AIMS Mathematics, 8(3): 5099–5123.
DOI: 10.3934/math.2023256
Received: 24 September 2022
Revised: 25 November 2022
Accepted: 01 December 2022
Published: 13 December 2022

http://www.aimspress.com/journal/Math

Research article

Improved salp swarm algorithm based on gravitational search and multi-leader search strategies

Xuncai Zhang[1,*], Guanhe Liu[1], Kai Zhao[1] and Ying Niu[2,*]

[1] School of Electrical and Information Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China
[2] School of Architecture Environment Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China

* **Correspondence:** Email: zhangxuncai@pku.edu.cn; niuying@zzuli.edu.cn.

Abstract: The salp swarm algorithm (SSA) will converge prematurely and fall into local optimum when solving complex high-dimensional multimodal optimization tasks. This paper proposes an improved SSA (GMLSSA) based on gravitational search and multi-swarm search strategies. In the gravitational search strategy, using multiple salp individuals to guide the location update of search agents can get rid of the limitation of individual guidance and improve the exploration ability of the algorithm. In the multi-swarm leader strategy, the original population is divided into several independent subgroups to increase population diversity and avoid falling into local optimization. In the experiment, 20 benchmark functions (including the well-known CEC 2014 function) were used to test the performance of the proposed GMLSSA in different dimensions, and the results were compared with the most advanced search algorithm and SSA variants. The experimental results are evaluated through four different analysis methods: numerical, stability, high-dimensional performance, and statistics. These results conclude that GMLSSA has better solution quality, convergence accuracy, and stability. In addition, GMLSSA is used to solve the tension/compression spring design problem (TCSD). The proposed GMLSSA is superior to other competitors in terms of solution quality, convergence accuracy, and stability.

Keywords: salp swarm optimization; gravitational search strategy; multi-leader search strategy; Wilcoxon's rank-sum test; tension/compression spring design problem
Mathematics Subject Classification: 93-08, 90C29, 65K10

1. Introduction

Because optimization problems are widely present in various research fields, this makes optimization technology develops rapidly. Since traditional optimization methods are easily affected by objective functions, constraint functions, and variable types[1–4], researchers have become increasingly interested in meta-heuristics [5,6]. The main characteristics of meta-heuristic methods are as follows: they are naturally inspired; random components are a necessary component of these methods; do not rely on gradient information. The meta-heuristic algorithm combines intelligent processes to guide the basic heuristic algorithm [7,8]. These algorithms are inspired by natural selection, physical phenomena, animal group habits, etc., and are used to solve optimization problems.

The swarm-based algorithm, also known as the swarm intelligence optimization method (SI) [9], is among the most famous examples of meta-heuristic algorithms. Because of its simple concept and easy programming, SI has been extensively applied in various practical engineering problems, such as wind turbine optimization [10], and fault diagnosis [11]. The most representative swarm intelligence methods are artificial bee colony algorithm (ABC) [12,13], ant colony optimization (ACO) [14,15], and particle swarm optimization (PSO) [16,17]. The idea of the PSO algorithm stems from the foraging behavior of bird swarms. The PSO algorithm has a simple structure and high computational efficiency. However, it tends to fall into the local optima in multimodal problems, and the exploration and exploitation capabilities are easily affected by the parameters. The ABC simulates the collective behavior of bees collecting honey. Because of its excellent global exploration capabilities, ABC has been used in various real-world optimization algorithms, such as portfolio optimization [18] and reliability optimization [19]. Based on No Free Lunch (NFL) [20], algorithms' effectiveness in one optimization problem cannot be extended to other optimization problems. In other words, no algorithm can be fully applicable to all optimization problems. Researchers are studying new swarm intelligence algorithms or improving proposed algorithms' performance to explore better solutions to artificial or real-world optimization problems. Recently, more advanced SI algorithms have been proposed, such as cuckoo search algorithm(CS) [21], grey wolf optimizer (GWO) [22], fruit fly optimization algorithm (FFOA) [23], ant lion optimizer (ALO) [24], crow search algorithm (CSA) [25], krill herd algorithm (KH) [26], firefly algorithm (FF) [27], squirrel search algorithm [28], slime mold algorithm [29]. These algorithms are inspired by cuckoo, grey wolf, fruit fly, ant lion, crow, krill, firefly, squirrel, slime mold swarm behavior.

The SSA is a novel SI method first proposed by Mirjalili in 2017, which was inspired by the chain movement and foraging behavior of salps in the ocean[30]. In SSA, the leader (the individual with the best fitness) in the swarm leads the follower (other individuals) to find food sources (global optima) to achieve the optimization process. The SSA's performance is tested on 19 traditional benchmark functions and 20 CEC 2015 function sets. The test results show that the SSA has higher accuracy and robustness in solving high-dimensional numerical optimization problems than PSO, gravitational search algorithm (GSA), bat algorithm (BA), and genetic algorithm (GA). Compared with the existing SI algorithms, SSA's principle is simple and fewer parameters need to be adjusted. Based on these advantages, since SSA was proposed, it has been widely applied to various practical engineering problems such as feature selection [29], parameter identification [31], and power dispatching [32].

Table 1. Some variants for SSA.

Variant	Algorithms	Method and Strategy	analyze
Modifications of SSA	CMSSA	Apply Gaussian and Cauchy mutations to the update process of candidate solutions.	It can significantly improve the detection capability of SSA, but the computational complexity is relatively high.
	E-SSA	Introduce adaptive weight and scale-free network mechanism.	It alleviates the slow convergence speed of the algorithm and the tendency of falling into the local minimum, but increases the computational complexity.
	MSSA	The original salp group is divided into multiple salp groups to independently perform global and local searches.	Significantly improve the global search capability of SSA, but more parameters need to be adjusted, and the robustness is poor
	RSSA	Using Refractive Reverse Learning Strategies	Improve the detection capability of SSA to avoid falling into local optimization and high-dimensional problems. The performance of RSSA is insufficient in terms of convergence speed and optimization accuracy
Hybridization of SSA	DMSSA	Introducing Cuckoo Mutation Strategy (CMS) and Adaptive DE Mutation Strategy (ADMS) into the structure of the original SSA	Improve the utilization rate of population information, and balance exploration and development
	HSSASCA	Combining SSA with sine cosine algorithm	Enhance the performance of the original SSA, which can be used for unconstrained and constrained optimization problems
Binary SSA	BSSA	Use eight transfer functions to represent the eight variants of binary SSA	Improve the global search capability of the original SSA
Chaotic SSA	CSSA	Replace the original random variable with the variable generated by the chaotic sequence	Improve the exploration ability and robustness of SSA, but the precision of CSSA is low when solving high-dimensional optimization problems
	CSCA	Use tent mapping to adjust leader movement in the population	Increase global search mobility for powerful global optimization
Conclusion	The above research generally exchanges for the optimization effect of exploration or exploitation capability, increasing the time complexity and lacking a balanced method between exploration and exploitation that has a significant impact on the overall performance of the algorithm. Also, these researches still have some shortcomings in solving problems with complex mathematics.		

Although SSA has achieved success in many areas, the algorithm itself still has some shortcomings. For example, the exploration capability of SSA is weak, and there is the problem of premature convergence[31,33,34]. Besides, the convergence rate of SSA is not high enough to obtain high-precision solutions for complex problems. Therefore, to improve SSA's global search capability, researchers have conducted a lot of research. For example, Sayed *et al*. proposed a novel Chao-induced SSA (CSSA) [35], and variables generated by chaotic sequences were used to replace original random variables. The test results on 34 benchmark problems show that CSSA can improve the exploratory ability and robustness of the SSA, but CSSA has low accuracy in solving high-dimensional optimization problems. Zhang et al. [36] applied Gaussian and Cauchy mutations to the update process of candidate solutions and used chaotic vectors to define food sources. Experimental results show that

CMSSA can significantly improve SSA's exploration capability. However, the computational complexity of CMSSA is relatively high, mainly because the amount of evaluations of the CMSSA function in each iteration is the square of the original SSA. In the study of Faris *et al*., eight transfer functions are used to represent eight variants of binary SSA, and the crossover operator is used instead of the average operator(BSSA), which improves the global search capability of the original SSA [37]. However, the exploitation of ssa is slightly inadequate.Yang *et al*. proposed a memetic salp swarm algorithm (MSSA) [38]. The algorithm divides the original salp swarm into multiple salp swarms to perform global and local searches independently. Simulation experiments and statistical results show that the MSSA can significantly improve the global search capability of SSA. However, MSSA needs to adjust more parameters and is less robust. El-Fergany *et al*. used the SSA to optimize the parameters of the fuel cell model [39]. The simulation results show that the improved algorithm effectively improves the accuracy of the fuel cell model. Fan *et al*. improved the original SSA by using a refracted oppositional learning strategy and proposed a refracted salp swarm algorithm (RSSA) [40]. This strategy can enhance the exploration capability of SSA and avoid falling into local optima. Experimental results show that RSSA is superior to other optimizers in terms of accuracy and robustness and achieves higher recognition accuracy in structural parameter recognition problems with a low signal-to-noise ratio. However, the performance of RSSA is inadequate in terms of convergence rate and optimization accuracy in high dimensions problems. Hegazy *et al*. added an inertia weight parameter to the original SSA to modify the leader and follower's position equation. This method improves the performance of SSA in solving the feature selection problem [41]. In the study of Wang et al. [42], two effective mechanisms, adaptive weight and scale-free network, were integrated into the following evolutionary process (E-SSA) of the salp algorithm, which alleviated the slow convergence speed of the salp swarm algorithm and the tendency of falling into local minima. However, changes in these strategies increased the computational complexity, but they do not always effectively obtain the exploration ability of the optimization algorithm. Nasri *et al*. introduced chaos (CSCA) into the salp swarm algorithm and used tent maps to adjust the attractive movement of the leader in the population around the food source, to increase the global search mobility and achieve strong global optimization [43]. Lin et al. added mutation structure to the original SSA, and introduced Cuckoo Mutation Strategy (CMS) and Adaptive DE Mutation Strategy (ADMS) into the structure of the original SSA to obtain a Double Mutation Salp Swarm Algorithm (DMSSA) [44]. In the former mutation, judgment, shuffling, and mutation act on leaders. The latter mutation selection, mutation, and adaptation act on followers to improve the utilization rate of population information, and also make some balance in exploration and development. Sigh *et al*. proposed a hybrid algorithm (HSSASCA) to enhance the performance of the original SSA by combining SSA with the sine cosine algorithm [45]. The convergence effect with global and local search is better than other comparative test algorithms and can be applied to unconstrained and constrained optimization problems. Table 1 shows some variations for SSA. However, the above research generally to exchange for the optimization effect of exploration or exploitation capability, increasing the time complexity and lacks a balanced method between exploration and exploitation that has a significant impact on the overall performance of the algorithm. Also, these researches still have some shortcomings in solving problems with complex mathematics. For example, there are premature convergence phenomena in highly complex multi-dimensional and multimodal optimization problems. This paper proposes an improved SSA (GMLSSA) based on gravitational search and multi-swarm search strategies to overcome these problems, which has the following two main improvements:

1) A gravitational search strategy is proposed, which replaces the original single individual guidance method with multiple individual coordinated guidance methods to update the follower salps' position. This strategy avoids the shortcomings of the single neighborhood topology of the traditional SSA and improves the algorithm's ability to explore.

