Research Article

An Elastic Collision Seeker Optimization Algorithm for Optimization Constrained Engineering Problems

Shaomi Duan,1,2 Huilong Luo,1 and Haipeng Liu2

1Faculty of Civil Engineering and Mechanics, Kunming University of Science and Technology, Kunming 650500, China
2Faculty 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 3 July 2021; Revised 23 September 2021; Accepted 11 December 2021; Published 10 January 2022

Academic Editor: Aditya Rio Prabowo

Copyright © 2022 Shaomi Duan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

To improve the seeker optimization algorithm (SOA), an elastic collision seeker optimization algorithm (ECSOA) was proposed. The ECSOA evolves some individuals in three situations: completely elastic collision, completely inelastic collision, and non-completely elastic collision. These strategies enhance the individuals’ diversity and avert falling into the local optimum. The ECSOA is compared with the particle swarm optimization (PSO), the simulated annealing and genetic algorithm (SA_GA), the gravitational search algorithm (GSA), the sine cosine algorithm (SCA), the multiverse optimizer (MVO), and the seeker optimization algorithm (SOA); then, fifteen benchmark functions, four PID control parameter models, and six constrained engineering optimization problems were selected for the experiment. According to the experimental results, the ECSOA can be used in the benchmark functions, the PID control parameter optimization, and the optimization constrained engineering problems. The optimization ability and robustness of ECSOA are better.

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], the particle swarm optimization (PSO) [3], the simulated annealing (SA) [4], the harmony search (HS) [5], the gravitational search algorithm (GSA) [6], the moth-flame optimization (MFO) [7], the sine cosine algorithm (SCA) [8], the multiverse optimizer (MVO) [9], the seeker optimization algorithm (SOA) [10], the artificial bee colony (ABC) algorithm [11], the krill herd (KH) [12], the monarch butterfly optimization (MBO) [13], the elephant herding optimization (EHO) [14], the moth search (MS) algorithm [15], the slime mould algorithm (SMA) [16], and the Harris hawks optimization (HHO) [17].

However, some optimization algorithms are still not very successful in optimization problems. The optimization problems include issues with low optimization precision, being premature, having only a local optimal solution, slow convergence speed, and insufficient robustness. To better overcome the issues of 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 Harris hawks optimization algorithm, salp swarm algorithm, grasshopper optimization algorithm, and dragonfly algorithm are used for the structural design optimization of vehicle components [18]. The adaptive inertia weight factor in the traditional PSO optimizes path planning [19]. The PSO based on Gaussian and quantum behavior optimizes constrained engineering problems [20]. The least squares support vector machines based on Gaussian are proposed [21]. A Levy flights discrete bat algorithm is adopted to solve the Euclidean traveling salesman problem [22]. The cuckoo optimization algorithm in reverse logistics is used to design a network for COVID-19 waste management [23]. A chaotic cuckoo optimization algorithm based on a Levy flight, backward learning, and interfere operator is used to classify the optimal feature subspace [24]. An elite symbiotic organisms search algorithm with mutually
beneficial factor is adopted to optimize the functions [25]. An artificial bee colony with dynamic Cauchy mutation is adopted to solve feature selection [26]. A new elastic collision optimization algorithm is applied in sensor cloud resource scheduling [27].

Dai et al. proposed the SOA in 2006 [28]; the goal is to mimic the seekers’ behavior and the way they exchange information and solve practical application optimization problems. In the recent decade, the SOA has been used in many fields, such as unconstrained optimization problems [29], optimal reactive power dispatch [30], challenging set of benchmark problems [31], design of a digital filter [32], optimizing parameters of artificial neural networks [33], optimizing model and structures of fuel cell [34], novel human group optimizer algorithm [35], and several practical applications [36]. However, in the initial stage of dealing with optimization problems, SOA converges faster than others. When all individuals are near the best individual for solving the optimization problem, the individuals will lose diversity and fall into prematurity.

In this article, we propose an elastic collision seeker optimization algorithm (ECSOA), which evolves some individuals in three situations: complete elastic collision, complete inelastic collision, and incomplete elastic collision. These strategies enhance the individuals’ diversity and avert premature convergence. The ECSOA is compared to seven improved SOAs, such as the changing algorithm parameters, the adaptive transformation of empirical value parameters, the levy motion of some individuals, the reverse learning, the addition of mutual benefit factor, and the Cauchy mutation. This article chose fifteen benchmark functions to test. According to the experimental results, the convergence speed and accuracy of ECSOA are higher. The improved strategy enables the SOA to maintain the individuals’ diversity, avert falling into the local optimum, and make up for the shortcomings of SOA in the aspect of easy precisocity. Finally, compared with PSO, SA, GA, GSA, SCA, MVO, and SOA, the ECSOA has been implemented and tested on a complete set of well-known fifteen benchmark functions, four PID control parameter optimization models, and six optimization constrained engineering problems taken from the literature. According to the experimental results, ECSOA is feasible in the benchmark functions, the PID parameter optimization problems, and the constrained engineering optimization problems. The ECSOA can find better values for solving the questions. The improved SOA successfully overcomes its tendency to prematurely converge to local optima for problems. The ECSOA has better optimization performance and robustness. The algorithm also has an improvement over the original SOA. The advantages of the ECSOA are summed up as follows:

1. An ECSOA is raised to enhance the precision and robustness of the optimization process.

2. The elastic collision strategies, the completely elastic collision, the completely inelastic collision, and the non-complete elastic collision, can improve the diversity of individuals, enhance local search, and avert premature convergence.

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

2. Basic SOA and Algorithm Improvement Strategies

The SOA carries out in-depth search mimicking human search behavior. It considers optimization as a search for an optimal solution by a search team in search space, taking the 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. SOAs have three main updating steps.

