Hindawi
Complexity
Volume 2020, Article ID 6084917, 36 pages
https://doi.org/10.1155/2020/6084917

Research Article

An Adaptive Chaotic Sine Cosine Algorithm for Constrained and Unconstrained Optimization

Yetao Ji,[1] **Jiaze Tu**,[1] **Hanfeng Zhou**,[1] **Wenyong Gui**,[1] **Guoxi Liang**,[2] **Huiling Chen**,[1] **and Mingjing Wang**[3]

[1] *College of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou, Zhejiang 325035, China*
[2] *Department of Information Technology, Wenzhou Polytechnic, Wenzhou 325035, China*
[3] *Institute of Research and Development, Duy Tan University, Da Nang 550000, Vietnam*

Correspondence should be addressed to Guoxi Liang; guoxiliang2017@gmail.com and Huiling Chen; chenhuiling.jlu@gmail.com

Received 26 April 2020; Revised 3 July 2020; Accepted 18 September 2020; Published 31 October 2020

Academic Editor: Zong Woo Geem

Sine cosine algorithm (SCA) is a new meta-heuristic approach suggested in recent years, which repeats some random steps by choosing the sine or cosine functions to find the global optimum. SCA has shown strong patterns of randomness in its searching styles. At the later stage of the algorithm, the drop of diversity of the population leads to locally oriented optimization and lazy convergence when dealing with complex problems. Therefore, this paper proposes an enriched SCA (ASCA) based on the adaptive parameters and chaotic exploitative strategy to alleviate these shortcomings. Two mechanisms are introduced into the original SCA. First, an adaptive transformation parameter is proposed to make transformation more flexible between global search and local exploitation. Then, the chaotic local search is added to augment the local searching patterns of the algorithm. The effectiveness of the ASCA is validated on a set of benchmark functions, including unimodal, multimodal, and composition functions by comparing it with several well-known and advanced meta-heuristics. Simulation results have demonstrated the significant superiority of the ASCA over other peers. Moreover, three engineering design cases are employed to study the advantage of ASCA when solving constrained optimization tasks. The experimental results have shown that the improvement of ASCA is beneficial and performs better than other methods in solving these types of problems.

1. Introduction

1.1. Motivation and Incitement. In the field of engineering, optimization algorithms have always been considered as some core stochastic methods [1], which often perform better than the traditional descent-based approach [2]. Many researchers have worked to develop new intelligent optimization algorithms to solve complex engineering cases [3–16]. Swarm intelligence algorithms (SI) are widely used, such as gray wolf algorithm (GWO) [17], whale optimizer (WOA) [18], artificial bee colony algorithm (ABC) [19], particle swarm optimizer (PSO) [20], moth-flame optimizer (MFO) [21], genetic algorithm (GA) [22], slime mould algorithm (SMA) [23], and Harris hawks optimizer (HHO) [24]. These algorithms are inspired by the natural biological evolution of foraging behavior [25–29]. Different from the algorithms above, the sine cosine algorithm (SCA) [30] is a new group of intelligence optimization algorithms proposed by Mirjalili in 2016. Compared with other SI, SCA is simple in structure. SCA does not have a complex mathematical model like many other algorithms, and its updating formulas rely on sine and cosine. The SCA also involves fewer parameters. There are only four parameters included in the update formula. Despite this, SCA still has a good ability. This makes it valued and applied in many fields. For example, Yang et al. [31] proposed a multigroup multistrategy SCA (MMSCA) for solving the capacitated vehicle routing problem in transportation. Of course, SCA also has deficiencies, and it will inevitably fall into local optimality, but its simple structure makes the improvement of the algorithm have great potential. We hope to make up for the shortcomings of SCA by adding mechanisms, so that it can have better effect.

1.2. Literature Review. There are still many problems that need to be solved with the existing intelligent algorithms [32–38]. For example, when the convergence speed of an algorithm is slow, it is easy to fall into local optima [4, 39–45]. In order to solve these problems, many scholars have carried out related researches [41, 46–53]. Nenavath and Jatoth [54] proposed a new optimization algorithm called hybrid SCA with a differential evolution algorithm (DE), which has a more exceptional ability to jump out from local optimization compared with the standard SCA and DE, and then they applied it to object tracking. Rizk-Allah [55] presented the multiorthogonal sine cosine algorithm (MOSCA) by combining the SCA with a multiorthogonal search strategy (MOSS). MOSS can help SCA with in-depth exploration and search, eliminating the disadvantages of unbalanced exploitation and easy to fall into local optimal value. Abd Elaziz et al. [56] used opposition-based learning (OBL) to optimize SCA, which enabled SCA to analyze the solution space better and increase the accuracy of the optimization process. Zhang et al. [57] proposed an elite opposition-based algorithm named the sine cosine WWO algorithm (SCWWO). The waveform of the water wave is very similar to the sine and cosine curves. SCA has a robust global search capability. The combination of SCA and WWO improved the exploitation and exploration capabilities of the algorithms.

Rizk-Allah [58] improved the SCA via orthogonal parallel information (SCA-OPI) in which multiple-orthogonal parallel information is used to maintain the diversity and enhance the exploration search, and an opposition direction strategy based on experience preserves the exploration capability. Qu et al. [59] presented a modified SCA based on neighborhood search and greedy Levy mutation. First, to balance the global search capability and local exploitation capability of the algorithm, the method of exponential decreasing conversion parameter and linear decreasing inertia weight is adopted. Second, to escape from local optimization easily, they replaced the optimal individuals with the random ones near them. Finally, the greedy Levy mutation strategy was used for the optimal individuals so that the local exploitation capability could be improved. Tawhid and Savsani [60] proposed an efficient multi-objective SCA (MO-SCA). The algorithm employed the elitist nondominated sorting and crowding distance method to keep the diversity of optimal solution sets. Sindhu et al. [61] proposed an SCA for feature selection using an elitism strategy and a new updating mechanism. It was used for feature selection to find out the optimal feature to improve the classification accuracy. Zamli et al. [62] proposed a new hybrid Q learning strategy called Q learning sine cosine algorithm (QLSCA). The QLSCA dynamically identifies the best practices at run time with Q learning algorithms (based on punishment and reward mechanisms) and adds Levy flight (LF) motion and crossover to facilitate the algorithm jumping out of local optimum and increase solution diversity. Turgut [63] proposed a hybrid optimization algorithm by combining the advantages of the backtracking search algorithm (BSA) and SCA, which achieved the optimal design of a shell and tube evaporator. Chegini et al. [64] combined the PSO algorithm with the SCA position update equation and the LF method to propose a new hybrid algorithm PSOSCALF. LF is a random walk that generates the search step by the Levy distribution. As the number of jumps increases, more effective searches were performed in the search space, which improved the search capability of the original SCA and avoided trapping into the local minimum. Issa et al. [65] combined the PSO with the SCA, but the difference is that Issa et al. improved the SCA by introducing tuning parameters in the SCA, which was named ASCA-PSO. Another improved SCA called M-SCA was proposed by Gupta and Deep [66], and they used the relative numbers based on the perturbation rate to generate the opposite population so that the population diversity of the original algorithm was increased. Abdel-Fatah et al. [67] proposed a modified version of SCA called MSCA. The MSCA is dependent upon LF distribution and adaptive operators for enhancing the searching capabilities of the basic SCA. Huang et al. [68] proposed a new, improved SCA called CLSCA. The improved algorithm combines a chaotic local search (CLS) mechanism and an LF operator. Abdo et al. [69] proposed a newly developed version of GWO called DGWO. The DGWO applied a random mutation to avoid the algorithm falling into local optima. It also improved the exploitation process by updating the populations in a spiral path. Zhang et al. [44] proposed a new version of FOA with a Gaussian mutation operator and the CLS strategy called MCFOA. It avoids premature convergence by using a Gaussian mutation operator and then used CLS to improve local search capability. Gupta et al. [70] proposed a modified version of the SCA called MSCA. MSCA introduced a nonlinear transition rule and the leading guidance based on the elite candidate solution. Besides, MSCA used the mutation operator to generate a new position when the newly added mechanism cannot provide a better solution. Xu et al. [43] proposed an improved MFO called CLSGMFO. Gaussian variation is used to improve the population diversity of MFO. Then, CLS is applied to the flame updating process of MFO to make better use of the solution locality. Taher et al. [71] proposed a modified grasshopper optimization algorithm (MGOA) to solve the optimal power flow (OPF) problem. MGOA is modifying the mutation process in the traditional GOA to avoid falling into local optima. Mohamed et al. [72] used GWO to solve the distribution problem of PV and DTSTACOM and proved its effectiveness by simulation experiments. Anguluri [73] proposed a bee-inspired algorithm based on observing the foraging and mating behavior of honey bees. Zhao et al. [74] combined the dynamic multiswarm particle swarm optimization (DMS-PSO) algorithm with the subregion coordinated search algorithm (SHS) to get an improved algorithm called DMS-PSO-SHS. Long et al. [75] proposed an improved SCA solution to the high-dimensional global optimization problem. They used a modified position-updating equation based on inertia weight and a nonlinear conversion parameter strategy based on the Gaussian function to improve the performance of the SCA.

1.3. Contribution and Paper Organization. The main contribution of this paper is as follows: a new adaptive parameter r_1 updating formula is proposed. SCA involves less parameters than other methods. Adjusting these parameters has an opportunity to improve the performance of the algorithm. So that, the r_1 update formula is replaced, where r_1 is a conversion parameter which affects the performance of the algorithm directly. An improved SCA called ASCA is proposed. The disadvantages of the SCA algorithm are apparent. The local convergence speed of the algorithm is weak in the later stage. To expand the ability of local exploitation, chaos-based local search is added based on the changing parameter r_1. Then, compared with the other competitive rivals, the superiority of ASCA is proved. The impact of both mechanisms on SCA was experimentally tested on a comprehensive set of benchmarks. ASCA has better results in engineering problems than other peers.

The structure of this paper is presented as follows. Section 2 gives an overview of SCA. Section 3 describes ASCA in detail. Sections 4.1 to 4.5 show the experimental results. Both mechanisms are tested in Section 4.6. In Section 4.7, the experimental results are statistically analyzed. Section 4.8 presents the application of ASCA to engineering cases. The research results are summarized in Section 5.

2. Sine Cosine Algorithm (SCA)

SCA starts from a set of randomly generated solutions. Through formula updating, the algorithm achieves the global optimal solution continuously after the global exploration stage and the local exploitation stage.

First, the algorithm initializes the position of the solution, calculates the fitness, and records the position of the optimal solution. Then, the position is updated according to

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times \sin(r_2) \times \left| r_3 \times P_i^t - X_i^t \right|, & r_4 < 0.5, \\ X_i^t + r_1 \times \cos(r_2) \times \left| r_3 \times P_i^t - X_i^t \right|, & r_4 \geq 0.5, \end{cases} \tag{1}$$

where t represents the current iteration number, X_i^t represents the components in dimension i at iteration t, P_i^t is the ith dimension component of the optimal population individual after iteration t, and r_1 is responsible for a linear decreasing function, which can be formulated as follows:

$$r_1 = a - t\frac{a}{T}, \tag{2}$$

where a is a constant, T is the maximum number of iterations, and $r_{2,}$ $r_{3,}$ and r_4 are the random numbers in the ranges $[0, 2\pi]$, $[-2, 2]$, and $[0, 1]$, respectively.

The updated solution is calculated for its fitness and then compared with the current optimal solution, and it will be replaced if a better solution is obtained. This is an iterative process. Algorithm 1 presents the pseudocode of SCA, and Figure 1 shows the concrete process.

3. Proposed ASCA

Compared with other algorithms, the original SCA has the advantages of simple structure and fewer parameters. The algorithm just relies on the sine and cosine functions for iteration to find the optimal solution. Original SCA has good global search ability, but its parameters cannot be suitable for the search process in the later period of the algorithm, which will slow down the convergence speed and reduce the population diversity. In this paper, two improved mechanisms are proposed to improve the search ability of SCA in the later period.

3.1. Adaptive Parameter. Original SCA has four main parameters, namely, r_1, r_2, r_3, and r_4, mentioned earlier, respectively. The values of sine and cosine reflect the stage of the algorithm. When the value of the sine function ($r_1 \sin(r_2)$) or the cosine function ($r_1 \cos(r_2)$) is between −1 and 1, the algorithm is in the local exploitation stage; otherwise, it is in the global search stage. Although both parameters r_1 and r_2 involved in the formula have an effect on the value, r_1 has a more significant influence. Original r_1 has a form of equation (2) which will decrease linearly with the iteration. However, the search process of the SCA is complex and not linear. Equation (2) does not meet the requirements. In order to make SCA have better computing power, r_1 is used to balance the global search and local exploitation of the algorithm. This idea has been mentioned in [75], but the idea of this paper is different from that one. We do not want the adaptive r_1 to be reduced too quickly in the early stages because that is not conducive to SCA global search. According to the operation process of SCA, the algorithm first enters the global search and then enters the local exploitation. This requires the r_1 update formula to have a larger value in the early stage to ensure a better global search capability and then to enter the local exploitation stage with a smaller value. Therefore, a new r_1 updating formula is proposed, as shown in the following formula:

$$r_1 = 4 \times \left(1 - \frac{t}{T}\right) \times \left(1 - 2^{((t/T)-1)}\right), \tag{3}$$

where T represents the maximum number of iterations and t represents the current number of iterations.

3.2. Chaotic Local Search (CLS). Chaos is one of the most common phenomena in nature, which looks disorganized but has a delicate internal structure. The main characteristics of chaos are "randomness," "ergodicity," and "regularity." Because the update process of SCA is entirely random, the

```
Parameter initialization: $N$ is the population size, dim is the dimension, $a$ is the control parameter, and $T$ is the maximum iteration number. Set current iteration number $t = 0$.
Initialize population $X$
**For** $i = 1$ to $N$
  Calculate the fitness
  Record the optimal individual $X_{\text{best}}$
**End For**
**While** ($t \le T$)
  Update $r_1$ by equation (2)
  **For** $i = 1$ to $N$
    **For** $j = 1$ to dim
      Update $r_2$, $r_3$, $r_4$
    **If** $r_4 < 0.5$
      Update $X$ by the sine part of equation (1)
    **Else**
      Update $X$ by the cosine part of equation (1)
    **End if**
    **End for**
  **End for**
  **For** $i = 1$ to $N$
    Calculate the fitness of the updated $X$
    Update $X_{\text{best}}$
  **End**
  $t = t + 1$
**End While**
Return best solution
```

Algorithm 1: Pseudocode of SCA.

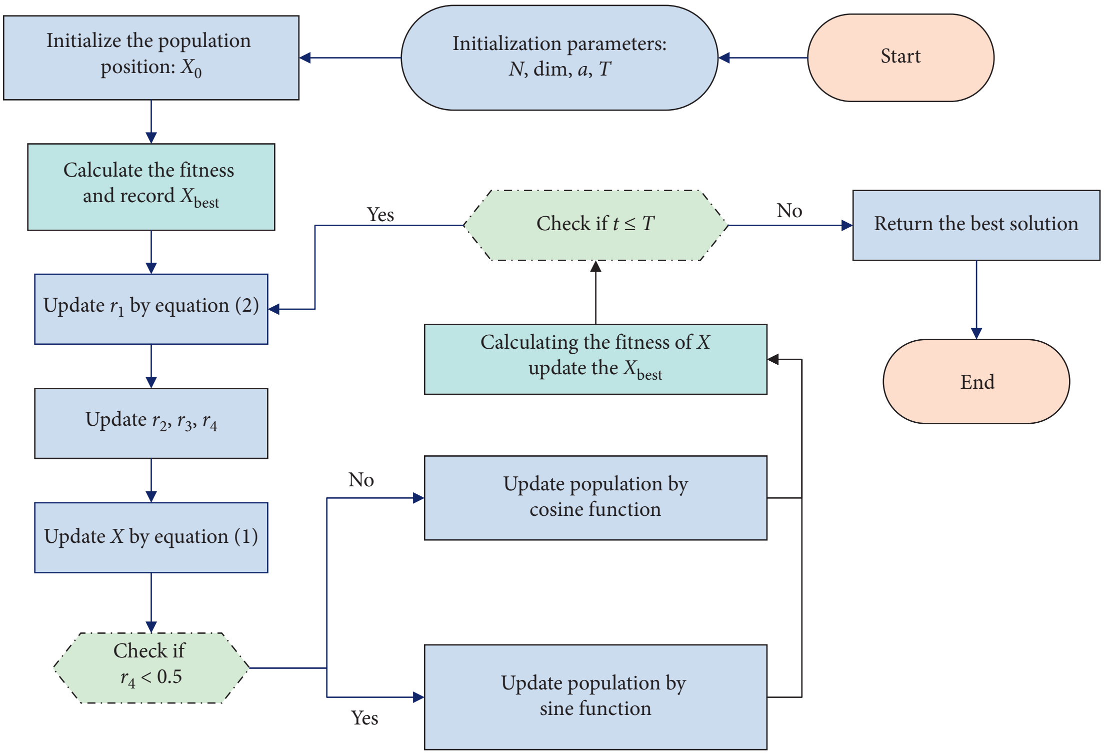
Initialize the population position: X_0
Initialization parameters: N, dim, a, T
Start
Calculate the fitness and record X_{best}
Yes
Check if $t \le T$
No
Return the best solution
Update r_1 by equation (2)
Calculating the fitness of X update the X_{best}
End
Update r_2, r_3, r_4
No
Update population by cosine function
Update X by equation (1)
Check if $r_4 < 0.5$
Update population by sine function
Yes

Figure 1: Flowchart of SCA.

chaos system can keep the randomness and traverse the whole search space as much as possible. CLS searches near the optimal individual and generates new individuals without repetition. Since population diversity decreases at the later stage of SCA, CLS can be used to improve population diversity and local exploitation ability as done in many studies [76–78]. There are many kinds of chaotic systems. In this paper, we choose a common logistic map, as shown in the following equation:

$$y_{k+1} = a y_k \left(1 - y_k\right), \tag{4}$$

where k is the number of iterations and a is the control parameter. When $a = 4$ and $y_1 \notin \{0.25, 0.5, 0.75, 1\}$, equation (4) is a chaotic system.

Local search (LS) can be used to search within the restricted region. Using LS to search the neighborhood of the current optimal solution has the opportunity to obtain a higher quality of the solution. However, in the case of insufficient probe depth, LS can also cause the algorithm to fall into local optimization. CLS is exactly the way to avoid this problem. It can help the algorithm avoid premature convergence due to the "randomness" of a chaotic system. The local search for chaos is shown in the following equation:

$$V = (1 - \lambda) \times X_{\text{best}} + \lambda \times \left(\text{lb} + y_k \times (\text{ub} - \text{lb})\right), \tag{5}$$

where V represents the new location formed by CLS, X_{best} is the current optimal solution, lb and ub represent the lower and upper limits of the space, y_k represents the chaotic sequence formed by equation (4), and λ is obtained from the following:

$$\lambda = \frac{(T - t + 1)}{T}, \tag{6}$$

where T is the maximum number of iterations and t is the current number of iterations.

3.3. Proposed ASCA. This section adds the two improvements mentioned above to SCA and details the entire ASCA process. In ASCA, first, the population is initialized like SCA and then r_1 is updated by equation (3). After updating with sine or cosine, chaotic, random search is performed near the current optimal solution and new individuals are generated. Algorithm 2 shows the pseudocode of the ASCA.

Figure 2 shows the main ASCA process. The time complexity of ASCA is determined by the number of algorithm iterations (T), the dimensions (dim), and the number of individuals (N). After analysis, the total complexity of ASCA is O (ASCA) = O (initialization) + O (calculation fitness) + O (fitness selection) + $T *$ (O (update position of sine cosine function) + O (calculation fitness) + O (fitness selection) + O (chaotic local search) + O (calculation fitness) + O (update position)). The complexity of the initial population is $O(N * \text{dim})$, the fitness $O(N)$ is calculated, the complexity $O(\mathbf{N})$ of the fitness selection is $O(\mathbf{N})$, the sine and cosine update is $O(N * \text{dim})$, and the CLS is $O(N * \text{dim})$. Therefore, its final complexity is as follows:

$$\begin{aligned} O(\text{ASCA}) &= O(N * \text{dim}) + O(N) + O(N) + T * \left(O(N * \text{dim}) + O(N) + O(N) + O(N * \text{dim}) + O(N) + O(N)\right) \\ &= O(N * \text{dim}) + 2O(N) + T * \left(2O(N * \text{dim}) + 4O(N)\right). \end{aligned} \tag{7}$$

4. Experimental Results and Discussions

In this section, the ASCA is compared with the traditional meta-heuristic algorithm, improved SCA variants, and improved meta-heuristic algorithms on 31 benchmark functions. All algorithms mentioned above were coded on MATLAB R2014b. For fair experimentation, we tested all the algorithms under uniform conditions. Table 1 lists the parameters involved in the simulation experiment. N is the number of populations; dim represents a dimension; *MaxFEs* represents the maximum number of function evaluations; *Flod* is the number of random runs. The parameters in Table 1 are used in the simulation experiments in Sections 4.3 and 4.4. The simulation experiment in Section 4.2 only changes the dimensions, and the other parameters are set the same.

4.1. Benchmark Function Validation. In this section, we will describe in detail the 31 benchmark functions which were used to test the performance of ASCA. The formulas of the functions are presented in Table 2, where Dim is the dimension, Range means the limit of the search space, and F (min) represents the optimal solution. We divide the 31 functions into three categories. F1~F7 are unimodal functions, which have only one global best solution, excluding the trap of local optimum. F8~F13 are multimodal functions, which have a great quantity of local optimum. F14~F31 are composite functions, which are more complex. The composite functions were taken from the last eight functions in CEC14 and the last ten functions in CEC17. The purpose of choosing different functions is to test the performance of ASCA more comprehensively and observe whether ASCA can meet the requirements of various problems.

