

Article

Performance of a Novel Enhanced Sparrow Search Algorithm for Engineering Design Process: Coverage Optimization in Wireless Sensor Network

Rui Liu ¹  and Yuanbin Mo ^{2,3,*}¹ College of Electronic Information, Guangxi Minzu University, Nanning 530006, China² Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, Guangxi Minzu University, Nanning 530006, China³ School of Artificial Intelligence, Guangxi Minzu University, Nanning 530006, China

* Correspondence: moyuanbin2020@gxmzu.edu.cn

Abstract: Burgeoning swarm intelligence techniques have been creating a feasible theoretical computational method for the modeling, simulation, and optimization of complex systems. This study aims to increase the coverage of a wireless sensor network (WSN) and puts forward an enhanced version of the sparrow search algorithm (SSA) as a processing tool to achieve this optimization. The enhancement of the algorithm covers three aspects. Firstly, the Latin hypercube sampling technique is utilized to generate the initial population to obtain a more uniform distribution in the search space. Secondly, a sine cosine algorithm with adaptive adjustment and the Lévy flight strategy are introduced as new optimization equations to enhance the convergence efficiency of the algorithm. Finally, to optimize the individuals with poor fitness in the population, a novel mutation disturbance mechanism is introduced at the end of each iteration. Through numerical tests of 13 benchmark functions, the experimental results show that the proposed enhanced algorithm can converge to the optimum faster and has a more stable average value, reflecting its advantages in convergence speed, robustness, and anti-local extremum ability. For the WSN coverage problem, this paper established a current optimization framework based on the swarm intelligence algorithms, and further investigated the performance of nine algorithms applied to the process. The simulation results indicate that the proposed method achieves the highest coverage rate of 97.66% (on average) among the nine algorithms in the calculation cases, which is increased by 13.00% compared with the original sparrow search algorithm and outperforms other methods by 1.47% to 15.34%.

Keywords: swarm intelligence; wireless sensor network; modeling; simulation; coverage optimization; sparrow search algorithm



Citation: Liu, R.; Mo, Y. Performance of a Novel Enhanced Sparrow Search Algorithm for Engineering Design Process: Coverage Optimization in Wireless Sensor Network. *Processes* **2022**, *10*, 1691. <https://doi.org/10.3390/pr10091691>

Academic Editors: Amir H. Gandomi and Laith Abualigah

Received: 26 July 2022

Accepted: 22 August 2022

Published: 25 August 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A wireless sensor network (WSN) is a practical intelligent information acquisition system composed of a certain number of sensor nodes with communication, computing, sensing, and other functions deployed in the area to be monitored. Owing to its advantages of economic applicability and fault tolerance, WSNs have been increasingly prevalent in agricultural management [1], intelligent medical services [2,3], advanced communication [4], and Internet of Things (IoT) [5,6]. By and large, the design of a WSN confronts many constraints and challenges, and the coverage optimization for monitoring areas is a fundamental issue that any type of WSNs need to solve [7]. In some scenarios, the application of a WSN is restricted by the actual physical conditions. Random deployment is the simplest method, but it often leads to uneven distribution, high overlap, and low coverage, which remarkably affects the quality of service for a WSN. Consequently, an intelligent nodes deployment method is necessary. As one direct means of implementation,

the deployment optimization of sensor nodes is of great significance to improve the sensing performance and work efficiency of a WSN.

In recent years, metaheuristics have provided flexible solutions to various optimization problems. As an efficient branch, swarm intelligence has the characteristics of coevolution and more robust population optimization [8,9], which has attracted the attention of many scholars and has made some progress in the application study of WSN coverage optimization. Liao W et al. (2011) [10] and Yoon Y et al. (2013) [11], respectively, proposed sensor nodes deployment schemes based on the ant colony optimization algorithm and genetic algorithm for the problem, which increased the coverage area of a WSN and extended its service life. Wang L et al. (2018) [12] used the reverse learning strategy to improve the whale optimization algorithm and applied it to a probability-aware WSN model to enhance its coverage. Wang X et al. (2018) [13] improved particle swarm optimization by using the resampling method based on particle swarm optimization to ensure the activity of the population to a certain extent and applied the optimized algorithm to the coverage control problem. Miao Z et al. (2020) [14] proposed an improved grey wolf optimizer that consolidated the leadership of α gray wolf, and the applicability of the algorithm has been confirmed in the solution of WSN coverage optimization. Zhu F et al. (2021) [15] proposed a node simulation optimization method based on the improved weed algorithm before the actual deployment of the network. By introducing the differential evolution strategy and the disturbance mechanism, the improved algorithm has a more efficient convergence speed and a deployment scheme that can guide manual or robot placement of nodes is obtained. Mohammad Shokouhifar (2021) [16] proposed a practical multi-objective optimization algorithm based on the combination of the whale optimization algorithm and simulated annealing. With the objective function of maximizing the network coverage and minimizing the total cost, it has been successfully applied to radio frequency identification (RFID) network planning in the hospital environment, greatly reducing the total cost of the RFID network and achieving the goal of tracking medical assets. He Q et al. (2022) [17] proposed a marine predator algorithm with dynamic inertia weight and a multi-elite learning mechanism, which has improved the WSN coverage rate compared with the original algorithm. However, the improved algorithm converges slowly in the early iterations, which leads to unstable coverage. In this problem, the coverage optimization algorithms based on swarm intelligence are capable of obtaining the deployment schemes of sensor nodes more conveniently through the black box operation mechanism. In these cases, the performance of the algorithm is a key factor affecting the deployment quality. Hence, it is also crucial to perfect the algorithm, which can considerably make a difference to the results of WSN coverage optimization.

In this context, by correcting the defects of the sparrow search algorithm [18], this paper introduces a WSN sensor nodes deployment optimization method based on the augmentation of SSA, and names the new algorithm as a novel enhanced SSA (NESSA). Firstly, NESSA uses Latin hypercube sampling (LHS) to initialize the population, which ensures that the initial variables can effectively cover the search space and facilitates the rationality of the initial population distribution compared with the original random initialization method. Secondly, from the perspective of improving the performance and applicability to specific optimization problems, the original version is enhanced by using a sine cosine algorithm iteration and Lévy flight strategy, respectively. The results obtained are validated by the experimental data feedback and have achieved good performance in the subsequent WSN coverage optimization. Finally, based on the warning disturbance mechanism of the original algorithm, NESSA introduces a disruption phase for the population with poor fitness. Its principle is to utilize the information of the optimal individual to guide the worse population, so as to improve their optimization quality. This measure enriches the diversity of the population and is conducive to boosting the convergence and efficiency of the algorithm. To sum up, the contributions of our paper are mainly reflected in the following three aspects:

- A novel enhanced SSA version is implemented from the perspective of applicability and utilized to maximize the WSN coverage rate.
- A swarm intelligence applicable optimization process for the WSN coverage enhancement problem is established.
- The performance of other well-known swarm intelligence algorithms in WSN coverage optimization is further investigated and analyzed.

The rest of the paper is organized in the following manner. In Section 2, we first discuss the iterative mechanism of SSA and clarify the motivation for the improvement in this study, then expound in detail on the mathematical model of NESSA based on the augmentation part, and finally use the benchmark functions to conduct a numerical comparison experiment on the enhanced version to verify the effectiveness of the improved strategy. In Section 3, we introduce the mathematical model of WSN coverage optimization and set up a universal optimization process based on the swarm intelligence algorithm. Then, we further investigate the coverage optimization performance of nine swarm intelligence algorithms for three cases under this framework. In Section 4, we summarize some conclusions of this paper and look forward to future research directions.

2. Mathematical Model of Optimization Algorithm

The beginning of this section discusses the standard SSA metaheuristics, describing its known drawbacks. The improvement strategies to overcome these shortcomings are provided in detail and a novel enhanced SSA metaheuristics is proposed. Finally, the performance of the new algorithm is validated by numerical experiments.

2.1. Overview of the Standard SSA Metaheuristics

The standard SSA metaheuristic is a swarm intelligence algorithm proposed by Jiankai Xue et al. (2020) [18]. Its motivation comes from the foraging behavior of the sparrow population and the biological characteristics of avoiding natural enemies. To make the algorithm work in the actual optimization, SSA idealizes the sparrow individuals through several approximation rules and divides the constructed mathematical model into the following four iterative stages.

2.1.1. Initialization of Sparrow Population

SSA initializes the population in the form of random distribution, and the sparrows used for searching can be expressed as $x_i = (x_{1,d}, x_{2,d}, \dots, x_{i,d})$, where d represents the dimension of the optimization problem variable. Generally, when all sparrow individuals have the same boundary, the randomly initialized population is:

$$X = rand(pop, d) \times (ub - lb) + lb \quad (1)$$

where X represents the sparrow population, $rand(pop, d)$ represents a random number matrix of $pop \times d$, and ub and lb represent the upper and lower boundaries of sparrows, respectively.

2.1.2. The Producer Update Phase

SSA stipulates that producers are sparrows with the highest fitness in the population, are the learning objects of other individuals, and play a role in providing an evolutionary direction for the whole population. The updated equation for the producer is as follows:

$$x_i^{t+1} = \begin{cases} x_i^t \cdot \exp(-\frac{i}{\alpha \cdot Iter_{max}}), & R_2 < ST \\ x_i^t + Q \cdot L, & R_2 \geq ST \end{cases} \quad (2)$$

where t is the current iteration, $Iter_{max}$ is the maximum iteration, α is the random number on $(0,1]$, Q is the random number subject to a Normal Distribution, and L is the ones matrix of $1 \times d$. $R_2 \in [0,1]$ and $ST \in [0.5,1]$ represent the alarm value and the safety threshold, respectively, which are used as the adjustment parameters of the producer. In this paper, R_2

is set as the random value, while ST is defined as a fixed value of 0.8, which is consistent with the reference [18].

2.1.3. The Scrounger Update Phase

SSA defines the part whose fitness ranks behind all producers as the scrounger. These sparrows follow producers to improve their locations in the search space and are a supplement to producers. Their identities are in dynamic balance with producers in each iteration. The updated equation for the scrounger is as follows:

$$x_i^{t+1} = \begin{cases} Q \cdot \exp\left(\frac{x_{worst}^t - x_i^t}{i^2}\right), & i > pop/2 \\ x_p^t + |x_i^t - x_p^t| \cdot A^+ \cdot L, & else \end{cases} \quad (3)$$

where x_p is the best location found by the producer, x_{worst}^t is the current worst location, A is a $1 \times d$ matrix, its internal elements are 1 or -1 , and $A^+ = A^T(AA^T)^{-1}$, Q and L are the same as Equation (2).

2.1.4. The Scouter Update Phase

After the iteration of producers and scroungers, SSA randomly selects a certain proportion of sparrows in the population as scouters and updates their locations in each iteration. The updated equation for the scouter is as follows:

$$x_i^{t+1} = \begin{cases} x_{best}^t + \beta \cdot |x_i^t - x_{best}^t|, & f_i > f_{best} \\ x_i^t + K \cdot \left(\frac{|x_i^t - x_{worst}^t|}{(f_i - f_{worst}) + \varepsilon}\right), & f_i = f_{best} \end{cases} \quad (4)$$

where x_{best}^t is the current global optimal location, β and K are the control parameters, f_i is the individual fitness, and ε is a minimal constant to avoid the denominator being 0, which is set as 1×10^{-8} , f_{best} and f_{worst} , respectively, represent the current global optimal fitness and the worst fitness.

2.2. Motivation for Improvements

Although previous studies have indicated that SSA has made significant progress in some practical challenges in the engineering field [19,20], these successful applications are often established based on the improved algorithms [21–24]. Solving the defects faced by standard SSA and adopting more appropriate improvement strategies to supplement and replace the algorithm framework are crucial prerequisites for attaining better performance indicators in engineering problems. For the iterative framework of standard SSA, we have the following conclusions to expound.

For the initialization phase of SSA, mapping variables in the search space using the random method tends to cause a lack of population diversity and a reduction in the ability to resist local extremum, which harms the convergence speed and accuracy of the algorithm. Thus, improving the quality of the initial population is meaningful work. Moreover, SSA also faces the disadvantage that the convergence becomes slower in the later iteration. Through the analysis of Equation (2), it can be seen that the update of producers is only affected by the individuals of the previous generation and lacks the guidance of the global optimal solution. When $R_2 < ST$, the value range of the population is gradually reduced by the influence of $\exp\left(-\frac{i}{\alpha \cdot Iter_{max}}\right)$ [25], forming a typical “narrow search mode” tending to zero. Although this strategy facilitates the development ability of the algorithm and enables it to deeply mine the current region, in solving some problems, it is more necessary to ensure the breadth of the search scope and to find a promising area in the search space. Consequently, the current strategy at this phase is not always applicable. In addition, enhancing the exploration ability of scroungers to boost the chances of turning them into

producers can reduce the possibility of the algorithm falling into stagnation, and effectively disturb the individuals with lower fitness ranking, which is helpful with the algorithm to search for the global optimal solution of the problem more efficiently.

It is worth noting that, as the “no free lunch” theorem reveals [26], no learning algorithm can provide the best solution to all problems. Hence, based on extensive empirical simulation, this study considers the improvement of SSA from the application side of WSN coverage optimization and enables the designed new algorithm to attain positive feedback results in the simulation test of the benchmark functions.

2.3. Proposed Novel Enhanced SSA Metaheuristics

A novel enhanced SSA version proposed in this study addresses issues of the standard SSA by assimilating the following strategies:

- Uniform population initialization based on Latin hypercube sampling;
- The sine and cosine iteration equations for the producer update phase;
- The scrounger update phase with Lévy flight;
- The disruption phase acts on the worse population with poor fitness.

2.3.1. Latin Hypercube Sampling Initialization

Latin hypercube sampling (LHS) is a multidimensional stratified sampling technique proposed by McKay et al. (1979) [27], which has the advantage of obtaining tail sample values under fewer sampling conditions.

Assuming that the sample size is 30, the distribution of LHS and random method in $[0, 1]$ under two-dimensional conditions is illustrated in Figure 1. It can be seen that the sample distribution generated by LHS is wider than the area covered by the random one, and it meets the randomness of sampling [28]. Thus, LHS can form more uniform points in the search space, improve the diversity of the population, and lay a solid foundation for the optimization of the algorithm.

