decrease in oscillation frequency is inversely proportional to the number of iterations, thus ensuring sufficient search in the early stage and rapid convergence in the later stage. Combining the information in the figure, we conclude that SFA can balance each search stage well, and complete global search and fast convergence respectively.

4.4. Quantitative analysis

In this subsection, the results of SFA are compared with other optimization algorithms. Table 3 shows the numerical results of SFA and other algorithms in all test functions. As shown in Table 1, benchmark functions include 50 dimensional unimodal functions, 50 dimensional multimodal functions and fixed dimensional functions. According to the data in Table 3, in the unimodal function F1–F5, multimodal function F8–F11, and fixed-dimensional function F21–F23, SFA can be compared with the other 6 algorithms under the same conditions to obtain the best results. In the tests of other benchmark functions, although the results of SFA are not the best among all algorithms, it can always lead by a large margin with most algorithms. SFA can always find or approach the theoretical optimal value, which is very fulfilling in terms of performance. For example, in the fixed-dimensional function F21–F23, SFA can be perfectly close to the optimal value, while other algorithms cannot reach.

In the test functions F1–F4, F10–F14, the data shows that SFA has an order of magnitude advantage over most of the algorithms in the test. By comprehensively analyzing of the data in the table, SFA has sufficient optimization capabilities to solve various problems. Especially in the test of multimodal function and fixed-dimensional function, the results are more meaningful, because practical problems often have many local minima, which shows that SFA is obviously better than other algorithms in most of the time.

Fig. 8.1 and Fig. 8.2 are a comparison of the convergence curves of SFA and other algorithms in several test functions, which shows the performance of these algorithms more clearly. Generally, unimodal performance reflects the local optimization capability of the algorithm, while the multimodal function emphasizes the global search capability of the algorithm. In addition to showing some of the same information as the qualitative analysis part,

