Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 1048879, 26 pages
https://doi.org/10.1155/2021/1048879

Research Article

A Novel Crow Search Algorithm Based on Improved Flower Pollination

Qian Cheng,[1] **Huajuan Huang**,[2] **and Minbo Chen**[1]

[1]*College of Electronic Information, Guangxi University for Nationalities, Nanning 530000, China*
[2]*College of Artificial Intelligence, Guangxi University for Nationalities, Nanning 530000, China*

Correspondence should be addressed to Huajuan Huang; hhj-025@163.com

Received 11 August 2021; Revised 18 September 2021; Accepted 22 September 2021; Published 26 October 2021

Academic Editor: Qingzheng XU

Crow search algorithm (CSA) is a new type of swarm intelligence optimization algorithm proposed by simulating the crows' intelligent behavior of hiding and retrieving food. The algorithm has the characteristics of simple structure, few control parameters, and easy implementation. Like most optimization algorithms, the crow search algorithm also has the disadvantage of slow convergence and easy fall into local optimum. Therefore, a crow search algorithm based on improved flower pollination algorithm (IFCSA) is proposed to solve these problems. First, the search ability of the algorithm is balanced by the reasonable change of awareness probability, and then the convergence speed of the algorithm is improved. Second, when the leader finds himself followed, the cross-pollination strategy with Cauchy mutation is introduced to avoid the blindness of individual location update, thus improving the accuracy of the algorithm. Experiments on twenty benchmark problems and speed reducer design were conducted to compare the performance of IFCSA with that of other algorithms. The results show that IFCSA has better performance in function optimization and speed reducer design problem.

1. Introduction

In modern societies where global resources are increasingly scarce, optimization has become one of the most important and hottest research topics [1]. It is in the core process of various engineering fields such as engineering, science, energy, and computer. With the increasing complexity of scientific and engineering problems, traditional mathematical methods sometimes cannot solve them well. Therefore, some scholars have proposed metaheuristic algorithms, such as particle swarm optimization (PSO) [2], bat algorithm (BA) [3], butterfly optimization algorithm (BOA) [4], flower pollination algorithm (FPA) [5], pigeon inspired optimization (PIO) [6], whale optimization algorithm (WOA) [7], gray wolf optimizer (GWO) [8], and teaching-learning-based optimization (TLBO) [9]. Compared with traditional algorithms, these intelligent algorithms can make up for the defects of the traditional algorithm in the problem of great complexity. However, they still have the problems of low solution accuracy and slow convergence speed.

Crow search algorithm (CSA) [10] is a new swarm intelligence optimization algorithm proposed by Askarzadeh in 2016. The algorithm has the advantages of simple structure, fewer control parameters, and easy implementation. Now it has been widely used to solve various practical optimization problems in chemical engineering and QSAR [11], image processing [12], feature selection [13], neural network and support vector machine [14], aircraft maintenance inspection [15], wireless sensor network [16], and other major engineering fields. However, like most optimization algorithms, crow search algorithm itself also has the defects of slow convergence speed and easy fall into local optimum.

In view of the shortcomings of crow search algorithm, many scholars have put forward their own improvement schemes, which can be roughly divided into two categories. One is to analyze and improve the parameters of the standard crow search algorithm, and the other is to integrate it with other intelligent algorithms to make up for the shortcomings of crow search algorithm. For the first kind of

improvement, Wu et al. proposed a crow search algorithm incorporated in Levy flight (LFCSA) [17] and applied it to update the finite element model. The chaotic sequence was used to initialize the population, so that the particles were evenly distributed in the solution space and the diversity of the population was increased. Therefore, Liu et al. proposed the chaotic binary crow search algorithm (CBCSA) [18] for the discrete space, using it to solve the problem of {0–1} knapsack. By introducing adaptive step size, Mohammdi and Abdi proposed self-adaptive step size crow search algorithm (MCSA) [19] and applied it to nonconvex economic load scheduling. For the second kind of improvement, Xiao et al. [20] combined CSA with sine cosine algorithm to solve pressure vessel design problem. Arora et al. [21] combined the crow search algorithm with the gray Wolf algorithm and used it to solve the problem of feature selection.

All the above improved algorithms have improved the performance of the algorithm to some extent, but some improvements were only optimized for a certain strategy in the crow update mechanism or simply mixed optimization with other optimization algorithms. For example, LFCSA uses Levy flight strategy, CBCSA takes advantage of the particularity of chaos mapping, and MCSA introduces self-adaptive flight step. Although these improved strategies can make the algorithm jump out of local optimum better, they cannot make up for the slow convergence speed. Other improvements are simply hybrid optimizations with other algorithms. This kind of improvement scheme ignores the limitations of the fusion algorithm itself, which also makes the optimization ability of the improved algorithm have some defects. In response to these shortcomings, this article mainly improves the standard crow search algorithm from three aspects: increasing population diversity, self-adaptive awareness probability, and improved cross-pollination mechanism of flower pollination algorithm. The population is initialized by tent chaotic sequence so that the particles are evenly distributed in the search space, and the increase of population diversity enables the algorithm to better jump out of the local optimum and accelerate the convergence speed. The next strategies are self-adaptive awareness probability and improved cross-pollination mechanism. It is beneficial to balance the global search ability and local search ability of the crow search algorithm and avoid the blindness of updating the random search position of crows, thus improving the solution accuracy and convergence speed of the algorithm.

In Section 2 of the paper, we will review the crow search algorithm and the flower pollination algorithm. In Section 3, the improved strategy will be described to produce an improved crow search algorithm. In Section 4, the proposed algorithm will be tested by using 20 well-known benchmark functions and applied to a practical engineering problem. Finally, the conclusion is given in Section 5.

2. Overview of Crow Search Algorithm and Flower Pollination Algorithm

2.1. Crow Search Algorithm. The crow is a very clever bird, which can remember the face of human beings and warn its kind when encountering danger. One of the most obvious characteristics of the cleverness of crows is their ability to hide food and remember the location of the hidden food. At the same time, they will follow each other to get a better source of food, but when the crow finds itself followed by other crows, it will try to change the hidden place of its food to avoid food theft. Based on the living habits of crows, the crow search algorithm has the following principles:

(1) Crows are social animals

(2) Crows can remember the location of the hidden food

(3) Crows will follow each other and steal others' food

(4) Crows do their best to protect their food from being stolen

Based on the four principles, the basic process of CSA is described as follows:

Step 1: initializing the parameters of CSA. These include population size (n), maximum iteration number (Maxsize), flight step size (fl), and awareness probability (AP).

Step 2: initializing the individual crows and memory matrix. n crows are generated in the search space of d – dimension, and each crow $x_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,d})$ represents a feasible solution of a problem. Since the initial population has no experience, it is assumed that the initial memory matrix is the initial position.

Step 3: evaluating the quality of each crow according to the fitness function.

Step 4: generating a new location for each crow in the d – dimensional search space. Assuming that crow i randomly follows a crow (for example, crow j), in order to find the place of the hidden food of crow j, the position update of crow i can be divided into the following two situations:

Case 1: crow j does not find that crow i is following it. In this case, the position update formula of crow i is

$$x^{i,\text{iter}+1} = x^{i,\text{iter}} + r_i + fl^{i.\text{iter}} \times \left(m^{i,\text{iter}} - x^{i,\text{iter}}\right). \quad (1)$$

Case 2: crow j finds that crow i is following it, and crow j will take crow i to a random position.

To sum up, the position update formula of crow i is

$$x^{i,\text{iter}+1} = \begin{cases} x^{i,\text{iter}} + r_i \times fl^{i,\text{iter}} \times \left(m^{i,\text{iter}} - x^{i,\text{iter}}\right), & r_j \geq \text{AP}, \\ a \text{ random position}, & \text{otherwise}. \end{cases} \quad (2)$$

Here, r_i, r_j are random numbers that obey the uniform distribution of [0, 1]. AP represents the perception probability. When the AP is smaller, the probability of occurrence of Case 1 is greater, and the algorithm tends to search locally. When the AP is larger, the probability of finding in Case 2 is greater, and the algorithm tends to search globally. $fl^{i.\text{iter}}$ is the flight step length of crow i. When $fl^{i.\text{iter}} < 1$, the next position of crow i is between $x^{i,\text{iter}}$ and $m^{j,\text{iter}}$, as shown in Figure 1. When $fl^{i.\text{iter}} > 1$, the next position of crow i is outside the line between $x^{i,\text{iter}}$ and $m^{j,\text{iter}}$, as shown in Figure 2. Therefore, fl will affect the search ability of the algorithm. If the value is too large, it tends to search globally, and the algorithm has poor convergence. If the value is too small, it is easy to fall into the local optimum.

Step 5: checking whether the new position of each crow is feasible. If possible, change the crow's position. Otherwise, it is not updated.

Step 6: calculating the fitness value of the new position of each crow.

Step 7: updating the memory matrix of each crow.

$$m^{i,\text{iter}+1} = \begin{cases} x^{i,\text{iter}+1}, & f\left(x^{i.\text{iter}+1}\right) \text{is better than } f\left(m^{i.\text{iter}}\right), \\ x^{i,\text{iter}}, & \text{otherwise.} \end{cases} \tag{3}$$

Step 8: repeating Steps 4–7 until the termination condition is reached.

2.2. Flower Pollination Algorithm. Flower pollination algorithm is a new metaheuristic swarm intelligence optimization algorithm proposed by Yang, a scholar from Cambridge University, UK, in 2012. The algorithm simulates the two processes of cross-pollination and self-pollination of flowering plants, which corresponds to the global search mechanism and local search mechanism in the algorithm. Because the algorithm has few parameters, simple structure, and easy implementation, it has widely attracted the interest of other scholars.

In order to simplify the problem and make the algorithm more efficient, considering that there is only one solution to the optimization problem, Yang assumes that each flowering plant can only produce one flower and each flower can only produce one pollen gamete. Flower pollination process can be summarized as the following four rules:

(1) Biological cross-pollination is considered a global pollination process in which pollen carriers carry pollen on Levy flights

(2) Abiotic self-pollination is regarded as a local pollination process

(3) The reproduction probability is the constancy of flowers, and the value of reproduction probability is proportional to the similarity of two flowers

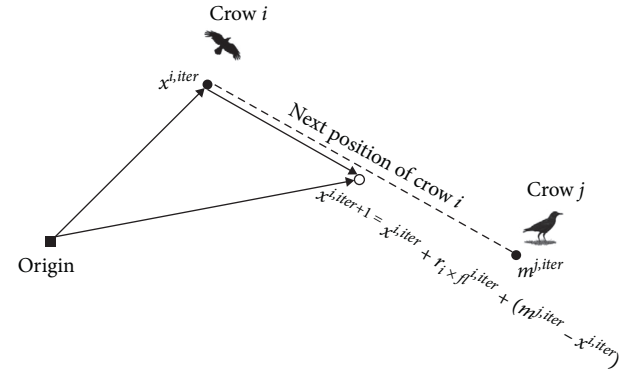

Figure 1: Schematic diagram for updating the position of crow i when $fl < 1$.

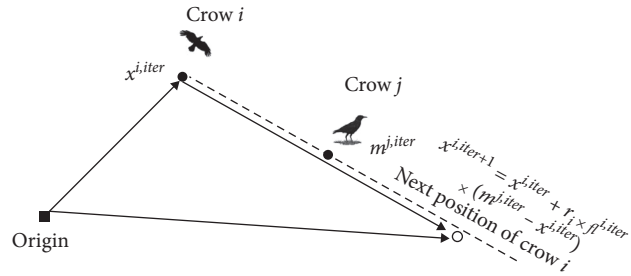

Figure 2: Schematic diagram for updating the position of crow i when $fl > 1$.

(4) The transition probability $p \in [0, 1]$ is used to control the transition between local pollination and global pollination

Through the above rules, the following mathematical model is established.

Definition 1. In the process of global pollination, the formula of pollen position update is

$$x^{i,\text{iter}+1} = x^{i,\text{iter}} + L \times \left(x^{i,\text{iter}} - g\text{best}\right). \tag{4}$$

Here, $x^{i,\text{iter}+1}$ and $x^{i,\text{iter}}$, respectively, represent the solution of the iter + 1 generation and the iter generation; $g$best is the global optimal solution in an iteration process; and L is the step size, which obeys the Levy distribution. The calculation formula of L is as follows:

$$L \sim \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \cdot \frac{1}{S^{1+\lambda}} \left(S \gg S_0 \gg 0\right). \tag{5}$$

Here, $\Gamma(\lambda)$ is the standard gamma function; $\lambda = 1.5$.

Definition 2. The position update formula for the partial pollination stage is as follows:

$$x^{i,\text{iter}+1} = x^{i,\text{iter}} + \varepsilon \times \left(x^{j,\text{iter}} - x^{k,\text{iter}}\right). \tag{6}$$

Here, $x^{j,\text{iter}}$ and $x^{k,\text{iter}}$ randomly select solutions different from $x^{i,\text{iter}}$ in the population, and ε is the probability of

reproduction, which is a random number subject to the uniform distribution of [0, 1].

Definition 3. The transition between global pollination and local pollination is controlled by the value of transition probability $p \in [0, 1]$. A large number of simulation experiments show that the algorithm can obtain the best optimization performance when $p = 0.8$.

3. Crow Search Algorithm Based on Improved Flower Pollination Algorithm (IFCSA)

In order to improve the convergence speed and accuracy of the algorithm, two strategies are used, namely, self-adaptive awareness probability and improved cross-pollination mechanism of flower pollination algorithm. The following is a detailed description of these three strategies.

