Hindawi
Scientific Programming
Volume 2022, Article ID 8249030, 35 pages
https://doi.org/10.1155/2022/8249030

Research Article

A Complex-Valued Encoding Multichain Seeker Optimization Algorithm for Engineering Problems

Shaomi Duan,[1,2] **Huilong Luo**,[1] **and Haipeng Liu**[2]

[1]*The Faculty of Civil Engineering and Mechanics, Kunming University of Science and Technology, Kunming 650500, China*
[2]*The Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China*

Correspondence should be addressed to Huilong Luo; huilongluo@kmust.edu.cn

Received 1 August 2021; Revised 7 October 2021; Accepted 8 December 2021; Published 6 January 2022

Academic Editor: Jiwei Huang

This article comes up with a complex-valued encoding multichain seeker optimization algorithm (CMSOA) for the engineering optimization problems. The complex-valued encoding strategy and the multichain strategy are leaded in the seeker optimization algorithm (SOA). These strategies enhance the individuals' diversity, enhance the local search, avert falling into the local optimum, and are the influential global optimization strategies. This article chooses fifteen benchmark functions, four proportional integral derivative (PID) control parameter models, and six constrained engineering problems to test. According to the experimental results, the CMSOA can be used in the benchmark functions, in the PID control parameter optimization, and in the optimization of constrained engineering problems. Compared to the particle swarm optimization (PSO), simulated annealing based on genetic algorithm (SA_GA), gravitational search algorithm (GSA), sine cosine algorithm (SCA), multiverse optimizer (MVO), and seeker optimization algorithm (SOA), the optimization ability and robustness of the CMSOA are better than those of others algorithms.

1. Introduction

Recently, the heuristic algorithm has received a lot of attention. Such algorithms create random methods for many optimization problems. Since the no free lunch (NFL) theorem, no one optimization solution can optimize overall questions [1]. Therefore, researchers pose new algorithms or enhance the current algorithms to deal with optimization problems. The current algorithms are the genetic algorithm (GA) [2], particle swarm optimization (PSO) [3], simulated annealing (SA) [4], harmony search (HS) [5], gravitational search algorithm (GSA) [6], moth-flame optimization (MFO) [7], sine cosine algorithm (SCA) [8], multiverse optimizer (MVO) [9], seeker optimization algorithm (SOA) [10], monarch butterfly optimization (MBO) [11], slime mould algorithm (SMA) [12], moth search algorithm (MSA) [13], hunger games search (HGS) [14], Runge–Kutta method (RUN) [15], and Harris hawks optimization (HHO) [16].

However, some optimization algorithms are still not very successful in many optimization problems. The optimization problems include the following: being premature, issues with low optimization precision, having only a local optimal solution, slow convergence speed, and insufficient robustness. To better overcome the issues of common optimization precision, prematurity, having only a local optimal solution, slow convergence rate, and poor robustness, some improved algorithms have proven to be feasible optimization algorithms and have been used in practical engineering. For instance, the evolutionary algorithms are improved by the adaptive parameter control methods [17]. The simulated annealing algorithm based on the particle swarm algorithm is adopted to optimize the extracting multiple tests [18]. A whale optimization algorithm based on hybrid algorithm framework with learning and complementary is used in function optimization and engineering design problems [19]. A multilayered gravitational search algorithm is used in function optimization and real-world problems [20]. An artificial bee colony algorithm search is improved by scale-free networks [21]. The chaotic local search-based differential evolution algorithm is applied to optimize the

function optimization and the real-world optimization problems [22].

Also, the complex-valued encoding heuristic algorithms have been proposed according to the characteristics of some algorithms. These complex-valued encoding intelligent optimization algorithms have proven to be feasible optimization algorithms and have been used in practical engineering. For instance, the complex-valued encoding dragonfly algorithm optimized the power systems [23]. A gray wolf optimization based on plural encoding optimized the filter model [24]. The complex-valued encoding satin bowerbird optimization algorithm solved the benchmark functions [25]. The complex-valued encoding-driven optimization optimized the 0-1 knapsack problem [26]. The complex-valued encoding symbiotic organism search algorithm was proposed for the overall optimization [27]. The complex-valued encoding flower pollination algorithm optimized the constrained engineering optimization problems [28]. A comprehensive survey was offered for the complex-valued encoding metaheuristic optimization algorithm [24].

Dai et al. proposed SOA in 2006 [29]; the goal is to mimic the seekers' behavior and the way they exchange information and solve practical application optimization problems. Recently, SOA has been used in many fields, such as in unconstrained optimization problems [30], optimal reactive power dispatch [31], a challenging set of benchmark problems [32], the design of a digital filter [33], optimizing parameters of artificial neural networks [34], the optimizing model and structures of fuel cell [35], the novel human group optimizer algorithm [36], and several practical applications [37].

However, in the initial stage of dealing with optimization problems, the SOA converges faster than others. When all individuals are near to the best individual for solving the optimization problem, the individuals will lose diversity and fall into prematurity.

To overcome the shortcomings of the SOA, there are various strategies for improving the SOA, such as the best empirical parameter strategy, the dynamic adaptive Gaussian variation of empirical parameter strategy, the Chebyshev chaos of order three strategy, the real coding double-link strategy, the complex-valued encoding strategy, and the complex-valued encoding multichain strategy. After improving the SOA with the above strategies and making an experimental comparison, this paper selects several improved strategies with better results to improve the SOA together. In this article, complex number coding and a multichain strategy are used to enhance the global optimization and the local search. We propose the complex-valued encoding multichain seeker optimization algorithm (CMSOA). The multichain strategy includes the complex-valued multichain and the stochastic complex multichain strategy. The CMSOA has been tested on fifteen benchmark functions, four PID control parameter optimizations, and six engineering optimizations taken from the literature. In comparison with PSO, SA_GA, GSA, SCA, MVO, and SOA, the CMSOA can find better values to solving the questions, and the precision and robustness of the CMSOA are better. The complex-valued encoding and the multichain methods enhance the diversity of the individuals and avert premature convergence. The CMSOA overcomes the premature convergence of the SOA. The advantages of the CMSOA are summed up as follows:

(1) CMSOA is proposed to enhance the precision and robustness of optimization.

(2) With the complex-coded multichain strategy, in complex-valued coding, the real part, imaginary part, and real number are used as parallel individual variables to solve the objective function problem.

(3) The stochastic multichain strategy is introduced in the SOA. According to the initial solution generation rule of the complex number coding, the real part, the imaginary part, and the real number are randomly generated as the parallel individual variables to solve the objective function.

(4) The complex-coded strategy, the multichain strategy, and the stochastic multichain strategy can improve the diversity of individuals, enhance local search, and avert premature convergence.

The rest of the article is organized as follows. Section 2 presents the SOA and the algorithm improvement strategies. Section 3 describes the CMSOA. Section 4 shows the algorithm optimization experiments, the results, and the analyses. At last, Section 5 gives some conclusions.

2. The Basic SOA and Algorithm Improvement Strategies

The SOA carries out in-depth research on human search behavior. It considers optimization as a search for an optimal solution by a search team in search space, taking search team as population and the site of the searcher as task method. Using "experience gradient" to determine the search direction, we use uncertain reasoning to resolve the search step measurement, through the scout direction and search step size to complete the searchers' position in the search interspace update, to attain the optimization of the solution.

2.1. Key Update Points for SOA. The SOAs have three main updating steps. In this section, i is the ith searcher individual and j represents the individual dimension. s is the total number of individuals; D is the total number of dimensions of the variable; t means the current algebra; and $\text{iter}_{\max}$ represents the maximum optimization algebra. $x_{ij}(t)$ and $x_{ij}(t+1)$, respectively, represent the searchers' site at algebras t and $(t+1)$.

2.1.1. Search Direction. The forward orientation of a search is defined by the experience gradient obtained from the individuals' movement and the evaluation of other individuals' search historical position. The egoistic direction $\overrightarrow{f}_{i,e}(t)$, altruistic direction $\overrightarrow{f}_{i,a}(t)$, and preemptive direction $\overrightarrow{f}_{i,p}(t)$ of the ith individual in any dimension can be obtained.

$$\begin{aligned}\overrightarrow{f}_{i,e}(t) &= \overrightarrow{p}_{i,\text{best}} - \overrightarrow{x}_i(t),\\ \overrightarrow{f}_{i,a}(t) &= \overrightarrow{g}_{i,\text{best}} - \overrightarrow{x}_i(t),\\ \overrightarrow{f}_{i,p}(t) &= \overrightarrow{x}_i(t_1) - \overrightarrow{x}_i(t_2).\end{aligned} \tag{1}$$

The searcher uses the method of a random weighted average to obtain the search orientation.

$$\overrightarrow{f}_i(t) = \text{sign}\left(\omega \overrightarrow{f}_{i,p}(t) + \varphi_1 \overrightarrow{f}_{i,e}(t) + \varphi_2 \overrightarrow{f}_{i,a}(t)\right), \tag{2}$$

where $t_1, t_2 \in \{t, t-1, t-2\}$; $\overrightarrow{x}_i(t_1)$ and $\overrightarrow{x}_i(t_2)$ are the best advantages of $\{\overrightarrow{x}_i(t-2), \overrightarrow{x}_i(t-1), \overrightarrow{x}_i(t)\}$ separately; $g_{i,\text{best}}$ is the historical optimal location in the neighborhood where the ith search factor is located; $p_{i,\text{best}}$ is the optimal locality from the ith search factor to the current locality; ψ_1 and ψ_1 are random numbers in [0, 1]; and ω is the weight of inertia.

2.1.2. Search Step Size. The SOA refers to the reasoning of the fuzzy approximation ability. The SOA, through the computer language, describes some of the human natural languages that can simulate human intelligence reasoning search behavior. If the algorithm expresses a simple fuzzy rule, it adapts to the best approximation of the objective optimization problems. Greater search step length is more important. However, the smaller fitness corresponds to the smaller search step length. The Gaussian distribution function is adopted to describe the search step measurement.

$$\mu(\alpha) = e^{-\alpha^2/2\delta^2}, \tag{3}$$

whereα andδ are parameters of a membership function.

According to equation (3), the probability of the output variable exceeding $[-3\delta, 3\delta]$ is less than 0.0111. Therefore, $\mu_{\min} = 0.0111$. Under normal circumstances, the optimal position of an individual has $\mu_{\max} = 1.0$ and the worst place is 0.0111. However, to accelerate the convergence speed and get the optimal individual to have an uncertain step size, $\mu_{\max}$ is set as 0.9 in this paper. Select the following function as the fuzzy variable with a "small" target function value:

$$\mu_i = \mu_{\max} - \frac{s - I_i}{s - I}(\mu_{\max} - \mu_{\min}), \quad i = 1, 2, \ldots, s, \tag{4}$$

$$\mu_{ij} = \text{rand}(\mu_i, 1), \quad j = 1, 2, \ldots, D, \tag{5}$$

where μ_{ij} is determined by equations (4) and (5) and I_i is the count of the sequence $x_i(t)$ of the current individuals arranged from high to low by function value. The function rand(μ_i, 1) is the real number in any partition [μ_i, 1]. It can be seen from equation (4) that it simulates the random search behavior of human beings. Step measurement of j-dimensional search interspace is determined by

$$\alpha_{\text{ij}} = \delta_{\text{ij}} - \sqrt{-\ln\left(\mu_{\text{ij}}\right)}, \tag{6}$$

where δ_{ij} is a parameter of the Gaussian distribution function, which is defined by

$$\omega = \frac{\left(\text{iter}_{\max} - t\right)}{\text{iter}_{\max}}, \tag{7}$$

$$\delta_{\text{ij}} = \omega\left|\overrightarrow{x}_{\min} - \overrightarrow{x}_{\max}\right|, \tag{8}$$

where ω is the weight of inertia. As the evolutionary algebra increases, ω decreases linearly from 0.9 to 0.1. $\overrightarrow{x}_{\min}$ and $\overrightarrow{x}_{\max}$ are, respectively, the variate of the minimum value and maximum value of the function.

2.1.3. Individual Location Updates. After obtaining the scout direction and scout step measurement of the individual, the location update is represented by

$$x_{ij}(t+1) = x_{ij}(t) + \alpha_{ij}(t) f_{ij}(t), \quad i = 1, 2, \ldots, s;\ j = 1, 2, \ldots, D. \tag{9}$$

$f_{ij}(t)$ and $\alpha_{ij}(t)$, respectively, represent the searchers' search direction and search step size at time t.

2.2. The Algorithm Improvement Strategies. Five strategies for improving the algorithm are listed in this paper.

2.2.1. The Best Empirical Parameter Strategy. The first strategy is an empirical parameter change strategy. In the basic SOA, equation (8) is changed to equation (10), and the empirical value C is changed to a fixed empirical value. Through a large number of experimental tests, the empirical value is $C = 0.2$. The individual position update is still the same as equation (9).

$$\delta_{\text{ij}} = \omega\left|\overrightarrow{x}_{\min} - C \times \text{rand}(1, D)\right|, \tag{10}$$

where δ_{ij} is a parameter of the Gaussian membership function [38, 39] and $\overrightarrow{x}_{\min}$ is the variate of the minimum value of the function.

2.2.2. The Dynamic Adaptive Gaussian Variation of Empirical Parameter. In the SOA, equation (8) is changed to equation (11), and the empirical value C_1 is changed to an adaptive empirical value that varies between 0.1 and 0.5 with the change of optimization algebra according to equation (12). The individual position update is still the same as equation (9).

$$\delta_{\text{ij}} = \omega\left|\overrightarrow{x}_{\min} - C_1 \times \text{rand}(1, D)\right|, \tag{11}$$

$$C_1 = 0.5 - t\left(\frac{0.1}{\text{iter}_{\max}}\right), \tag{12}$$

where δ_{ij} is a parameter of the Gaussian membership function [38, 39] and $\overrightarrow{x}_{\min}$ is the variate of the minimum value of the function.

2.2.3. The Chebyshev Chaos of Order Three. The Chebyshev map of order ω is defined as

$$x_{ij}^{*} = \cos\left(\omega \cdot \arccos x_{ij}\right), \quad -1 \le x_{ij} \le 1, \tag{13}$$

$$x_{\mathrm{ij}}(t+1) = x_{\mathrm{ij}}(t) + x_{\mathrm{ij}}^{*}(t), \quad i = 1, 2, \ldots, s;\ j = 1, 2, \ldots, D, \tag{14}$$

where when $\omega \ge 2$, x_{ij} is chaotic and ergodic and has orthogonality. In this case, no matter how close different initial values are, the sequences derived from multiple iterations are not correlated with each other.

2.2.4. The Multichain Strategy/the Double-Chain Strategy. The multichain strategy includes taking the real and the imaginary parts of the plural as separate parallel solutions and the randomly generating parallel solutions according to the complex number coding law.

In this paper, the meaning of the multichain strategy is that a single individual variable in the original SOA is converted into six parallel individual parameters when the CMSOA optimizes a problem. In complex-valued coding, there are real part X_{R}, imaginary part X_{I}, and real number X_{K}. In each iterative loop optimization, X_{R}, X_{I}, and X_{K} are adjusted to the variables that meet the scope of X ($X_{\min} = A_k$, $X_{\max} = B_k$). X_{R}, X_{I}, and X_{K} were taken as the relative optimal solution variables, respectively, to solve the objective function problem. Secondly, a group of variables that randomly generate $X_{\mathrm{R\_Random}}$, $X_{\mathrm{I\_Random}}$, and $X_{\mathrm{K\_Random}}$ according to formulas (9)–(11) and meet the scope of X ($X_{\min} = A_k$, $X_{\max} = B_k$) should be added in each cycle optimization and taken as the relative optimal solution variable to solve the objective function, respectively. At the end individual of the single solution, the respective optimal solutions are saved, and the global optimal value is saved as the current optimal value after the comparison of each optimal solution. The optimal solution variables of the next generation of X_{R}, X_{I}, and X_{K} are changed according to formulas (13)–(15). The next generation optimal solution variables of $X_{\mathrm{R\_Random}}$, $X_{\mathrm{I\_Random}}$, and $X_{\mathrm{K\_Random}}$ are generated randomly according to formulas (9)–(11). In other words, a single individual variable X in the original SOA is converted to six individual variables X_{R}, X_{I}, X_{K}, $X_{\mathrm{R\_Random}}$, $X_{\mathrm{I\_Random}}$, and $X_{\mathrm{K\_Random}}$ when solved by the CMSOA, and this is shown in Figure 1. So, instead of solving for one main chain, we are solving for six parallel chains. A multichain strategy is used in the CMSOA; the strategy adds the variety of the individual, enhances the local scout, and averts premature convergence.

For the SOA of real number coding, the real number coding is one chain, and the random generation of real number population is another chain. So, a double-chain is made up of a real number coding chain and a random generation of real number chain.

2.2.5. The Complex-Valved Encoding

(1) Initial Population Generation. In light of the variable interval $[A_k, B_k]$, $k = 1, 2, \ldots, 2s-1, 2s$, the modules ρ_k, the phase angles θ_k, and the plural are produced [40] as follows:

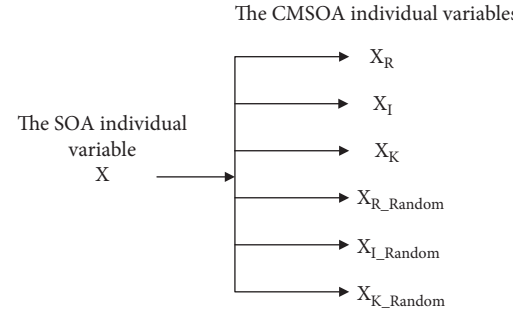


Figure 1: Multichain strategy of the CMSOA.

$$\rho_k = \frac{A_k, B_k}{2} \times \mathrm{rand} \in \left[0, \frac{A_k, B_k}{2}\right], \quad k = 1, 2, \ldots, 2s-1, 2s, \tag{15}$$

$$\theta_k = 4\pi(\mathrm{rand} - 0.5) \in [-2\pi, 2\pi], \quad k = 1, 2, \ldots, 2s-1, 2s, \tag{16}$$

$$X_{\mathrm{Rk}} + iX_{\mathrm{Ik}} = \rho_k\left(\cos\theta_k + i\sin\theta_k\right), \quad k = 1, 2, \ldots, 2s-1, 2s. \tag{17}$$

(2) Individual Location Updates. The real part is updated by

$$X_R(t+1) = X_R(t) + \alpha_R(t) f_R(t), \tag{18}$$

where α_R represents the scout direction of the real parts, f_R is the scout step measurement of the real parts, and X_R represents the location of the real number parts.

The imaginary part is updated by

$$X_I(t+1) = X_I(t) + \alpha_I(t) f_I(t), \tag{19}$$

where α_I represents the scout direction of the imaginary part, f_I is the scout step measurement of the imaginary part, and X_I represents the location of the imaginary number part.

(3) Fitness Evaluation Method. When calculating fitness values using the SOA, we convert plural to real numbers. The formula is as follows.

(1) Take the plural mathematical module as real number:

$$\rho_k = \sqrt{X_{\mathrm{Rk}}^2 + X_{\mathrm{Ik}}^2}, \quad k = 1, 2, \ldots, s-1, s. \tag{20}$$

(2) Define the sign according to the phase angle:

$$X_k = \rho_k \mathrm{sgn}\left(\sin\left(\frac{X_{\mathrm{Ik}}}{\rho_k}\right)\right)\frac{B_k + A_k}{2}, \quad k = 1, 2, \ldots, s-1, s, \tag{21}$$

where X_k is the real number.

3. The CMSOA Process

The chromosomes of complex organisms are regarded as double-stranded or multistranded construction. Since a complex value is made up of the real part and the imaginary part [26, 41–43], the complex value is represented as a

double-chain. A double-chain represents a chromosome pair, and the individuals that make up the double-chain have the same length. The two-body framework enhances the variety of individuals and makes the algorithm have better searching and calculation capacity.

The CMSOA is based on a multiple population evolution model, three populations evolved by the SOA, and three other populations evolved from random generation. The individual groups use the information-sharing mechanisms to realize coevolution. Algorithm 1 shows the primary process of the CMSOA.

4. Experimental Results

4.1. Experimental Setup. The algorithms used in the experiment in this paper were run under MATLAB R2016a. The computer is configured as Intel (R) Core (TM) i7-7500U CPU @2.7 GHz 2.9 GHz processor with 8 GB of memory, and the operating system is Windows 10.

4.2. Algorithm Performance Comparison in Benchmark Functions. To ensure that the comparison of these algorithms is fair, the population number of algorithms is 30, and the evolutionary algebra is 1000. At the same time, for further ensuring the fairness of algorithm comparison and reducing the effect of randomness, the results of the seven algorithms after 30 independent runs were selected for comparison.