4.2. Scalability Test. To carry out a comprehensive evaluation of the performance of ASCA, various dimensions of ASCA and SCA were tested on functions F1–F13, while other conditions are kept unchanged. In this experiment, four dimensions of 100, 200, 1000, and 2000 were selected to focus on how the change of problem dimension and overall proportion affects the algorithm performance. Table 3 shows

```
Initialization algorithm parameters: $N$ is the population size, dim is the dimension, $a$ is the control parameter, and $T$ is the maximum iteration number. Set current iteration number $t = 0$.
Initialize population $X$
For $i = 1$ to $N$
  Calculate the fitness
  Record the optimal individual $X_{\text{best}}$
End For
  While ($t \leq T$)
    Update $\boldsymbol{r}_1$ by equation (2)
    For $i = 1$ to $N$
      For $j = 1$ to dim
        Update $r_2$, $r_3$, $r_4$
        If $\boldsymbol{r}_4 < 0.5$
          Update $\boldsymbol{X}$ by the sine part of equation (1)
        Else
          Update $\boldsymbol{X}$ by the cosine part of equation (1)
        End if
      End for
    End for
    For $\boldsymbol{i} = 1$ to $N$
      Calculate the fitness of the updated $\boldsymbol{X}$
      Update $\boldsymbol{X}_{\text{best}}$
    End for
    Update $\lambda$ by equation (6)
    Generate chaotic sequence by equation (4)
    Substitute $X_{\text{best}}$ into equation (5) to generate new individuals $V$
    Calculate the fitness of $V$ and compared with $X_{\text{best}}$
    If $V$ is better
      Take $V$ as the optimal solution
    End if
    $t = t + 1$
  End While
Return the best solution
```

Algorithm 2: Pseudocode of ASCA.

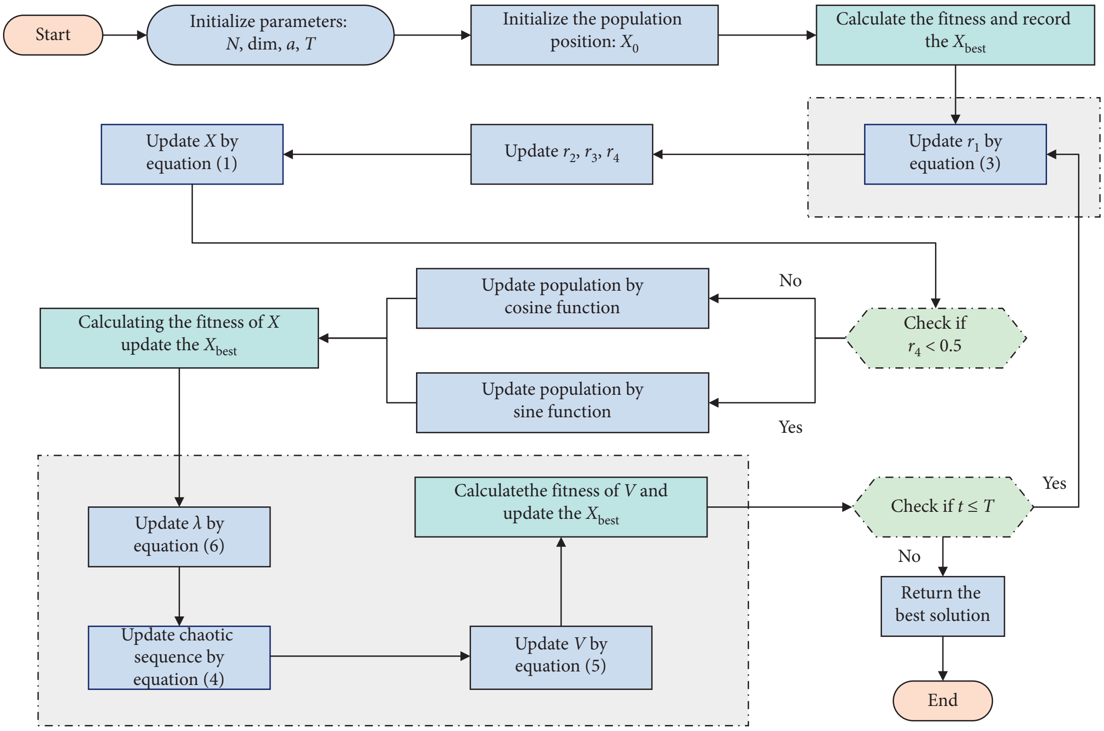
Start
Initialize parameters: N, dim, a, T
Initialize the population position: X_0
Calculate the fitness and record the X_{best}
Update r_1 by equation (3)
Update r_2, r_3, r_4
Update X by equation (1)
Check if $r_4 < 0.5$
No
Yes
Update population by cosine function
Update population by sine function
Calculating the fitness of X update the X_{best}
Update λ by equation (6)
Update chaotic sequence by equation (4)
Update V by equation (5)
Calculatethe fitness of V and update the X_{best}
Check if $t \leq T$
Yes
No
Return the best solution
End

Figure 2: Flowchart of ASCA.

Table 1: The parameters involved in the experiment.

N	Dim	MaxFEs	Flod
30	30	dim $*$ 10000	30

the experimental results, where Avg and Std represent the mean value and the standard deviation of the results, respectively. The experimental results show that ASCA performance does decrease with the increase of the problem dimension compared with the low dimension. However, ASCA still has an advantage over SCA in these 13 benchmark functions. When the dimension reached 2000, the performance of SCA had declined significantly, but ASCA could still achieve good results. It shows that ASCA can maintain the performance of the algorithm in high dimensions.

4.3. Comparison with Conventional Algorithms. In this section, ASCA was compared with 8 well-known algorithms in this field, namely, whale optimization algorithm (WOA), gray wolf optimizer (GWO), moth-flame optimization (MFO), bat algorithm (BA) [79], gravitational search algorithm (GSA) [80], sine cosine optimization (SCA), firefly algorithm (FA) [81], and particle swarm optimization (PSO). Table 4 presents the experimental data of ASCA and conventional meta-heuristic algorithms on 31 benchmark functions in detail. In Table 4, Avg represents the results of all test algorithms and Std represents the standard deviation. The smaller the Avg value, the higher the quality of the solution, and the smaller the Std, the more stable the algorithm. The symbols "+," "−," and "=" in the table represent that ASCA is superior, inferior to, and equal to other algorithms. In the last two columns of the table, Avg represents the average ranking of the test algorithm in each function; Rank indicates the rank of the average value of the Friedman test rank.

According to the results in Table 4, the Avg value of ASCA is 2.39, while the Avg value of SCA is 5.74, indicating that ASCA has a significant performance improvement over SCA. Among the nine algorithms tested, the average ranking of ASCA is the highest, which indicates that ASCA has a better comprehensive level in the selected function compared with SCA and other algorithms.

Table 5 shows the p values for ASCA and other algorithms via the Wilcoxon signed-rank test [82]. A p value of less than 0.05 indicates that the proposed algorithm is statistically significantly improved compared with other algorithms. As can be seen from the table, compared with FA, GSA, and PSO, the p value is greater than 0.05 only once, indicating that ASCA has significantly improved most functions. It must be mentioned that in the last few composite functions, the p values are all less than 0.05, which indicates that ASCA performs well in these functions.

Table 6 lists the running time of ASCA and other algorithms in seconds. The time complexity of each algorithm is certain, so the ranking of operation time is basically the same. It can be seen from the table that PSO takes the least time to run. The running time of ASCA is basically ranked second. Better quality solutions with less time can also indicate that improvements to SCA are effective.

The convergence curves of the algorithms can be seen in Figure 3. For F5 and F6, ASCA still has the opportunity to find better solutions in the later iteration of the algorithm after all other algorithms are stabilized. For F11~F14 and F24~F31, the quality of the ASCA solution is the highest, which indicates that ASCA performs better in multimode and composite functions. From the above experimental analysis, compared with the original SCA, ASCA has a smaller improvement in unimodal functions but a significant improvement in multimodal and composite functions.

4.4. Comparison with Advanced Algorithms. In this section, to further verify the performance and advantages of the ASCA algorithm, a set of algorithms, including OBSCA [56], ASCA_PSO [65], m_SCA, CBA [83], RCBA [84], and CDLOBA [85], are compared with ASCA on 31 benchmark functions. All the tested algorithms are tested under a unified framework. Table 7 shows the results, and it can be seen from the table that OBSCA and m_SCA are two powerful competitors of ASCA. Both algorithms performed well in unimodal functions, but they were ranked lower than ASCA in the end because ASCA performed very well in multimodal functions and composite functions. ASCA is stable in the top three in multimodal functions, which indicates that ASCA is easier to jump out of the local optimum than other improved algorithms.

A p value of less than 0.05 in the Wilcoxon signed-rank test indicates that the proposed algorithm is statistically significantly improved compared with other algorithms. The smaller the sort value, the better the method. Table 8 records the p values of ASCA and other algorithms. From Table 8, all the other algorithms have only one function with a p value higher than 0.05 except m_SCA and OBSCA. This shows that ASCA has better comprehensive performance compared with other improved algorithms.

Table 9 lists the running time of ASCA and other SCA improvement algorithms in seconds. As we mentioned before, the time complexity of the algorithm is roughly the same as the order of the running time. It can be seen from the table that OBSCA takes the least time to run. The running time of ASCA is ranked second. Compared with other improved algorithms, the time complexity of ASCA is relatively low.

The convergence curves of the ASCA algorithm and other modified, improved algorithms are shown in Figure 4. For F5, F12, and F13, ASCA has better convergence performance and continues to decline in the later stage of the algorithm, while other algorithms have entered the local optimum. For F11, the convergence speed of ASCA is higher than other algorithms. Although the

Table 2: Test benchmark functions.

ID	Function equation	Dim	Range	F (min)
F1	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	[−100, 100]	0
F2	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[−10, 10]	0
F3	$f_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{i} x_j)^2$	30	[−100, 100]	0
F4	$f_4(x) = \max_i\{\lvert x_i \rvert, 1 \le i \le n\}$	30	[−100, 100]	0
F5	$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[−30, 30]	0
F6	$f_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	[−100, 100]	0
F7	$f_7(x) = \sum_{i=1}^{n} ix_i^4 + \text{random}[0, 1)$	30	[−1.28, 1.28]	0
F8	$f_8(x) = \Sigma_{i=1}^{n} - x_i \sin(\sqrt{\lvert x_i \rvert})$	30	[−500, 500]	−418.982
F9	$f_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[−5.12, 5.12]	0
F10	$f_{10}(x) = -20\exp\left\{-0.2\sqrt{(1/n)\sum_{i=1}^{n} x_i}\right\} - \exp\{(1/n)\sum_{i=1}^{n} \cos(2\pi x_i)\} + e$	30	[−32, 32]	0
F11	$f_{11}(x) = (1/4000)\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1$	30	[−600, 600]	0
F12	$f_{12}(x) = (\pi/n)\left\{10\sin(ay_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2 + \sum_{i=1}^{n} \mu(x_i, 10, 100, 4)\right\}$	30	[−50, 50]	0
F13	$f_{13}(x) = 0.1\left\{\sin^2(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] + \sum_{i=1}^{n} \mu(x_i, 5, 100, 4)\right\}$	30	[−50, 50]	0
F14	Composition function 1	30	[−100, 100]	2300
F15	Composition function 2	30	[−100, 100]	2400
F16	Composition function 3	30	[−100, 100]	2500
F17	Composition function 4	30	[−100, 100]	2600
F18	Composition function 5	30	[−100, 100]	2700
F19	Composition function 6	30	[−100, 100]	2800
F20	Composition function 7	30	[−100, 100]	2900
F21	Composition function 8	30	[−100, 100]	3000
F22	Composition function 9	30	[−100, 100]	2100
F23	Composition function 10	30	[−100, 100]	2200
F24	Composition function 11	30	[−100, 100]	2300
F25	Composition function 12	30	[−100, 100]	2400
F26	Composition function 13	30	[−100, 100]	2500
F27	Composition function 14	30	[−100, 100]	2600
F28	Composition function 15	30	[−100, 100]	2700
F29	Composition function 16	30	[−100, 100]	2800
F30	Composition function 17	30	[−100, 100]	2900
F31	Composition function 18	30	[−100, 100]	3000

Table 3: Experimental results of ASCA and SCA in four dimensions.

F	Method	$D = 100$		$D = 200$		$D = 1000$		$D = 2000$	
		Avg	Std	Avg	Std	Avg	Std	Avg	Std
F1	ASCA	**7.0800E − 08**	**1.1879E − 07**	**9.6422E − 08**	**1.2534E − 07**	**6.3758E − 07**	**9.8578E − 07**	**1.1352E − 06**	**1.9652E − 06**
	SCA	3.2637E − 03	1.1400E − 02	5.7569E + 03	4.3222E + 03	2.4564E + 05	7.1098E + 04	5.9111E + 05	1.4387E + 05
F2	ASCA	1.9994E − 04	1.9807E − 04	**4.1180E − 04**	**4.9644E − 04**	**2.3731E − 03**	**3.0579E − 03**	**4.7445E − 03**	**3.3163E − 03**
	SCA	**7.2555E − 42**	**2.5542E − 41**	2.6017E − 03	1.4245E − 02	6.5535E + 04	0.0000E + 00	6.5535E + 04	0.0000E + 00
F3	ASCA	2.5736E − 02	3.1277E − 02	2.2191E − 01	3.1716E − 01	2.2472E + 01	2.3216E + 01	**1.9218E + 02**	**2.4396E + 02**
	SCA	7.5318E + 04	2.2346E + 04	4.3918E + 05	7.4108E + 04	1.1831E + 07	1.9791E + 06	4.9826E + 07	6.1056E + 06
F4	ASCA	**2.5085E − 05**	**2.7368E − 05**	**1.8588E − 05**	**1.4198E − 05**	**1.8031E − 05**	**2.0301E − 05**	**2.0301E − 05**	**2.0456E − 05**
	SCA	5.8653E + 01	7.1351E + 00	8.3517E + 01	2.6737E + 00	9.8554E + 01	2.7689E − 01	9.9490E + 01	8.8996E − 02
F5	ASCA	**1.4088E − 06**	**2.3201E − 06**	**8.9254E − 07**	**1.5840E − 06**	**7.6363E − 06**	**1.1879E − 05**	**8.7737E − 06**	**1.6475E − 05**
	SCA	2.8002E + 05	4.8604E + 05	6.4483E + 07	3.4125E + 07	1.9486E + 09	3.4141E + 08	4.6074E + 09	8.3557E + 08
F6	ASCA	**7.9142E − 08**	**1.4030E − 07**	**1.8936E − 07**	**3.2907E − 07**	**5.3854E − 07**	**1.0086E − 06**	**9.6695E − 07**	**1.7191E − 06**
	SCA	2.1122E + 01	2.8804E + 00	6.2992E + 03	4.0380E + 03	2.4397E + 05	7.5925E + 04	5.8281E + 05	1.9987E + 05
F7	ASCA	**7.2931E − 05**	**6.1406E − 05**	**1.2659E − 04**	**1.1311E − 04**	**1.4469E − 04**	**9.5112E − 05**	**2.4415E − 04**	**1.8825E − 04**
	SCA	5.9995E − 01	1.6576E + 00	1.5796E + 02	9.2936E + 01	2.8621E + 04	4.9537E + 03	1.4943E + 05	3.1153E + 04
F8	ASCA	**−4.1898E + 04**	**2.9465E − 07**	**−8.3797E + 04**	**3.1457E − 07**	**−4.1898E + 05**	**1.2922E − 06**	**−8.3797E + 05**	**6.5449E − 06**
	SCA	−8.6998E + 03	3.9497E + 02	−1.2057E + 04	4.8797E + 02	−2.6986E + 04	1.2661E + 03	−3.8845E + 04	1.9972E + 03
F9	ASCA	**4.8475E − 08**	**1.5995E − 07**	**1.7277E − 07**	**4.1340E − 07**	**2.8737E − 07**	**6.7205E − 07**	**1.0367E − 06**	**2.3071E − 06**
	SCA	6.0184E + 01	6.4215E + 01	3.1423E + 02	1.5382E + 02	1.5254E + 03	8.4085E + 02	2.6418E + 03	1.2477E + 03
F10	ASCA	**2.6680E − 05**	**2.4479E − 05**	**3.0772E − 05**	**3.0132E − 05**	**2.6694E − 05**	**1.8501E − 05**	**2.0674E − 05**	**1.9432E − 05**
	SCA	1.7720E + 01	7.0734E + 00	1.7346E + 01	6.1273E + 00	1.9298E + 01	3.9345E + 00	1.8818E + 01	4.1409E + 00
F11	ASCA	**3.6407E − 07**	**1.3782E − 06**	**4.8644E − 08**	**8.9373E − 08**	**1.7230E − 07**	**3.2889E − 07**	**1.6901E − 07**	**3.1936E − 07**
	SCA	1.6622E − 01	5.3482E − 01	5.1622E + 01	3.8783E + 01	2.3073E + 03	7.5753E + 02	4.3663E + 03	1.3280E + 03
F12	ASCA	**1.2584E − 10**	**2.0943E − 10**	**1.1907E − 10**	**2.4509E − 10**	**6.5825E − 11**	**1.3885E − 10**	**2.5642E − 11**	**3.4249E − 11**
	SCA	3.9778E + 05	1.1657E + 06	1.4309E + 08	1.0262E + 08	4.7085E + 09	8.3916E + 08	1.1367E + 10	3.0100E + 09
F13	ASCA	**4.0803E − 09**	**7.1966E − 09**	**4.8761E − 09**	**9.1712E − 09**	**2.3380E − 08**	**4.2260E − 08**	**4.2026E − 08**	**1.3053E − 07**
	SCA	2.7823E + 06	4.4572E + 06	2.4192E + 08	1.1153E + 08	8.7708E + 09	1.7624E + 09	2.1152E + 10	3.3759E + 09

convergence speed of m_SCA is overridden in the late stage, the quality of the solution obtained is lower than that of ASCA.

In general, compared with other improved mechanisms, the mechanism used in this paper improves the balance of SCA search exploitation more effectively. The strong ability to jump out of the local optimum of the algorithm enables ASCA to get a better solution. This makes up for SCA's shortcomings.

4.5. Comparison with SCA Variants on CEC 2014. In this section, to verify the performance and advantage of the ASCA, we compared OBSCA, SCADE, CESCA [86], and CLSCA with ASCA on 30 benchmark functions of CEC 2014. Table 10 records experimental comparisons between ASCA and other algorithms on the CEC 2014 functions. As can be seen from Table 10, ASCA has been ranked first in most cases on the first 22 functions. This suggests that ASCA performs better in unimodal functions, simple multimodal functions, and hybrid functions. The last eight are composite functions, and the best of which is CLSCA. However, we can see that there is not a big difference in the average between them.

A p value of less than 0.05 of the Wilcoxon signed-rank test indicates that the proposed algorithm is statistically significantly improved compared with other algorithms. Table 11 records the p values of ASCA and other SCA variants. The values greater than 0.05 have been given in bold in the table. We can see from the p values that ASCA has significant improvement in all functions compared with CESCA.

The convergence curves of ASCA and other modified SCA variants are shown in Figure 5. From Figure 5, we can see that the convergence of ASCA on the unimodal functions is faster than other improved algorithms. In F15, we can see that the ASCA curve was relatively stable for some time in the early period and then continued to decline. This shows that ASCA did not fall into the local optimum and successfully found a better solution.

4.6. The Impact of CLS and Adaptive r_1. The improved SCA introduces two strategies, namely, adaptive r_1 and CLS. In this section, we examine the impact of these two strategies on SCA. CLSSCA only uses the CLS strategy, ADSCA only changes the adaptive r_1, and ASCA uses both strategies.

Figure 6 shows the results of the qualitative analysis of 23 benchmark functions by ASCA and three other algorithms. The graphs of the first column show the three-dimensional location distribution of ASCA search history. The graphs of the second column show the two-dimensional location distribution of ASCA search history. The graphs of the third column show the trajectory of ASCA. The graphs of the

Table 4: Comparison of ASCA with conventional meta-heuristic algorithms.