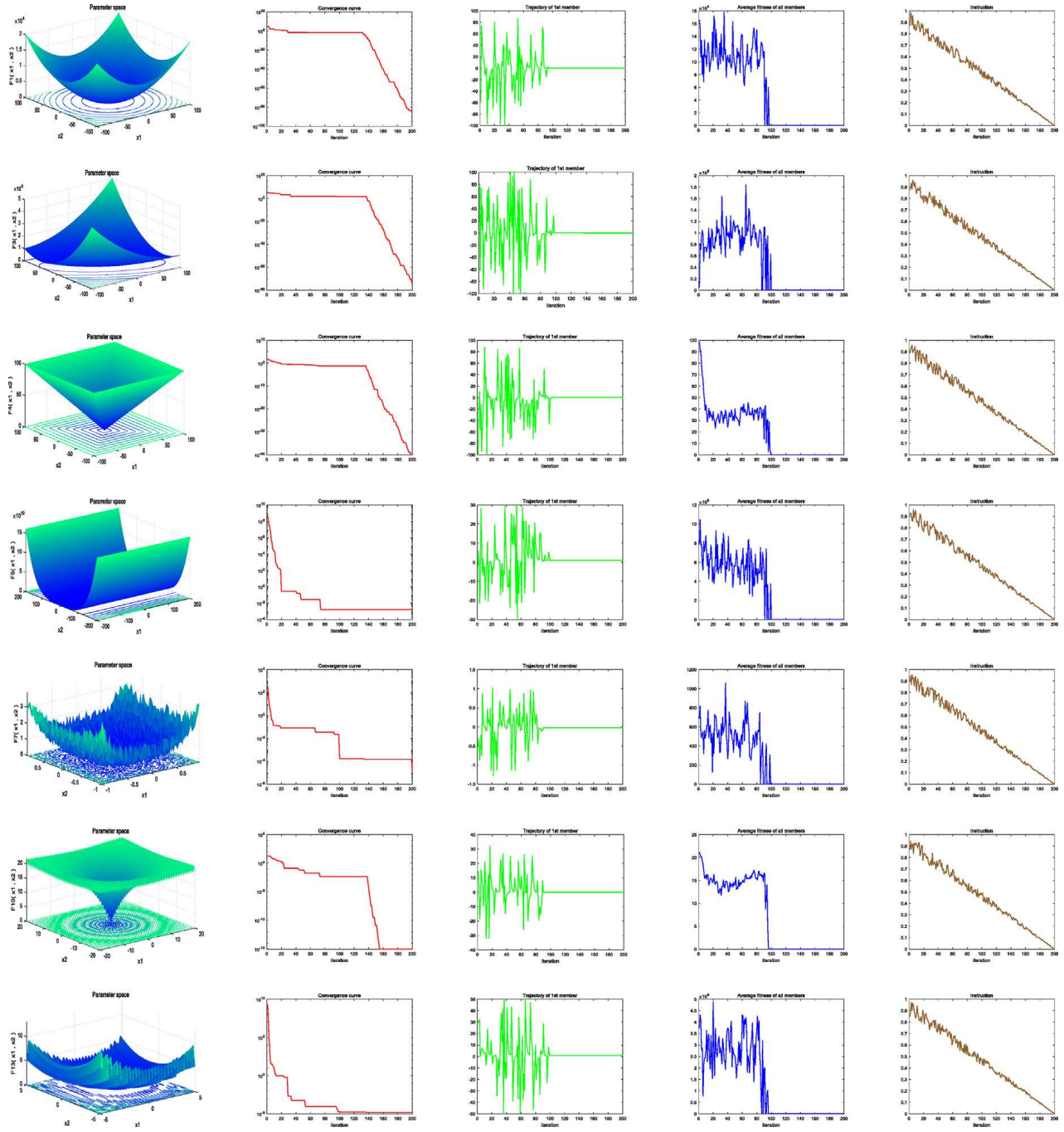


Fig. 7. Qualitative results of SFA for several test functions.

Fig. 8.1 and Fig. 8.2 fully illustrate the performance and search differences between various algorithms. Considering the stability of the algorithms and a large number of tests, our conclusion is reliable. It also shows that SFA has excellent optimization performance and excellent convergence ability, because the algorithm has good exploitation potential and exploration advantage at the same time. According to all the experiments performed, SFA can achieve a better balance between the ability of exploration and exploitation.

In Fig. 8.1, SFA has good performance on unimodal functions. Except for F7, F1–F6 are better than other algorithms, which shows that the convergence accuracy of the algorithm is high. From the point of view of the convergence process, the algorithm converges slowly on F1–F4 in the early stage, and only jumps out of the local