4.2.1. The Benchmark Functions. In this field, it is common to base the capability of algorithms on mathematic functions that are known to be globally optimal. Fifteen benchmark functions in the literature are used as the comparative test platform [7, 10, 44–46]. Table 1 shows the functions in the experiment. Variables are set to one thousand.

4.2.2. Algorithm Performance Comparison of the SOA with Different Improvement Methods. In this paper, the SOA is improved by six different methods: the parameter changing SOA (PCSOA), the parameter adaptive Gaussian transform SOA (PAGTSOA), the SOA based on the Chebyshev chaos of order three (CCSOA), the SOA based on real coding double-link (DSOA), the SOA based on complex-valued encoding (CSOA), and the complex-valued encoding multichain seeker optimization algorithm (CMSOA).

(1) Parameter Setting of SOA with Different Improvement Methods. This section will introduce the parameter setting of the improved SOAs used in the experiment in this paper. Dai et al. have done a lot of research on the parameter set of the SOA [33], and we did a lot of practice tests and comparative studies about the parameters. The specific parameters of the improved SOA are shown in Table 2. In the next section, we use these improved SOAs for experimental comparison and choose a relatively optimal improved algorithm to compare with other advanced intelligent algorithms.

(2) Improved Algorithm Performance Comparison in the Benchmark Functions. The SOA is improved in six different ways: the parameter changing SOA (PCSOA), the parameter adaptive Gaussian transform SOA (PAGTSOA), the SOA based on the Chebyshev chaos of order three (CCSOA), the SOA based on real coding double-link (DSOA), the SOA based on complex-valued encoding (CSOA), and the complex-valued encoding multichain seeker optimization algorithm (CMSOA). To test the performance, each improved algorithm was optimized for the fifteen functions in Table 1. Each algorithm and each function were run independently 30 times. The performance of the SOA and the six improved SOAs in fifteen function optimizations was compared by the mean (Mean), standard deviation (Std.), best fitness (Best), the program running time (Time), and the best fitness rank (Rank) of 30 running results. The optimal fitness reflects the optimization accuracy of the algorithm, the average value and standard deviation reflect the robustness of the algorithms, and the running time reflects the time of the program. The results of functions f_1–f_{15} are displayed in Table 3. The values in bold and italics indicate that the optimal result is better.

Based on Table 3, for the benchmark functions f_1–f_{15}, the comparison between the seven improved SOAs in this paper and the original SOA shows that the optimization result of the CMSOA is the best value. The mean (Mean), standard deviation (Std.), best fitness (Best), and best fitness rank (Rank) of the CMSOA were the best after 30 independent runs. The total program running time of f_1–f_{15} ranks fifth among the seven algorithms compared in this paper. The running time of the CMSOA is longer than that of others algorithms. From the perspective of optimization accuracy and robustness, the CMSOA has the best optimization performance than these improved SOAs in this paper. Section 4.2.3 compares the CMSOA with other intelligent optimization algorithms widely used today.

(3) Search History of the CMSOA. Figure 2 shows the graph of the optimized function f_1, the convergence curves, the initial population's positions, and the search history; the search history behaviors of the search seekers are marked with red mark +. Based on Figure 2, for the benchmark function f_1, the convergence curve of the CMSOA is fast. From the search history of the CMSOA, the search seekers of the CMSOA extensively move towards promising search regions in the search space; the search seekers searched the given search space by the moment in the change search step size and different search directions; this gives the way to increase local search, escape local optima, and avoid premature convergence.

Similarly, Figure 3 shows the graph of the optimized function f_{10}, the convergence curves, the initial population's positions, and the search history; the search history behaviors of the search seekers are marked with red mark +. Based on Figure 3, for the benchmark function f_{10}, the convergence curve of the CMSOA is fast. From the search history of the CMSOA, the search seekers of the CMSOA extensively move towards promising search regions in the search space; the search seekers searched the given search

(1) $t \leftarrow 0$
(2) **Initialization**: generate initial species group based on formulas (15)–(17).
(3) Convert plural into real numbers based on formulas (20) and (21).
(4) Determine the range of $X_{\text{R\_CMSOA},G}$, $X_{\text{I\_CMSOA},G}$, and $X_{\text{CMSOA},G}$ to satisfy the range of X.
(5) **Evaluate** each seeker. Compute the fitness.
(6) While stopping condition is not satisfied
(6.1) Running process of the CMSOA
(6.1.1) Renew the real parts by formula (18), $X_{\text{R\_CMSOA},G}$.
(6.1.2) Renew the imaginary parts based on formula (19), $X_{\text{I\_CMSOA},G}$.
(6.1.3) Convert plural into real number based on formulas (20) and (21), $X_{\text{CMSOA},G}$.
(6.1.4) Determine the range of $X_{\text{R\_CMSOA},G}$, $X_{\text{I\_CMSOA},G}$, and $X_{\text{CMSOA},G}$ to satisfy the range of X.
(6.1.5) Scout strategy giving scout direction and scout range.
(6.1.6) Calculate the fitness $(X_{\text{R\_CMSOA},G})$, $(X_{I\text{\_CMSOA},G})$, $(X_{\text{CMSOA},G})$.
(6.1.7) Identify the best solution $X_{\text{CMSOAbest},G}$
$F_{\text{CMSOA,G}} = \min[f(X_{\text{R\_CMSOA},G})\ f(X_{\text{I\_CMSOA},G})\ f(X_{\text{CMSOA},G})]$
$X_{\text{CMSOAbest,G}} = \min(F_{\text{CMSOA,G}})$
(6.2) Random generation and calculation
(6.2.1) Generate Initial population according to formulas (15)–(17).
(6.2.2) Convert complex numbers into real numbers according to formulas (20) and (21).
(6.2.3) Determine the $X_{\text{R\_Random},G}$, $X_{\text{I\_Random},G}$, and $X_{\text{Random},G}$ to satisfy the range of X.
(6.2.4) Calculate the fitness $(X_{\text{R\_Random},G})$, $(X_{I\text{\_Random},G})$, $(X_{\text{Random},G})$.
(6.2.5) Identify the best value $X_{\text{Randombest,G}}$
$F_{\text{Random,G}} = \min[f(X_{\text{R\_Random},G})\ f(X_{\text{I\_Random},G})\ \text{f}(X_{\text{Random},G})]$
$X_{\text{Randombest},G} = \min(F_{\text{Random},G})$
(6.3) Confirm the global best value X_{best}
If $(X_{\text{CMSOAbest},G}) \le (X_{\text{Randombest},G})$
$X_{\text{best}} = X_{\text{CMSOAbest},G}$
else $X_{\text{best}} = X_{\text{Randombest},G}$
end if
(7) $t = t + 1$
(8) if $t < \text{iter}_{\max}$, then jump to 3; Else Stop.

Algorithm 1: CMSOA.

space by the moment in the change search step size and different search directions; this gives the way to increase local search, escape local optima, and avoid premature convergence.

Similarly, Figure 4 shows the graph of the optimized function f_{14}, the convergence curves, the initial population's positions, and the search history; the search history behaviors of the search seekers are marked with red mark +. Based on Figure 4, for the benchmark function f_{14}, the convergence curve of the CMSOA is the fast. From the search history of the CMSOA, the search seekers of the CMSOA extensively move towards promising search regions in the search space; the search seekers searched the given search space by the moment in the change search step size and different search directions; this gives the way to increase local search, escape local optima, and avoid premature convergence.

4.2.3. The Algorithm Performance Comparison of Different Algorithms in the Benchmark Functions. To test the performance of the CMSOA, the CMSOA is compared with the PSO, SA-GA, GSA, SCA, MVO, and SSA, using fifteen benchmark functions [7, 10, 44–46] in Table 1, which have been widely used in the test.

(1) The Parameter Setting of Different Algorithms. In this section, the parameter set of the PSO [47], SA_GA [48], GSA [6], SCA [8], MVO [9], SOA [29], and CMSOA is presented. According to references [6, 8, 23, 29, 47, 48], we did a lot of practice tests and comparative studies for the parameter set. The parameters of the seven algorithms depend on the real experience to take the right value. Table 4 lists the parameters in the test.

(2) The Result Comparison of Different Algorithms in Benchmark Functions. This section uses the same fifteen functions as in Table 1, but we have expanded the dimension of the variables to 1000 dimensions. The mean values, standard deviation, best fitness, and best fitness rank of the algorithms of 30 all-alone runs and the data of optimization results of functions f_1–f_{15} are shown in Table 5. The values in bold and italics indicate that the optimal outcome is better.

For the benchmark functions f_1–f_{15}, based on Table 5, except f_4, f_7, f_9, f_{10}, f_{11}, f_{14}, and f_{15}, the optimal value of the CMSOA is better than that of the others. To f_9, the optimal value of the CMSOA has reached the theoretical best value, although the optimal fitness value of the CMSOA is inferior to that of the PSO and the GSA. For f_7 function, the optimal value of the CMSOA is only worse than that of the PSO

Table 1: Description of the benchmark functions.

Name	Test functions	Search	Minimum
Sphere	$f_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]$	0
Schwefel 2.22	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]$	0
Rotated hyperellipsoid	$f_3(x) = \sum_{i=1}^{n} \sum_{j=1}^{i} x_j^2$	$[-100, 100]$	0
Schwefel 2.21	$f_4(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	$[-100, 100]$	0
Rosenbrock	$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 (x_i - 1)^2]$	$[-30, 30]$	0
Step	$f_6(x) = \sum_{i=1}^{n} (\lvert x_i + 0.5 \rvert)^2$	$[-100, 100]$	0
Quartic	$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	$[-1.28, 1.28]$	0
SumSquares	$f_8(x) = \sum_{i=1}^{n} i x_i^2$	$[-10, 10]$	0
SumPower	$f_9(x) = \sum_{i=1}^{n} \lvert x_i \rvert^{(i+1)}$	$[-1.28, 1.28]$	0
Schwefel	$f_{10}(x) = \sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500]$	$-418.9829 * \text{Dimension}$
Rastrigin	$f_{11}(x) = \sum_{i=1}^{n} [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[-5.12, 5.12]$	0
Ackley	$f_{12}(x) = 20 + e - 20e^{-0.2\sqrt{1/n \sum_{i=1}^{n} x_i^2}} - e^{1/n \sum_{i=1}^{n} \cos(2\pi x_i)}$	$[-32, 32]$	0
Griewank	$f_{13}(x) = 1/4000 \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1$	$[-600, 600]$	0
Penalized1	$f_{14}(x) = \pi/n \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, $y_i = 1 + (1/4)(x_i + 1)$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a \le x_i \le a \\ (-x_i - a)^m x_i < -a \end{cases}$	$[-50, 50]$	0
Penalized2	$f_{15}(x) = 1/10 \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \right\} a + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]$	0

Table 2: Parameter settings of the SOA with different improvement methods.

Algorithm	Parameters and values
SOA [5]	The maximum membership degree value: $U_{max} = 0.95$, the minimum membership degree value: $U_{min} = 0.0111$, the maximum inertia weight value: $W_{max} = 0.9$, and the minimum inertia weight value: $W_{min} = 0.1$
PCSOA	The maximum membership degree value: $U_{max} = 0.95$, the minimum membership degree value: $U_{min} = 0.0111$, the maximum inertia weight value: $W_{max} = 0.9$, the minimum inertia weight value: $W_{min} = 0.1$, and the empirical value: $W = 0.2$
PAGTSOA	The maximum membership degree value: $U_{max} = 0.95$, the minimum membership degree value: $U_{min} = 0.0111$, the maximum inertia weight value: $W_{max} = 0.9$, the minimum inertia weight value: $W_{min} = 0.1$, and the empirical value: $W = 0.1$–0.5
CCSOA	The maximum membership degree value: $U_{max} = 0.95$, the minimum membership degree value: $U_{min} = 0.0111$, the maximum inertia weight value: $W_{max} = 0.9$, the minimum inertia weight value: $W_{min} = 0.1$, and the empirical value: $W = 0.2$
DSOA	The maximum membership degree value: $U_{max} = 0.95$, the minimum membership degree value: $U_{min} = 0.0111$, the maximum inertia weight value: $W_{max} = 0.9$, the minimum inertia weight value: $W_{min} = 0.1$, and the empirical value: $W = 0.2$
CSOA	The maximum membership degree value: Umax = 0.95, the minimum membership degree value: Umin = 0.0111, the maximum inertia weight value: Wmax = 0.9, the minimum inertia weight value: Wmin = 0.1, and the empirical value: $W = 0.2$
CMSOA	The maximum membership degree value: $U_{max} = 0.95$, the minimum membership degree value: $U_{min} = 0.0111$, the maximum inertia weight value: $W_{max} = 0.9$, the minimum inertia weight value: $W_{min} = 0.1$, and the empirical value: $W = 0.2$

Table 3: Performance comparison of the SOA improved by different strategies.

Function	Result	Algorithms						
		SOA	PCSOA	PAGSOA	CCSOA	DSOA	CSOA	CMSOA
f_1 ($D = 100$)	Mean	1.0524957	10.651095	5.8792970	0.3009934	0.3532587	*0.1116982*	0.2077269
	Std.	0.2051906	49.006900	26.483115	0.2488472	0.2958520	0.4738650	*0.1376184*
	Best	0.6215967	0.0194690	0.0126392	0.0042290	0.0461541	0.0088426	*0.0016885*
	Time	*45.50061*	55.33285	53.03482	59.300793	309.13098	367.64198	389.55755
	Rank	7	5	4	2	6	3	**1**
f_2 ($D = 100$)	Mean	13.002085	0.8551306	1.1247211	0.8780035	0.7974362	0.6564412	*0.6368774*
	Std.	2.0008395	0.3078019	0.1241726	0.1928515	0.1878251	*0.0956433*	0.1775901
	Best	9.4729280	*0.3884975*	0.8443990	0.5441874	0.5156629	0.3976921	0.3978455
	Time	54.16754	51.23705	*49.17031*	64.507580	365.14220	353.13727	363.79830
	Rank	7	**1**	6	5	4	2	3
f_3 ($D = 100$)	Mean	$9.869e+03$	$3.392e+03$	$3.452e+03$	$2.622e+03$	$2.7367e+3$	$6.6670+3$	*1.1027e + 3*
	Std.	$3.472e+03$	$1.658e+03$	$1.959e+03$	$2.016e+03$	$2.0207e+3$	$3.7306e+3$	*8.3009e + 2*
	Best	$3.318e+03$	$4.645e+02$	37.813122	*2.3714625*	$1.7054e+2$	14.746709	6.4672568
	Time	423.27283	*393.18358*	468.54936	491.11858	4151.7979	1620.2993	1260.7060
	Rank	7	6	4	**1**	5	3	2
f_4 ($D = 100$)	Mean	21.743821	18.412017	18.418584	17.524233	17.793596	31.7111918	*16.484619*
	Std.	5.6809434	6.8890641	6.6275327	7.6897380	6.5363678	*2.5395843*	5.5082548
	Best	2.9045088	*0.2149959*	0.5369130	0.2588117	0.7984465	26.660747	2.6256219
	Time	*47.98446*	55.85934	49.86405	52.858896	325.71654	370.6040	431.75863
	Rank	6	**1**	3	2	4	7	5
f_5 ($D = 100$)	Mean	$7.093e+02$	$1.334e+02$	$1.082e+02$	$1.017e+02$	$2.4188e+3$	*98.533551*	$1.0088e+2$
	Std.	$1.675e+02$	$1.489e+02$	14.576662	7.7724141	$1.2306e+4$	*3.0082579*	11.123575
	Best	$2.789e+02$	96.133568	97.022113	96.548173	96.290670	96.158695	*95.869293*
	Time	54.515291	62.169550	*50.996834*	63.045733	439.86830	452.46103	416.64079
	Rank	7	2	6	5	4	3	**1**
f_6 ($D = 100$)	Mean	*1.1365348*	10.241728	8.3207707	9.4211577	15.867512	7.0792797	9.8327363
	Std.	*0.1751583*	7.3068384	4.9997548	4.0535203	14.622004	4.5554408	2.4321441
	Best	0.7992740	0.2161040	*0.0424130*	0.4878507	1.3808158	0.3954699	2.3988934
	Time	47.876927	52.583180	*46.059290*	58.970068	401.34745	473.79587	366.27538
	Rank	5	2	**1**	4	6	3	7
f_7 ($D = 100$)	Mean	3.4645595	0.0979540	0.1900503	0.0921230	0.1046785	0.1017466	*0.0909504*
	Std.	0.9443482	0.0295516	0.0493722	0.0295674	0.0303486	0.0279235	*0.0250917*
	Best	2.0472780	0.0497600	0.0989682	*0.0423312*	0.0625095	0.0665980	0.0546596
	Time	90.52067	95.46660	*85.11105*	159.51879	966.15743	671.42939	531.30085
	Rank	7	2	6	**1**	4	5	3

Table 3: Continued.

Function	Result	Algorithms						
		SOA	PCSOA	PAGSOA	CCSOA	DSOA	CSOA	CMSOA
f_8 ($D = 100$)	Mean	$1.391e+02$	0.3830840	1.2523149	0.5511988	0.4522278	*0.269201*	0.2693774
	Std.	33.925547	0.1176738	0.2579516	0.5240817	0.3370242	0.079379	*0.0710191*
	Best	79.288776	0.2063376	0.7213791	0.1941864	0.1860271	0.125793	*0.1116089*
	Time	55.51168	51.83529	*47.67628*	64.360044	394.86795	410.81844	273.29456
	Rank	7	5	6	4	3	2	**1**
f_9 ($D = 100$)	Mean	$1.471e-05$	$2.988e-08$	$1.597e-07$	$4.0858e-8$	$2.3918e-08$	$5.8221e-08$	*$1.7706e-8$*
	Std.	$1.279e-05$	$1.944e-08$	$1.098e-07$	$2.8308e-8$	$1.8113e-08$	$3.8119e-08$	*$1.2641e-8$*
	Best	$6.318e-07$	$4.835e-09$	$9.134e-09$	$3.6158e-9$	$4.6751e-09$	$1.7833e-08$	*$1.4250e-9$*
	Time	102.88770	105.16369	*94.27801*	131.1281	1015.2811	536.45466	411.95393
	Rank	7	5	4	2	3	6	**1**
f_{10} ($D = 100$)	Mean	$-2.324e+4$	$-2.184e+4$	$-2.296e+4$	$-2.302e+4$	$-2.2314e+4$	$-2.2379e+4$	*$-3.125e+4$*
	Std.	$3.396e+3$	$3.105e+3$	$3.478e+3$	*$2.2286e+3$*	$3.5627e+3$	$3.0105e+3$	$3.8419e+3$
	Best	$-3.174e+4$	$-2.906e+4$	$-3.383e+4$	$-2.787e+4$	$-3.7082e+4$	$-2.8728e+4$	*$-4.127e+4$*
	Time	*60.474337*	64.865537	69.718244	85.194918	542.92266	410.10538	335.93689
	Rank	4	5	3	7	2	6	**1**
f_{11} ($D = 100$)	Mean	$4.147e+02$	40.408968	$1.671e+02$	31.424385	22.010136	27.498996	*12.314455*
	Std.	47.133648	58.557280	52.557303	44.425051	31.162826	41.300447	*11.507374*
	Best	$3.230e+02$	*0.1867154*	49.014907	0.3522364	0.2567124	0.1969691	0.2647166
	Time	59.59937	70.80121	*53.881313*	63.770943	410.67985	400.34173	344.88583
	Rank	7	**1**	6	5	3	2	4
f_{12} ($D = 100$)	Mean	2.4523846	0.4491129	0.2960575	0.3149221	0.5812064	*0.1151882*	0.1668715
	Std.	0.3106584	0.6639778	0.5909855	0.9299543	0.9919764	0.1091937	*0.0811535*
	Best	1.9634230	*0.0154282*	0.0320302	0.0160403	0.0232704	0.0168475	0.0230998
	Time	64.89326	68.48468	*56.26708*	68.939722	481.161501	426.233718	309.17212
	Rank	7	**1**	6	2	5	3	4
f_{13} ($D = 100$)	Mean	0.5354222	0.9578562	0.143592	0.2880654	0.4551868	0.0607244	*0.0334294*
	Std.	0.3343660	3.0280038	0.336573	0.5765596	1.0663812	0.1861735	*0.1280935*
	Best	0.0806019	0.0020953	0.007806	0.0021233	*0.0016367*	0.0024141	0.0017944
	Time	63.67314	65.48267	*60.59472*	88.145403	525.980833	517.816804	386.16642
	Rank	7	3	6	4	**1**	5	2
f_{14} ($D = 100$)	Mean	20.231984	11.239536	13.286278	15.089459	11.896348	15.510852	5.3014263
	Std.	8.8688013	6.6059206	8.0516954	8.2104226	7.1902974	6.5123009	5.7168081
	Best	8.9451797	0.3752616	0.2490416	0.4653713	0.1050871	1.6355938	*0.0700603*
	Time	165.94752	214.22263	*158.94344*	188.54280	2343.8473	1101.5058	639.49732
	Rank	7	4	3	5	2	6	**1**
f_{15} ($D = 100$)	Mean	$1.301e+02$	$1.080e+02$	95.968154	82.716420	82.827205	82.843043	*44.937907*
	Std.	62.979168	78.512763	75.121239	70.432659	70.510311	61.171959	*49.375729*
	Best	*1.7998261*	9.9178264	9.7032945	9.0159025	9.8337273	9.9984043	9.1050026
	Time	166.65126	164.31928	*159.80885*	187.79078	1834.4237	759.75083	576.75249
	Rank	**1**	6	4	2	5	7	3
Average rank		6.2	3.266666667	4.533333333	3.4	3.8	4.2	**2.6**
Overall rank		7	2	6	3	4	5	**1**

algorithm. The optimal fitness value result of the CMSOA to f_4 is worse than that of the PSO, GSA, and SOA. The optimal fitness value results of the CMSOA for f_{15} function are only worse than those of the PSO algorithm, the optimal fitness value result of the CMSOA for f_{14} function is worse than that of the PSO and SOA, the optimal fitness value result of the CMSOA for f_{11} function is worse only than that of the SCA algorithm, and the result of the CMSOA for f_{10} is worse than that of the SOA and the MVO.