	F1		F2	
	Avg	Std	Avg	Std
ASCA	2.3330E − 06	1.2421E − 05	2.2478E − 20	5.7436E − 20
WOA	**0.0000E + 00**	**0.0000E + 00**	**0.0000E + 00**	**0.0000E + 00**
GWO	**0.0000E + 00**	**0.0000E + 00**	**0.0000E + 00**	**0.0000E + 00**
MFO	1.6667E + 03	4.6113E + 03	3.7667E + 01	2.4870E + 01
BA	7.4407E − 01	3.2216E − 01	3.6896E + 00	2.4760E + 00
GSA	1.4593E + 01	1.3802E + 00	1.5447E + 01	8.2525E − 01
SCA	4.1528E − 51	2.2488E − 50	2.1233E − 58	1.1254E − 57
FA	1.1332E + 04	1.2794E + 03	4.9150E + 01	3.3958E + 00
PSO	9.8138E + 01	1.1854E + 01	4.5920E + 01	2.8925E + 00
	F3		F4	
	Avg	Std	Avg	Std
ASCA	6.4143E − 04	1.9433E − 03	7.8924E − 05	8.4861E − 05
WOA	2.0359E + 01	4.7215E + 01	1.1973E + 01	2.1214E + 01
GWO	**1.6086E − 182**	**0.0000E + 00**	**5.9974E − 153**	**1.7657E − 152**
MFO	1.5400E + 04	1.3639E + 04	5.8938E + 01	1.2657E + 01
BA	2.5609E − 01	1.7056E − 01	4.3619E + 00	4.5128E + 00
GSA	3.0143E + 02	6.3563E + 01	1.7752E + 00	9.2470E − 02
SCA	8.9302E − 02	2.8451E − 01	1.9325E − 03	6.1880E − 03
FA	1.9568E + 04	2.8752E + 03	4.0517E + 01	1.9862E + 00
PSO	1.7674E + 02	2.0807E + 01	3.7145E + 00	2.3219E − 01
	F5		F6	
	Avg	Std	Avg	Std
ASCA	**3.0535E − 06**	**3.7961E − 06**	**2.0236E − 07**	**4.3142E − 07**
WOA	2.4246E + 01	4.0697E − 01	5.6645E − 06	2.8108E − 06
GWO	2.6177E + 01	8.7615E − 01	4.1693E − 01	3.0299E − 01
MFO	1.5254E + 04	3.4012E + 04	2.3367E + 03	5.0321E + 03
BA	2.4678E + 02	2.3923E + 02	8.2282E − 01	4.9969E − 01
GSA	8.5617E + 03	1.9118E + 03	1.4275E + 01	2.0027E + 00
SCA	2.7260E + 01	6.6550E − 01	3.6259E + 00	2.7338E − 01
FA	7.6689E + 06	1.8764E + 06	1.1399E + 04	1.3266E + 03
PSO	8.6097E + 04	1.5884E + 04	1.0160E + 02	1.0641E + 01
	F7		F8	
	Avg	Std	Avg	Std
ASCA	1.2104E − 04	9.2594E − 05	**−1.2569E + 04**	**1.6811E − 07**
WOA	1.5120E − 04	1.6994E − 04	−1.2329E + 04	6.7277E + 02
GWO	**5.1494E − 05**	**3.5004E − 05**	−6.0782E + 03	7.1476E + 02
MFO	2.6939E + 00	5.4397E + 00	−8.8728E + 03	9.6610E + 02
BA	1.2141E + 01	8.1178E + 00	−7.2169E + 03	9.1056E + 02
GSA	3.0284E + 01	4.7061E + 00	−2.5459E + 03	3.6549E + 02
SCA	2.7688E − 03	2.3669E − 03	−4.4052E + 03	2.6122E + 02
FA	3.9469E + 00	9.2795E − 01	−4.1121E + 03	2.0480E + 02
PSO	1.0107E + 02	2.7005E + 01	−6.5495E + 03	8.4124E + 02
	F9		F10	
	Avg	Std	Avg	Std
ASCA	1.0832E − 08	4.4654E − 08	1.6988E − 05	5.4503E − 05
WOA	**0.0000E + 00**	**0.0000E + 00**	**3.7303E − 15**	**2.5380E − 15**
GWO	**0.0000E + 00**	**0.0000E + 00**	7.6383E − 15	1.0840E − 15
MFO	1.5223E + 02	4.8546E + 01	1.4332E + 01	8.1908E + 00
BA	2.4543E + 02	2.1497E + 01	2.7730E + 00	3.3513E + 00
GSA	1.9564E + 02	9.8568E + 00	4.1481E + 00	1.2331E − 01
SCA	2.0911E + 00	8.2195E + 00	9.9345E + 00	9.7043E + 00
FA	2.3036E + 02	1.1671E + 01	1.6007E + 01	3.4583E − 01
PSO	3.4207E + 02	1.9904E + 01	7.8352E + 00	2.5812E − 01

Table 4: Continued.

	F11		F12	
	Avg	Std	Avg	Std
ASCA	**3.7007E − 18**	**2.0270E − 17**	**1.2807E − 09**	**2.7203E − 09**
WOA	3.7052E − 04	2.0294E − 03	8.7365E − 07	2.4615E − 07
GWO	3.3143E − 04	1.8153E − 03	3.0267E − 02	1.5556E − 02
MFO	1.2101E + 01	3.1293E + 01	2.3916E − 01	5.5055E − 01
BA	1.0961E − 02	1.1122E − 02	8.4409E + 00	2.9890E + 00
GSA	5.5899E − 01	3.4159E − 02	1.3623E + 00	2.9445E − 01
SCA	5.2726E − 05	2.8879E − 04	3.4995E − 01	4.6493E − 02
FA	1.0095E + 02	1.0635E + 01	2.7597E + 06	1.0047E + 06
PSO	1.0061E + 00	2.3434E − 02	3.3894E + 00	4.5443E − 01
	F13		F14	
	Avg	Std	Avg	Std
ASCA	**3.2190E − 08**	**6.2088E − 08**	**2.5000E + 03**	**2.1268E − 03**
WOA	4.7335E − 04	2.0232E − 03	2.6335E + 03	1.0899E + 01
GWO	4.8328E − 01	1.8192E − 01	2.6342E + 03	1.0084E + 01
MFO	1.3669E + 07	7.4867E + 07	2.6903E + 03	7.0682E + 01
BA	1.5213E − 01	1.0744E − 01	2.6152E + 03	2.3465E − 03
GSA	8.2279E + 00	8.8182E − 01	2.6153E + 03	4.8116E + 00
SCA	1.9909E + 00	1.3596E − 01	2.6674E + 03	1.2114E + 01
FA	1.7010E + 07	4.7690E + 06	2.7269E + 03	2.1255E + 01
PSO	1.5120E + 01	1.9659E + 00	2.6160E + 03	4.2593E − 01
	F15		F16	
	Avg	Std	Avg	Std
ASCA	2.6001E + 03	6.3320E − 02	**2.7000E + 03**	**2.6918E − 04**
WOA	2.6050E + 03	3.8676E + 00	2.7158E + 03	1.5678E + 01
GWO	**2.6000E + 03**	**6.7444E − 04**	2.7114E + 03	5.3113E + 00
MFO	2.6724E + 03	2.2094E + 01	2.7155E + 03	7.1497E + 00
BA	2.6586E + 03	2.5374E + 01	2.7289E + 03	1.0158E + 01
GSA	2.6082E + 03	4.4958E − 01	2.7019E + 03	1.2530E − 01
SCA	2.6001E + 03	5.4654E − 02	2.7237E + 03	8.5886E + 00
FA	2.7056E + 03	6.1706E + 00	2.7338E + 03	5.4653E + 00
PSO	2.6270E + 03	5.5839E + 00	2.7121E + 03	6.4390E + 00
	F17		F18	
	Avg	Std	Avg	Std
ASCA	2.7125E + 03	2.9661E + 01	**2.9000E + 03**	**1.8884E − 03**
WOA	**2.7005E + 03**	**1.4932E − 01**	3.7265E + 03	3.6375E + 02
GWO	2.7446E + 03	5.9818E + 01	3.3440E + 03	1.2607E + 02
MFO	2.7026E + 03	1.4303E + 00	3.6291E + 03	1.9394E + 02
BA	2.7005E + 03	1.4678E − 01	4.0225E + 03	2.0074E + 02
GSA	2.7752E + 03	4.2755E + 01	3.2177E + 03	1.2743E + 02
SCA	2.7020E + 03	6.1195E − 01	3.4289E + 03	3.1343E + 02
FA	2.7024E + 03	2.5708E − 01	3.7991E + 03	3.4213E + 01
PSO	2.7772E + 03	4.3036E + 01	3.4131E + 03	2.8290E + 02
	F19		F20	
	Avg	Std	Avg	Std
ASCA	**3.0000E + 03**	**4.8946E − 04**	8.9941E + 05	3.4033E + 06
WOA	4.9758E + 03	6.4609E + 02	5.9654E + 06	4.6258E + 06
GWO	3.9277E + 03	2.7353E + 02	3.6914E + 05	6.0632E + 05
MFO	3.9186E + 03	1.4967E + 02	1.4215E + 06	2.7965E + 06
BA	5.3665E + 03	7.3927E + 02	4.0463E + 07	4.1550E + 07
GSA	4.6069E + 03	3.3691E + 02	3.6009E + 07	5.5973E + 07
SCA	4.7703E + 03	2.9643E + 02	1.1225E + 07	4.3019E + 06
FA	4.2024E + 03	4.9487E + 01	3.2717E + 06	1.0524E + 06
PSO	6.8226E + 03	8.9983E + 02	**6.9453E + 04**	**1.1996E + 05**
	F21		F22	
	Avg	Std	Avg	Std
ASCA	1.4512E + 05	1.7708E + 05	3.3775E + 03	3.0252E + 02
WOA	7.9432E + 04	6.3625E + 04	2.2744E + 03	3.6506E + 01
GWO	3.9819E + 04	3.3602E + 04	2.3243E + 03	7.0801E + 01
MFO	4.3043E + 04	2.8771E + 04	*3.1364E + 03*	*9.4782E + 02*
BA	2.1116E + 04	4.8976E + 04	**2.1660E + 03**	**4.4141E + 01**

Table 4: Continued.

GSA	**7.9349E+03**	**8.3345E+02**	2.1833E+03	3.6774E+01
SCA	2.2197E+05	9.4119E+04	3.1791E+03	2.0410E+02
FA	1.7450E+05	3.9292E+04	3.1668E+03	1.8591E+02
PSO	1.5745E+04	7.3227E+03	2.1840E+03	4.2251E+01
	F23		F24	
	Avg	Std	Avg	Std
ASCA	2.4651E+03	2.3283E+01	**2.5000E+03**	**1.1801E−05**
WOA	2.4165E+03	4.6503E+01	3.1893E+03	1.5144E+02
GWO	**2.2885E+03**	**1.9050E+01**	2.8812E+03	4.1603E+01
MFO	2.3939E+03	4.4459E+01	2.9633E+03	3.1091E+01
BA	2.4858E+03	5.8057E+01	3.6076E+03	2.4092E+02
GSA	2.2990E+03	1.1303E+01	3.2643E+03	2.2707E+02
SCA	2.4617E+03	1.7731E+01	3.2720E+03	4.1922E+01
FA	2.4454E+03	1.1666E+01	3.0974E+03	1.9441E+01
PSO	2.4326E+03	3.7082E+01	4.6577E+03	5.0388E+02
	F25		F26	
	Avg	Std	Avg	Std
ASCA	**2.6000E+03**	**7.0119E−04**	**2.7000E+03**	**5.1240E−02**
WOA	2.7120E+03	3.4202E+02	2.7118E+03	6.4542E+01
GWO	3.1621E+03	3.2584E+02	3.2363E+03	9.4893E+01
MFO	3.4989E+03	2.5820E+01	3.7493E+03	7.8417E+02
BA	2.8712E+03	5.3437E+02	3.0090E+03	6.7624E+01
GSA	2.6296E+03	1.0893E+00	2.9440E+03	1.7963E+01
SCA	3.8943E+03	9.4149E+01	3.6661E+03	1.4514E+02
FA	3.6861E+03	2.3440E+01	4.0768E+03	1.6952E+02
PSO	2.6671E+03	4.1424E+00	2.9662E+03	5.6622E+01
	F27		F28	
	Avg	Std	Avg	Std
ASCA	**2.8000E+03**	**2.7881E−02**	**2.9000E+03**	**4.8742E−03**
WOA	3.5536E+03	1.9767E+03	3.9318E+03	2.2550E+02
GWO	5.1441E+03	8.0607E+02	3.6977E+03	1.0234E+02
MFO	6.7116E+03	4.5434E+02	3.6229E+03	9.1116E+01
BA	4.7719E+03	3.4172E+03	3.9589E+03	2.1087E+02
GSA	3.0394E+03	1.1567E+01	4.4216 E+03	2.3957E+02
SCA	7.8893E+03	3.9529E+02	4.0688E+03	1.0206E+02
FA	7.2147E+03	1.5432E+02	3.8936E+03	8.6066E+01
PSO	3.5324E+03	6.7693E+02	4.7154E+03	6.7030E+02
	F29		F30	
	Avg	Std	Avg	Std
ASCA	**3.0000E+03**	**1.5934E−03**	**3.1000E+03**	**8.7049E−03**
WOA	3.2742E+03	6.8647E+02	4.3872E+03	5.0739E+02
GWO	3.8199E+03	4.8415E+02	3.5279E+03	1.2876E+02
MFO	4.8795E+03	6.9903E+02	4.1000E+03	2.9011E+02
BA	3.4817E+03	6.8071E+02	4.6901E+03	2.9879E+02
GSA	3.3553E+03	6.1989E+01	3.5926E+03	1.9505E+02
SCA	5.6329E+03	6.1762E+02	4.1683E+03	3.3903E+02
FA	4.3515E+03	6.4816E+02	4.4790E+03	1.8092E+02
PSO	3.2998E+03	2.8212E+01	3.9709E+03	2.9056E+02

	F31				
	Avg	Std	+/−/=	Avg	Rank
ASCA	**5.9579E+03**	**9.6918E+03**	~	**2.39**	**1**
WOA	2.1807E+06	2.9807E+06	20/7/4	3.77	3
GWO	7.0971E+05	9.6621E+05	18/0/2	3.19	2
MFO	2.2985E+06	4.0448E+06	25/1/5	6.00	8
BA	1.1769E+06	9.2858E+05	27/2/2	5.58	5
GSA	3.0195E+05	1.3115E+05	28/2/1	4.81	4
SCA	1.2701E+07	3.2456E+07	22/4/5	5.74	6
FA	1.1495E+08	3.4824E+07	27/3/1	7.61	9
PSO	2.5264E+06	9.3111E+05	27/3/1	5.81	7

Table 5: The p values of ASCA versus other algorithms.

	WOA	GWO	MFO	BA	GSA	SCA	FA	PSO
F1	1.7344E − 06	1.7344E − 06	1.4785E − 02	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F2	1.7344E − 06	1.7344E − 06	3.0434E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F3	1.7344E − 06	1.7344E − 06	7.6909E − 06	1.7344E − 06	1.7344E − 06	1.1499E − 04	1.7344E − 06	1.7344E − 06
F4	3.3173E − 04	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	**9.5899E − 01**	1.7344E − 06	1.7344E − 06
F5	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F6	1.7344E − 06	2.3534E − 06	**1.6503E − 01**	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F7	**5.7165E − 01**	1.7423E − 04	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F8	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F9	4.8828E − 04	4.8828E − 04	1.7344E − 06	1.7344E − 06	1.7344E − 06	**5.6943E − 01**	1.7344E − 06	1.7344E − 06
F10	1.7344E − 06	1.7344E − 06	3.1817E − 06	1.7344E − 06	1.7344E − 06	1.7988E − 05	1.7344E − 06	1.7344E − 06
F11	**1.0000E + 00**	**1.0000E + 00**	1.2279E − 05	1.7344E − 06	1.7344E − 06	**5.0000E − 01**	1.7344E − 06	1.7344E − 06
F12	1.7344E − 06	1.7344E − 06	**1.0201E − 01**	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F13	1.7344E − 06	1.7344E − 06	2.0671E − 02	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F14	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F15	1.9209E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	**3.2857E − 01**	1.7344E − 06	1.7344E − 06
F16	3.6094E − 03	4.8009E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F17	1.7344E − 06	**5.5774E − 01**	**9.3676E − 02**	1.7344E − 06	6.8923E − 05	4.0715E − 05	4.5336E − 04	1.4839E − 03
F18	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F19	1.9209E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F20	6.1564E − 04	1.0357E − 03	1.2866E − 03	5.2165E − 06	8.1878E − 05	2.3534E − 06	3.5888E − 04	3.1603E − 02
F21	**2.5364E − 01**	**1.1093E − 01**	**1.0201E − 01**	**1.0201E − 01**	**1.0201E − 01**	2.8486E − 02	**1.7791E − 01**	**1.0201E − 01**
F22	1.7344E − 06	1.7344E − 06	**1.1093E − 01**	1.7344E − 06	1.7344E − 06	1.1079E − 02	2.5846E − 03	1.7344E − 06
F23	2.2248E − 04	1.7344E − 06	1.0246E − 05	**6.8714E − 02**	1.7344E − 06	**2.6230E − 01**	7.1570E − 04	1.1138E − 03
F24	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F25	3.8931E − 03	1.3601E − 05	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F26	3.1123E − 05	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F27	1.4793E − 02	2.3534E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F28	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F29	**3.7094E − 01**	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F30	1.9209E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F31	4.7292E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06

Table 6: Running time of ASCA and other algorithms.

F	ASCA	WOA	GWO	MFO	BA	GSA	SCA	FA	PSO
F1	2.35E + 02	1.19E + 03	3.19E + 02	2.62E + 02	3.86E + 02	3.79E + 03	2.50E + 02	2.85E + 03	**1.58E + 02**
F2	2.75E + 02	1.19E + 03	3.48E + 02	2.86E + 02	3.95E + 02	3.67E + 03	2.70E + 02	2.92E + 03	**1.90E + 02**
F3	1.27E + 03	2.16E + 03	1.32E + 03	1.29E + 03	1.39E + 03	4.75E + 03	1.22E + 03	3.82E + 03	**1.16E + 03**
F4	2.81E + 02	1.16E + 03	3.42E + 02	2.93E + 02	4.13E + 02	3.77E + 03	2.88E + 02	2.36E + 03	**2.02E + 02**
F5	3.32E + 02	1.26E + 03	4.45E + 02	3.31E + 02	4.74E + 02	3.85E + 03	3.52E + 02	2.87E + 03	**2.64E + 02**
F6	2.79E + 02	1.23E + 03	3.94E + 02	3.21E + 02	4.23E + 02	3.74E + 03	3.13E + 02	2.90E + 03	**2.27E + 02**
F7	3.78E + 02	1.42E + 03	4.98E + 02	3.92E + 02	5.48E + 02	3.81E + 03	4.14E + 02	3.19E + 03	**3.24E + 02**
F8	3.13E + 02	1.30E + 03	4.26E + 02	3.38E + 02	4.65E + 02	3.75E + 03	3.31E + 02	2.85E + 03	**2.62E + 02**
F9	3.29E + 02	1.31E + 03	4.33E + 02	3.52E + 02	5.03E + 02	3.89E + 03	3.47E + 02	3.12E + 03	**2.62E + 02**
F10	3.13E + 02	1.28E + 03	4.25E + 02	3.67E + 02	5.11E + 02	3.84E + 03	3.56E + 02	2.83E + 03	**2.84E + 02**
F11	3.48E + 02	1.22E + 03	4.18E + 02	3.63E + 02	5.20E + 02	3.92E + 03	3.73E + 02	2.81E + 03	**2.91E + 02**
F12	6.15E + 02	1.47E + 03	7.10E + 02	6.30E + 02	7.80E + 02	3.94E + 03	6.38E + 02	3.07E + 03	**5.45E + 02**
F13	6.08E + 02	1.48E + 03	6.95E + 02	6.08E + 02	7.64E + 02	3.94E + 03	6.23E + 02	3.01E + 03	**5.30E + 02**
F14	6.47E + 02	1.63E + 03	8.08E + 02	6.66E + 02	8.09E + 02	3.87E + 03	6.48E + 02	3.08E + 03	**5.66E + 02**
F15	5.63E + 02	1.54E + 03	6.73E + 02	5.57E + 02	7.06E + 02	3.83E + 03	5.67E + 02	2.81E + 03	**4.87E + 02**
F16	5.73E + 02	1.54E + 03	7.44E + 02	5.97E + 02	7.45E + 02	3.88E + 03	6.16E + 02	2.88E + 03	**5.11E + 02**
F17	1.83E + 03	2.81E + 03	1.98E + 03	1.76E + 03	2.05E + 03	4.78E + 03	1.81E + 03	3.99E + 03	**1.72E + 03**
F18	1.72E + 03	2.66E + 03	1.87E + 03	1.70E + 03	2.02E + 03	4.92E + 03	1.78E + 03	3.84E + 03	**1.69E + 03**
F19	6.62E + 02	1.53E + 03	7.93E + 02	6.21E + 02	8.48E + 02	3.79E + 03	6.76E + 02	2.83E + 03	**5.87E + 02**
F20	7.43E + 02	1.54E + 03	9.16E + 02	7.39E + 02	9.37E + 02	3.76E + 03	7.74E + 02	2.89E + 03	**6.92E + 02**
F21	5.35E + 02	1.51E + 03	7.09E + 02	5.68E + 02	7.26E + 02	3.43E + 03	5.78E + 02	2.73E + 03	**4.71E + 02**
F22	6.14E + 02	1.40E + 03	8.00E + 02	6.58E + 02	8.21E + 02	3.59E + 03	6.54E + 02	2.81E + 03	**5.44E + 02**
F23	6.65E + 02	1.52E + 03	8.51E + 02	7.06E + 02	8.80E + 02	3.63E + 03	7.11E + 02	2.83E + 03	**5.78E + 02**
F24	7.62E + 02	1.69E + 03	8.93E + 02	7.13E + 02	8.43E + 02	3.76E + 03	7.40E + 02	2.80E + 03	**6.13E + 02**

Table 6: Continued.