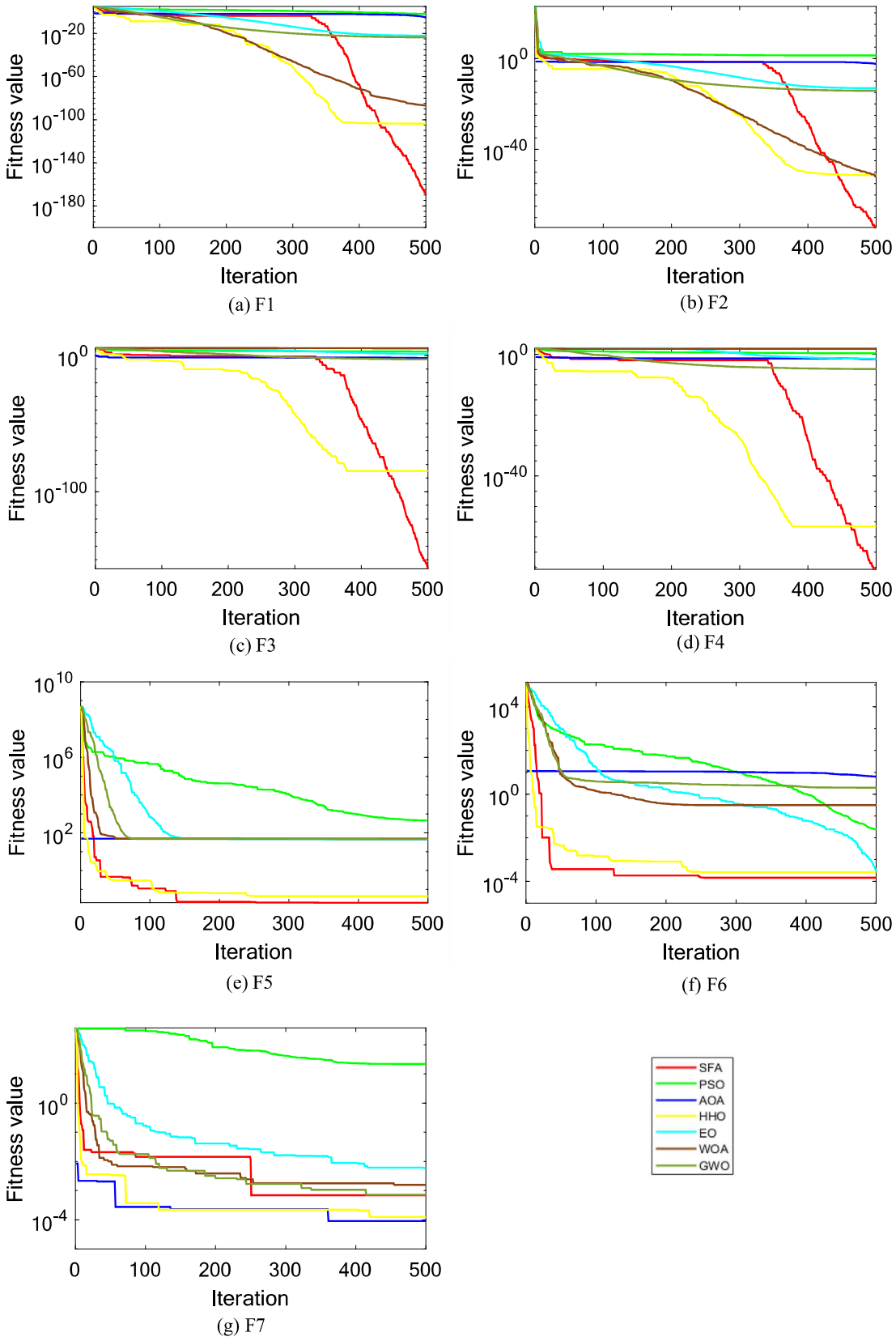


Fig. 8.1. Comparisons between SFA and other algorithms (F1–F7).