For the benchmark functions f_1–f_{15}, based on Table 5, except f_2, f_3, f_4, f_7, f_9, f_{10}, f_{11}, f_{12}, and f_{14}, the standard deviation results of the CMSOA are better than those of the others. The standard deviation results of the CMSOA for f_9 are only worse than those of the PSO algorithm, the result of the CMSOA for f_7 is worse than that of the PSO, GSA, and SOA, the result of the CMSOA for f_4 is worse than that of the SCA, GSA, SA_GA, PSO, and the MVO, the result of the CMSOA for f_3 is worse than that of the MVO and SOA, and the result of the CMSOA for f_2 is only worse than that of the SOA. To f_2 function, the standard deviation results of the PSO, SA_GA, and the SCA algorithm have no solution, and the GSA and the MVO algorithm have an infinite standard deviation. The CMSOA is better than the others. For f_{14}, the CMSOA is worse than the PSO and the GSA. For f_{11} and f_{12}, the CMSOA is worse than the PSO, SA_GA, GSA, MVO, and SOA, and the standard deviation results of the CMSOA

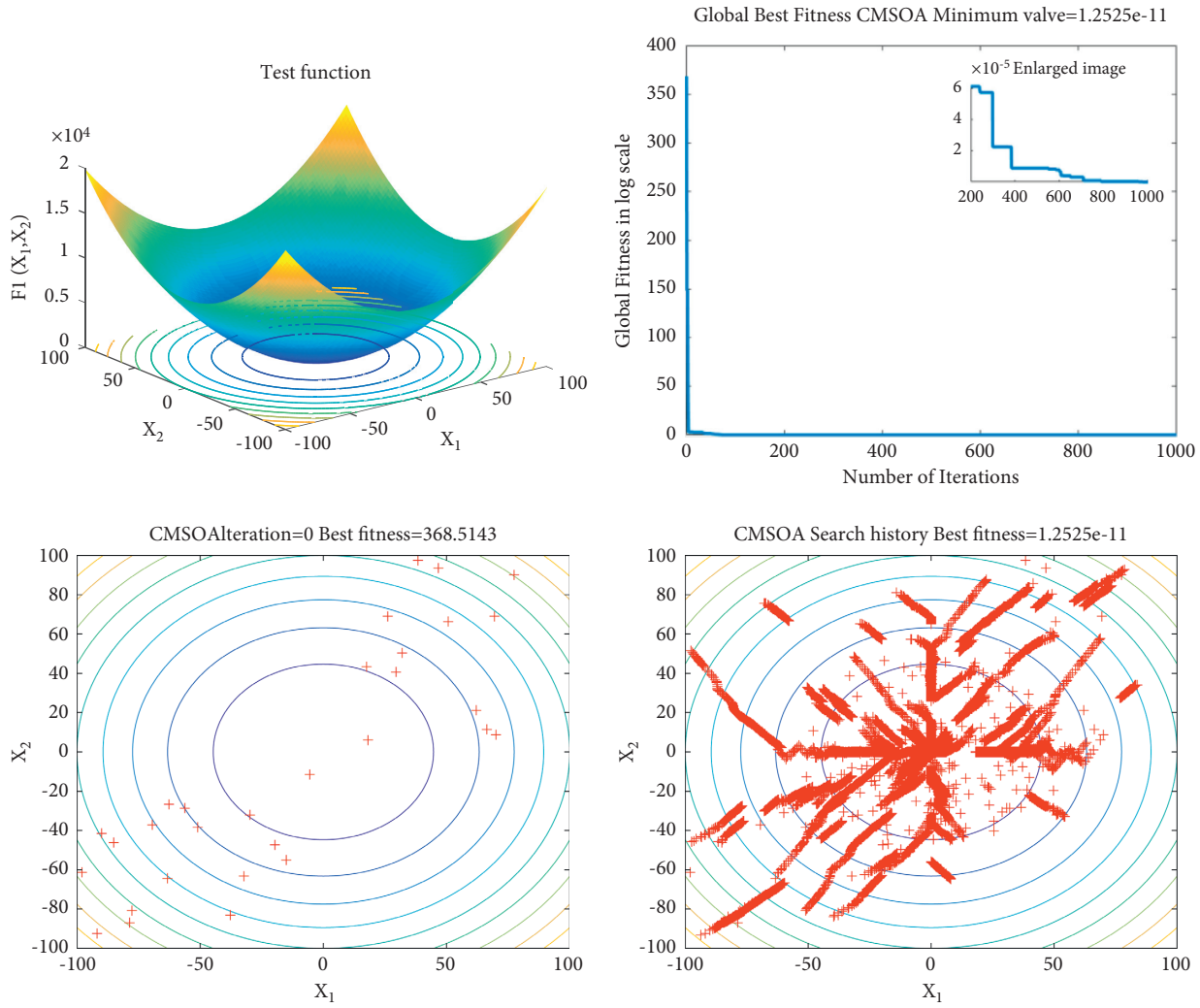


Figure 2: The graph of f_1, convergence curve, initial population's positions, and the search history.

for f_{10} function are worse than those of the PSO, SA_GA, GSA, SCA, and the MVO algorithm.

For the benchmark functions f_1–f_{15}, based on Table 5, except $f_3, f_4, f_7, f_9, f_{10}$, and f_{14}, the mean values of the CMSOA are better than those of the others. For f_9, the mean test results of the CMSOA have reached the theoretical best value; although the mean test result of the CMSOA is worse than that of the PSO, the result of the CMSOA for f_7 is worse than that of the PSO, the result of the CMSOA for f_4 is worse than that of the PSO, GSA, and SOA, and the result of the CMSOA for f_3 is only worse than that of the GSA and SOA. The CMSOA is better than the others, for f_{10}, the CMSOA is worse than the MVO and SOA, and for f_{14}, the CMSOA is only worse than the PSO algorithm.

According to the optimal fitness value mean rank and all rank results from Table 5, the CMSOA can find solutions and has strong optimization ability and strong robustness to benchmark function.

(3) Convergence Curve Comparison of Algorithms in the Benchmark Functions. Figure 5 shows the fitness curves of the best fitness for the benchmark functions f_1–f_{15} ($D = 1000$). As seen from Figure 5, the CMSOA is compared to the other six algorithms; the convergence of the CMSOA is faster, and the precision of the CMSOA is better, except $f_4, f_7, f_9, f_{10}, f_{14}$, and f_{15}. Although the CMSOA for f_9 is worse than the PSO in terms of convergence and the precision, the CMSOA has reached the theoretical best value, for f_7, the CMSOA is only worse than the PSO, and for f_4, the CMSOA is worse than the PSO, GSA, and SOA. The CMSOA for f_{15} is only worse than the SOA in terms of convergence and precision, for f_{14}, the CMSOA is worse than the SOA and the PSO, and for f_{10}, the CMSOA is worse than the MVO and SOA. Because of the multichain strategy to augment the individuals' diversity and local scout intensity, the CMSOA has better optimization property.

(4) ANOVA Test Comparison of Algorithms in Benchmark Functions. Figure 6 shows the ANOVA of the global best values for benchmark functions f_1–f_{15} ($D = 1000$). As seen from Figure 6, the CMSOA is the most robust, except $f_3, f_4, f_7, f_{10}, f_{12}$, and f_{14}. The ANOVA test results of the CMSOA for f_7 are only worse than those of the PSO algorithm, the result of the CMSOA for f_4 is worse than that of the PSO, GSA, and SOA, and the result of the CMSOA for f_3 function is only worse than that of the GSA and SOA. The ANOVA test results of the

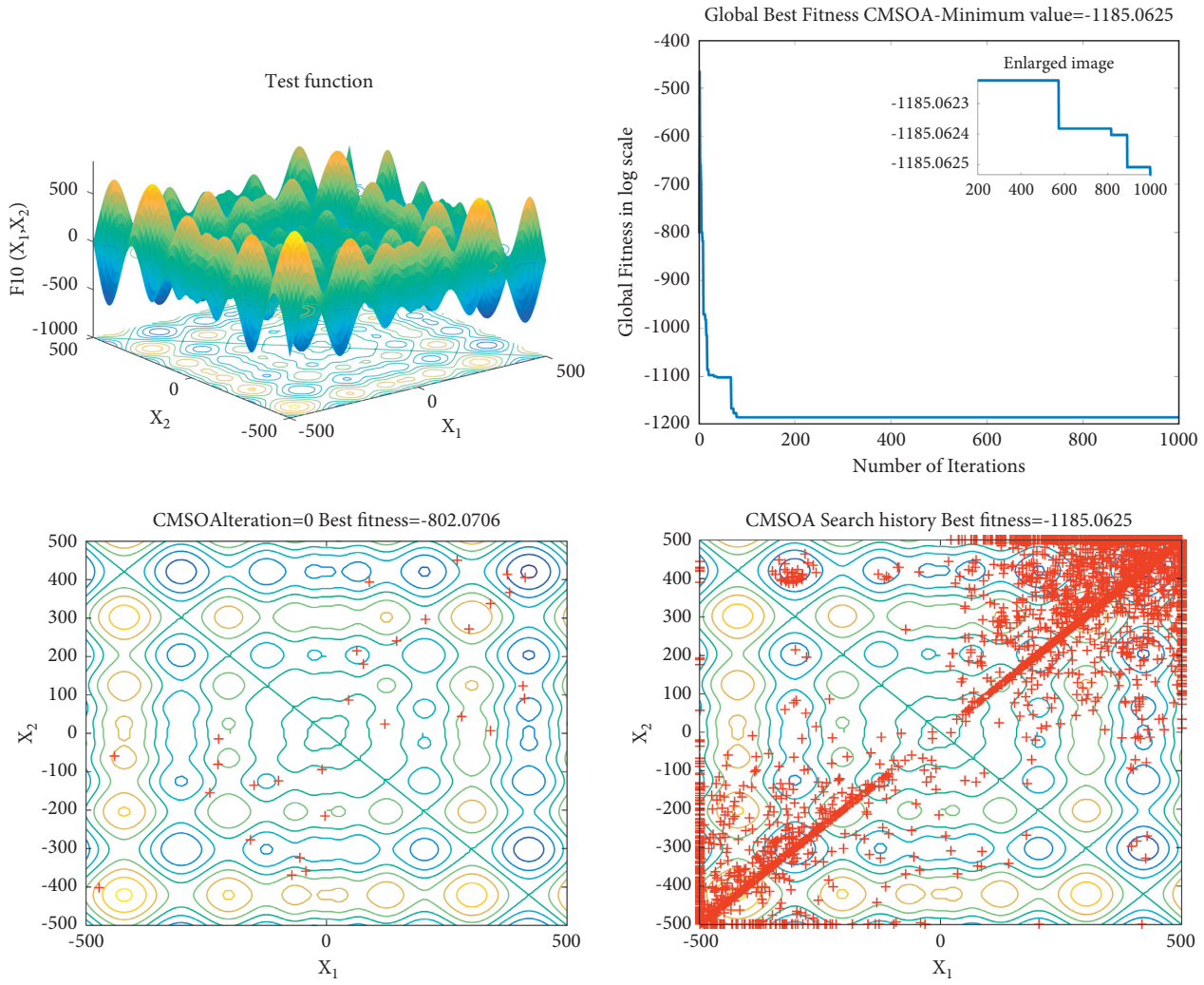


Figure 3: The graph of f_{10}, convergence curve, initial population's positions, and the search history.

CMSOA for f_{10} function are only worse than those of the MVO algorithm, and the results of f_{12} and f_{14} are worse than that of the PSO and SOA. The CMSOA showed better robustness.

4.2.4. Complexity Analysis. The calculation complexity of the basic SOA is O ($N.D.M$), where N is the total individual count, D is the dimension count, and M is the maximum count of algebras. The computational complexity of the first phase of the SOA stage is O ($N.D.M$). The complex coding strategy is introduced to calculate the O ($N.D.M$) value. The introduced multichain strategy's calculational complexity value is O ($N.D.M$). So, the overall complexity of the CMSOA is O ($N.D.M + N.D.M + N.D.M$). Based on the principle of Big-O representation [49], if the count of algebras is high ($M \gg N$, D), the calculational complexity is O ((N.D.M). Therefore, the overall calculational complexity of the CMSOA is almost the same as the basic SOA.

4.2.5. Run Time Comparison of Algorithms in Benchmark Functions. In this section, we recorded the running time of each algorithm under the same conditions: population number 30, evolution algebra 1000, and 30 independent runs of the above fifteen benchmark functions f_1–f_{15} ($D = 1000$). Then, the running time of the fifteen functions is added to obtain the sum of the 30 independent running times of each algorithm for the fifteen functions listed in this paper and the ranking of the total time, as shown in Table 6. As seen from Table 6, the PSO algorithm has the most minor program running time, followed by the SCA algorithm, which has more minor program running time, and the CMSOA ranks sixth, which has relatively more program running time. At the bottom of the list is the SA_GA algorithm, which takes the most running time.

To learn more about the program running time of the seven algorithms in the fifteen functions, a bar chart (Figure 7) was made for the total time of each algorithm after 30 independent runs. From Figure 7, the program running time of the PSO is the least, while that of the SA_GA algorithm is the most,

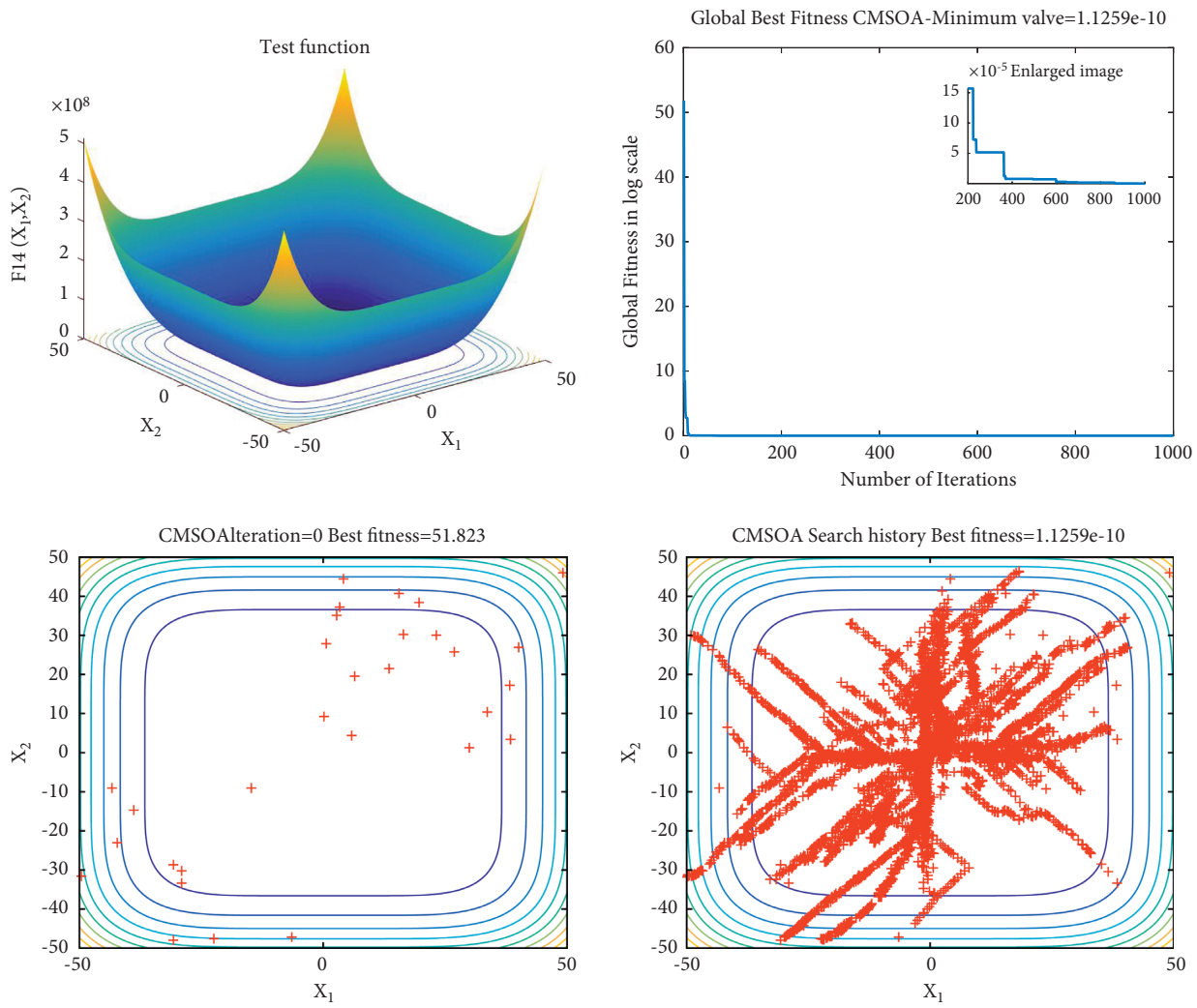


Figure 4: The graph of the f_{14}, convergence curve, initial population's positions, and the search history.

and the program running time of the CMSOA is less than half of that of the SA_GA algorithm, which is relatively large.

4.2.6. Exploration and Exploitation in Benchmark Functions. According to [50–52], formulas (22)–(25) represent the exploration and development capability of an algorithm.

$$\mathrm{Div}_j = \frac{1}{n}\sum_{i=1}^{n} \mathrm{median}\left(x^j\right) - x_i^j, \tag{22}$$

$$\mathrm{Div} = \frac{1}{D}\sum_{i=1}^{D} \mathrm{Div}_j, \tag{23}$$

$$\mathrm{Xpl\%} = \frac{\mathrm{Div}}{\mathrm{Div}_{\max}} \times 100, \tag{24}$$

$$\mathrm{Xpt\%} = \frac{\left|\mathrm{Div} - \mathrm{Div}_{\max}\right|}{\mathrm{Div}_{\max}} \times 100, \tag{25}$$

where median x^j is the median of dimension j in whole swarm, x_i^j is the dimension j of the swam individual i, n is the size of swarm, Div_j is the average for all the individuals, Div is the diversity of swarm in an iteration, $\mathrm{Div}_{\max}$ is the maximum diversity in all iterations, and Xpl% and Xpt% are the exploration and exploitation percentages for an iteration, respectively.

Figure 8 shows the exploration and exploitation abilities of the CMSOA as the number of iterations increases in the benchmark functions f_1–f_{15}. As observed from the plotted curves shown in Figure 8, the CMSOA maintains good balance between the exploration and exploitation ratios as the number of iterations increases.

4.2.7. Performance Profiles of Algorithms in Benchmark Functions. The average fitness was selected as the capability index. The algorithmic capability is expressed in performance profiles, which are calculated by the following formulas:

Table 4: The parameter set of algorithms.