3.1. Self-Adaptive Awareness Probability. The awareness probability has a greater impact on the performance of the standard crow search algorithm. When the perception probability AP is larger, the individuals in the population are more inclined to search globally, but it is not conducive to improving the convergence accuracy. When the perception probability AP is smaller, the individuals in the population are more inclined to search locally and easily fall into the local optimum. However, when AP is a fixed value, the global search and local search capabilities cannot be balanced. In response to the problem, the inverse incomplete Γ function is introduced to make the awareness probability drop nonlinearly, so as to balance the global search and local search capabilities. The formula is as follows:

$$\mathrm{AP} = 1 \div \left(\left(\mathrm{AP}_1 \times \left(\frac{\mathrm{AP}_2 - \mathrm{AP}_1}{\lambda}\right) \times \Gamma\left(\lambda, \left(1 - \frac{\mathrm{iter}}{\mathrm{Maxiter}}\right)\right)\right) \times 100\right). \tag{7}$$

Here, AP_2 and AP_1 are, respectively, the upper and lower limits of AP; λ is a random variable; and Maxiter is the maximum number of iterations.

Figure 3 shows the decreasing curve of nonlinear awareness probability AP when $\mathrm{AP}_1 = 0.05$, $\mathrm{AP}_2 = 0.25$, and $\lambda = 0.01$. It can be seen from Figure 3 that the AP is large at the beginning of iteration, which makes the algorithm focus on global search. With the iterative process, the AP value decreases gradually, which makes the algorithm tend to search locally, the population concentrate quickly, and the convergence process improve. Nonlinear decreasing awareness probability can better balance global search and local search, thus improving the performance of the algorithm.

3.2. Based on Improved Cross-Pollination. From the analysis of (2), it can be seen that when the leader finds that he is being followed, the position of the individual is randomly generated. Although this location update strategy can prevent the algorithm from falling into the local optimum to a certain extent, it also reduces the convergence speed and accuracy of the algorithm. For this reason, this article solves this problem by introducing improved cross-pollination.

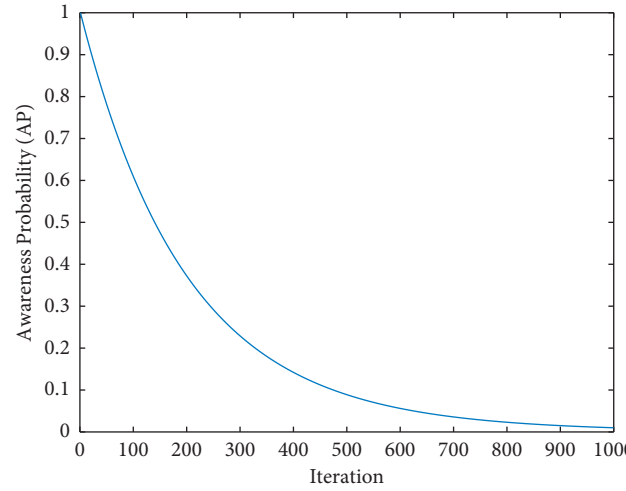

Figure 3: Decreasing curve of nonlinear awareness probability.

Firstly, it is assumed that all crows obtain the global optimal position based on their own experience and memory as thieves. Secondly, the idea of cross-pollination is used to effectively guide the crow individuals to fly close to the optimal individual to reach the optimal value. However, the cross-pollination strategy itself has some limitations. In the later stage of iteration, it will lead to the decrease of population diversity, which will result in falling into local optimum and difficulty in jumping out. Introducing the cross-pollination strategy of Cauchy mutation proposed by Zhang and Gao [22], the formula is as follows:

$$x^{i,\mathrm{iter}+1} = g\mathrm{best} + x^{i.\mathrm{iter}} \times C(0, 1). \tag{8}$$

Here, $g$best is the optimal value and $C(0, 1)$ is Cauchy distribution.

To sum up, the crow position update formula is as follows:

$$x^{i,\mathrm{iter}+1} = \begin{cases} x^{i,\mathrm{iter}} + r_i \times fl^{i.\mathrm{iter}} \times \left(m^{i,\mathrm{iter}} - x^{i,\mathrm{iter}}\right), & r_j \geq \mathrm{AP}, \\ g\mathrm{best} + x^{i.\mathrm{iter}} \times C(0, 1), & \text{otherwise.} \end{cases} \tag{9}$$

3.3. Basic Flow of IFCSA. The pseudocode of the algorithm (IFCSA), which is based on the improved flower pollination algorithm, is shown in Figure 4.

4. Experimental Simulation and Result Analysis

4.1. Experimental Parameter Settings. The IFCSA is validated by comparing it with five famous optimization algorithms, namely, BOA [4], SSA [23], GWO [8], CSA [10], and MISCSA [24]. We used 20 well-known benchmark test functions for validation. The set of benchmark test functions is explained in Tables 1~3 . We used the most common parameter settings that exist in the literature for the seven

Initialize the initial position
for iter = 1 : *Maxiter*
  for i = 1 : *n*
    Calculate the AP
    if $r_j = AP$
      $x^{i,iter+1} = x^{i,iter} + r_i \times fl^{i.iter} \times (m^{i,iter} - x^{i,iter})$
    else
      $x^{i.iter+1} = gbest + x^{i.iter} \times C(0,1)$
    endif
  endfor
  Check boundary
  Evaluate the new position of crows
  Update the memory matrix of each crow
endfor

Figure 4: Pseudocode of the IFCSA.

Table 1: High-dimensional unimodal benchmark functions.

Benchmark function	Dim	Range	$f_{\min}$
$f_1 = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$f_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$f_3 = \max\{\lvert x_i \rvert, 1 \le i \le n\}$	30	$[-100, 100]$	0
$f_4 = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]$	0
$f_5 = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]$	0
$f_6 = 10^6 x_1^2 + \sum_{i=2}^{n} x_i^2$	30	$[-100, 100]$	0
$f_7 = \sum_{i=1}^{n} 10^{6(i-1)/n-1} x_i^2$	30	$[-100, 100]$	0

Table 2: High-dimensional multimodal benchmark functions.

Benchmark function	Dim	Range	$f_{\min}$
$f_8 = \sum_{i=1}^{n} [x_i^2 - 10 \cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]$	0
$f_9 = -20 \exp(-0.2\sqrt{1/n \sum_{i=1}^{n} x_i^2}) - \exp(1/n \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + e$	30	$[-32, 32]$	0
$f_{10} = 1/4000 \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1$	30	$[-600, 600]$	0
$f_{11} = \sum_{i=1}^{n} \lvert x_i \sin(x_i) + 0.1 x_i \rvert$	30	$[-10, 10]$	0
$f_{12} = \sum_{i=1}^{n/4} [(x_{4i-3} + 10 x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4]$	30	$[-4, 5]$	0
$f_{13} = \sin^2(\pi w_1) + \sum_{i=1}^{n} (w_i - 1)^2 [1 + 10 \sin^2(\pi w_i +)] + (w_n - 1)^2 [1 + \sin^2(2\pi w_n)]$, where $w_i = 1 + x_i + 1/4$, for all $i = 1, 2, 1, \ldots, n$	30	$[-10, 10]$	0
$f_{14} = 0.1\{\sin^2(32\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$			
$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a < x_i < a \\ k(-x_i - a)^m, & x_i < a \end{cases}$	30	$[-50, 50]$	0

Table 3: Fixed-dimensional multimodal benchmark functions.

Benchmark function	Dim	Range	$f_{\min}$
$f_{15} = 0.5 + \sin^2(\sqrt{x_1^2 + x_2^2 + 0.5})/(1.0 + 0.0001(x_1^2 + x_2^2))^2$	2	$[-100, 100]$	0
$f_{16} = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2$	2	$[-4.5, 4.5]$	0
$f_{17} = \sum_{i=1}^{11} [a_j - x_1(b_i^2 + b_i x_2)/b_i^2 + b_i x_3 + x_4]^2$	4	$[-5, 5]$	0.00030
$f_{18} = 2x_1^2 - 1.05x_1^4 + x_1^6/6 + x_1 x_2 + x_2^2$	2	$[-5, 5]$	0
$f_{19} = (x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6)^2 + 10(1 - 1/8\pi)\cos x_1 + 10$	2	$[-5, 5]$	0.398
$f_{20} = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2)]$	2	$[-2, 2]$	3

algorithms used in validation. The details about parameter settings are explained in Table 4. For the sake of fairness, we used a fixed population size for all algorithms: $n = 50$ individuals; all algorithms are executed in 30 independent runs. All algorithms are implemented in MATLAB (version R2020a) and executed on HP computer (Windows 10, Intel Core i5-6300HQ, 2.3 GHz, 8 GB RAM). The number of iterations in each run for each algorithm equals 1000. The experimental data obtained after running are shown in Tables 5~7, where Best, Worst, Mean, and Std represent the optimal value, worst value, average value, and Std value obtained by the algorithm independently running 30 times. The algorithm performance is ranked according to the accuracy of the optimal value. The best results are formatted in bold type.

4.2. Improved Strategy Effectiveness Analysis

4.2.1. Effectiveness Analysis of Self-Adaptive Awareness Probability. The awareness probability is one of the decision variables of the standard crow search algorithm, which has a greater impact on the performance of the algorithm. In order to verify whether the adaptive decreasing perceived probability can improve the convergence speed of the algorithm. Six groups of benchmark test functions in Table 1–3 are selected for comparative experiment. Among them, f_1 and f_2 are high-dimensional unimodal functions, f_8 and f_{10} are high-dimensional multimodal functions, and f_{14} and f_{16} are fixed multimodal functions.

As can be seen from Figures 5–8, for f_1, f_2, f_8, f_{10}, and f_{14}, IFCSA with adaptive change awareness probability has better convergence effect under the same number of iterations. It can be seen from Figure 10 that, for function f_{16}, although the convergence effect of IFCSA with adaptively changing awareness probability is worse than IFCSA without adaptively changing perception probability, it can find the theoretical optimal value, which is acceptable.

4.2.2. Effectiveness Analysis of the Cross-Pollination Strategy with Cauchy Mutation. This sections aims to verify whether the improved strategy can avoid the blindness of individual location update and improve the accuracy and convergence ability of the algorithm. At the same time, it aims to further clarify that under the condition that the leader finds himself followed, the performance of the algorithm under the cross-pollination mechanism with Cauchy mutation is better than that under the standard cross-pollination mechanism. The standard crow search algorithm, the crow search algorithm that introduces the standard cross-pollination strategy, and the crow search algorithm that introduces the alienation pollination strategy with Cauchy mutation are used for comparison experiments on 20 benchmark functions. They were run independently 30 times, and the average value was taken as the final criterion.

From the simulation results in Table 8, it can be seen that, for most benchmark functions, the optimization performance of the algorithm that introduces the standard cross-pollination strategy is worse than that of the standard crow search algorithm. In the benchmark function f_{16}, the optimization performance of the algorithm that introduces the standard cross-pollination strategy is the best, but the performance of the algorithm that introduces the cross-pollination strategy with Cauchy mutation is also good, which is acceptable. However, in the 20 benchmark functions, the standard crow search algorithm introduces the cross-pollination strategy with Cauchy mutation, the performance of the algorithm is greatly improved, and the effect is better than the introduction of the standard cross-pollination strategy, except for the benchmark function f_{16}. In short, the crow search algorithm that introduces the cross-pollination strategy with Cauchy mutation has better performance in function optimization.

4.3. Comparative Experiment of Different Intelligent Algorithms

4.3.1. High-Dimensional Unimodal Function Test Results. It can be seen from the experimental results in Table 5 that the optimization performance of IFCSA is better than that of the other algorithms for most high-dimensional unimodal functions. In the test function of f_5, although the performance of IFCSA is inferior to that of MISCSA, the accuracy obtained by them is almost the same. In the other six test functions of $f_1, f_2, f_3, f_4, f_6, f_7$, compared with other algorithms, the optimization accuracy of IFCSA is improved to a certain extent. In addition, the variance of IFCSA is lower than that of the other algorithms, which indicates that IFCSA has good stability. In a word, IFCSA can solve high-dimensional unimodal problems well, which demonstrates its effectiveness and feasibility in solving high-dimensional unimodal problems.

Figures 11–24 are the convergence curve graphs and variance graphs of the algorithms of BOA [4], SSA [23], GWO [8], CSA [10], MISCSA [24], and IFCSA for different test functions. All convergence curves are drawn based on the optimal values. Figures 13 and 21 show that GWO converges faster than IFCSA in the middle of the iteration, but GWO will fall into the local optimum, and IFCSA can jump out of the local optimum to get a better solution. It can be seen from Figures 11, 15, 17, and 23 that the convergence speed and optimization accuracy of IFCSA are better than those of other algorithms. As can be seen from Figures 12, 14, 16, 18, 20, 22, and 24, the performance of IFCSA algorithm is very stable. This shows that IFCSA can better solve high-dimensional unimodal problems compared to other intelligent algorithms and some improved algorithms.

4.3.2. High-Dimensional Multimodal Function Test Results. The experimental data in Table 6 shows that the optimal value and average value of IFCSA for functions f_8~f_{13} rank first in the comparison algorithm, which shows that IFCSA can solve most high-dimensional multimodal functions well. In the test function of f_{14}, the optimal value obtained by IFCSA is slightly smaller than those of GWO and SSA, ranking third in the comparison algorithm. Experimental results show that IFCSA can effectively solve high-dimensional multimodal functions and has strong global search capabilities.