2) based on the gravitational search strategy, a multi-swarm strategy is introduced. This strategy expands the single salp swarm of the traditional SSA into multiple independent salp swarms. Moreover, dividing the followers of all sub-swarms into ordinary followers and communication followers promotes the exchange of information between sub-swarms. In this way, a balance between the exploration and exploitation capabilities can be managed appropriately and avoid the algorithm falling into local optima.

The remaining sections are arranged as follows: Section 2 briefly introduces the traditional SSA. Section 3 describes the proposed GMLSSA, and Section 4 gives the experimental results and discussion. Section 5 summarizes the research conclusions of this paper.

2. An overview of the salp swarm algorithm

The SSA mimics the chain movement mode and dynamic foraging salp populations' behavior in the ocean. It is a new SI algorithm proposed by Mirjalili et al. [30]. The movement of salps in the sea has aggregation, usually forming a long salp chain. Many scholars believe that this chain-like structure can help salps to coordinate their activities quickly and effectively foraging. Individuals in a salp chain can be divided into leaders and followers, and the structure is shown in Figure 1. The leader is the individual at the top of the chain. It updates the position based on the food source. Therefore, the leader's exploration and exploitation are always carried out near the food source. The leader's movement model is as follows:

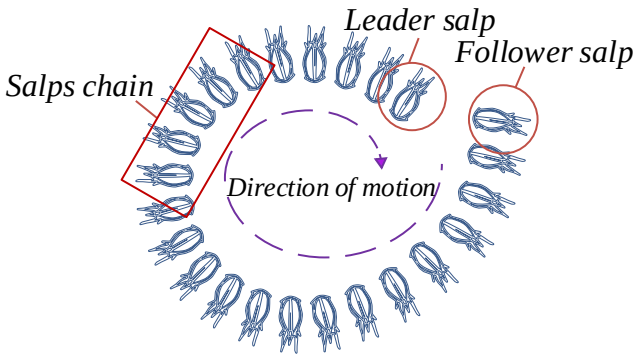

Figure 1. Swarm behavior of salps.

$$X_{1,j} = \begin{cases} Food_j + c_1\left((ub_j - lb_j)c_2 + lb_i\right) & c_3 \geq 0.5 \\ Food_j - c_1\left((ub_j - lb_j)c_2 + lb_i\right) & c_3 < 0.5 \end{cases} \tag{1}$$

where $X_{1,j}$ is the first salp position in the jth dimension, $Food_j$ represents the food source position in the j dimension. ub_j and lb_j are the upper and lower bounds of the jth dimension in the search space. c_2 and c_3 are random numbers uniformly distributed between 0 and 1. c_1 is a convergence factor, which balances the algorithm's exploration and exploitation capabilities in the iterative process.

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

where T is the maximum iteration, and t is the current iteration. From the formula (2), the value of the convergence factor c_1 adaptively decreases with iteration. At the beginning of the iteration, the convergence factor c_1 slowly decreases, and the leader leads the followers to conduct a large-scale global exploration. At the later stage of the iteration, the value of c_1 decreased significantly, and the leader carried out detailed exploitation of promising areas. Follower salps update their positions after the leader, and this equation can be expressed as follows:

$$X_{i,j} = \frac{(X_{i,j} + X_{i-1,j})}{2}, \ i \geq 2 \tag{3}$$

where $X_{i,j}$ is the i-th salp individual (follower) position vector on the jth dimension. When i=2, the follower's position update is directly related to the leader $X_{1,j}$, that is, the leader directly leads the follower. When i>2, the follower's position update is only related to the previous follower $X_{i-1,j}$, that is, the leader indirectly leads the follower.

The flow chart of SSA is shown in Figure 2.

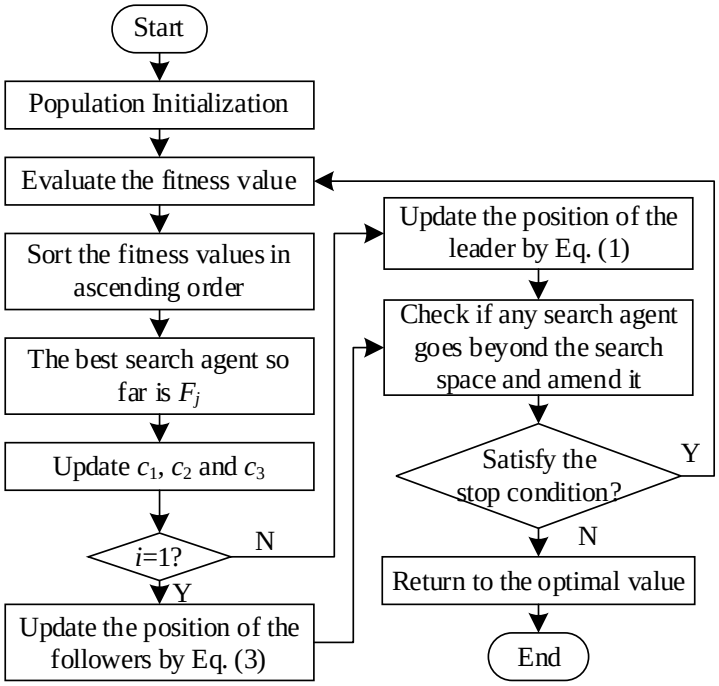

Figure 2. Flow chart of traditional SSA.

3. Improved salp swarm algorithm

3.1. Gravitational search strategy

In SSA, the only leader in the salp chain and the rest of the followers have a clear labor division. As the number of iterations increases, the leader gradually approaches the food source, and followers connect and follow the leader. From formula (3), the i-th individual in the salp swarm is updated

according to the (i-1)-th individual, where i = 1 represents the leader. Therefore, once the leader's position is determined, the search trajectories of other individuals are determined. This structure makes the salp populations only move around the leader without escaping the current search trajectory. When the leader falls into the local optima, it will mislead all followers into the local optima. Essentially, there is a lack of exploration ability in the follower's search equation. Inspired by the Gravitational Search Algorithm (GSA) [46], the gravitational search mechanism between particles is introduced into the salp swarm to guide the follower's search. This strategy effectively improves the search performance of SSA.

First, suppose that the salp swarm forms a chain structure according to the mass of the individual. The salp individual with the highest mass has the least fitness (for minimizing problems) and is considered the leader. According to Newton's law of gravity, the greater the salp individual's mass, the more attractive it is to other individuals around it. Also, the closer the distance between the two individuals, the greater the gravitational force. Therefore, in the t-th generation, the gravitational force of individual j on individual i is defined as follows:

$$F_{ij,d}^t = G\frac{Mass_i^t \times Mass_j^t}{R_{ij}^t}\left(x_{j,d}^t - x_{i,d}^t\right) \tag{4}$$

where $Mass_i^t$ and $Mass_j^t$ are the inertial masses of salp individuals i and j, the mass is calculated according to fitness. The updated formulas are as formulas (6) and (7); R_{ij} is the Euclidean distance between two salp individuals i and j, such as $R_{ij,d}^t = \left\|X_i^t, X_j^t\right\|_2$. G is the gravitational constant of the t-th generation, and its calculation formula is:

$$G = G_0 \times e^{-\frac{\delta t}{T}} \tag{5}$$

where T is the maximum iteration, δ is the attenuation coefficient with a value of 20. G_0 is the initial gravitational constant, and when its value is 100, the algorithm's optimization ability is more stable.

$$m_i^t = \frac{fit_i^t - worst^t}{best^t - worst^t} \tag{6}$$

$$M_i^t = \frac{m_i^t}{\sum_{j=1}^N m_j^t} \tag{7}$$

where fit_i^t represents the fitness of the individual i of salp in the t generation.

Second, the salp swarm must have enough opportunities to expand the search space to avoid falling into local optima. Different from the single-individual guidance method in SSA, in our method, each salp individual will be attracted by all individuals with greater mass than its own. Therefore, the location update of each salp individual is guided by multiple individuals in the salp swarm, which increases the exploration capability of SSA. In dimension d, the total force of salp individual i is the resultant force of the forces exerted by individuals with greater mass than its own. Therefore, for the follower salps (i≥2), the resultant force can be calculated by the following formula:

$$F_{i,d}^t = \sum_{j=1, j\neq i}^N rand_j F_{ij,d}^t \tag{8}$$

where $rand_j$ represents a random number between [0, 1], it is worth noting that if j is equal to i, the attractive force F_{ii} is set to 0 to avoid selecting the current individual itself.

Finally, according to Newton's second law, the i-th salp's acceleration in the d-th dimension is shown in formula (9). Also, for the follower salps ($i \geq 2$), the speed and position update equations are shown in formulas (10) and (11), respectively.

$$a_{i,d}^t = \frac{F_{i,d}^t}{Mass_i^t} \tag{9}$$

$$v_{i,d}^{t+1} = rand_i + v_{i,d}^t + a_{i,d}^t, i \geq 2 \tag{10}$$

$$X_{i,d}^{t+1} = X_{i,d}^t + v_{i,d}^{t+1}, i \geq 2 \tag{11}$$

where $rand_i$ represents a random number between [0, 1].

It is worth noting that when individuals with higher masses are in the salp swarm, other individuals will move towards them, making the algorithm efficiently converge to the optima. Besides, the effect of gravity does not require any propagation medium, and all salp individuals will be attracted by other individuals of greater mass regardless of distance. Therefore, the proposal of the gravitational search strategy makes the SSA have a more robust exploration performance. The schematic diagram of the gravitational search strategy is shown in Figure 3.

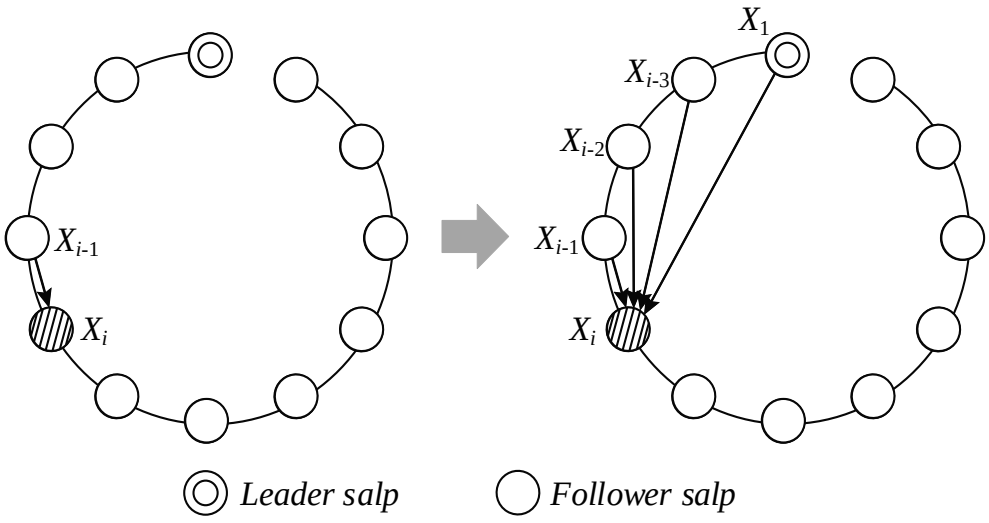

Figure 3. Schematic diagram of the gravitational search strategy.