2.1.1. Search Direction. The forward orientation of 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.

$$\overrightarrow{f_{i, e}}(t) = \overrightarrow{x}_{i, \text{best}}(t) - \overrightarrow{x}_{i, t_1}(t),$$
$$\overrightarrow{f_{i, a}}(t) = \overrightarrow{y}_{i, \text{best}} - \overrightarrow{x}_{i, t_1}(t),$$
$$\overrightarrow{f_{i, p}}(t) = \overrightarrow{x}_{i, t_1}(t) - \overrightarrow{x}_{i, t_2}(t).$$

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

$$\overrightarrow{f}_i(t) = \text{sign}(\omega \overrightarrow{f}_{i, p}(t) + \psi_1 \overrightarrow{f}_{i, a}(t) + \psi_2 \overrightarrow{f}_{i, e}(t)),$$

where $t_1, t_2 \in [t, t-1, t-2]$; $\overrightarrow{x}_{i, t_1}(t)$ and $\overrightarrow{x}_{i, t_2}(t)$ are the best advantages of $\overrightarrow{x}_{i, (t-2)}$, $\overrightarrow{x}_{i, (t-1)}$, $\overrightarrow{x}_{i, t}(t)$ separately; $g_{i, \text{best}}$ is the historical optimal location in the neighborhood where the ith search factor is located; $P_{i, \text{best, p}}$ is the optimal locality from the ith search factor to the current locality; $\psi_1$ and $\psi_2$ are random numbers in $[0, 1]$; and $\omega$ 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. The 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.
where \( \alpha \) and \( \delta \) are parameters of a membership function. According to (3), the probability of the output variable exceeding \([-3\delta, 3\delta]\) is less than 0.0111. Therefore, \( \mu_{\text{min}} = 0.0111 \). Under normal circumstances, the optimal position of an individual has \( \mu_{\text{max}} = 1.0 \), and the worst place is 0.0111. However, to accelerate the convergence speed and get the optimal individual to have an uncertain step size, \( \mu_{\text{max}} \) is set as 0.9 in this paper. Select the following function as the fuzzy variable with a “small” target function value:

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

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

where \( \mu_i \) is determined by (4) and (5), \( I_i \) is the count of the sequence \( x_i(t) \) of the current individuals arranged from high to low by function value, and the function \( \text{rand}(\mu_{ij}, 1) \) is the real number in any partition \([\mu_{ij}, 1] \).

It can be seen that (4) simulates the random search behavior of human beings. Step measurement of \( j \)-dimensional search interspace is determined by the following equation:

\[
\alpha_{ij} = \delta_{ij} - \sqrt{-\ln(\mu_{ij})}. \tag{6}
\]

where \( \delta_{ij} \) is a parameter of the Gaussian distribution function, which is defined by

\[
\omega = \frac{\text{iter}_{\text{max}} - t}{\text{iter}_{\text{max}}}, \tag{7}
\]

\[
\delta_{ij} = \omega \times \text{abs}(\overline{x}_{\text{min}} - \overline{x}_{\text{max}}). \tag{8}
\]

where \( \omega \) is the weight of inertia. As the evolutionary algebra increases, \( \omega \) decreases linearly from 0.9 to 0.1. \( \overline{x}_{\text{min}} \) and \( \overline{x}_{\text{max}} \) 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 (8).

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

\( i \) is the \( i \)-th searcher individual; \( j \) represents the individual dimension; \( f_{ij}(t) \) and \( \alpha_{ij}(t) \), respectively, represent the searchers’ search direction and search step size at time \( t \); and \( x_{ij}(t) \) and \( x_{ij}(t + 1) \), respectively, represent the searchers’ site at time \( t \) and \( t + 1 \).

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

2.2.1. Dynamic Adaptive Gaussian Variation of Empirical Parameters. In the SOA, (8) is changed to (10), 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 (11). The individual position update is still the same as (9).

\[
\delta_{ij} = \omega \times \text{abs}(\overline{x}_{\text{min}} - C_1 \times \text{rand}(1, d)), \tag{10}
\]

\[
C_1 = 0.5 - t \times \left( \frac{0.1}{\text{iter}_{\text{max}}} \right), \tag{11}
\]

where \( i \) represents the \( i \)-th individual, \( j \) represents the individual dimension, \( \delta_{ij} \) is a parameter of the Gaussian membership function [20, 21], \( t \) means the current algebra, \( \text{iter}_{\text{max}} \) represents the maximum optimization algebra, and \( d \) represents the dimension of the optimized object.

2.2.2. The Levy Movement. A Levy movement [22, 24] is a random searching path alternating between short and occasionally long walks following the Levy distribution. The position update equation of the Levy motion is as follows:

\[
x_{ij}(t + 1) = x_{ij}(t) \times \text{Levy}(d),
\]

\[
\text{Levy}(\beta) = 0.01 \times \frac{r_1 \times \sigma}{|r_2|^{\beta+1}}, \tag{12}
\]

\[
\sigma = \left( \frac{\Gamma(1 + \beta) \times \sin(\pi\beta/2)}{(\Gamma(1 + \beta/2) \times \beta \times 2^{(\beta-1)/2})} \right)^{1/\beta}.
\]

\( i \) represents the \( i \)-th individual and \( j \) represents the number of individuals. \( \Gamma(\beta) = (\beta - 1)! \). \( t \) is the current algebra. \( d \) is the dimension of the optimized object. \( r_1, r_2 \in \text{rand}(0, 1) \). \( \beta \) is the partial real constant, which is 1.5 in this paper. After judging whether the fitness value is good or bad based on the newly generated individual position vector in (14), the original individual will be replaced by the best.

2.2.3. The Refraction Reverse Learning. If the projection of refraction points \( x'' \) on the \( x \)-axis represents \( x' \), \( x'' \) represent the reverse solution of individual \( x \) based on the
refraction principle [23, 24]. The value of the boundary point in the refraction reverse learning is \((a + b)/2\) of the search interval \([a, b]\). As shown in Figure 1, the calculation of \(\sin \alpha\) and \(\sin \beta\) is shown in (13) and (14).

\[
\sin \alpha = \frac{(a + b)/2 - x}{h}, \quad (13)
\]

\[
\sin \beta = \frac{(x^* - (a + b)/2)}{h^*}. \quad (14)
\]

According to (13) and (14), we can get

\[
\frac{\sin \alpha}{\sin \beta} = \left( \frac{(a + b)/2 - x}{x^* - (a + b)/2} \right) \left( \frac{h}{h^*} \right). \quad (15)
\]

Assuming \(k = h/h^*\), we can write (15) as (16), and then (17), \(n = 1\) and \(k = 1\), can be simplified to (18).

\[
kn = \left( \frac{(a + b)/2 - x}{x^* - (a + b)/2} \right). \quad (16)
\]

\[
x^* = \frac{a + b}{2} + \frac{(a + b)}{2kn} - \frac{x}{(kn)}. \quad (17)
\]

\[
x^* = a + b - x. \quad (18)
\]