Table 4: Parameter settings.

Algorithm	Parameter settings
BOA	$p = 0.8, c = 0.01, \alpha = 0.1$
FPA	$p = 0.8$
CSA	AP = 0.1, $fl = 2$
MISCSA	AP = 0.1, $fl = 2, \alpha_1 = 0.4, \alpha_2 = 0.7, \delta_1 = 0.4, \delta_2 = 0.001, s = 0.8$
IFCSA	AP = 0.1, $fl = 2$

Table 5: Results of high-dimensional unimodal benchmark functions.

Function	Index	BOA	SSA	GWO	CSA	MISCSA	IFCSA
f_1	Best	$1.53E-14$	$5.66E-09$	$1.40E-68$	$3.41E-03$	$7.59E-33$	**$2.26E-102$**
	Worst	$1.85E-14$	$1.40E-08$	$1.26E-72$	$3.73E-02$	$7.83E-30$	$1.03E-89$
	Mean	$1.69E-14$	$8.84E-09$	$9.11E-70$	$1.36E-02$	$7.98E-31$	$6.03E-91$
	Std	$9.60E-16$	$1.74E-09$	$2.82E-69$	$7.20E-03$	$1.56E-30$	$2.03E-90$
	Rank	4	5	2	6	3	1
f_2	Best	$2.02E-12$	$1.56E-04$	$3.43E-42$	$3.80E-01$	$3.85E-17$	**$1.69E-50$**
	Worst	$1.17E-11$	$3.05E+00$	$4.46E-40$	$2.75E+00$	$2.02E-15$	$4.66E-43$
	Mean	$9.19E-12$	$6.77E-01$	$6.50E-41$	$1.47E+00$	$4.55E-16$	$1.56E-44$
	Std	$2.97E-12$	$7.52E-01$	$9.17E-41$	$6.29E-01$	$4.20E-16$	$8.51E-44$
	Rank	4	5	2	6	3	1
f_3	Best	$1.01E-11$	$8.78E-01$	$1.46E-18$	$1.22E+00$	$5.80E-17$	**$1.57E-50$**
	Worst	$1.27E-11$	$9.46E+00$	$6.98E-17$	$5.48E+00$	$2.28E-15$	$4.79E-44$
	Mean	$1.15E-11$	$4.50E+00$	$1.25E-17$	$3.09E+00$	$5.24E-16$	$2.87E-45$
	Std	$6.68E-13$	$2.38E+00$	$1.57E-17$	$1.02E+00$	$5.20E-16$	$1.06E-44$
	Rank	4	5	2	6	3	1
f_4	Best	$1.48E-14$	$5.93E+00$	$4.57E-26$	$4.07E+02$	$1.06E-30$	**$4.02E-100$**
	Worst	$1.89E-14$	$2.51E+02$	$4.53E-17$	$1.53E+03$	$1.33E-27$	$1.38E-88$
	Mean	$1.70E-14$	$6.46E+01$	$1.55E-18$	$8.79E+02$	$2.52E-28$	$6.89E-90$
	Std	$9.12E-16$	$5.98E+01$	$8.27E-18$	$3.26E+02$	$3.93E-28$	$2.59E-89$
	Rank	4	5	3	6	2	1
f_5	Best	$8.80E-05$	$2.27E-02$	$9.33E-05$	$7.07E-03$	**$6.38E-06$**	$9.98E-06$
	Worst	$1.18E-03$	$1.16E-01$	$1.35E-03$	$3.99E-02$	$3.53E-04$	$3.39E-04$
	Mean	$5.96E-04$	$5.79E-02$	$5.17E-04$	$2.09E-02$	$5.35E-05$	$9.12E-05$
	Std	$3.01E-04$	$2.28E-02$	$3.35E-04$	$7.87E-03$	$6.36E-05$	$7.17E-05$
	Rank	3	6	4	5	1	2
f_6	Best	$1.34E-14$	$3.45E+03$	$7.91E-72$	$6.67E+02$	$5.90E-30$	**$2.17E-97$**
	Worst	$1.81E-14$	$3.09E+04$	$1.18E-68$	$2.62E+03$	$6.76E-27$	$7.54E-88$
	Mean	$1.61E-14$	$8.82E+03$	$1.08E-69$	$1.45E+03$	$9.79E-28$	$2.71E-89$
	Std	$1.13E-15$	$5.69E+03$	$2.77E-69$	$5.15E+02$	$1.53E-27$	$1.38E-88$
	Rank	4	6	2	5	3	1
f_7	Best	$1.49E-14$	$4.97E+04$	$1.44E-70$	$7.57E+03$	$6.03E-29$	**$1.63E-95$**
	Worst	$1.90E-14$	$3.65E+05$	$2.08E-67$	$5.13E+04$	$1.23E-25$	$1.34E-87$
	Mean	$1.74E-14$	$1.57E+05$	$2.30E-68$	$2.19E+04$	$8.07E-27$	$1.26E-88$
	Std	$1.08E-15$	$8.29E+04$	$4.33E-68$	$9.20E+03$	$2.40E-26$	$2.94E-88$
	Rank	4	6	2	5	3	1

Figures 25–31 are the convergence curves of all algorithms for solving high-dimensional multimodal functions independently run 30 times. Figures 25–30 show that IFCSA can find the optimal value faster and better for some high-dimensional multimodal functions. Furthermore, in the functions f_8 and f_{10}, IFCSA can find the theoretical optimal value. Figures 32–38 are the variance graphs of f_8~f_{14}, which shows that IFCSA has good stability. The experimental results show that, compared with some classic intelligent algorithms and some improved algorithms, IFCSA can better solve high-dimensional multimodal functions.

4.3.3. Fixed-Dimensional Multimodal Function Test Results. Table 7 shows the experimental results of different algorithms for fixed multimodal functions. From the experimental results in Table 7, it can be seen that the best values of IFCSA in functions $f_{15}, f_{16}, f_{17}, f_{19}$, and f_{20} are all ranked first, and they are all theoretical best values. It can be also seen from the standard deviation in Table 7 that the performance of IFCSA is relatively stable. Although the IFCSA does not obtain a better optimal value than GWO in the function f_{18}, the optimal value obtained by the IFCSA is of the order of 100^{-178}, which is acceptable.

Table 6: Results of high-dimensional multimodal benchmark functions.

Function	Index	BOA	SSA	GWO	CSA	MISCSA	IFCSA
f_8	Best	**0.00E + 00**	2.49E + 01	**0.00E + 00**	1.09E + 01	**0.00E + 00**	**0.00E + 00**
	Worst	2.10E + 02	7.46E + 01	4.51E + 00	4.08E + 01	0.00E + 00	0.00E + 00
	Mean	1.24E + 01	4.60E + 01	1.50E − 01	2.27E + 01	0.00E + 00	0.00E + 00
	Std	4.76E + 01	1.27E + 01	8.24E − 01	8.63E + 00	0.00E + 00	0.00E + 00
	Rank	1	6	1	5	1	1
f_9	Best	6.05E − 12	1.78E + 00	7.99E − 15	1.91E + 00	**8.88E − 16**	**8.88E − 16**
	Worst	1.32E − 11	3.63E + 00	1.51E − 14	4.70E + 00	8.88E − 16	8.88E − 16
	Mean	1.12E − 11	1.78E + 00	1.31E − 14	3.30E + 00	8.88E − 16	8.88E − 16
	Std	1.39E − 12	8.54E − 01	2.59E − 15	6.50E − 01	0.00E + 00	0.00E + 00
	Rank	4	5	3	6	1	1
f_{10}	Best	**0.00E + 00**	2.27E − 08	**0.00E + 00**	3.11E − 02	**0.00E + 00**	**0.00E + 00**
	Worst	3.66E − 15	4.91E − 02	1.35E − 02	2.58E − 01	0.00E + 00	0.00E + 00
	Mean	9.25E − 16	1.01E − 02	1.10E − 03	1.05E − 01	0.00E + 00	0.00E + 00
	Std	1.05E − 15	1.17E − 02	3.43E − 03	4.18E − 02	0.00E + 00	0.00E + 00
	Rank	1	6	1	5	1	1
f_{11}	Best	1.52E − 15	5.72E − 02	1.57E − 41	6.60E − 03	4.12E − 18	**1.88E − 52**
	Worst	1.24E − 12	4.92E + 00	5.50E − 04	5.17E − 01	5.08E − 17	1.08E − 46
	Mean	4.70E − 14	1.73E + 00	2.21E − 05	1.04E − 01	1.91E − 16	7.20E − 48
	Std	2.25E − 13	1.15E + 00	1.00E − 04	1.45E − 01	3.83E − 17	2.24E − 47
	Rank	4	5	2	6	3	1
f_{12}	Best	9.11E − 15	9.43E − 02	1.45E − 07	2.82E − 01	2.49E − 34	**1.79E − 103**
	Worst	1.65E − 14	1.17E + 00	1.17E − 05	1.94E + 00	1.21E − 30	5.87E − 92
	Mean	1.39E − 14	4.51E − 01	2.80E − 06	6.99E − 01	1.14E − 31	2.86E − 93
	Std	1.70E − 15	2.98E − 01	2.77E − 06	3.74E − 01	2.24E − 31	1.12E − 92
	Rank	3	5	4	6	2	1
f_{13}	Best	1.51E + 00	2.69E − 01	7.22E − 01	2.70E − 01	6.31E − 01	**1.37E − 04**
	Worst	2.52E + 00	1.27E + 01	1.27E + 00	3.21E + 00	2.34E + 00	8.21E − 01
	Mean	2.00E + 00	6.61E + 00	9.89E − 01	1.06E + 00	1.47E + 00	5.99E − 02
	Std	2.47E − 01	3.05E + 00	1.52E − 01	7.62E − 01	3.94E − 01	1.68E − 01
	Rank	6	2	5	3	4	1
f_{14}	Best	1.55E + 00	**2.91E − 10**	1.43E − 05	4.90E + 00	1.40E + 00	1.60E − 03
	Worst	3.00E + 00	1.10E − 01	6.23E − 01	3.95E + 01	4.22E + 00	2.90E + 00
	Mean	2.57E + 00	7.93E − 03	3.16E − 01	2.03E + 01	2.92E + 00	7.19E − 01
	Std	3.50E − 01	2.07E − 02	1.52E − 01	9.93E + 00	7.59E − 01	6.88E − 01
	Rank	5	1	2	6	4	3

Figures 39–44 are the convergence curves of the 6 algorithms for different fixed multimodal functions, and Figures 45–50 are the variance graphs of the 6 algorithms. From Figures 39, 41, 43, and 44, it can be seen that IFCSA can converge to the global optimum faster than the other five algorithms. Figure 40 shows that the optimization performance of IFCSA is weaker than MISCSA, while the overall performance of IFCSA is better than that of the other algorithms. It can be seen from Figure 42 that the optimization performance of IFCSA is better than that of the other algorithms except GWO. In short, IFCSA has certain advantages in optimizing fixed multimodal functions.

4.4. Statistical Validation. According to the paper by Derrac et al. [25], it is not rigorous to only use the average value, standard deviation, optimal value, and worst value obtained after the algorithm is independently run 30 times as the evaluation index of the algorithm performance. Therefore, Wilcoxon rank sum test is carried out on the 20 benchmark test functions in Tables 1~3 for the six algorithms in this paper. Moreover, the p values obtained after rank sum check of the six algorithms are recorded in Table 9. If the algorithm with the best performance is IFCSA, a pair comparison is performed between IFCSA and BOA, IFCSA and SSA, etc. Since the best algorithm cannot be compared with itself, it is marked as "NA," indicating that it is not applicable, which also means that the corresponding algorithm does not have corresponding data to be compared with itself in the rank sum verification process. When the p value is less than 0.05, there is a big difference between the two comparison algorithms. Otherwise, it indicates that there is some similarity between the two comparison algorithms, and the value of p is marked in bold type.

4.5. Engineering Optimization Problem. As can be seen from the previous section, IFCSA has better performance in function optimization than other intelligent algorithms and some improved algorithms. In order to further verify whether IFCSA is effective in practical engineering applications, it is applied to the speed reducer design problem. The problem is one of the most fully researched problems in the optimization test. It represents a simple gearbox design,

Table 7: Results of fixed-dimensional multimodal benchmark functions.