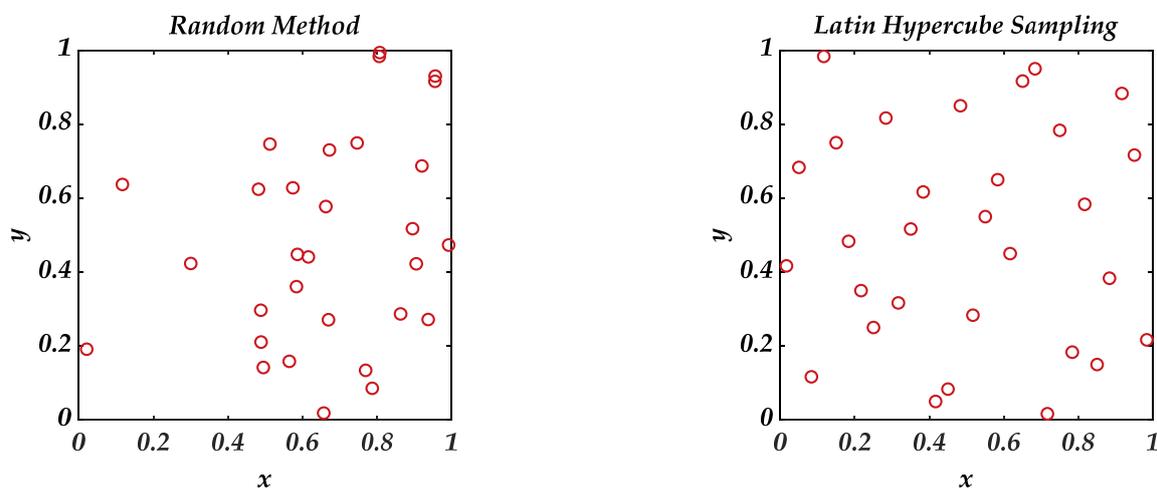


Figure 1. Comparison of two initialization methods.

The steps of initializing the sparrow population using the LHS strategy with uniform stratification and equal probability sampling are as follows:

- (1) The population pop , dimension d , and the boundary of individual sparrows $[lb, ub]$ are set.
- (2) The $[lb, ub]$ of each individual is divided into three sub-intervals that do not overlap each other and have the same probability.
- (3) One point is randomly selected from each sub-interval.
- (4) The points extracted from each dimension are combined to form an initial population.

2.3.2. Sine and Cosine Iteration Equations

The hybrid strategy can integrate the advantages of different metaheuristics and often bring more reliable and efficient optimization solutions to the problem [29]. Similar to SSA, the sine cosine algorithm (SCA) is also a metaheuristic method based on population optimization. It was proposed by Mirjalili (2016) [30] with the characteristics of simplicity and effectiveness, which is composed of only one set of iterative equations, as shown below:

$$x_i^{t+1} = \begin{cases} x_i^t + r_1 \cdot \sin(r_2) \cdot |r_3 \cdot x_{best} - x_i^t|, & r_4 < 0.5 \\ x_i^t + r_1 \cdot \cos(r_2) \cdot |r_3 \cdot x_{best} - x_i^t|, & r_4 \geq 0.5 \end{cases} \tag{5}$$

where r_2, r_3 and r_4 are random numbers, and $r_2 \in (0, 2\pi), r_3 \in (0, 2),$ and $r_4 \in (0, 1), r_1$ is the control factor as follows:

$$r_1 = a - t \cdot \frac{a}{Iter_{max}} \tag{6}$$

where a is a constant. The function of r_1 is to adjust the exploration and development process of the algorithm, and its essence is an inertia weight that decreases linearly with iteration. In this paper, we modify Equation (5), apply r_1 to the current iteration location x_i^t to further play its role in the adaptive adjustment of the algorithm, and introduce the modified equation into the SSA optimization framework as a new updated producer strategy:

$$x_i^{t+1} = \begin{cases} r_1 \cdot x_i^t + r_1 \cdot \sin(r_2) \cdot |r_3 \cdot x_{best} - x_i^t|, & R_2 < ST \\ r_1 \cdot x_i^t + r_1 \cdot \cos(r_2) \cdot |r_3 \cdot x_{best} - x_i^t|, & R_2 \geq ST \end{cases} \tag{7}$$

Furthermore, the simulation results show that the value of constant a can affect the convergence of the hybrid algorithm. By solving the sphere function, Figure 2 shows the iterative trajectory of six hybrid algorithms with different a values when $d = 100, pop = 30,$ and $Iter_{max} = 500.$

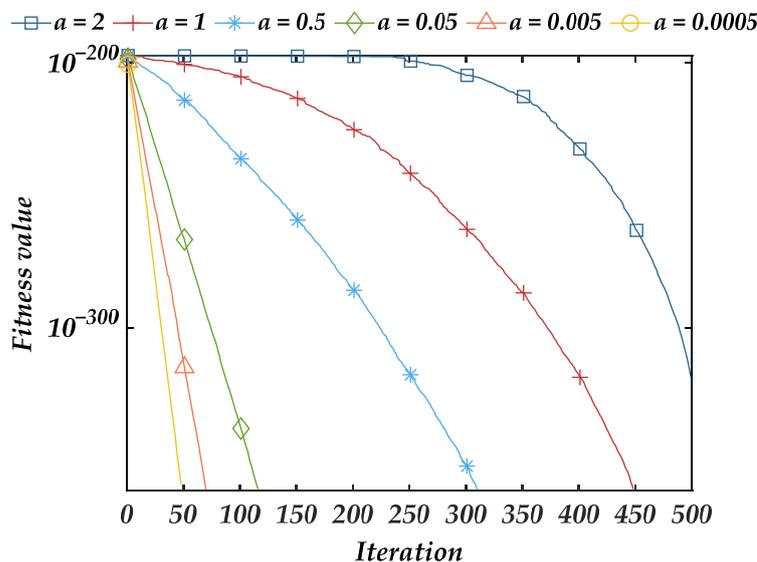


Figure 2. Iterative trajectory of hybrid algorithm with different parameter values.

Six algorithms with different a values can successfully find the optimal value 0 of the sphere function. It can be seen from Figure 2 that the convergence speed of the algorithm gradually increases with the decrease in the parameter value. When $a = 0.0005,$ the hybrid algorithm can converge in less than 50 iterations. When $a < 0.0005,$ the reduction in iteration is no longer significant. Hence, this paper sets $a = 0.0005$ to ensure that the enhanced version has a faster convergence speed.

2.3.3. Lévy Flight Strategy

As a supplement to the producer in the population, scroungers usually concentrate on a narrow region under the influence of the local optimal, which makes the algorithm premature. It will also slow down the evolution of scroungers into producers and reduce the diversity of the search process.

Lévy flight is one special random walk, which can provide a certain probability of large-scale disturbance for the algorithm [31]. When scroungers fall into the local optimal, they can step into a wider area for optimization according to the generated random step size, which improves the exploration ability of the algorithm. Thus, we designed the Lévy flight strategy for the scrounger update phase. The new scrounger update strategy is as follows:

$$x_i^{t+1} = x_p^t + x_p^t \cdot Lévy(d) \quad (8)$$

where d is the dimension, and the Lévy function is calculated as follows:

$$Lévy(x) = 0.01 \times \frac{\lambda_1 \cdot \sigma}{|\lambda_2|^{\frac{1}{\beta}}} \quad (9)$$

where λ_1 and λ_2 are random numbers on $[0,1]$, β is a constant, which is taken as 1.5 in this paper, and σ is calculated as follows:

$$\sigma = \left[\frac{\Gamma(1 + \beta) \cdot \sin(\pi \cdot \beta / 2)}{\Gamma((1 + \beta) / 2) \cdot \beta \cdot 2^{(\beta-1)/2}} \right]^{\frac{1}{\beta}} \quad (10)$$

where $\Gamma(x) = (x - 1)!$, x belongs to the set of natural numbers.

2.3.4. Disruption Phase

The simulation of the disruption phenomenon is a disturbance operation inspired by astrophysics and proposed by Sarafrazi et al. (2011) [32]. It is realized by adding a disruption factor to the gravitational search algorithm (GSA) [33]. The purpose is to prevent premature convergence and activate the exploration ability of the algorithm. In this study, we introduce the disruption factor to effectively disturb the part with poor fitness in the SSA population, define X_w ($w = k + 1, k + 2, \dots, pop$) as the worse population in the algorithm, and calculate the equation of boundary k is as follows:

$$k = \left[\frac{3 \cdot pop}{4} + pop \cdot \left(0.5 - \frac{t}{Iter_{max}} \right)^3 \right] \quad (11)$$

The disruption phase is designed to disturb X_w to improve the flexibility of the algorithm to solve complex application problems. The principle of this phase is to define the Euclidean distances between the target individual i and its adjacent individual j and the current optimal individual as $R_{i,j}$ and $R_{i,best}$ respectively, and conduct the disturbance to e when the following conditions are met:

$$\frac{R_{i,j}}{R_{i,best}} < C \quad (12)$$

where C is the threshold value to prevent the complexity of the algorithm from increasing too much. When the algorithm does not converge, a larger C enables individuals to explore more space, and with the iteration, its value should become smaller to speed up the convergence of the algorithm. Therefore, C is designed as a variable, and the calculation equation is as follows:

$$C = \theta \cdot \left(1 - \frac{t}{Iter_{max}} \right) \quad (13)$$

where θ is the initial threshold. Reference [28] indicates that $\theta = 100$ is the most appropriate value. When the conditions of Equation (12) are satisfied, disturbance to the target individual is conducted through the following equation:

$$x_{i,w}^{t+1} = \frac{t}{Iter_{max}} \cdot x_{i,w}^t + \left(1 - \frac{t}{Iter_{max}}\right) \cdot x_{i,w}^t \cdot D \quad (14)$$

where D is the disruption factor, and its calculation equation is as follows:

$$D = \begin{cases} U\left(-\frac{R_{i,j}}{2}, \frac{R_{i,j}}{2}\right), R_{i,best} \geq 1 \\ R_{i,j} + U\left(-\frac{R_{i,j}}{2}, \frac{R_{i,j}}{2}\right), else \end{cases} \quad (15)$$

where U is a random number subject to the Uniform Distribution from $-\frac{R_{i,j}}{2}$ to $\frac{R_{i,j}}{2}$. When $R_{i,best} > 1$, means i is far from the optimal individual, and D is utilized to guide it for exploration. On the contrary, when $R_{i,best} \leq 1$, which means i is near the optimal individual, D will conduct it to develop. Hence, Equation (14) contains two parts: the first part $\frac{t}{Iter_{max}} \cdot x_{i,w}^t$ is the original population information, and the other part $\left(1 - \frac{t}{Iter_{max}}\right) \cdot x_{i,w}^t \cdot D$ includes the disturbance process to the algorithm. Because $\left(1 - \frac{t}{Iter_{max}}\right)$ is decreasing, i satisfying the condition will be disturbed more in the early iteration, so it can explore a wider area and reduce the probability of falling into the local extremum. In the later iteration, the convergence mainly depends on the current population information, which ensures the algorithm can achieve rapid convergence.

2.3.5. Operation Process of NESSA

In this study, we named the algorithm introduced in Sections 2.3.1–2.3.4 as the novel enhanced sparrow search algorithm (NESSA), and its flowchart is shown in Figure 3.

2.4. Benchmark Function Numerical Tests

The performance evaluation of the enhanced version depends on the resulting feedback of the benchmark functions. Taking SSA, SCA, particle swarm optimization (PSO) [34], the whale optimization algorithm (WOA) [35], and the gray wolf optimizer (GWO) [36] as the comparisons, we tested the performance of NESSA on 13 sets of benchmark functions of different types. The operating system used is windows 10 (64 bit), the environment is Intel (R) core (TM) i7-6700h CPU @ 2.60 GHz 2.60 GHz, and all programs are run by MATLAB R2018b, which is developed by MathWorks company in the United States. The benchmark functions are shown in Table 1.

2.4.1. Parameter Settings

Based on the fairness of the experimental results, the six algorithms are tested when $pop = 30$ and $Iter_{max} = 500$. The parameters to be set are as follows: PSO: learning factor $c_1 = c_2 = 1.49445$, weight $\omega = 0.729$. WOA: the logarithmic spiral shape parameter $b = 1$. GWO: convergence factor $\alpha = 2 \times \left(1 - \frac{t}{Iter_{max}}\right)$. SCA: control parameter $a = 2$. SSA: safety threshold $ST = 0.8$, the proportion of producers $PD = 0.2$, the proportion of scouts $SD = 0.1$. NESSA: safety threshold $ST = 0.8$, the proportion of producers $PD = 0.2$, the proportion of scouts, control parameter $a = 0.0005$, the initial threshold of disruption phase $\theta = 100$. It should be noted that due to the randomness of metaheuristics, the above parameters are set based on artificial experimental experience [37], and a suitable parameter selection scheme is conducive to a more effective algorithm.

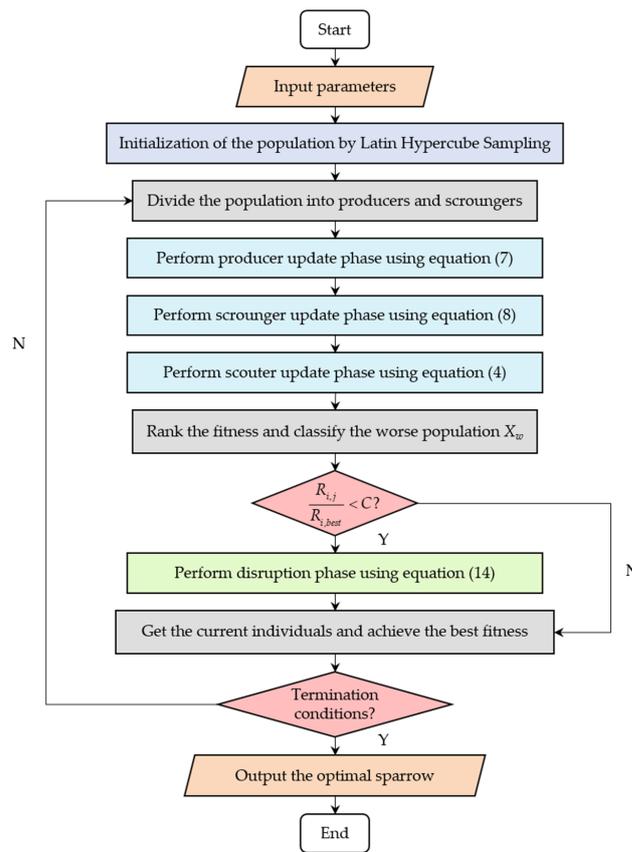


Figure 3. The flowchart of NESSA.

Table 1. Benchmark functions.