F	ASCA	WOA	GWO	MFO	BA	GSA	SCA	FA	PSO
F25	$7.23E+02$	$1.45E+03$	$8.61E+02$	$7.13E+02$	$8.92E+02$	$3.64E+03$	$7.26E+02$	$2.99E+03$	**6.38E + 02**
F26	$7.27E+02$	$1.57E+03$	$8.61E+02$	$7.23E+02$	$8.55E+02$	$3.67E+03$	$7.57E+02$	$2.91E+03$	**6.05E + 02**
F27	$7.92E+02$	$1.58E+03$	$9.20E+02$	$7.87E+02$	$9.33E+02$	$3.95E+03$	$8.08E+02$	$3.10E+03$	**6.86E + 02**
F28	$8.93E+02$	$1.82E+03$	$9.78E+02$	$8.55E+02$	$1.08E+03$	$3.91E+03$	$8.73E+02$	$2.96E+03$	**7.54E + 02**
F29	$8.23E+02$	$1.65E+03$	$9.23E+02$	$7.92E+02$	$9.65E+02$	$3.71E+03$	$7.99E+02$	$2.95E+03$	**7.21E + 02**
F30	$7.12E+02$	$1.55E+03$	$8.38E+02$	$7.02E+02$	$8.61E+02$	$3.69E+03$	$7.39E+02$	$2.86E+03$	**5.81E + 02**
F31	$8.95E+02$	$1.68E+03$	$1.02E+03$	$8.71E+02$	$1.06E+03$	$4.12E+03$	$9.48E+02$	$3.19E+03$	**8.30E + 02**

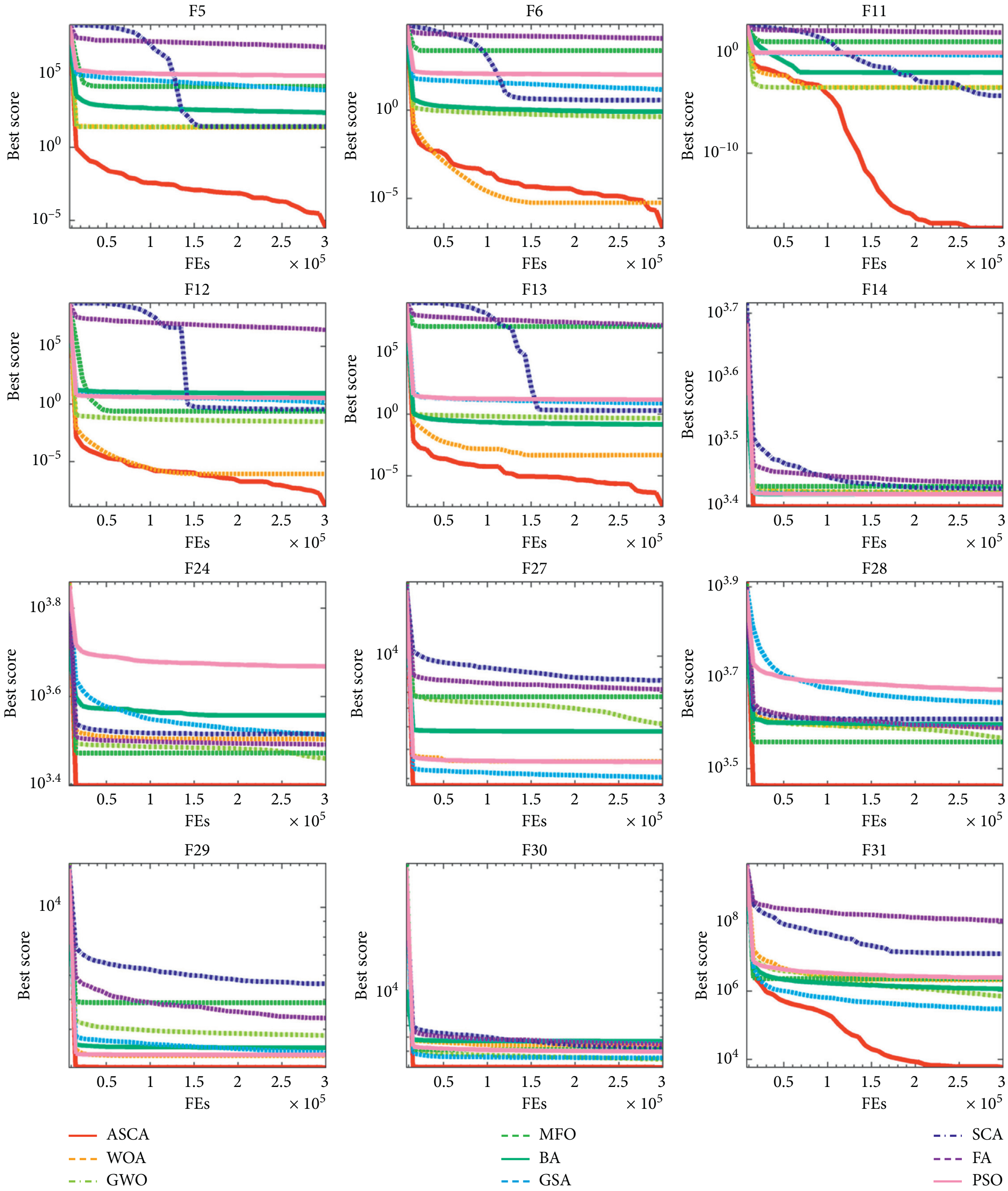

Figure 3: Convergence curves of ASCA and other algorithms (first row: F5, F6, and F11; second row: F12, F13, and F14; third row: F24, F27, and F28; forth row: F29, F30, and F31).

Table 7: Comparison results of ASCA and several SCA variants.

	F1		F2	
	Avg	Std	Avg	Std
ASCA	2.3329*E* − 06	1.2421*E* − 05	3.0304*E* − 21	8.0602*E* − 21
OBSCA	**3.5996E − 107**	**1.6806E − 106**	**4.4814E − 91**	**2.3248E − 90**
ASCA_PSO	8.1682*E* + 01	1.2055*E* + 01	4.1769*E* + 01	3.2941*E* + 00
m_SCA	5.6353*E* − 51	2.6561*E* − 50	1.0909*E* − 32	5.5385*E* − 32
CBA	1.7069*E* − 07	6.3009*E* − 07	2.4781*E* + 01	4.5089*E* + 01
RCBA	9.9471*E* − 03	3.0946*E* − 03	5.5939*E* + 00	2.7791*E* + 01
CDLOBA	2.4702*E* − 04	7.8071*E* − 05	1.6388*E* + 04	8.9692*E* + 04
	F3		F4	
	Avg	Std	Avg	Std
ASCA	3.3350*E* − 04	7.6138*E* − 04	1.1417*E* − 04	1.2780*E* − 04
OBSCA	**7.2337E − 25**	**3.5450E − 24**	**9.6689E − 23**	**3.3562E − 22**
ASCA_PSO	2.1223*E* + 02	5.4968*E* + 01	3.4817*E* + 00	2.8767*E* − 01
m_SCA	1.9296*E* − 08	1.0536*E* − 07	3.2717*E* − 12	1.5171*E* − 11
CBA	1.1836*E* + 01	6.7681*E* + 00	1.0994*E* + 01	1.0937*E* + 01
RCBA	2.1303*E* + 00	6.1352*E* − 01	1.9141*E* − 01	5.6666*E* − 02
CDLOBA	7.4984*E* − 04	3.2778*E* − 04	4.0013*E* + 01	8.6534*E* + 00
	F5		F6	
	Avg	Std	Avg	Std
ASCA	**2.7199E − 06**	**3.7612E − 06**	4.0769*E* − 07	8.5150*E* − 07
OBSCA	2.7751*E* + 01	5.1041*E* − 01	3.9138*E* + 00	2.5900*E* − 01
ASCA_PSO	6.2579*E* + 04	1.5907*E* + 04	8.0500*E* + 01	8.1493*E* + 00
m_SCA	2.7511*E* + 01	9.1015*E* − 01	2.4146*E* + 00	4.0044*E* − 01
CBA	5.3925*E* + 01	6.0816*E* + 01	**2.5427E − 08**	**1.2786E − 07**
RCBA	5.8979*E* + 01	8.5793*E* + 01	9.3051*E* − 03	3.4449*E* − 03
CDLOBA	8.0166*E* + 01	1.1057*E* + 02	2.3946*E* − 04	5.3245*E* − 05
	F7		F8	
	Avg	Std	Avg	Std
ASCA	**1.3299E − 04**	**1.0637E − 04**	**−1.2569E + 04**	**1.9050E − 06**
OBSCA	9.3437*E* − 04	8.1762*E* − 04	−4.1677*E* + 03	2.6656*E* + 02
ASCA_PSO	3.2258*E* + 01	6.0570*E* + 00	−7.0385*E* + 03	6.0790*E* + 02
m_SCA	4.6963*E* − 04	3.6739*E* − 04	−6.2606*E* + 03	6.8451*E* + 02
CBA	1.3568*E* − 01	2.2997*E* − 01	−7.0517*E* + 03	7.1704*E* + 02
RCBA	1.0233*E* − 01	3.5293*E* − 02	−7.1558*E* + 03	9.4342*E* + 02
CDLOBA	3.1067*E* − 02	3.2849*E* − 02	−7.2357*E* + 03	6.7461*E* + 02
	F9		F10	
	Avg	Std	Avg	Std
ASCA	1.4758*E* − 07	6.2532*E* − 07	7.1093*E* − 05	2.8806*E* − 04
OBSCA	**0.0000E + 00**	**0.0000E + 00**	**4.3225E − 15**	**6.4863E − 16**
ASCA_PSO	2.7331*E* + 02	3.3538*E* + 01	7.3024*E* + 00	2.7408*E* − 01
m_SCA	1.2208*E* + 00	5.7124*E* + 00	4.0581*E* + 00	8.1615*E* + 00
CBA	1.2947 *E* + 02	4.0483*E* + 01	1.5717*E* + 01	1.8215*E* + 00
RCBA	1.9549*E* + 01	3.6705*E* + 00	9.9456*E* − 02	1.7839*E* − 02
CDLOBA	1.3011*E* + 02	5.8979*E* + 01	1.9450*E* + 01	8.2782*E* − 01
	F11		F12	
	Avg	Std	Avg	Std
ASCA	3.7007*E* − 18	2.0270*E* − 17	**1.9635E − 09**	**3.5719E − 09**
OBSCA	**0.0000E + 00**	**0.0000E + 00**	4.0232*E* − 01	9.6078*E* − 02
ASCA_PSO	9.9972*E* − 01	2.2768*E* − 02	3.2080*E* + 00	4.6941*E* − 01
m_SCA	**0.0000E + 00**	**0.0000E + 00**	2.0154*E* − 01	1.0942*E* − 01
CBA	1.8527*E* − 02	2.6884*E* − 02	1.9634*E* + 01	6.7241*E* + 00
RCBA	1.3771*E* − 02	9.6930*E* − 03	8.1621*E* + 00	3.3077*E* + 00
CDLOBA	9.1105*E* + 01	6.3381*E* + 01	1.6905*E* + 01	5.7779*E* + 00

Table 7: Continued.

	F13		F14	
	Avg	Std	Avg	Std
ASCA	**3.9238E − 08**	**7.0206E − 08**	**2.5000E + 03**	**3.2800E − 02**
OBSCA	2.1999E + 00	1.0340E − 01	2.6926E + 03	1.6858E + 01
ASCA_PSO	1.2424E + 01	1.6305E + 00	2.6257E + 03	7.9106E + 00
m_SCA	1.6437E + 00	1.5198E − 01	2.6479E + 03	1.3218E + 01
CBA	2.1222E + 01	2.9486E + 01	2.6158E + 03	2.1610E − 01
RCBA	4.4221E − 03	3.5905E − 03	2.6153E + 03	5.1459E − 03
CDLOBA	3.5553E + 01	1.4040E + 01	2.6163E + 03	3.5415E + 00
	F15		F16	
	Avg	Std	Avg	Std
ASCA	2.6001E + 03	6.4178E − 02	**2.7000E + 03**	**3.3409E − 04**
OBSCA	**2.6000E + 03**	**4.3959E − 04**	2.7000E + 03	5.3531E − 02
ASCA_PSO	2.6358E + 03	7.0313E + 00	2.7119E + 03	5.9507E + 00
m_SCA	2.6000E + 03	1.3318E − 02	2.7134E + 03	7.0792E + 00
CBA	2.6850E + 03	3.6844E + 01	2.7290E + 03	1.1916E + 01
RCBA	2.6742E + 03	2.5890E + 01	2.7324E + 03	1.6308E + 01
CDLOBA	2.7092E + 03	6.1610E + 01	2.7216E + 03	1.0848E + 01
	F17		F18	
	Avg	Std	Avg	Std
ASCA	2.7125E + 03	2.9656E + 01	**2.9000E + 03**	**2.4342E − 03**
OBSCA	2.7041E + 03	3.5834E − 01	3.2525E + 03	4.7594E + 01
ASCA_PSO	**2.7006E + 03**	**1.3969E − 01**	3.5571E + 03	2.1780E + 02
m_SCA	2.7011E + 03	5.3555E − 01	3.3909E + 03	2.6859E + 02
CBA	2.7108E + 03	5.6503E + 01	4.1193E + 03	3.5662E + 02
RCBA	2.7340E + 03	8.0989E + 01	3.9929E + 03	4.1781E + 02
CDLOBA	2.7146E + 03	5.7321E + 01	3.7308E + 03	4.5597E + 02
	F19		F20	
	Avg	Std	Avg	Std
ASCA	**3.0000E + 03**	**1.8913E − 03**	**1.2954E + 06**	**5.3302E + 06**
OBSCA	5.3910E + 03	3.2724E + 02	1.8430E + 07	8.1290E + 06
ASCA_PSO	4.4631E + 03	3.0397E + 02	4.7057E + 06	6.1008E + 06
m_SCA	4.3630E + 03	4.5102E + 02	5.3563E + 06	7.0461E + 06
CBA	5.3333E + 03	8.9710E + 02	4.7917E + 07	5.4737E + 07
RCBA	5.8345E + 03	1.3150E + 03	1.3852E + 07	1.2806E + 07
CDLOBA	5.2844E + 03	8.5184E + 02	1.6276E + 07	1.7310E + 07
	F21		F22	
	Avg	Std	Avg	Std
ASCA	1.7988E + 05	2.3217E + 05	3.5607E + 03	4.4763E + 02
OBSCA	4.4562E + 05	1.3802E + 05	4.3392E + 03	4.9305E + 02
ASCA_PSO	5.6350E + 04	6.7763E + 04	2.2431E + 03	5.2721E + 01
m_SCA	1.2254E + 05	8.3596E + 04	2.6892E + 03	1.8716E + 02
CBA	2.3401E + 04	3.7859E + 04	2.2219E + 03	3.8285E + 01
RCBA	**9.4427E + 03**	**2.4434E + 03**	**2.1834E + 03**	**2.8655E + 01**
CDLOBA	6.4429E + 04	1.2155E + 05	2.1964E + 03	3.3965E + 01
	F23		F24	
	Avg	Std	Avg	Std
ASCA	2.4652E + 03	2.4315E + 01	**2.5000E + 03**	**3.9715E − 05**
OBSCA	2.4872E + 03	1.9399E + 01	3.3268E + 03	6.2155E + 01
ASCA_PSO	2.4101E + 03	2.9596E + 01	3.0813E + 03	6.8496E + 01
m_SCA	**2.3770E + 03**	**3.0234E + 01**	3.0490E + 03	7.2760E + 01
CBA	2.4862E + 03	4.9832E + 01	3.5097E + 03	2.8063E + 02
RCBA	2.5005E + 03	5.2624E + 01	3.6498E + 03	3.6514E + 02
CDLOBA	2.5401E + 03	5.2788E + 01	3.4766E + 03	2.7800E + 02
	F25		F26	
	Avg	Std	Avg	Std
ASCA	**2.6000E + 03**	**1.5717E − 03**	**2.7000E + 03**	**7.9694E − 03**
OBSCA	2.6018E + 03	4.7416E + 00	2.7596E + 03	1.6331E + 02
ASCA_PSO	3.5157E + 03	2.8617E + 02	3.0009E + 03	6.2050E + 01
m_SCA	3.5312E + 03	1.4113E + 02	3.4098E + 03	2.5084E + 02
CBA	2.8617E + 03	5.7555E + 02	3.0258E + 03	3.9260E + 01
RCBA	2.9867E + 03	6.0049E + 02	3.0078E + 03	9.8780E + 01
CDLOBA	3.7130E + 03	3.9680E + 02	3.1348E + 03	8.5414E + 01

Table 7: Continued.

	F27		F28	
	Avg	Std	Avg	Std
ASCA	**2.8000E + 03**	**1.7825E − 02**	**2.9000E + 03**	**4.4560E − 02**
OBSCA	2.8047E + 03	1.7384E + 01	4.1615E + 03	1.1617E + 02
ASCA_PSO	6.4961E + 03	1.5093E + 03	3.8528E + 03	1.4620E + 02
m_SCA	6.5709E + 03	5.4529E + 02	3.9182E + 03	1.2934E + 02
CBA	4.6927E + 03	3.2016E + 03	3.9596E + 03	1.9017E + 02
RCBA	5.5345E + 03	3.7043E + 03	3.9764E + 03	2.1662E + 02
CDLOBA	9.9710E + 03	2.5509E + 03	4.0081E + 03	2.2675E + 02

	F29		F30	
	Avg	Std	Avg	Std
ASCA	**3.0000E + 03**	**2.6856E − 02**	**3.1000E + 03**	**6.8348E − 02**
OBSCA	3.7384E + 03	7.7618E + 02	3.7305E + 03	2.3955E + 02
ASCA_PSO	3.7348E + 03	7.0436E + 02	4.1063E + 03	2.1778E + 02
m_SCA	5.1443E + 03	5.2692E + 02	3.8767E + 03	2.4887E + 02
CBA	3.5152E + 03	6.1480E + 02	4.9447E + 03	4.0187E + 02
RCBA	3.6971E + 03	8.9152E + 02	4.7782E + 03	4.0004E + 02
CDLOBA	3.9040E + 03	1.0920E + 03	4.9475E + 03	3.9017E + 02

	F31		+/−/=	Avg	Rank
	Avg	Std			
ASCA	**3.9128E + 03**	**2.2213E + 03**	~	2.03	1
OBSCA	3.1705E + 06	8.1720E + 06	20/8/3	3.52	3
ASCA_PSO	8.1125E + 06	4.5686E + 06	27/3/1	4.58	5
m_SCA	2.8345E + 06	4.4910E + 06	19/8/4	3.48	2
CBA	1.8989E + 06	1.2608E + 06	26/4/1	4.65	6
RCBA	4.1213E + 05	2.4036E + 05	28/2/1	4.35	4
CDLOBA	2.7393E + 05	2.2803E + 05	29/1/1	5.35	7

Table 8: p values of the Wilcoxon test on ASCA versus other algorithms.

No.	OBSCA	ASCA_PSO	m_SCA	CBA	RCBA	CDLOBA
F1	1.7344E − 06	1.7344E − 06	1.7344E − 06	**3.7094E − 01**	1.7344E − 06	1.7344E − 06
F2	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F3	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.4839E − 03
F4	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F5	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F6	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7988E − 05	1.7344E − 06	1.7344E − 06
F7	3.1817E − 06	1.7344E − 06	4.0715E − 05	1.7344E − 06	1.7344E − 06	1.7344E − 06
F8	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F9	7.8125E − 03	1.7344E − 06	**4.1797E − 01**	1.7344E − 06	1.7344E − 06	1.7344E − 06
F10	1.7333E − 06	1.7344E − 06	**1.2044E − 01**	1.7344E − 06	1.7344E − 06	1.7344E − 06
F11	**1.0000E + 00**	1.7344E − 06	**1.0000E + 00**	3.7896E − 06	1.7344E − 06	1.7344E − 06
F12	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F13	1.7344E − 06	1.7344E − 06	1.7344E − 06	4.7292E − 06	1.7344E − 06	1.7344E − 06
F14	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F15	1.7344E − 06	1.7344E − 06	6.9838E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F16	1.6046E − 04	1.7344E − 06	7.6909E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F17	2.7653E − 03	1.7344E − 06	2.3534E − 06	3.1123E − 05	**1.0201E − 01**	6.4242E − 03
F18	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F19	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F20	4.7292E − 06	1.1499E − 04	1.1499E − 04	1.7344E − 06	2.3704E − 05	1.4936E − 05
F21	1.3595E − 04	**1.0639E − 01**	**2.9894E − 01**	4.7162E − 02	4.7162E − 02	**1.5886E − 01**
F22	2.1266E − 06	1.7344E − 06	1.9209E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F23	1.1138E − 03	1.2381E − 05	1.9209E − 06	3.5009E − 02	4.1140E − 03	6.9838E − 06
F24	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F25	3.2541E − 05	1.7344E − 06	1.7344E − 06	3.4053E − 05	1.7344E − 06	1.7344E − 06
F26	**7.4987E − 01**	1.7344E − 06	2.3534E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F27	**6.8836E − 01**	1.7344E − 06	1.7344E − 06	2.3038E − 02	1.7344E − 06	1.7344E − 06
F28	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F29	3.1817E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F30	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F31	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06

Table 9: Running time of ASCA and other algorithms.