Function	Index	BOA	SSA	GWO	CSA	MISCSA	IFCSA
f_{15}	Best	$3.33E-16$	$8.66E-15$	**$0.00E+00$**	**$0.00E+00$**	**$0.00E+00$**	**$0.00E+00$**
	Worst	$1.02E-02$	$9.72E-03$	$9.72E-03$	$9.72E-03$	$0.00E+00$	$0.00E+00$
	Mean	$8.15E-03$	$1.94E-03$	$2.27E-03$	$7.00E-04$	$0.00E+00$	$0.00E+00$
	Std	$3.71E-03$	$3.95E-03$	$4.18E-03$	$2.46E-03$	$0.00E+00$	$0.00E+00$
	Rank	5	6	1	1	1	1
f_{16}	Best	$1.55E-05$	$4.57E-18$	$1.29E-09$	$3.68E-27$	**$0.00E+00$**	**$0.00E+00$**
	Worst	$3.98E-01$	$5.52E-15$	$2.37E-07$	$7.56E-24$	$1.40E-28$	$0.00E+00$
	Mean	$8.38E-02$	$1.03E-15$	$3.28E-08$	$7.73E-25$	$5.56E-30$	$0.00E+00$
	Std	$1.26E-01$	$1.34E-15$	$4.51E-08$	$1.54E-24$	$2.56E-29$	$0.00E+00$
	Rank	6	4	5	3	1	1
f_{17}	Best	0.00031	0.00031	**0.00030**	**0.00030**	0.00031	**0.00030**
	Worst	0.00038	0.00124	0.02036	0.00122	0.00133	0.00122
	Mean	0.00033	0.00081	0.00381	0.00034	0.00043	0.00037
	Std	0.00002	0.00026	0.00754	0.00017	0.00020	0.00023
	Rank	4	4	1	1	4	1
f_{18}	Best	$2.29E-19$	$7.08E-18$	**$0.00E+00$**	$7.24E-28$	$1.22E-43$	$2.41E-177$
	Worst	$2.61E-17$	$2.86E-15$	$0.00E+00$	$1.04E-24$	$7.79E-41$	$4.06E-166$
	Mean	$6.01E-18$	$7.11E-16$	$0.00E+00$	$1.88E-25$	$1.19E-41$	$1.45E-167$
	Std	$7.05E-18$	$7.43E-16$	$0.00E+00$	$2.45E-25$	$1.70E-41$	$0.00E+00$
	Rank	5	6	1	4	3	2
f_{19}	Best	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**
	Worst	1.212	0.398	0.398	0.398	0.398	0.398
	Mean	0.441	0.398	0.398	0.398	0.398	0.398
	Std	$1.532E-01$	$2.410E-15$	$1.252E-07$	$0.000E+00$	$0.000E+00$	$0.000E+00$
	Rank	1	1	1	1	1	1
f_{20}	Best	3.0002	**3.0000**	**3.0000**	**3.0000**	**3.0000**	**3.0000**
	Worst	3.2338	3.0000	3.0000	3.0000	3.0000	3.0000
	Mean	3.0297	3.0000	3.0000	3.0000	3.0000	3.0000
	Std	$5.10E-02$	$7.55E-14$	$3.91E-06$	$1.47E-15$	$1.65E-15$	$1.37E-15$
	Rank	6	1	1	1	1	1

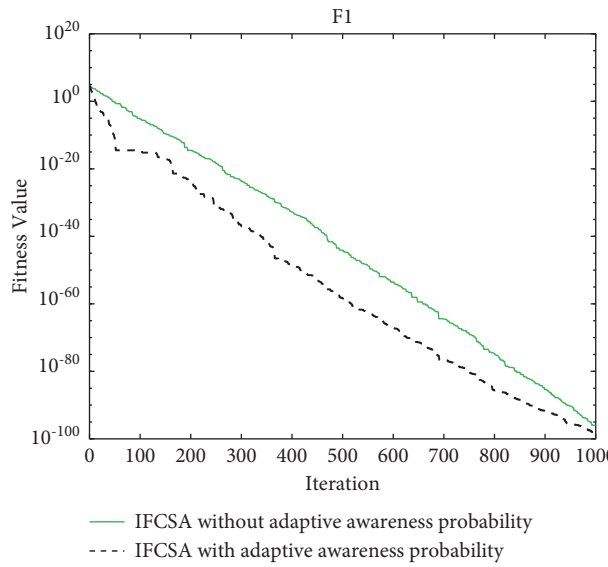

Figure 5: Comparison chart of f_1 convergence curve.

which can be used between the aircraft engine and the propeller to make the components rotate at the most effective speed.

The goal of speed reducer design is to minimize the weight of the gear under bending stress, the lateral deflection of the shaft, and the constraint of the shaft. As shown in

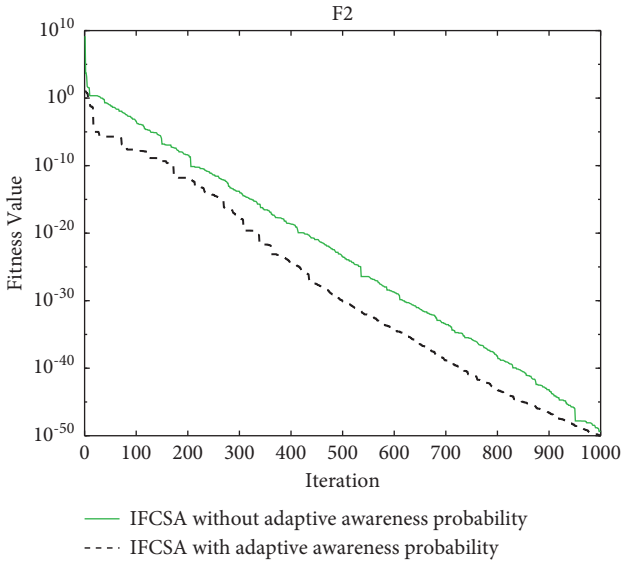

Figure 6: Comparison chart of f_2 convergence curve.

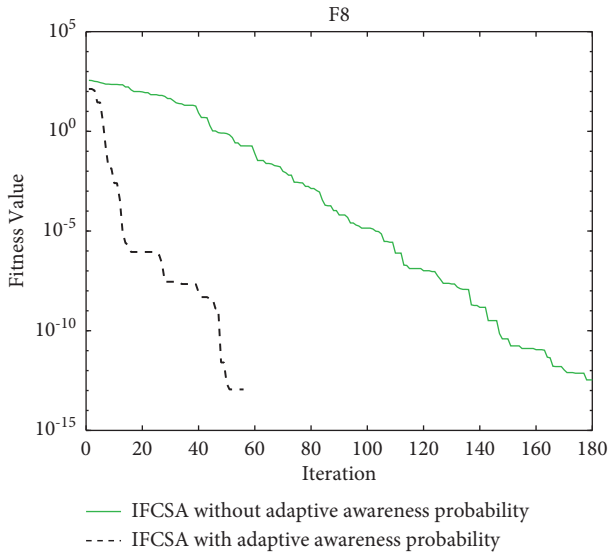

Figure 7: Comparison chart of f_8 convergence curve.

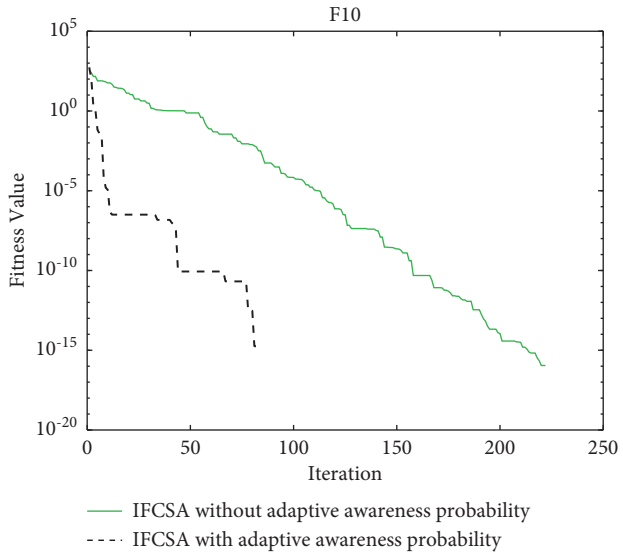

Figure 8: Comparison chart of f_{10} convergence curve.

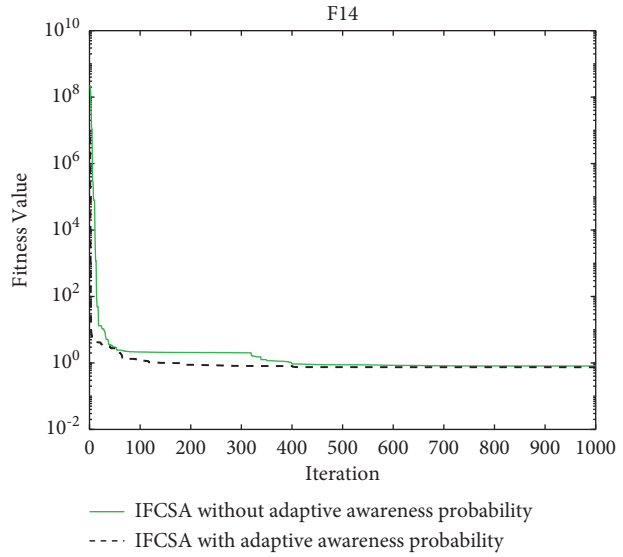

Figure 9: Comparison chart of f_{14} convergence curve.

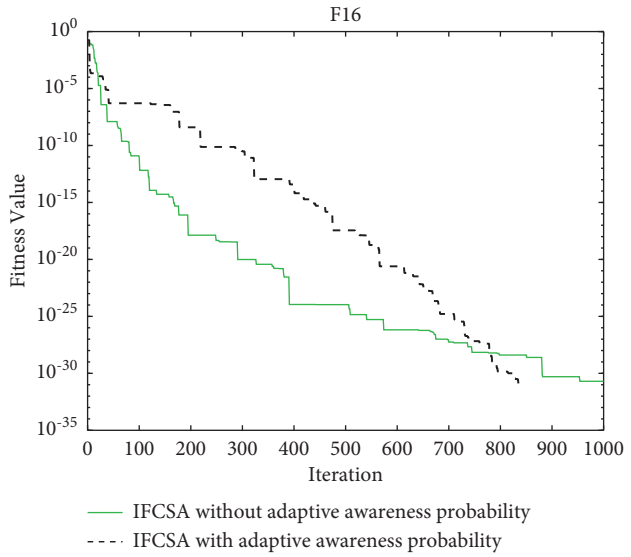

Figure 10: Comparison chart of f_{16} convergence curve.

Table 8: Results of experiments with different strategies.

Function	CSA	CSA + FPA (without tent chaos to the initial position)	CSA + FPA + Cauchy mutation (without tent chaos to the initial position)
f_1	$1.43E-02$	$1.01E+03$	$2.90E-94$
f_2	$1.40E+00$	$9.04E+00$	$5.31E-48$
f_3	$2.62E+00$	$1.37E+01$	$1.04E-47$
f_4	$8.61E+02$	$3.18E+03$	$9.65E-95$
f_5	$2.17E-02$	$1.35E-01$	$3.43E-04$
f_6	$1.47E+03$	$2.41E+03$	$2.30E-92$
f_7	$2.54E+04$	$3.03E+05$	$1.39E-90$
f_8	$2.18E+01$	$7.79E+01$	$0.00E+00$
f_9	$3.29E+00$	$8.99E+00$	$8.88E-16$
f_{10}	$9.90E-02$	$9.84E+00$	$0.00E+00$
f_{11}	$2.15E-01$	$4.60E+00$	$1.37E-49$
f_{12}	$6.01E-01$	$4.55E+01$	$1.21E-95$

Table 8: Continued.

Function	CSA	CSA + FPA (without tent chaos to the initial position)	CSA + FPA + Cauchy mutation (without tent chaos to the initial position)
f_{13}	$7.77E-01$	$3.21E+00$	$1.26E-01$
f_{14}	$2.04E+01$	$1.14E+04$	$1.30E+00$
f_{15}	$9.72E-04$	$3.89E-03$	$0.00E+00$
f_{16}	$4.01E-25$	$0.00E+00$	$1.52E-30$
f_{17}	0.00037	0.00034	0.00030
f_{18}	$2.35E-25$	$2.78E-163$	$2.90E-182$
f_{19}	0.398	0.398	0.398
f_{20}	$3.00E+00$	$3.00E+00$	$3.00E+00$

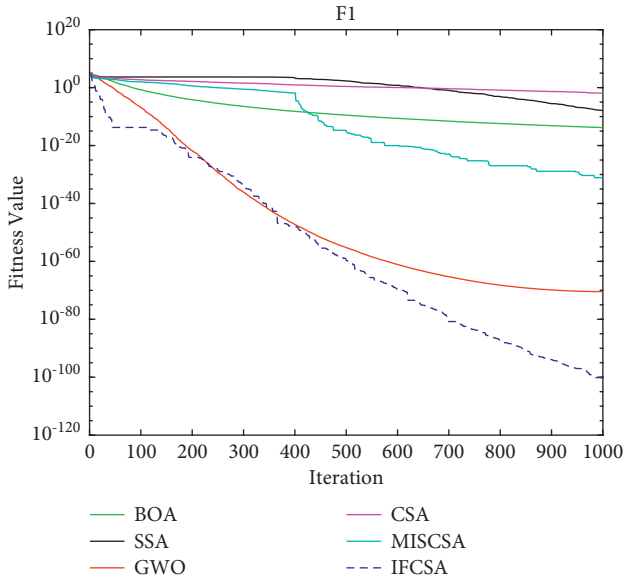

Figure 11: Convergence curve of f_1.

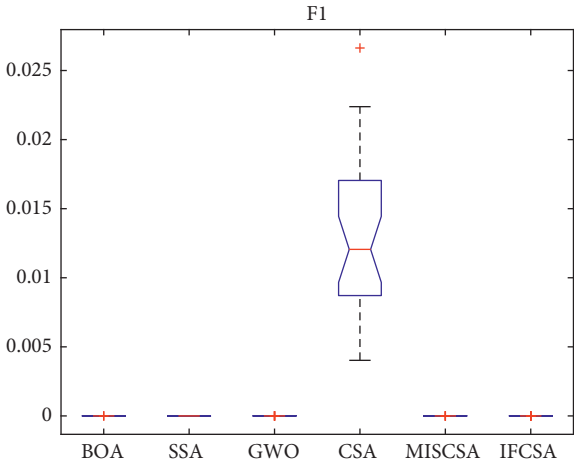

Figure 12: Variance diagram of f_1.