Benchmark	Equation	d	Range	F _{min}
Bent Cigar Function	$F_1(x) = x_1^2 + 10^6 \sum_{i=2}^d x_i^2$	100	[−100, 100]	0
Sum of Different Powers Function	$F_2(x) = \sum_{i=1}^d x_i ^{i+1}$	100	[−100, 100]	0
Rotated Hyper-Ellipsoid Function	$F_3(x) = \sum_{i=1}^d \sum_{j=1}^i x_j^2$	100	[−65, 65]	0
Zakharov Function	$F_4(x) = \sum_{i=1}^d x_i^2 + \left(\sum_{i=1}^d 0.5ix_i^2 \right)^2 + \left(\sum_{i=1}^d 0.5ix_i^2 \right)^4$	100	[−5, 10]	0
Sum Squares Function	$F_5(x) = \sum_{i=1}^d ix_i^2$	100	[−10, 10]	0
Quartic Function	$F_6(x) = \sum_{i=1}^d ix_i^4 + random(0, 1)$	100	[−1.28, 1.28]	0
Sphere Model	$F_7(x) = \sum_{i=1}^d x_i^2$	100	[−100, 100]	0
Schwefel’s problem 2.22	$F_8(x) = \sum_{i=1}^d x_i + \prod_{i=1}^d x_i $	100	[−10, 10]	0
Schwefel’s problem 1.2	$F_9(x) = \sum_{i=1}^d \left(\sum_{j=1}^i x_j \right)^2$	100	[−100, 100]	0
Schwefel’s problem 2.21	$F_{10}(x) = \max_i \{ x_i , 1 \leq i \leq d \}$	100	[−100, 100]	0
Rastrigin’s Function	$F_{11}(x) = \sum_{i=1}^d [x_i^2 - 10 \cos(2\pi x_i) + 10]$	100	[−5.12, 5.12]	0
Ackley’s Function	$F_{12}(x) = -20 \exp \left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^d x_i^2} \right) - \exp \left(\frac{1}{D} \sum_{i=1}^d \cos(2\pi x_i) \right) + 20 + e$	100	[−32, 32]	0
Kowalik’s Function	$F_{13}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_i (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	[−5, 5]	0.000307

2.4.2. Analysis of Test Results

Based on the objectivity of the data, each test runs independently 50 times, takes the average value, standard deviation, and calculation time (s) as the final evaluation criteria, and ranks the six algorithms based on the mean results, which are recorded in Table 2. Moreover, the boxplot is drawn to illustrate the characteristic information of each algorithm solving the benchmark functions, and the Wilcoxon test is utilized to indicate the statistical differences between the six algorithms, as shown in Table 3.

The types of the above benchmarks include unimodal and multimodal. When the dimension is 100, it can be seen from the test data in Table 2 that the adaptability of PSO and SCA is weak; only F_{13} achieves results close to the optimal value, while there is a large gap in other tests. The average accuracy of WOA ranks 1st one time and 2nd six times. On the whole, WOA performs better than GWO and SSA. However, when solving F_4 , F_9 , and F_{10} , it performs poorly, and GWO also has the same shortcomings. Although SSA is not as accurate as WOA in several benchmarks, it shows strong stability, and there is no large deviation from the optimal value in all tests.

On this basis, NESSA proposed in this paper inherits the relatively stable advantages of SSA, further promotes the convergence accuracy, obtains the best average accuracy in all tests, and has significant advantages in robustness compared with other algorithms, which proves the effectiveness and feasibility of the improved strategy in performance. Furthermore, the calculation time of the proposed algorithm is slightly shorter than that of the standard SSA. This is because NESSA utilizes the shorter sine and cosine iteration equation and the Lévy flight strategy to replace the original iteration, which improves the operation efficiency of the algorithm. Hence, NESSA, with the disruption phase, does not increase the time complexity of the algorithm in general. In addition, PSO has the fastest running speed, and other rankings are WOA > SCA > GWO > NESSA > SSA. However, the time difference between NESSA and the fastest one in any test is not more than 1 s, but the accuracy and robustness are greatly improved.

The Wilcoxon test is utilized to indicate the statistical performance difference between NESSA and other algorithms by comparing the p -value [38]. In this study, the significance level is taken as 0.05. If the p -value is less than 0.05, it indicates that there is a significant difference between NESSA and the algorithm. Otherwise, it means that there is no significant difference between the two algorithms. N/A also means that the performance of the two algorithms is the same, so they cannot be compared. From the data recorded in Table 3, we can see that in most comparisons (62/65), the p -value is less than 0.05, indicating that the optimization results of NESSA are statistically significantly different from other algorithms, further revealing the superiority of the proposed algorithm. Figures 4–16 show the optimization details of the six algorithms on the benchmark function. Through the iterative trajectory and boxplot, we can learn more about the differences between NESSA and other algorithms.

The visual model of the benchmark function reflects the distribution of each feasible solution in the search space, which is illustrated by contour lines and color bars. The iteration trajectory reflects the convergence speed of the search process and can measure the optimization efficiency of the algorithm. The boxplot shows the solution information of 50 independent runs of each algorithm, which reflects the stability of the results.

From the iteration trajectory of fitness, we can see that NESSA can converge to the optimal value in less than 50 iterations from F_1 to F_5 , F_7 , F_9 , F_{11} to F_{12} , with the fastest convergence speed among the six algorithms, which reflects a better optimization efficiency. Compared with other algorithms, NESSA can effectively avoid falling into the local extremum. The boxplot of NESSA illustrates that there is no abnormal value that is significantly different from the optimal value in the optimization of each benchmark function, which further indicates that the proposed algorithm has good robustness.

Table 2. Test results of six algorithms.

Benchmark	Result	PSO	WOA	GWO	SCA	SSA	NESSA
F ₁	Mean	1.8905×10^{10}	4.6133×10^{-64}	1.3138×10^{-6}	1.0975×10^{10}	5.8424×10^{-45}	0
	Std.	5.0627×10^9	2.5180×10^{-63}	1.2085×10^{-6}	8.4216×10^9	4.1233×10^{-44}	0
	Time	0.1162	0.1712	0.3835	0.3062	0.9161	0.7782
F ₂	Mean	8.9275×10^5	1.3847×10^{-67}	4.9847×10^{-11}	4.9191×10^5	6.0385×10^{-41}	0
	Std.	2.6495×10^5	9.3225×10^{-67}	4.5438×10^{-11}	3.6447×10^5	4.2699×10^{-40}	0
	Time	0.0927	0.1735	0.3772	0.2986	0.9762	0.7717
F ₃	Mean	3.7963×10^5	6.6164×10^{-71}	2.5773×10^{-11}	1.4308×10^5	5.9923×10^{-46}	0
	Std.	9.1736×10^4	2.4545×10^{-71}	1.7921×10^{-11}	9.7908×10^4	4.2372×10^{-45}	0
	Time	0.2379	0.2991	0.5325	0.4151	1.1570	1.0807
F ₄	Mean	6.5223×10^9	1.7066×10^3	1.1456×10^2	6.7071×10^2	9.3710×10^{-44}	0
	Std.	4.2494×10^{10}	2.9734×10^2	5.3017×10^1	1.3924×10^2	6.6262×10^{-43}	0
	Time	0.1069	0.1659	0.3756	0.3171	0.8996	0.7399
F ₅	Mean	9.0534×10^3	4.3646×10^{-71}	6.5252×10^{-13}	3.5847×10^3	5.1037×10^{-34}	0
	Std.	2.4400×10^3	3.0630×10^{-70}	5.7752×10^{-13}	23976×10^3	3.6089×10^{-33}	0
	Time	0.0992	0.1761	0.3991	0.2775	0.9566	0.8670
F ₆	Mean	1.3882×10^1	2.9906×10^{-3}	7.2474×10^{-3}	1.4196×10^2	2.9581×10^{-4}	7.2163×10^{-5}
	Std.	6.2807	4.1427×10^{-3}	2.3951×10^{-3}	5.7890×10^1	2.2796×10^{-4}	7.7187×10^{-5}
	Time	0.3111	0.3877	0.5716	0.5165	1.1721	0.8637
F ₇	Mean	1.9774×10^4	1.9564×10^{-71}	1.6134×10^{-12}	1.2025×10^4	3.0173×10^{-49}	0
	Std.	4.6507×10^3	1.1392×10^{-70}	1.3372×10^{-12}	7.9423×10^3	1.5383×10^{-48}	0
	Time	0.0850	0.1792	0.3643	0.2922	0.9325	0.6870
F ₈	Mean	1.3447×10^2	5.9195×10^{-49}	4.2666×10^{-8}	9.0974	9.1117×10^{-29}	0
	Std.	2.7261×10^1	3.1623×10^{-48}	1.6837×10^{-8}	7.4861	5.7121×10^{-28}	0
	Time	0.0921	0.1825	0.4209	0.2790	0.9327	0.7957
F ₉	Mean	1.2283×10^5	1.0690×10^6	6.5699×10^2	2.3771×10^5	8.2407×10^{-43}	0
	Std.	6.4941×10^4	2.3307×10^5	8.9532×10^2	6.0667×10^4	5.6712×10^{-42}	0
	Time	0.6435	0.7011	0.8707	0.8089	1.3878	1.1600
F ₁₀	Mean	3.2619×10^1	7.8270×10^1	8.8309×10^{-1}	8.9710×10^2	1.7378×10^{-29}	0
	Std.	3.7684	2.3100×10^1	8.1956×10^{-1}	2.7997	1.0589×10^{-28}	0
	Time	0.1178	0.1680	0.3698	0.2757	0.8630	0.7585
F ₁₁	Mean	7.5226×10^2	0	1.0379×10^1	2.6752×10^2	0	0
	Std.	4.5534×10^1	0	8.4162	1.3756×10^2	0	0
	Time	0.1505	0.1919	0.3981	0.3133	0.9070	0.6757
F ₁₂	Mean	1.3862×10^1	4.2988×10^{-15}	1.2197×10^{-7}	1.8467×10^2	8.8818×10^{-16}	8.8818×10^{-16}
	Std.	8.0286×10^{-1}	2.3762×10^{-15}	4.1440×10^{-8}	4.6796	0	0
	Time	0.1359	0.1838	0.3957	0.3415	0.9276	0.7167
F ₁₃	Mean	8.5555×10^{-3}	5.9264×10^{-4}	5.4827×10^{-3}	1.0546×10^{-3}	2.3203×10^{-3}	3.2338×10^{-4}
	Std.	1.3163×10^{-2}	3.1154×10^{-4}	1.2621×10^{-2}	3.8164×10^{-4}	6.0756×10^{-3}	1.1952×10^{-5}
	Time	0.0652	0.0783	0.0711	0.0727	0.1702	0.2031

Table 3. Wilcoxon p -value test results.

Benchmark	NESSA vs. PSO	NESSA vs. WOA	NESSA vs. GWO	NESSA vs. SCA	NESSA vs. SSA
F_1	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	4.6715×10^{-19}
F_2	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	1.6907×10^{-18}
F_3	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	4.6715×10^{-19}
F_4	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	1.6907×10^{-18}
F_5	3.3111×10^{-20}				
F_6	7.0661×10^{-18}	1.1738×10^{-15}	7.0661×10^{-18}	7.0661×10^{-18}	3.5360×10^{-9}
F_7	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	1.6907×10^{-18}
F_8	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	1.2593×10^{-19}
F_9	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	4.6715×10^{-19}
F_{10}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	3.3111×10^{-20}	1.2593×10^{-19}
F_{11}	3.3111×10^{-20}	N/A	3.3111×10^{-20}	3.3111×10^{-20}	N/A
F_{12}	3.3111×10^{-20}	1.1011×10^{-14}	3.3111×10^{-20}	3.3111×10^{-20}	N/A
F_{13}	1.5991×10^{-4}	9.0593×10^{-10}	1.4307×10^{-3}	1.1417×10^{-17}	8.7729×10^{-9}

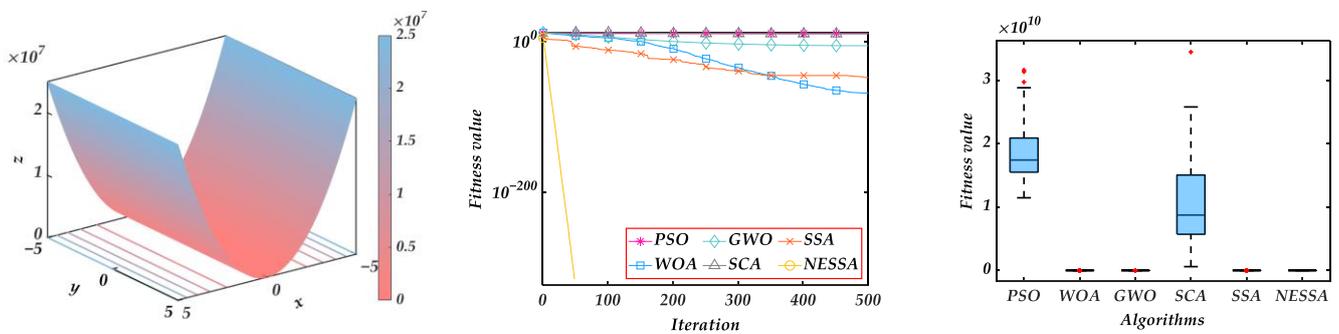


Figure 4. Bent Cigar Function.

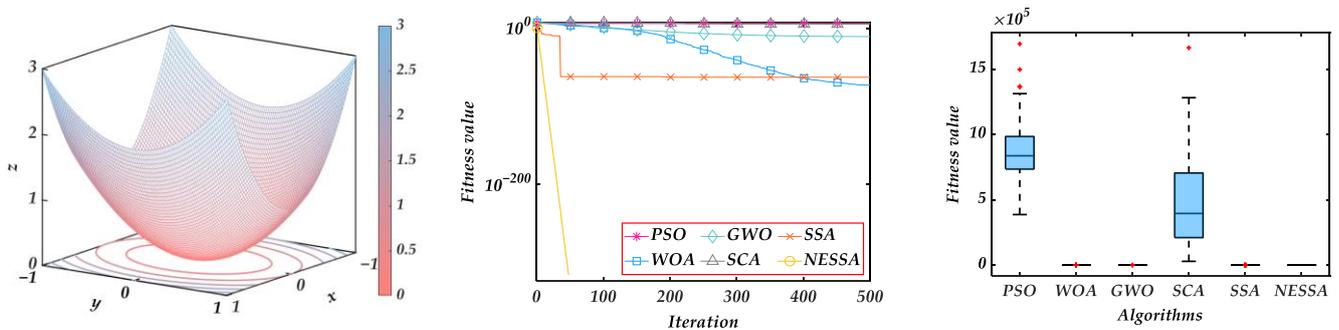


Figure 5. Sum of Different Powers Function.

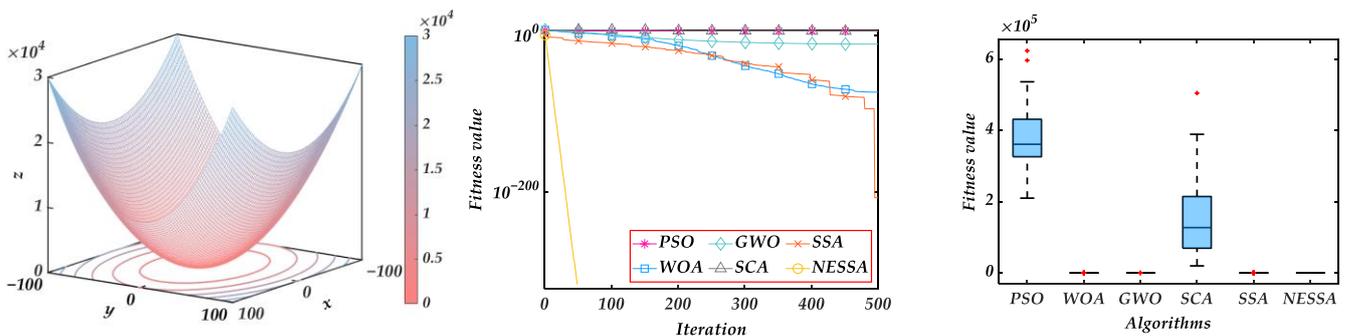


Figure 6. Rotated Hyper-Ellipsoid Function.