As shown in Figure 3, the salp individual X_i is only affected by the neighbor's individual X_{i-1} in the traditional SSA. This topology has a low degree of freedom, resulting in low exploration efficiency. In the SSA with a gravitational search strategy, the salp individual X_i will be attracted by all individuals with better fitness when the search process is in the exploration stage. In this way, each exchanges information with at least one outstanding individual. This improved topology means that the swarm can obtain a higher level of diversity than the SSA's leader-follower structure. Therefore, it is not surprising that it performs better on complex multimodal problems.

The pseudo-code of the gravitational search strategy is provided in Algorithm 1.

Algorithm 1. Gravitational search strategy (GSS)

Input: Maximum iteration T, the total number of salps N, parameter G_0, δ.
Output: $X_{i,d}^{t+1}$

Initialize the search agent population $X_{i,d}^{t}(i = 1, 2,\ldots, N)$
for each salp ($X_{i,d}^{t}$) **do**
  if $X_{i,d}^{t}$ is the leader **then**
    Update the leading salp's position using Eq (1)
  else
  Updates the G, $best^t$, $worst^t$ and $Mass_i^t$ of the population
  Compute the total forces in different directions with the Eq (8)
  Find the accelerations and velocities with the Eq (9) and Eq (10)
  Update the follower salp's position $X_{i,d}^{t+1}$ using Eq (11)
  end if
end for

3.2. Multi-leader strategy

In the SSA, only one leader is responsible for the food search, while other individuals in the population only follow the previous individual, which means that the SSA has lower population diversity[39,40]. When solving more complex problems, this strategy tends to fall into local optima. In reality, salp swarm formation is through the independent feeding of multiple small-scale salp swarms, and they gradually gather together according to a specific method [47].

Based on this, a multi-leader strategy is proposed. In this strategy, the salp swarm is divided into multiple sub-swarms led by different leaders, and each sub-swarm uses the GSS topology to perform search tasks independently. Because the population is divided into various leaders to search the solution space in parallel, it encourages the population to explore more and helps maintain the population's diversity during the initial stage of the iteration. Also, the followers in each subgroup are divided into ordinary followers and communication followers. Ordinary followers follow the previous individual in the salp chain in the traditional way, while communication followers leave their subgroups to find a source of food (global optima). This effectively promotes the exchange of information between sub-swarms.

First, the original single salp swarm is divided into several sub-swarms according to population fitness. For example, We can divide the salp swarm with a population size of M*Q into M sub-swarms according to specific rules. As shown in Figure 4, the best individual is assigned to salp swarm #1, and the second-best individual is assigned to salp swarm #2, and so on. It is worth noting that the best individual in each sub-swarm is also considered the sub-swarm leader, and the remaining individuals are followers. Based on this idea, all salp swarm individuals are no longer constrained by a single leader, but multiple leaders independently guide their sub-swarm.

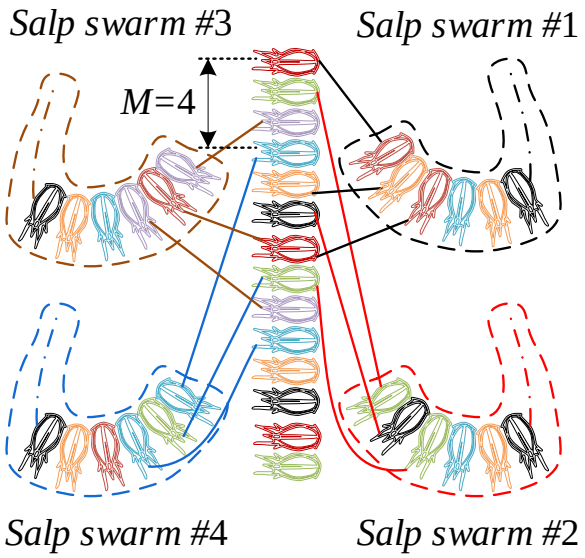

Figure 4. Partition rules of sub-swarms.

Algorithm 2 Multi-leader strategy (MLS)

Input: Maximum iterations T, the total number of salps N, parameter M, Q.
Output: $X_{i,d}^{t+1}$
01: Initialize M*Q salp individuals. Divide the salp swarm with a population size of M*Q into M sub-swarms. Each sub-swarm contains Q salp individuals.
02: **for** t=1 to T **do**
03: **for** i=1 to M * R **do**
04: Generate a random number between [0, 1].
05: **if** *rand*< t/T **then**
06: Update the follower salp's position $X_{i,d}^{t+1}$ using Eq (12)
07: **else**
08: Update the follower salp's position $X_{i,d}^{t+1}$ using Eq (11)
09: **end if**
10: **End for**

Second, divide the followers of all sub-swarms into ordinary followers and communication followers. Ordinary followers are responsible for their evolution under the leadership of the sub-swarm leader, and the leader still controls their search methods. In contrast, communication followers can break through the control of the sub-swarm leader and play a role in information exchange between sub-swarms. We define individuals with probability 1-p as ordinary followers and update their positions according to the formula (11). Individuals with probability p are defined as communication particles, and information exchange is realized according to the position update formula (12). This strategy sets up a dynamic control mechanism in the search process to realize the classification of followers. As mentioned above, the two types of followers' division are based on the probability $p \in [0, 1]$. When p=0, all followers have not evolved into communication followers, and individual location updates are performed traditionally. When p=1, all followers evolve into communication followers. When p increases from 0 to 1, the population gradually transitions from exploration to exploitation. Also, the parameter p is set as a function of iteration t: $p=t/T$ to control the classification mechanism. The p gradually increases with iteration, leading to the growth of communication salps. In this way, ordinary followers are continuously provided with opportunities to evolve into

communication followers, which has a continuous and effective impact on exploration and exploitation. The increasing number of communication individuals between each sub-swarm promotes exchanging information between the sub-swarm and increases the population's diversity. Algorithm 2 gives the pseudo-code of the multi-leader strategy.

$$X_{i,d}^{t+1} = \left(\frac{1}{M}\sum_{m=1}^{M} lbest_{m,d}^{t} - X_{i,d}^{t}\right) + \left(Food_{m,d}^{t} - X_{i,d}^{t}\right), i \geq 2 \tag{12}$$

where $lbest_m$ represents the leader of the n-th sub-swarm, the Food is the food source's position. It can be seen from formula (12) that ordinary followers do not completely rely on the leader for location update operations but evolve into communication followers to find food sources. This method avoids the shortcomings of traditional SSA. Besides, communication followers also realize the exchange of information between sub-swarms in finding food sources. A schematic diagram of the multi-leader strategy is shown in Figure 5. Algorithm 3 gives the pseudo-code of the GMLSSA.

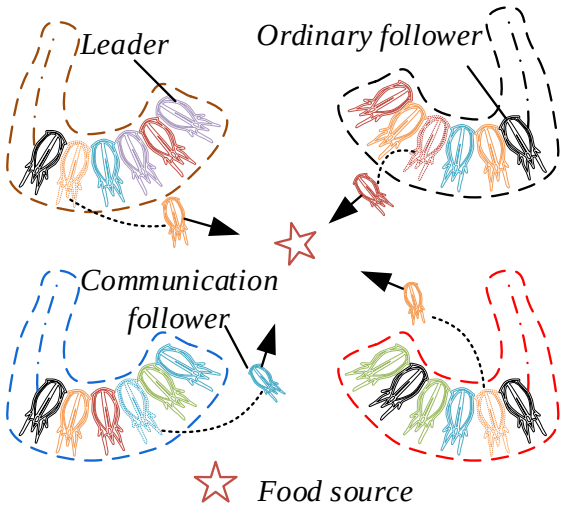

Figure 5. Schematic diagram of multi-leader strategy.

It is worth noting that in population initialization and population location update when the individual information in the population exceeds the constraint conditions of the decision variables, the information will be placed at the boundary of the constraint.

GMLSSA does not change the computational complexity of SSA. During the cycle, the computational complexity of evaluating the fitness of each search agent is $O(T{\cdot}N)$, and the computational complexity of updating the position vector of each search agent is $O(T{\cdot}N{\cdot}D)$. Finally, the computational complexity of GMLSSA is $O(T{\cdot}N{\cdot}D)$.

The flow chart of GMLSSA is shown in Figure 6.

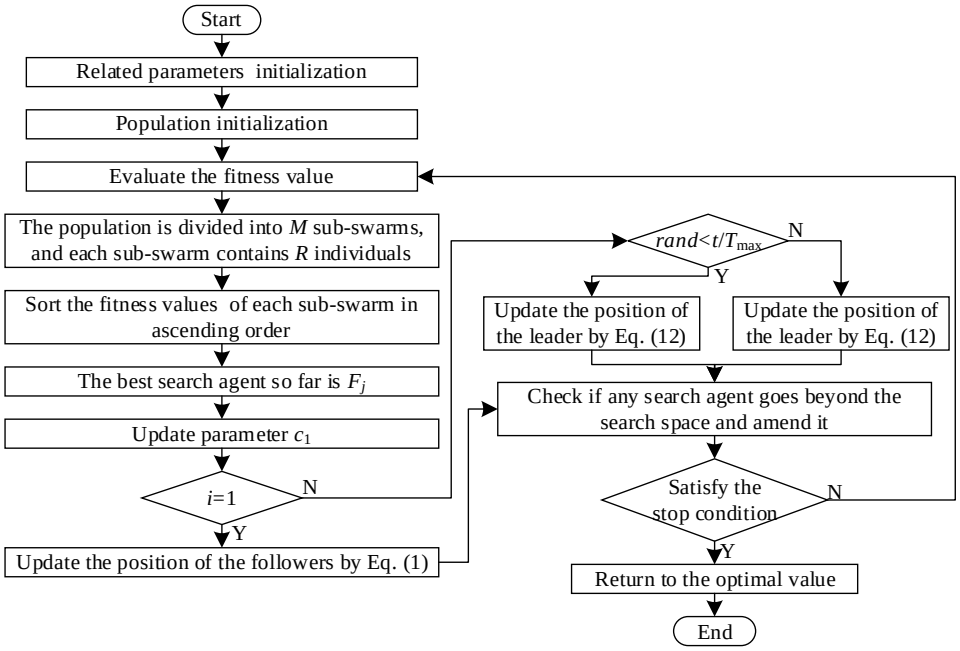

Figure 6. Flow chart of traditional GMLSSA.