Algorithm	Parameters and values
PSO [47]	Constant inertia: 0.9~0.4, the two acceleration coefficients: 1.4962
SA_GA [48]	Select probability: 0.6, crossover probability: 0.7, mutation scale factor: 0.05, original temperature: 100, temperature reduction parameter: 0.98
GSA [6]	The gravitational constant: G0 = 100, $\alpha = 20$
SCA [8]	The random numbers: r1 = 0~2, r2 = 0~2π, r3 = 0~2, r4 = 0~1
MVO [9]	Probability of wormhole existence: WEP_Max = 1, WEP_Min = 0.2, travelling distance rate: TDR = 0~1, the random numbers: r1 = 0~1, r2 = 0~1, r3 = 0~1
SOA [29]	The membership degree value: MDV_Max = 0.95, MDV_Min = 0.0111, the inertia weight value: IWV_Max = 0.9, IWV_Min = 0.1
CMSOA	The membership degree value: MDV_Max = 0.95, MDV_Min = 0.0111, the inertia weight value: IWV_Max = 0.9, IWV_Min = 0.1

Table 5: Performance comparison of algorithms for benchmark functions.

Functions	Result	Algorithm						
		PSO	SA_GA	GSA	SCA	MVO	SOA	CMSOA
f_1 ($D = 1000$)	Mean	9.7477$e+3$	2.5731$e+6$	9.7519$e+4$	3.5105$e+5$	3.5977$e+5$	1.1631$e+3$	*2.7677e+2*
	Std.	1.6106$e+3$	5.6740$e+4$	4.9089$e+3$	1.3646$e+5$	1.6121$e+4$	2.1241$e+3$	*1.1208e+3*
	Best	6.9495$e+3$	2.4536$e+6$	8.7230$e+4$	4.0211$e+4$	3.3044$e+5$	69.63801	*13.41123*
	Rank	3	7	5	4	6	2	**1**
f_2 ($D = 1000$)	Mean	Inf	Inf	1.8010$e+281$	Inf	1.3436$e+273$	1.3642$e+2$	*84.4833*
	Std.	NaN	NaN	Inf	NaN	Inf	*5.69798*	6.8676
	Best	2.9025$e+2$	Inf	4.6083$e+244$	Inf	2.3145$e+209$	1.2392$e+2$	*74.4520*
	Rank	3	6	5	6	4	2	**1**
f_3 ($D = 1000$)	Mean	3.3245$e+6$	3.3451$e+7$	1.9611$e+6$	2.4026$e+7$	6.1928$e+6$	*1.3029e+6*	2.0443$e+6$
	Std.	1.4068$e+6$	1.1343$e+7$	8.1529$e+5$	4.7320$e+6$	*4.3229e+5*	4.9041$e+5$	7.7353$e+5$
	Best	1.6748$e+6$	1.9532$e+7$	9.5585$e+5$	1.3982$e+7$	5.5268$e+6$	1.8272$e+5$	*1.7429e+4*
	Rank	4	7	3	6	5	2	**1**
f_4 ($D = 1000$)	Mean	*18.3429*	99.5440	28.7444	99.5223	97.4126	28.6012	61.3865
	Std.	1.3182	0.1918	1.7962	*0.1313*	0.7405	10.4316	3.1861
	Best	16.3774	98.9588	25.8834	99.1965	95.6086	*2.3434*	53.9155
	Rank	2	6	3	7	5	**1**	4
f_5 ($D = 1000$)	Mean	2.7249$e+5$	1.119$e+10$	1.6750$e+7$	3.2676$e+9$	6.7902$e+8$	2.2675$e+5$	*4.8206e+4*
	Std.	1.4792$e+5$	3.343$e+8$	1.7268$e+6$	7.0048$e+8$	7.3930$e+7$	1.2833$e+5$	*2.7505e+4*
	Best	1.2499$e+5$	1.057$e+10$	1.4932$e+7$	2.0841$e+9$	5.4843$e+8$	2.8880$e+4$	*1.1542e+4*
	Rank	3	7	4	6	5	2	**1**
f_6 ($D = 1000$)	Mean	1.0284$e+4$	2.5635$e+6$	9.8733$e+4$	4.1315$e+5$	3.5579*e5*	1.3438$e+3$	*2.7896e+2*
	Std.	1.3817$e+3$	4.0795$e+4$	4.9212$e+3$	1.2787$e+5$	1.7687$e+4$	2.2562$e+3$	*66.6179*
	Best	6.1887$e+3$	2.4668$e+6$	9.0772$e+4$	1.6278$e+5$	3.1703$e+5$	2.5694$e+2$	*2.1193e+2*
	Rank	3	7	4	5	6	2	**1**
f_7 ($D = 1000$)	Mean	*1.1474e+2*	1.7871$e+5$	5.2986$e+3$	4.7647$e+4$	8.7236$e+3$	5.1827$e+3$	2.2353$e+3$
	Std.	*22.2657*	6.0811$e+3$	5.8169$e+2$	1.1048$e+4$	771.5517	6.5217$e+2$	7.6850$e+2$
	Best	*80.9849*	1.6720$e+5$	4.37029$e+3$	2.6553$e+4$	7.2739$e+3$	3.7872$e+3$	1.0774$e+3$
	Rank	**1**	7	4	6	5	3	2
f_8 ($D = 1000$)	Mean	4.7291$e+4$	1.2547$e+7$	3.8862$e+5$	1.8012$e+6$	1.5454$e+6$	2.6560$e+4$	*7.9139e+3*
	Std.	6.6212$e+3$	3.5539$e+5$	2.2329$e+4$	5.5949$e+5$	6.4511$e+4$	3.9984$e+3$	*1.4647e+3*
	Best	3.6574$e+4$	1.1381$e+7$	3.5017$e+5$	4.5453$e+5$	1.4470$e+6$	1.8651$e+4$	*4.4462e+3*
	Rank	3	7	4	5	6	2	**1**
f_9 ($D = 1000$)	Mean	*9.7670e−9*	1.5616$e+92$	7.6578$e−5$	9.1116$e+83$	1.0103$e+56$	0.3859	2.2868$e−6$
	Std.	*3.3014e−8*	7.3622$e+92$	2.5647$e−4$	4.6589$e+84$	5.5321$e+56$	0.6475	1.7299$e−6$
	Best	*5.497e−16*	1.1184$e+7$	1.9892$e−9$	9.2943$e+69$	5.6083$e+38$	2.0490$e−5$	4.4192$e−7$
	Rank	**1**	5	2	7	6	4	3
f_{10} ($D = 1000$)	Mean	−1.701$e+4$	−5.8434$e+4$	−1.474$e+4$	−2.2849$e+4$	*−1.339e+5*	−1.181$e+5$	−7.995$e+4$
	Std.	2.5847$e+3$	3.4159$e+3$	2.3262$e+3$	*1.3757e+3*	*5.9809e+3*	3.0107$e+4$	1.5591$e+4$
	Best	−2.155$e+4$	−6.7056$e+4$	*−1.844e+4*	−2.665$e+4$	−1.467$e+5$	*−2.157e+5*	−1.178$e+5$
	Rank	6	4	7	5	2	***1***	3

Table 5: Continued.

Functions	Result	Algorithm						
		PSO	SA_GA	GSA	SCA	MVO	SOA	CMSOA
f_{11} ($D = 1000$)	Mean	2.8217$e+3$	1.5245$e+4$	5.7869$e+3$	1.8362$e+3$	1.3788$e+4$	6.3358$e+3$	*1.7847e + 3*
	Std.	1.7027$e+2$	*1.3944e + 2*	1.5766$e+2$	874.1338	300.8269	5.0727$e+2$	5.5996$e+2$
	Best	2.5372$e+3$	1.4955$e+4$	5.4142e+3	*502.646*	1.3250$e+4$	5.1713$e+3$	1.0788$e+3$
	Rank	3	7	5	**1**	6	4	2
f_{12} ($D = 1000$)	Mean	4.4947	20.7977	10.3286	18.8639	20.9497	3.1246	*1.5427*
	Std.	0.1746	0.0267	0.1538	4.0427	*0.0222*	0.3837	0.9780
	Best	4.2442	20.7492	9.9359	8.4914	20.8992	2.3556	*0.9571*
	Rank	3	6	5	4	7	2	**1**
f_{13} ($D = 1000$)	Mean	1.0210$e+3$	2.3100$e+4$	1.3996$e+4$	3.1433$e+3$	3.2496$e+3$	16.2019	*1.6504*
	Std.	32.8774	5.1266$e+2$	2.1571$e+2$	1.0529$e+3$	169.9815	27.1817	*5.2240*
	Best	9.5388$e+2$	2.1976$e+4$	1.3539$e+4$	1.1471$e+3$	2.8639$e+3$	0.19051	*0.0449*
	Rank	3	7	6	4	5	2	**1**
f_{14} ($D = 1000$)	Mean	*3.5824*	2.6235$e+10$	3.8689$e+4$	9.5072$e+9$	9.0076$e+8$	2.0865$e+4$	2.0766$e+4$
	Std.	*0.4677*	9.4178$e+8$	2.4728$e+4$	2.0642$e+9$	1.0468$e+8$	4.1982$e+4$	3.6384$e+4$
	Best	2.7478	2.4014$e+10$	5.6569$e+3$	5.1822$e+9$	7.1553$e+8$	*1.3229*	44.7877
	Rank	2	7	4	6	5	**1**	3
f_{15} ($D = 1000$)	Mean	1.4226$e+4$	4.9243$e+10$	6.1049$e+6$	1.6105$e+10$	2.2732$e+9$	9.6850$e+4$	*8.6561e + 3*
	Std.	2.2790$e+4$	1.6004$e+9$	1.2137$e+6$	3.5103$e+9$	2.8365$e+8$	1.7378$e+5$	*1.2465e + 4*
	Best	*4.0719e + 2*	4.6279$e+10$	4.2910$e+6$	8.3903$e+9$	1.8130$e+9$	4.2437$e+2$	1.2094$e+3$
	Rank	**1**	7	4	6	5	3	2
Average rank		2.777778	6.444445	4.472222	5.055556	5.16667	2.194445	**1.833334**
Overall rank		3	7	4	5	6	2	**1**

$$r_{f,g} = \frac{\mu_{f,g}}{\min\{\mu_{f,g}: g G\}}, \tag{26}$$

$$\rho_g(\tau) = \frac{\text{size}\{fF: r_{f,g} \le \tau\}}{n_f}, \tag{27}$$

where g represents an algorithm, G is the algorithm set, f represents a function, F is the function set, n_g is the number of algorithms in the experiment, n_f is the number of functions in the experiment, $\mu_{f,g}$ is the average fitness after the algorithm g solves function f, $r_{f,g}$ is the capability ratio, ρ_g is the algorithmic capability, and τ is a factor of the best probability [53].

Figure 9 shows the capability ratios of the mean fitness for the seven algorithms on the benchmark functions f_1–f_{15} ($D = 1000$). The results are displayed by a log scale 2. As shown in Figure 9, the CSMOA has the highest probability. When $\tau = 1$, the CMSOA is about 0.6, which is better than others. When $\tau = 4$, the CMSOA is about 0.87, the PSO is 0.53, the SOA is 0.40, the GSA is 0.067, the MVO is 0.067, the SCA is 0.067, and the SA_GA is 0.067. When $\tau = 12$, the CMSOA is 0.87, the PSO is 0.73, the SOA is 0.80, the GSA is 0.33, the MVO is 0.33, the SCA is 0.27, and the SA_GA is 0.2. The capability curve of the CMSOA lies above others, and the CMSOA can achieve about 0.87 when $\tau \ge 4$. Thus, the property of the CMSOA is better than that of other algorithms.

4.3. Algorithm Performance Comparison in PID Controller Parameter Optimization Problems. In this section, we use four test control system models optimizing the PID parameters to test the capability of the CMSOA. For the $g1 \sim g3$, the population number of all algorithms is 20, the max number of algebras is 20, the step response time of $g1 \sim g2$ is set to 10 s, and the step response time of $g3$ is set to 30 s. For $g4$, the population number of all algorithms is 50, the max number of algebras is 50, and the step response time is set at 50 s.

4.3.1. Control System Models. Equations (28)–(31) show the test control system models optimizing PID parameters used in our experiment. Figure 10 shows the process diagram for optimizing the test control system PID parameters by the CMSOA. Figure 11 shows the optimization of PID parameter model structure of the test control system.

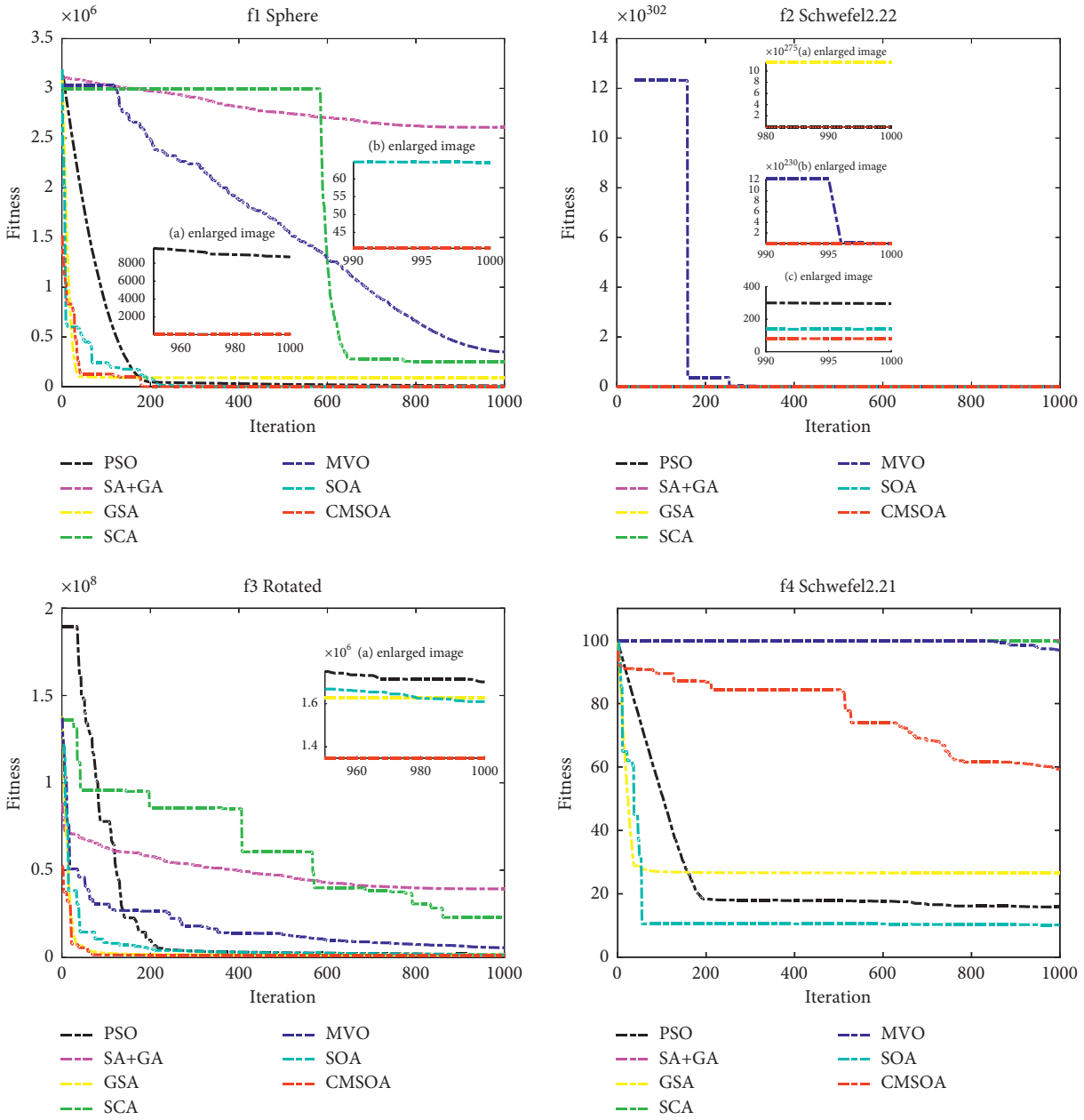
$$g_1(s) = \frac{2.6}{(2.7s+1)(0.3s+1)}, \tag{28}$$

$$g_2(s) = \frac{5}{(2.7s+1)} e^{-0.5s}, \tag{29}$$

$$g_3(s) = \frac{3}{(2s+1)} e^{-3s}, \tag{30}$$

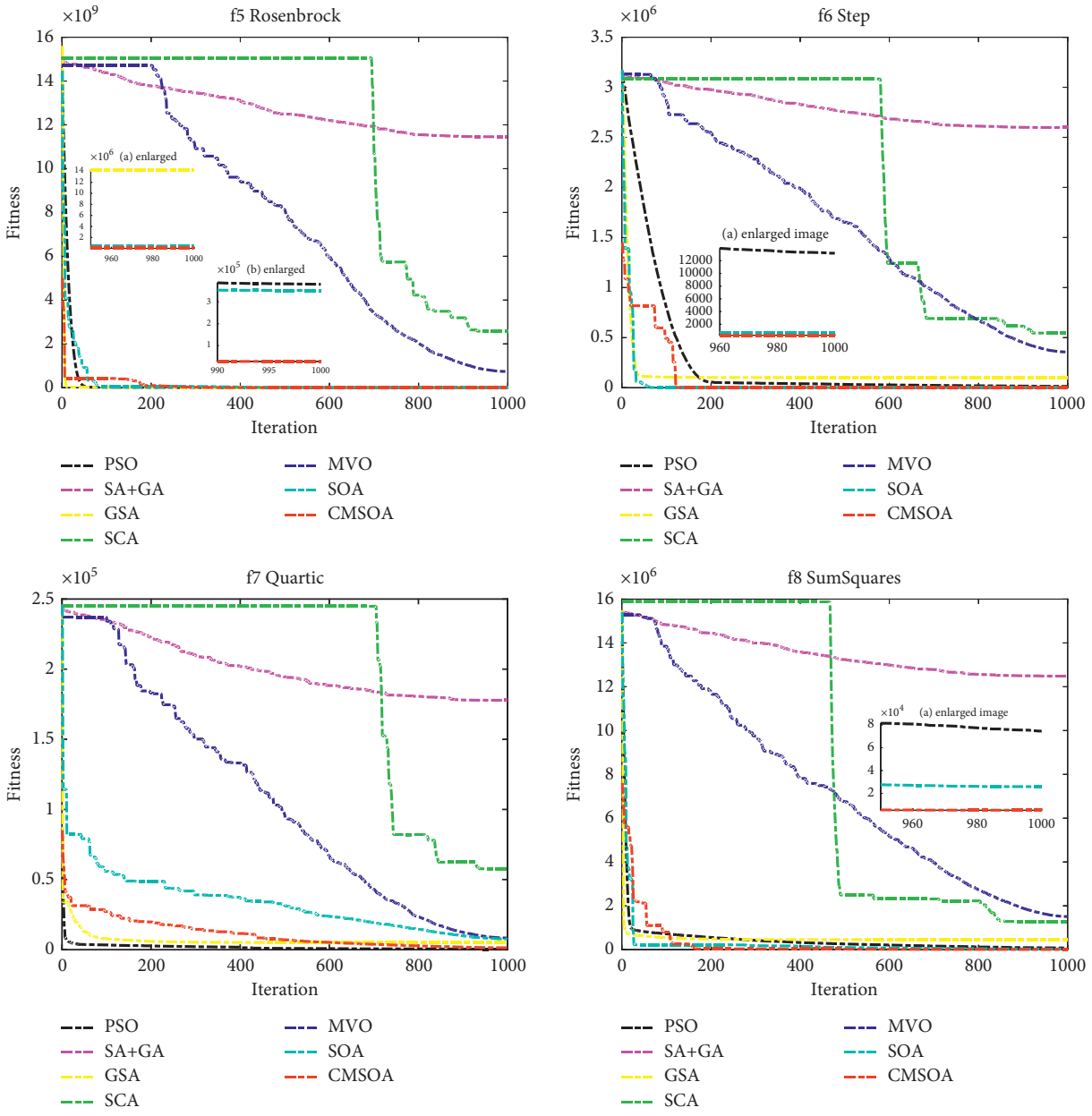
$$g_4(s) = \frac{1}{(s+1)^8}. \tag{31}$$

4.3.2. Result Comparison of Algorithms in PID Controller Parameter Optimization. For testing the capability of the CMSOA, the CMSOA is compared with the PSO, SA-GA, GSA,