F	ASCA	OBSCA	ASCA_PSO	m_SCA	CBA	RCBA	CDLOBA
F1	1.23*E* + 02	**7.56E + 01**	1.64*E* + 03	1.35*E* + 02	1.98*E* + 02	2.24*E* + 02	1.46*E* + 02
F2	1.34*E* + 02	**8.75E + 01**	1.66*E* + 03	1.47*E* + 02	2.11*E* + 02	2.37*E* + 02	1.66*E* + 02
F3	6.29*E* + 02	**5.82E + 02**	2.16*E* + 03	6.86*E* + 02	7.16*E* + 02	7.39*E* + 02	9.07*E* + 02
F4	1.42*E* + 02	**9.42E + 01**	1.68*E* + 03	1.53*E* + 02	2.25*E* + 02	2.49*E* + 02	1.79*E* + 02
F5	1.70*E* + 02	**1.24E + 02**	1.70*E* + 03	1.90*E* + 02	2.56*E* + 02	2.81*E* + 02	2.24*E* + 02
F6	1.45*E* + 02	**9.91E + 01**	1.67*E* + 03	1.63*E* + 02	2.34*E* + 02	2.60*E* + 02	1.91*E* + 02
F7	2.13*E* + 02	**1.68E + 02**	1.74*E* + 03	2.39*E* + 02	3.07*E* + 02	3.34*E* + 02	2.99*E* + 02
F8	1.61*E* + 02	**1.19E + 02**	1.69*E* + 03	1.83*E* + 02	2.51*E* + 02	2.77*E* + 02	2.13*E* + 02
F9	1.64*E* + 02	**1.22E + 02**	1.69*E* + 03	1.85*E* + 02	2.59*E* + 02	2.84*E* + 02	2.29*E* + 02
F10	1.75*E* + 02	**1.30E + 02**	1.69*E* + 03	2.02*E* + 02	2.67*E* + 02	2.91*E* + 02	2.46*E* + 02
F11	1.85*E* + 02	**1.40E + 02**	1.70*E* + 03	2.05*E* + 02	2.75*E* + 02	2.97*E* + 02	2.54*E* + 02
F12	3.59*E* + 02	**3.15E + 02**	1.87*E* + 03	3.96*E* + 02	4.50*E* + 02	4.73*E* + 02	5.07*E* + 02
F13	3.58*E* + 02	**3.13E + 02**	1.87*E* + 03	3.94*E* + 02	4.51*E* + 02	4.74*E* + 02	5.10*E* + 02
F14	3.72*E* + 02	**3.26E + 02**	1.92*E* + 03	4.22*E* + 02	4.69*E* + 02	5.02*E* + 02	5.17*E* + 02
F15	3.22*E* + 02	**2.74E + 02**	1.87*E* + 03	3.66*E* + 02	4.18*E* + 02	4.46*E* + 02	4.44*E* + 02
F16	3.47*E* + 02	**3.01E + 02**	1.89*E* + 03	4.00*E* + 02	4.40*E* + 02	4.70*E* + 02	4.81*E* + 02
F17	1.16*E* + 03	**1.12E + 03**	2.72*E* + 03	1.29*E* + 03	1.27*E* + 03	1.30*E* + 03	1.71*E* + 03
F18	1.12*E* + 03	**1.11E + 03**	2.70*E* + 03	1.29*E* + 03	1.28*E* + 03	1.30*E* + 03	1.70*E* + 03
F19	4.12*E* + 02	**3.77E + 02**	1.97*E* + 03	4.79*E* + 02	5.21*E* + 02	5.52*E* + 02	5.93*E* + 02
F20	4.66*E* + 02	**4.34E + 02**	2.03*E* + 03	5.39*E* + 02	5.69*E* + 02	6.05*E* + 02	6.74*E* + 02
F21	3.47*E* + 02	**3.01E + 02**	1.89*E* + 03	3.95*E* + 02	4.42*E* + 02	4.72 *E* + 02	4.76*E* + 02
F22	3.98*E* + 02	**3.42E + 02**	1.94*E* + 03	4.42*E* + 02	4.96*E* + 02	5.21*E* + 02	5.42*E* + 02
F23	4.33*E* + 02	**3.74E + 02**	1.98*E* + 03	4.82*E* + 02	5.28*E* + 02	5.55*E* + 02	5.90*E* + 02
F24	4.58*E* + 02	**4.00E + 02**	2.00*E* + 03	5.08*E* + 02	5.57*E* + 02	5.86*E* + 02	6.29*E* + 02
F25	4.43*E* + 02	**3.87E + 02**	1.98*E* + 03	4.94*E* + 02	5.42*E* + 02	5.75*E* + 02	6.08*E* + 02
F26	4.49*E* + 02	**3.91E + 02**	1.99*E* + 03	4.97*E* + 02	5.44*E* + 02	5.76*E* + 02	6.14*E* + 02
F27	5.04*E* + 02	**4.45E + 02**	2.05*E* + 03	5.59*E* + 02	6.00*E* + 02	6.26*E* + 02	6.93*E* + 02
F28	5.36*E* + 02	**4.80E + 02**	2.09*E* + 03	5.94*E* + 02	6.36*E* + 02	6.64*E* + 02	7.46*E* + 02
F29	4.95*E* + 02	**4.36E + 02**	2.04*E* + 03	5.48*E* + 02	5.94*E* + 02	6.20*E* + 02	6.80*E* + 02
F30	4.34*E* + 02	**3.75E + 02**	1.99*E* + 03	4.80*E* + 02	5.32*E* + 02	5.60*E* + 02	5.93*E* + 02
F31	5.52*E* + 02	**4.95E + 02**	2.10*E* + 03	6.10*E* + 02	6.59*E* + 02	6.82*E* + 02	7.75*E* + 02

fourth column show the average fitness of ASCA, and the graphs of the fifth column show the convergence curve of the algorithms.

Figures 6(a) and 6(b) record the location and distribution of individuals in each iteration. In Figure 6(b), we can see that most of the search locations are around the optimal solution, and a small number of locations are scattered throughout the space. It shows that ASCA can be developed in the target area after searching most of the space. In Figure 6(c), we can see that the curve fluctuates significantly in the early stage, indicating that ASCA has good search ability and can traverse the space as much as possible. Figure 6(d) shows the change in average fitness. The figure shows that the curve fluctuates greatly but drops rapidly, indicating that ASCA has good convergence ability. Figure 6(e) shows that ASCA can find the optimal solution faster than the other three test algorithms.

For further exploration, we analyze the balance and diversity of these four algorithms on the 31 functions in this paper. Figure 7 shows the balance analysis of the four algorithms. We have added an incremental-decremental curve to the figure. It can be seen from the figure that when the value of the exploration effect is greater than or equal to the exploitation effect, it is an increment. Conversely, it is a decrement. When the value is negative, it is set to zero. Therefore, its high value represents a wide range of exploration activities, while its low value represents a strong exploitation effect. Similarly, the duration of high or low values in the graph reflects the continuing effect of exploration or exploitation in the search strategy. When the exploration and exploitation effects are at the same level, the incremental-decremental curve maximizes.

As can be seen from the figure, the search strategy is exploitation most of the time to get a better solution. The exploration effect is always very short compared with the exploitation effect. We can see that as the function starts to become more complex, SCA will take longer to explore, which instructions its poor exploitation capability. From the diagram of ADSCA, it can be seen that the addition of adaptive r_1 can effectively balance the exploration and exploitation phases of SCA. Equilibrium is the result of the search mechanism used by each meta-heuristic scheme. ASCA and CLSSCA have similar search mechanisms so that the equilibrium response is similar to the graph. Numerically, the exploitation rate of ASCA is slightly higher than that of CLSSCA.

Figure 8 displays the evolution of the diversity of the algorithms during the optimization procedure. The X-axis represents the number of iterations and the Y-axis represents the diversity measurement. We can clearly see that all algorithms start with vast diversity due to random initialization. As the number of iterations increases, the population diversity decreases gradually. The curve of

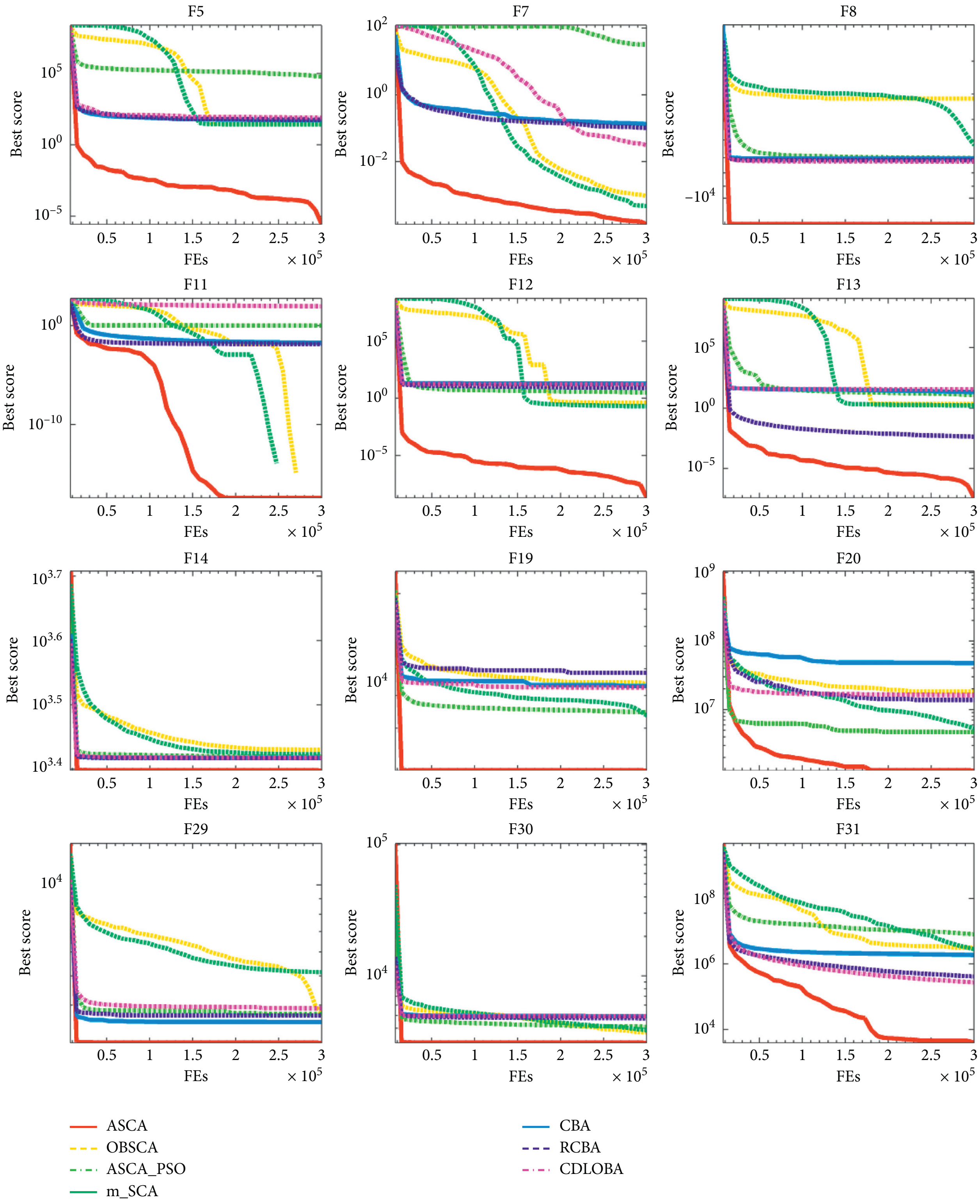

Figure 4: Convergence curves of ASCA and other variant algorithms (first row: F5, F7, and F8; second row: F11, F12, and F13; third row: F14, F19, and F20; forth row: F29, F30, and F31).

ASCA in the figure drops the fastest. The fast convergence of diversity indicates that this algorithm enters the exploitation stage earlier than other algorithms. It can intuitively see that the exploitation capability of ASCA is better than the other three algorithms.

4.7. Statistical Analysis of ASCA. In this section, a statistical analysis was performed to validate the performance of ASCA. We first use the nonparametric Friedman test to see if the performance difference between all the algorithms in the benchmark function is statistically significant. Friedman's test

Table 10: Comparison results of ASCA and other algorithms.

	F1		F2	
	Avg	Std	Avg	Std
ASCA	**2.5312E + 08**	**5.9606E + 07**	**1.8921E + 10**	**3.0216E + 09**
OBSCA	4.2757E + 08	1.0267E + 08	2.5650E + 10	3.1035E + 09
SCADE	4.6147E + 08	9.2850E + 07	2.9440E + 10	4.5661E + 09
CESCA	1.3401E + 09	1.7893E + 08	7.4616E + 10	5.0732E + 09
CLSCA	4.1679E + 08	1.0092E + 08	3.1822E + 10	4.8916E + 09
	F3		F4	
	Avg	Std	Avg	Std
ASCA	**3.8658E + 04**	**5.2741E + 03**	**1.5309E + 03**	**2.8468E + 02**
OBSCA	5.1818E + 04	7.4341E + 03	2.4250E + 03	7.2627E + 02
SCADE	5.5419E + 04	5.6944E + 03	2.3593E + 03	7.3631E + 02
CESCA	1.0286E + 05	1.3834E + 04	1.1961E + 04	1.6839E + 03
CLSCA	4.5075E + 04	8.8290E + 03	2.4739E + 03	8.2932E + 02
	F5		F6	
	Avg	Std	Avg	Std
ASCA	5.2095E + 02	4.2232E − 02	6.3407E + 02	3.0317E + 00
OBSCA	5.2095E + 02	5.6337E − 02	6.3258E + 02	1.4605E + 00
SCADE	**5.2094E + 02**	**5.2414E − 02**	6.3451E + 02	2.1178E + 00
CESCA	5.2105E + 02	3.8741E − 02	6.4202E + 02	1.6327E + 00
CLSCA	5.2094E + 02	6.6109E − 02	**6.3221E + 02**	**2.1307E + 00**
	F7		F8	
	Avg	Std	Avg	Std
ASCA	**8.5830E + 02**	**3.5557E + 01**	**1.0523E + 03**	**1.5940E + 01**
OBSCA	9.1767E + 02	4.1763E + 01	1.0610E + 03	1.8518E + 01
SCADE	9.0492E + 02	2.7044E + 01	1.0722E + 03	1.3982E + 01
CESCA	1.4303E + 03	4.6110E + 01	1.2145E + 03	1.1594E + 01
CLSCA	9.3798E + 02	4.3327E + 01	1.0674E + 03	2.0630E + 01
	F9		F10	
	Avg	Std	Avg	Std
ASCA	1.1828E + 03	1.5176E + 01	7.0429E + 03	5.4427E + 02
OBSCA	1.2037E + 03	1.7048E + 01	**6.2169E + 03**	**4.3813E + 02**
SCADE	1.2046E + 03	1.8408E + 01	7.4213E + 03	3.4278E + 02
CESCA	1.2961E + 03	1.8675E + 01	8.8241E + 03	3.3121E + 02
CLSCA	**1.1751E + 03**	**1.7691E + 01**	6.8660E + 03	5.1284E + 02
	F11		F12	
	Avg	Std	Avg	Std
ASCA	8.1091E + 03	3.0177E + 02	1.2026E + 03	2.1750E − 01
OBSCA	**7.3103E + 03**	**3.6428E + 02**	**1.2023E + 03**	**3.0055E − 01**
SCADE	8.2168E + 03	2.7287E + 02	1.2025E + 03	2.4713E − 01
CESCA	9.2501E + 03	1.6997E + 02	1.2035E + 03	2.9109E − 01
CLSCA	8.0448E + 03	4.9974E + 02	1.2025E + 03	2.9523E − 01
	F13		F14	
	Avg	Std	Avg	Std
ASCA	**1.3031E + 03**	**2.5591E − 01**	**1.4524E + 03**	**8.4244E + 00**
OBSCA	1.3037E + 03	3.7164E − 01	1.4673E + 03	1.0614E + 01
SCADE	1.3040E + 03	2.6825E − 01	1.4886E + 03	1.2394E + 01
CESCA	1.3078E + 03	3.9944E − 01	1.6502E + 03	1.9751E + 01
CLSCA	1.3040E + 03	3.1713E − 01	1.4794E + 03	1.6072E + 01
	F15		F16	
	Avg	Std	Avg	Std
ASCA	**5.8187E + 03**	**3.9040E + 03**	**1.6127E + 03**	**2.5255E − 01**
OBSCA	1.6315E + 04	8.1576E + 03	1.6130E + 03	1.8896E − 01
SCADE	1.9286E + 04	8.5848E + 03	1.6127E + 03	1.2646E − 01
CESCA	4.6774E + 05	1.3148E + 05	1.6136E + 03	1.9078E − 01
CLSCA	1.6199E + 04	7.0656E + 03	1.6129E + 03	2.3468E − 01

Table 10: Continued.

	F17		F18	
	Avg	Std	Avg	Std
ASCA	**7.6886E + 06**	**3.5825E + 06**	1.7391E + 08	1.2640E + 08
OBSCA	1.1241E + 07	4.7696E + 06	1.6232E + 08	1.0488E + 08
SCADE	1.4396E + 07	5.5321E + 06	1.7135E + 08	1.4407E + 08
CESCA	8.3783E + 07	2.9877E + 07	4.2993E + 09	1.2665E + 09
CLSCA	9.7934E + 06	5.4166E + 06	**1.4934E + 08**	**5.3713E + 07**
	F19		F20	
	Avg	Std	Avg	Std
ASCA	**1.9928E + 03**	**1.4945E + 01**	**1.8103E + 04**	**5.2822E + 03**
OBSCA	2.0077E + 03	1.7269E + 01	2.7532E + 04	1.0141E + 04
SCADE	2.0150E + 03	1.4599E + 01	2.8080E + 04	1.0159E + 04
CESCA	2.2786E + 03	6.6103E + 01	3.9817E + 05	2.6749E + 05
CLSCA	2.0255E + 03	2.3988E + 01	1.9504E + 04	8.9409E + 03
	F21		F22	
	Avg	Std	Avg	Std
ASCA	**1.4089E + 06**	**8.3687E + 05**	**2.9955E + 03**	**1.4914E + 02**
OBSCA	2.0254E + 06	1.0773E + 06	3.1347E + 03	2.0435E + 02
SCADE	2.1898E + 06	9.9802E + 05	3.1097E + 03	1.3481E + 02
CESCA	3.7127E + 07	1.5830E + 07	6.1738E + 03	2.7847E + 03
CLSCA	2.2785E + 06	1.6796E + 06	3.1145E + 03	1.5020E + 02
	F23		F24	
	Avg	Std	Avg	Std
ASCA	2.5000E + 03	3.0116E − 03	2.6001E + 03	6.1360E − 02
OBSCA	2.6900E + 03	2.0732E + 01	2.6000E + 03	3.3545E − 04
SCADE	**2.5000E + 03**	**0.0000E + 00**	2.6000E + 03	1.9364E − 06
CESCA	3.1116E + 03	1.0729E + 02	2.6575E + 03	1.8678E + 01
CLSCA	**2.5000E + 03**	**0.0000E + 00**	**2.6000E + 03**	**0.0000E + 00**
	F25		F26	
	Avg	Std	Avg	Std
ASCA	2.7000E + 03	2.0887E − 05	2.7094E + 03	2.4639E + 01
OBSCA	2.7000E + 03	2.5958E − 04	**2.7040E + 03**	**3.8816E − 01**
SCADE	**2.7000E + 03**	**0.0000E + 00**	2.7070E + 03	1.7575E + 01
CESCA	2.7173E + 03	6.5231E + 00	2.7124E + 03	1.4793E + 00
CLSCA	**2.7000E + 03**	**0.0000E + 00**	2.7127E + 03	2.9613E + 01
	F27		F28	
	Avg	Std	Avg	Std
ASCA	2.9000E + 03	6.5255E − 04	3.0000E + 03	3.4461E − 03
OBSCA	3.2478E + 03	3.7228E + 01	5.4095E + 03	3.8395E + 02
SCADE	3.2536E + 03	2.3566E + 02	4.8912E + 03	9.9748E + 02
CESCA	4.0160E + 03	1.7301E + 02	5.4762E + 03	2.9328E + 02
CLSCA	**2.9000E + 03**	**0.0000E + 00**	**3.0000E + 03**	**0.0000E + 00**
	F29		F30	
	Avg	Std	Avg	Std
ASCA	3.9114E + 03	4.1676E + 03	1.4202E + 05	1.6281E + 05
OBSCA	1.9006E + 07	1.0051E + 07	4.4466E + 05	1.8444E + 05
SCADE	1.2517E + 07	7.8033E + 06	5.0936E + 05	1.4549E + 05
CESCA	1.7552E + 07	3.5750E + 06	1.4698E + 06	2.5334E + 05
CLSCA	**3.1000E + 03**	**0.0000E + 00**	**3.2000E + 03**	**0.0000E + 00**

requires a certain number of algorithms and functions to be compared. At least more than 10 benchmark functions and more than 5 algorithms are required to participate in the test. The Friedman statistic requires calculating the mean ranked value. For the best out of k algorithms on the ith function, rank 1 is assigned; for the second best, rank 2 is assigned. After that, the average ranking of an algorithm is computed. Under the null hypothesis, which states that all the algorithms are equivalent, their ranks should be equal. To see whether the null hypothesis is rejected, a comparison is needed to review the critical values obtained for the significance level ($a = 0.05$ and 0.1) with Friedman statistics. The formula and explanations can

Table 11: p values of the Wilcoxon test on ASCA versus SCA variants.

	OBSCA	SCADE	CESCA	CLSCA
F1	4.7292E − 06	1.7344E − 06	1.7344E − 06	2.1630E − 05
F2	3.5152E − 06	2.6033E − 06	1.7344E − 06	1.9209E − 06
F3	2.8786E − 06	1.7344E − 06	1.7344E − 06	1.4839E − 03
F4	2.1266E − 06	5.2165E − 06	1.7344E − 06	2.8786E − 06
F5	**8.9364E − 01**	**6.5833E − 01**	1.9209E − 06	**6.2884E − 01**
F6	1.9569E − 02	**6.1431E − 01**	1.9209E − 06	6.0350E − 03
F7	3.7243E − 05	6.8923E − 05	1.7344E − 06	3.5152E − 06
F8	**5.1931E − 02**	5.7924E − 05	1.7344E − 06	3.0010E − 02
F9	2.0515E − 04	1.2506E − 04	1.7344E − 06	**1.3591E − 01**
F10	1.3601E − 05	2.7653E − 03	1.7344E − 06	**1.4704E − 01**
F11	6.3391E − 06	**2.1336E − 01**	1.7344E − 06	**5.1705E − 01**
F12	1.3194E − 02	**2.0589E − 01**	1.7344E − 06	**7.3433E − 01**
F13	2.8786E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F14	1.4936E − 05	1.7344E − 06	1.7344E − 06	3.1817E − 06
F15	5.7517E − 06	2.8786E − 06	1.7344E − 06	3.5152E − 06
F16	1.0246E − 05	**7.1889E − 01**	1.7344E − 06	4.6818E − 03
F17	1.4795E − 02	9.7110E − 05	1.7344E − 06	**8.9718E − 02**
F18	**7.9710E − 01**	**8.4508E − 01**	1.7344E − 06	**7.8126E − 01**
F19	3.3789E − 03	1.9729E − 05	1.7344E − 06	1.1265E − 05
F20	3.1123E − 05	2.1630E − 05	1.7344E − 06	**5.8571E − 01**
F21	5.6672E − 03	7.1570E − 04	1.7344E − 06	4.1140E − 03
F22	9.6266E − 04	8.2167E − 03	1.7344E − 06	2.2551E − 03
F23	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F24	1.7344E − 06	1.7344E − 06	1.7344E − 06	1.7344E − 06
F25	1.0424E − 02	1.7116E − 06	1.7344E − 06	1.7116E − 06
F26	3.5888E − 04	5.2872E − 04	3.5888E − 04	**4.2843E − 01**
F27	1.7344E − 06	2.5967E − 05	1.7344E − 06	1.7289E − 06
F28	1.7344E − 06	1.3601E − 05	1.7344E − 06	2.5614E − 06
F29	1.7344E − 06	2.3534E − 06	1.7344E − 06	1.7344E − 06
F30	4.7292E − 06	3.1817E − 06	1.7344E − 06	1.7344E − 06

be found in [87]. If the null hypothesis is Friedman rejected in tests, we can continue with Bonferroni–Dunn's test.

In this paper, three groups of comparative experiments were conducted firstly. For all three groups of experimental results, the null hypothesis was rejected. We can continue with Bonferroni–Dunn's test to them. In the Bonferroni–Dunn's test, the quality of two algorithms is significantly different if the corresponding average of rankings is at least as great as its critical difference (CD). The formula for the CD is as follows:

$$\mathrm{CD} = q_{\alpha}\sqrt{\frac{k(k+1)}{6N}}, \tag{8}$$

where q_a is the critical value for multiple nonparametric comparisons, k is the number of algorithms, and N is the number of functions.