Algorithm 3 Pseudo-code of GMLSSA

01: Initialize maximum iteration T, the total number of salps N, attenuation coefficient δ, initial gravitational constant G_0, sub-swarm size Q, and sub-swarm's number M;
02: Determine the fitness function according to the specific optimization problem;
03: Divide the salp swarm with a population size of $M*Q$ into M sub-swarms and each sub-swarm contains Q salp individuals;
04: Initialize swarm at first-generation;
05: Calculate each salp's fitness;
06: Sort all salps' fitness in descending order and the individual with the best fitness was selected as F;
07: Update parameter c_1;
08: **while** $t<T$ **do**
09: **for** i = 1 to M **do**
10: **for** j = 1 to R **do**
11: Evaluate the fitness of the jth salp in the ith salp swarm;
12: **end for**
13: **end for**
14: **for** i=1 to M **do**
15: Determine the food source of the ith salp swarm;
16: Update the position of the leader in the ith salp swarm by Eq (1);
17: **for** j = 2 to R **do**
18: Generate a random number between [0, 1];
19: **if** $rand< t/T$ **then**
20: Update the position of the jth follower salp in the ith salp swarm by Eq (12)
21: **else**
22: Update the position of the jth follower salp in the ith salp swarm by Eq (11)
23: **end if**
24: **end for**
25: Update the salps which go further than the search space limits based on the upper and lower limits of the problem variables
26: $t = t+1$
27: **End**

4. Experimental results and analysis

To evaluate GMLSSA's performance, the traditional SSA [40] and other state-of-the-art SI algorithms including the ABC [48], invasive weed optimization (IWO) [49], Hybridization Approach between the fireworks algorithm and grey wolf optimizer (FWAGWO) [50] and The improved SSA version (HSSASCA) [51] was used for comparative experiments. For the fairness of the experiment, all algorithms' population size is set to 50, and the maximum iterations are 1000. For the complex CEC 2014 function involved in the experiment, the maximum iterations are set to 6000 to obtain a sufficient number of function evaluations (NFE). For statistical analysis experiments, each benchmark function performs 30 independent runs to minimize the results' statistical error. The parameter settings of SSA, IWO, ABC, FWAGWO, and HSSASCA are consistent with the literature[40,48–51]. The parameters of the proposed GMLSSA are set as follows: attenuation coefficient δ=0.5, initial gravitational constant conversion G_0=1, number of sub-swarm M=5, each swarm's population R=10. The specific parameter settings are shown in Table 2. All algorithms were implemented using Matlab.

Table 2. Parameter settings of various algorithms.

Algorithm	Parameter setting
SSA	c_2, $c_3 \in [0,1]$, the parameter c_1 adaptively decreases with iteration.
IWO	Initial population size $N_{initial}$ = 10, initial standard deviation $\sigma_{initial}$ = 10, final standard deviation σ_{final} = 0.02, nonlinear modulation index n = 3.
ABC	Number of Onlookers $nOnlooker$ = 0.25, expansion coefficient acc = 1.
FWAGWO	The parameter a linearly decreased from 2 to 0.
HSSASCA	Parameter c_1, c_2, $c_3 \in [0,1]$, $r_1=2\pi\times rand()$, $r_2=2\times rand()$, v_0=0 and I_{runs}=30.
GMLSSA	Attenuation coefficient δ =0.5, initial gravitational constant conversion G_0 =1, number of sub-swarm M=5, each swarm's population R=10.

4.1. Benchmark function

Table 3 shows all the benchmark functions used in this experiment. These benchmark functions include three categories: unimodal functions (F_1- F_6), multimodal functions (F_7- F_{12}), and complex composite functions (F_{13}- F_{20}). Among them, the unimodal functions are used to test algorithms' convergence rate and exploitation capability. The multimodal functions contain multiple local optima, and as the dimension of the problem increases, the number of local optima increases exponentially. Therefore, these multimodal functions are suitable for further evaluating the algorithm's exploration capability and global optimization performance. The composite functions used in this experiment are the CEC2014 benchmark functions. These functions have shifted, rotated, expanded, and combined the most complex numerical optimization problems. Therefore, solving such functions is the most challenging. These functions are used to evaluate the comprehensive optimization capabilities of the algorithm.

Table 3. Benchmark functions.

Function	d	Range	f_{min}	*Type*
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30/50/100	[-100,100]	0	Unimodal
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30/50/100	[-10,10]	0	Unimodal
$F_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{i} x_j)^2$	30/50/100	[-100,100]	0	Unimodal
$F_4(x) = max_i\{\lvert x_i \rvert, 1 \le i \le n\}$	30/50/100	[-100,100]	0	Unimodal
$F_5(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30/50/100	[-100,100]	0	Unimodal
$F_6(x) = \sum_{i=1}^{n} i{x_i}^4 + random(0,1)$	30/50/100	[-1.28,1.28]	0	Unimodal
$F_7(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30/50/100	[-500, 500]	-418.98×d	Multimodal
$F_8(x) = \sum_{i=1}^{n} [{x_i}^2 - 10\cos(2\pi x_i) + 10]$	30/50/100	[-5.12, 5.12]	0	Multimodal
$F_9(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} {x_i}^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30/50/100	[-32, 32]	0	Multimodal
$F_{10}(x) = \frac{1}{4000}\sum_{i=1}^{n} {x_i}^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30/50/100	[-600, 600]	0	Multimodal
$F_{11}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i+1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < a \end{cases}$	30/50/100	[-50, 50]	0	Multimodal
$F_{12}(x) = 0.1\{sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_1 - 1)^2[1 + sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30/50/100	[-50, 50]	0	Multimodal
$F_{13}(x)$ = (CEC1: Rotated High Conditioned Elliptic Function)	30/50/100	[-100, 100]	100	CEC 2014
$F_{14}(x)$ = (CEC2: Rotated Bent Cigar Function)	30/50/100	[-100, 100]	200	CEC 2014
$F_{15}(x)$ = (CEC8: Shifted and Rastrigin's Function)	30/50/100	[-100, 100]	800	CEC 2014
$F_{16}(x)$ = (CEC19: Hybrid Function 3 (N=4))	30/50/100	[-100, 100]	1900	CEC 2014
$F_{17}(x)$ = (CEC20: Hybrid Function 4 (N=4))	30/50/100	[-100, 100]	2000	CEC 2014
$F_{18}(x)$ = (CEC21: Hybrid Function 5 (N=5))	30/50/100	[-100, 100]	2100	CEC 2014
$F_{19}(x)$ = (CEC24: Composition Function 2 (N=3))	30/50/100	[-100, 100]	2400	CEC 2014
$F_{20}(x)$ = (CEC25: Composition Function 3 (N=3))	30/50/100	[-100, 100]	2500	CEC 2014

4.2. Numerical analysis

To ensure the fairness of the experiment, when the dimension of the test function is 30, each algorithm runs 30 times on 20 benchmark functions independently. The specific results are shown in Table 4 (the best solution (best), average (Mean), and standard deviation (Std) of the experimental results, which are recorded (the best results are highlighted in bold). For the minimum problem, the smaller the mean, the better the optimization ability of the algorithm. The smaller the standard deviation, the better the algorithm stability.

From Table 4, for unimodal functions F_1-F_6, the proposed GMLSSA obtains the best average and standard deviation on 5 unimodal functions (F_1, F_2, F_3, F_4, and F_6). For F_5, the results of the GMLSSA are not as good as HSSASCA and SSA in terms of average values. For most unimodal functions, GMLSSA has the smallest standard deviation. For most unimodal functions, GMLSSA has the smallest standard deviation. These results indicate that GMLSSA exhibits better stability in the unimodal functions. Therefore, the experimental results prove that the GMLSSA has better exploration capability.

Table 4. Experimental results of benchmark functions (F_1-F_{20}) with D=30 dimensions using IWO, FWAGWO, ABC, HSSASCA, SSA, and GMLSSA.

Function		IWO	FWAGWO	ABC	HSSASCA	SSA	GMLSSA
F_1	Best	2.6081E-48	1.8593E-79	2.3918E-63	0.0000E+00	1.2313E-75	0.0000E+00
	Mean	3.8046E-42	7.4525E-78	8.9267E-60	0.0000E+00	2.9299E-72	**0.0000E+00**
	Std	1.5178E-42	8.0547E-78	9.4909E-60	0.0000E+00	4.6601E-72	**0.0000E+00**
F_2	Best	2.3359E-22	9.0727E-25	3.0143E-48	0.0000E+00	6.2682E-42	0.0000E+00
	Mean	4.0453E-21	7.7868E-24	1.3308E-43	0.0000E+00	9.9800E-39	**0.0000E+00**
	Std	9.8118E-20	4.1931E-25	6.5932E-43	0.0000E+00	9.5909E-39	**0.0000E+00**
F_3	Best	1.0403E-18	2.8120E-21	5.5415E-53	1.6417E-66	3.2586E-39	4.0777E-126
	Mean	3.0751E-17	3.1214E-19	2.3359E-52	8.6639E-65	1.2290E-38	**7.5697E-122**
	Std	7.3926E-17	2.9378E-20	1.4448E-52	6.6679E-66	1.7526E-37	**2.1285E-121**
F_4	Best	9.4909E-15	2.9836E-34	7.9027E-21	2.3359E-48	3.6801E-112	1.1973E-137
	Mean	1.5047E-14	7.0597E-33	8.5656E-20	5.0046E-46	6.8733E-111	**3.9791E-136**
	Std	5.9393E-15	6.7752E-31	6.7752E-21	1.0917E-47	5.7649E-110	**3.0237E-136**
F_5	Best	4.2575E-06	1.6299E-07	5.4987E-07	**5.4745E-13**	5.1202E-09	7.4120E-08
	Mean	7.3791E-05	4.5464E-06	3.5138E-06	**3.7815E-12**	9.7448E-08	7.9302E-07
	Std	7.7337E-06	7.1975E-07	7.6589E-05	1.5325E-13	5.2053E-08	**7.4969E-14**
F_6	Best	8.2002E-03	6.0786E-02	9.5941E-06	6.8284E-08	8.3166E-12	5.2093E-19
	Mean	3.6341E-02	1.7895E-01	7.2428E-05	5.3984E-07	9.8026E-11	**7.2136E-18**
	Std	6.8152E-03	6.0501E-02	1.1931E-06	8.9524E-08	4.9335E-10	**6.0192E-18**
F_7	Best	5.9504E-02	8.6871E-03	6.2144E-03	7.4032E-05	2.5664E-03	6.3857E-04
	Mean	8.9235E-01	9.0543E-02	5.5163E-02	7.6882E-04	7.3455E-02	2.4619E-03
	Std	8.8516E-02	5.4971E-03	8.7123E-03	1.0964E-05	3.4416E-02	2.3875E-03
F_8	Best	1.2894E+01	4.4875E+01	5.5275E+00	8.0633E-07	8.6749E-14	**0.0000E+00**
	Mean	8.0064E+02	6.2205E+02	8.2599E+01	6.0001E-07	3.2434E-13	**0.0000E+00**
	Std	1.5505E+02	1.7831E+01	8.3875E+00	3.3711E-07	6.3708E-13	**0.0000E+00**
F_9	Best	4.8941E-05	1.3028E-11	4.2700E-06	7.7839E-14	4.4092E-13	6.6229E-12
	Mean	3.9468E-04	4.1276E-10	5.4626E-05	**9.7331E-13**	8.7804E-12	6.0738E-11
	Std	5.5993E-05	5.2911E-10	8.1977E-05	**2.7352E-13**	8.0281E-13	8.0774E-12
F_{10}	Best	4.1144E-08	3.8551E-11	9.9557E-14	6.3544E-14	2.3831E-12	**5.9966E-32**
	Mean	3.0542E-07	3.1985E-10	1.5129E-13	7.1096E-13	8.6988E-11	**8.5321E-31**
	Std	4.8943E-08	7.7261E-11	8.4553E-13	2.2311E-14	2.5847E-10	**6.4589E-31**
F_{11}	Best	9.8049E-03	8.0633E-07	1.9977E-04	6.2854E-08	9.4361E-11	9.0467E-19
	Mean	3.0223E-02	6.0001E-07	3.7533E-03	7.6095E-07	4.3303E-10	**9.3678E-18**
	Std	4.5166E-03	3.3711E-07	3.9525E-03	1.6296E-08	9.6364E-11	**3.9176E-18**
F_{12}	Best	6.4982E-02	3.7126E-04	9.7395E+00	5.2254E-06	4.6799E-07	3.9127E-09
	Mean	2.7038E-01	3.8022E-03	5.5969E+01	9.8782E-05	2.2053E-07	**5.7973E-08**
	Std	8.1339E-01	6.1099E-04	3.0966E+00	6.5066E-06	1.6212E-06	**1.5533E-09**
F_{13}	Best	2.3482E+02	3.4194E+03	5.3809E+03	2.3281E+02	2.4153E+02	2.2501E+02
	Mean	7.8951E+02	5.3553E+03	5.9991E+03	3.0937E+02	3.3705E+02	**2.4461E+02**
	Std	7.0729E+03	2.574E+03	9.9176E+02	3.0069E+02	2.5272E+02	**2.9251E+02**
F_{14}	Best	1.5807E+04	9.2451E+04	4.9462E+05	2.7958E+03	3.2078E+02	4.1263E+02
	Mean	9.9361E+04	8.2934E+05	6.1345E+05	4.7618E+03	2.6954E+03	**3.1796E+03**
	Std	5.1871E+03	1.4931E+04	2.7738E+05	4.2665E+04	6.1027E+03	**7.8993E+03**
F_{15}	Best	8.9231E+05	6.8772E+03	1.7763E+04	1.0494E+04	1.8971E+03	1.9772E+03
	Mean	4.8673E+06	7.1543E+04	1.7999E+05	1.9671E+05	**2.2237E+03**	2.5811E+03
	Std	7.2773E+05	5.3726E+03	5.9354E+05	7.2915E+04	**2.3721E+03**	2.9373E+04
F_{16}	Best	5.9768E+05	1.9318E+06	4.0681E+04	3.0714E+04	4.5712E+05	1.9679E+03
	Mean	6.6879E+06	3.1788E+08	3.3792E+05	4.6051E+05	5.3261E+06	**2.1421E+04**
	Std	6.8161E+06	2.7297E+07	7.4911E+05	1.8512E+05	9.6331E+06	**2.7796E+04**
F_{17}	Best	8.0495E+05	4.0814E+06	6.7969E+04	2.2392E+04	2.3261E+03	2.8577E+03
	Mean	9.5221E+06	9.0595E+06	8.6331E+04	7.1014E+05	2.5463E+03	**5.8231E+04**
	Std	4.0845E+06	3.4474E+07	5.5345E+05	9.1659E+05	8.9283E+04	**5.0558E+04**
F_{18}	Best	4.3167E+06	5.6387E+05	1.8402E+04	8.2811E+04	2.4549E+04	5.1566E+04
	Mean	7.1424E+06	6.2974E+06	8.9546E+06	2.2664E+05	**2.6029E+04**	9.5815E+05
	Std	7.1368E+06	6.7653E+05	5.2802E+06	9.6823E+06	**8.8775E+05**	3.4174E+05
F_{19}	Best	3.8917E+04	8.4372E+04	1.1043E+04	2.8206E+03	2.6176E+03	2.5604E+03
	Mean	5.3351E+05	8.363E+05	8.9571E+04	2.8237E+04	2.6138E+03	**2.5168E+03**
	Std	2.9921E+05	8.7512E+04	3.4528E+04	5.3012E+03	1.1543E+00	**2.5687E-10**
F_{20}	Best	2.8269E+04	5.2459E+03	4.5164E+03	3.5788E+03	3.1842E+03	2.8335E+03
	Mean	4.4951E+04	8.0381E+03	5.8928E+03	3.7683E+03	3.2123E+03	**2.8822E+03**
	Std	7.2506E+04	1.1702E+02	2.7486E+01	6.3667E+00	8.4287E+00	**7.8388E-02**