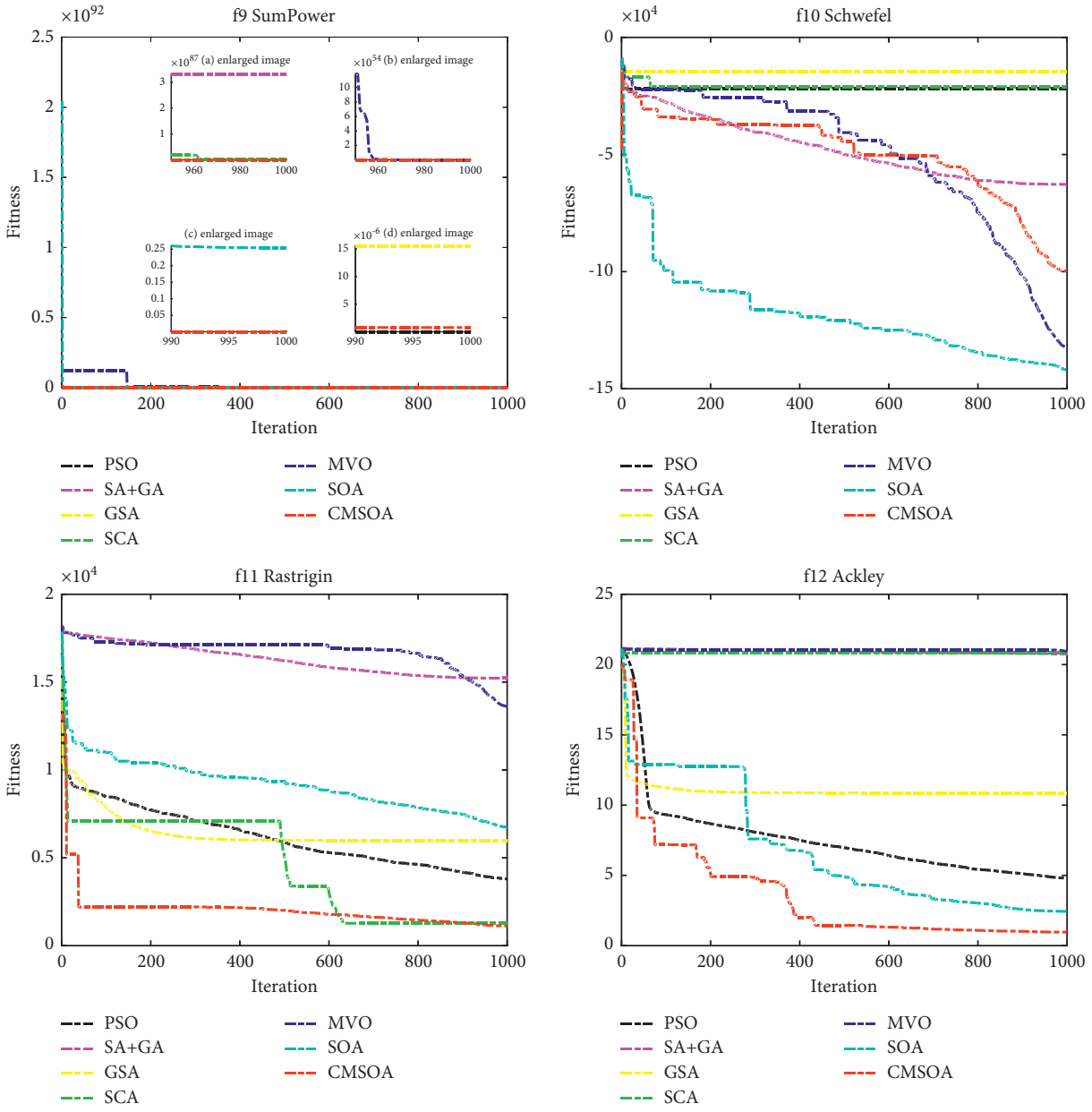
(a)

Figure 5: Continued.



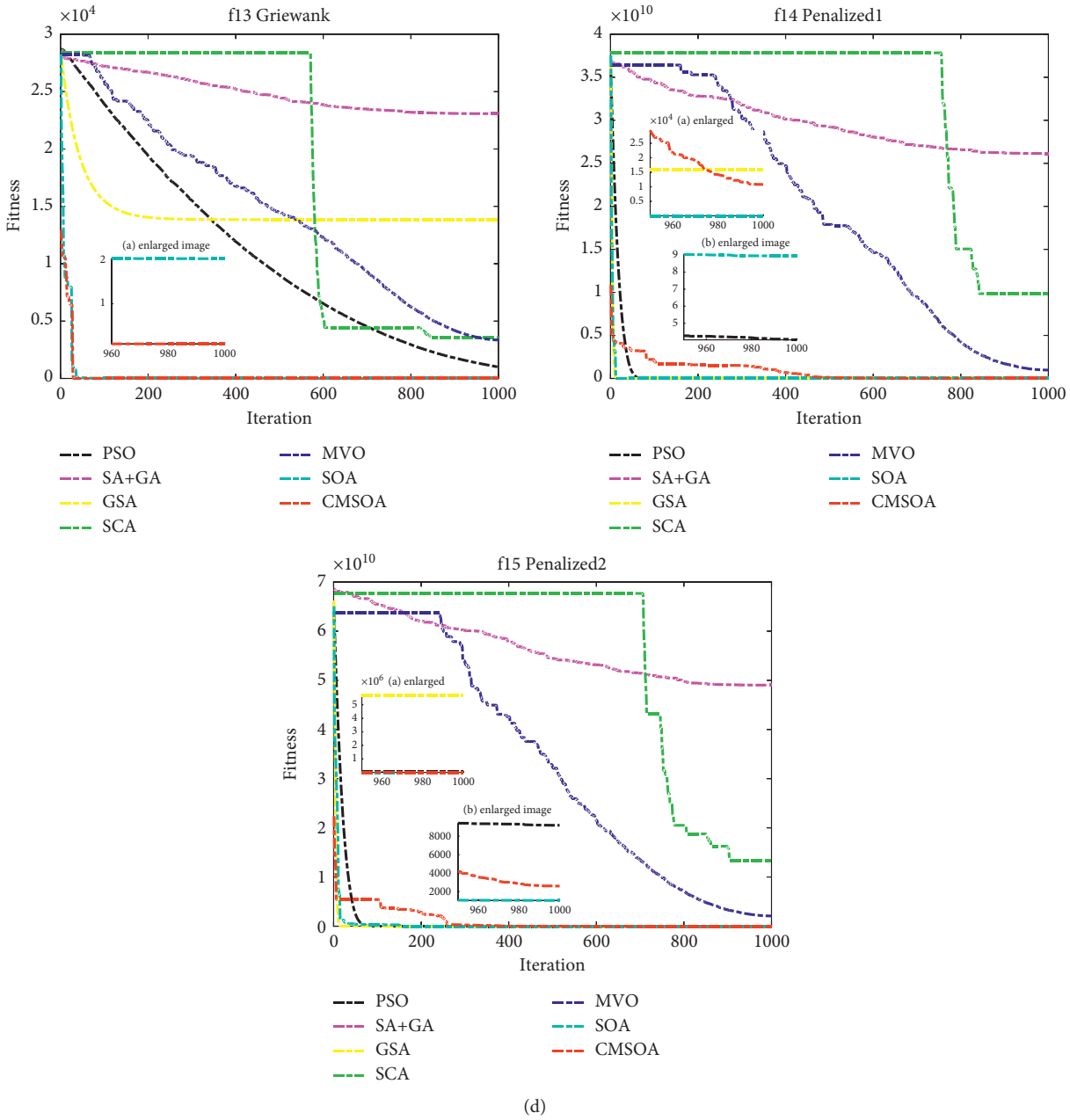
(b)

Figure 5: Continued.



(c)

Figure 5: Continued.



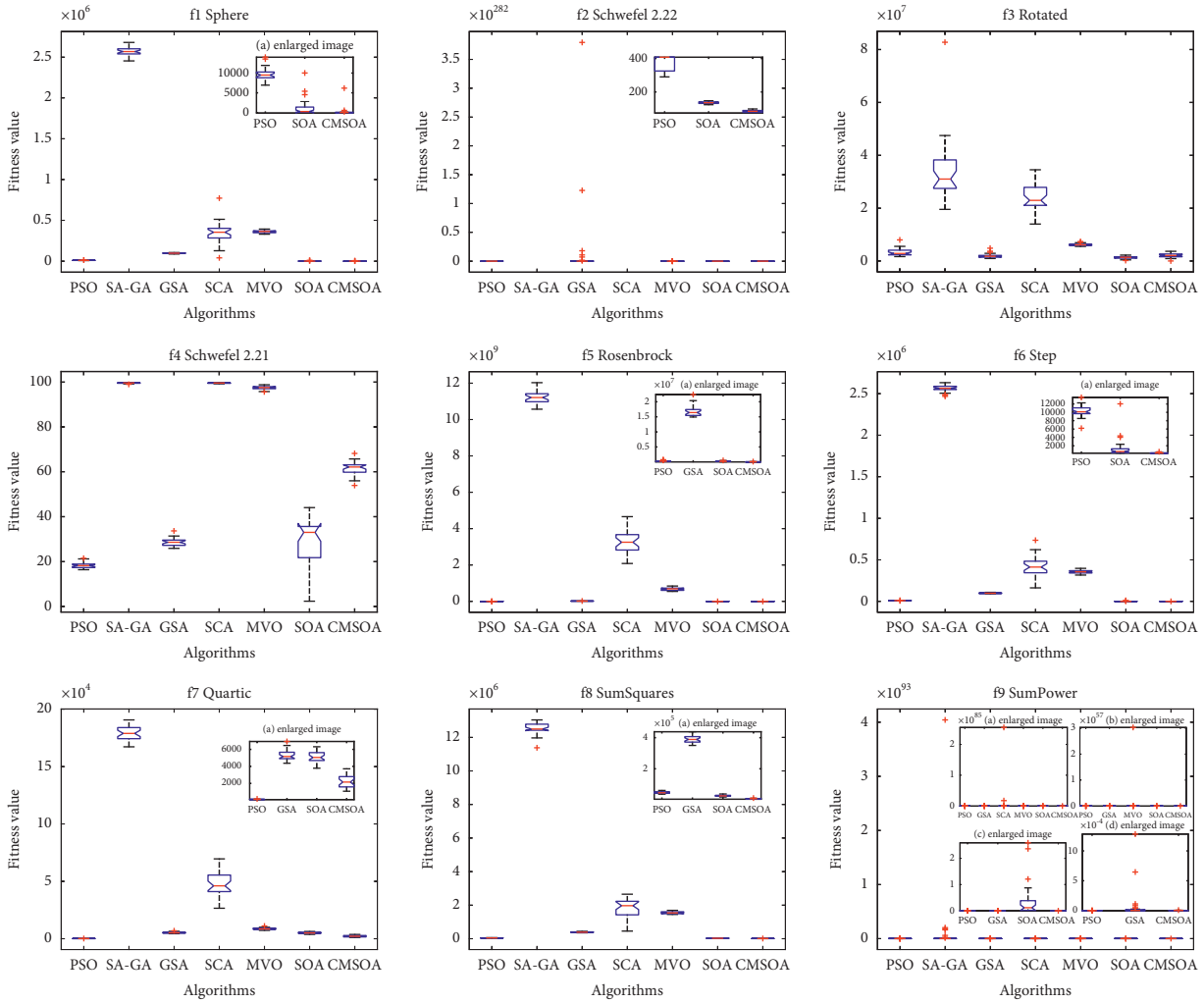
(d)

Figure 5: Convergence curves for benchmark functions f_1–f_{15} ($D = 1000$).

SCA, MVO, and SOA in the PID controller parameter optimization. The mean values, standard deviation values, best fitness values, and best fitness value ranks of the algorithms of 30 all-alone runs, for $g1 \sim g4$, are displayed in Table 7. The values in bold and italics indicate that the optimal result is better.

For the PID controller parameter optimization problems, according to Table 7, except $g3$ and $g4$, in terms of best fitness, the CMSOA is better than others. The optimal fitness value results of the CMSOA for $g3$ model are only worse than those of the SA_GA algorithm; the optimal fitness value result of the CMSOA for g4 model is only worse than that of the PSO algorithm. Except for $g2$ and $g3$, in terms of standard deviation results, the CMSOA is better than others, and the CMSOA is only worse than the SA_GA. In terms of mean, the CMSOA is better than others. According to the optimal fitness value mean rank and all rank results from Table 7, the CMSOA can find solutions and has very strong robustness for the PID controller parameter optimization problems.

4.3.3. The Convergence Curve Comparison of Algorithms in PID Controller Parameter Optimization. Figure 12 shows the fitness curves of PID controller parameter optimization for $g1 \sim g4$. As shown in Figure 12, the CMSOA is compared with the other six algorithms; the convergence of the CMSOA is fast, and the precision of the CMSOA is best. The CMSOA can find the optimal value.



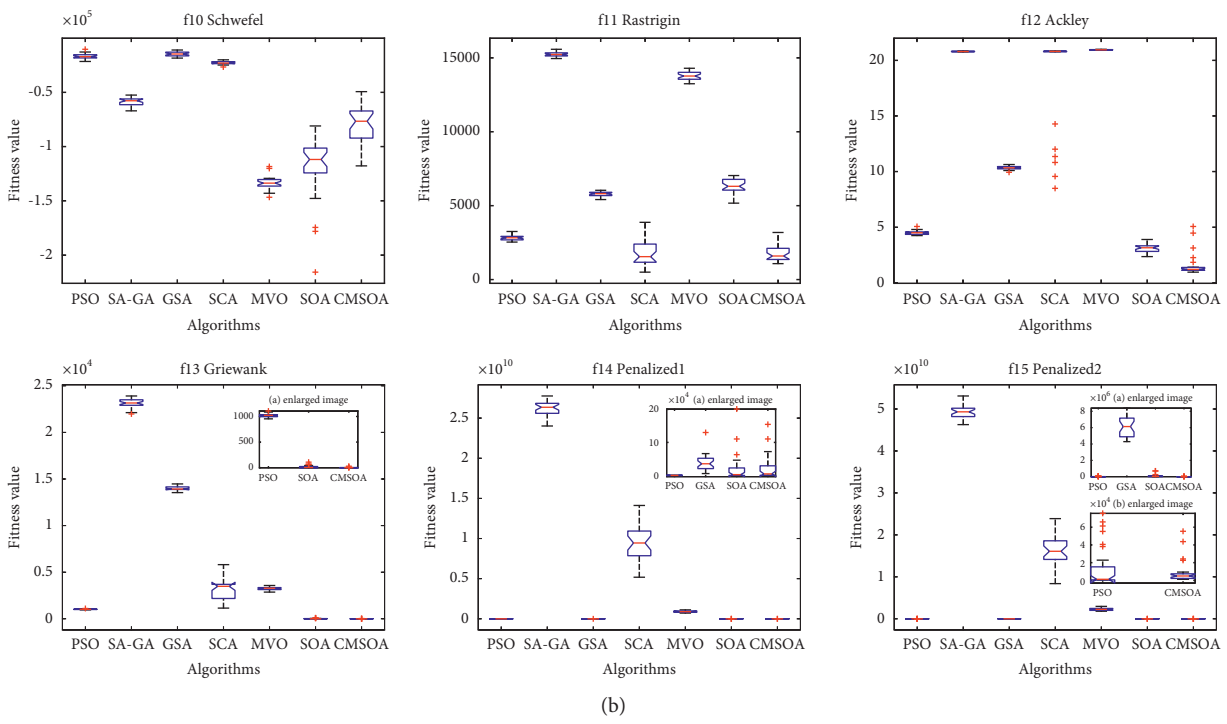


Figure 6: ANOVA tests for benchmark functions f_1–f_{15} ($D = 1000$).

Table 6: Run time comparison of 30 independent runs for benchmark functions f_{10}–f_{15} ($D = 1000$).

Functions ($D = 1000$)	Run time of algorithms						
	PSO	SA_GA	GSA	SCA	MVO	SOA	CMSOA
f_1	100.719730	880.039134	1238.641454	123.011947	291.672145	151.787862	1240.526227
f_2	68.229668	848.231807	1319.453189	158.694010	120.413171	201.005115	1120.092672
f_3	2083.966419	59198.227940	3001.030713	1813.227548	1978.830114	6464.103793	19353.182034
f_4	55.090061	763.100004	1182.766535	148.610507	282.693611	127.111629	991.587944
f_5	59.726643	907.492580	1190.101594	133.724941	347.492206	171.346819	1036.987842
f_6	60.136433	768.869660	1321.559282	127.490008	283.733081	137.581005	1054.889547
f_7	171.086743	2893.833975	1345.984756	203.877433	366.515217	378.499704	1809.255687
f_8	56.468720	824.393464	1361.309649	131.849271	296.880364	141.850056	1226.496492
f_9	119.475367	2814.347692	1338.479782	202.890012	270.227397	373.595086	1727.571955
f_{10}	96.665578	1331.028369	1207.890758	162.894968	163.620252	300.359964	1248.552418
f_{11}	67.987998	1166.055564	1200.353798	147.172828	321.039558	243.508636	1135.691572
f_{12}	85.468484	1298.972295	1504.250466	168.363508	421.893238	329.211669	1409.548409
f_{13}	105.918299	1489.631271	1217.56270	147.627982	330.277401	224.838827	1231.581318
f_{14}	227.567551	5817.019363	1364.117264	381.158993	485.116503	782.243419	2873.449473
f_{15}	252.124024	5953.028083	1367.575562	320.156020	494.213399	1149.480076	2800.126364
The total time	*3610.632*	86954.27	21161.08	4370.74998	6454.6177	11176.52	40259.54
Overall rank	**1**	7	5	2	3	4	6

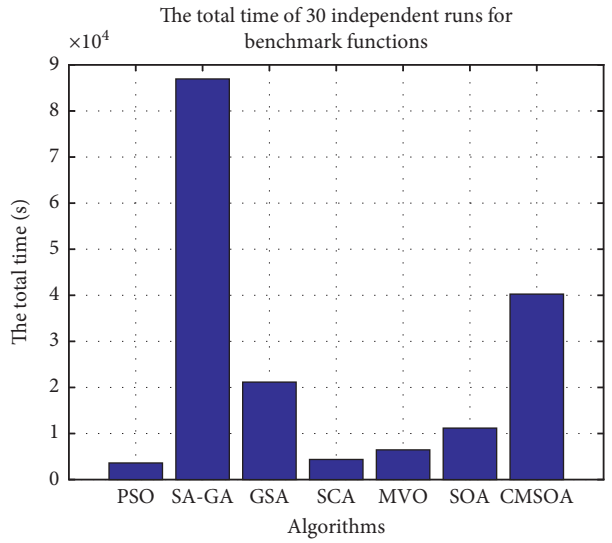


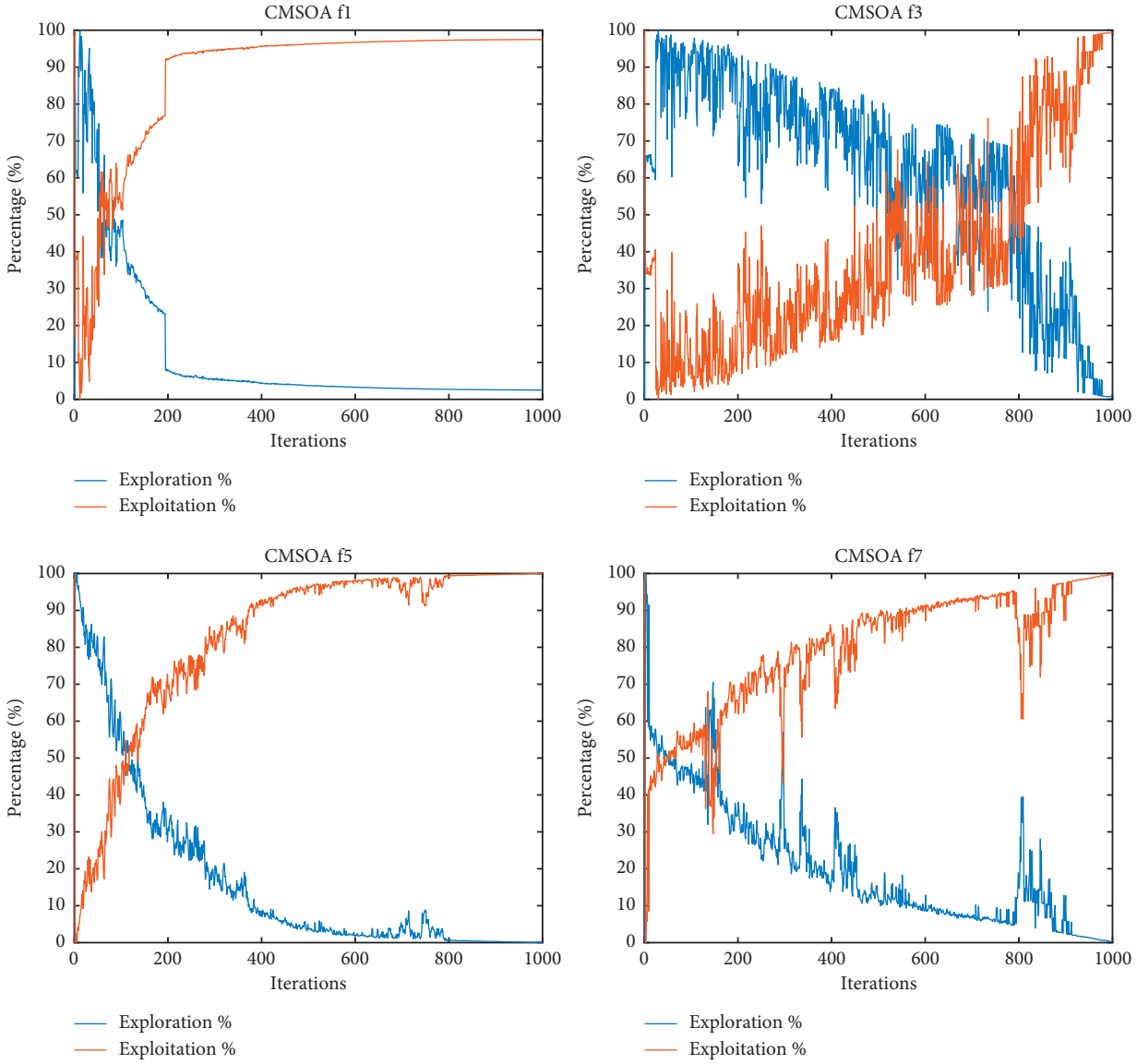
Figure 7: The total time of 30 independent runs of seven algorithms on 15 benchmark functions.