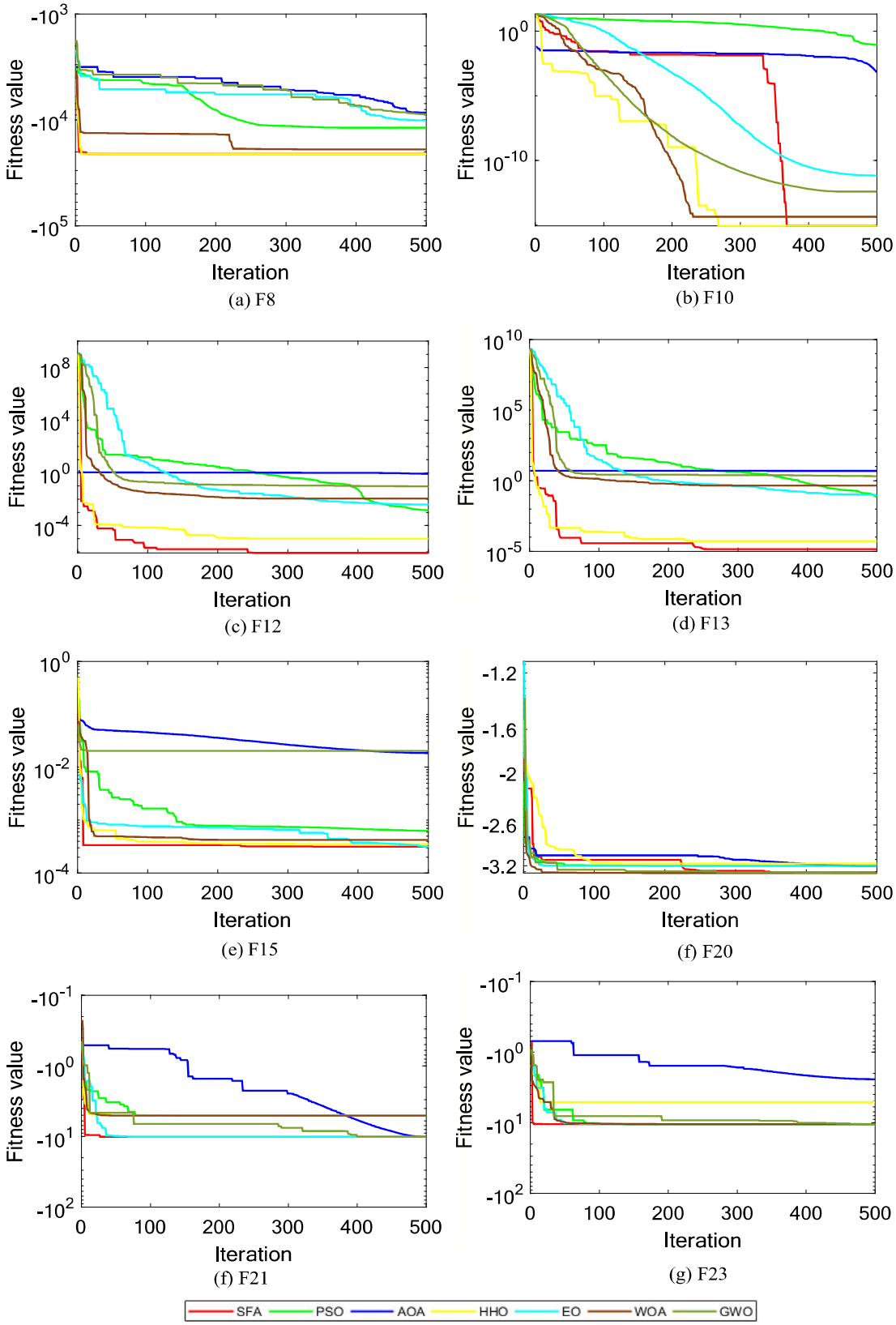


Fig. 8.2. Comparisons between SFA and other algorithms for several test functions.

Table 3
Results of benchmark functions (F1–F23).