In the test experiment on the multimodal functions F_7-F_{12}, the mean and standard deviation of GMLSSA on F_8, F_{10}, F_{11}, and F_{12} are significantly better than other algorithms. Besides, the GMLSSA can obtain accurate optima on F_8 and F_{10}. GMLSSA only lost to the HSSASCA in F_9 but ranked second among all algorithms. GMLSSA has the smallest standard deviation on most multimodal functions (F_7, F_8, F_{10}, F_{11}, and F_{12}). Since multimodal functions contain multiple local minima, the algorithm must have a strong exploration ability in solving such functions. From the analysis of experimental results, the GMLSSA is also the best algorithm for multimodal function problems.

Among the 8 selected CEC2014 functions, GMLSSA beats other algorithms on 6 functions. Only the average value on F_{15} and F_{18} is inferior to the SSA. These experimental results prove that the GMLSSA still has strong stability and effectiveness in solving complex high-dimensional and multimodal optimization problems. This is due to the collaboration and information exchange of multiple leaders making the GMLSSA maintain high population diversity.

4.3. Convergence analysis

To show the convergence characteristics of the GMLSSA more clearly, this section records the average convergence curve of all algorithms on the 30-dimensional benchmark function (Figure 7).

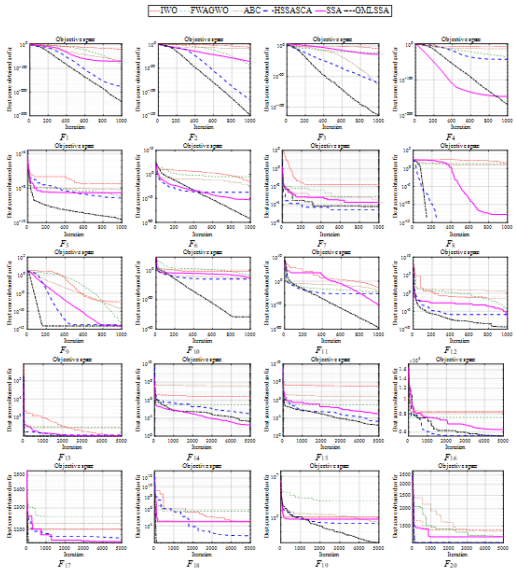

Figure 7. The convergence graphs of average best-so-far solutions obtained by GMLSSA and five heuristic algorithms on F_1-F_{20} with 30 dimensions.

GMLSSA has the fastest convergence speed and highest accuracy on most unimodal functions. Especially for F_1, F_2, F_3, and F_5, the GMLSSA converges to the optima very early.

On the multimodal function (F_7-F_{12}), The GMLSSA still has the fastest convergence speed on most multimodal functions and obtained the optima on F_8 and F_{10}. Only on F_7, the convergence rate of HSSASCA is slightly higher than that of GMLSSA. Therefore, GMLSSA still has good exploration capabilities.

For complex CEC 2014 functions, most algorithms such as IWO, ABC, and FWAGWO fall into local

optima, and the convergence rate is slow. However, the GMLSSA converges faster than other algorithms on F_{15}, F_{18}, and F_{19} and obtains higher accuracy. Even successfully obtain the optimal solution on F_{19}.

In summary, the experimental results prove the effectiveness of the proposed multi-leader search strategy and gravitational search strategy. The co-evolution among multiple salp swarms has expanded the search range and developed several valuable areas.

4.4. Stability analysis

To test the stability of the GMLSSA, this part studies and analyzes the distribution characteristics of the GMLSSA and the comparison algorithms in 30 independent experimental results. Randomly select the test results of 6 test functions (respectively from 2 unimodal, multimodal, and CEC2014 functions) to draw the box plots. As shown in Figure 8, the mean, worst, and best results obtained by GMLSSA are almost the same as the global optima, especially for F_3, F_7, F_{12}, and F_{20}. The box plot results prove that compared with other algorithms, the proposed GMLSSA has more powerful optimization capability and more excellent stability.

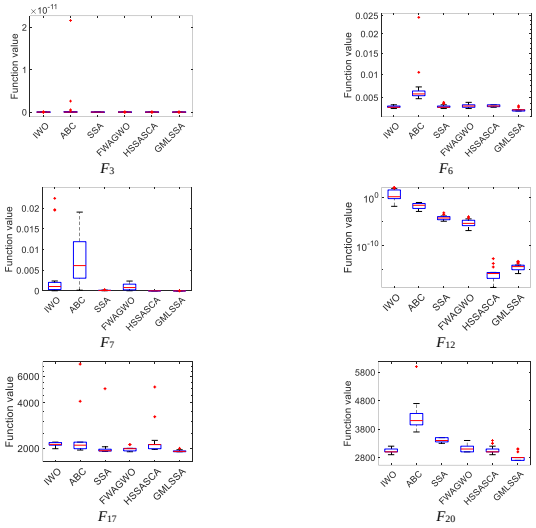

Figure 8. Boxplots on some selected functions.

4.5. High-dimensional performance analysis

To test the performance of GMLSSA on high-dimensional problems, we increased the dimensionality of the benchmark function from 30 to 50 and 100. All parameter settings remain unchanged and run independently 30 times on the benchmark functions. Table 5 records the simulation results of 20 benchmark functions of the six algorithms.

From Table 5, as the problem scale increases, it becomes more and more challenging for all algorithms to obtain the optima. However, the proposed GMLSSA has achieved good results in more than 14 benchmark functions. As the scale and complexity of the problem increase, GMLSSA still has higher accuracy and better robustness. As shown in Table 6, when the problem dimension increases to 100, GMLSSA achieves the best effect on 12 benchmark functions. These results also prove that GMLSSA still has high optimization performance on high-dimensional optimization tasks.

Table 5. Experimental results of benchmark functions (F_1-F_{20}) with D=50 dimensions using IWO, FWAGWO, ABC, HSSASCA, SSA, and GMLSSA.