When it is applied to the SOA, the probability of mutation is 0.8. The individual positions are taken for the refraction reverse learning according to (19) to get the new individual positions. In the formula, \(i\) is the \(i\)th individual, and \(j\) is the individual dimension. After judging whether the fitness value is good or bad based on the newly generated individual position vector in (19), the original individual is replaced by the best one.

\[
x_{ij}^{\ast}(t + 1) = \frac{a_j + b_j}{2} + \frac{(a_j + b_j)}{(2kn)} - \frac{x_{ij}(t)}{(kn)}. \quad (19)
\]

2.2.4. The Mutually Beneficial Factor. The individuals \(x_n\) were randomly selected, and \(x_m\) was determined by (20) to determine the mutually beneficial factor \(C\) [25].

\[
C = \frac{(x_h + x_m)}{2}, \quad (20)
\]

\[
x_{ij}(t + 1) = x_{ij}(t) \times \varphi \times (x_{\text{gbest}} - C \times R), \quad (21)
\]

where \(i\) represents the \(i\)th individual, \(j\) represents the individual dimension, \(\varphi\) represents a random number in \((0, 1)\), \(x_{\text{gbest}}\) represents the \(j\)-dimensional component of the current optimal position of the entire population, \(C\) is the mutual benefit factor, \(R\) is the benefit parameter, and 1 or 2 is randomly selected. After judging whether the fitness value is good or bad based on the newly generated individual position vector in (21), the original individual is replaced by the best one.

2.2.5. The Cauchy Variation. In this paper, the Cauchy inverse can mutate the population under certain probability. The Cauchy inverse function [26] is shown in (22). Referring to (22), we can write the new position of the individual as (23); that is, the new position of the individual is obtained by the Cauchy mutation. After judging whether the fitness value is good or bad based on the newly generated individual position vector in (23), the original individual is replaced by the best one.

\[
F^{-1}(p; x_0, y) = x_0 + y \cdot \tan(\pi \cdot (p - \frac{1}{2})). \quad (22)
\]

\[
x_{ij}(t + 1) = x_{ij}(t) + r_1 \cdot \tan(\pi \cdot (r_2 - \frac{1}{2})). \quad (23)
\]

where \(F^{-1}\) is the Cauchy inverse function and \(r_1\) and \(r_2\) are random values within \([0, 1]\).

2.2.6. Elastic Collision Variation. For the individual \(x_{ij} (x_{ij}\) is a solution distributed in the solution space of the optimization problem and can be abstractly represented as a unit mass object at a certain position in the space), \(\delta = \{x'(x' \in P (t) \land x' \neq x_i)\}; x_{ij} and x' move in each other’s direction at the velocities \(f(x_{ij})\) and \(f(x')\), respectively; and \(x_{ij}\) and \(x'\) will collide at \(\Delta t\), and then after \(\Delta t\), \(x_i\) reaches the new position \(x_{i,\text{new}}\). The derivation is as follows. For the complete elasticity (CE) collision, according to the law of conservation of momentum and energy [27],

\[
m f(x) + m f(x') = m f(x_{\text{new}}) + m f(x_{\text{new}}'), \quad \text{CE:}
\]

\[
\frac{1}{2} m[f(x)]^2 + \frac{1}{2} m[f(x')]^2 = \frac{1}{2} m[f(x_{\text{new}})]^2 + \frac{1}{2} m[f(x_{\text{new}}')]^2,
\]

\[
x_{\text{new}} = CE(x + x') = \frac{2 f(x')}{f(x') + f(x)} + x' \frac{f(x) - f(x')}{f(x') + f(x)}
\]
Similarly, for the complete inelastic (CI) collisions and the non-complete elastic (NCE) collisions,

\[
CI: \quad mf(x) + mf(x') = 2mf(x_{\text{new}}),
\]

\[
x_{\text{new}} = CI(x + x') = x + \frac{3f(x') - f(x)}{2(f(x') + f(x))} + x' \frac{3f(x) - f(x')}{2(f(x') + f(x))},
\]

\[NCE: \quad x_{\text{new}} = NCE(x + x') = \beta CE(x + x'); \beta \in (0, 1).\]

The individual updating mechanism is as follows:

\[
x_{ij}(t + 1) = \begin{cases} 
    ECO(x_{ij}(t), \delta), & r_1 \geq \varepsilon, \\
    x_{ij}(t), & r_1 < \varepsilon,
\end{cases}
\]

\[
ECO(x_{ij}(t), \delta) = \begin{cases} 
    CE(x_{ij}(t), G_{ij}(t)), & r_2 \leq 0.5, \\
    CI(x_{ij}(t), B_{ij}(t)), & 0.5 < r_2 \leq 0.5 + \xi, \\
    \alpha[CE(x_{ij}(t), G_{ij}(t)) + CI(x_{ij}(t), B_{ij}(t))], & \varepsilon \leq r_2 < 1,
\end{cases}
\]

where \(i\) represents the \(i\)th individual, \(j\) represents the individual dimension, \(\varepsilon \in (0, 1)\) and \(\xi \in (0, 0.5)\) are newer systems, \(G_{ij}(t)\) is the optimal solution of \(x_{ij}(t)\) history, and \(B_{ij}(t)\) is the optimal solution of species. \(r_1 \in (0, 1)\), \(r_2 \in (0, 1)\), and \(\alpha \in (0, 0.5)\) are the random numbers. After judging whether the fitness value is good or bad based on the newly generated individual position vector in (27), the original individual is replaced by the best one.

### 3. ECSOA

The ECSOA evolves some individuals in the CE collision, the CI collision, and the NCE collision to improve the diversity of individuals and boost partial scouting. Algorithm 1 is the primary process of the ECSOA.

### 4. Experimental Results

#### 4.1. Experimental Setup

The algorithms used in the experiments in this paper were running under MATLAB R2016a. The computer is configured as Intel® Core™ i7-7500U CPU @2.7 GHz, 2.9 GHz processor with 8 GB of memory and Windows 10 operating system.

#### 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. 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, 37–39]. Table 1 shows the functions in the experiment. Variables are set to one hundred.