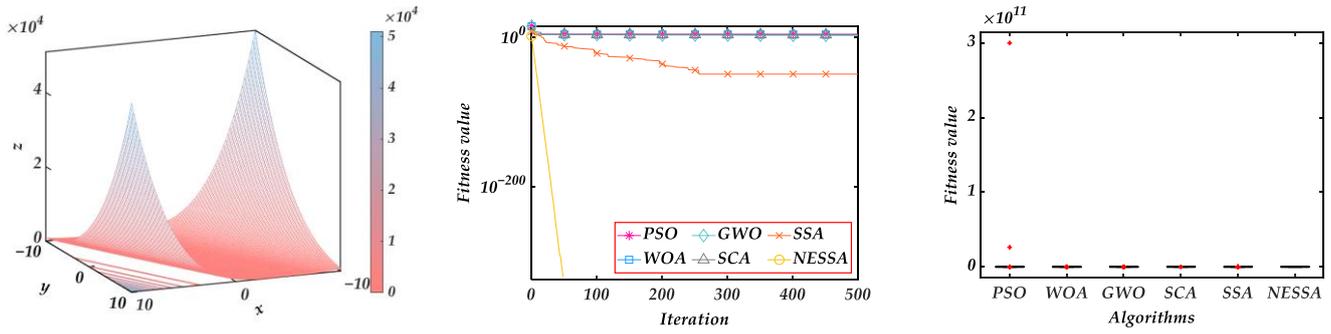


Figure 7. Zakharov Function.

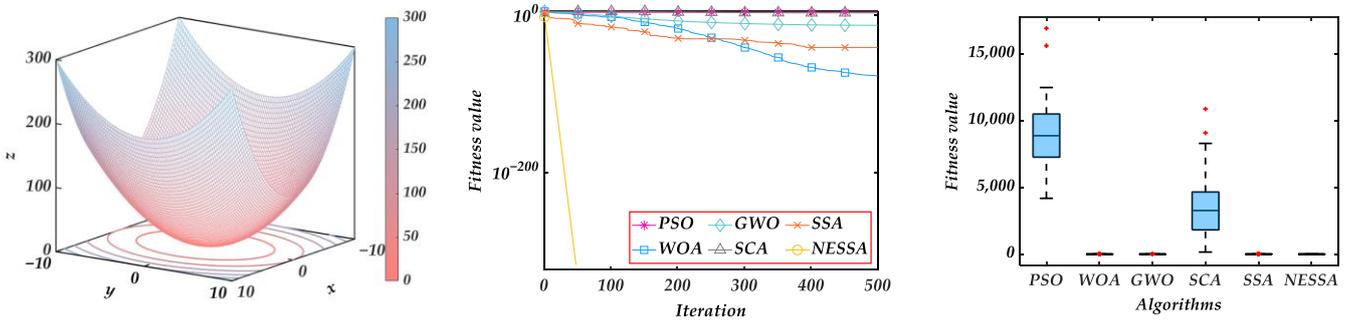


Figure 8. Sum Squares Function.

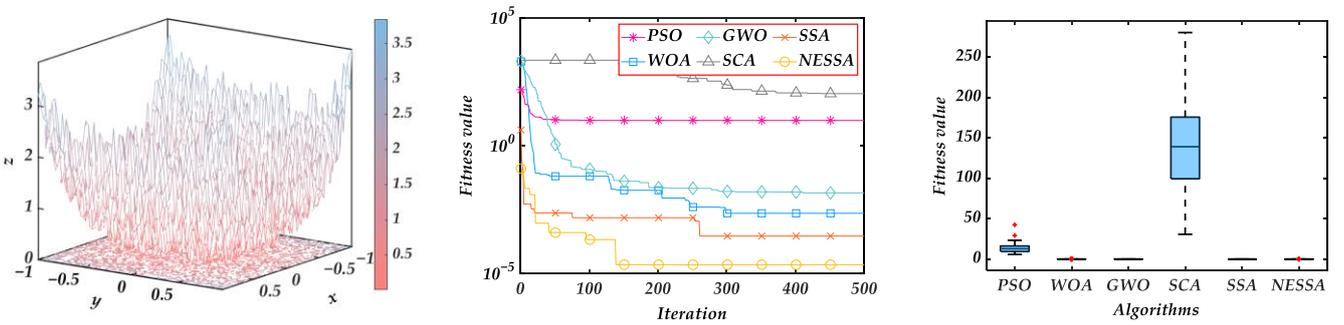


Figure 9. Quartic Function.

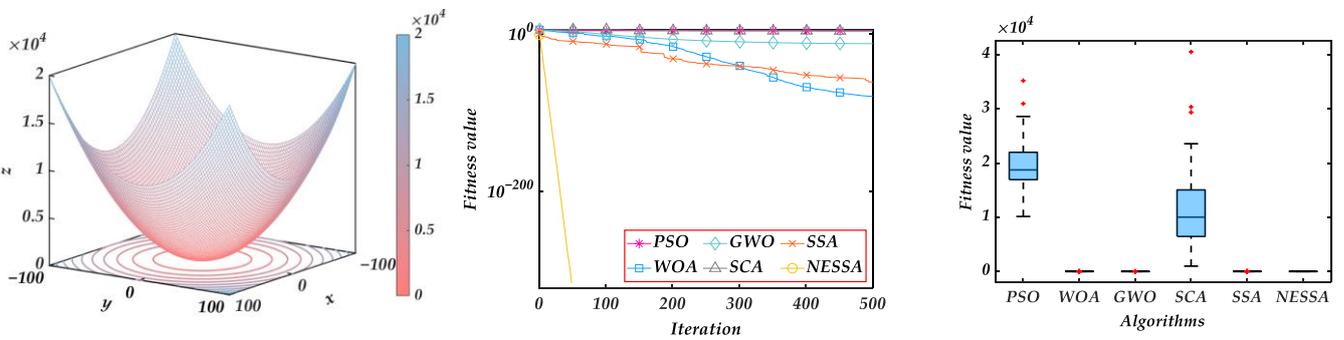


Figure 10. Sphere Model.

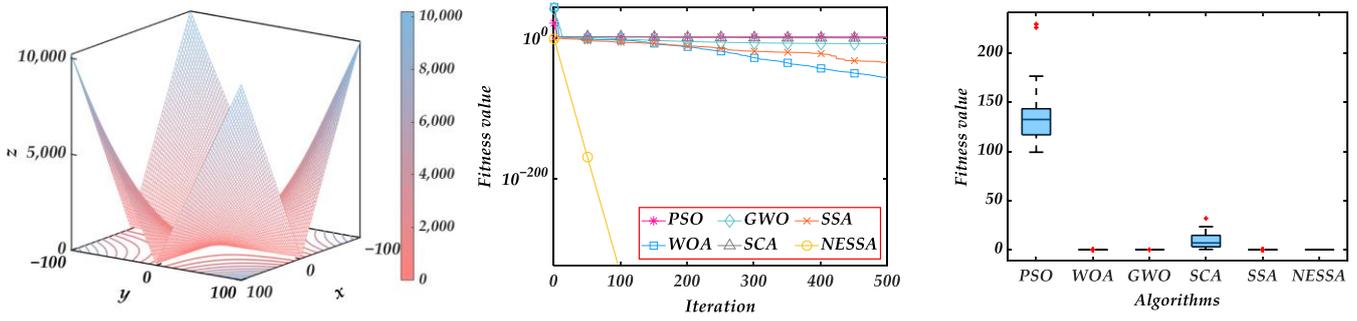


Figure 11. Schwefel's problem 2.22.

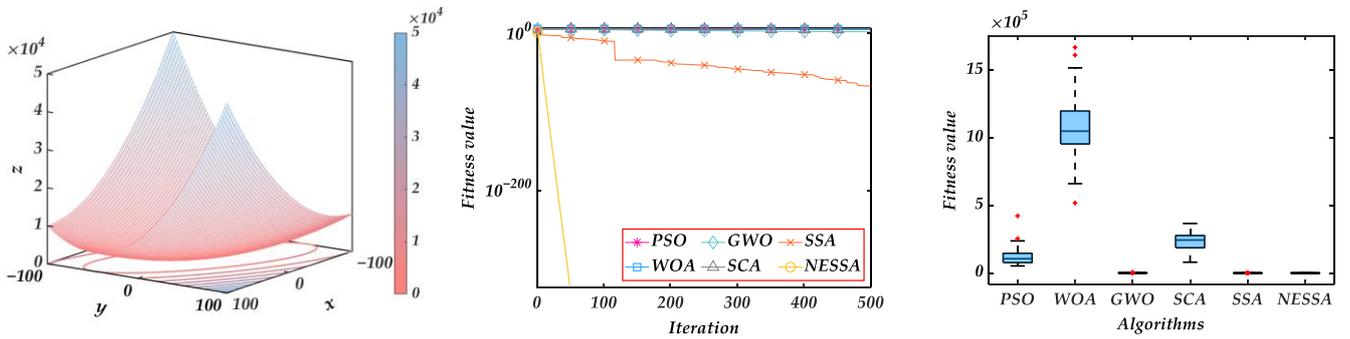


Figure 12. Schwefel's problem 1.2.

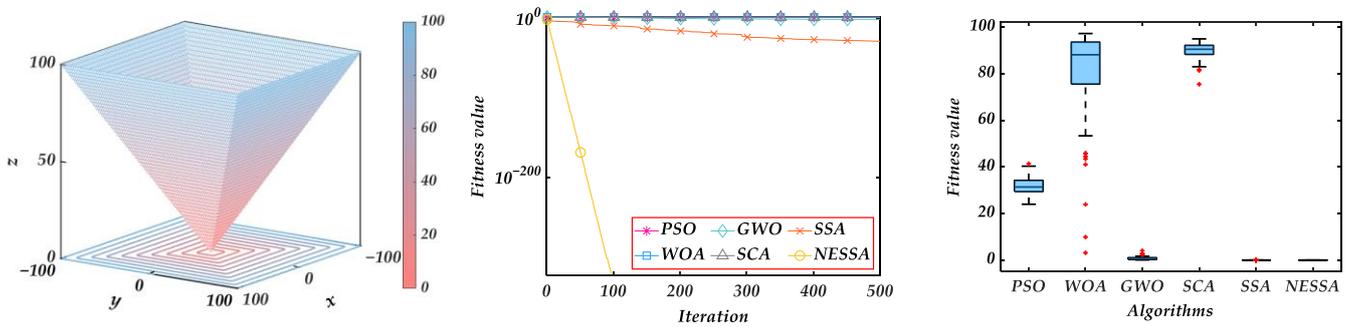


Figure 13. Schwefel's problem 2.21.

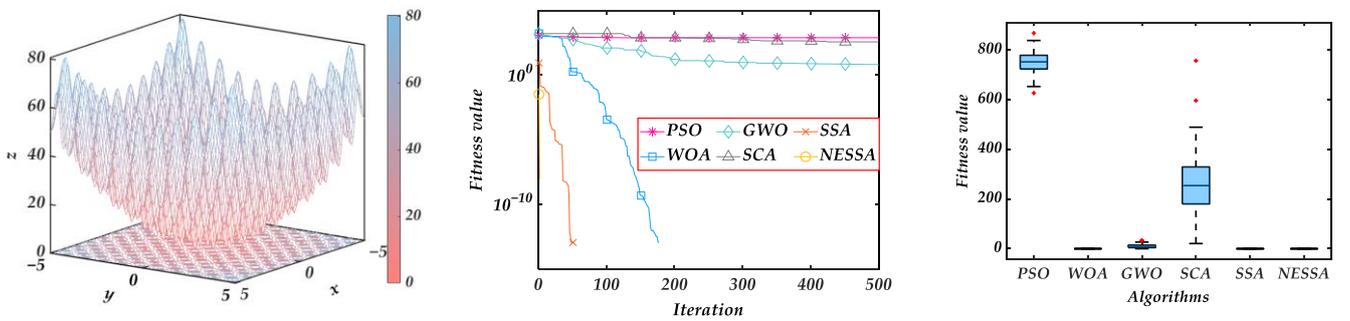


Figure 14. Rastrigin's Function.

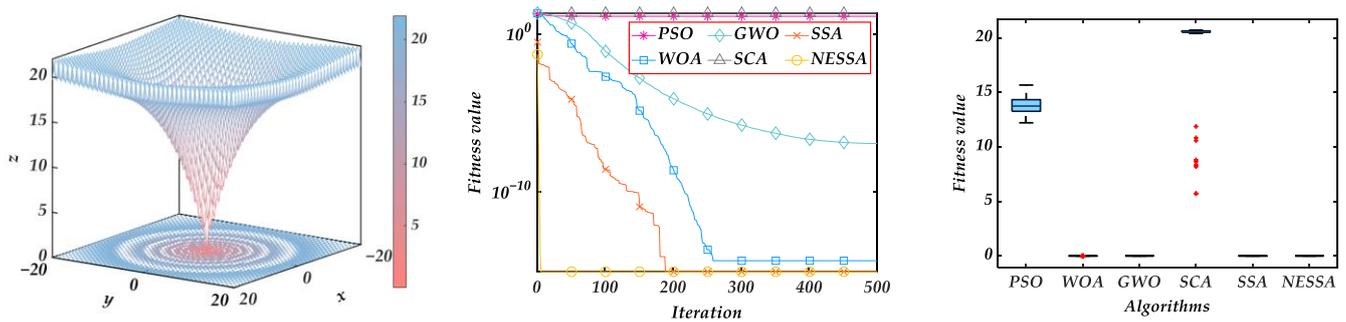


Figure 15. Ackley's Function.

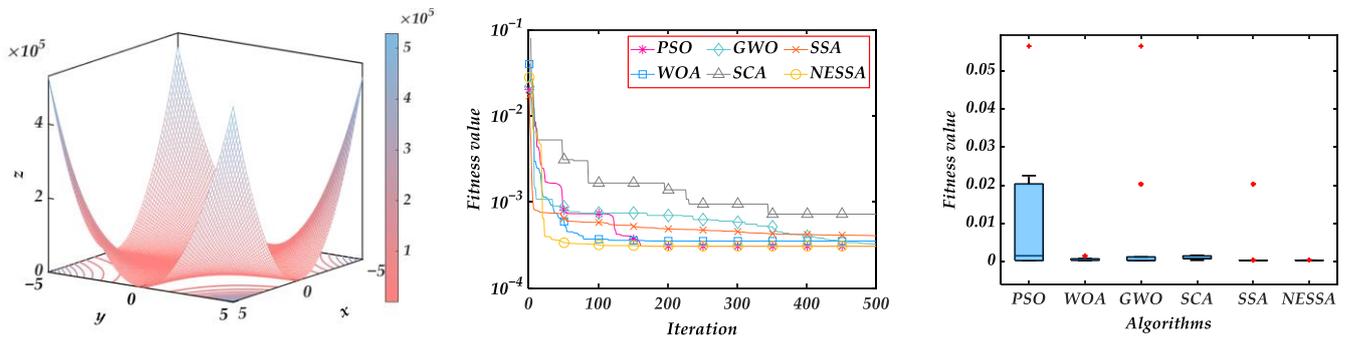


Figure 16. Kowalik's Function.