Function		IWO	FWAGWO	ABC	HSSASCA	SSA	GMLSSA
F_1	Best	6.5625E-26	5.5869E-56	5.5483E-42	3.6977E-53	7.8577E-54	4.9187E-74
	Mean	9.7707E-24	2.9092E-55	3.3066E-41	2.9615E-51	2.1217E-52	4.0162E-71
	Std	6.2661E-24	6.8138E-55	4.9931E-42	3.1818E-51	3.1968E-52	8.5859E-71
F_2	Best	2.3722E-05	8.9659E-13	4.1913E-26	1.4164E-51	9.4418E-24	3.1153E-63
	Mean	5.5366E-03	9.7085E-12	5.2346E-26	5.0884E-50	5.9027E-23	7.7148E-62
	Std	9.1132E-03	6.9932E-11	2.4792E-26	8.2075E-51	2.7747E-22	6.4778E-62
F_3	Best	3.3984E-06	6.4612E-08	8.0628E-34	5.7192E-45	1.0816E-18	2.3861E-48
	Mean	2.4448E-04	2.7668E-06	3.4511E-31	4.2074E-43	7.4066E-17	2.4917E-42
	Std	1.8906E-04	8.6568E-05	4.4322E-31	4.5251E-42	1.2923E-17	6.8984E-41
F_4	Best	6.9927E-08	3.8649E-17	9.3002E-16	7.1154E-28	1.1628E-39	2.4427E-52
	Mean	3.9217E-06	5.4359E-15	9.9557E-14	3.4419E-26	3.3224E-38	7.0162E-51
	Std	5.3148E-06	7.9224E-15	3.8786E-12	5.4505E-27	9.9777E-38	2.3952E-50
F_5	Best	2.4681E-03	8.8824E-04	2.7534E-04	4.4005E-10	4.5366E-04	1.3812E-06
	Mean	8.6621E-02	4.4623E-03	2.2239E-03	**9.9229E-09**	8.2432E-05	7.4278E-05
	Std	9.2091E-03	6.5445E-04	4.6295E-02	**9.7557E-10**	8.4244E-08	8.8741E-05
F_6	Best	9.1757E-02	6.6024E-01	2.3389E-04	4.7318E-06	3.7363E-09	6.3534E-16
	Mean	9.6208E-01	4.3808E+00	5.8323E-03	4.4059E-06	5.1158E-08	3.9032E-15
	Std	5.4815E-02	3.4015E-01	3.9863E-04	3.6296E-05	4.6117E-07	8.8052E-15
F_7	Best	3.1723E-01	5.9986E-02	5.4602E-02	7.1396E-04	7.5961E-02	2.1838E-03
	Mean	4.3701E+00	3.9698E-01	4.9704E-01	7.3411E-03	5.4504E-01	6.2766E-02
	Std	5.5444E-01	5.5391E-02	7.2187E-02	4.5624E-04	1.7984E-01	8.3074E-02
F_8	Best	8.5213E+03	8.9355E+02	2.3281E+01	9.3205E-28	6.8885E-11	4.9353E-31
	Mean	9.4955E+04	5.5752E+03	7.0235E+02	1.3612E-26	4.6528E-10	2.4808E-30
	Std	9.2205E+05	7.8922E+02	5.1623E+01	2.2996E-25	7.3361E-10	9.3749E-28
F_9	Best	8.1573E-04	3.8392E-08	5.7682E-04	5.0566E-10	9.8749E-09	3.0731E-08
	Mean	7.2976E-03	3.0304E-07	7.7088E-03	**3.9392E-09**	9.6408E-08	6.3845E-07
	Std	2.6971E-04	4.2961E-07	1.5138E-03	**9.8881E-09**	1.5058E-09	3.2692E-08
F_{10}	Best	9.5193E-06	6.9743E-08	8.7291E-11	8.4918E-11	1.3318E-10	4.1468E-28
	Mean	7.3464E-04	8.9365E-07	2.9814E-10	1.6895E-10	7.8312E-09	8.9055E-26
	Std	8.3284E-04	3.0582E-06	7.8083E-10	8.2925E-11	3.5323E-08	5.5175E-26
F_{11}	Best	1.0335E-02	6.2432E-04	3.3467E-02	2.6992E-06	6.7225E-09	4.4421E-14
	Mean	2.2393E-01	7.5968E-04	5.8466E-01	2.1973E-05	4.8416E-08	1.7692E-13
	Std	2.7641E-02	7.3541E-04	5.9252E-01	8.7461E-05	7.4698E-08	2.7443E-13
F_{12}	Best	5.9113E+00	1.3262E-07	2.6539E+01	3.1541E-04	9.4644E-05	2.2238E-03
	Mean	2.7465E+01	**4.8938E-06**	9.5758E+02	6.5907E-03	8.0103E-05	4.1045E-02
	Std	4.4378E+00	**5.6771E-07**	4.8528E+01	8.2349E-04	1.7171E-04	8.9537E-03
F_{13}	Best	3.6232E+02	5.7344E+03	6.8169E+03	3.8734E+02	4.3283E+02	5.7885E+02
	Mean	8.2417E+02	6.6395E+03	7.3368E+03	**4.0191E+02**	6.1476E+02	5.9028E+03
	Std	9.5695E+03	1.3227E+03	1.8318E+03	**2.7669E+02**	4.9701E+03	5.2799E+03
F_{14}	Best	2.0613E+04	9.9267E+04	5.1665E+05	3.6163E+03	4.9008E+02	5.0535E+02
	Mean	8.3161E+04	9.7345E+05	7.1442E+06	5.0369E+03	3.3269E+03	4.7412E+03
	Std	7.0224E+03	5.8417E+04	9.8754E+05	4.4241E+04	6.7299E+03	4.7128E+04
F_{15}	Best	9.9701E+05	7.5304E+03	3.9673E+04	3.4533E+04	3.0307E+03	3.7067E+03
	Mean	9.0618E+06	8.8663E+04	3.0318E+05	4.2056E+05	**4.5612E+03**	4.7108E+03
	Std	4.3823E+05	2.2288E+03	4.9066E+05	7.2162E+04	**4.3726E+03**	9.2597E+04
F_{16}	Best	6.7656E+05	3.0597E+06	5.0819E+04	4.9404E+04	5.4888E+05	3.1312E+03
	Mean	7.2346E+06	4.7219E+08	4.4285E+05	5.1041E+05	6.7541E+06	4.6512E+04
	Std	7.3735E+06	3.4076E+07	8.9683E+05	2.2639E+05	2.2376E+06	1.4962E+04
F_{17}	Best	8.5478E+05	5.9604E+06	7.7398E+04	3.5606E+04	3.0117E+03	3.4378E+03
	Mean	9.7044E+06	9.7223E+07	9.1678E+04	8.9731E+05	4.0568E+03	6.5791E+04
	Std	4.8381E+06	3.3982E+07	1.8378E+05	8.3413E+05	1.6421E+04	6.2723E+04
F_{18}	Best	5.7988E+06	6.0692E+05	2.0716E+04	9.1436E+04	3.3641E+04	6.3291E+04
	Mean	8.1442E+06	7.0712E+06	9.9547E+06	3.9243E+05	**3.2655E+05**	9.9447E+05
	Std	7.1232E+06	6.7044E+05	5.9107E+06	9.9644E+06	**3.6668E+05**	3.8345E+05
F_{19}	Best	4.1627E+04	9.7415E+04	2.5391E+04	2.5968E+03	4.7786E+03	4.0305E+03
	Mean	6.1216E+05	9.3754E+05	9.5933E+04	4.3539E+04	5.5272E+03	4.8299E+03
	Std	3.2172E+05	6.8492E+04	9.4336E+04	1.7031E+03	4.3527E+00	1.7068E-08
F_{20}	Best	4.9025E+04	7.0107E+03	6.8072E+03	5.9022E+03	4.5313E+03	4.2624E+03
	Mean	6.8215E+04	9.8124E+03	7.8591E+03	5.6549E+03	4.9402E+03	4.0431E+03
	Std	4.0513E+04	1.1134E+02	5.6432E+02	5.9431E+00	5.1874E+00	4.0975E+00

Table 6. Experimental results of benchmark functions (F_1-F_{20}) with D=100 dimensions using IWO, FWAGWO, ABC, HSSASCA, SSA, and GMLSSA.

Function		IWO	FWAGWO	ABC	HSSASCA	SSA	GMLSSA
F_1	Best	4.2986E-14	9.0076E-34	5.6571E-25	3.7772E-32	2.4246E-36	9.2857E-52
	Mean	1.7481E-12	4.0679E-33	4.1531E-24	8.9059E-31	6.7324E-34	1.7498E-50
	Std	8.1698E-12	4.6231E-32	8.2772E-24	1.1216E-31	3.3736E-34	4.4317E-50
F_2	Best	4.8451E-03	7.6312E-10	5.5201E-21	4.7364E-32	4.4834E-18	7.1913E-44
	Mean	7.8051E-02	3.3859E-09	2.1827E-20	4.4779E-30	7.2621E-16	5.4484E-41
	Std	7.0876E-02	8.7247E-08	6.8919E-20	6.9884E-30	9.9512E-16	2.9465E-40
F_3	Best	6.8101E-05	4.1929E-06	4.4779E-16	6.5178E-23	8.7187E-14	3.0686E-25
	Mean	1.8832E-03	8.0939E-04	4.3806E-14	4.4159E-22	2.6791E-13	9.3378E-24
	Std	7.6583E-03	4.9208E-03	5.7046E-14	1.9586E-20	8.0782E-13	9.1392E-22
F_4	Best	8.1046E-06	5.4634E-10	6.1525E-09	7.9804E-11	6.0274E-21	8.8006E-31
	Mean	3.5405E-04	6.0042E-10	8.4974E-08	5.6192E-10	6.8729E-19	1.7846E-30
	Std	2.4141E-03	8.8699E-09	7.1558E-08	2.3076E-10	7.0736E-19	5.3798E-30
F_5	Best	9.2952E-02	6.3237E-03	1.7623E-03	9.9396E-08	3.2621E-03	9.9384E-04
	Mean	3.8407E-01	6.1498E-02	6.2403E-02	**7.1916E-07**	6.5544E-04	4.4021E-03
	Std	9.1279E-02	3.9741E-03	9.4783E-01	**2.1358E-07**	5.6524E-08	8.6439E-03
F_6	Best	3.2139E-01	2.5448E+00	4.2298E-03	1.1659E-05	8.6186E-07	5.0535E-11
	Mean	8.8233E+00	6.0751E+01	1.1055E-02	1.0829E-05	1.3211E-06	9.3161E-10
	Std	3.2521E-01	4.9712E+00	8.2668E-03	9.5455E-04	1.8031E-07	5.8435E-10
F_7	Best	1.2154E+01	6.4072E-01	5.1931E+00	1.7061E-02	5.3657E+00	7.9623E+00
	Mean	2.2769E+01	**6.5643E+00**	5.7773E+01	9.8965E-01	1.5639E+01	8.5791E+01
	Std	6.8343E+00	**5.8762E-03**	6.2241E+00	8.9669E-01	4.8282E-01	7.1042E+00
F_8	Best	8.8241E+03	1.6801E+03	6.8394E+03	4.6917E-14	8.9556E-08	5.9551E-26
	Mean	4.5953E+04	2.6352E+03	7.1442E+03	8.5237E-13	3.9433E-07	9.4541E-24
	Std	3.8072E+02	9.7936E+03	1.7847E+02	7.0525E-12	8.0424E-05	4.9393E-23
F_9	Best	4.9262E-03	2.1156E-06	1.3066E-03	3.2388E-08	9.7621E-07	4.4023E-07
	Mean	1.2763E-02	2.2836E-05	7.8842E-02	**7.2577E-07**	4.3674E-06	8.7865E-06
	Std	6.1967E-03	1.0189E-05	6.3476E-02	**6.8478E-05**	2.5248E-04	9.9832E-05
F_{10}	Best	7.2513E-06	2.5342E-07	3.2845E-10	2.9439E-09	9.8874E-08	3.4152E-21
	Mean	7.1472E-04	6.2765E-06	9.8959E-09	4.0423E-08	4.9383E-07	9.2796E-20
	Std	3.8254E-04	8.6791E-05	4.4869E-09	8.4029E-06	5.6543E-05	5.1611E-18
F_{11}	Best	4.2237E-01	5.3848E-03	1.5835E-01	5.4483E-05	5.6423E-08	6.1647E-11
	Mean	6.4504E-01	7.5412E-03	6.0988E+00	2.6484E-04	8.4386E-07	2.0503E-10
	Std	6.1743E-01	4.5961E-02	9.1524E+00	2.2245E-04	5.8194E-05	8.3719E-10
F_{12}	Best	9.0567E+00	5.0146E-05	6.9793E+01	9.8428E-03	5.8585E-04	3.3267E-02
	Mean	6.6118E+01	**7.3783E-04**	7.1856E+02	3.5696E-02	9.7514E-04	2.7533E-01
	Std	2.6271E+00	**7.3322E-05**	6.8782E+00	7.7619E-03	2.2375E-03	4.2072E-01
F_{13}	Best	6.2848E+02	6.4736E+03	7.0481E+03	5.9585E+02	4.9146E+02	4.4053E+02
	Mean	6.5908E+03	2.0679E+04	8.2136E+03	6.5393E+03	6.8076E+02	5.5295E+02
	Std	2.4471E+02	1.7985E+02	1.9632E+02	6.5645E+03	3.9949E+03	3.3839E+02
F_{14}	Best	2.1032E+04	6.0802E+05	6.6072E+05	7.9744E+02	5.4718E+02	3.9891E+03
	Mean	9.9419E+04	9.5842E+05	8.5232E+06	**7.6321E+03**	4.2284E+03	4.2446E+03
	Std	3.1373E+03	7.2811E+04	9.1567E+05	**4.6275E+02**	8.1189E+02	6.8028E+02
F_{15}	Best	5.9299E+06	7.8622E+03	4.2711E+04	4.5759E+04	3.2709E+03	4.8343E+03
	Mean	8.6796E+06	9.2891E+04	4.3128E+05	5.0167E+05	**4.6244E+03**	5.5814E+03
	Std	2.1871E+05	4.8055E+03	1.8658E+04	9.9523E+03	**1.9864E+02**	4.8415E+05
F_{16}	Best	7.5666E+05	4.1268E+06	6.0756E+05	1.9356E+05	6.7949E+05	4.0709E+03
	Mean	7.8429E+06	8.5496E+08	7.1576E+05	8.9972E+05	1.0162E+07	4.8435E+04
	Std	6.3753E+05	2.7245E+05	6.3637E+05	4.1524E+04	8.6638E+04	8.7163E+03
F_{17}	Best	3.3427E+06	5.2524E+04	3.6387E+05	7.5465E+04	7.8017E+03	9.8447E+07
	Mean	9.7692E+07	**6.6661E+05**	2.6191E+05	2.0286E+06	6.9228E+03	9.1268E+07
	Std	6.6735E+06	**7.9546E+04**	1.3064E+04	5.2666E+05	3.3435E+04	2.7195E+06
F_{18}	Best	2.3545E+07	8.8332E+06	2.5913E+05	4.8306E+05	6.8061E+04	7.6244E+04
	Mean	5.5479E+06	6.2467E+07	3.5145E+07	7.8867E+05	**4.8241E+05**	3.3209E+06
	Std	4.8569E+05	7.8581E+06	8.7763E+06	2.4986E+04	**9.9825E+04**	4.6641E+04
F_{19}	Best	5.2581E+05	1.6266E+05	4.4188E+05	8.5506E+04	7.7727E+03	6.6683E+03
	Mean	5.4088E+06	3.2572E+06	7.0784E+05	4.2107E+05	7.6382E+03	6.6264E+03
	Std	8.6232E+06	6.9688E+05	2.3764E+05	1.3292E+05	2.2319E+02	7.1028E+00
F_{20}	Best	2.1444E+05	8.7627E+04	3.3055E+04	7.5205E+03	6.7764E+03	5.0894E+03
	Mean	6.9416E+05	6.6174E+04	2.2624E+04	7.3329E+03	6.8794E+03	6.5084E+03
	Std	6.9806E+05	1.9808E+03	5.9156E+03	3.8302E+03	8.6008E+03	7.7965E+01