The first groups of experiments are ASCA in comparison with conventional algorithms. In this experiment, nine algorithms were used, so k is equal to 9. Thirty-one functions were selected for the experiment, so N is equal to 31. The significance level was selected as 0.05 and 0.1, respectively, and then the CD was calculated. The calculated CD is 1.89 when $a = 0.05$. When $a = 0.1$, the CD is 1.73. Figure 9 records the results of the Bonferroni–Dunn test for the first groups of experiments. The y-axis in the figure represents the average ranking of the algorithm. To compare the difference between ASCA and other algorithms, ASCA is set as the control algorithm. The horizontal lines in the figure represent the threshold of the ASCA. The dotted line in the figure represents the threshold when the significance level is 0.05, and the solid line represents the threshold when the significance level is 0.1. When the average ranking of other algorithms is higher than this threshold, there is a significant difference between this algorithm and ASCA. From Figure 9, we can see that the average ranking of ASCA is lowest at 2.39. The performance of ASCA was significantly better than MFO, BA, GSA, SCA, FA, and PSO at both significance levels.

We performed the same Bonferroni–Dunn for the second and third experiments. Figure 10 shows the results of the Bonferroni–Dunn test for the second group of experiments. In this experiment, ASCA is compared with some advanced algorithms. 7 algorithms and 31 functions are involved in the comparison, so $k = 7$ and $N = 31$. The calculated CD is 1.45 when $a = 0.05$. When $a = 0.1$, the CD is 1.31. We can see from the figure that the performance of ASCA was significantly better than that of OBSCA, ASCA_PSO, CBA, RCBA, and CDLOBA at both significance levels.

Figure 11 shows the results of the Bonferroni–Dunn test for the third group of experiments. In this experiment, ASCA compares the SCA variants to CEC14. 5 algorithms and 30 functions are involved, so $k = 5$ and $N = 30$. The calculated CD is 1.01 when $a = 0.05$. When $a = 0.1$, the CD is 0.91. We can see from the figure that the performance of

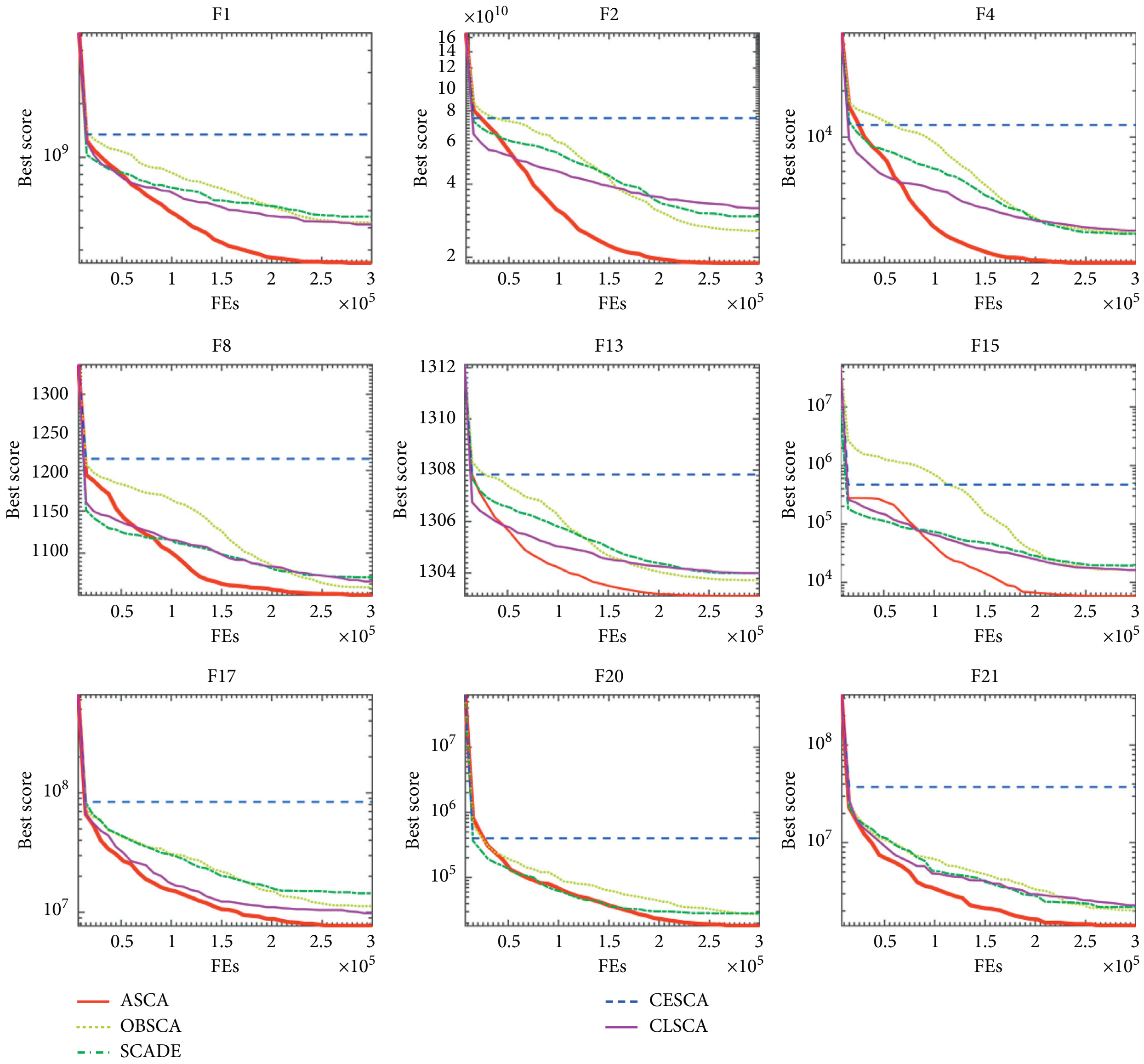

Figure 5: Convergence curves of ASCA and other variant algorithms (first row: F1, F2, and F4; second row: F8, F13, and F15; third row: F17, F20, and F21).

ASCA was significantly better than SCADE and CESCA at the significance level of 0.1.

4.8. Application to Engineering Benchmarks. In this section, the effectiveness of ASCA was further validated on the following engineering problems: tension-compression spring problem (TCSD), pressure vessel design (PVD), welded beam design (WBD), and rolling element bearing design problem (RED).

4.8.1. TCSD Problem. The goal of this engineering problem is to minimize the weight of the tension-compression string. In this engineering problem, the variables are wire diameter (d), mean coil diameter (D), and the number of active coils (N). The mathematical model of the problem is as follows:

$$\begin{aligned}
&\text{consider } \vec{x} = [x_1 \;\; x_2 \;\; x_3] = [d \;\; D \;\; N]ta,\\
&\text{minimize } f(\vec{x}) = x_1^2 x_2 (x_3 + 2),\\
&\text{subject to} \quad g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0,\\
&\qquad g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} - 1 \le 0,\\
&\qquad g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0,\\
&\qquad g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0,\\
&\text{variable range } 0.05 \le x_1 \le 2, 0.25 \le x_2 \le 1.3, 2 \le x_3 \le 15.
\end{aligned} \tag{9}$$

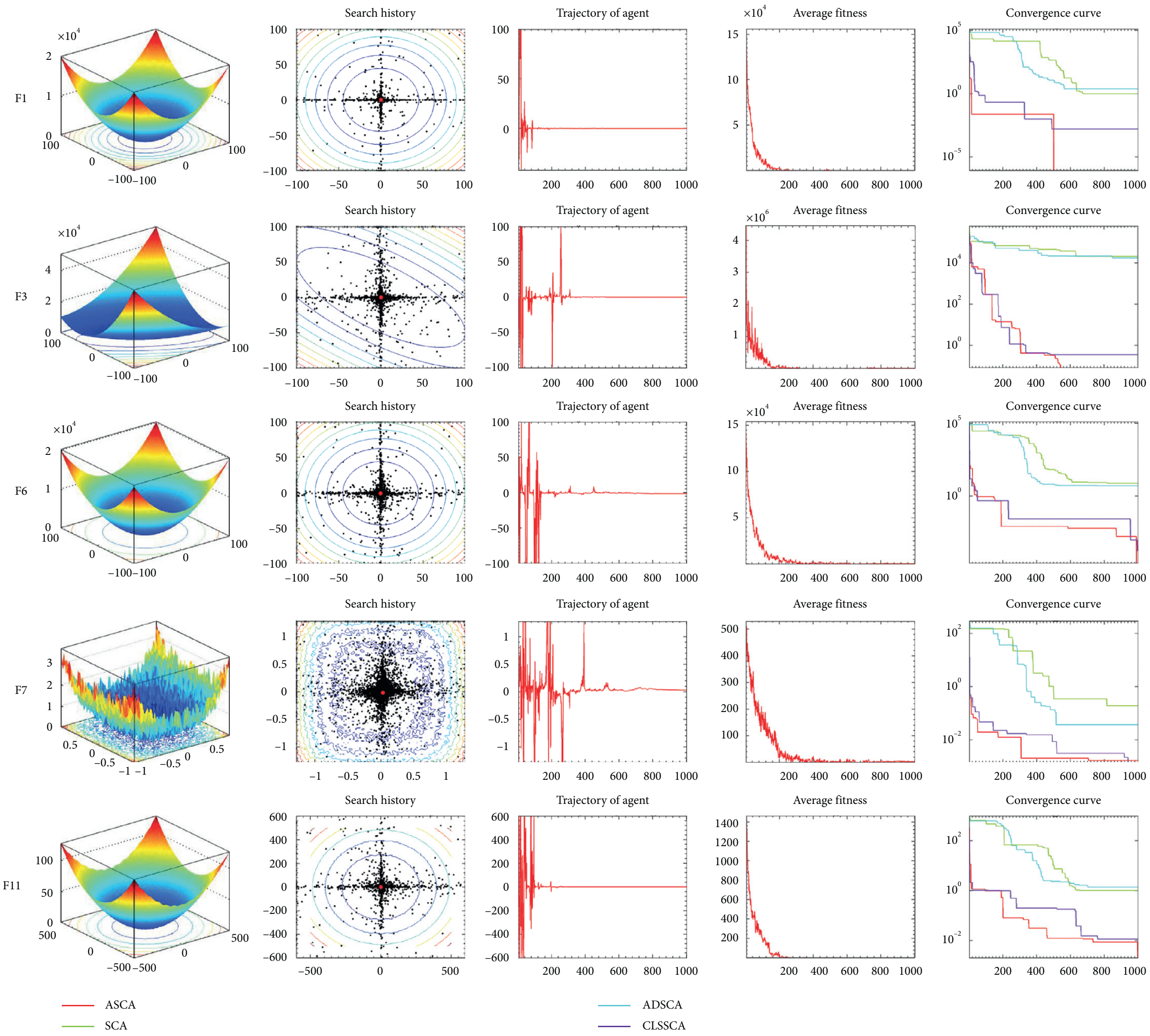

Figure 6: (a) Three-dimensional location distribution of ASCA, (b) two-dimensional location distribution of ASCA, (c) trajectory of ASCA in the first dimension, (d) average fitness of ASCA, (e) and convergence curves of algorithms.

ASCA is applied to the TCSD problem, and the convergence of ASCA on this problem is recorded in Figure 12.

To verify ASCA's ability to solve this engineering problem, PSO, GSA, SCA, CGWO [88], and EPO [89] are chosen to compare with ASCA. ASCA, PSO, GSA, and SCA used 2000 iterations in this experiment. The CGWO and EPO data in the table are taken from the original papers. CGWO and EPO used 500 iterations and 1000 iterations, respectively. Table 12 records the optimal solutions of tension-compression string design problems. Table 13 records the comparison results of the involved algorithms. From the table, we can see that ASCA has made great progress on this engineering issue compared with the classic SCA. The standard deviation of ASCA in the table ranks second, indicating that ASCA is also relatively stable on this engineering issue.

4.8.2. PVD Problem. This problem aims to minimize the design cost of a cylindrical pressure vessel. One section of the container is covered, while the other end is hemispherical. Its cost is related to the four variables of thickness shell (T_s), thickness head (T_h),inner radius (R), and section range minus head (l). The mathematical model is presented as follows:

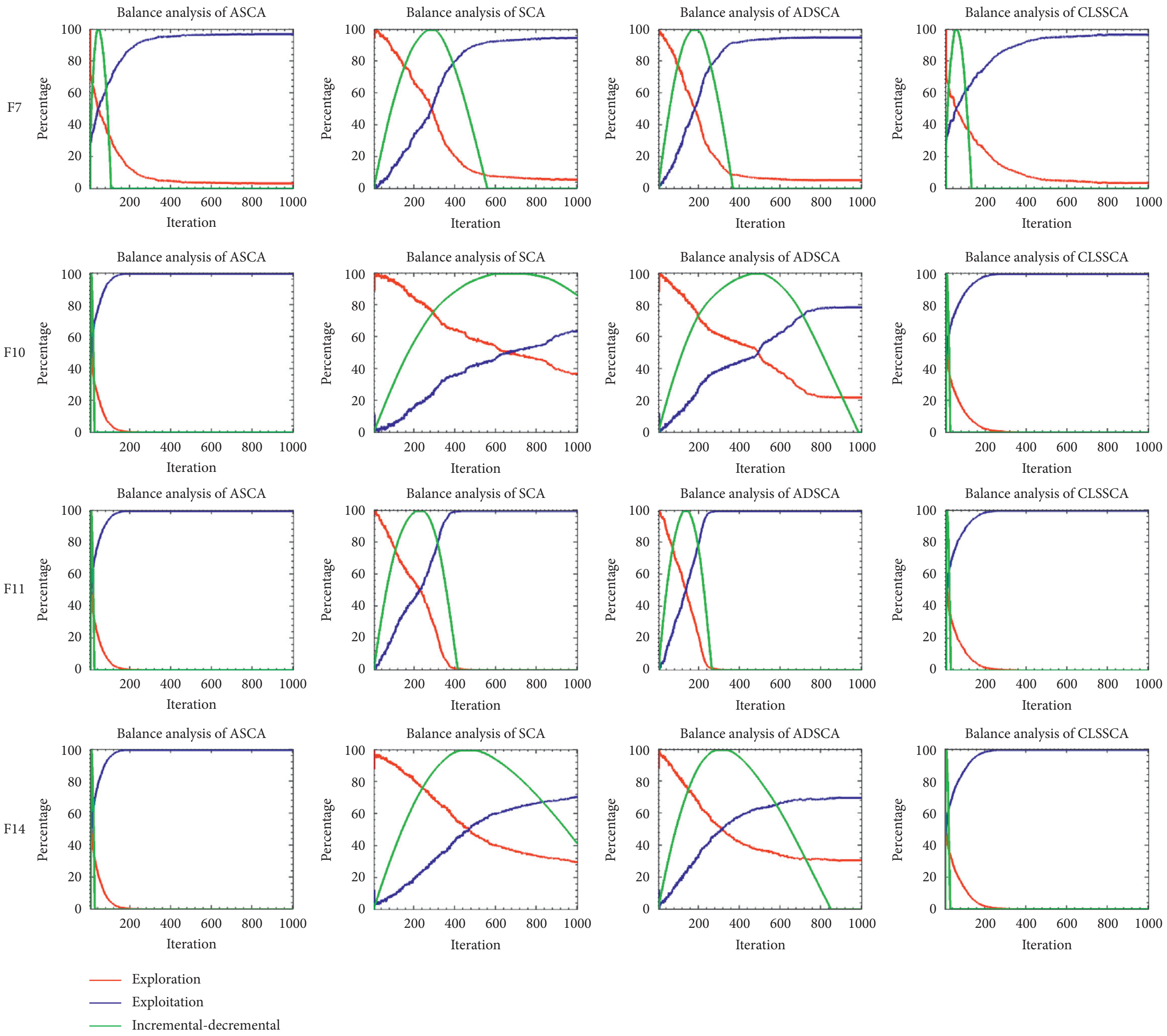

Figure 7: Balance analysis of algorithms.

$$
\begin{aligned}
&\text{consider } \vec{x} = [x_1 \;\; x_2 \;\; x_3 \;\; x_4] = [T_s \;\; T_h \;\; R \;\; L],\\
&\text{objective } f(\vec{x})_{\min} = 0.6224x_1x_3x_4 + 1.7781x_3x_1^2 + 3.1661x_4x_1^2 + 19.84x_3x_1^2,\\
&\text{subject to} \quad g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0,\\
&\qquad\qquad\quad\; g_2(\vec{x}) = -x_2 + 0.00954x_3 \le 0,\\
&\qquad\qquad\quad\; g_3(\vec{x}) = -\pi x_4 x_3^2 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0,\\
&\qquad\qquad\quad\; g_4(\vec{x}) = x_4 - 240 \le 0,\\
&\text{variable ranges } 0 \le x_1 \le 99,\, 0 \le x_2 \le 99,\, 10 \le x_3 \le 200,\, 10 \le x_4 \le 200.
\end{aligned}
\tag{10}
$$

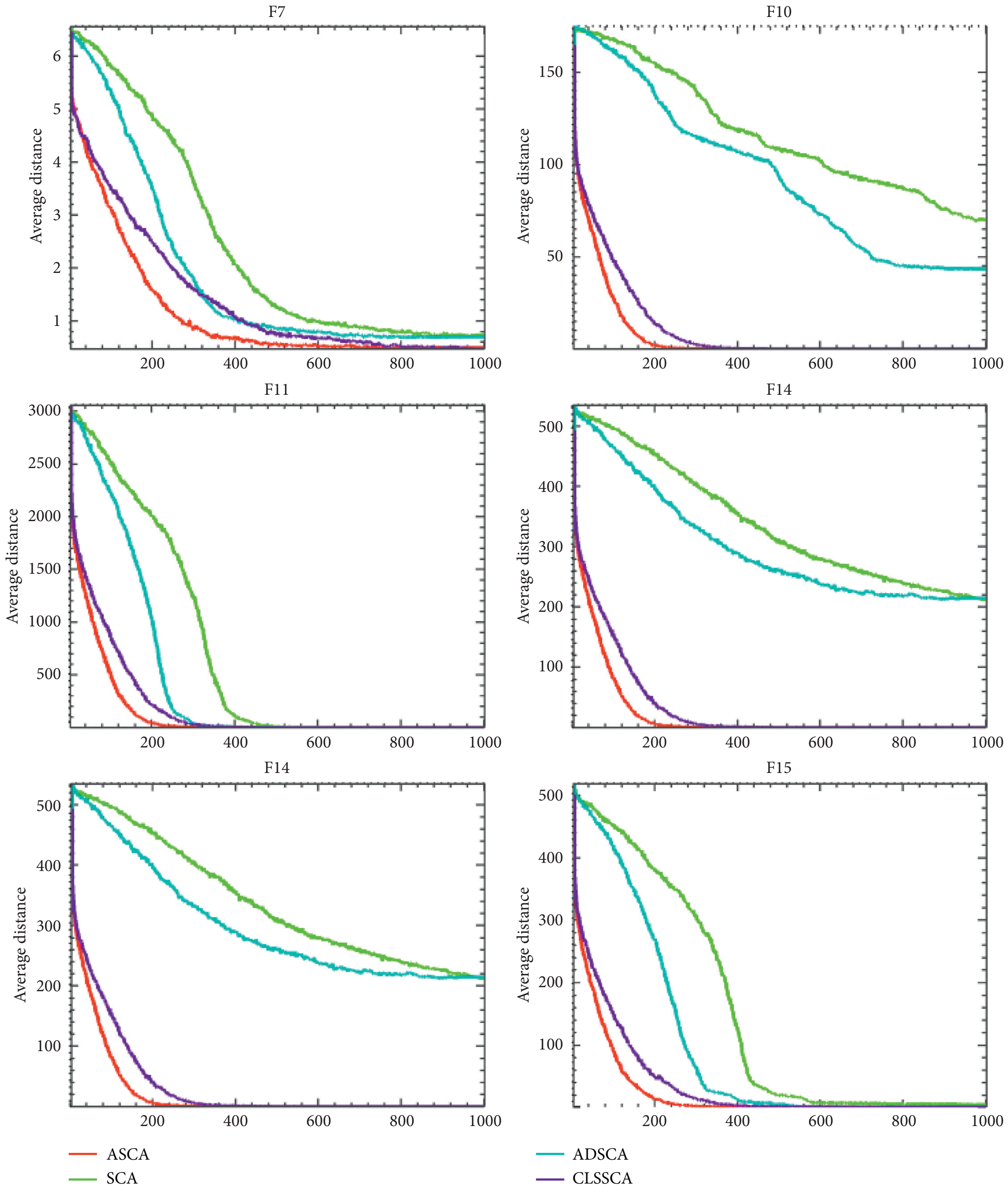

Figure 8: Diversity analysis of algorithms.

Figure 13 shows the convergence curve for ASCA on this engineering problem.

To verify ASCA's ability to solve this engineering problem, GWO, CBA, SCA, SCADE, CGWO, and EPO are chosen to compare with ASCA. ASCA, GWO, CBA, SCA, and SCADE used 2000 iterations in this experiment. The CGWO and EPO data in the table are taken from their original papers. CGWO used 500 iterations, and EPO used 1000 iterations. Table 14 records the optimal solutions of various methods to the pressure vessel design problem. Table 15 records the comparison results of involved algorithms. As seen from the table, the performance of ASCA on this problem is not the best, but it is far better than some conventional algorithms.

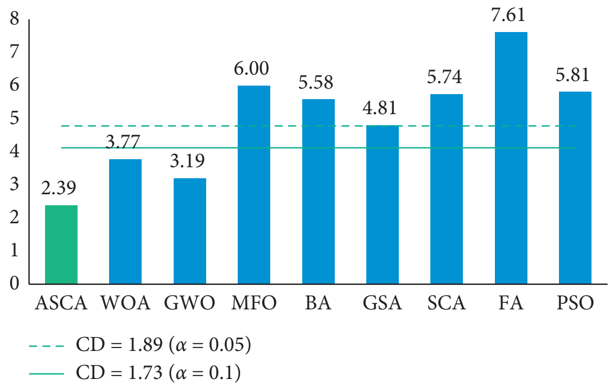

Figure 9: Bonferroni–Dunn test results of experiments in Section 4.3.