		SFA	PSO [12]	AOA [4]	GWO [40]	WOA [38]	EO [15]	HHO [23]
F1	Ave	**9.2699e−136**	0.0339	4.6332e−06	5.3628e−24	4.9212e−84	1.4245e−21	3.7491e−100
	Std	5.0772e−135	0.0225	1.6904e−06	4.1473e−24	2.4193e−83	1.1831e−21	2.0298e−99
F2	Ave	**1.9458e−71**	20.2653	2.1428e−03	1.3922e−14	9.2989e−52	1.1636e−13	2.5084e−53
	Std	9.2215e−71	13.1765	1.7952e−03	7.3057e−15	4.6110e−51	6.3903e−14	8.8571e−53
F3	Ave	**2.2280e−135**	860.5072	1.1528e−03	0.0112	1.4961e+05	28.2184	1.8187e−77
	Std	1.2203e−134	242.3980	7.8544e−04	0.0170	2.4409e+04	33.4470	9.9597e−77
F4	Ave	**7.8269e−68**	2.6892	1.6784e−02	2.8063e−05	63.4729	0.0212	9.4698e−51
	Std	4.2869e−67	0.4012	1.1126e−02	2.1247e−05	25.5241	0.0245	3.3784e−50
F5	Ave	**0.0163**	276.4977	2.7862e+01	47.0105	47.7499	45.1474	0.0196
	Std	0.0161	157.3219	2.2688e−01	0.6968	0.3679	0.4026	0.0164
F6	Ave	0.0038	0.0336	3.1352e+00	1.9156	0.4569	0.0250	**7.4108e−05**
	Std	0.0040	0.0236	2.2857e−01	0.5322	0.2374	0.0755	9.9175e−05
F7	Ave	0.0016	21.9250	**5.6643e−05**	0.0019	0.0019	0.0044	9.3641e−05
	Std	0.0014	19.5640	4.5464e−05	8.5802e−04	0.0015	0.0016	7.8797e−05
F8	Ave	−20949	−8746.4	−5486.2	−9047.2	−1831.5	−11632	−20949
	Std	0.8838	2028.2	436.6	1496.2	2532.8	778.3004	0.9053
F9	Ave	**0.0000**	265.4557	1.5661e−06	2.4067	0.0000	8.0839	**0.0000**
	Std	0.0000	36.3133	8.7428e−07	3.5157	0.0000	3.9042	0.0000
F10	Ave	**8.8818e−16**	0.9904	4.4105e−04	5.4753e−13	4.9146e−15	6.1170e−12	**8.8818e−16**
	Std	0.0000	0.5835	1.9622e−04	2.6438e−13	2.2340e−15	3.0387e−12	0.0000
F11	Ave	**0.0000**	0.0059	2.3628e−05	0.0017	0.0000	5.7536e−04	**0.0000**
	Std	0.0000	0.0072	1.1152e−05	0.0046	0.0000	0.0022	0.0000
F12	Ave	2.0363e−05	0.0214	7.4282e−01	0.0721	0.0112	0.0010	**2.1928e−06**
	Std	2.8392e−05	0.0413	2.7439e−02	0.0315	0.0071	0.0020	3.3109e−06
F13	Ave	1.8056e−04	0.0503	2.9659e+00	1.5011	0.5504	0.0609	**4.4561e−05**
	Std	1.6167e−04	0.0396	1.2652e−05	0.3672	0.2255	0.0657	6.8784e−05
F14	Ave	**0.9980**	1.5930	9.1842	4.5560	2.7663	**0.9980**	1.2618
	Std	9.9296e−11	0.9913	3.9652	4.2063	3.0314	1.4857e−16	0.9320
F15	Ave	0.0012	0.0055	4.4328e−03	0.0043	8.0036e−04	0.0024	**3.8699e−04**
	Std	0.0033	0.0084	6.9826e−03	0.0082	4.9874e−04	0.0061	2.3118e−04

(*continued on next page*)

optimum in the later stage, which shows that the speed of light mechanism and the disconnection mechanism are not effective for all function models, and the decline in the later stage shows that Rationality of the SFA formulation

Table 3 (*continued*).

		SFA	PSO [12]	AOA [4]	GWO [40]	WOA [38]	EO [15]	HHO [23]
F16	Ave	**−1.3016**	**−1.3016**	**−1.3016**	**−1.3016**	**−1.3016**	**−1.3016**	**−1.3016**
	Std	8.5887e−09	6.6468e−16	2.4879e−11	1.8004e−08	1.3637e−10	6.1849e−16	4.7695e−11
F17	Ave	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**
	Std	3.6674e−05	0.0000	3.1854e−15	7.9053e−07	2.1220e−06	0.0000	1.3227e−06
F18	Ave	**3.0000**	**3.0000**	1.3022e+01	**3.0000**	**3.0000**	**3.0000**	**3.0000**
	Std	4.9673e−06	9.7225e−16	1.7826e+01	1.7673e−05	8.5129e−06	1.4496e−15	9.4235e−09
F19	Ave	−3.8625	−3.8625	−3.8627	−3.8617	−3.8611	**−3.8628**	−3.8621
	Std	0.0011	0.0014	2.7952e−04	0.0023	0.0023	2.3557e−15	0.0011
F20	Ave	−3.2294	−3.2534	**−3.2703**	−3.2655	−3.2106	−3.2576	−3.1252
	Std	0.1107	0.0720	6.0282e−02	0.0698	0.0797	0.0573	0.1257
F21	Ave	**−10.1531**	−8.0516	−7.2183	−9.6452	−9.2160	−8.9855	−5.2238
	Std	1.2311e−04	2.6529	3.1258	1.5462	2.1513	2.6852	0.9308
F22	Ave	**−10.4027**	−8.8053	−7.8623	−10.2258	−8.7848	−9.7499	−5.1847
	Std	2.4045e−04	2.7723	3.2292	0.9632	2.7875	2.0168	1.0629
F23	Ave	**−10.5363**	−9.4712	−6.3214	−10.2644	−8.6265	−10.3561	−5.6327
	Std	6.6312e−05	2.4760	3.5528	1.4813	3.0473	0.9873	1.5467