4.6. Statistical analysis

Although the advantages of the GMLSSA have been proven through the mean, standard deviation, convergence, and stability, a significance test is still needed to prove whether there are significant differences between the proposed method and competitors [52]. Wilcoxon's rank-sum test [52] is a non-parametric statistical method used to detect significant differences between the two algorithms. It is performed at a significance level of 5%. Table 7 records the statistical results, and the symbol "+/=/-" indicates that GMLSSA is better, closer, or worse than the comparison method. From Table 7, the proposed GMLSSA is significantly different from other algorithms in most cases, and the results are 91/1/8, 93/1/6, and 94/1/5, respectively. The results show that with the increase of problem dimension, the results provided by GMLSSA are more significant, indicating the superiority of GMLSSA.

Table 7. Wilcoxon's rank-sum test results.

GMLSSA VS.		F_1-F_{20} (Dim=30)	F_1-F_{20} (Dim=50)	F_1-F_{20} (Dim=100)
Wilcoxon's rank sum test (+/=/-)	IWO	20/0/0	20/0/0	20/0/0
	FWAGWO	19/0/1	19/1/0	19/0/1
	ABC	18/0/2	20/0/0	19/1/0
	HSSASCA	16/1/3	16/0/4	18/0/2
	SSA	18/0/2	18/0/2	18/0/2
	Overall(+/=/-)	91/1/8	93/1/6	94/1/5

4.7. GMLSSA for tension/compression spring design problem

In recent years, stochastic optimization technology has become a hot spot in structural design research [53,54]. To test the effectiveness of GMLSSA in handling practical engineering optimization problems, we apply GMLSSA to an engineering design problem called the tension/compression spring design problem (TCSD) [55]. The optimization goal of this problem is to determine the best values of the three variables, namely wire diameter (d), mean coil diameter (D), and the number of active coils (P), to minimize the weight of the tension/compression spring. The specific structural parameters of the problem are shown in Figure 9, and the following equation gives its mathematical definition:

Consider $\vec{x}=[x_1\ x_2\ x_3]=[d\ D\ P]$,

Minimize $f(\vec{x}) = (x_3+2)\,x_2\,x_1^2$

Subject to *Disturbance*: $g_1(\vec{x})=1-\frac{x_2^3 x_3}{71785x_1^4}\leq 0$,

Shear stress: $g_2(\vec{x})=\frac{4x_2^3-x_1x_2}{12566(x_2x_1^3-x_1^4)}+\frac{1}{5108x_1^2}\leq 0$,

Fluctuation frequency: $g_3(\vec{x})=1-\frac{140.45x_1}{x_2^2x_3}\leq 0$,

Outside diameter: $g_4(\vec{x})=\frac{x_1+x_2}{1.5}-1\leq 0$,

Variable range: $0.05\leq x_1\leq 2.00$; $0.25\leq x_2\leq 1.30$; $2.00\leq x_3\leq 15.0$.

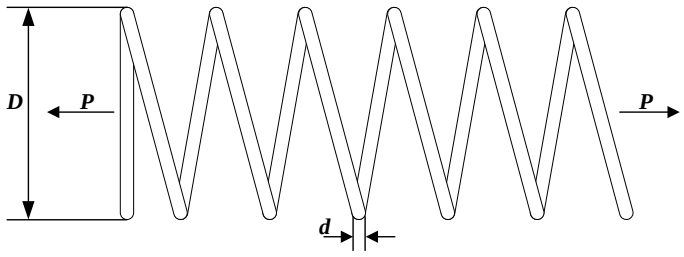

Figure 9. Tension/compression spring design problem.

In this experiment, GMLSSA ran 30 times independently and compared the best solutions obtained with IWO [45], FWAGWO [46], ABC [44], HSSASCA [47], and SSA [40]. The comparison results are shown in Table 8. Compared with IWO, FWAGWO, ABC, HSSASCA, and SSA, the optimization results of GMLSSA are the best among all algorithms. The design with the minimum weight is $\vec{x}$=[0.051621 0.35510 11.384638]，Minimizef($\vec{x}$)=0.012665.

The comparison results prove that the proposed GMLSSA is superior to the SSA and other SI algorithms. GMLSSA enhances global search capability and avoids local optima, which can effectively solve engineering design problems.

Table 8. Comparison results of various algorithms for TCSD problem.

Algorithm	Optimal values for variables			Optimal cost
	d	*D*	*P*	
GMLSSA	**0.051621**	**0.355100**	**11.384638**	**0.012665**
SSA	0.052021	0.364768	10.832300	0.012667
HSSASCA	0.051918	0.362248	10.971940	0.012666
ABC	0.051728	0.357644	11.244543	0.012674
FWAGWO	0.051690	0.356737	11.288850	0.012666
IWO	0.051989	0.363965	10.890522	0.012681

5. Conclusions and future work

To improve the SSA's performance on complex optimization problems, an improved SSA based on gravitational search and multi-leader search strategies(GMLSSA) is proposed. First, a gravitational search strategy is proposed. This strategy uses multiple individuals with better fitness to guide the current individual to search, eliminating a single individual's limitation to guide the current individual to search and improving the algorithm's exploration ability. Second, a multi-leader strategy is proposed. Divide the entire population into multiple sub-swarms, containing a leader and multiple followers to maintain the population's diversity. Moreover, dividing the followers in the sub-swarms into ordinary followers and communication followers to realize the sub-swarms' information exchange. In this way, independent sub-swarms can maintain information exchange through collaboration, thereby achieving a balance between exploration and exploitation.

The analysis of search behavior supports the advantages of GMLSSA in finding better solutions. Besides, the experimental results and statistical analysis of 20 benchmark functions also show that compared with other state-of-the-art algorithms and other improved SSA versions, GMLSSA has better quality solutions and stability. Also, the proposed GMLSSA is used to solve the TCSD problem. On this problem, the performance of GMLSSA is better than most competitors. Therefore, the proposed

GMLSSA has achieved significant improvements. However, the topology of GMLSSA algorithm leads to a slow convergence rate, and the gravitational search strategy inhibits the exploitation ability of the algorithm. Therefore, it is necessary to do more research on how to improve the efficiency of individual search. In addition, our future work also includes multi-objective optimization problems and practical engineering problems, such as vehicle path planning and workshop schedule.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 62072417, 62102374, and U1804262 and in part by the Henan provincial science and technology research project under Grants 202102210177 and 212102210028.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. T. Back, *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*, Oxford: Oxford university press, 1996.
2. H. Chen, Y. Xu, M. Wang, X. Zhao, A balanced whale optimization algorithm for constrained engineering design problems, *Appl. Math. Model.*, **71** (2019), 45–59. https://doi.org/10.1016/j.apm.2019.02.004
3. Y. Zhong, L. Wang, M. Lin, H. Zhang, Discrete pigeon-inspired optimization algorithm with Metropolis acceptance criterion for large-scale traveling salesman problem, *Swarm Evol. Comput.*, **48** (2019), 134–144. https://doi.org/10.1016/j.swevo.2019.04.002
4. A. Chakraborty, A. K. Kar, Swarm intelligence: a review of algorithms, *Nature-Inspired Comput. Optimiz.*, 2017, 475–494. https://doi.org/10.1007/978-3-319-50920-4_19
5. X. S. Yang, *Nature-inspired metaheuristic algorithms*, Beckington: Luniver press, 2010.
6. C. Özgüven, L. Özbakır, Y. Yavuz, Mathematical models for job-shop scheduling problems with routing and process plan flexibility, *App. Math. Model.*, **34** (2010), 1539–1548. https://doi.org/10.1016/j.apm.2009.09.002
7. H. Salimi, Stochastic fractal search: a powerful metaheuristic algorithm, *Know.-Based Syst.*, **75** (2015), 1–18. https://doi.org/10.1016/j.knosys.2014.07.025
8. P. Savsani, V. Savsani, Passing vehicle search (PVS): a novel metaheuristic algorithm, *Appl. Math. Model.*, **40** (2016), 3951–3978. https://doi.org/10.1016/j.apm.2015.10.040
9. Z. Cui, X. Gao, Theory and applications of swarm intelligence, *Neural Comput. Applic.*, 2012, 205–206. https://doi.org/10.1007/s00521-011-0523-8
10. A. Ribeiro, A. Awruch, H. Gomes, An airfoil optimization technique for wind turbines, *Appl. Math. Model.*, **36** (2012), 4898–4907. https://doi.org/10.1016/j.apm.2011.12.026
11. D. Binu, B. Kariyappa, RideNN: A new rider optimization algorithm-based neural network for fault diagnosis in analog circuits, *IEEE T. Instrum. Meas.*, **68** (2018), 2–26. https://doi.org/10.1109/TIM.2018.2836058