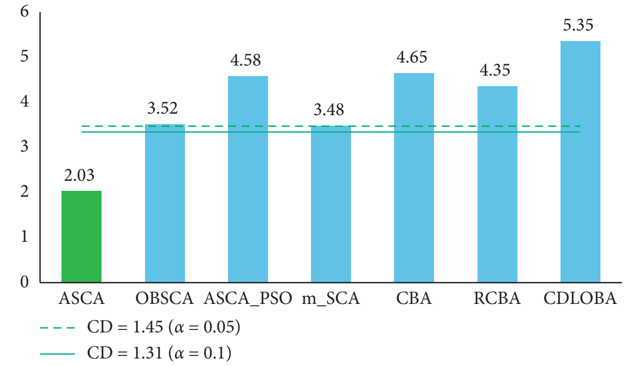

Figure 10: Bonferroni–Dunn test results of experiments in Section 4.4.

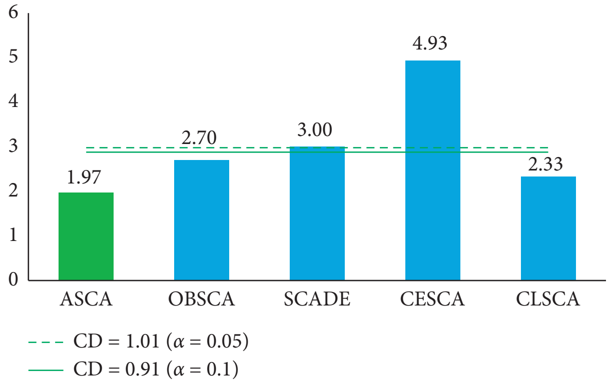

Figure 11: Bonferroni–Dunn test results of experiments in Section 4.5.

4.8.3. WBD Problem. In this problem, a welded beam is designed for minimum cost subjected to constraints. The purpose of this problem is to find the lowest cost of welded beams under the four constraints of shear stress (τ), bending stress (θ), buckling load (P_c), and deflection (δ). This problem involves the following four variables: welding seam thickness (h); welding joint length (l); beam width (t); beam thickness (b). The mathematical model is as follows:

$$\begin{aligned}
&\text{consider } \vec{x} = [x_1,\ x_2,\ x_3,\ x_4] = [h\ \ l\ \ t\ \ b],\\
&\text{minimize } f(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_4),\\
&\text{subject to}\quad g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0,\\
&\qquad g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0,\\
&\qquad g_3(\vec{x}) = x_1 - x_4 \le 0,\\
&\qquad g_4(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \le 0,\\
&\qquad g_5(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0,\\
&\qquad g_6(\vec{x}) = P - P_C(\vec{x}) \le 0,\\
&\qquad g_7(\vec{x}) = 0.125 - x_1 \le 0,\\
&\text{variable range of } 0.1 \le x_1 \le 2, 0.1 \le x_2 \le 10, 0.1 \le x_3 \le 10, 0.1 \le x_4 \le 2,
\end{aligned}\tag{11}$$

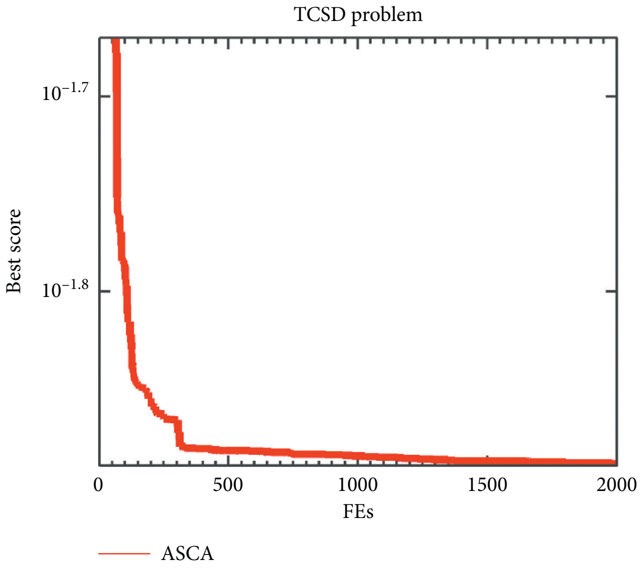

FIGURE 12: Convergence curves of ASCA on TCSD.

where

$$\begin{aligned}
\tau(\overrightarrow{x}) &= \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau\prime\prime)^2},\\
\tau' &= \frac{P}{\sqrt{2}x_1x_2},\\
\tau'' &= \frac{MR}{J},\\
M &= P\left(L + \frac{x_2}{2}\right),\\
R &= \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},\\
J &= 2\left\{\sqrt{2}\,x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\},\\
\sigma(\overrightarrow{x}) &= \frac{6PL}{x_4x_3^2},\\
\delta(\overrightarrow{x}) &= \frac{6PL^3}{Ex_3^2x_4},\\
P_C(\overrightarrow{x}) &= \frac{4.013E\sqrt{x_3^2x_4^6/36}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),\\
P &= 60001b,\\
L &= 14 \in \ldots \delta_{\max} = 0.25 \in \ldots,\\
E &= 30 \times 1^6\,\text{psi},\\
G &= 12 \times 10^6\,\text{psi},\\
\tau_{\max} &= 13600\,\text{psi},\\
\alpha_{\max} &= 30000\,\text{psi}.
\end{aligned} \tag{12}$$

Figure 14 shows the convergence curve for ASCA on this engineering problem.

To solve this problem, this paper selects PSO, RCBA, EPO, WCA [90], and MBA [91] to compare with ASCA. ASCA, PSO, RCBA, and EPO used 2000 iterations in this experiment. WCA and MBA data in the table are taken from their respective papers. WCA used 30000 iterations, and MBA used 2,000 iterations. Table 16 records the optimal solutions of ASCA and other peers to the WBD problem. Table 17 records comparison results of the involved algorithms. We can see that ASCA has achieved the smallest value among the involved algorithms.

4.8.4. RED Problem. The objective of this problem is to maximize the dynamic load-carrying capacity of a rolling element bearing. There are 10 decision variables, such as pitch diameter (D_m), ball diameter (D_b), number of balls (Z), inner (f_i) and outer (f_o) raceway curvature coefficients, $K_{D\min}$, $K_{D\max}$, ε, e, and ζ. The mathematical representation of this problem is given as follows:

$$\begin{aligned}
&\text{maximize}\, C_d = \begin{cases} f_c Z^{2/3} D_b^{1.8}, & \text{if } D \le 25.4\,\text{mm},\\ C_d = 3.647 f_c Z^{2/3} D_b^{1.4}, & \text{if } D > 25.4\,\text{mm}, \end{cases} \quad -Z + 1 \ge 0,\\
&\text{subject to}\, g_1(\overrightarrow{z}) = \frac{\varnothing_0}{2\sin^{-1}(D_d/D_m)}\\
&g_2(\overrightarrow{z}) = 2D_b - K_{Dimn}(D - d) \ge 0,\\
&g_3(\overrightarrow{z}) = K_{Dime}(D - d) - 2D_b \ge 0,\\
&g_4(\overrightarrow{z}) = \zeta B_w - D_b \le 0,\\
&g_5(\overrightarrow{z}) = D_m - 0.5(D + d) \ge 0,\\
&g_6(\overrightarrow{z}) = (0.5 + e)(D + d) - D_m \ge 0,\\
&g_7(\overrightarrow{z}) = 0.5(D - D_m - D_b) - \varepsilon D_d \ge 0,\\
&g_8(\overrightarrow{z}) = f_i \ge 0.515,\\
&g_9(\overrightarrow{z}) = f_0 \ge 0.515,
\end{aligned} \tag{13}$$

Table 12: Optimal solutions of various algorithms to the TCSD problem.

DV	ASCA	PSO	GSA	SCA	CGWO	EPO
X_1 (d)	0.051910	0.057786	0.063231	0.051140	**0.052796**	0.051087
X_2 (D)	0.361930	0.515902	0.700380	0.341042	**0.804380**	0.342910
X_3 (N)	10.995033	6.575697	4.408507	12.390019	**2.000000**	12.089800
$g_1(X)$	−7.48E − 05	−1.28E − 01	−3.20E − 01	−9.65E − 04	**−8.66E − 01**	3.04E − 03
g_2 (X)	−2.81E − 04	−9.79E − 03	−3.51E − 03	−6.07E − 03	**9.02E − 01**	1.17E − 03
g_3 (X)	−4.06E + 00	−3.64E + 00	−3.11E + 00	−3.98E + 00	**−4.73E + 00**	−4.05E + 00
g_4 (X)	−7.24E − 01	−6.18E − 01	−4.91E − 01	−7.39E − 01	**−4.29E − 01**	−7.37E − 01
f (X)	0.012674	0.014774	0.017945	0.012835	**0.01196**	0.012657

Table 13: Comparison results of ASCA and other algorithms on the TCSD problem.

Algorithm	Worst	Best	Mean	Std
ASCA	0.012982	0.012674	0.012806	6.06E-05
PSO	0.111486	0.014774	0.030728	0.015376
GSA	0.265630	0.017945	0.073234	0.050822
SCA	0.076355	0.012835	0.016502	0.006265
CGWO	**0.012179**	**0.011950**	**0.012174**	**1.04E − 05**
EPO	0.012668	0.012656	0.126789	0.001021

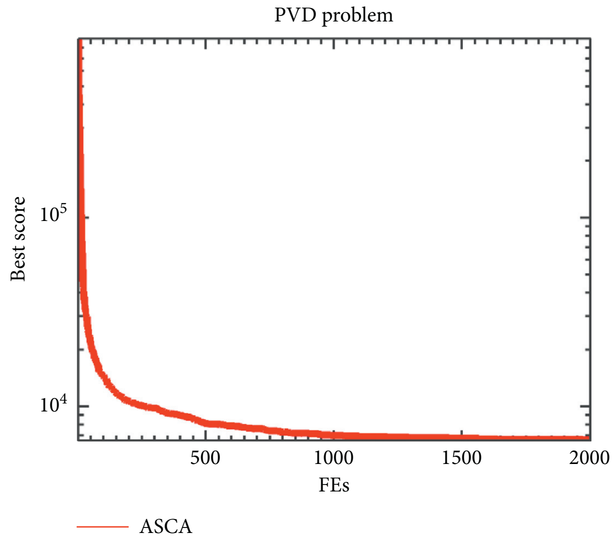

Figure 13: Convergence curves of ASCA on the PVD problem.

Table 14: Optimal solutions of ASCA and other algorithms to the PVD problem.

DV	ASCA	GWO	CBA	SCA	SCADE	CGWO	EPO
X_1 (T_s)	0.785577	0.790799	23.316246	0.979980	1.152076	**1.187150**	0.778099
X_2 (T_h)	0.385137	0.404517	0.000000	0.546429	0.896541	**0.600000**	0.383241
X_3 (R)	40.321290	40.960309	10.000000	50.630927	58.835791	**69.707500**	40.315121
X_4 (L)	200.000000	192.359305	10.000000	93.499845	43.908085	**7.798400**	200.00000
g_1 (X)	−7.38E − 03	−2.65E − 04	−2.31E + 01	−2.80E − 03	−1.65E − 02	**1.58E − 01**	−1.72E − 05
g_2 (X)	−4.72E − 04	−1.38E − 02	9.54E − 02	−6.34E − 02	−3.35E − 01	**6.50E − 02**	1.37E − 03
g_3 (X)	−1.19E + 02	−5.74E + 03	1.29E + 06	−6.67E + 02	−3.46E + 04	**−2.42E + 05**	3.20E + 02
g_4 (X)	−4.00E + 01	−4.76E + 01	−2.30E + 02	−1.47E + 02	−1.96E + 02	**−2.32E + 02**	−4.00E + 01
f (X)	5940.8077	5973.8597	11239.5210	6627.1421	9104.6135	**5034.18**	5880.07

Table 15: Comparison results of ASCA and other peers on the PVD problem.

Algorithm	Worst	Best	Mean	Std
ASCA	7592.8787	5940.8077	6277.0858	223.6783
GWO	7529.1508	5973.8597	6498.0119	343.1708
CBA	1451652.5429	11239.5210	245412.7742	193779.9802
SCA	24346.8096	6627.1421	11723.9470	2983.2579
SCADE	97279.2257	9104.6135	24561.8932	11965.5631
CGWO	**6188.1100**	**5034.1800**	**5783.5830**	254.5050
EPO	5891.3099	5880.0700	5884.1401	**24.3410**

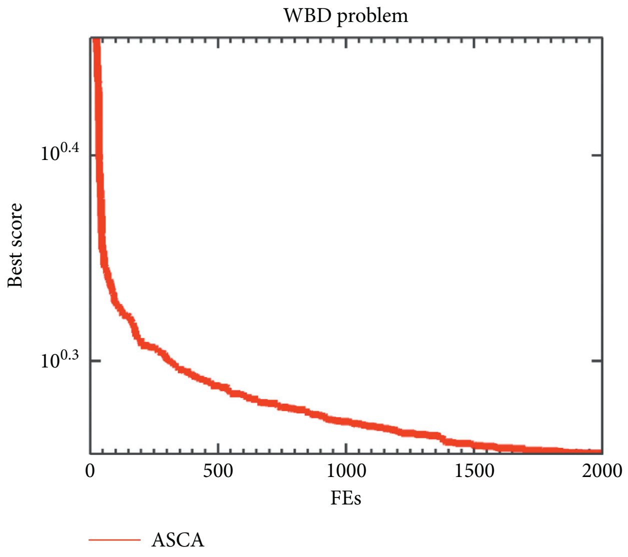

Figure 14: Convergence curves of ASCA on the WBD problem.

Table 16: Optimal solutions of ASCA and other peers to the WBD problem.

DV	ASCA	SCA	SCADE	RCBA	EPO	WCA	MBA
X_1 (h)	**0.207144**	0.193327	0.175004	2.000000	0.205411	0.205728	0.205729
X_2 (l)	**3.333426**	3.590290	4.081111	7.058738	3.472341	3.470522	3.470493
X_3 (t)	**8.995628**	9.046059	9.551483	10.000000	9.035216	9.03662	9.036626
X_4 (b)	**0.207690**	0.209242	0.203722	2.000000	0.201153	0.205729	0.205729
g_1 (X)	**−3.47E + 02**	−3.58E + 02	−1.01E + 03	−1.30E + 04	−7.56E + 02	−7.71E + 02	−7.71E + 02
g_2 (X)	**−1.17E + 01**	−5.65E + 02	−2.88E + 03	−2.75E + 04	6.92E + 02	1.19E − 01	7.94E − 02
g_3 (X)	**−5.46E − 04**	−1.59E − 02	−2.87E − 02	0.00E + 00	4.26E − 03	−1.00E − 06	0.00E + 00
g_4 (X)	**−3.44E + 00**	−3.39E + 00	−3.30E + 00	1.57E + 01	−3.47E + 00	−3.43E + 00	−3.43E + 00
g_5 (X)	**−5.41E − 02**	−5.77E − 02	−7.28E − 02	−2.34E − 01	−4.95E − 02	−5.40E − 02	−5.40E − 02
g_6 (X)	**−1.55E + 02**	−3.17E + 02	−3.78E + 01	−5.87E + 06	3.92E + 02	5.77E − 02	5.51E − 02
g_7 (X)	**−8.21E − 02**	−6.83E − 02	−5.00E − 02	−1.88E + 00	−8.04E − 02	−8.07E − 02	−8.07E − 02
f (X)	**1.716009**	1.750070	1.830738	2.081965	1.723589	1.724856	1.724853

Table 17: Comparison results of ASCA and other peers on the WBD problem.

Algorithm	Worst	Best	Mean	Std
ASCA	1.847513	**1.716009**	1.774598	0.0259
SCA	3.096202	1.750070	2.082322	0.1823
SCADE	6.705880	1.830738	2.568787	0.6072
RCBA	5.771191	2.081965	2.877497	0.6333
EPO	1.727211	1.723589	1.725124	0.0043
WCA	1.801127	1.724856	1.735940	0.0189
MBA	**1.724853**	1.724853	**1.724853**	**6.94E − 19**

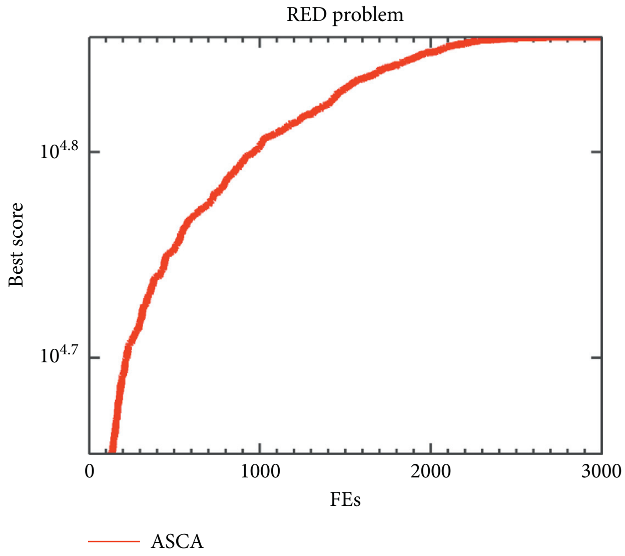

Figure 15: Convergence curves of ASCA on the RED problem.

Table 18: Optimal solutions of ASCA and other peers to the RED problem.

DV	ASCA	SCA	m_SCA	MFO	SCADE	EPO	MBA
X_1 (D_m)	125.3637	126.5664	125.5356	125.0000	125.2300	125.0000	**125.7200**
X_2 (D_b)	21.3786	20.8216	20.8942	25.5701	20.9759	21.4189	**21.4233**
X_3 (Z)	10.8857	9.4246	10.8788	7.6510	10.8860	10.9410	**11.0000**
X_4 (f_i)	0.515	0.515	0.515	0.5172	0.515	0.515	**0.515**
X_5 (f_o)	0.515	0.515	0.518677	0.5303	0.515	0.515	**0.515**
X_6 (K_{dmin})	0.4	0.4	0.449950	0.4	0.4	0.4	**0.4881**
X_7 (K_{dmax})	0.7	0.6	0.607467	0.6027	0.6	0.7	**0.6278**
X_8 (ε)	0.308764	0.3	0.301550	0.3769	0.3	0.3	**0.3001**
X_9 (e)	0.02	0.1	0.044249	0.0499	0.0608	0.02	**0.0973**
X_{10} (ζ)	0.690878	0.6	0.626479	0.85	0.85	0.6	**0.6461**
g_1 (X)	$1.04E-01$	$1.88E+00$	$3.12E-01$	$2.00E+00$	$2.48E-01$	$3.75E-03$	**9.41E − 04**
g_2 (X)	$1.48E+01$	$1.36E+01$	$1.03E+01$	$2.31E+01$	$1.40E+01$	$1.48E+01$	**8.68E + 00**
g_3 (X)	$6.24E+00$	$3.57E-01$	$7.34E-01$	$-8.95E+00$	$4.82E-02$	$6.16E+00$	**1.10E + 00**
g_4 (X)	$-6.52E-01$	$-2.82E+00$	$-2.10E+00$	$-7.01E-02$	$4.52E+00$	$-3.42E+00$	**−2.04E + 00**
g_5 (X)	$3.64E-01$	$1.57E+00$	$5.36E-01$	$0.00E+00$	$2.30E-01$	$0.00E+00$	**7.20E − 01**
g_6 (X)	$4.64E+00$	$2.34E+01$	$1.05E+01$	$1.25E+01$	$1.50\ E+01$	$5.00E+00$	**2.36E + 01**
g_7 (X)	$2.79E-02$	$5.96E-02$	$4.84E-01$	$-4.92E+00$	$6.04E-01$	$3.65E-01$	**−7.82E − 04**
g_8 (X)	$5.16E-01$	$5.15E-01$	$5.28E-01$	$4.31E-01$	$5.26E-01$	$5.15E-01$	**5.15E − 01**
g_9 (X)	$5.16E-01$	$5.15E-01$	$5.28E-01$	$4.31E-01$	$5.26E-01$	$5.15E-01$	**5.15E − 01**
f (X)	80973.1101	70000.5075	76486.5050	53468.4636	66341.4999	85067.983	**85535.961**

Table 19: Comparison results of ASCA and other peers on the RED problem.

Algorithm	Worst	Best	Mean	Std
ASCA	56422.9931	80973.1101	71127.3324	4063.4106
SCA	14815.3956	70000.5075	35939.6478	13004.9651
m_SCA	27462.1666	76486.5050	53645.0661	12801.6385
MFO	16482.6478	53468.4636	31152.8968	9458.8799
SCADE	15623.0122	66341.4999	31947.7077	11796.2137
EPO	**85042.3520**	85067.9830	**86551.5990**	1877.0900
MBA	84440.1948	**85535.9611**	85321.4030	**211.5200**

where

$$
\begin{aligned}
f_c &= 37.91\left[1+\left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i\left(2f_0-1\right)}{f_0\left(2f_i-1\right)}\right)^{0.41}\right\}^{10/3}\right]^{-0.3}\times\left[\frac{\gamma^{0.3}\left(1-\gamma\right)^{1.39}}{\left(1+\gamma\right)^{1/3}}\right]\left[\frac{2f_i}{2f_i-1}\right]^{0.41},\\
x &= \left[\left\{\frac{(D-d)}{2}-3\left(\frac{T}{4}\right)\right\}^2+\left\{\frac{D}{2}-\frac{T}{4}-D_b\right\}^2-\left\{\frac{d}{2}+\frac{T}{4}\right\}^2\right],\\
y &= 2\left\{\frac{(D-d)}{2}-3\left(\frac{T}{4}\right)\right\}\left\{\frac{D}{2}-\frac{T}{4}-D_b\right\},\\
\varnothing_0 &= 2\pi-2\cos^{-1}\left(\frac{x}{y}\right),\\
\gamma &= \frac{D_b}{D_m},\\
f_i &= \frac{r_i}{D_b},\\
f_o &= \frac{r_o}{D_b},\\
T &= D-d-2D_b,\\
D &= 160,\\
d &= 90,\\
B_w &= 30,\\
r_i &= r_o = 11.033,\\
0.5\left(D+d\right) &\leq D_m \leq 0.6\left(D+d\right),\\
0.15\left(D-d\right) &\leq D_b \leq 0.45\left(D-d\right),\\
4 &\leq Z \leq 50,\\
0.515 &\leq f_i, f_o \leq 0.6,\\
0.4 &\leq K_{D\min} \leq 0.5,\\
0.6 &\leq K_{D\max} \leq 0.7,\\
0.3 &\leq e \leq 0.4,\\
0.02 &\leq e \leq 0.1,\\
0.6 &\leq \zeta \leq 0.85.
\end{aligned}
\tag{14}
$$

Figure 15 shows the convergence curve for ASCA on this engineering problem.