At present, improved SSA versions based on different strategies have been proposed by scholars. In this paper, the sparrow search algorithm based on the Sobol sequence and crisscross strategy (SSASC), the improved sparrow search algorithm combining the sine search and the diversity mutation processing (ISSA1), and the improved sparrow algorithm combining Cauchy mutation and opposition-based learning (ISSA2) in the references [39–41] are selected as a comparison, and further tests are conducted on 12 benchmark functions of 500 dimensions. All parameter settings are the same as SSA, and each algorithm runs 50 times independently.

In the initialization phase, ISSA1 and the original SSA adopt the random population method, while SSASC and ISSA2 adopt the chaotic sequence generated by the Sobol map and the Sin map as the initial population, respectively. In the iteration phase, the three improved SSA versions all introduce dynamic inertia weights into the iterative equations, which played an adaptive role in adjusting the algorithm. In addition, SSASC introduces the concept of the genetic algorithm (GA) [42] to optimize the fitness of some individuals by crossover and mutation in each iteration. ISSA1 implements Cauchy mutation on the individuals trapped in the local optimum to improve the diversity of the population. ISSA2 adds a reverse learning strategy based on the Cauchy mutation of ISSA1, and the two operations are performed alternately according to the selection probability in each iteration.

Similar to NESSA, the above three improved SSA versions have made efforts in balancing the local search and global exploration of the algorithm, and also have supplemented the mutation mechanism in each iteration, which has improved the ability of the algorithm to escape the local extremum. However, the convergence characteristics introduced by different iterative equations need to be further investigated on the high-dimensional benchmark functions. Table 4 shows the solution results of SSA, SSASC, ISSA1, ISSA2, and NESSA on F_1 to F_{12} when the dimension is 500, and their iterative trajectories are provided in Figure 17.

Table 4. Test results of five algorithms.

Benchmark	Algorithm	Mean	Std.	Time	p-Value
F ₁	SSA	6.1212×10^{-41}	4.1750×10^{-40}	9.3090	1.6907×10^{-18}
	SSASC	0	0	12.9206	N/A
	ISSA1	4.6707×10^{-80}	1.7467×10^{-79}	9.2791	7.3684×10^{-16}
	ISSA2	8.3657×10^{-176}	3.0297×10^{-175}	9.2900	3.3111×10^{-20}
	NESSA	0	0	7.5399	
F ₂	SSA	3.1521×10^{-42}	2.9376×10^{-41}	9.2487	4.6715×10^{-19}
	SSASC	0	0	11.7667	N/A
	ISSA1	1.1675×10^{-122}	7.7152×10^{-121}	9.5825	1.4596×10^{-12}
	ISSA2	6.7188×10^{-198}	3.8761×10^{-197}	9.5071	3.3111×10^{-20}
	NESSA	0	0	7.2796	
F ₃	SSA	2.0271×10^{-43}	1.4299×10^{-42}	16.5706	1.2593×10^{-19}
	SSASC	0	0	23.7165	N/A
	ISSA1	5.3247×10^{-76}	3.7646×10^{-75}	12.3822	5.2454×10^{-13}
	ISSA2	2.8774×10^{-181}	2.0346×10^{-180}	13.2727	3.3111×10^{-20}
	NESSA	0	0	14.8767	
F ₄	SSA	1.5016×10^{-41}	9.7589×10^{-41}	9.3427	4.6715×10^{-19}
	SSASC	0	0	12.7269	N/A
	ISSA1	1.9506×10^{-88}	1.3375×10^{-87}	9.3298	1.4596×10^{-12}
	ISSA2	4.5823×10^{-182}	3.2275×10^{-181}	9.5706	3.3111×10^{-20}
	NESSA	0	0	6.9401	
F ₅	SSA	1.8602×10^{-32}	1.3011×10^{-31}	9.8776	3.3111×10^{-20}
	SSASC	0	0	14.4102	N/A
	ISSA1	5.3900×10^{-63}	3.8108×10^{-63}	9.1247	1.4596×10^{-12}
	ISSA2	1.5771×10^{-107}	1.0312×10^{-107}	9.4143	3.3111×10^{-20}
	NESSA	0	0	7.7876	
F ₆	SSA	1.0350×10^{-3}	6.8883×10^{-4}	11.0312	1.7355×10^{-15}
	SSASC	1.5502×10^{-4}	1.3788×10^{-4}	15.2722	8.5865×10^{-4}
	ISSA1	3.6258×10^{-4}	2.5674×10^{-4}	10.9255	9.4565×10^{-12}
	ISSA2	2.4691×10^{-4}	3.0829×10^{-4}	11.1934	2.8105×10^{-12}
	NESSA	7.7617×10^{-5}	6.6362×10^{-5}	7.5121	
F ₇	SSA	1.3791×10^{-46}	9.6292×10^{-46}	9.0189	1.2593×10^{-19}
	SSASC	0	0	12.5317	N/A
	ISSA1	1.3618×10^{-70}	1.7302×10^{-71}	9.2598	1.4596×10^{-12}
	ISSA2	3.2087×10^{-163}	2.2657×10^{-162}	9.4015	3.3111×10^{-20}
	NESSA	0	0	7.2973	
F ₈	SSA	4.3985×10^{-25}	2.6406×10^{-24}	9.2677	3.3111×10^{-20}
	SSASC	0	0	13.7007	N/A
	ISSA1	8.5571×10^{-38}	4.5731×10^{-37}	8.8911	7.3687×10^{-16}
	ISSA2	2.0627×10^{-97}	1.4497×10^{-96}	9.0786	3.3111×10^{-20}
	NESSA	0	0	7.0836	

Table 4. Cont.

Benchmark	Algorithm	Mean	Std.	Time	<i>p</i> -Value
F_9	SSA	5.9257×10^{-37}	4.1898×10^{-36}	14.3982	4.6715×10^{-19}
	SSASC	0	0	23.2557	N/A
	ISSA1	1.2759×10^{-56}	9.0185×10^{-55}	14.3770	1.4596×10^{-12}
	ISSA2	8.7874×10^{-186}	6.1896×10^{-185}	15.3228	3.3111×10^{-20}
	NESSA	0	0	12.7107	
F_{10}	SSA	2.7796×10^{-21}	1.4596×10^{-20}	9.1717	3.3111×10^{-20}
	SSASC	0	0	13.9827	N/A
	ISSA1	1.2358×10^{-61}	6.1296×10^{-60}	9.0212	2.0607×10^{-17}
	ISSA2	5.7257×10^{-92}	4.0786×10^{-91}	9.1532	3.3111×10^{-20}
	NESSA	0	0	6.5867	
F_{11}	SSA	0	0	9.1857	N/A
	SSASC	0	0	15.5796	N/A
	ISSA1	0	0	9.3172	N/A
	ISSA2	0	0	9.3216	N/A
	NESSA	0	0	6.6278	
F_{12}	SSA	8.8818×10^{-16}	0	9.2862	N/A
	SSASC	8.8818×10^{-16}	0	11.7655	N/A
	ISSA1	8.8818×10^{-16}	0	9.1125	N/A
	ISSA2	8.8818×10^{-16}	0	9.3269	N/A
	NESSA	8.8818×10^{-16}	0	6.7362	

It can be seen from Table 4 that the average value of SSA in all tests has slightly decreased due to the increased dimensions of the benchmark functions, while the accuracy of F_{11} and F_{12} is maintained at a high level. For the three different improved algorithms, the average values obtained by them are more accurate than the standard SSA from F_1 to F_{10} . Among them, SSASC has the best performance, and its results are the same as NESSA in terms of numerical value, with the highest accuracy and the smallest standard deviation, representing the stability of the solution. However, SSASC has the longest running time and is slower than SSA in all tests, while NESSA is the shortest, which indicates the superiority of the proposed algorithm in terms of iteration efficiency. It can be seen from the iterative trajectory in Figure 17 that NESSA maintains a fast convergence speed, and it can converge to the optimum in less than 50 iterations from F_1 to F_5 , F_7 , F_9 , and F_{11} to F_{12} . It is also the algorithm with the highest convergence speed among the four improved SSA versions, which, once again, reflects the better optimization efficiency.

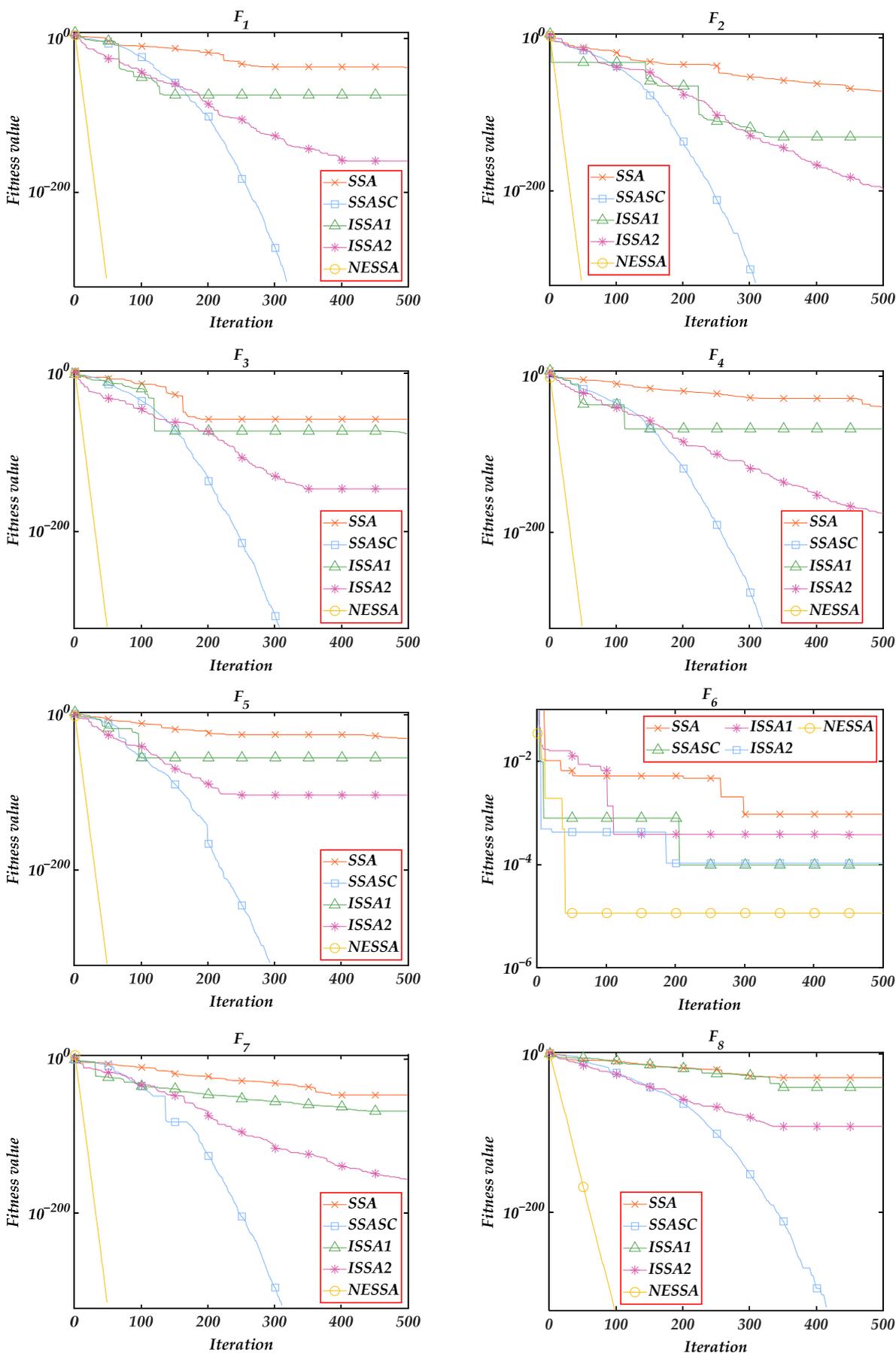


Figure 17. Cont.

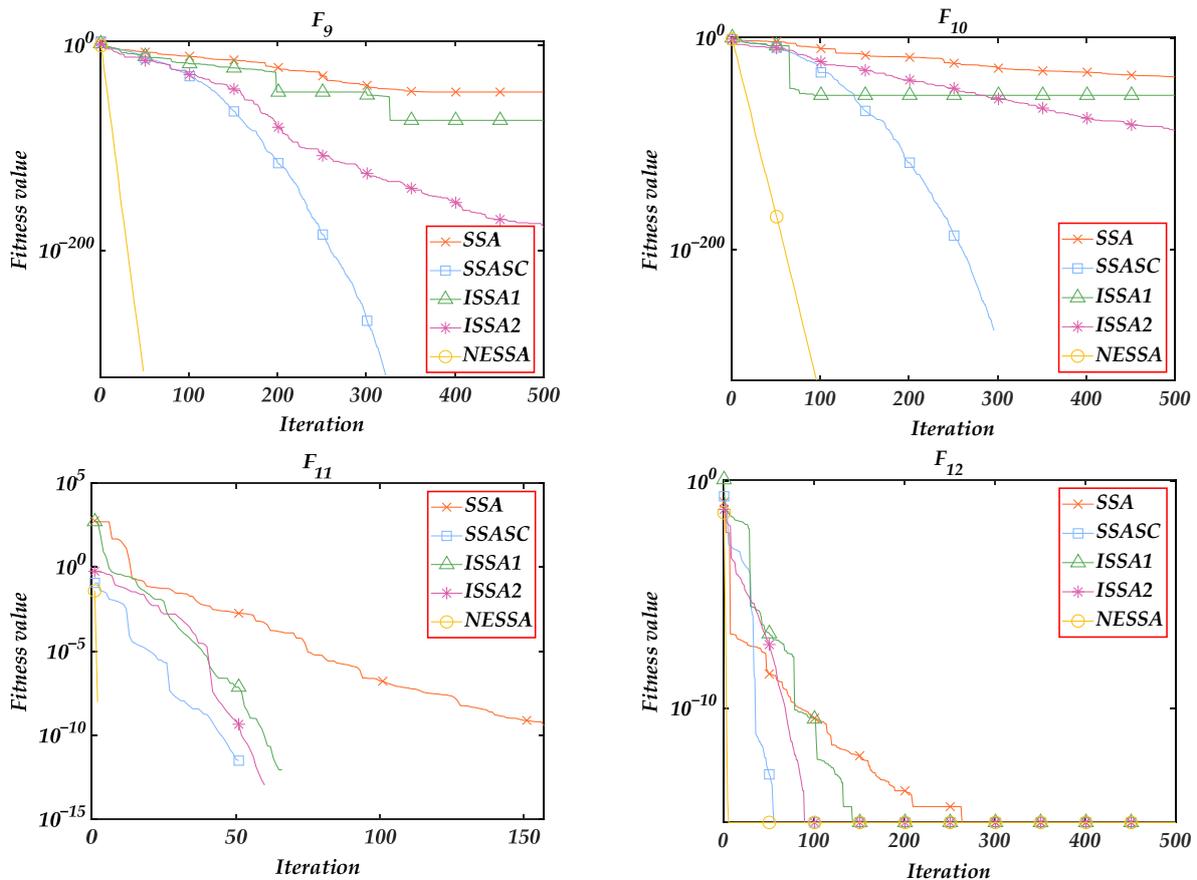


Figure 17. Iterative trajectories of five algorithms.