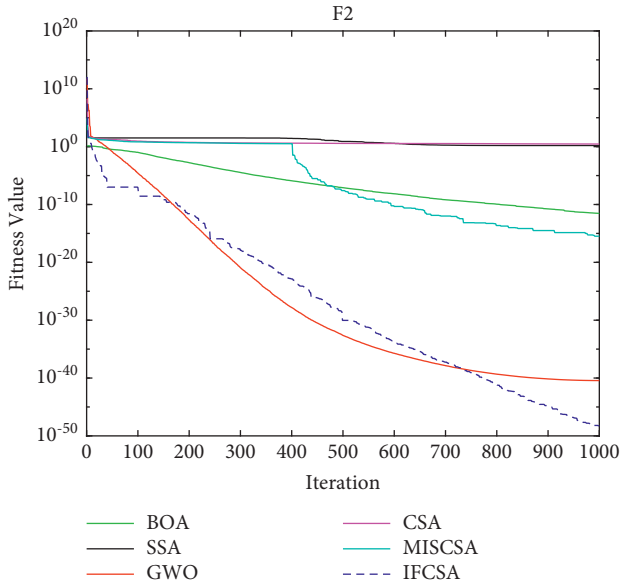

Figure 13: Convergence curve of f_2.

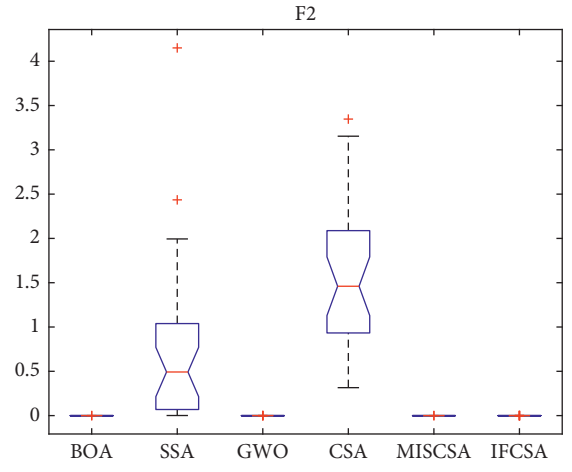

Figure 14: Variance diagram of f_2.

Figure 51, the optimization problem includes seven decision variables, namely, surface width (b or x_1), module of teeth (m or x_2), number of teeth on pinion (z or x_3), length of shaft 1 between bearings (l_1 or x_4), length of shaft 2 between bearings (l_2 or x_5), diameter of shaft 1 (d_1 or x_6), and diameter of shaft 2 (d_2 or x_7).

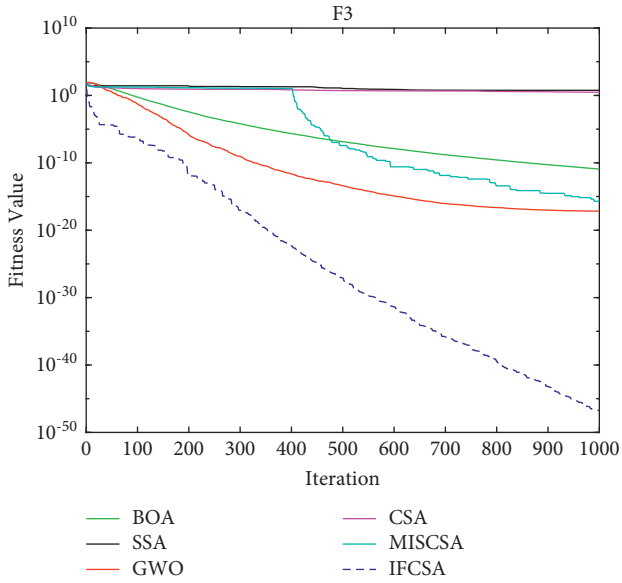

Figure 15: Convergence curve of f_3.

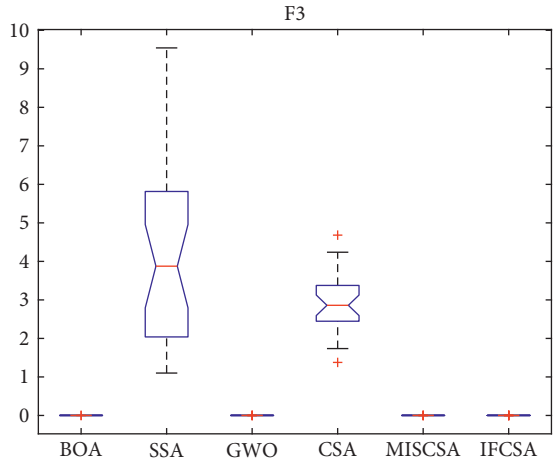

Figure 16: Variance diagram of f_3.

The mathematical model of the problem is as follows:

$$\text{Min. } f(x) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.477\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right),$$

$$\text{S.t.} \quad g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \le 0,$$

$$g_2(x) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0,$$

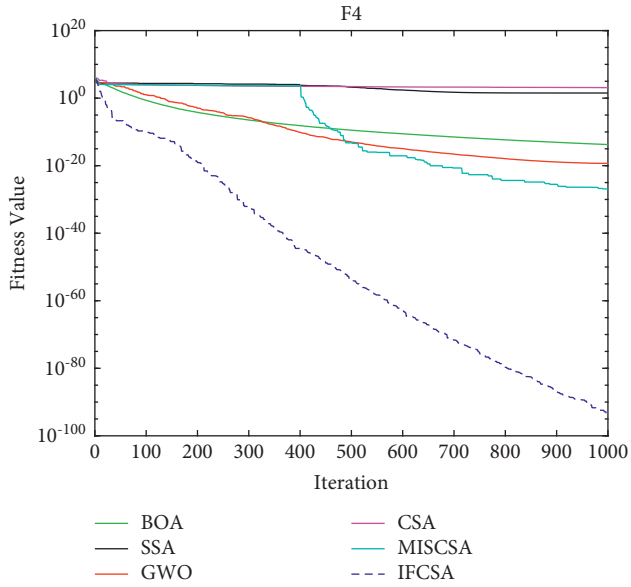

Figure 17: Convergence curve of f_4.

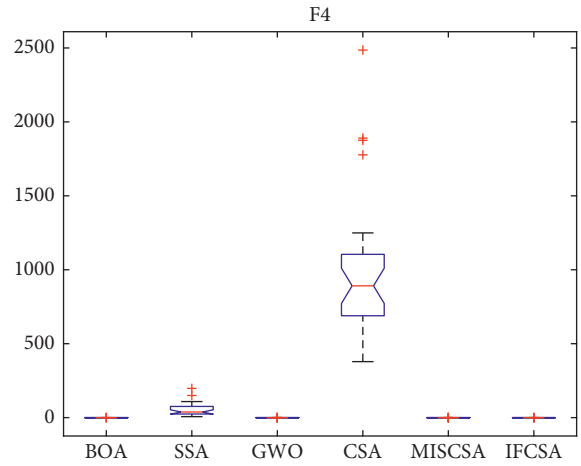

Figure 18: Variance diagram of f_4.

$$g_3(x) = \frac{1.93x_4^3}{x_1x_6^4x_3} - 1 \le 0,$$

$$g_4(x) = \frac{1.93x_5^3}{x_1x_7^4x_3} - 1 \le 0,$$

$$g_5(x) = \frac{\sqrt{(745x_4/x_2x_3)^2 + 1.69 \times 10^6}}{110x_6^3} - 1 \le 0,$$

$$g_6(x) = \frac{\sqrt{(745x_5/x_2x_3)^2 + 157.5 \times 10^6}}{85x_7^3} - 1 \le 0,$$

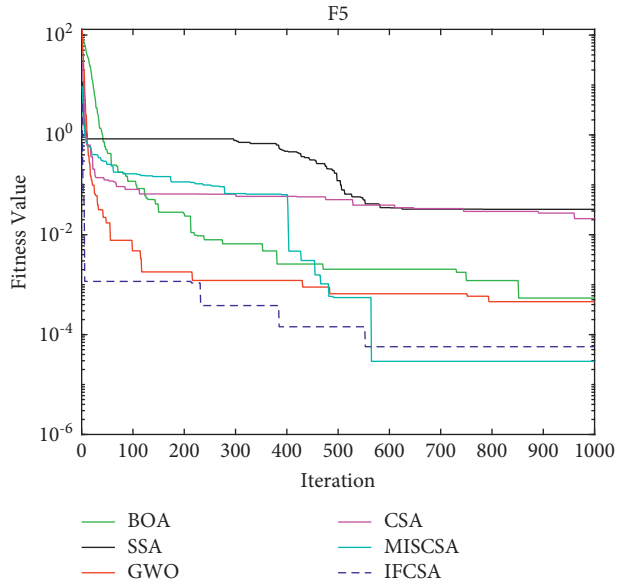

Figure 19: Convergence curve of f_5.

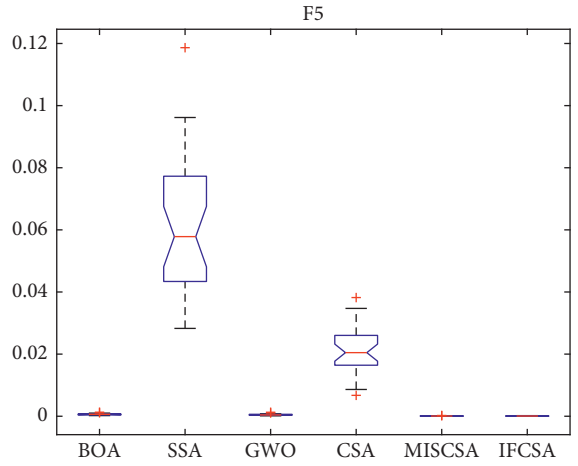

Figure 20: Variance diagram of f_5.

$$
\begin{aligned}
g_7(x) &= \frac{x_2 x_3}{40} - 1 \le 0,\\
g_8(x) &= \frac{5x_2}{x_1} - 1 \le 0,\\
g_9(x) &= \frac{x_1}{12x_2} - 1 \le 0,\\
g_{10}(x) &= \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0,\\
g_{11}(x) &= \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0,
\end{aligned}
\tag{10}
$$

$$2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.8 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5 \le x_7 \le 5.5.$$

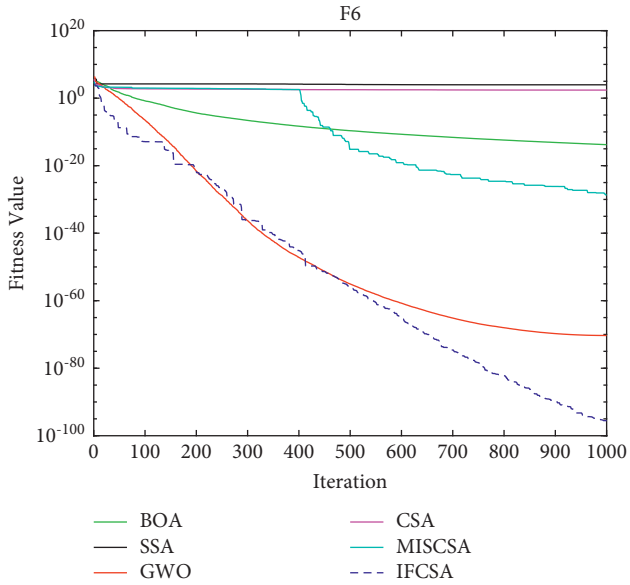

Figure 21: Convergence curve of f_6.

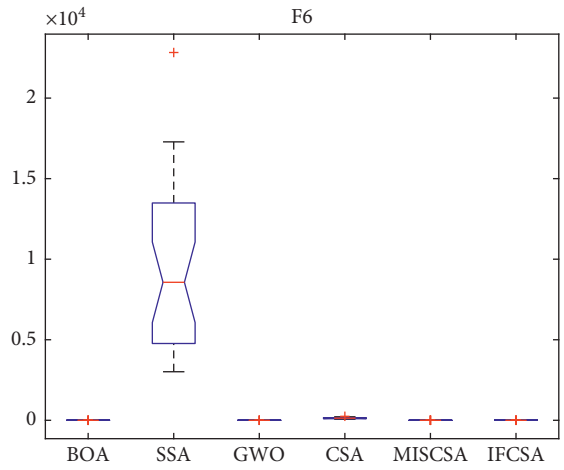

Figure 22: Variance diagram of f_6.

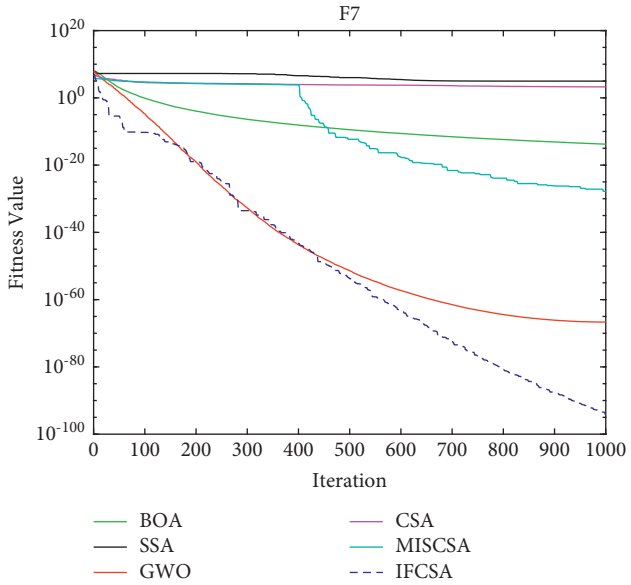

Figure 23: Convergence curve of f_7.