4.3.4. ANOVA Test Comparison of Algorithms in the PID Controller Parameter Optimization. Figure 13 shows the ANOVA of the global best values' PID controller parameter optimization for $g1 \sim g4$. As seen from Figure 13, the CMSOA is the most robust compared to other algorithms.

4.3.5. The Unit Step Functions of PID Controller Parameter Optimization. Figure 14 shows the unit step functions of PID controller parameter optimization for $g1 \sim g4$. As seen from Figure 14, by the CMSOA to optimization the unit step models PID controller parameter for $g1 \sim g4$, the unit step functions tend to stabilize very quickly and accurately.

Therefore, the CMSOA is an effective and feasible PID parameter optimization solution for the control system model.

4.4. Algorithm Performance Comparison in Constrained Engineering Optimization Problems. We use six engineering problems to test the capability of the CMSOA further. The engineering problems are very popular in the literature. The penalty function is used to calculate the constrained problem. The parameter set for all of the heuristic algorithms still adopts the parameter setting from Table 4 of Section 4.2.3. The formulations of these problems are available in Appendix.



(a)

Figure 8: Continued.

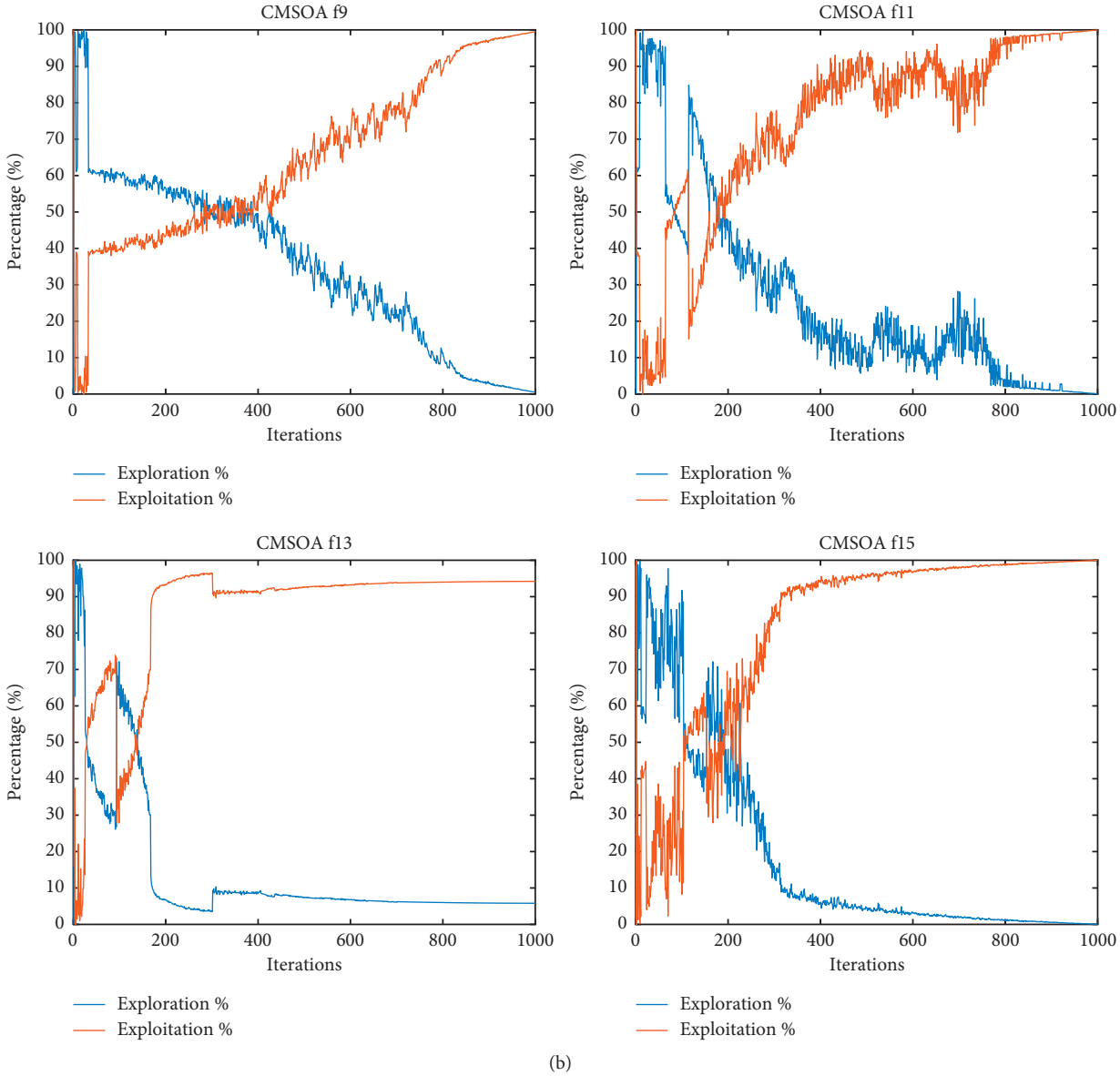


Figure 8: The exploration and exploitation abilities of the CMSOA in benchmark functions.

4.4.1. Welded Beam Design Problem. This is a least fabrication cost problem, which has four parameters and seven constraints. The parameters of the structural system are shown in Figure 15 [9]. Some of the works come from these kinds of literature: GSA [6], MFO [7], MVO [9], coevolutionary particle swarm optimization (CPSO) [54], and harmony search (HS) [55]. For the problem in this paper, the CMSOA is compared with the PSO, SA_GA, GSA, SCA, MVO, and SOA, and it provides the best-obtained values in Table 8.

In Table 8, the CMSOA is better than the GSA, MFO, MVO, GA, CPSO, and HS algorithms. The CMSOA is also better than the PSO, SA_GA, GSA, SCA, MVO, and SOA. Therefore, the CMSOA is an effective and feasible solution to the problem.

4.4.2. Pressure Vessel Design Problem. This is also a least fabrication cost problem of four parameters and four constraints. The parameters of the structural system are shown in Figure 16 [9]. Some of the works come from the literature: the MFO [7], the evolution strategies (ESs) [56], the differential evolution (DE) [57], the ant colony optimization (ACO) [58], and the GA [59]. For the problem, the CMSOA is compared with the PSO, SA_GA, GSA, SCA, MVO, and SOA, and it provides the best-obtained values in Table 9.

In Table 9, the CMSOA is better than the MFO, ES, DE, ACO, and GA. The CMSOA is also better than the PSO, SA_GA, GSA, SCA, MVO, and SOA. Therefore, the CMSOA is an effective and feasible solution to the problem.

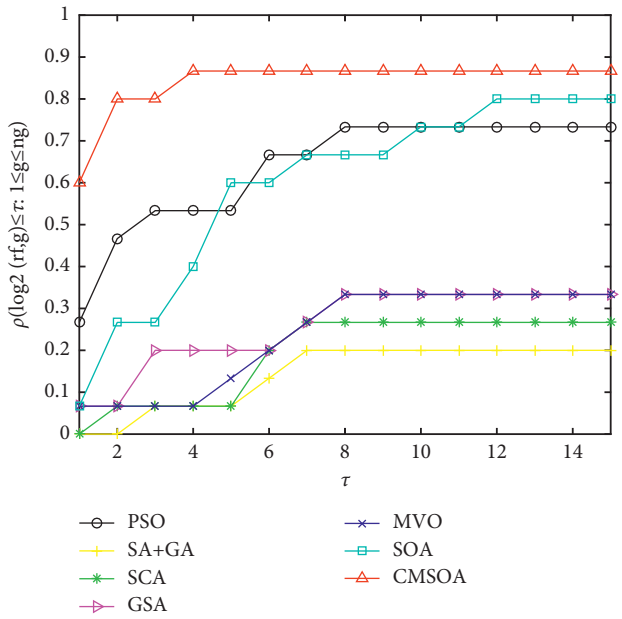


Figure 9: Performance profile of seven algorithms on 15 benchmark functions.

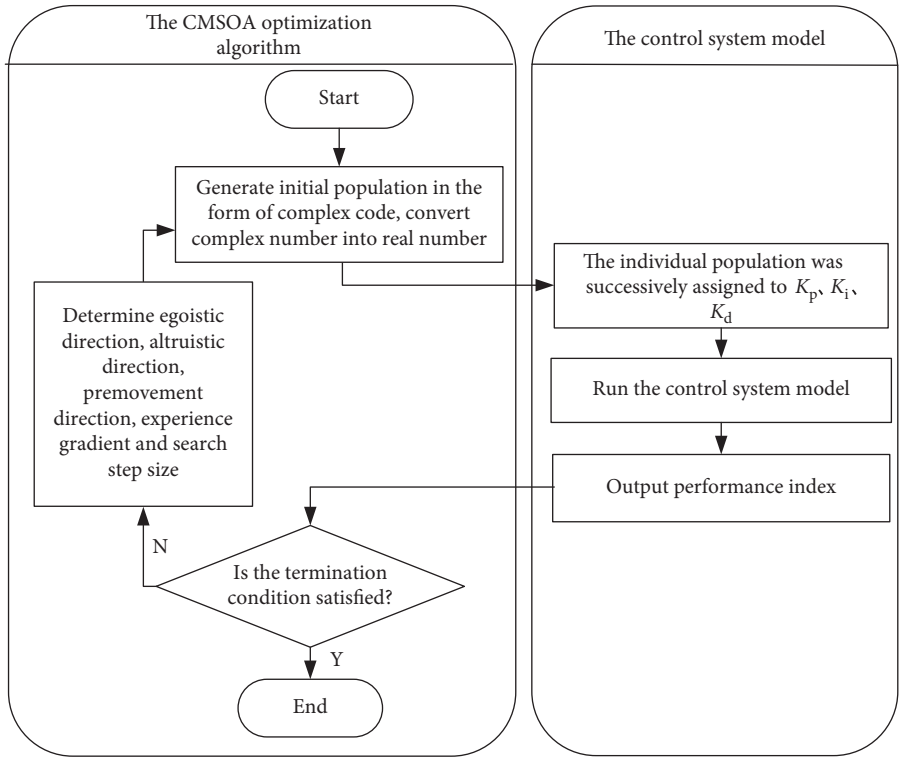


Figure 10: A process diagram for optimizing test control system PID parameters by the CMSOA.

4.4.3. Cantilever Beam Design Problem. This is a problem that is determined by five parameters and is only applied to the scope of the variables of constraints. The parameters of the structural system are shown in Figure 17 [7]. Some of the works come from these kinds of literature: the MFO [7], the cuckoo search algorithm (CS) [60], the generalized convex approximation (GCA) [61], the method of moving asymptotes (MMA) [61], and the symbiotic organism search

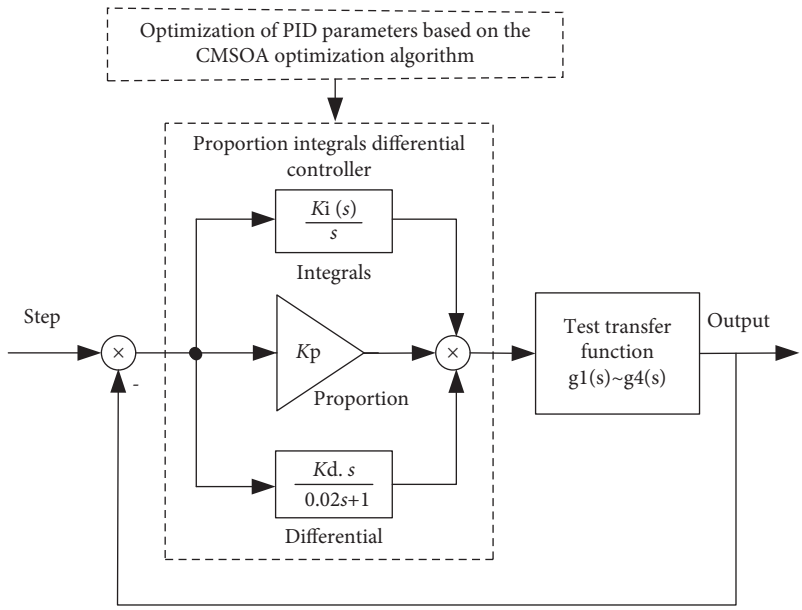


Figure 11: Optimization of PID parameter model structure of the test control system.

Table 7: Performance comparison of algorithms in PID parameter optimization of 30 independent runs.

Test	Result	Algorithm						
		PSO	SA_GA	GSA	SCA	MVO	SOA	CMSOA
$g1$	Mean	0.2267	0.3169	0.4571	0.0918	0.2501	0.1917	*0.0539*
	Std.	0.0877	0.0649	0.1569	0.0263	0.0532	0.11226	*0.0127*
	Best	0.0485	0.1002	0.2732	0.0483	0.0513	0.05774	*0.0479*
	Rank	3	6	7	2	4	5	**1**
$g2$	Mean	58.4757	62.4599	60.7787	24.8454	59.5805	42.1538	*0.8233*
	Std.	7.75976	*0.1216*	5.3034	21.5239	7.6556	27.9025	0.6631
	Best	36.0409	62.0356	42.7711	0.4898	32.6095	0.39301	*0.3299*
	Rank	5	7	6	3	4	2	**1**
$g3$	Mean	$1.8481e+2$	$2.7179e+2$	$2.7665e+2$	29.0458	$1.0848e+2$	$2.6269e+2$	*13.0293*
	Std.	59.6434	*0.62334*	10.3088	11.9839	56.6750	44.8106	1.7189
	Best	32.5445	*2.71191*	$2.7139e+2$	14.5588	20.0492	26.5763	10.1561
	Rank	6	**1**	7	3	4	5	2
$g4$	Mean	$1.7713e+2$	55.3556	$2.3413e+2$	85.196656	35.721213	46.10528	*34.63334*
	Std.	$4.2182e+2$	36.00807	$2.1754e+2$	$1.0050e+2$	1.411226	26.992197	*0.00686*
	Best	*34.625063*	34.6294	58.321733	34.867448	34.643162	34.745734	34.625096
	Rank	**1**	3	7	6	4	5	2
Average rank		3.75	4.25	6.75	3.5	4	4.25	1.5
Overall rank		3	5	7	2	4	5	**1**

(SOS) [62]. For the problem, the CMSOA is compared with the PSO, SA_GA, GSA, SCA, MVO, and SOA, and it provides the best-obtained values in Table 10.

In Table 10, the CMSOA is better than the MFO, CS, GCA, MMA, and SOS. The CMSOA is also better than the PSO, SA_GA, GSA, SCA, MVO, and SOA. Therefore, the CMSOA is an effective and feasible solution to the problem.

4.4.4. Gear Train Design Problem. This is a gear ratio minimization problem, which has four variables and the scope of variables of constraints. Figure 18 shows the schematic diagram [63]. Some of the works come from these kinds of the literature: the MFO [7], the MVO [9], the CS [60], the artificial bee colony (ABC) [64], and the mine blast algorithm (MBA) [64]. In this paper, the CMSOA is compared with the PSO, SA_GA, GSA, SCA, MVO, and SOA, and it provides the best-obtained values in Table 11.

In Table 11, the CMSOA proves to be better than the MFO, MVO, CS, ABC, and MBA. Except for the SA_GA, GSA, and PSO, the CMSOA is better than the SCA, MVO, and SOA. The optimal fitness value of the CMSOA has reached the theoretical best value, although the optimal fitness value of the CMSOA is worse than that of the SA_GA,

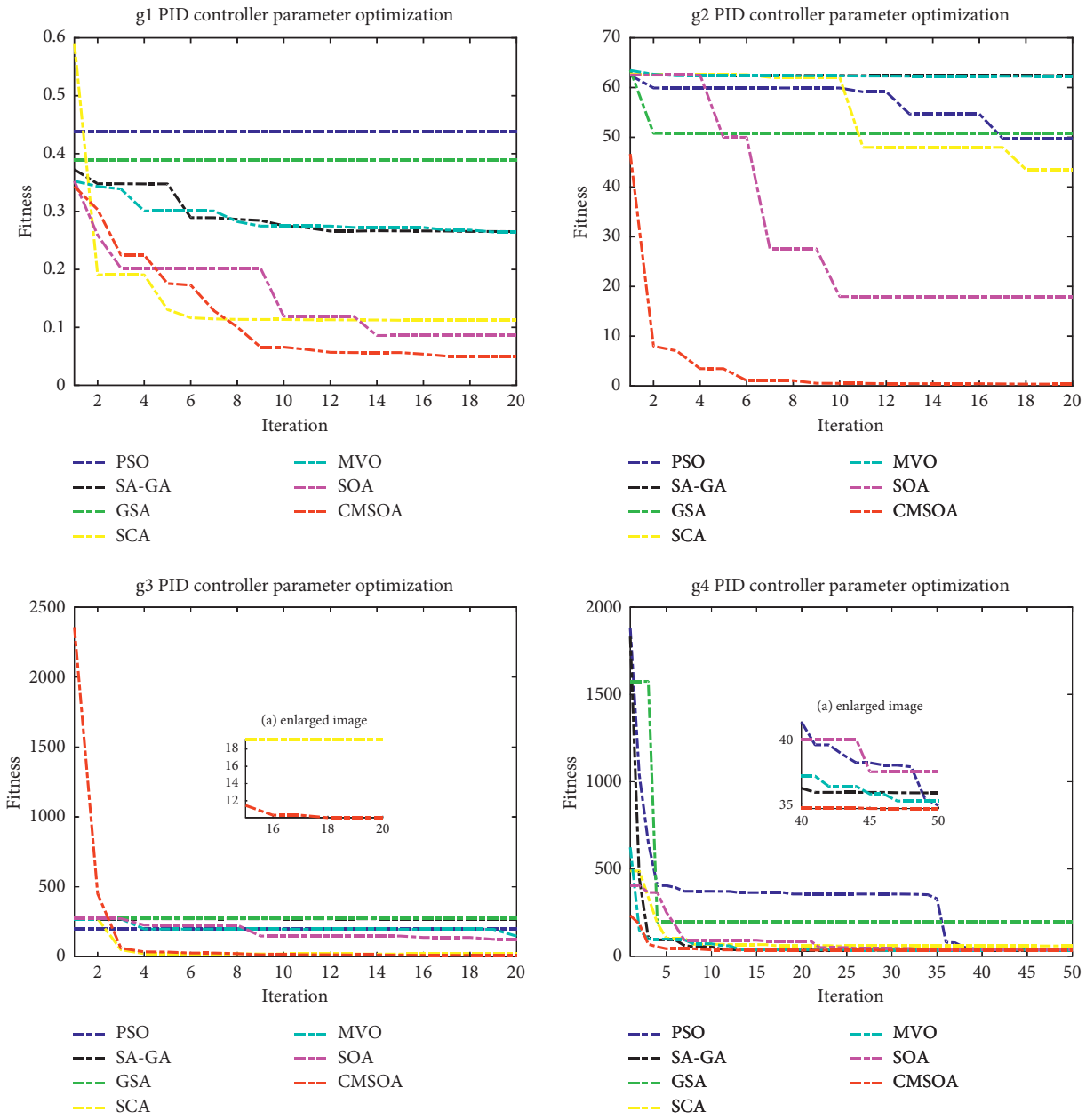


Figure 12: Convergence curves for PID controller parameter optimization ($g1 \sim g4$).