3. WSN Coverage Optimization

In this section, we introduce a coverage optimization framework of WSN based on the swarm intelligence algorithm, follow the parameter conditions in Section 2.4.1, and further discuss the optimization performance of NESSA concerning application problems in three cases.

3.1. Mathematical Modeling

In a WSN, sensor nodes are the basic unit for sensing and collecting external information and the coverage rate of all nodes can reflect the monitoring and tracking of information in the region, which is an essential embodiment of WSN service quality [43–45]. Hence, this paper takes the coverage rate as the evaluation index of coverage optimization and focuses on the maximization of coverage rate in monitoring areas.

Assume S is a $l \times l$ two-dimensional plane in which n sensor nodes with the same attributes are placed, which is defined as $c = (c_1, c_2, \dots, c_n)$. Each node has the same sensing radius r and communication radius R and meets $R = 2r$, where $c_i = (x_i, y_i), i = 1, 2, \dots, n$ represents a closed circular area with (x_i, y_i) as the center and r as the radius. The area S is discretized into $p \times q$ target points, and their coordinates are defined as $z_j = (x_j, y_j), j = 1, 2, \dots, p \times q$, then the Euclidean distance from the sensor node to the target point can be expressed as follows:

$$d(c_i, z_j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \quad (16)$$

the Boolean model is utilized to describe the probability that any target point z_j is covered by the sensor node c_i as follows:

$$p(c_i, z_j) = \begin{cases} 1, & d(c_i, z) \leq r \\ 0, & \text{else} \end{cases} \quad (17)$$

where $p(c_i, z_j)$ is the sensing probability. If $d \leq r$ is satisfied, it indicates that the target point has been covered by the sensing node. In the monitoring area, if the target point can be sensed by multiple sensor nodes, the joint sensing probability of z_j is defined as follows:

$$p(c, z_j) = 1 - \prod_{i=1}^n (1 - p(c_i, z_j)) \quad (18)$$

where $p(c, z_j)$ is the joint sensing probability, n is the number of sensor nodes of the node, and c is the set of all sensors. Then, the WSN coverage rate can be expressed as the ratio of the number of target points covered by the sensor set to all target points in the region, as follows:

$$p_{\text{cov}} = \frac{\sum_{j=1}^{m \times n} p(c, z_j)}{m \times n} \quad (19)$$

where p_{cov} is the area coverage. In this paper, Equation (19) is utilized as the objective function of the WSN coverage optimization problem to find an effective node deployment scheme that can increase p_{cov} .

3.2. Optimization Process Based on Swarm Intelligence Algorithms

In this study, swarm intelligence algorithms optimize the deployment coordinates of sensor nodes to maximize the coverage of the monitoring area. In the two-dimensional plane, the dimension value of an individual is twice the number of sensor nodes. The $2i$ th and $2i - 1$ th dimensions are the x -axis coordinates and y -axis coordinates of the i th sensor node, respectively, and each individual in the population represents a deployment scheme containing the coordinates of all sensor nodes. Take Equation (19) as the objective the function of the swarm intelligence algorithm; that is, the individual with the best fitness in the population, represents the deployment scheme with the largest coverage rate.

In the simulation, the parameters to be set include the side length of the monitoring area l , the number of sensors n , the sensing radius r , the discrete step size $Step$, the number of populations pop , the maximum number of iterations $Iter_{\text{max}}$, and the control parameters of the specific algorithm. The universality process of using a swarm intelligence algorithm to optimize the application problem is as follows, and its flowchart is shown in Figure 18.

- (1) Relevant parameters of the monitoring area and specific control parameters of the swarm intelligence algorithm are set.
- (2) The population is initialized and the initial coverage rate is obtained by calculating the objective function.
- (3) The algorithm iterates circularly to update the location of individuals in the search space.
- (4) The objective function value is evaluated to find the optimal individual with the best fitness; that is, the current optimal node deployment scheme is obtained.
- (5) Whether the maximum number of iterations is reached is determined. If yes, the algorithm is terminated and the optimal coverage rate and corresponding node coordinates are output. Otherwise, step 3 is repeated to continue the program, and one is added to the current iterations.

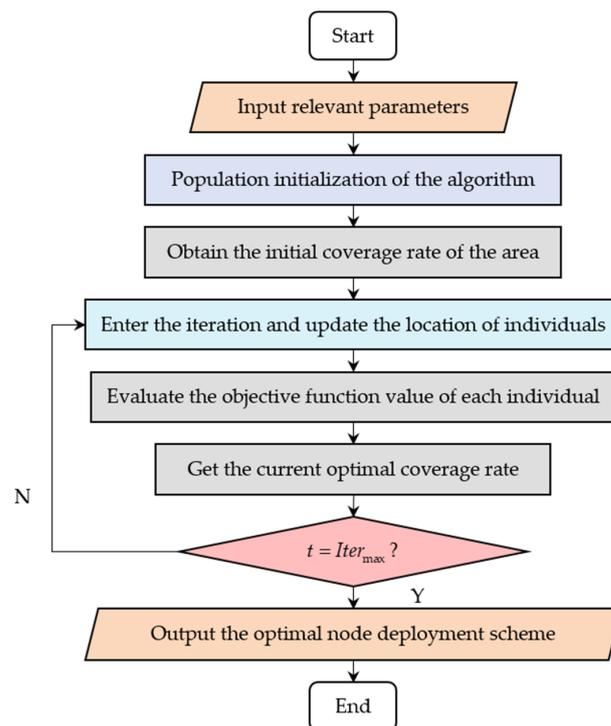


Figure 18. Optimization process based on swarm intelligence algorithm.

3.3. Case Studies

Based on the coverage optimization framework of the swarm intelligence algorithm, we further investigate the optimization performance of PSO, WOA, GWO, SCA, SSA, NESSA, and the three different improved SSA versions (SSASC, ISSA1, and ISSA2) in three WSN deployment cases. The control parameters of all algorithms are the same as those in Section 2.4.1, and the general parameters are set as $pop = 30$, $Iter_{max} = 500$, and $Step = 1$. Each case runs 30 times independently and records the optimal value, average value, standard deviation, and running time(s) of the coverage rate as the evaluation indicators of the optimization performance.

3.3.1. Case 1

In case 1, the monitoring area is a two-dimensional square plane of 30×30 (m²), which is discretized into 31×31 target points, in which 20 sensor nodes with the same structure are deployed, and the sensing radius of each node is $r = 5$ (m). Table 5 records the optimization results of case 1. The coverage rate obtained by each algorithm and the node deployment scheme is shown in Figures 19 and 20.

Table 5. Optimization results for case 1.

Algorithm	Coverage Rate				<i>p</i> -Value vs. NESSA	Average Time
	Optimal	Worst	Mean	Std.		
PSO	0.9771	0.9344	0.9585	1.2067×10^{-2}	1.1960×10^{-12}	11.2371
WOA	0.9812	0.9094	0.9521	1.6991×10^{-2}	1.2019×10^{-12}	11.5263
GWO	1.0000	0.9906	0.9983	2.0740×10^{-3}	1.2045×10^{-7}	14.1711
SCA	0.9282	0.8607	0.8937	1.6331×10^{-2}	1.2049×10^{-12}	12.7758
SSA	0.9563	0.8821	0.9190	2.2586×10^{-2}	1.2039×10^{-12}	15.1322
SSASC	0.9625	0.8533	0.9116	2.8690×10^{-2}	1.2049×10^{-12}	40.0176
ISSA1	0.9823	0.8595	0.9433	2.7137×10^{-2}	1.1921×10^{-12}	14.2296
ISSA2	0.9761	0.8824	0.9205	2.2727×10^{-2}	1.2198×10^{-12}	14.8573
NESSA	1.0000	1.0000	1.0000	0		14.7867

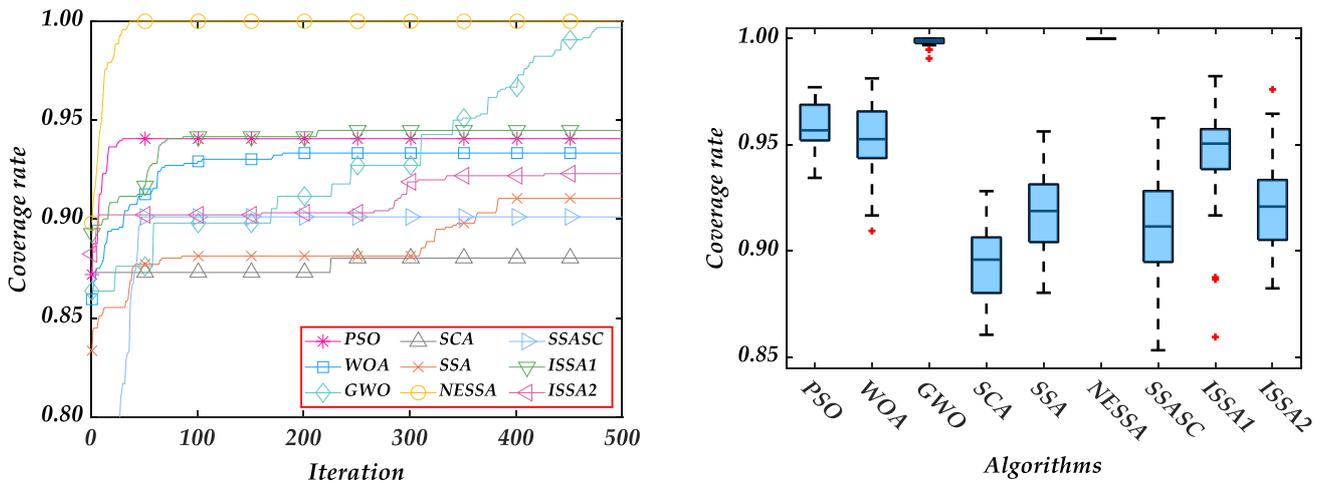


Figure 19. Coverage rate curves and boxplots for case 1.

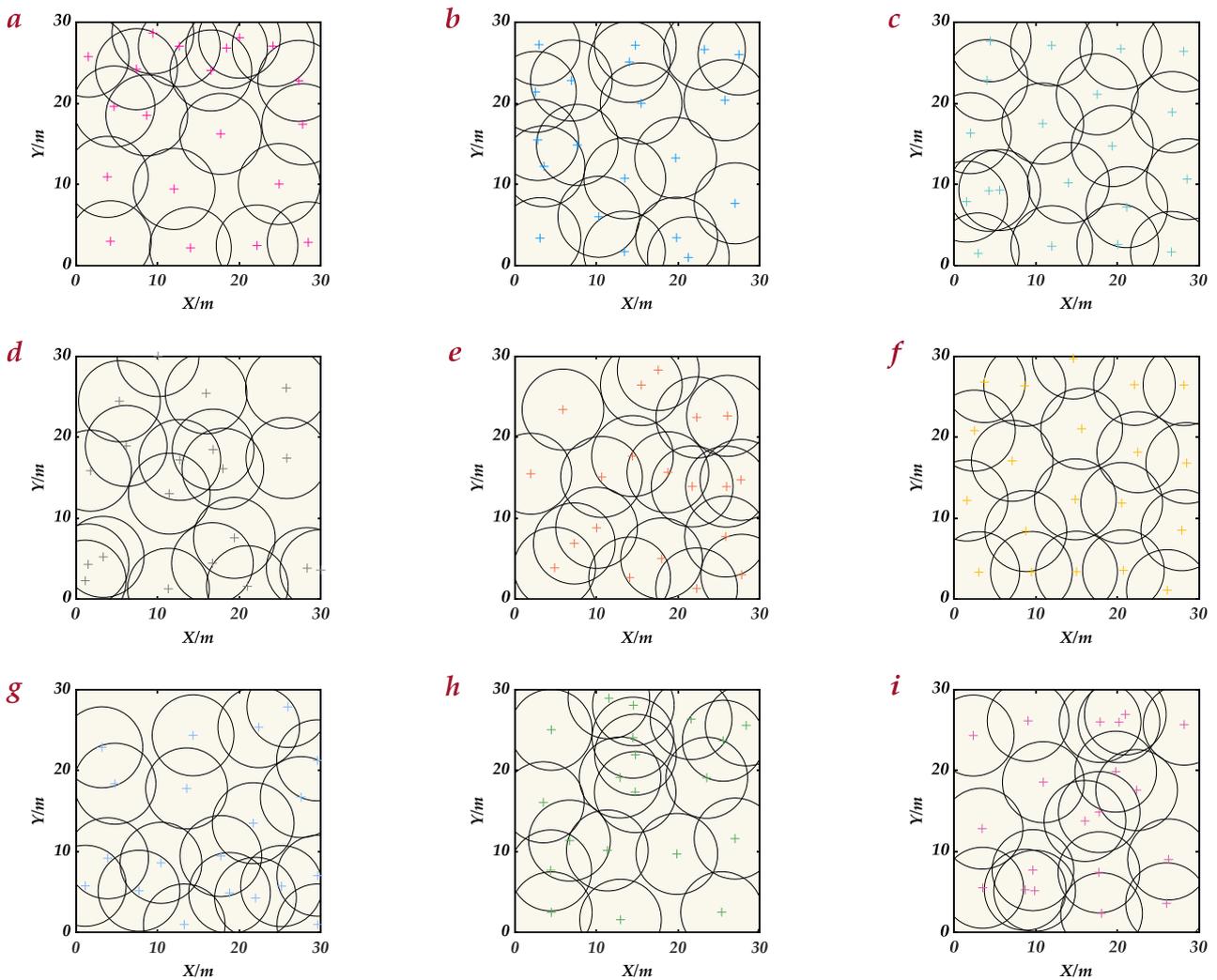


Figure 20. Node deployment of nine algorithms for case 1.