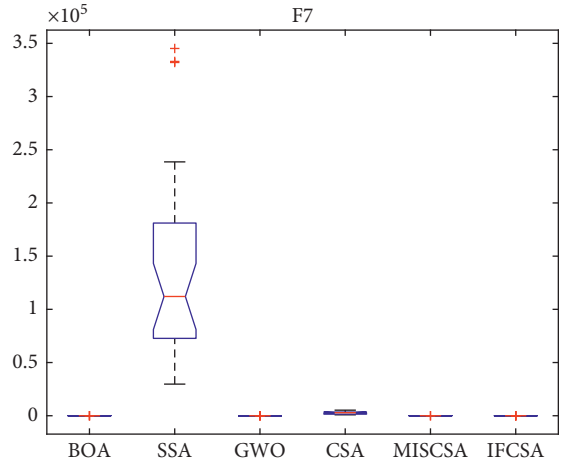

Figure 24: Variance diagram of f_7.

Table 10 shows the optimal values and values of decision variables obtained after 30 independent runs of different algorithms for deceleration design problems. The results are compared with those of CSO [26], Gandomi et al. [27], ABC [28], Akhtar et al. [29], and Montes et al. [30]. According to Table 10, IFCSA finds the optimal cost of 2896.26, which is the least expensive among the comparison algorithms. The solution to find the optimal value is as follows: $x_1 = 3.5, x_2 = 0.7, x_3 = 17, x_4 = 7.3, x_5 = 7.8, x_6 = 2.9, x_7 = 5.286683$. The results show that IFCSA has good performance in dealing with the optimization of speed reducer design problem.

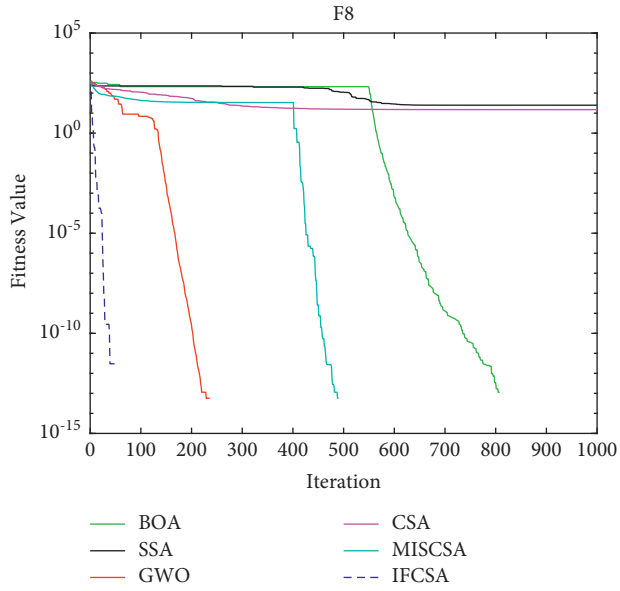

Figure 25: Convergence curve of f_8.

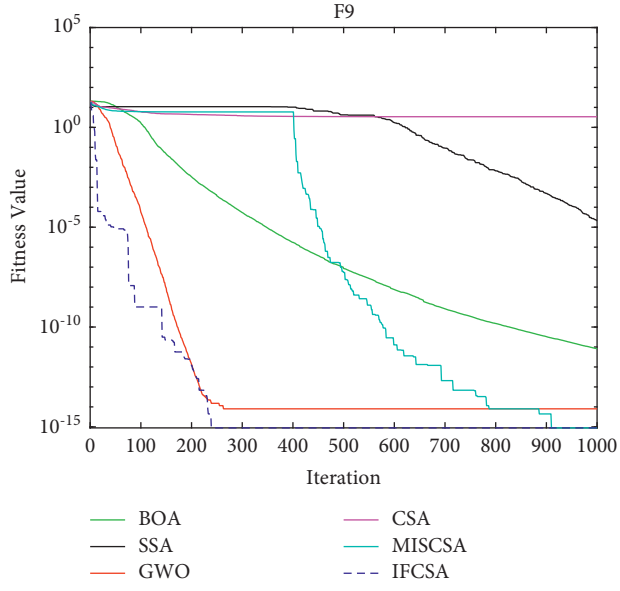

Figure 26: Convergence curve of f_9.

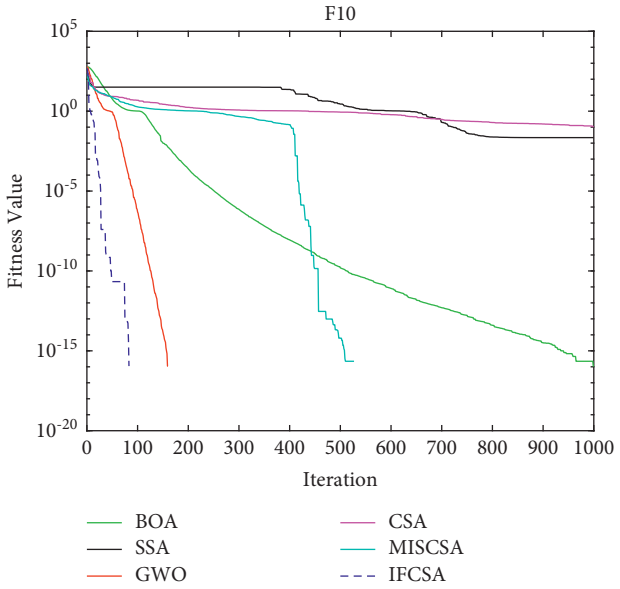

Figure 27: Convergence curve of f_{10}.

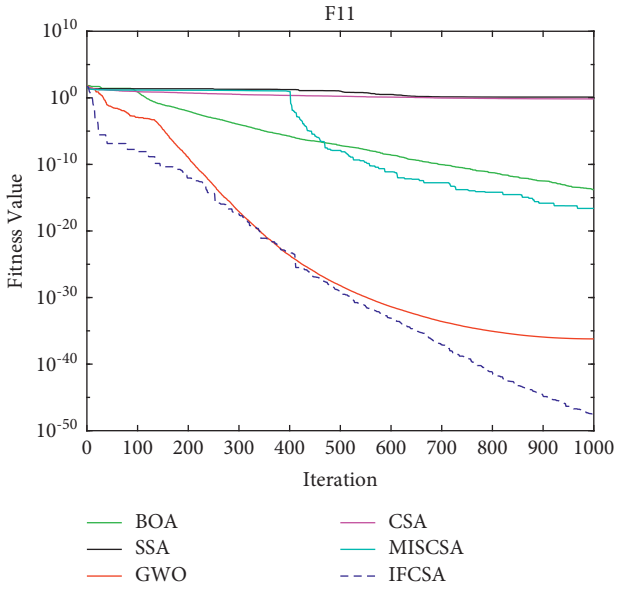

Figure 28: Convergence curve of f_{11}.

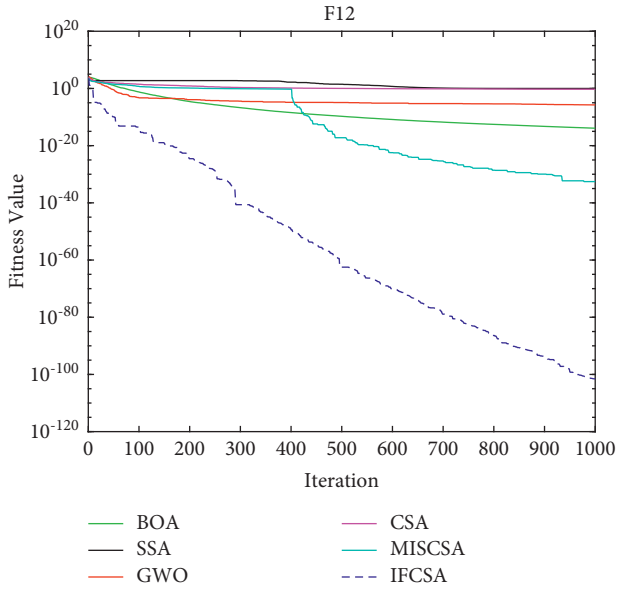

Figure 29: Convergence curve of f_{12}.

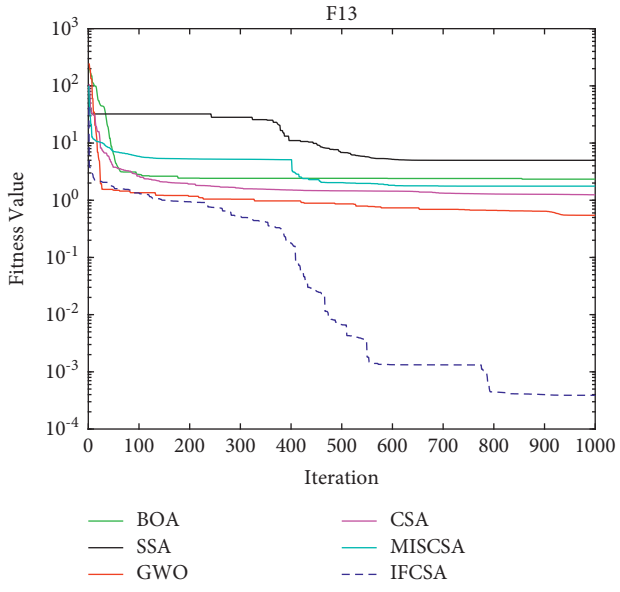

Figure 30: Convergence curve of f_{13}.

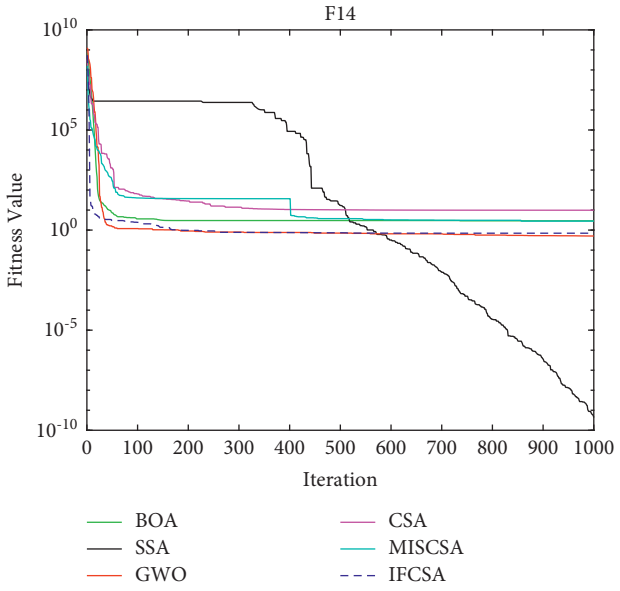

Figure 31: Convergence curve of f_{14}.

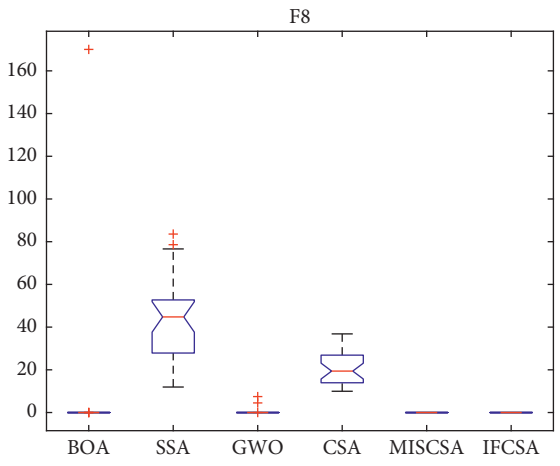

Figure 32: Variance diagram of f_8.

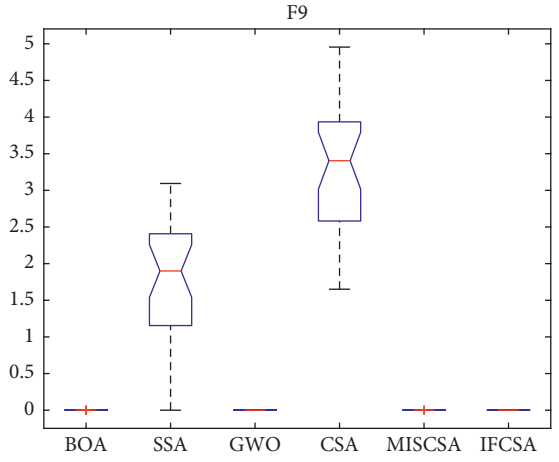

Figure 33: Variance diagram of f_9.

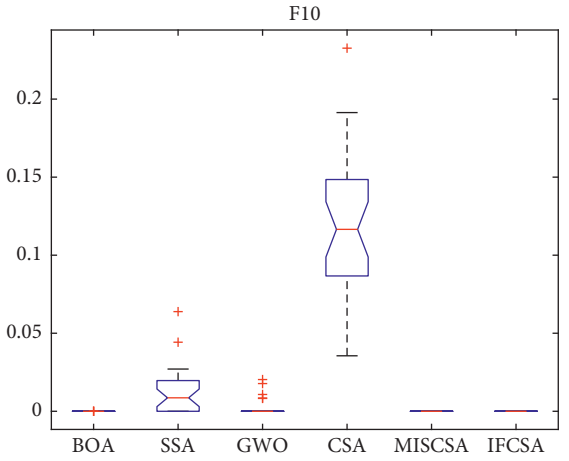

Figure 34: Variance diagram of f_{10}.