#### 4.2.2. Performance Comparison of SOA with Different Improvement Methods

In this paper, the SOA is improved by seven different methods: the parameter changing SOA (PCSOA), the parameter adaptive Gaussian transform SOA (PAGTSOA), the SOA based on Levy variation (LVSOA), the SOA based on refraction reverse learning mechanism (RRLSOA), the SOA based on mutually beneficial factor strategy (MBFSOA), the SOA based on Cauchy variation (CVSOA), and the elastic collision seeker optimization algorithm (ECSOA).

(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 [32], 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 will use these improved algorithms for experimental comparison and choose a relatively optimal improved algorithm to compare it with other advanced intelligent algorithms.

(2) Improved Algorithms’ Performance Comparison in Benchmark Functions. The SOA is improved in seven different ways: the SOA based on parameter change (PCSOA),
the SOA based on parameter adaptive Gaussian transform (PAGTSOA), the SOA based on Levy variation (LVSOA), the SOA based on refraction reverse learning mechanism (RRLOOA), the SOA based on mutual benefit factor strategy (MBFSOA), the SOA based on Cauchy variation (CVSOA), and the SOA based on elastic collision (ECSOA). 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 seven improved SOAs in fifteen-function optimization was compared in terms of the mean (Mean), standard deviation (Std.), best fitness (Best), program running time (Time), and 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 boldface indicates that the optimal result is better.

Based on Table 3, for the benchmark functions f1-f15, the comparison between the seven improved SOAs in this paper and the original SOA shows that the optimization result of the ECSOA is the best value. The mean (Mean), standard deviation (Std.), best fitness (Best), and best fitness rank (Rank) of the ECSOA were the best after 30 independent runs. The f1-f15 total program running time (Time) rank is the fourth among all the eight algorithms compared in this paper. The ECSOA is better than the others. According to the optimal fitness perspective of optimization accuracy and robustness, the ECSOA has the best optimization performance among the improved SOAs in this paper. Section 4.2.3 will compare the ECSOA with the other intelligent optimization algorithms that are widely used at present.

### 4.2.3. Performance Comparison of Different Algorithms in Benchmark Functions

To test the performance of the ECSOA, it is compared to the PSO, SA_GA, GSA, SCA, MVO, and SSA, using the fifteen benchmark functions [7, 10, 37-39] in Table 1, which have been widely used in the test.

#### (1) The Parameter Setting of Different Algorithms

In this section, the parameters’ set of the PSO [40], SA_GA [41], GSA [6], SCA [8], MVO [9], SOA [28], and ECSOA is presented. According to [6, 8, 9, 28, 40, 41], we did a lot of practice tests and comparative studies for the parameters set. Table 4 shows the parameters set of different algorithms.

#### (2) The Results Comparison of Different Algorithms in Benchmark Functions

The mean values, standard deviation, best fitness, and best fitness rank of the algorithms of 30 independent runs and the data of functions f1-f15 optimization results are shown in Table 5. The boldface indicates that the optimal outcome is better.
| Name               | Test functions                                                                 | Search          | Minimum       |
|--------------------|--------------------------------------------------------------------------------|-----------------|---------------|
| Sphere             | $f_1(x) = \sum_{i=1}^{n} x_i^2$                                                  | $[-100,100]$    | 0             |
| Schwefel 2.22      | $f_2(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|$                  | $[-10,10]$      | 0             |
| Rotated hypereLLipsoid | $f_3(x) = \sum_{i=1}^{n} x_i^2$                                              | $[-100,100]$    | 0             |
| Schwefel 2.21      | $f_4(x) = \max(|x_i|, 1 \leq i \leq n)$                                        | $[-100,100]$    | 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               | $f_6(x) = \sum_{i=1}^{n} (|x_i + 0.5|)^2$                                       | $[-100,100]$    | 0             |
| Quartic            | $f_7(x) = \sum_{i=1}^{n} ix_i^2 + \text{random (0, 1)}$                        | $[-1.28,1.28]$  | 0             |
| SumSquares         | $f_8(x) = \sum_{i=1}^{n} x_i^2$                                              | $[-10,10]$      | 0             |
| SumPower           | $f_9(x) = \sum_{i=1}^{n} |x_i|^{0.1}$                                         | $[-1.28,1.28]$  | 0             |
| Schwefel           | $f_{10}(x) = \sum_{i=1}^{n} -x_i\sin(\sqrt{|x_i|})$                          | $[-500,500]$    | $-418.9829\times100$ |
| Rastrigin          | $f_{11}(x) = \sum_{i=1}^{n} |x_i^2 - 10\cos(2\pi x_i)| + 10$                  | $[-5.12,5.12]$  | 0             |
| Ackley             | $f_{12}(x) = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{10\sum_{i=1}^{n} \cos (2\pi x_i)}$ | $[-32,32]$      | 0             |
| Griewank           | $f_{13}(x) = 1/4000 \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(x_i/\sqrt{n}) + 1$ | $[-600,600]$    | 0             |
| Penalized1         | $f_{14}(x) = \frac{1}{2} \left[ 10 \sin^2(\pi y_1) + \sum_{i=1}^{n} (y_i - 1)^2 [1 + \sin^2(\pi y_{i+1})] + \sum_{i=1}^{n} u(x_i, 10, 100, 4) \right] + \sum_{i=1}^{n} u(x_i, 10, 100, 4) \cdot \left\{ \begin{array}{ll} k(x_i - a)^+ \cdot (a - x_i) & x_i > a \\ 0 & a \leq x_i \leq a \end{array} \right.$ | $[-50,50]$      | $[-50,50]$     |
| Penalized2         | $f_{15}(x) = \frac{1}{2} \left[ \sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \right] + \sum_{i=1}^{n} u(x_i, 5, 100, 4) \cdot \left\{ \begin{array}{ll} k(x_i - a)^+ \cdot (a - x_i) & x_i > a \\ 0 & a \leq x_i \leq a \end{array} \right.$ | $[-50,50]$      | 0             |
Table 2: Parameter settings of SOA with different improvement methods.