For the monitoring area of case 1, NESSA can stably achieve the optimization results of complete coverage in 30 independent tests, and the p -value obtained from the Wilcoxon test is always less than 0.05, which is significantly different from other algorithms. GWO can also achieve the optimal result of 100% coverage of the region, but its stability is not

as good as NESSA. It can be seen from Figure 19 that GWO only obtains a faster rising speed in the later iterations and maintains a low coverage rate in the earlier iterations. The optimization process of NESSA can achieve the optimal coverage rate in less than 40 iterations on average. Moreover, the average coverage rate of PSO and WOA reached more than 95%, ahead of 91.90% of SSA and 89.37% of SCA. Compared with the standard SSA, NESSA has boosted the optimal, worst, and average coverage rate by 4.37%, 11.79%, and 8.10%, respectively, which validates the effectiveness of the improved strategy. We can see the result distribution of each algorithm through the boxplot. Since the coverage rate of NESSA in 30 independent tests is always 100%, its boxplot is a straight line, while the results of other algorithms show the divergent state to a certain extent, which further indicates that the proposed algorithm has high stability and solution performance.

For the three improved SSA versions, the average coverage rates of ISSA1 and ISSA2 are lower than NESSA by 5.67% and 7.95%, respectively, and 2.43% and 0.15% higher than the standard SSA. Compared with NESSA, their improvement in case 1 is not significant. SSASC is even lower than the standard SSA by 0.74%, and the time spent is 2.7 times that of the proposed method. Although SSASC has reached a higher optimal coverage rate than the standard SSA, the results obtained in 30 tests are not stable, which can also be confirmed from the boxplot.

Subfigures *a–i* in Figure 20 correspond to PSO, WOA, GWO, SCA, SSA, NESSA, SSASC, ISSA1, and ISSA2, respectively. Through observation, it can be seen that the node distribution optimized by NESSA is more uniform than other algorithms, and the effective coverage of the region is realized.

3.3.2. Case 2

In case 2, the monitoring area is a two-dimensional square plane of 20×20 (m²), which is discretized into 21×21 target points, in which 24 sensor nodes with the same structure are deployed, and the sensing radius of each node is $r = 2.5$ (m). Table 6 records the optimization results of case 2. The coverage rate obtained by each algorithm and the node deployment scheme is shown in Figures 21 and 22.

Table 6. Optimization results for case 2.

Algorithm	Coverage Rate				<i>p</i> -Value vs. NESSA	Average Time
	Optimal	Worst	Mean	Std.		
PSO	0.8707	0.7619	0.8057	2.1728×10^{-2}	2.8809×10^{-11}	7.1351
WOA	0.8367	0.7607	0.7973	2.1916×10^{-2}	2.9045×10^{-11}	7.4728
GWO	0.9387	0.7437	0.9057	3.8881×10^{-2}	8.7576×10^{-9}	7.5063
SCA	0.7709	0.7233	0.7421	9.9217×10^{-3}	2.8287×10^{-11}	6.7501
SSA	0.7981	0.7188	0.7572	1.8913×10^{-2}	2.8827×10^{-11}	8.8687
SSASC	0.8752	0.7528	0.8215	2.9191×10^{-2}	2.8682×10^{-11}	21.6326
ISSA1	0.8357	0.7486	0.7878	2.1933×10^{-2}	2.8871×10^{-11}	8.6371
ISSA2	0.8299	0.7211	0.7680	2.2973×10^{-2}	2.8682×10^{-11}	9.1599
NESSA	0.9523	0.9161	0.9371	9.6561×10^{-3}		7.7711

NESSA continued its high-performance advantage in the optimization of case 2, and finally achieved an average coverage rate of 93.71. Compared with 90.57% of GWO, the optimal and the worst of NESSA are higher than 1.36% and 17.24%, respectively, while the average values of other algorithms are below 90%. The minimum standard deviation of NESSA indicates that the results have high stability, which is further shown in the boxplot. It can be seen from Figure 21 that the divergent state of NESSA results is smaller and there is no abnormal value. Furthermore, different from the acceleration of GWO at the later iterations, the coverage curve of NESSA has been ahead of other algorithms since the beginning of iterations and has finally increased by 15.42%, 19.73%, and 17.99%, respectively, in terms of optimal, worst and average value compared with the original version.

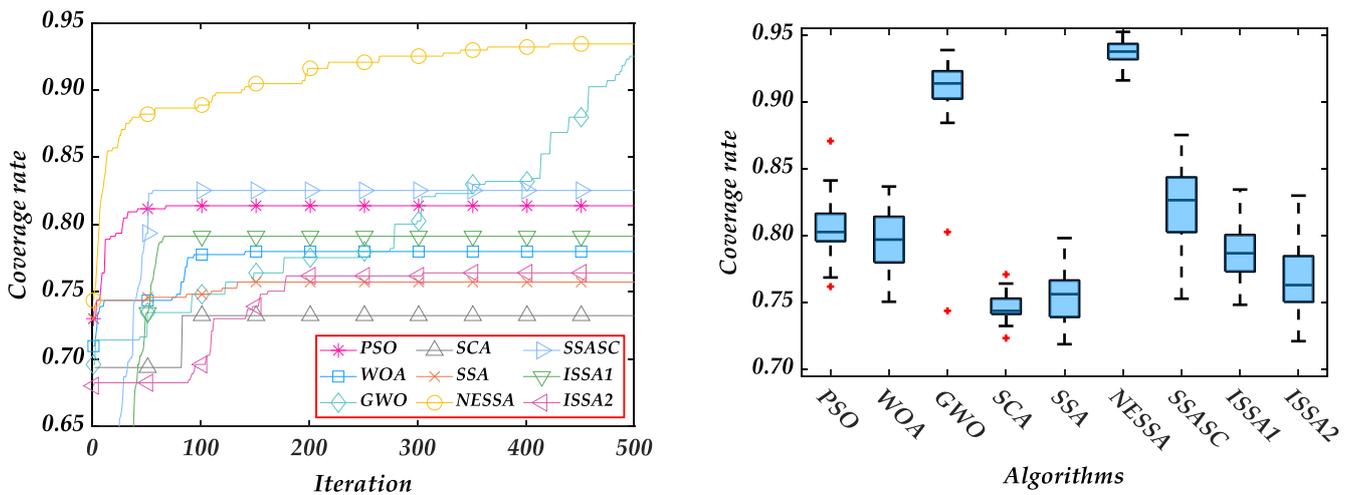


Figure 21. Coverage rate curves and boxplots for case 2.

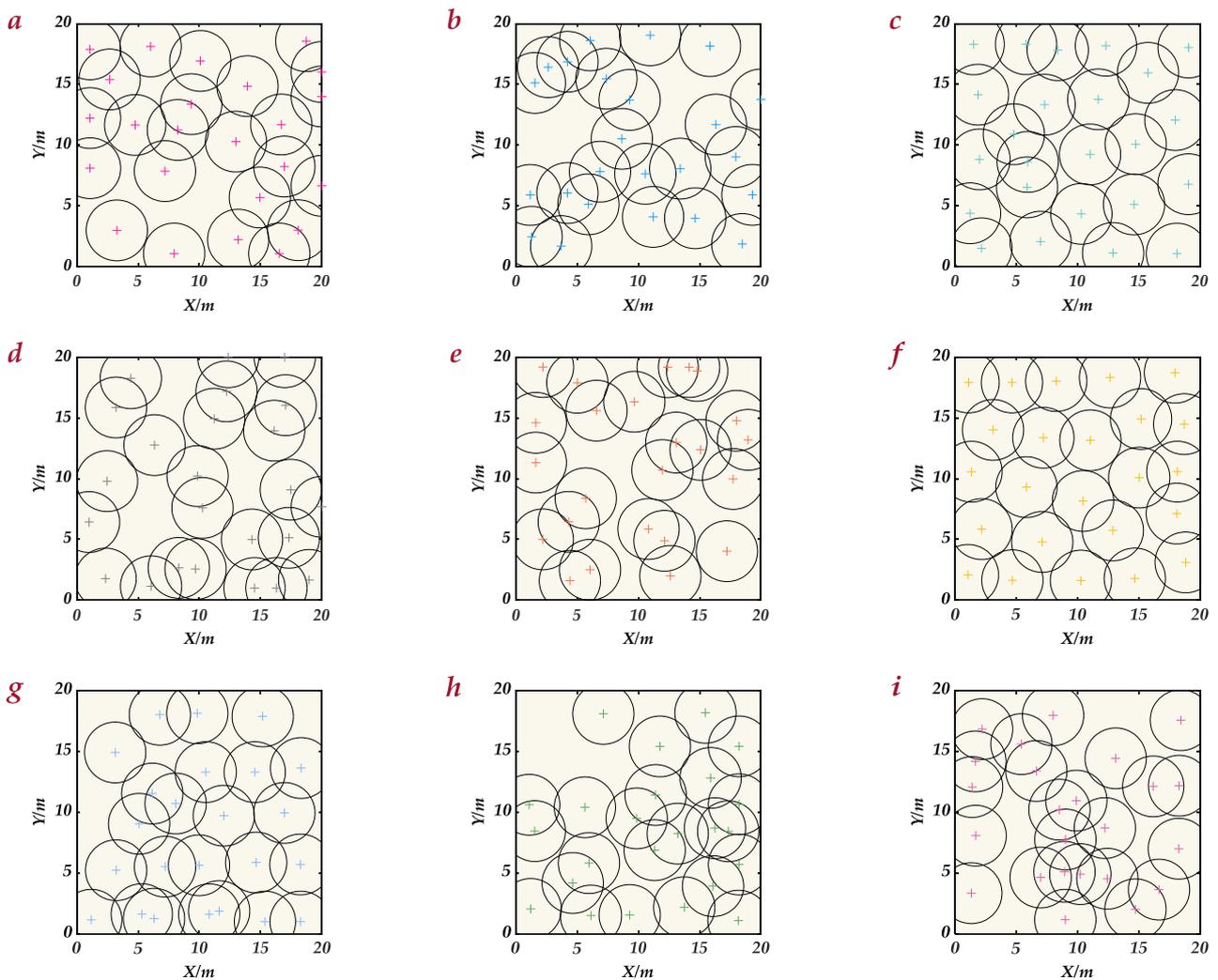


Figure 22. Node deployment of nine algorithms for case 2.

In case 2, SSASC, ISSA1, and ISSA2 increased their average coverage rates by 6.43%, 3.06%, and 1.08% based on the standard SSA. Compared with NESSA, their improvement is still not significant, and the coverage optimization effect is not ideal. It can be seen from

the node distribution in Figure 22 that similar to PSO, WOA, SCA, and SSA, the three improved algorithms also have large monitoring blind areas, while NESSA can better avoid this situation, and its node distribution shows a better uniformity than GWO, which further confirms the feasibility of the proposed method in coverage optimization.

3.3.3. Case 3

In case 3, the monitoring area is a two-dimensional square plane of 100×100 (m²), which is discretized into 101×101 target points, in which 50 sensor nodes with the same structure are deployed, and the sensing radius of each node is $r = 10$ (m). Table 7 records the optimization results of case 3. The coverage rate obtained by each algorithm and the node deployment scheme is shown in Figures 23 and 24.

Table 7. Optimization results for case 3.

Algorithm	Coverage Rate				p-Value vs. NESSA	Average Time
	Optimal	Worst	Mean	Std.		
PSO	0.9378	0.8958	0.9171	1.2070×10^{-2}	2.9972×10^{-11}	195.7561
WOA	0.9507	0.8888	0.9264	1.4455×10^{-2}	2.9953×10^{-11}	201.8856
GWO	0.9925	0.9687	0.9818	5.7332×10^{-3}	1.8486×10^{-10}	207.3233
SCA	0.8756	0.7897	0.8337	1.5766×10^{-2}	2.9953×10^{-11}	197.1167
SSA	0.8862	0.8271	0.8636	1.2899×10^{-2}	2.9935×10^{-11}	208.3121
SSASC	0.9411	0.7082	0.8532	6.1694×10^{-2}	3.0161×10^{-11}	702.6788
ISSA1	0.9447	0.8913	0.9156	1.3141×10^{-2}	2.9972×10^{-11}	216.7396
ISSA2	0.9207	0.8522	0.8770	1.3337×10^{-2}	3.0161×10^{-11}	219.6671
NESSA	0.9957	0.9885	0.9927	1.8950×10^{-3}		202.7887

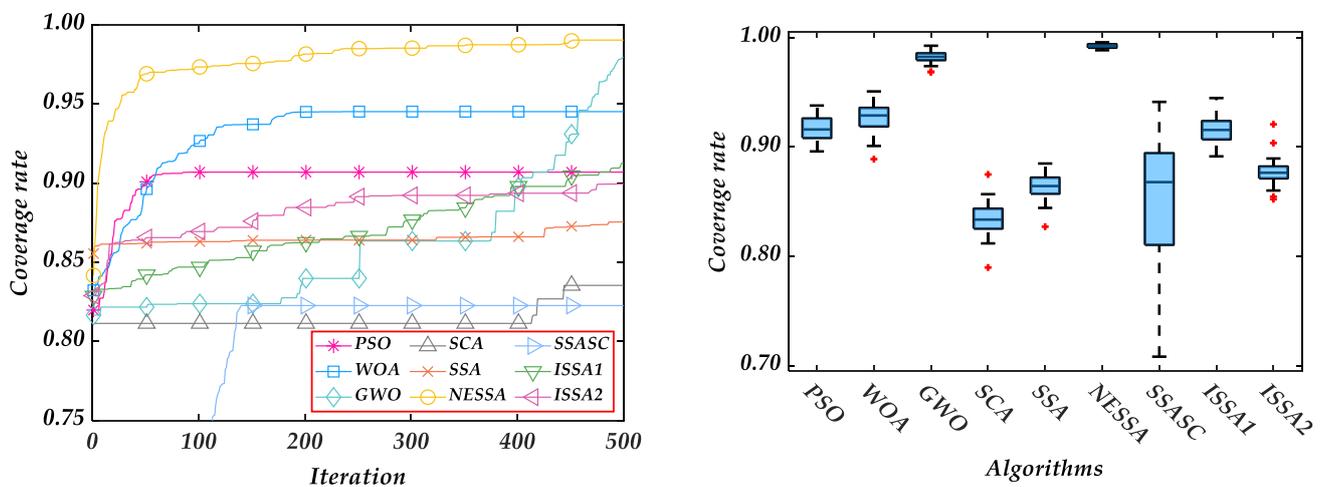


Figure 23. Coverage rate curves and boxplots for case 3.

In case 3, the number of discrete target points has reached 10201, so the complexity of the calculation process is significantly improved, and the running time of each algorithm is correspondingly increased. However, the slowest SSASC reaches 702 s, 3.2 to 3.6 times that of other algorithms, about 500 s slower than NESSA, and 13.95% lower than NESSA in the average coverage rate; ISSA1 and ISSA2 are also lower than NESSA by 7.71% and 11.57%, respectively. We can see from Figure 23 that the coverage curve of NESSA continues to maintain the leading trend. After 30 tests, it finally reaches the average value of 99.27%, which is increased by 10.95%, 16.14%, and 12.91% in terms of the optimal, worst, and average values compared with the original version. From the boxplots listed above, the results of NESSA show high concentration and stability. The optimization result of GWO is only inferior to that of NESSA, reaching an average value of 98.18%, better than that of

WOA, 92.64%, PSO, 91.71%, and ISSA1, 91.56%, while the optimal of ISSA1, SSA and SCA does not exceed 90%.