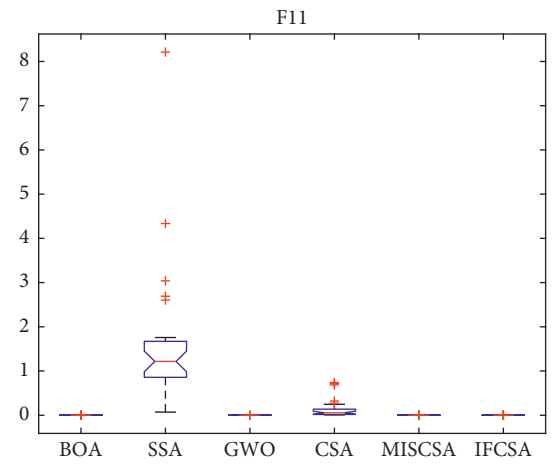

Figure 35: Variance diagram of f_{11}.

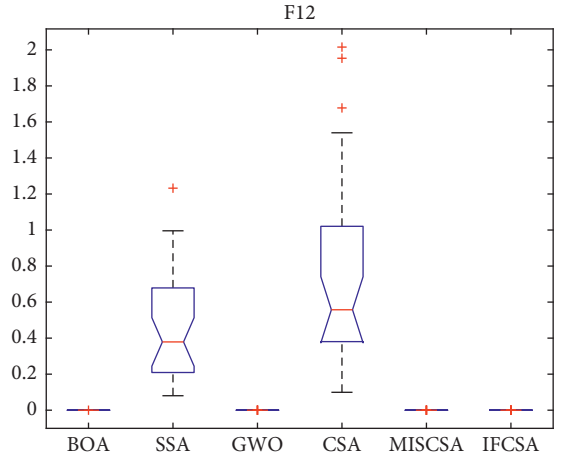

Figure 36: Variance diagram of f_{12}.

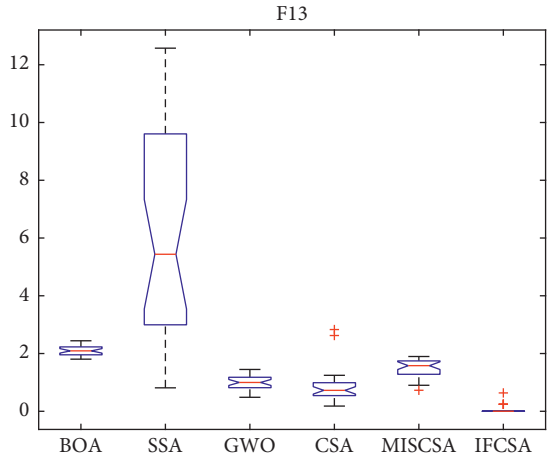

Figure 37: Variance diagram of f_{13}.

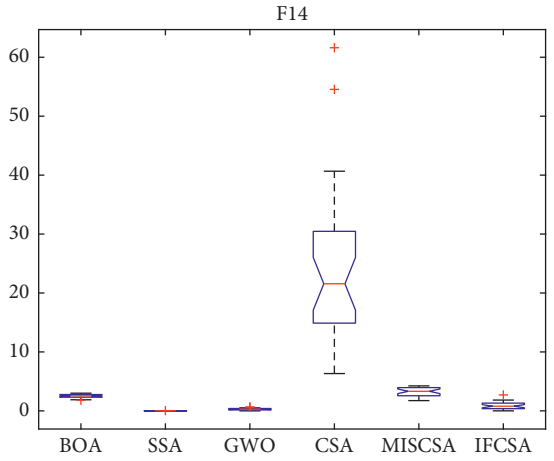

Figure 38: Variance diagram of f_{14}.

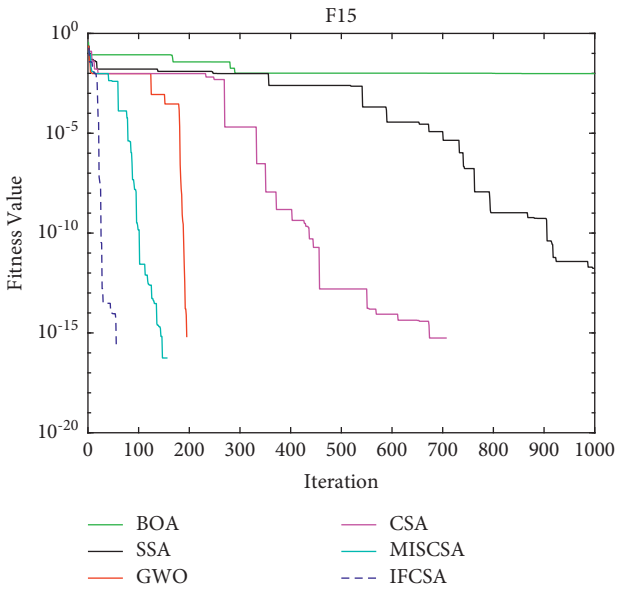

Figure 39: Convergence curve of f_{15}.

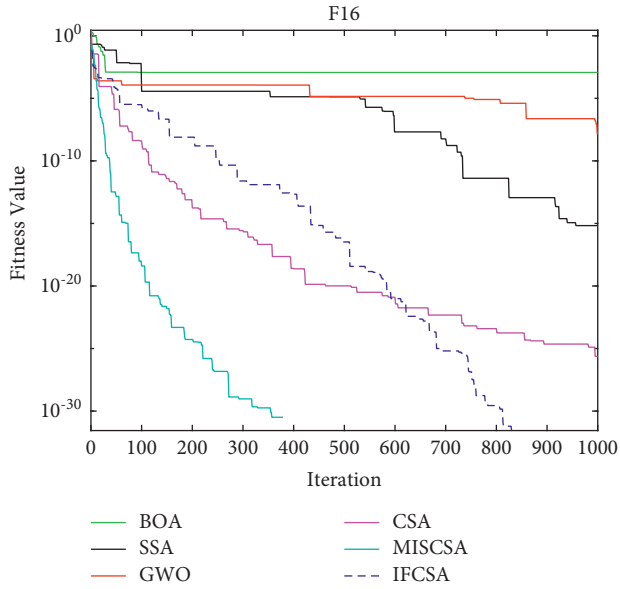

Figure 40: Convergence curve of f_{16}.

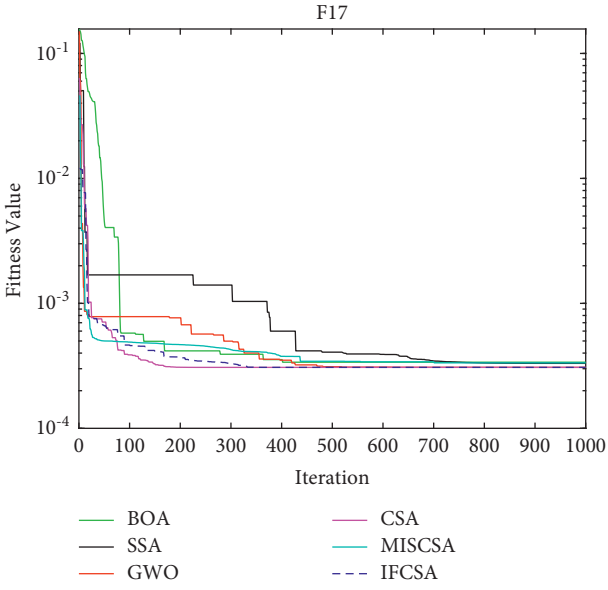

Figure 41: Convergence curve of f_{17}.

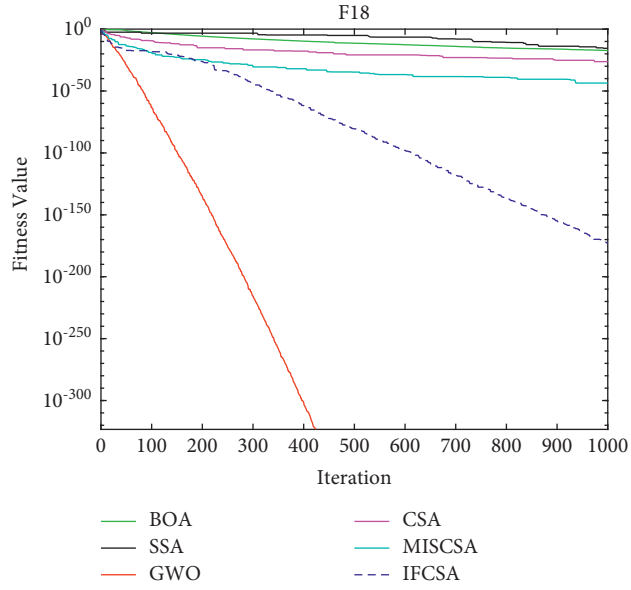

Figure 42: Convergence curve of f_{18}.

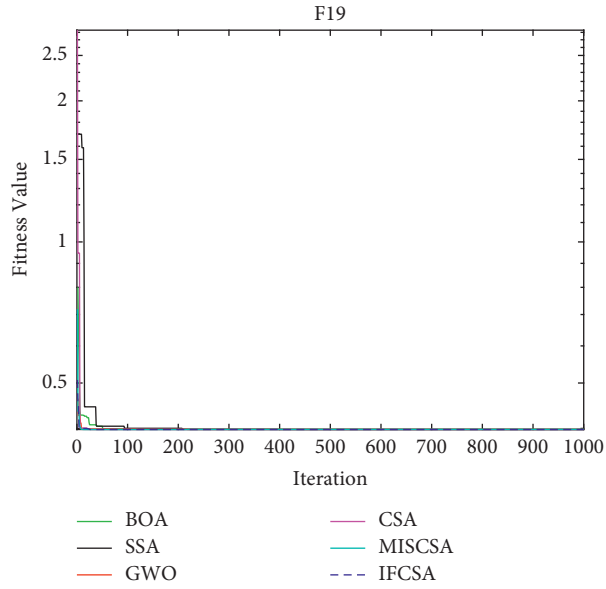

Figure 43: Convergence curve of f_{19}.

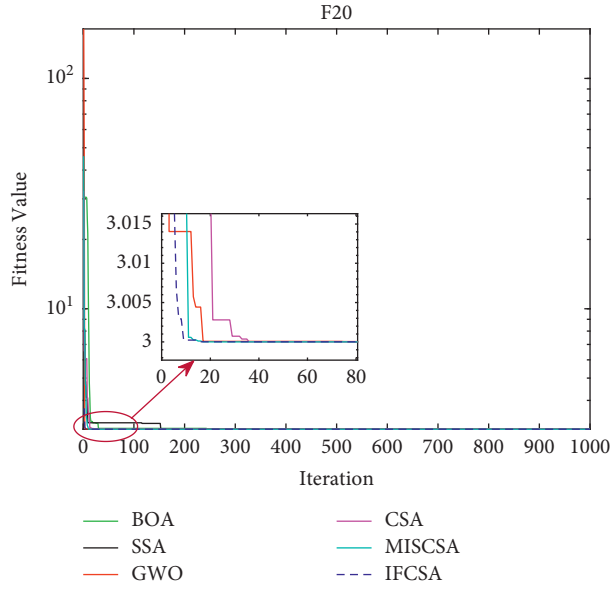

Figure 44: Convergence curve of f_{20}.

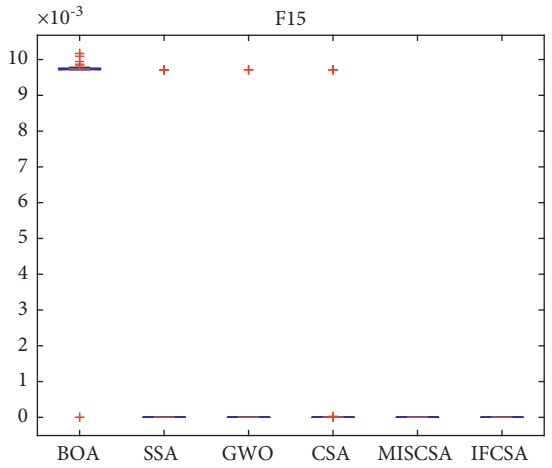

Figure 45: Variance diagram of f_{15}.

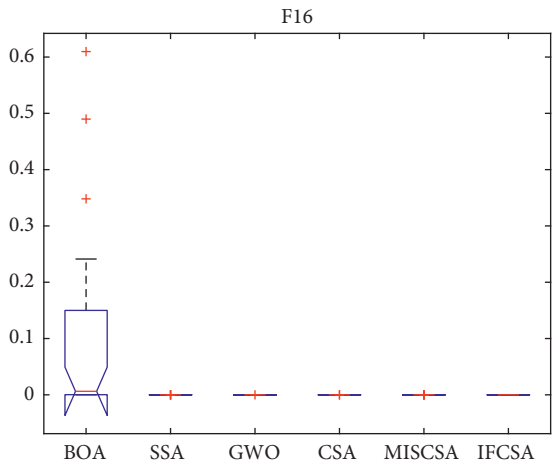

Figure 46: Convergence curve of f_{16}.