during the development phase. The rapid convergence at the initial stage of iteration on F5 and F6 verifies the effectiveness of the strategy proposed in this paper. Although the convergence accuracy of F7 is not as good as that of HHO and AOA, it is similar in magnitude.

In Fig. 8.2, SFA has good performance on multimodal functions. It can be seen from F8–F23 that the convergence accuracy and convergence speed of the algorithm are better than other algorithms. In F10, although SFA fell into the local optimum in the middle of the iteration, it can still jump out of the local optimum in the later stage, which also shows that the strategy of the algorithm in the development stage is effective.

4.5. Practical engineering problem tests

In this section, the proposed SFA optimizer is used to solve three classical engineering optimization problems with practical application significance, including three bar truss design problem, pressure vessel design problem and tension/compression spring design problem. However, most engineering optimization problems are constrained by special conditions such as design specifications, safety requirements and resource constraints. The goal of dealing with constrained optimization problems is to find a feasible solution that can verify the actual performance of the algorithm. It is important to note that the three engineering problems are 30 independent run tests based on the SFA algorithm, with the same test conditions guaranteed: 30 search agents and 500 iterations.

(1) Three-bar truss design problem

The structural model for this engineering problem, shown in Fig. 9, contains two design variables: the area of bar 1 and bar 2 (A1 and A2) and the area of bar 3 (A3). The optimization goal of this engineering is to minimize the total weight of the structure under constraints of stress, deflection and buckling.

Consider:

$$\text{Consider}: \overrightarrow{X} = [x_1 x_2] = [A_1 A_2]$$

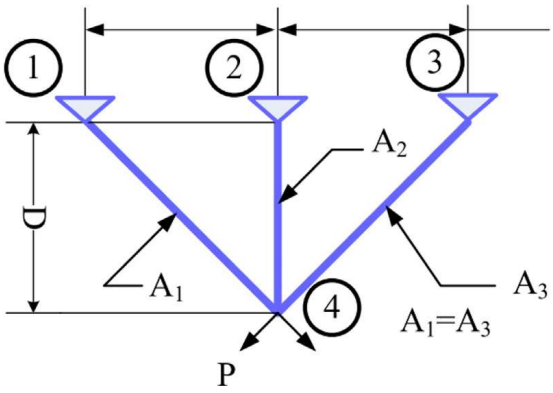


Fig. 9. Three-bar truss design problem.

Table 4
Optimization results for three-bar truss design problem.