| Algorithm | Parameters and values |
|-----------|-----------------------|
| SOA [5]   | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \). |
| PCSOA     | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.2 \). |
| PAGTSOA   | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.1\text{–}0.5 \). |
| LVSOA     | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.2 \), the Levy mutation probability: \( P_L = 0.8 \). |
| RRLSOA    | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.2 \), the refraction reverse probability: \( P_r = 0.8 \). |
| MBFSOA    | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.2 \), the mutually beneficial factor probability: \( P_m = 0.8 \). |
| CVSOA     | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.2 \), the Cauchy variation probability: \( P_C = 0.8 \). |
| ECSOA     | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.2 \), the elastic collision probability: \( P_e = 0.8 \). |

Table 3: Performance comparison of different SOA improvement strategies of 30 independent runs for benchmark functions.

| Function | Result | Algorithms |
|----------|--------|------------|
| \( f_1 \) (D = 100) | Mean 1.0524957 | 10.651095 |
|          | Std. 0.2051906 | 46.890641 |
|          | Best 0.8215967 | 6.627537 |
|          | Time 45.50061 | 55.33285 |
|          | Rank 8       | 7          | 6          | 3          | 2          | 5          | 4          | 1          |
| \( f_2 \) (D = 100) | Mean 13.002085 | 10.651095 |
|          | Std. 2.0008395 | 46.890641 |
|          | Best 9.4729280 | 6.627537 |
|          | Time 54.16754 | 51.23703 |
|          | Rank 8       | 4          | 7          | 6          | 3          | 2          | 5          | 6          | 1          |
| \( f_3 \) (D = 100) | Mean 9.8969+03 | 10.651095 |
|          | Std. 3.472e+03 | 46.890641 |
|          | Best 3.318+03  | 6.627537 |
|          | Time 423.27283 | 59.58934 |
|          | Rank 8       | 4          | 7          | 6          | 3          | 2          | 5          | 7          | 4          | 1          |
| \( f_4 \) (D = 100) | Mean 21.743821 | 10.651095 |
|          | Std. 5.6809434 | 46.890641 |
|          | Best 2.9045088 | 6.627537 |
|          | Time 47.98446 | 55.85934 |
|          | Rank 8       | 4          | 7          | 6          | 3          | 2          | 5          | 7          | 4          | 1          | 1          | 3          |
| \( f_5 \) (D = 100) | Mean 7.093+02 | 10.651095 |
|          | Std. 1.675e+02 | 46.890641 |
|          | Best 2.789+02  | 6.627537 |
|          | Time 54.51529 | 62.16955 |
|          | Rank 8       | 4          | 7          | 6          | 3          | 2          | 5          | 7          | 4          | 1          | 1          | 3          | 1          |
| \( f_6 \) (D = 100) | Mean 1.1365348 | 10.651095 |
|          | Std. 0.1751583 | 46.890641 |
|          | Best 0.7992740 | 6.627537 |
|          | Time 47.876927 | 52.58318 |
|          | Rank 8       | 6          | 5          | 7          | 4          | 1          | 2          | 3          | 1          | 2          | 3          |
value mean rank and all rank results from Table 5, the ECSOA has a strong optimization ability and strong robustness to a benchmark function.

Figure 2 shows the fitness curves of the best values for the benchmark functions f1–f15 (D = 100). As seen from Figure 2, the convergence of the ECSOA is faster, and the precision of the ECSOA is better.

Figure 3 is the ANOVA for the benchmark functions f1–f15 (D = 100). As seen from Figure 3, the ECSOA showed better robustness and improved SOA. Therefore, the ECSOA is a feasible solution in the optimization of benchmark functions.

### 4.2.4. Complexity Analysis

The calculational complexity of the SOA is O (NDM), N represents the total individual count, D represents the dimension count, and M represents the maximum count of algebras. The computational complexities for the sub-techniques are as follows:

- **f1 (D = 100)**
  
  \[
  \text{Time} = \frac{2.324 \times 10^4}{2.296 \times 10^{-4}} + 0.0018590 = 0.0977197
  \]
  
  - **f2 (D = 100)**
    
    Mean: 4.147e-05, Std: 3.105e-05, Best: 3.105e-05
    
    \[
    \text{Time} = \frac{3.105e-05}{0.0977197} = 0.1793979
    \]
  
  - **f3 (D = 100)**
    
    Mean: 4.147e-05, Std: 3.105e-05, Best: 3.105e-05
    
    \[
    \text{Time} = \frac{3.105e-05}{0.1793979} = 0.0018590
    \]
  
  - **f4 (D = 100)**
    
    Mean: 1.301e+02, Std: 6.9291e+01, Best: 6.9291e+01
    
    \[
    \text{Time} = \frac{1.301e+02}{0.0018590} = 0.0154282
    \]
  
  - **f5 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0154282} = 0.0320302
    \]
  
  - **f6 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f7 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f8 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f9 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f10 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f11 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f12 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f13 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f14 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
  
  - **f15 (D = 100)**
    
    Mean: 1.665e+01, Std: 8.050e+01, Best: 8.050e+01
    
    \[
    \text{Time} = \frac{1.665e+01}{0.0320302} = 0.0320302
    \]
Table 4: The parameters set of different algorithms.

| Algorithm | Parameters and values |
|-----------|-----------------------|
| PSO [40]  | Constant inertia: \( c_0 = 0.9 - 0.4 \), two acceleration coefficients: \( c_1 = c_2 = 1.4962 \). |
| SA_GA [41]| Selection probability: \( P_s = 0.6 \), crossover probability: \( P_c = 0.7 \), mutation scale factor: \( P_m = 0.05 \), initial temperature: \( T_0 = 100 \), temperature reduction parameter: \( P_r = 0.98 \). |
| GSA [6]   | The gravitational constant: \( G_0 = 100 \), alfa = 20. |
| SCA [8]   | The random numbers: \( r_1 = 0 - 2 \), \( r_2 = 0 - 2 \), \( r_3 = 0 - 2 \), \( r_4 = 0 - 1 \). |
| MVO [9]   | The wormhole existence probability: \( W_{E_{\text{Max}}} = 1 \), \( W_{E_{\text{Min}}} = 0.2 \), traveling distance rate: \( T_{D} = 0 - 1 \), the random numbers: \( r_1 = 0 - 1 \), \( r_2 = 0 - 1 \), \( r_3 = 0 - 1 \). |
| SOA [28]  | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \). (The same as Table 2.) |
| ECSOA     | The maximum membership degree value: \( U_{\text{max}} = 0.95 \), the minimum membership degree value: \( U_{\text{min}} = 0.0111 \), the maximum inertia weight value: \( W_{\text{max}} = 0.9 \), the minimum inertia weight value: \( W_{\text{min}} = 0.1 \), the empirical value: \( W = 0.2 \), the elastic collision probability: \( P_r = 0.8 \). (The same as Table 2.) |