To solve this problem, SCA, m_SCA, MFO, SCADE, EPO, and MBA are chosen to compare with ASCA. ASCA, SCA, m_SCA, MFO, and SCADE used 3000 iterations in this experiment. EPO and MBA data are taken from their respective papers. EPO uses 1000 iterations in its paper. MBA uses 15100 iterations in this paper. Table 18 records the optimal solutions of ASCA and other peers to the problem. Table 19 records comparison results of involved algorithms. ASCA has no outstanding effect on this project. As can be seen from the results in the table, ASCA has no particularly outstanding performance on this engineering issue. However, ASCA has a significant improvement over both the original SCA and the improved SCA, and it can obtain better solutions and more stability.

5. Conclusions and Future Works

This paper introduced an adaptive transformation strategy and a chaotic local search strategy to improve the performance of the original SCA. The adaptive transformation strategy was designed to balance the global exploration and local exploitation of SCA. The chaotic local search strategy was introduced to further explore the promising area around the optimal solution found by SCA. The proposed ASCA has been compared with various well-known and advanced meta-heuristics on a comprehensive set of benchmark problems. The experimental results of benchmark functions show that the proposed ASCA is superior to the involved competitive peers in convergence accuracy and solution accuracy. Additionally, for dealing with engineering cases, we can see that ASCA has improved significantly compared with SCA and some other SCA variants. Of course, compared with some other excellent algorithms, there is still a gap, and we need to continue to study further to improve the proposed method in the future.

In the future work, the effectiveness of tuning other parameters of SCA can be investigated. The proposed ASCA can be applied in many other scenarios, such as image segmentation, clustering, parameter tuning for deep learning models [92–94], social manufacturing optimization [95], and video coding optimization [96].

Data Availability

The data involved in this study are all public data, which can be downloaded through public channels.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Authors' Contributions

Guoxi Liang and Huiling Chen contributed equally to this work.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (62076185 and U1809209), Science and Technology Plan Project of Wenzhou (G20190020), and general research project of Zhejiang Provincial Education Department (Y201942618).

References

[1] S. Zhao, P. N. Suganthan, and S. Das, "Dynamic multi-swarm particle swarm optimizer with sub-regional harmony search," in *Proceedings of the 2010 IEEE Congress on Evolutionary Computation*, Barcelona, Spain, 2010.

[2] X. Zhang, D. Wang, Z. Zhou, and Y. Ma, "Robust low-rank tensor recovery with rectification and alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1, 2019.

[3] Q. Zhang, H. Chen, A. A. Heidari et al., "Chaos-induced and mutation-driven schemes boosting salp chains-inspired optimizers," *IEEE Access*, vol. 7, pp. 31243–31261, 2019.

[4] H. Faris, A. A. Heidari, A. M. Al-Zoubi et al., "Time-varying hierarchical chains of salps with random weight networks for feature selection," *Expert Systems with Applications*, vol. 140, p. 112898, 2020.

[5] M. Mafarja, A. Qasem, A. A. Heidari, I. Aljarah, H. Faris, and S. Mirjalili, "Efficient hybrid nature-inspired binary optimizers for feature selection," *Cognitive Computation*, vol. 12, no. 1, pp. 150–175, 2020.

[6] S. Gupta, "Harmonized salp chain-built optimization," *Engineering with Computers*, pp. 1–31, 2019.

[7] M. Taradeh, M. Mafarja, A. A. Heidari et al., "An evolutionary gravitational search-based feature selection," *Information Sciences*, vol. 497, pp. 219–239, 2019.

[8] M. A. Elaziz, A. A. Heidari, H. Fujita, and H. Moayedi, "A competitive chain-based Harris Hawks optimizer for global optimization and multi-level image thresholding problems," *Applied Soft Computing*, Article ID 106347, 2020.

[9] E. Rodríguez-Esparza, L. A. Zanella-Calzada, D. Oliva et al., "An efficient Harris hawks-inspired image segmentation method," *Expert Systems with Applications*, vol. 155, Article ID 113428, 2020.

[10] X. Zhao, D. Li, B. Yang, C. Ma, Y. Zhu, and H. Chen, "Feature selection based on improved ant colony optimization for online detection of foreign fiber in cotton," *Applied Soft Computing*, vol. 24, pp. 585–596, 2014.

[11] M. Wang and H. Chen, "Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis," *Applied Soft Computing*, vol. 88, Article ID 105946, 2020.

[12] X. Zhao, X. Zhang, Z. Cai et al., "Chaos enhanced grey wolf optimization wrapped ELM for diagnosis of paraquat-poisoned patients," *Computational Biology and Chemistry*, vol. 78, pp. 481–490, 2019.

[13] X. Xu and H.-L. Chen, "Adaptive computational chemotaxis based on field in bacterial foraging optimization," *Soft Computing*, vol. 18, no. 4, pp. 797–807, 2014.

[14] L. Shen, H. Chen, Z. Yu et al., "Evolving support vector machines using fruit fly optimization for medical data classification," *Knowledge-Based Systems*, vol. 96, pp. 61–75, 2016.

[15] M. Wang, H. Chen, B. Yang et al., "Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses," *Neurocomputing*, vol. 267, pp. 69–84, 2017.

[16] Y. Xu, H. Chen, J. Luo, Q. Zhang, S. Jiao, and X. Zhang, "Enhanced Moth-flame optimizer with mutation strategy for global optimization," *Information Sciences*, vol. 492, pp. 181–203, 2019.

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

[18] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[19] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[20] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*, pp. 760–766, Springer US, Boston, MA, USA, 2010.

[21] S. Mirjalili, "Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, pp. 228–249, 2015.

[22] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–72, 1992.

[23] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: a new method for stochastic optimization," *Future Generation Computer Systems*, vol. 111, pp. 300–323, 2020.

[24] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[25] J. Del Ser, E. Osaba, D. Molina et al., "Bio-inspired computation: where we stand and what's next," *Swarm and Evolutionary Computation*, vol. 48, pp. 220–250, 2019.

[26] W. Qiao, H. Moayedi, and L. K. Foong, "Nature-inspired hybrid techniques of IWO, DA, ES, GA, and ICA, validated through a *k*-fold validation process predicting monthly natural gas consumption," *Energy and Buildings*, vol. 217, Article ID 110023, 2020.

[27] A. Abbassi, R. Abbassi, A. A. Heidari et al., "Parameters identification of photovoltaic cell models using enhanced exploratory salp chains-based approach," *Energy*, vol. 198, Article ID 117333, 2020.

[28] W. Deng, J. Xu, and H. Zhao, "An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem," *IEEE Access*, vol. 7, pp. 20281–20292, 2019.

[29] W. Deng, H. Zhao, L. Zou, G. Li, X. Yang, and D. Wu, "A novel collaborative optimization algorithm in solving complex optimization problems," *Soft Computing*, vol. 21, no. 15, pp. 4387–4398, 2017.

[30] S. Mirjalili, "SCA: a sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.

[31] Q. Yang, S.-C. Chu, J.-S. Pan, and C.-M. Chen, "Sine cosine algorithm with multigroup and multistrategy for solving CVRP," *Mathematical Problems in Engineering*, vol. 2020, Article ID 8184254, 10 pages, 2020.

[32] A. A. Heidari, I. Aljarah, H. Faris, H. Chen, J. Luo, and S. Mirjalili, "An enhanced associative learning-based exploratory whale optimizer for global optimization," *Neural Computing and Applications*, vol. 32, pp. 5185–5211, 2019.

[33] A. A. Heidari, H. Faris, I. Aljarah, and S. Mirjalili, "An efficient hybrid multilayer perceptron neural network with grasshopper optimization," *Soft Computing*, pp. 1–18, 2018.

[34] A. A. Heidari, "Ant lion optimizer: theory, literature review, and application in multi-layer perceptron neural networks," in *Nature-Inspired Optimizers: Theories, Literature Reviews and Applications*, pp. 23–46, Springer International Publishing, Cham, Switzerland, 2020.

[35] M. Mafarja, I. Aljarah, A. A. Heidari et al., "Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems," *Knowledge-Based Systems*, vol. 145, pp. 25–45, 2018.

[36] H. Moayedi and S. Hayati, "Modelling and optimization of ultimate bearing capacity of strip footing near a slope by soft computing methods," *Applied Soft Computing*, vol. 66, pp. 208–219, 2018.

[37] H. Moayedi and S. Hayati, "Applicability of a CPT-based neural network solution in predicting load-settlement responses of bored pile," *International Journal of Geomechanics*, vol. 18, no. 6, Article ID 06018009, 2018.

[38] H. Moayedi and A. Rezaei, "An artificial neural network approach for under-reamed piles subjected to uplift forces in dry sand," *Neural Computing and Applications*, vol. 31, no. 2, pp. 327–336, 2019.

[39] H. Faris, A. M. Al-Zoubi, A. A. Heidari et al., "An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks," *Information Fusion*, vol. 48, pp. 67–83, 2019.

[40] H. Faris, M. M. Mafarja, A. A. Heidari et al., "An efficient binary salp swarm algorithm with crossover scheme for feature selection problems," *Knowledge-Based Systems*, vol. 154, pp. 43–67, 2018.

[41] G. Liu, W. Jia, M. Wang et al., "Predicting cervical hyperextension injury: a covariance guided sine cosine support vector machine," *IEEE Access*, vol. 8, pp. 46895–46908, 2020.

[42] M. Mafarja, I. Aljarah, A. A. Heidari et al., "Binary dragonfly optimization for feature selection using time-varying transfer functions," *Knowledge-Based Systems*, vol. 161, pp. 185–204, 2018.

[43] Y. Xu, H. Chen, A. A. Heidari et al., "An efficient chaotic mutative moth-flame-inspired optimizer for global optimization tasks," *Expert Systems with Applications*, vol. 129, pp. 135–155, 2019.

[44] X. Zhang, Y. Xu, C. Yu et al., "Gaussian mutational chaotic fruit fly-built optimization and feature selection," *Expert Systems with Applications*, vol. 141, Article ID 112976, 2020.

[45] X. Liang, Z. Cai, M. Wang, X. Zhao, H. Chen, and C. Li, "Chaotic oppositional sine-cosine method for solving global optimization problems," *Engineering with Computers*, 2020.

[46] W. Zhu, C. Ma, X. Zhao et al., "Evaluation of sino foreign cooperative education project using orthogonal sine cosine optimized kernel extreme learning machine," *IEEE Access*, vol. 8, pp. 61107–61123, 2020.

[47] H. Chen, M. Wang, and X. Zhao, "A multi-strategy enhanced sine cosine algorithm for global optimization and constrained practical engineering problems," *Applied Mathematics and Computation*, vol. 369, Article ID 124872, 2020.

[48] H. Chen, A. A. Heidari, X. Zhao, L. Zhang, and H. Chen, "Advanced orthogonal learning-driven multi-swarm sine cosine optimization: framework and case studies," *Expert Systems with Applications*, vol. 144, Article ID 113113, 2020.

[49] J. Tu, A. Lin, H. Chen, Y. Li, and C. Li, "Predict the entrepreneurial intention of fresh graduate students based on an adaptive support vector machine framework," *Mathematical Problems in Engineering*, vol. 2019, Article ID 2039872, 16 pages, 2019.

[50] H. Chen, S. Jiao, A. A. Heidari, M. Wang, X. Chen, and X. Zhao, "An opposition-based sine cosine approach with local search for parameter estimation of photovoltaic models," *Energy Conversion and Management*, vol. 195, pp. 927–942, 2019.

[51] Y. Sun, G. G. Yen, and Z. Yi, "IGD indicator-based evolutionary algorithm for many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 173–187, 2019.

[52] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "Evolving deep convolutional neural networks for image classification," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 394–407, 2020.

[53] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, "Automatically designing CNN architectures using the genetic algorithm for image classification," *IEEE Transactions on Cybernetics*, pp. 1–15, 2020.

[54] H. Nenavath and R. K. Jatoth, "Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking," *Applied Soft Computing*, vol. 62, pp. 1019–1043, 2018.

[55] R. M. Rizk-Allah, "Hybridizing sine cosine algorithm with multi-orthogonal search strategy for engineering design problems," *Journal of Computational Design and Engineering*, vol. 5, no. 2, pp. 249–273, 2018.

[56] M. Abd Elaziz, D. Oliva, and S. Xiong, "An improved opposition-based sine cosine algorithm for global optimization," *Expert Systems with Applications*, vol. 90, pp. 484–500, 2017.

[57] J. Zhang, Y. Zhou, and Q. Luo, "An improved sine cosine water wave optimization algorithm for global optimization," *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 4, pp. 2129–2141, 2018.

[58] R. M. Rizk-Allah, "An improved sine–cosine algorithm based on orthogonal parallel information for global optimization," *Soft Computing*, vol. 23, pp. 7135–7161, 2018.

[59] C. Qu, Z. Zeng, J. Dai, Z. Yi, and W. He, "A modified sine-cosine algorithm based on neighborhood search and greedy Levy mutation," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 4231647, 19 pages, 2018.

[60] M. A. Tawhid and V. Savsani, "Multi-objective sine-cosine algorithm (MO-SCA) for multi-objective engineering design problems," *Neural Computing and Applications*, vol. 31, pp. 915–929, 2017.

[61] R. Sindhu, R. Ngadiran, Y. M. Yacob, N. A. H. Zahri, and M. Hariharan, "Sine-cosine algorithm for feature selection with elitism strategy and new updating mechanism," *Neural Computing and Applications*, vol. 28, no. 10, pp. 2947–2958, 2017.

[62] K. Z. Zamli, F. Din, B. S. Ahmed, and M. Bures, "A hybrid Q-learning sine-cosine-based strategy for addressing the combinatorial test suite minimization problem," *PLoS One*, vol. 13, no. 5, Article ID e0195675, 2018.

[63] O. E. Turgut, "Thermal and economical optimization of a shell and tube evaporator using hybrid backtracking search-sine-cosine algorithm," *Arabian Journal for Science and Engineering*, vol. 42, no. 5, pp. 2105–2123, 2017.

[64] S. N. Chegini, A. Bagheri, and F. Najafi, "PSOSCALF: a new hybrid PSO based on sine cosine algorithm and Levy flight for solving optimization problems," *Applied Soft Computing*, vol. 73, pp. 697–726, 2018.

[65] M. Issa, A. E. Hassanien, D. Oliva, A. Helmi, I. Ziedan, and A. Alzohairy, "ASCA-PSO: adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment," *Expert Systems with Applications*, vol. 99, pp. 56–70, 2018.

[66] S. Gupta and K. Deep, "A hybrid self-adaptive sine cosine algorithm with opposition based learning," *Expert Systems with Applications*, vol. 119, pp. 210–230, 2019.

[67] S. Abdel-Fatah, M. Ebeed, and S. Kamel, "Optimal reactive power dispatch using modified sine cosine algorithm," in *Proceedings of the 2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, Aswan, Egypt, February 2019.

[68] H. Huang, X. A. Feng, A. A. Heidari et al., "Rationalized sine cosine optimization with efficient searching patterns," *IEEE Access*, vol. 8, pp. 61471–61490, 2020.

[69] M. Abdo, S. Kamel, M. Ebeed, J. Yu, and F. Jurado, "Solving non-smooth optimal power flow problems using a developed grey wolf optimizer," *Energies*, vol. 11, no. 7, p. 1692, 2018.

[70] S. Gupta, K. Deep, S. Mirjalili, and J. H. Kim, "A modified sine cosine algorithm with novel transition parameter and mutation operator for global optimization," *Expert Systems with Applications*, vol. 154, Article ID 113395, 2020.

[71] M. A. Taher, S. Kamel, F. Jurado, and M. Ebeed, "Modified grasshopper optimization framework for optimal power flow solution," *Electrical Engineering*, vol. 101, no. 1, pp. 121–148, 2019.

[72] M. A. E. S. Mohamed, A. A. E. Mohammed, A. M. Abd Elhamed, and M. E. Hessean, "Optimal allocation of photovoltaic based and DSTATCOM in a distribution network under multi load levels," *European Journal of Engineering Research and Science*, vol. 4, no. 8, pp. 114–119, 2019.

[73] R. Anguluri, "Computing with the collective intelligence of honey bees—a survey," *Swarm and Evolutionary Computation*, vol. 32, 2016.

[74] S.-Z. Zhao, P. Suganthan, and S. Das, "Dynamic multi-swarm particle swarm optimizer with subregional harmony search," in *Proceedings of the 2010 IEEE World Congress on Computational Intelligence*, Barcelona, Spain, 2010.

[75] W. Long, T. Wu, X. Liang, and S. Xu, "Solving high-dimensional global optimization problems using an improved sine cosine algorithm," *Expert Systems with Applications*, vol. 123, pp. 108–126, 2018.

[76] H. Chen, Q. Zhang, J. Luo, Y. Xu, and X. Zhang, "An enhanced bacterial foraging optimization and its application for training kernel extreme learning machine," *Applied Soft Computing*, vol. 86, Article ID 105884, 2019.

[77] H. Chen, S. Jiao, M. Wang, A. A. Heidari, and X. Zhao, "Parameters identification of photovoltaic cells and modules using diversification-enriched Harris Hawks optimization with chaotic drifts," *Journal of Cleaner Production*, vol. 244, Article ID 118778, 2020.

[78] H. Chen, Y. Xu, M. Wang, and X. Zhao, "A balanced whale optimization algorithm for constrained engineering design problems," *Applied Mathematical Modelling*, vol. 71, pp. 45–59, 2019.

[79] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, pp. 65–74, Springer, Berlin, Germany, 2010.

[80] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[81] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*-Springer, Berlin, Germany, 2009.

[82] J. Carrasco, S. García, M. M. Rueda, S. Das, and F. Herrera, "Recent trends in the use of statistical tests for comparing

swarm and evolutionary computing algorithms: practical guidelines and a critical review," *Swarm and Evolutionary Computation*, vol. 54, Article ID 100665, 2020.

[83] B. R. Adarsh, T. Raghunathan, T. Jayabarathi, and X.-S. Yang, "Economic dispatch using chaotic bat algorithm," *Energy*, vol. 96, pp. 666–675, 2016.

[84] H. Liang, Y. Liu, Y. Shen, F. Li, and Y. Man, "A hybrid bat algorithm for economic dispatch with random wind power," *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5052–5061, 2018.

[85] J. A. Yong, F. He, H. Li, and W. Zhou, "Novel bat algorithm based on collaborative and dynamic learning of opposite population," in *Proceedings of the 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, Nanjing, China, May 2018.

[86] A. Lin, Q. Wu, A. A. Heidari et al., "Predicting intentions of students for master programs using a chaos-induced sine cosine-based fuzzy *K*-nearest neighbor classifier," *IEEE Access*, vol. 7, pp. 67235–67248, 2019.

[87] S.-h. Liu, M. Mernik, D. Hrnčič, and M. Črepinšek, "A parameter control method of evolutionary algorithms using exploration and exploitation measures with a practical application for fitting Sovova's mass transfer model," *Applied Soft Computing*, vol. 13, no. 9, pp. 3792–3805, 2013.

[88] M. Kohli and S. Arora, "Chaotic grey wolf optimization algorithm for constrained optimization problems," *Journal of Computational Design and Engineering*, vol. 5, no. 4, pp. 458–472, 2018.

[89] G. Dhiman and V. Kumar, "Emperor penguin optimizer: a bio-inspired algorithm for engineering problems," *Knowledge-Based Systems*, vol. 159, pp. 20–50, 2018.

[90] H. Eskandar, A. Sadollah, A. Bahreininejad, and M. Hamdi, "Water cycle algorithm—a novel metaheuristic optimization method for solving constrained engineering optimization problems," *Computers & Structures*, vol. 110-111, pp. 151–166, 2012.

[91] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, "Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems," *Applied Soft Computing*, vol. 13, no. 5, pp. 2592–2612, 2013.

[92] X. Zhang, T. Wang, J. Wang, G. Tang, and L. Zhao, "Pyramid channel-based feature attention network for image dehazing," *Computer Vision and Image Understanding*, vol. 197-198, Article ID 103003, 2020.

[93] Y. Li, W.-G. Cui, H. Huang, Y.-Z. Guo, K. Li, and T. Tan, "Epileptic seizure detection in EEG signals using sparse multiscale radial basis function networks and the Fisher vector approach," *Knowledge-Based Systems*, vol. 164, pp. 96–106, 2019.

[94] Y. Li, J. Liu, Z. Tang, and B. Lei, "Deep spatial-temporal feature fusion from adaptive dynamic functional connectivity for MCI identification," *IEEE Transactions on Medical Imaging*, vol. 39, no. 9, pp. 2818–2830, 2020.

[95] X. Xue, S. Wang, L. Zhang, Z. Feng, and Y. Guo, "Social learning evolution (SLE): computational experiment-based modeling framework of social manufacturing," *Ieee Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3343–3355, 2019.

[96] Y. Zhou, L. Tian, C. Zhu, X. Jin, and Y. Sun, "Video coding optimization for virtual reality 360-degree source," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 1, pp. 118–129, 2020.