GSA, and PSO. The CMSOA finds a new value. Therefore, the CMSOA can resolve the problem.

4.4.5. Three-Bar Truss Design Problem. This is a weight minimization problem under stress, which has two variables and only applies to the scope of the variables of constraints. The schematic diagram of the components [63] is shown in Figure 19 [9].

Some of the works come from these kinds of literatures: MFO [7], MVO [9], CS [60], MBA [64], and differential evolution with dynamic stochastic selection (DEDSS) [65]. In this paper, the problem is resolved by the CMSOA. For the problem, the CMSOA is compared with the PSO, SA_GA, GSA, SCA, MVO, and SOA, and it provides the best-obtained values in Table 12.

In Table 12, except MVO and PSO, the CMSOA is better than the others. The optimal fitness value of the CMSOA has reached the theoretical best value, although the optimal fitness value of the CMSOA is worse than that of the MVO and the PSO. Therefore, the CMSOA can resolve the problem.

4.4.6. I-Beam Design Problem. This is a vertical deflection minimization problem that has four variables and a constraint. Figure 20 shows the design diagram [7]. Some of the works come from these kinds of literatures: MFO

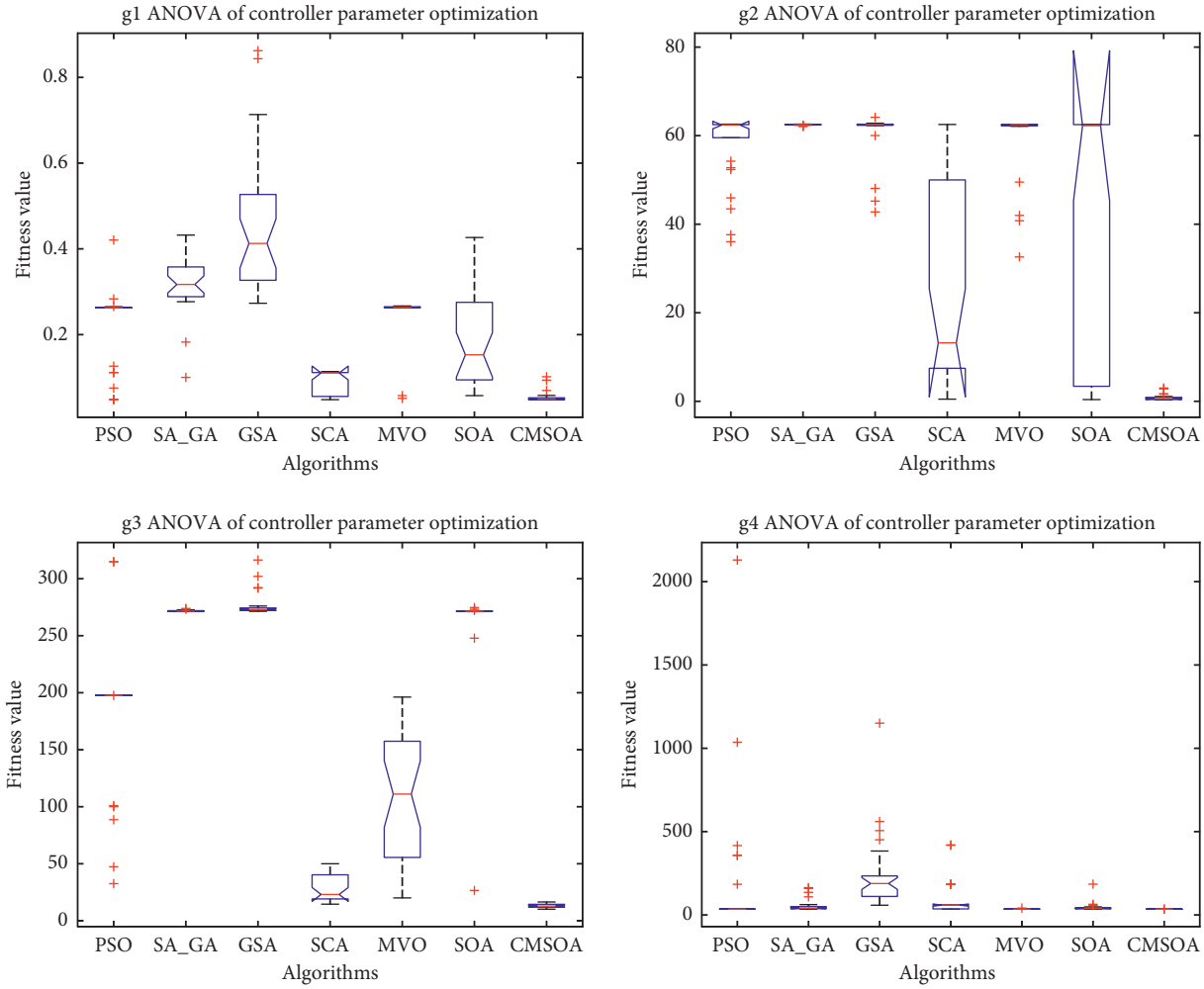


Figure 13: The ANOVA tests for the PID controller parameter optimization ($g1 - g4$).

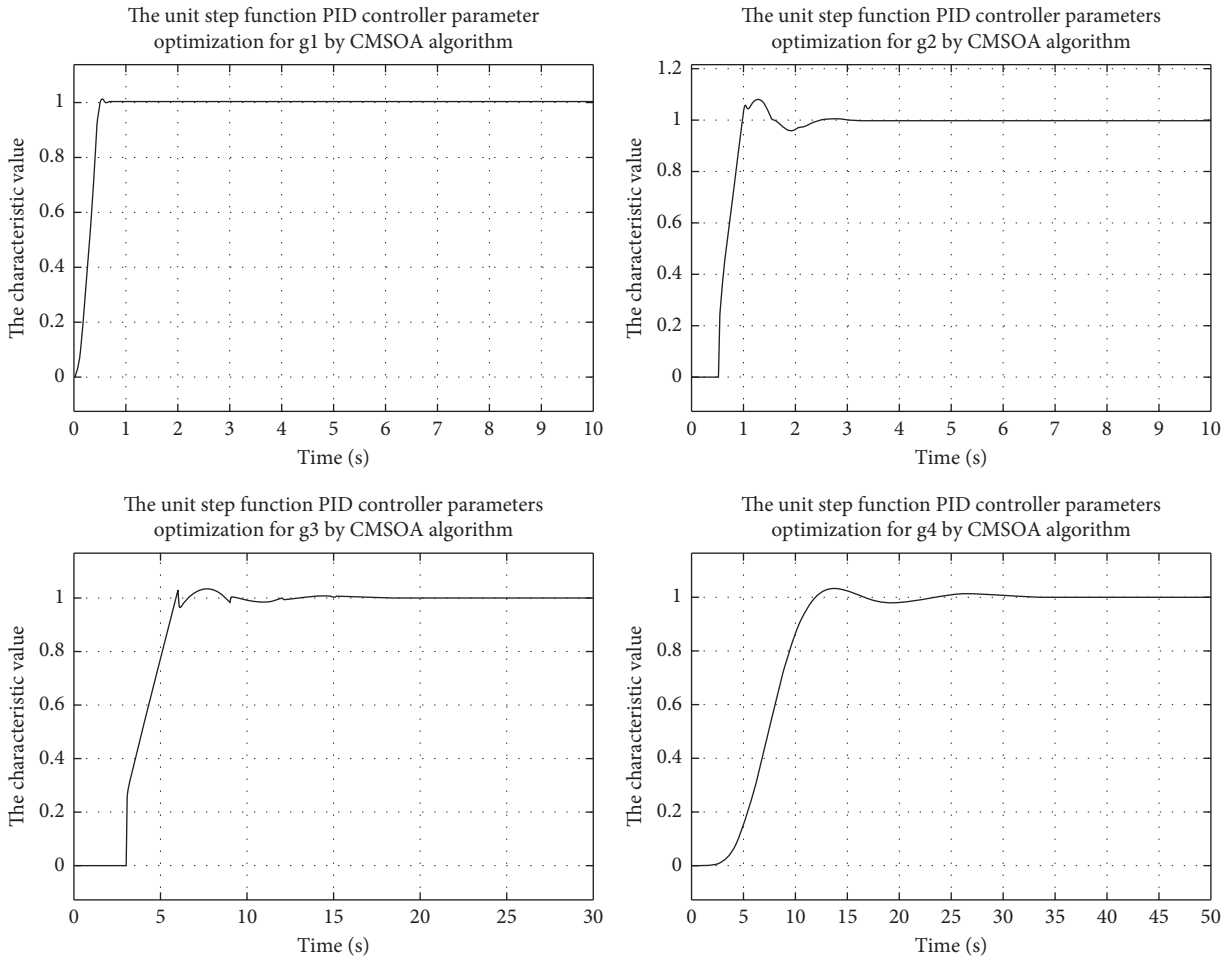


Figure 14: The unit step functions of PID controller parameter optimization ($g1 \sim g4$).

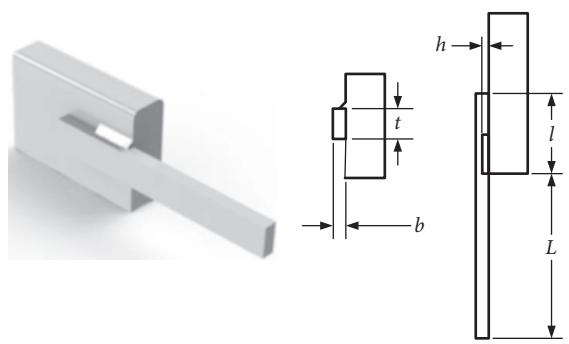


Figure 15: Design parameters of the welded beam.

Table 8: Comparison results of the welded beam design problem.

Algorithm	Optimal values for variables				Optimal cost	Rank
	h	l	t	b		
GSA [6]	0.182129	3.856979	10.0000	0.202376	1.87995	9
MFO [7]	0.2057	3.4703	9.0364	0.2057	1.72452	5
MVO [9]	0.205463	3.473193	9.044502	0.205695	1.72645	6
CPSO [54]	0.202369	3.544214	9.048210	0.205723	1.72802	7
HS [55]	0.2442	6.2231	8.2915	0.2443	2.3807	12
PSO	0.20437461682	3.27746206207	9.03907307954	0.20573458497	1.69700648019	2
SA-GA	0.26572876298	2.77789863579	7.63164040030	0.28853829376	1.99412873170	10
GSA	0.12743403146	5.89076184871	8.05262845397	0.25908004232	2.10212926568	11
SCA	0.20112344041	3.23948182622	9.40574225336	0.20795790595	1.76704865429	8
MVO	0.20397627841	3.28970350716	9.03536739179	0.20582407425	1.69811381975	4
SOA	0.19348578918	3.489546622637	9.027709656861	0.20615302629	1.69714450048	3
CMSOA	0.20568280035	3.25692824444	9.03941142183	0.20578118608	*1.69655946036*	*1*

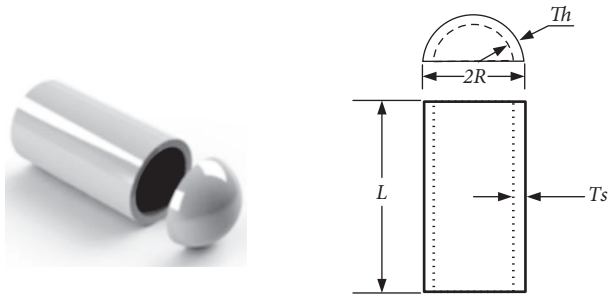


Figure 16: Pressure vessel design problem.

Table 9: Comparison results for pressure vessel design problem.

Algorithm	Optimal values for variables				Optimal cost	Rank
	T_s	T_h	R	L		
MFO [9]	0.8125	0.4375	42.098445	176.636596	6059.7143	8
ES [56]	0.8125	0.4375	42.098087	176.640518	6059.7456	10
DE [57]	0.8125	0.4375	42.098411	176.637690	6059.7340	9
ACO [58]	0.8125	0.4375	42.103624	176.572656	6059.0888	7
GA [59]	0.8125	0.4375	42.097398	176.654050	6059.9463	11
PSO	0.93627266112	0.41391783346	47.19019859907	123.06285131625	6317.0167340514	12
SA-GA	0.83804097369	0.41223740796	45.10610463950	142.64078515697	5931.2868373440	5
GSA	0.89533101776	0.43654377356	47.89640596198	115.96279725902	6057.9309555313	6
SCA	0.71165237901	0.39215740603	40.39056304889	200.00000000000	5903.0036698882	4
MVO	0.75462696023	0.37830685291	40.94839768196	191.64503059607	5764.4347452930	3
SOA	0.76961590364	41.5284631287	0.388196715944	183.84147207932	5735.1355906012	2
CMSOA	0.74366609133	40.3200509108	0.366463323335	200.00000000000	*5735.0844852589*	*1*

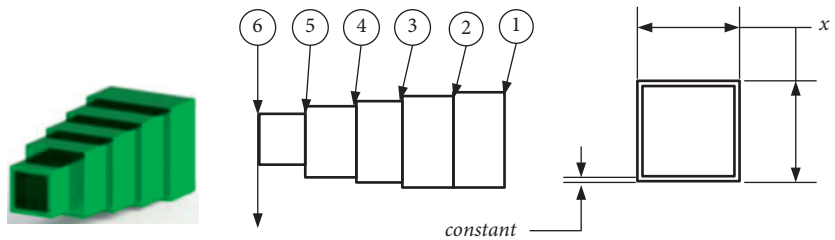


Figure 17: Cantilever beam design problem.

Table 10: Comparison results for cantilever beam design problem.

Algorithm	Optimal values for variables					Optimum weight	Rank
	x_1	x_2	x_3	x_4	x_5		
MFO [7]	5.9848717732	5.3167269243	4.4973325858	3.5136164677	2.1616202934	1.339988086	6
CS [60]	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999	7
GCA [61]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400	8
MMA [61]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400	8
SOS [62]	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996	3
PSO	6.007219438	5.311747232	4.505611438	3.4904346887	2.158626706	1.339963522	4
SA-GA	6.251285023	5.460509756	4.149903306	3.8032391760	1.974102742	1.350285757	11
GSA	6.020285873	5.305304583	4.512114944	3.4939372220	2.142187864	1.339969652	5
SCA	5.801308754	5.589807963	4.497563735	3.4994713866	2.262668613	1.351011196	12
MVO	6.017944991	5.336576175	4.493102726	3.4797461041	2.146292918	1.340024388	10
SOA	6.014092415	5.315583298	4.484154000	3.5033360363	2.156331174	1.339957455	2
CMSOA	6.013067642	5.292274798	4.491025571	3.5095364396	2.167826747	*1.339954592*	*1*

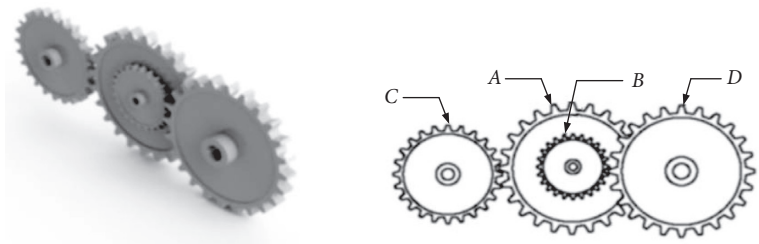


Figure 18: Gear train design problem.

Table 11: Comparison results of the gear train design problem.

Algorithm	Optimal values for variables				Optimal gear ratio	Rank
	n_A	n_B	n_C	n_D		
MFO [7]	43	19	16	49	$2.7009e-012$	7
MVO [9]	43	16	19	49	$2.7009e-012$	7
CS [60]	43	16	19	49	$2.7009e-012$	7
ABC [64]	49	16	19	43	$2.7009e-012$	7
MBA [64]	43	16	19	49	$2.7009e-012$	7
PSO	41.2676387267	12.0000000000	12.0000000000	24.1851491677	$5.321647791e-20$	3
SA-GA	32.3132176916	21.0818982120	12.1649288759	55.0091556193	*0*	**1**
GSA	54.7718113206	33.5951575204	12.0000000000	51.0148628266	$1.358936169e-30$	2
SCA	52.6322252242	15.4114043064	23.1179418870	46.9168162381	$5.431797718e-12$	12
MVO	60.0000000000	12.0000000000	41.8647883833	58.0329758032	$2.334953506e-16$	5
SOA	60.0000000000	12.0000000000	43.2835302093	60.0000000000	$2.567448245e-16$	6
CMSOA	43.3821083557	31.2957045927	12.0000000000	60.0000000000	$7.763414089e-17$	4

[7], CS [60], SOS [62], the adaptive response surface method (ARSM) [66], and the improved adaptive response surface method (IARSM) [66]. For the problem, the CMSOA is compared with the PSO, SA_GA, GSA, SCA, MVO, and SOA, and it provides the best-obtained values in Table 13.

In Table 13, except MFO, GSA, SOA, and SA-GA, the CMSOA is better than the others. The fitness of the MFO is best. Although the most minor vertical deviation of the CMSOA is not as good as that of the GSA, SOA, and SA-GA, it is very close to other relative optimal values. Therefore, the CMSOA is an effective and feasible solution to the I-beam design optimization problem.

In brief, the CMSOA proves to be better than the other algorithms in most actual studies. The CMSOA can resolve these practical problems.

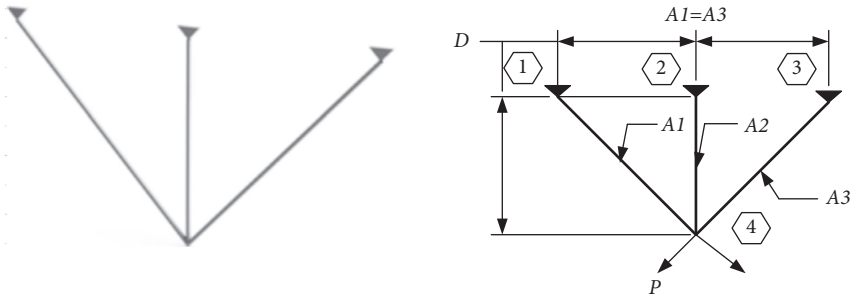


Figure 19: Three-bar truss design problem.

Table 12: Comparison results of the three-bar truss design problem.

Algorithm	Optimal values for variables		Optimum weight	Rank
	x_1	x_2		
MFO [7]	0.788244770931922	0.409466905784741	263.895979682	10
MVO [9]	0.78860276	0.40845307	263.8958499	8
CS [60]	0.78867	0.40902	263.9716	11
MBA [64]	0.7885650	0.4085597	263.8958522	9
DEDS [65]	0.78867513	0.40824828	263.8958434	7
PSO	0.788425434690935	0.408085596065985	263.8523465301364	2
SA-GA	0.787321758816231	0.411216143996852	263.8532291023197	5
GSA	0.761893501005708	0.493138841375638	264.8099085788021	12
SCA	0.789922169365255	0.403817724788810	263.8541885386347	6
MVO	0.788407496115311	0.408135122885127	*263.8523464859033*	**1**
SOA	0.788530250484097	0.407914579681955	263.8523714388302	4
CMSOA	0.788444195859439	0.408029807190657	263.8523473086418	3

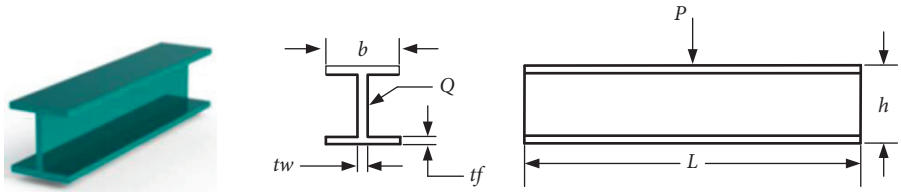


Figure 20: I-beam design problem.

Table 13: Comparison results for I-beam design problem.

Algorithm	Optimal values for variables				Optimum vertical deflection	Rank
	b	h	t_w	t_f		
MFO [7]	50	80	1.7647	5.0000	*0.0066259*	**1**
CS [60]	50	80	0.9 5	2.32167	0.0130747	9
SOS [62]	50	80	0.9	2.32179	0.0130741	8
IARSM [66]	48.42	79.99	0.90	2.40	0.131	11
ARSM [66]	37.05	80	1.71	2.31	0.0157	10
PSO	29.2349505988	77.7790428198	5.0000000000	3.5987373218	0.0114625520	12
SA-GA	34.9999839459	79.9999646294	4.9999802368	4.9999823841	0.0078637302	4
GSA	35.0000000001	80.0000000000	5.0000000000	5.0000000000	0.0078636959	2
SCA	34.9878089422	80.0000000000	5.0000000000	5.0000000000	0.0078658199	7
MVO	34.9998614894	80.0000000000	4.9997841775	5.0000000000	0.0078637964	6
SOA	34.9999002914	80.0000000000	5.0000000000	5.0000000000	0.0078636963	3
CMSOA	34.9997858604	80.0000000000	5.00000000000	5.0000000000	0.0078637332	5

5. Conclusion

A CMSOA is presented, with a complex-valued encoding method and a multichain strategy. The CMSOA is tested in four stages from different perspectives such as benchmark function, PID control parameters, and constraint engineering. Besides, the CMSOA was compared with the PSO, SA-GA, GSA, SCA, MVO, and SOA.