Algorithms	x_1	x_2	Optimal weight
SFA	0.78800000	0.40800000	263.6800574299998
HHO [23]	0.7884536	0.4088751	263.8958794
CS [17]	0.78867	0.40902	263.9716
MVO [39]	0.78860276	0.408453070000000	263.8958499
SSA [37]	0.788665414	0.408275784444547	263.8958434
AOA [4]	0.79369	0.39426	263.9154
GJO [8]	0.788657163482708	0.408299125193296	263.8958439
RFO [44]	0.75356	0.55373	268.51195

$$\text{Minimize}: f(\vec{X}) = (2\sqrt{2}x_1 + x_2) \times l$$

$$\text{Subject to}: g_1(\vec{X}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0$$

$$g_2(\vec{X}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0$$

$$g_3(\vec{X}) = \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \leq 0$$

$$\text{where } l = 100 \text{ cm}, \quad P = 2 \text{ kN/cm}^2, \sigma = 2 \text{ kN/cm}^2$$

$$\text{Variable range}: 0 \leq x_1, x_2 \leq 1$$

Table 4 lists the optimization results of SFA and other algorithms for each variable and objective function in the three-bar truss design problem. The data in the table show that the optimized results of SFA are better than those of HHO [23], CS [17], Multi-Verse Optimizer (MVO) [39], Salp swarm algorithm (SSA) [37], Arithmetic optimization algorithm (AOA) [4], Golden jackal optimization (GJO) [8] and Red fox optimization (RFO) [44], which provide a good solution and show its strong competitiveness.

(2) Pressure vessel design problem

The purpose of pressure vessel design problem is to design a pressure vessel with the minimum production cost of f(∼x) under the premise of safety requirements. It contains four design variables: the thickness of the vessel shell and head (Ts and Th), the inner radius and length of the shell (R and L). The structural model diagram is shown in Fig. 10. The mathematical model is shown as follows.

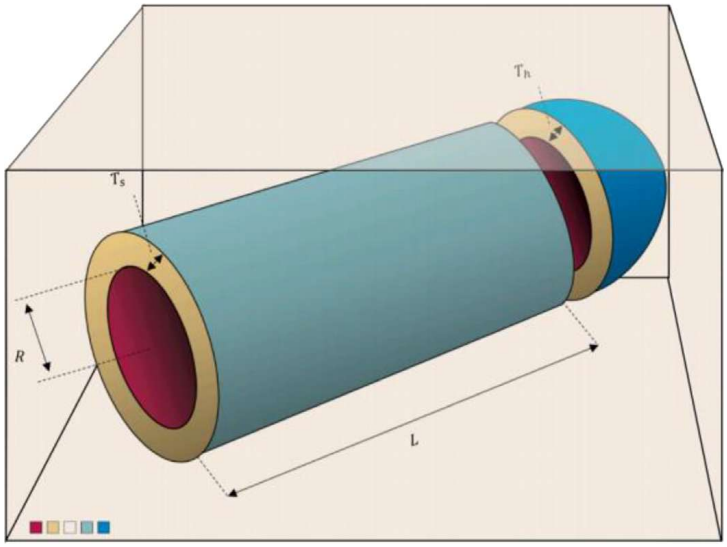


Fig. 10. Pressure vessel design problem.

Table 5
Optimization results for pressure vessel design problem.

Algorithms	$T_s(x_1)$	$T_h(x_2)$	$R(x_3)$	$L(x_4)$	Optimal cost
SFA	0.8000	0.4050	40.0000	170.0000	5390.4405
HHO [23]	0.8512	0.4224	44.08865	153.5456	6032.6746
PSO [12]	0.8125	0.4375	42.0892	176.7465	6061.0777
GSA [48]	1.125	0.625	55.9886598	84.4542025	8538.8359
GWO [40]	0.8125	0.4345	42.089181	176.758731	6051.5639
AOA [4]	0.8303737	0.4162057	42.75127	169.3454	6048.7844
WOA [38]	0.8125	0.4375	42.0893	176.6390	6059.7410
RFO [44]	0.81425	0.44521	42.20231	176.62145	6113.3195
RSA [3]	0.8401	0.4190	43.3812	161.5556	6034.7591