Table 5: Performance comparison of algorithms for benchmark functions.

| Test functions | Result | Algorithms |
|----------------|--------|------------|
| \( f_1(D = 100) \) | Mean: 0.0031 | PSO | 3.018e+04 |
|                | Std.: 6.31E-04 | SA_GA | 6.21E+02 |
|                | Best: 0.0022 | GSA | 6.42E+03 |
|                | Rank: 2 | SCA | 40.269355 |
|                |          | MVO | 1.0524957 |
|                |          | SOA | 0 |
| \( f_2(D = 100) \) | Mean: 0.219280 | PSO | 118.6375 |
|                | Std.: 0.0260935 | SA_GA | 7.138767 |
|                | Best: 0.1732089 | GSA | 1.3795842 |
|                | Rank: 3 | SCA | 7.34 E+20 |
|                |          | MVO | 13.002085 |
|                |          | SOA | 0 |
| \( f_3(D = 100) \) | Mean: 2.27E+03 | PSO | 3.77e+05 |
|                | Std.: 1.30E+03 | SA_GA | 8.80E+03 |
|                | Best: 8.43E+02 | GSA | 1.84 E+05 |
|                | Rank: 2 | SCA | 4.24 E+04 |
|                |          | MVO | 9.87 E+03 |
|                |          | SOA | 0 |
| \( f_4(D = 100) \) | Mean: 1.4624169 | PSO | 89.626880 |
|                | Std.: 0.2035038 | SA_GA | 15.671857 |
|                | Best: 1.0929345 | GSA | 85.539422 |
|                | Rank: 2 | SCA | 49.557442 |
|                |          | MVO | 21.743821 |
|                |          | SOA | 0 |
| \( f_5(D = 100) \) | Mean: 2.515e+02 | PSO | 5.783e+07 |
|                | Std.: 59.018749 | SA_GA | 1.784e+04 |
|                | Best: 1.236e+02 | GSA | 6.427e+07 |
|                | Rank: 2 | SCA | 2.440e+03 |
|                |          | MVO | 7.093e+02 |
|                |          | SOA | 98.124573 |
| \( f_6(D = 100) \) | Mean: 0.0119147 | PSO | 3.074e+04 |
|                | Std.: 0.0020782 | SA_GA | 6.995e+02 |
|                | Best: 0.008661 | GSA | 6.131e+03 |
|                | Rank: 1 | SCA | 40.0892130 |
|                |          | MVO | 1.1365348 |
|                |          | SOA | 0.0471172 |
| \( f_7(D = 100) \) | Mean: 0.2211229 | PSO | 77.487242 |
|                | Std.: 0.0419681 | SA_GA | 2.3136998 |
|                | Best: 0.1189066 | GSA | 73.024146 |
|                | Rank: 2 | SCA | 0.3355664 |
|                |          | MVO | 3.4645959 |
|                |          | SOA | 9.621e-05 |
| \( f_8(D = 100) \) | Mean: 0.0624146 | PSO | 1.29e+04 |
|                | Std.: 0.0164020 | SA_GA | 1.32E+02 |
|                | Best: 0.0291246 | GSA | 1.52E+03 |
|                | Rank: 2 | SCA | 1.05E+02 |
|                |          | MVO | 1.39E+02 |
|                |          | SOA | 0 |
| \( f_9(D = 100) \) | Mean: 2.08E-26 | PSO | 1.78e-12 |
|                | Std.: 3.99E-26 | SA_GA | 3.67E-12 |
|                | Best: 2.07E-29 | GSA | 35.077327 |
|                | Rank: 2 | SCA | 1.21E-06 |
|                |          | MVO | 1.47E-05 |
|                |          | SOA | 0 |
| \( f_{10}(D = 100) \) | Mean: -4.746e+3 | PSO | -2.475e+04 |
|                | Std.: 6.889e+02 | SA_GA | -4.729e+3 |
|                | Best: -6.440e+3 | GSA | -7.256e+3 |
|                | Rank: 7 | SCA | -2.408e+4 |
|                |          | MVO | -2.324e+4 |
|                |          | SOA | 0.2627e+4 |
complexity of the first phase of the SOA stage is O (NDM). The elastic collision strategy is introduced to calculate the O (NDM) value. Therefore, the overall complexity of the ECSOA is O (NDM + NDM). Based on the principle of the Big-O representation [42], if the count of algebras is high (M ≫ N, D), the calculational complexity is O (NDM).

Therefore, the overall calculational complexity of the ECSOA is almost the same as the basic SOA.

4.2.5. Statistical Testing of Algorithms in Benchmark Functions. Using Wilcoxon’s rank-sum test [43], we can discover the important differences between the two algorithms. This test gives the value \( p < 0.05 \).

Table 6 indicates the results of statistical testing. N/A represents the best algorithm. From Table 6, ECSOA is suitable for the fifteen functions. Therefore, the ECSOA is better than the other algorithms.

4.2.6. Run Time Comparison of Algorithms in Benchmark Functions. In this subsection, the running time of the algorithms for each function is recorded under the same conditions: population number of 30, evolution algebra of 1000, and 30 independent runs of the above fifteen benchmark functions \( f_1–f_{15} \) \( (D = 100) \). 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 7. As seen from Table 7, the SCA has the most minor program running time, followed by the PSO algorithm, which has more program running time. The ECSOA ranks fifth, which has a relatively longer program running time. At the bottom of the list is the SA_GA, which takes the most running time.