12. F. Wang, H. Zhang, K. Li, Z. Lin, J. Yang, X. L. Shen, A hybrid particle swarm optimization algorithm using adaptive learning strategy, *Inform. Sciences*, **436** (2018), 162–177. https://doi.org/10.1016/j.ins.2018.01.027
13. J. Chen, W. Yu, J. Tian, L. Chen, Z. Zhou, Image contrast enhancement using an artificial bee colony algorithm, *Swarm Evol. Comput.*, **38** (2018), 287–294. https://doi.org/10.1016/j.swevo.2017.09.002
14. R. Skinderowicz, Improving ant colony optimization efficiency for solving large TSP instances, *Appl. Soft Comput.*, **120** (2022), 108653. https://doi.org/10.1016/j.asoc.2022.108653
15. W. Deng, J. Xu, H. Zhao, An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem, *IEEE Access*, **7** (2019), 20281–20292. https://doi.org/10.1109/ACCESS.2019.2897580
16. J. Lu, J. Zhang, J. Sheng, Enhanced multi-swarm cooperative particle swarm optimizer, *Swarm Evol. Comput.*, **69** (2022), 1942–1948. https://doi.org/10.1016/j.swevo.2021.100989
17. D. Wang, D. Tan, L. Liu, Particle swarm optimization algorithm: an overview, *Soft Comput.*, **22** (2018), 387–408. https://doi.org/10.1007/s00500-016-2474-6
18. D. Kumar, K. Mishra, Portfolio optimization using novel co-variance guided Artificial Bee Colony algorithm, *Swarm Evol. Comput.*, **33** (2017), 119–130. https://doi.org/10.1016/j.swevo.2016.11.003
19. S. Ghambari, A. Rahati, An improved artificial bee colony algorithm and its application to reliability optimization problems, *Appl. Soft Comput.*, **62** (2018), 736–767. https://doi.org/10.1016/j.asoc.2017.10.040
20. D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, *IEEE T. Evol. Comput.*, **1** (1997), 67–82. https://doi.org/10.1109/4235.585893
21. X. S. Yang, S. Deb, Cuckoo search via Lévy flights, 2009 World congress on nature & biologically inspired computing (NaBIC), 2009, 210–214.
22. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Software*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
23. W. T. Pan, A new fruit fly optimization algorithm: taking the financial distress model as an example, *Know.-Based Syst.*, **26** (2012), 69–74. https://doi.org/10.1016/j.knosys.2011.07.001
24. S. Mirjalili, The ant lion optimizer, *Adv. Eng. Software*, **83** (2015), 80–98. https://doi.org/10.1016/j.advengsoft.2015.01.010
25. A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm, *Comput. Struct.*, **169** (2016), 1–12. https://doi.org/10.1016/j.compstruc.2016.03.001
26. A. H. Gandomi, A. H. Alavi, Krill herd: a new bio-inspired optimization algorithm, *Commun. Nonlinear Sci.*, **17** (2012), 4831–4845. https://doi.org/10.1016/j.cnsns.2012.05.010
27. X. S. Yang, Firefly algorithms for multimodal optimization, *International symposium on stochastic algorithms*, **5792** (2009). https://doi.org/10.1007/978-3-642-04944-6_14
28. M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, *Swarm Evol. Comput.*, **44** (2019), 148–175. https://doi.org/10.1016/j.swevo.2018.02.013
29. S. Li, H. Chen, M. Wang, A. A. Heidari, S. Mirjalili, Slime mould algorithm: a new method for stochastic optimization, *Future Gener. Comp. Sy.*, **111** (2020), 300–323. https://doi.org/10.1016/j.future.2020.03.055

30. S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp Warm Algorithm: A bio-inspired optimizer for engineering design problems, *Adv. Eng. Software*, **114** (2017), 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
31. R. Abbassi, A. Abbassi, A. A. Heidari, S. Mirjalili, An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models, *Energ. Convers. Manage.*, **179** (2019), 362–372. https://doi.org/10.1016/j.enconman.2018.10.069
32. M. Tolba, H. Rezk, A. A. Z. Diab, M. Al-Dhaifallah, A novel robust methodology based salp swarm algorithm for allocation and capacity of renewable distributed generators on distribution grids, *Energies*, **11** (2018), 2556. https://doi.org/10.3390/en11102556
33. R. A. Ibrahim, A. A. Ewees, D. Oliva, M. Abd Elaziz, S. Lu, Improved salp swarm algorithm based on particle swarm optimization for feature selection, *J. Amb. Intell. Hum. Comp.*, **10** (2019), 3155–3169. https://doi.org/10.1007/s12652-018-1031-9
34. N. Neggaz, A. A. Ewees, M. Abd Elaziz, M. Mafarja, Boosting salp swarm algorithm by sine cosine algorithm and disrupt operator for feature selection, *Expert Syst. Appl.*, **145** (2020), 113103. https://doi.org/10.1016/j.eswa.2019.113103
35. G. I. Sayed, G. Khoriba, M. H. Haggag, A novel chaotic salp swarm algorithm for global optimization and feature selection, *Appl. Intell.*, **48** (2018), 3462–3481. https://doi.org/10.1007/s10489-018-1158-6
36. Q. Zhang, H. Chen, A. A. Heidari, X. Zhao, Y. Xu, P. Wang, et al., Chaos-induced and mutation-driven schemes boosting salp chains-inspired optimizers, *IEEE Access*, **7** (2019), 31243–31261. https://doi.org/10.1109/ACCESS.2019.2902306
37. H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A. Z. Ala'M, S. Mirjalili, et al., An efficient binary salp swarm algorithm with crossover scheme for feature selection problems, *Know.-Based Syst.*, **154** (2018), 43–67. https://doi.org/10.1016/j.knosys.2018.05.009
38. B. Yang, L. Zhong, X. Zhang, H. Shu, T. Yu, H. Li, et al., Novel bio-inspired memetic salp swarm algorithm and application to MPPT for PV systems considering partial shading condition, *J. Clean. Prod.*, **215** (2019), 1203–1222. https://doi.org/10.1016/j.jclepro.2019.01.150
39. A. A. El-Fergany, Extracting optimal parameters of PEM fuel cells using salp swarm optimizer, *Renew. Energ.*, **119** (2018), 641–648. https://doi.org/10.1016/j.renene.2017.12.051
40. Q. Fan, Z. Chen, Z. Li, Z. Xia, Y. Lin, An efficient refracted salp swarm algorithm and its application in structural parameter identification, *Eng. Comput.*, **38** (2021), 175–189. https://doi.org/10.1007/s00366-020-01034-7
41. A. E. Hegazy, M. Makhlouf, G. S. El-Tawel, Improved salp swarm algorithm for feature selection, *J. King Saud Uni. Comput. Inform. Sci.*, **32** (2020), 335–344. https://doi.org/10.1016/j.jksuci.2018.06.003
42. W. Chao, R. Xu, L. Ma, J. Zhao, L. Wang, An efficient salp swarm algorithm based on scale-free informed followers with self-adaption weight, *Appl. Intell.*, 2022, 1–33. https://doi.org/10.1007/s10489-022-03438-y
43. M. Tawhid, A. Ibrahim, Improved salp swarm algorithm combined with chaos, *Math. Comput. Simulat.*, **202** (2022), 113–148. https://doi.org/10.1016/j.matcom.2022.05.029
44. C. Lin, P. Wang, X. Zhao, H. Chen, Double mutational salp swarm algorithm: from optimal performance design to analysis, *J. Bionic Eng.*, 2022, 1–28. https://doi.org/10.1007/s42235-022-00262-5

45. N. Singh, L. Hoang, F. Chiclana, J. Magnot, A new fusion of salp swarm with sine cosine for optimization of non-linear functions, *Eng. Comput.*, **36** (2020), 185–212. https://doi.org/10.1007/s00366-018-00696-8
46. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, GSA: a gravitational search algorithm, *Inform. Sci.*, **179** (2009), 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004
47. V. Andersen, P. Nival, A model of the population dynamics of salps in coastal waters of the Ligurian Sea, *J. Plankton Res.*, **8** (1986), 1091–1110. https://doi.org/10.1093/plankt/8.6.1091
48. D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *J. Global Optim.*, **39** (2007), 459–471. https://doi.org/10.1007/s10898-007-9149-x
49. A. R. Mehrabian, C. Lucas, A novel numerical optimization algorithm inspired from weed colonization, *Ecol. Inform.*, **1** (2006), 355–366. https://doi.org/10.1016/j.ecoinf.2006.07.003
50. J. Barraza, L. Rodríguez, O. Castillo, P. Melin, F. Valdez, A new hybridization approach between the fireworks algorithm and grey wolf optimizer algorithm, *J. Optim.*, **2018** (2018), 6495362. https://doi.org/10.1155/2018/6495362
51. N. Singh, F. Chiclana, J. P. Magnot, A new fusion of salp swarm with sine cosine for optimization of non-linear functions, *Eng. Comput.*, **36** (2020), 185–212. https://doi.org/10.1007/s00366-018-00696-8
52. J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, *Swarm Evol. Comput.*, **1** (2011), 3–18. https://doi.org/10.1016/j.swevo.2011.02.002
53. A. Kaveh, V. R. Mahdavi, Colliding bodies optimization: a novel meta-heuristic method, *Comput. Struct.*, **139** (2014), 18–27. https://doi.org/10.1016/j.compstruc.2014.04.005
54. A. Kaveh, T. Bakhshpoori, E. Afshari, An efficient hybrid particle swarm and swallow swarm optimization algorithm, *Comput. Struct.*, **143** (2014), 40–59. https://doi.org/10.1016/j.compstruc.2014.07.012
55. W. Long, X. Liang, Y. Huang, Y. Chen, An effective hybrid cuckoo search algorithm for constrained global optimization, *Neural Comput. Appl.*, **25** (2014), 911–926. https://doi.org/10.1007/s00521-014-1577-1

AIMS Press