Although the three improved SSA versions have better accuracy than the standard SSA algorithm in the tests of benchmark functions, they perform mediocly in the optimization of coverage enhancement, and the results contrast sharply with the method proposed in this paper, which fully illustrates that the applicability of the improved algorithms based on different iteration mechanisms to the same problem is not completely consistent, and the specific performance of the algorithms needs to be further verified in simulation tests. In the three calculation cases, SSASC, ISSA1, and ISSA2 obtained an average coverage rate of 86.21%, 88.22%, and 85.52%, respectively. NESSA achieved the highest average coverage rate of 97.66% among the nine algorithms, which increased by 13.00% compared with the standard SSA algorithm, followed by 96.19% of GWO, 89.37% of PSO, 89.19% of WOA, and 82.32% of SCA with the worst performance.

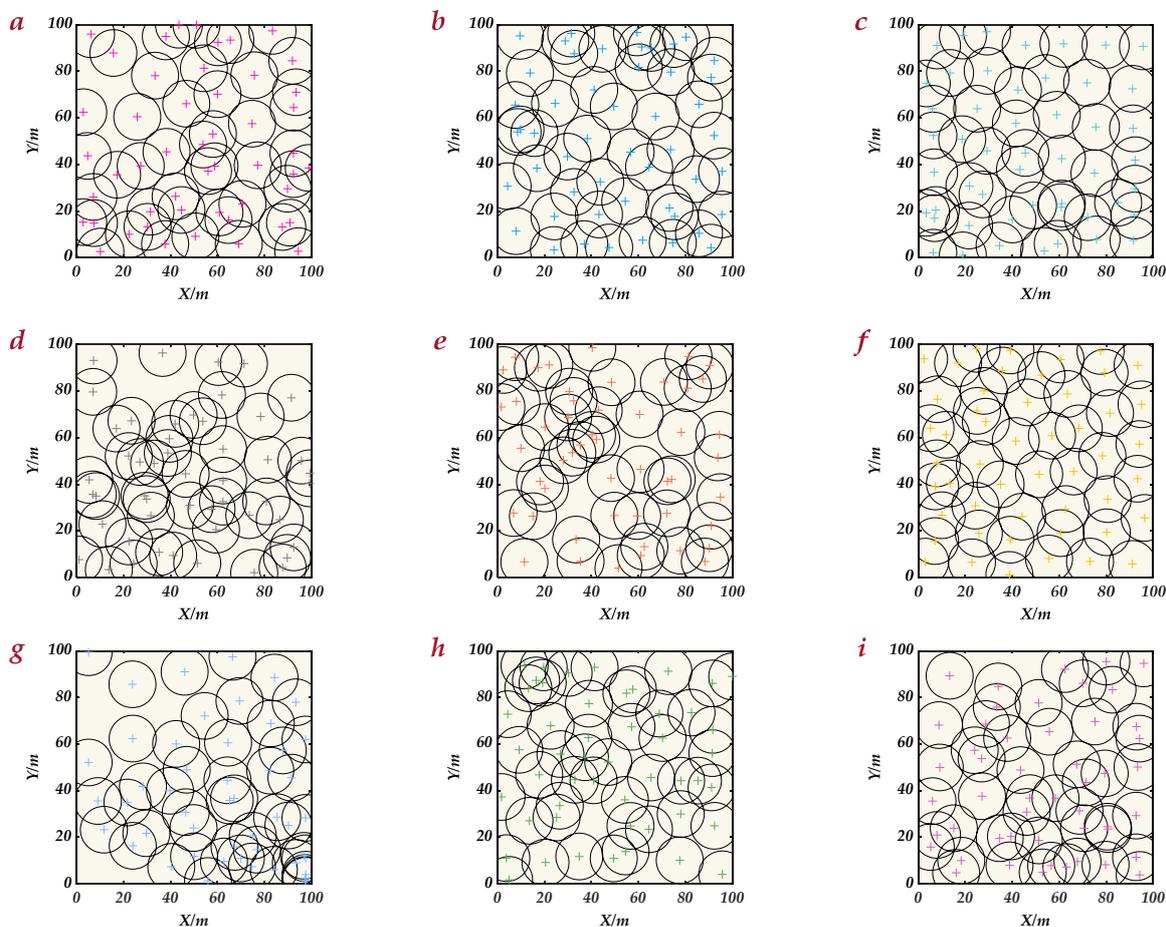


Figure 24. Node deployment of nine algorithms for case 3.

4. Conclusions

This paper focused on increasing the coverage rate of a WSN, proposed a node deployment optimization method based on NESSA, and improved the shortcomings of standard SSA from the perspective of solving application problems involving three aspects: population initialization, iterative search, and disturbance mutation, comprehensively improving the optimization performance of the algorithm, and validating the superiority of NESSA in convergence speed, accuracy, and robustness. Furthermore, the simulation results of three different cases show that NESSA is an effective WSN coverage optimization algorithm, which significantly improves the deployment quality of sensor nodes compared with its original version.

This study confirms the feasibility of NESSA in theory. However, more complex factors need to be considered in practical applications, such as the geographical environment, which is no longer two-dimensional but three-dimensional, the energy of sensor nodes, and the communication link between nodes after final deployment. Thus, the future research direction will be to complete the deployment optimization of the WSN under the premise of comprehensively considering multiple performance indicators and environmental factors.

Author Contributions: Conceptualization, R.L. and Y.M.; methodology, R.L.; software, R.L.; validation, Y.M.; formal analysis, R.L.; investigation, R.L.; writing—original draft preparation, R.L. and Y.M.; writing—review and editing, R.L. and Y.M.; visualization, R.L.; supervision, Y.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China under Grant No. 21466008; Guangxi Natural Science Foundation under Grant No. 2019GXNSFAA185017; the Scientific Research Project of Guangxi Minzu University under Grant No. 2021MDKJ004.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors acknowledge this research was supported by the National Natural Science Foundation of China under Grant No. 21466008; Guangxi Natural Science Foundation under Grant No. 2019GXNSFAA185017; the Scientific Research Project of Guangxi Minzu University under Grant No. 2021MDKJ004.

Conflicts of Interest: The authors declare no potential conflict of interest.

References

1. Du, C.; Zhang, L.; Ma, X. A Cotton High-Efficiency Water-Fertilizer Control System Using Wireless Sensor Network for Precision Agriculture. *Processes* **2021**, *9*, 1693. [[CrossRef](#)]
2. Peter, L.; Kracik, J.; Cerny, M. Mathematical model based on the shape of pulse waves measured at a single spot for the non-invasive prediction of blood pressure. *Processes* **2020**, *8*, 442. [[CrossRef](#)]
3. Adame, T.; Bel, A.; Carreras, A. CUIDATS: An RFID–WSN hybrid monitoring system for smart health care environments. *Future Gener. Comput. Syst.* **2018**, *78*, 602–615. [[CrossRef](#)]
4. Gong, C.; Guo, C.; Xu, H. A joint optimization strategy of coverage planning and energy scheduling for wireless rechargeable sensor networks. *Processes* **2020**, *8*, 1324. [[CrossRef](#)]
5. Ahmad, S.; Hussain, I.; Fayaz, M. A Distributed Approach towards Improved Dissemination Protocol for Smooth Handover in MediaSense IoT Platform. *Processes* **2018**, *6*, 46. [[CrossRef](#)]
6. Brezulianu, A.; Aghion, C.; Hagan, M. Active Control Parameters Monitoring for Freight Trains, Using Wireless Sensor Network Platform and Internet of Things. *Processes* **2020**, *8*, 639. [[CrossRef](#)]
7. Akyildiz, I.F.; Su, W.; Sankarasubramaniam, Y. Wireless sensor networks: A survey. *Comput. Netw.* **2002**, *38*, 393–422. [[CrossRef](#)]
8. Rajendran, S.; Čep, R.; RC, N.; Pal, S.; Kalita, K. A Conceptual Comparison of Six Nature-Inspired Metaheuristic Algorithms in Process Optimization. *Processes* **2022**, *10*, 197. [[CrossRef](#)]
9. Shokouhifar, M. FH-ACO: Fuzzy heuristic-based ant colony optimization for joint virtual network function placement and routing. *Appl. Soft Comput.* **2021**, *107*, 107401. [[CrossRef](#)]
10. Liao, W.H.; Kao, Y.; Wu, R.T. Ant colony optimization based sensor deployment protocol for wireless sensor networks. *Expert Syst. Appl.* **2011**, *38*, 6599–6605. [[CrossRef](#)]
11. Yoon, Y.; Kim, Y.H. An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks. *IEEE Trans. Cybern.* **2013**, *43*, 1473–1483. [[CrossRef](#)] [[PubMed](#)]
12. Wang, L.; Wu, W.; Qi, J. Wireless sensor network coverage optimization based on whale group algorithm. *Comput. Sci. Inf. Syst.* **2018**, *15*, 569–583. [[CrossRef](#)]
13. Wang, X.; Zhang, H.; Fan, S. Coverage control of sensor networks in IoT based on RPSO. *IEEE Internet Things J.* **2018**, *5*, 3521–3532. [[CrossRef](#)]
14. Miao, Z.; Yuan, X.; Zhou, F. Grey wolf optimizer with an enhanced hierarchy and its application to the wireless sensor network coverage optimization problem. *Appl. Soft Comput.* **2020**, *96*, 106602. [[CrossRef](#)]
15. Zhu, F.; Wang, W. A coverage optimization method for WSNs based on the improved weed algorithm. *Sensors* **2021**, *21*, 5869. [[CrossRef](#)]
16. Shokouhifar, M. Swarm intelligence RFID network planning using multi-antenna readers for asset tracking in hospital environments. *Comput. Netw.* **2021**, *198*, 108427. [[CrossRef](#)]

17. He, Q.; Lan, Z.; Zhang, D.; Yang, L.; Luo, S. Improved Marine Predator Algorithm for Wireless Sensor Network Coverage Optimization Problem. *Sustainability* **2022**, *14*, 9944. [[CrossRef](#)]
18. Xue, J.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control Eng.* **2020**, *8*, 22–34. [[CrossRef](#)]
19. Zhang, F.; Sun, W.; Wang, H. Fault diagnosis of a wind turbine gearbox based on improved variational mode algorithm and information entropy. *Entropy* **2021**, *23*, 794. [[CrossRef](#)]
20. Li, X.; Li, S.; Zhou, P. Forecasting Network Interface Flow Using a Broad Learning System Based on the Sparrow Search Algorithm. *Entropy* **2022**, *24*, 478. [[CrossRef](#)]
21. Liu, R.; Mo, Y.; Lu, Y. Swarm-Intelligence Optimization Method for Dynamic Optimization Problem. *Mathematics* **2022**, *10*, 1803. [[CrossRef](#)]
22. Nguyen, T.T.; Ngo, T.G.; Dao, T.K. Microgrid Operations Planning Based on Improving the Flying Sparrow Search Algorithm. *Symmetry* **2022**, *14*, 168. [[CrossRef](#)]
23. Ma, J.; Hao, Z.; Sun, W. Enhancing sparrow search algorithm via multi-strategies for continuous optimization problems. *Inf. Process. Manag.* **2022**, *59*, 102854. [[CrossRef](#)]
24. Xiong, Q.; Zhang, X.; He, S. A Fractional-Order Chaotic Sparrow Search Algorithm for Enhancement of Long Distance Iris Image. *Mathematics* **2021**, *9*, 2790. [[CrossRef](#)]
25. Zhang, C.; Ding, S. A stochastic configuration network based on chaotic sparrow search algorithm. *Knowl. Based Syst.* **2021**, *220*, 106924. [[CrossRef](#)]
26. Bacanin, N.; Stoean, R.; Zivkovic, M. Performance of a novel chaotic firefly algorithm with enhanced exploration for tackling global optimization problems: Application for dropout regularization. *Mathematics* **2021**, *9*, 2705. [[CrossRef](#)]
27. McKay, M.D.; Beckman, R.J.; Conover, W.J. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* **2000**, *42*, 55–61. [[CrossRef](#)]
28. Donovan, D.; Burrage, K.; Burrage, P. Estimates of the coverage of parameter space by Latin Hypercube and Orthogonal Array-based sampling. *Appl. Math. Model.* **2018**, *57*, 553–564. [[CrossRef](#)]
29. Chen, X.; Li, K.; Xu, B. Biogeography-based learning particle swarm optimization for combined heat and power economic dispatch problem. *Knowl. Based Syst.* **2020**, *208*, 106463. [[CrossRef](#)]
30. Mirjalili, S. SCA: A sine cosine algorithm for solving optimization problems. *Knowl. Based Syst.* **2016**, *96*, 120–133. [[CrossRef](#)]
31. Mirjalili, S. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural. Comput. Appl.* **2016**, *27*, 1053–1073. [[CrossRef](#)]
32. Sarafrazi, S.; Nezamabadi-pour, H.; Saryazdi, S. Disruption: A new operator in gravitational search algorithm. *Sci. Iran.* **2011**, *18*, 539–548. [[CrossRef](#)]
33. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [[CrossRef](#)]
34. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [[CrossRef](#)]
35. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [[CrossRef](#)]
36. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [[CrossRef](#)]
37. Hussain, K.; Mohd, M.N.; Cheng, S. Metaheuristic research: A comprehensive survey. *Artif. Intell. Rev.* **2019**, *52*, 2191–2233. [[CrossRef](#)]
38. Derrac, J.; García, S.; Molina, D. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [[CrossRef](#)]
39. Duan, Y.; Liu, C. Sparrow search algorithm based on Sobol sequence and crisscross strategy. *J. Comput. Appl.* **2022**, *42*, 36.
40. Zhang, X.; Zhang, Y.; Liu, L. Improved sparrow search algorithm fused with multiple strategies. *Appl. Res. Comput.* **2022**, *39*, 1086–1091.
41. Mao, Q.; Zhang, Q. Improved sparrow algorithm combining Cauchy mutation and Opposition-based learning. *J. Front. Comput. Sci. Technol.* **2021**, *15*, 1155.
42. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on genetic algorithm: Past, present, and future. *Multimed. Tools. Appl.* **2021**, *80*, 8091–8126. [[CrossRef](#)] [[PubMed](#)]
43. Wang, X.; Tan, G.; Lu, F.L. A molecular force field-based optimal deployment algorithm for UAV swarm coverage maximization in mobile wireless sensor network. *Processes* **2020**, *8*, 369. [[CrossRef](#)]
44. Lian, F.L.; Moyne, J.; Tilbury, D. Network design consideration for distributed control systems. *IEEE Trans. Control Syst. Technol.* **2002**, *10*, 297–307. [[CrossRef](#)]
45. Wang, X.; Wang, S.; Ma, J.J. An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment. *Sensors* **2007**, *7*, 354–370. [[CrossRef](#)]