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 is calculated by the following formulas:

\[
rf_g = \frac{\mu_{f,g}}{\min[\mu_{f,g}: g \in G]},
\]

\[
\rho_g (\tau) = \frac{\text{size}\{f \in F: r_{f,g} \leq \tau\}}{n_f}.
\]

Table 5: Continued.

| Test functions | Result | PSO | SA_GA | GSA | SCA | MVO | SOA | ECSOA |
|----------------|--------|-----|-------|-----|-----|-----|------|-------|
| \( f_{11} (D = 100) \) | Mean    | 33.896488 | 425.8326 | 1.36E+02 | 2.06E+02 | 6.41E+02 | 4.15E+02 | 0 |
|                | Std.    | 6.235085 | 48.908770 | 18.532649 | 92.249647 | 69.443553 | 47.133647 | 0 |
|                | Best    | 23.655297 | 3.34E+02 | 88.161003 | 67.314606 | 5.01E+02 | 3.30E+02 | 0 |
|                | Rank    | 2 | 6 | 4 | 3 | 7 | 6 | 5 | 1 |
| \( f_{12} (D = 100) \) | Mean    | 0.0220555 | 15.347972 | 3.1378392 | 18.309269 | 6.5910720 | 2.4523845 | 8.882e–16 |
|                | Std.    | 0.0032392 | 0.7816167 | 0.6404753 | 4.7183569 | 6.0322718 | 0.3106583 | 0 |
|                | Best    | 0.0161767 | 13.796287 | 2.0521342 | 6.872015 | 3.2031355 | 1.9634230 | 8.882e–16 |
|                | Rank    | 2 | 7 | 4 | 6 | 5 | 3 | 2 | 1 |
| \( f_{13} (D = 100) \) | Mean    | 9.1115292 | 274.5795 | 98.761595 | 53.145123 | 1.3775687 | 0.5354221 | 0 |
|                | Std.    | 1.8646352 | 66.687333 | 11.527169 | 38.195663 | 0.0626040 | 0.3343659 | 0 |
|                | Best    | 6.1937260 | 137.5120 | 81.688324 | 67.719690 | 1.2241138 | 0.0806019 | 0 |
|                | Rank    | 5 | 7 | 6 | 4 | 3 | 2 | 1 | 0 |
| \( f_{14} (D = 100) \) | Mean    | 0.0167129 | 4.418e+07 | 4.5497727 | 1.582e+08 | 11.6567511 | 20.2319839 | 0.0355875 |
|                | Std.    | 0.0241246 | 5.110e+07 | 1.21396147 | 1.352e+08 | 4.2010589 | 8.8688013 | 0.0135230 |
|                | Best    | 4.838e–05 | 3.883e+06 | 2.1673113 | 6.757e+06 | 6.7785747 | 9.8451797 | 0.0020366 |
|                | Rank    | 1 | 6 | 3 | 7 | 4 | 5 | 2 | 1 |
| \( f_{15} (D = 100) \) | Mean    | 0.0010358 | 1.435e+08 | 1.309e+02 | 2.674e+08 | 1.209e+02 | 1.301e+02 | 7.6774427 |
|                | Std.    | 0.0028000 | 1.008e+08 | 64.254054 | 1.527e+08 | 30.717732 | 62.9791683 | 4.1034906 |
|                | Best    | 1.898e–04 | 2.309e+07 | 76.4103682 | 3.106e+07 | 51.2656139 | 1.7998261 | 0.0977197 |
|                | Rank    | 1 | 6 | 5 | 7 | 4 | 3 | 2 | 1 |

Average rank 2.4 6.466667 4.4 5.466667 4.4 3.666667 1.2
Overall rank 2 7 4 6 5 3 1

To learn more traits about the program running time of the seven algorithms in the fifteen functions, a bar chart in Figure 4 was made for the total time of each algorithm after 30 independent runs. From Figure 4, as to the running time, the ECSOA is less than the SA_GA and GSA; the SCA is the least; the SA_GA is the most; the ECSOA is less than one-sixth of SA_GA; and the ECSOA is nearly four times the SCA, which is relatively large.
Figure 2: Continued.
Figure 2: Continued.
shown in Figure 5, the ECSOA has the highest probability. When $\tau = 1$, the ECSOA is about 0.8, which is better than that of the others. When $\tau = 8$, the ECSOA is the winner on the given test functions, ESOA is 1, PSO is 0.67, SA_GA is 0.067, SCA is 0.3, GSA is 0.3, MVO is 0.2, and SOA is 0.33. Regarding the performance curve, the ECSOA is the best; the ECSOA can achieve 100% when $\tau \geq 1$. Thus, the performance of the ECSOA is better than that of the other algorithms.

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

4.3.1. Control System Models. Equations (30)–(33) show the control system models optimizing PID parameters used in our experiment. Figure 6 shows the process diagram for optimizing the test control system PID parameters by the ECSOA. Figure 7 shows the optimization PID parameter model structure of the control system.

\[
g_1(s) = \frac{2.6}{(2.7s + 1)(0.3s + 1)}, \tag{30}
\]

\[
g_2(s) = \frac{5}{(2.7s + 1)}e^{-0.5s}, \tag{31}
\]

\[
g_3(s) = \frac{3}{(2s + 1)}e^{-3s}, \tag{32}
\]

\[
g_4(s) = \frac{1}{(s + 1)^5}. \tag{33}
\]

4.3.2. Results Comparison of Algorithms in the PID Controller Parameter Optimization. For testing the capability of the
Figure 3: ANOVA tests for benchmark functions f1−f15 (D = 100).
Table 6: The $p$ values of the Wilcoxon rank-sum test.

| Test functions | $p$ test values of various algorithms |
|----------------|---------------------------------------|
| ($D = 100$)    | PSO | SA_GA | GSA | SCA | MVO | SOA | ECSOA |
| $f_1$          | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_2$          | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_3$          | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_4$          | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_5$          | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | N/A |
| $f_6$          | N/A | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | N/A |
| $f_7$          | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | N/A |
| $f_8$          | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_9$          | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_{10}$       | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | N/A |
| $f_{11}$       | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_{12}$       | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_{13}$       | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | 1.21178e-12 | N/A |
| $f_{14}$       | N/A | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | N/A |
| $f_{15}$       | N/A | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | 3.01986e-11 | N/A |

Table 7: Run time comparison of 30 independent runs for benchmark functions $f_1$–$f_{15}$ ($D = 100$).