Consider:

$$\begin{aligned}
&\text{Consider}: \overrightarrow{X} = [x_1 x_2 x_3 x_4] = [T_s \quad T_h \quad R \quad L] \\
&\text{Minimize}: f(\overrightarrow{X}) = 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2 + 3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3 \\
&\text{Subject to}: g_1(\overrightarrow{X}) = -x_1 + 0.0193 x_3 \le 0 \\
&\qquad\qquad g_2(\overrightarrow{X}) = -x_2 + 0.00954 x_3 \le 0 \\
&\qquad\qquad g_3(\overrightarrow{X}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^2 + 1296000 \le 0 \\
&\qquad\qquad g_4(\overrightarrow{X}) = x_4 - 240 \le 0 \\
&\text{Variable range}: 0 \le x_1, x_2 \le 99, 0 \le x_3, x_4 \le 200
\end{aligned}$$

The optimization results of pressure vessel design problems by applying SFA and other algorithms such as HHO [23], PSO [12], GSA [48], GWO [40], AOA [4], WOA [38], RFO [44]and Reptile Search Algorithm (RSA) [3] are shown in Table 5.

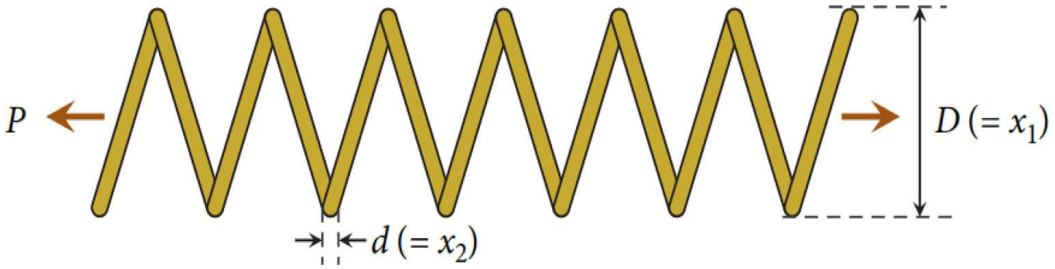


Fig. 11. Tension/compression spring design problem.

Table 6
Optimization results for tension/compression spring design problem.

Algorithms	$d(x_1)$	$D(x_2)$	$N(x_3)$	Optimal weight
SFA	0.051651	0.355737	11.35	0.01267
HHO [23]	0.051796393	0.359305355	11.138859	0.012665443
PSO [12]	0.051728	0.357644	11.244543	0.0126747
GSA [48]	0.050276	0.323680	13.525410	0.012702
GWO [40]	0.051690	0.356737	11.288850	0.012666
WOA [38]	0.0512	0.3452	12.0040	0.012676
SSA [37]	0.051207	0.345215	12.004032	0.012676
GJO [8]	0.0515793	0.354055	11.4484	0.01266752
RFO [44]	0.05189	0.36142	11.58436	0.01321

According to the result analysis, the solution of the proposed SFA is the best compared with other methods, which minimizes the production cost of pressure vessel and verifies its optimization ability.

(3) Tension/compression spring design problem

The aim of the tension/compression spring design is to minimize its weight f(x) under constraints, including three design variables, diameter d, average coil diameter D, and number of coils N. This problem has constraints of minimum detection, shear stress and surge frequency. The structure model is given in Fig. 11. The specific mathematical model is given as follows.

Consider:

$$\text{Consider}: \vec{X} = [x_1 x_2 x_3] = [D\ d\ N]$$

$$\text{Minimize}: f(\vec{X}) = (x_3 + 2) x_2 x_1^2$$

$$\text{Subject to}: g_1(\vec{X}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0$$

$$g_2(\vec{X}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \le 0$$

$$g_3(\vec{X}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0$$

$$g_4(\vec{X}) = \frac{x_1 + x_2}{1.5} - 1 \le 0$$

$$\text{Variable range}: 0.05 \le x_1 \le 2, 0.25 \le x_2 \le 1.3, 2 \le x_3 \le 15$$

The optimal results of SFA handling this problem and the solutions of other algorithms including HHO [23], PSO [12], GSA [48], GWO [40], WOA [38], SSA [37], GJO [8] and RFO [44] are recorded in Table 6. It can