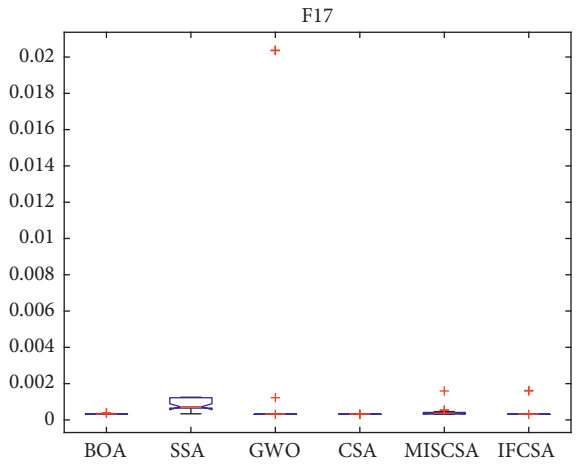

Figure 47: Convergence curve of f_{17}.

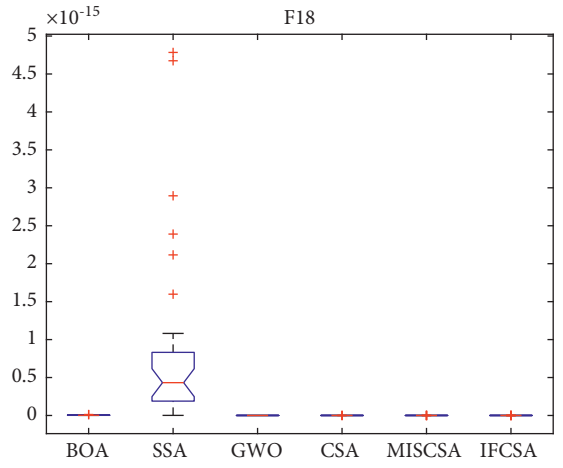

Figure 48: Convergence curve of f_{18}.

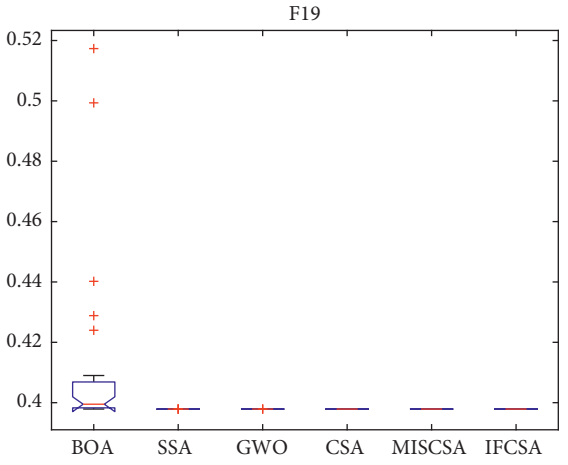

Figure 49: Convergence curve of f_{19}.

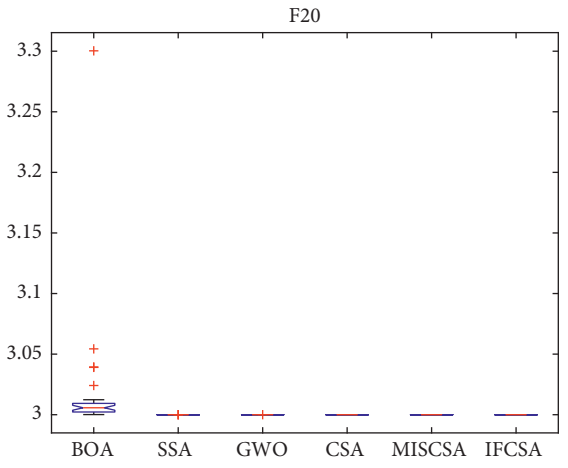

Figure 50: Convergence curve of f_{20}.

Table 9: Wilcoxon rank sum test and p value.

Function	IFCSA/BOA	IFCSA/SSA	IFCSA/GWO	IFCSA/CSA	IFCSA/MISCSA
f_1	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_2	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_3	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_4	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_5	$1.33E-10$	$3.02E-11$	$3.82E-10$	$3.02E-11$	$5.57E-03$
f_6	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_7	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_8	$2.16E-02$	$1.21E-12$	$\mathbf{8.15E-02}$	$1.21E-12$	NA
f_9	$1.21E-12$	$1.21E-12$	$5.47E-13$	$1.21E-12$	NA
f_{10}	$1.93E-09$	$1.21E-12$	$\mathbf{8.15E-02}$	$1.21E-12$	NA
f_{11}	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_{12}	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_{13}	$3.02E-11$	$3.69E-11$	$6.07E-11$	$1.61E-10$	$1.61E-10$
f_{14}	$5.57E-10$	$2.44E-09$	$7.62E-03$	$3.02E-11$	$2.61E-10$
f_{15}	$1.21E-12$	$1.21E-12$	$5.58E-03$	$1.37E-03$	NA
f_{16}	$1.21E-12$	$1.21E-12$	$1.21E-12$	$1.21E-12$	$5.83E-09$
f_{17}	$8.15E-09$	$2.34E-09$	$1.35E-09$	$6.76E-09$	$6.84E-09$
f_{18}	$3.02E-11$	$3.02E-11$	$1.21E-12$	$1.21E-12$	$1.21E-12$
f_{19}	$1.21E-12$	$3.09E-04$	$1.21E-12$	NA	NA
f_{20}	$1.10E-11$	$1.10E-11$	$1.10E-11$	$1.35E-06$	$3.55E-06$

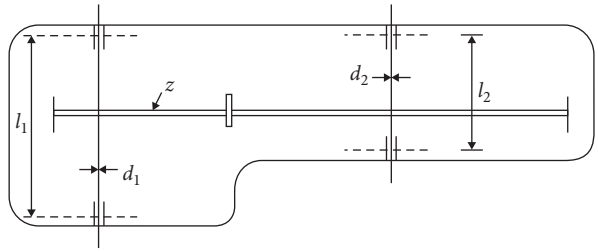

Figure 51: Schematic diagram of speed reducer design.

Table 10: Test results of different algorithms for speed reducer design.

	IFCSA	CSO [26]	Gandomi et al. [27]	ABC [28]	Akhtar et al. [29]	Montes et al. [30]
Best	**2896.26**	2996.60	3000.98	2997.06	3008.08	3025.01
x_1	3.500000	3.500000	3.501500	3.499999	3.506122	3.506163
x_2	0.700000	0.700000	0.700000	0.700000	0.700006	0.700831
x_3	17.00000	17.00000	17.00000	17.00000	17.00000	17.00000
x_4	7.300000	7.308000	7.605000	7.300000	7.549126	7.460181
x_5	7.800000	7.802000	7.818100	7.800000	7.859330	7.962143
x_6	2.900000	3.350000	3.352000	3.350215	3.365576	3.362900
x_7	5.286683	5.287000	5.287500	5.287800	5.289773	5.309000

5. Conclusions

By studying the principle and updating the formula of the standard crow search algorithm, IFCSA is proposed to solve the problem that the algorithm slowly converges and easily falls into local optimum in the later iteration. In this paper, so as to improve the convergence ability of the algorithm, the inverse incomplete gamma function is introduced to make the perceptual probability decrease nonlinearly. Aiming at the blindness of crows' random search for location updating, a cross-pollination strategy with Cauchy mutation was introduced to make crows tend to take the best individual direction, thus obtaining the best value. The experimental results in this paper also show that the optimization performance of IFCSA is better than that of the original algorithm and other intelligent algorithms.

In future work, IFCSA will be used to solve more complex optimization problems, such as multiresource constrained project sequencing, image processing, and UAV path planning. IFCSA will be also used to solve more engineering examples, which is to provide reference value for engineering applications.

Data Availability

The data, models, or code generated or used during the study are available at https://github.com/happyfate/IFCSA.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61662005); Guangxi Natural Science Foundation (no. 2018GXNSFAA294068); Basic Ability Improvement Project for Young and Middle-Aged Teachers in Colleges and Universities in Guangxi (no. 2019KY0195); and Research Project of Guangxi University for Nationalities (no. 2019KJYB006).

References

[1] Y. Tadepalli, M. Kollati, S. Kuraparthi, P. Kora, A. K. Budati, and L. Kala Pampana, "Content-based image retrieval using Gaussian–hermite moments and firefly and grey wolf optimization," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 2, pp. 135–146, 2021.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference On Neural Networks 1995*, vol. 4, pp. 1942–1948, Perth, WA, Australia, December 2002.

[3] X. S. Yang and X. He, "Bat algorithm: literature review and applications," *International Journal of Bio-Inspired Computation*, vol. 5, no. 3, pp. 141–149, 2013.

[4] S. Arora and S. Singh, "Butterfly optimization algorithm: a novel approach for global optimization," *Soft Computing - A Fusion of Foundations, Methodologies and Applications Archive*, vol. 23, no. 3, pp. 715–734, 2019.

[5] X.-S. Yang, "Flower pollination algorithm for global optimization," in *Proceedings of the 11th International Conference on Unconventional Computation and Natural Computation UCNC'12*, pp. 240–249, Orléans, France, September 2012.

[6] H. Duan and P. Qiao, "Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning," *International Journal of Intelligent Computing and Cybernetics*, vol. 7, no. 1, pp. 24–37, 2014.

[7] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, no. 95, pp. 51–67, 2016.

[8] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[9] R. V. Rao and V. Patel, "An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 3, no. 4, pp. 535–560, 2012.

[10] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm," *Computers & Structures*, vol. 169, pp. 1–12, 2016.

[11] G. Y. Abdallh and Z. Y. Algamal, "A QSAR classification model of skin sensitization potential based on improving binary crow search algorithm," *Electronic Journal of Applied Statistical Analysis*, vol. 13, no. 1, pp. 86–95, 2020.

[12] D. Oliva, S. Hinojosa, E. Cuevas, G. Pajares, O. Avalos, and J. Gálvez, "Cross entropy based thresholding for magnetic resonance brain images using crow search algorithm," *Expert Systems with Applications*, vol. 79, pp. 164–180, 2017.

[13] R. C. T. D. Souza, D. S. C. Leandro, A. D. M. Camila, and P. Juliano, "A V-shaped binary crow search algorithm for feature selection," in *Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, Rio de Janeiro, Brazil, July 2018.

[14] R. S. Chithra and P. Jagatheeswari, "Fractional crow search-based support vector neural network for patient classification and severity analysis of tuberculosis," *IET Image Processing*, vol. 13, no. 1, pp. 108–117, 2019.

[15] N. Siswanto, A. N. Adianto, H. A. Prawira, and A. Rusdiansyah, "A crow search algorithm for aircraft maintenance check problem and continuous airworthiness maintenance program," *Jurnal Sistem Dan Manajemen Industri*, vol. 3, no. 2, pp. 115–123, 2019.

[16] P. K. Kodoth and G. Edachana, "An energy efficient data gathering scheme for wireless sensor networks using hybrid crow search algorithm," *IET Communications*, vol. 15, no. 7, pp. 906–916, 2021.

[17] H. Wu, P. Wu, K. Xu, and F. Li, "Finite element model updating using crow search algorithm with Levy flight," *International Journal for Numerical Methods in Engineering*, vol. 121, no. 13, pp. 2916–2928, 2020.

[18] X. Liu, Y. He, C. Wu, and L. Li, "Chaotic binary crow algorithm for 0-1 knapsack problem," *Computer Engineering and Applications*, vol. 54, no. 10, pp. 173–179, 2018.

[19] F. Mohammadi and H. Abdi, "A modified crow search algorithm (MCSA) for solving economic load dispatch problem," *Applied Soft Computing*, vol. 71, pp. 51–65, 2018.

[20] Z. Xiao, S. Liu, F. Han, and J. Yu, "Crow search algorithm based on directing of sine cosine algorithm," *Computer Engineering and Applications*, vol. 55, no. 21, pp. 52–59, 2019.

[21] S. Arora, H. Singh, M. Sharma, S. Sharma, and P. Anand, "A new hybrid algorithm based on grey Wolf optimization and crow search algorithm for unconstrained function optimization and feature selection," *IEEE Access*, vol. 7, pp. 26343–26361, 2019.

[22] S. Zhang and D. Gao, "Flower pollination algorithm based on dynamic adjustment and collaborative search," *Computer Engineering and Applications*, vol. 55, no. 24, pp. 46–53, 2019.

[23] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm," *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017.

[24] Z. Xin, D. Zhang, Z. Chen, H. Zhang, and W. Yan, "Shared crow algorithm using multi-segment perturbation," *Computer Engineering and Applications*, vol. 56, no. 2, pp. 55–61, 2020.

[25] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[26] X. Meng, Y. Liu, X. Gao, and H. Zhang, "A new bio-inspired algorithm: chicken swarm optimization," in *Proceedings of the International Conference in Swarm Intelligence*, pp. 86–94, Hefei, China, October 2014.

[27] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering With Computers*, vol. 29, no. 1, pp. 17–35, 2013.

[28] B. Akay and D. Karaboga, "Artificial bee colony algorithm for large scale problems and engineering design optimization," *Journal of Intelligent Manufacturing*, vol. 23, no. 4, pp. 1001–1014, 2012.

[29] S. Akhtar, K. Tai, and T. Ray, "A SOCIO-behavioural simulation model for engineering design optimization," *Engineering Optimization*, vol. 34, no. 4, pp. 341–354, 2002.

[30] E. Mezura-Montes, C. A. Coello Coello, and R. Landa-Becerra, "Engineering optimization using simple evolutionary algorithm," in *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*, pp. 149–156, Sacramento, CA, USA, November 2003.