In the first phase, the SOA is improved in six different ways: the parameter changing SOA (PCSOA), the parameter adaptive Gaussian transform SOA (PAGTSOA), the SOA based on the Chebyshev chaos of order three (CCSOA), the SOA based on real coding double-link (DSOA), the SOA based on complex-valued encoding (CSOA), and the complex-valued encoding multichain seeker optimization algorithm (CMSOA). Each improved algorithm was optimized for the fifteen functions. The result is that the CMSOA is feasible in the benchmark functions. In this phase, we consider the ranking values of 30 all-alone runs between the CMSOA mean values, standard deviation values, best fitness values, best fitness value ranks, the convergence curves of functions f_1, f_{10}, and f_{14}, and the population's positions search history of functions f_1, f_{10}, and f_{14}.

In the second phase, fifteen benchmark function optimization problems are used to test the CMSOA further. The CMSOA is compared to the PSO, SA-GA, GSA, SCA, MVO, and SOA for verification. The CMSOA is feasible in the benchmark functions. The second phase is also about the ranking values of 30 all-alone runs between the CMSOA mean values, standard deviation values, best fitness values, best fitness value ranks, convergence curves, and the variance tests for the global minimum values. In the benchmark function optimization problems, the optimal solution curves obtained by the CMSOA are in good agreement with the theoretical optimal solution curves, and the accuracy of the CMSOA is better. The ANOVA of the global best values to benchmark functions is studied, and the CMSOA is the most robust algorithm. Based on the complexity analysis, the CMSOA is known as an efficient algorithm. Based on the run time comparison of seven algorithms in benchmark functions, the CMSOA has relatively more program running time, and it is not optimal in terms of running time. The exploration and exploitation abilities of the CMSOA in benchmark functions are studied, and the CMSOA maintains good balance between the exploration and exploitation abilities as the number of iterations increases. From the results of the performance ratios of the average solution for the seven algorithms, the optimization probability of the CMSOA is the highest.

In the third test phase, four PID control parameter optimization problems are used to test the CMSOA in practice further. The problems were a parameter optimization model of second-order PID controller without time delay, a parameter optimization model of PID controller with first-order microdelay, a parameter optimization model of first-order PID controller with significant time delay, and a parameter optimization model of high-order PID controller without time delay problems. The third test phase also considered the CMSOA mean values, standard deviation values, best fitness values, and best fitness values rank of 30 all alone runs, the convergence curves, and the ANOVA. From the results of PID parameter optimization problems, compared with the other six algorithms, the CMSOA is effective and feasible in the practical problem.

Eventually, in the last test phase, six engineering problems further tested the CMSOA. The CMSOA was compared with various algorithms. The results prove that the CMSOA is the highest competitive algorithm for the practical optimization problems.

According to the comparative analysis of the experiments, the conclusion is as follows:

(i) We use the complex-valued encoding and the multichain strategy for each seeker to increase the scout region and avoid convergence to local optimality.

(ii) Among the six improved algorithms: PCSOA, PAGSOA, CCSOA, DSOA, CSOA, and CMSOA, the CMSOA performed best in the benchmark function test.

(iii) Among the seven algorithms PSO, SA_GA, GSA, SCA, MVO, SOA, and CMSOA, the CMSOA optimization benchmark functions have higher optimization capability.

(iv) The CMSOA optimization benchmark function has almost the same calculational complexity as the SOA.

(v) The running time of the CMSOA optimization benchmark function is relatively long. Among the seven algorithms compared, the running time is only shorter than that of the SA-GA.

(vi) The CMSOA can solve real challenging problems, such as the PID control parameter optimization problems and the real constrained engineering optimization problems.

(vii) Further improvement and application can be incorporated into future studies.

Appendix

A. Welded Beam Design Problem

$$
\begin{aligned}
&\text{Consider } \vec{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b],\\
&\text{Minimize} \quad f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2),\\
&\qquad g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0,\\
&\qquad g_3(\vec{x}) = x_1 - x_4 \le 0,\\
&\qquad g_4(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0,\\
&\text{Subject to}\\
&\qquad g_5(\vec{x}) = 0.125 - x_1 \le 0,\\
&\qquad g_6(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0,\\
&\qquad g_7(\vec{x}) = P - P_c(\vec{x}) \le 0,\\
&\text{Variable range: } 0.1 \le x_1 \le 2, 0.1 \le x_2 \le 10, 0.1 \le x_3 \le 10, 0.1 \le x_4 \le 2,
\end{aligned}
\tag{A.1}
$$

where $\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''x_2/2R + (\tau'')^2}$, $\tau' = P/\sqrt{2}x_1x_2$, $\tau'' = \mathrm{MP}/J$, $M = P(L + x_2/2)$, $R = \sqrt{(x_2^2/4) + ((x_1 + x_3/2)^2)}$, $J = 2\left\{\sqrt{2}x_1x_2[(x_2^2/4) + ((x_1 + x_3/2)^2)]\right\}$, $\sigma(\vec{x}) = 6\mathrm{PL}/x_4x_3^2$, $\delta(\vec{x}) = 6\mathrm{PL}^3/Ex_4x_3^3$, $P_c(\vec{x}) = ((4.013E\sqrt{x_3^2x_4^6/36})/L^2)(1 - (x_3/2L)\sqrt{E/4G})$, $P = 6000\,\mathrm{lb}$, $L = 14\,\mathrm{in}$, $E = 30 \times 10^6\,\mathrm{psi}$, $G = 12 \times 10^6\,\mathrm{psi}$, $\tau_{\max} = 136000\,\mathrm{psi}$, $\sigma_{\max} = 30000\,\mathrm{psi}$, $\delta_{\max} = 0.25\,\mathrm{in}$.

B. Pressure Vessel Design Problem

$$
\begin{aligned}
&\text{Consider } \vec{x} = [x_1, x_2, x_3] = [T_s, T_h, R, L],\\
&\text{Minimize} \quad f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3\\
&\qquad g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0\\
&\qquad g_2(\vec{x}) = -x_2 + 0.00954x_3 \le 0\\
&\text{Subject to}\\
&\qquad g_3(\vec{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0\\
&\qquad g_4(\vec{x}) = x_4 - 240 \le 0,\\
&\text{Variable to: } 0 \le x_1 \le 99, 0 \le x_2 \le 99, 10 \le x_3 \le 200, 10 \le x_4 \le 200.
\end{aligned}
\tag{B.1}
$$

C. Cantilever Design Problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2, x_3, x_4, x_5],\\ &\text{Minimize} \quad f(\vec{x}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5),\\ &\text{Subject to} \quad g(\vec{x}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0,\\ &\text{Variable to: } 0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100.\end{aligned} \tag{C.1}$$

D. Gear Train Design Problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2, x_3, x_4] = [n_A, n_B, n_C, n_D],\\ &\text{Minimize} \quad f(\vec{x}) = \left(\frac{1}{6.931} - \frac{x_3 x_2}{x_1 x_4}\right)^2,\\ &\text{Variable range: } 12 \le x_1, x_2, x_3, x_4 \le 60.\end{aligned} \tag{D.1}$$

E. Three-Bar Truss Design Problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2] = [A_1, A_2],\\ &\text{Minimize} \quad f(\vec{x}) = \left(2\sqrt{2}\, x_1 + x_2\right) * l\\ &\qquad g_1(\vec{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1 x_2} P - \sigma \le 0\\ &\text{Subject to} \quad g_2(\vec{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1 x_2} P - \sigma \le 0\\ &\qquad g_3(\vec{x}) = \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \le 0,\\ &\text{Variable range: } 0 \le x_1, x_2 \le 1, l = 100\,\text{cm}, P = 2\,\text{kN/cm}^2, \sigma = 2\,\text{kN/cm}^2.\end{aligned} \tag{E.1}$$

F. I-Beam Design Problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2, x_3, x_4] = [b, h, t_w, t_f],\\ &\text{Minimize} \quad f(\vec{x}) = \frac{5000}{\left(x_3(x_2 - 2x_4)^3/12\right) + \left(x_1 x_4^3/6\right) + \left(2x_1 x_4 (x_2 - x_4/2)^2\right)},\\ &\text{Subject to} \quad g(\vec{x}) = 2x_1 x_3 - x_3(x_2 - 2x_4) \le 0,\\ &\text{Variable range: } 10 \le x_1 \le 50, 10 \le x_2 \le 80, 0.9 \le x_3 \le 5, 0.9 \le x_4 \le 5.\end{aligned} \tag{F.1}$$

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

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

Acknowledgments

This study was supported by the National Natural Science Foundation of China (grant nos. 51766005 and 52166001) and Science and Technology Project of Yunnan Tobacco Company of China (grant no. 2019530000241019).

References

[1] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

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

[3] R. C. Eberhart and J. A. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science (MHS '95)*, pp. 39–43, IEEE, Nagoya, Japan, October 1995.

[4] E. H. L. Aarts and P. J. M. Laarhoven, "Simulated annealing: an introduction," *Statistica Neerlandica*, vol. 43, no. 1, pp. 31–52, 1989.

[5] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm, harmony search," *SIMULATION*, vol. 76, pp. 60–68, 2001.

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

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

[8] S. Mirjalilia, "S. C. A.: A Sine Cosine Algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, no. 27, pp. 1–14, 2016.

[9] S. Mirjalilia, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization," *Neural Computing & Applications*, vol. 17, pp. 16–19, 2015.

[10] M. Tuba, I. Brajevic, and R. Jovanovic, "Hybrid seeker optimization algorithm for global optimization," *Applied Mathematics & Information Sciences*, vol. 7, no. 3, pp. 867–875, 2013.

[11] G. G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural Computing & Applications*, vol. 31, pp. 1995–2014, 2019.

[12] 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.

[13] G. G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Computing*, vol. 9, pp. 1–14, 2016.

[14] Y. Yang, H. Chen, A. A. Heidari, and A. H. Gandomi, "Hunger games search: visions, conception, implementation, deep analysis, perspectives, and towards performance shifts," *Expert Systems with Applications*, vol. 177, pp. 1–34, Article ID 114864, 2021.

[15] I. Ahmadianfar, A. A. Heidari, A. H. Gandomi, X. Chu, and H. Chen, "RUN beyond the metaphor: an efficient optimization algorithm based on Runge Kutta method," *Expert Systems with Applications*, vol. 181, pp. 1–22, Article ID 115079, 2021.

[16] 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.

[17] A. Alet and I. Moser, "A systematic literature review of adaptive parameter control methods for evolutionary algorithms," *ACM Computing Surveys*, vol. 10, pp. 1–33, 2016.

[18] T. Bui, T. Nguyen, H. M. Huynh, B. Vo, J. W. L. Chun, and T. P. Hong, "Multiswarm multiobjective particle swarm optimization with simulated annealing for extracting multiple tests," *Scientific Programming*, vol. 2020, Article ID 7081653, 15 pages, 2020.

[19] W. Tong, "A hybrid algorithm framework with learning and complementary fusion features for whale optimization algorithm," *Scientific Programming*, vol. 2020, pp. 1–25, Article ID 5684939, 2020.

[20] Y. Wang, S. Gao, M. Zhou, and Y. Yu, "A multi-layered gravitational search algorithm for function optimization and real-world problems," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, pp. 94–109, 2021.

[21] J. Ji, S. Song, T. Cheng, S. Gao, Z. Tang, and Y. Todo, "An artificial bee colony algorithm search guided by scale-free networks," *Information Sciences*, vol. 473, pp. 142–165, 2019.

[22] S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, and M. C. Zhou, "Chaotic local search-based differential evolution algorithms for optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 6, Article ID 39540, 2021.

[23] L. Yin, S. Luo, Y. Wang, F. Gao, and J. Yu, "Coordinated complex-valued encoding dragonfly algorithm and artificial emotional reinforcement learning for coordinated secondary voltage control and automatic voltage regulation in multigenerator power systems," *IEEE Access*, vol. 8, Article ID 180520, 2020.

[24] P. Wang, Y. Zhou, Q. Luo, C. Han, Y. Niu, and M. Lei, "Complex-valued encoding metaheuristic optimization algorithm: a comprehensive survey," *Neurocomputing*, vol. 407, pp. 313–342, 2020.

[25] S. Zhang, Y. Zhou, Q. Luo, and M. Abdel-Baset, "A complex-valued encoding satin bowerbird optimization algorithm for global optimization," *Intelligent Computing Methodologies*, Springer International Publishing AG, part of Springer Nature, Manhattan, New York City, 2018.

[26] Y. Zhou, Z. Bao, Q. Luo, and S. Zhang, "A complex-valued encoding wind driven optimization with greedy strategy for 0-1 knapsack problem," *Applied Intelligence*, vol. 46, pp. 684–702, 2017.

[27] F. Miao, Y. Zhou, and Q. Luo, "Complex-valued encoding symbiotic organisms search algorithm for global optimization," *Knowledge and Information Systems*, vol. 1, pp. 209–248, 2019.

[28] M. Abdel-Baset, H. Wu, and Y. Zhou, "A complex encoding flower pollination algorithm for constrained engineering optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 8, no. 2, pp. 108–126, 2017.

[29] C. Dai, Y. Zhu, and W. Chen, "Seeker optimization algorithm,"vol. 1, pp. 225–229, in *Proceedings of the 2006 Inter. Conf. Computational Intelligence and Security*, vol. 1, pp. 225–229, IEEE Press, Guangzhou, China, November 2006.

[30] C. Dai, Y. Zhu, and W. Chen, "Seeker optimization algorithm," *Lecture Notes in Computer Science*, vol. 4456, pp. 167–176, 2007.

[31] C. Dai, W. Chen, Y. Zhu, and X. Zhang, "Seeker optimization algorithm for optimal reactive power dispatch," *IEEE Transactions on Power Systems*, vol. 24, no. 3, pp. 1218–1231, 2009.

[32] C. Dai, W. Chen, Y. Song, and Y. Zhu, "Seeker optimization algorithm: a novel stochastic search algorithm for global numerical optimization," *Journal of Systems Engineering and Electronics*, vol. 21, no. 2, pp. 300–311, 2010.

[33] C. Dai, W. Chen, and Y. Zhu, "Seeker optimization algorithm for digital IIR filter design," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 5, pp. 1710–1718, 2010.

[34] C. Dai, W. Chen, Y. Zhu, Z. Jiang, and Z. You, "Seeker optimization algorithm for tuning the structure and parameters of neural networks," *Neurocomputing*, vol. 74, no. 6, pp. 876–883, 2011.

[35] C. Dai, Z. Cheng, Q. Li, Z. Jiang, and J. Jia, "Seeker optimization algorithm for global optimization: a case study on optimal modelling of proton exchange membrane fuel cell (PEMFC)," *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 3, pp. 369–376, 2011.

[36] C. Dai, W. Chen, L. Ran, Y. Zhang, and Y. Du, "Human group optimizer with local search," *Lecture Notes in Computer Science*, vol. 6728, pp. 310–320, 2011.

[37] Y. Zhu, C. Dai, and W. Chen, "Seeker optimization algorithm for several practical applications," *International Journal of Computational Intelligence Systems*, vol. 7, no. No. 2, pp. 353–359, 2014.

[38] L. S. Coelho, "Gaussian quantum - behaved particle swarm optimization approaches for constrained engineering design problems," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1676–1683, 2010.

[39] X. Song, L. Jian, and Y. Song, *A Chunk Updating LSSVMs Based on Block Gaussian Elimination Method*, Elsevier Science Publishers B, Amsterdam, Netherlands, pp. 96–104, 2017.

[40] Z. Zheng, Y. Zhang, and Y. Qiu, *Genetic Algorithm Based on Complex-Valued Encoding*, pp. 1–21, IET Control Theory, Stevenage, UK, 2003.

[41] D. Chen, H. Li, and Z. Li, "Particle swarm optimization based on complex-valued encoding and application in function optimization," *Computer Times*, vol. 45, no. 10, pp. 59–61, 2009.

[42] L. Li and Y. Zhou, "A novel complex-valued bat algorithm," *Neural Computing & Applications*, vol. 25, no. 6, pp. 1369–1381, 2014.

[43] Y. Zhou, L. Li, and M. Ma, "A complex-valued encoding bat algorithm for solving 0-1 knapsack problem," *Neural Processing Letters*, vol. 44, pp. 1–24, 2015.

[44] L. Li, Y. Zhou, and J. Xie, "A free search krill herd algorithm for functions optimization," *Mathematical Problems in Engineering*, vol. 2014, Article ID 936374, 21 pages, 2014.

[45] X. Li, J. Zhang, and M. Yin, "Animal migration optimization: an optimization algorithm inspired by animal migration behavior," *Neural Computing & Applications*, vol. 24, no. 7-8, pp. 1867–1877, 2014.

[46] M. Molga and C. Smutnicki, "Test functions for optimization needs," 2005, http://www.robertmarks.org/Classes/ENGR5358/Papers/functions.pdf.

[47] J. Kennedy, *Particle Swarm Optimization in Encyclopedia of Machine Learning*, pp. 760–766, Springer, New York, NY, USA, 2010.

[48] H. Yu, H. Fang, P. Yao, and Yi Yuan, "A combined genetic algorithm/simulated annealing algorithm for large scale system energy integration," *Computers & Chemical Engineering*, vol. 24, no. 8, pp. 2023–2035, 2000.

[49] S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, *Algorithms*, pp. 15–18, McGraw-Hill, Inc. Professional Book Group 11 West 19th Street, New York, NY, USA, 2009.

[50] M. Crepinsek, S. H. Lin, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: a survey," *ACM Computing Surveys*, vol. 45, no. 3, pp. 1–33, 2013.

[51] K. Hussain, M. N. S. Mohd, S. Cheng, and Y. Shi, "On the exploration and exploitation in popular swarm-based metaheuristic algorithms," *Neural Computing & Applications*, vol. 31, pp. 7665–7683, 2019.

[52] B. Morales-Castaeda, D. Zaldívar, E. Cuevas, F. Fausto, and A. Rodríguez, "A better balance in metaheuristic algorithms: does it exist?" *Swarm and Evolutionary Computation*, vol. 54, pp. 1–23, 2020.

[53] E. Dolan and J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, no. 2, pp. 201–213, 2002.

[54] R. A. Krohling and L. D. S. Coelho, "Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 36, pp. 1407–1416, 2006.

[55] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, pp. 3902–3933, 2005.

[56] E. M. Mezura and C. A. C. Coello, "An empirical study about the usefulness of evolution strategies to solve constrained optimization problems," *International Journal of General Systems*, vol. 37, pp. 443–473, 2008.

[57] L. Li, Z. Huang, F. Liu, and Q. Wu, "A heuristic particle swarm optimizer for optimization of pin connected structures," *Computers & Structures*, vol. 85, pp. 340–349, 2007.

[58] A. Kaveh and S. Talatahari, "An improved ant colony optimization for constrained engineering design problems," *Engineering Computers: Int. J. Comput. Aid. Eng.*vol. 27, pp. 155–182, 2010.

[59] K. Deb, "Optimal design of a welded beam via genetic algorithms," *AIAA Journal*, vol. 29, pp. 2013–2015, 1991.

[60] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering Computers*, vol. 29, pp. 17–35, 2013.

[61] H. Chickermane and H. Gea, "Structural optimization using a new local approximation method," *International Journal for Numerical Methods in Engineering*, vol. 39, pp. 829–846, 1996.

[62] M. Y. Cheng and D. Prayogo, "Symbiotic organisms search: a new metaheuristic optimization algorithm," *Computers & Structures*, vol. 139, pp. 98–112, 2014.

[63] E. Sandgren, "Nonlinear integer and discrete programming in mechanical design," *ASME Journal Mechanical Design*, vol. 112, pp. 223–229, 1990.

[64] 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, pp. 2592–2612, 2013.

[65] M. Zhang, W. Luo, and X. Wang, "Differential evolution with dynamic stochastic selection for constrained optimization," *Information Sciences*, vol. 178, pp. 3043–3074, 2008.

[66] G. G. Wang, "Adaptive response surface method using inherited Latin hypercube design points," *Journal of Mechanical Design*, vol. 125, pp. 210–220, 2003.