| Functions ($D = 100$) | Run time of algorithms |
|------------------------|------------------------|
|                        | PSO | SA_GA | GSA | SCA | MVO | SOA | ECSOA |
| $f_1$                  | 19.2614 | 542.3931 | 150.8607 | 15.6790 | 37.1642 | 45.5006 | 63.004 |
| $f_2$                  | 22.358433 | 521.8564 | 154.253122 | 16.774635 | 22.505080 | 54.167535 | 68.65450 |
| $f_3$                  | 131.065635 | 3509.5453 | 276.851090 | 157.841395 | 150.595822 | 423.272832 | 720.86686 |
| $f_4$                  | 21.427574 | 642.475529 | 154.476221 | 17.295688 | 40.792862 | 47.984646 | 67.91751 |
| $f_5$                  | 19.859456 | 651.808449 | 188.478202 | 18.794625 | 41.892321 | 54.152921 | 79.987520 |
| $f_6$                  | 20.542143 | 480.833828 | 153.064086 | 17.318195 | 39.282567 | 47.876927 | 68.786422 |
| $f_7$                  | 34.792130 | 890.981756 | 168.550287 | 30.16487 | 51.654684 | 90.520669 | 162.014 |
| $f_8$                  | 23.367437 | 481.638596 | 153.526963 | 17.055855 | 39.908609 | 55.51679 | 68.06153 |
| $f_9$                  | 35.632177 | 975.1530 | 173.578067 | 32.396346 | 45.258449 | 102.887695 | 120.19844 |
| $f_{10}$               | 26.354619 | 562.020557 | 158.499353 | 20.396761 | 22.749008 | 60.474337 | 82.93795 |
| $f_{11}$               | 21.033824 | 648.6857 | 184.680588 | 22.147268 | 41.649193 | 59.599374 | 70.32757 |
| $f_{12}$               | 23.395074 | 624.41362 | 173.135274 | 21.299206 | 43.723453 | 64.892256 | 74.985 |
| $f_{13}$               | 29.505900 | 615.5540 | 169.814804 | 23.426107 | 46.092607 | 63.673136 | 82.41825 |
| $f_{14}$               | 60.288823 | 1654.958657 | 193.881593 | 52.751386 | 74.326149 | 165.947519 | 211.787815 |
| $f_{15}$               | 59.640283 | 1629.496415 | 196.205228 | 53.109655 | 75.621728 | 166.651257 | 207.40906 |
| The total time         | 548.524908 | 14431.84265 | 2649.855578 | 516.602609 | 773.214903 | 1503.476571 | 2149.389274 |
| Overall rank           | 2 | 7 | 6 | 1 | 3 | 4 | 5 |

Figure 4: The total time of 30 independent runs of 7 algorithms on fifteen benchmark functions.
ECSOA, it is compared with the PSO, SA_GA, GSA, SCA, MVO, and SOA in terms of the PID controller parameter optimization. The mean values, standard deviation values, best fitness values, and best fitness values rank of the algorithms of 30 independent runs for g1–g4 are displayed in Table 8. The boldface indicates that the optimal result is better.

For the PID controller parameter optimization problems, according to Table 8, except g3 and g4, as to the best fitness, the ECSOA is better than the others. The optimal fitness value result of the ECSOA for g3 model is only worse than the SA_GA, the optimal fitness value result of the ECSOA for g4 model is only worse than the PSO algorithm. As to the standard deviation results, for g1 model, the ECSOA is only worse than the SA_GA, SCA, and the MVO; for g2 and g3 models, the ECSOA is only worse than the SA_GA; and for g4 model, the ECSOA is only worse than the MVO. Except for g1 and g4, as to the mean test results, the ECSOA is better than the others; for g1 model, the ECSOA is only worse than the SCA; and for g4 model, the ECSOA is only worse than the MVO. According to the optimal fitness value mean rank and all rank results from Table 8, the
ECSOA can find solutions and has very strong robustness for the PID controller parameter optimization problems.

4.3.3. Convergence Curves Comparison of Algorithms in PID Controller Parameter Optimization. Figure 8 shows the fitness curves of PID controller parameter optimization for g1–g4. The comparison between the seven algorithms in Figure 8 shows that the convergence of the ECSOA is fast and the precision of the ECSOA is the best. The ECSOA can find the optimal value.

4.3.4. ANOVA Tests Comparison of Algorithms in PID Controller Parameter Optimization. Figure 9 is the ANOVA of the global best values PID controller parameter optimization for g1–g4. As seen from Figure 9, ECSOA is the most robust algorithm.

4.3.5. Unit Step Function PID Controller Parameter Optimization. Figure 10 shows the unit step function PID controller parameter optimization for g1–g4. As seen from Figure 10, the ECSOA is used to optimize the unit step function PID controller parameters of g1–g4, and the unit step functions tend to stabilize very quickly and accurately. Therefore, the ECSOA is an effective and feasible solution in the control system models optimizing PID parameters.

Table 8: Performance comparison of algorithms in PID parameter optimization of 30 independent runs.

| Test | Result | PSO | SA_GA | GSA | SCA | MVO | SOA | ECSOA |
|------|--------|-----|-------|-----|-----|-----|-----|-------|
| g1   | Mean   | 0.2267 | 0.3169 | 0.4571 | **0.0918** | 0.2501 | 0.1917 | 0.106057 |
|      | Std.   | 0.0877 | 0.0649 | 0.1569 | **0.0263** | 0.0532 | 0.11226 | 0.072471 |
|      | Best   | 0.0485 | 0.1002 | 0.2732 | 0.0483 | 0.0513 | 0.05774 | **0.047885** |
|      | Rank   | 3 | 6 | 7 | 2 | 4 | 5 | 1 |
| g2   | Mean   | 58.4757 | 62.4599 | 60.7787 | 24.8454 | 59.5805 | 42.1538 | **3.846738** |
|      | Std.   | 7.75976 | 5.3034 | 7.6556 | 21.5239 | 27.9025 | 2.706181 | 1.434925 |
|      | Best   | 36.0409 | 62.0356 | 42.7711 | 0.4898 | 32.6095 | 0.4898 | **0.310553** |
|      | Rank   | 5 | 7 | 6 | 2 | 3 | 4 | 1 |
| g3   | Mean   | 1.8481e+2 | 2.7179e+2 | 2.7665e+2 | 10.0458 | 1.0848e+2 | 2.6269e+2 | **14.234925** |
|      | Std.   | 59.6434 | **0.62334** | 10.3088 | 11.9839 | 56.6750 | 44.8106 | 28.40193 |
|      | Best   | 32.5445 | **2.71191** | **2.7139e+2** | 14.5588 | 20.0492 | 26.5763 | 8.737658 |
|      | Rank   | 6 | 7 | 6 | 2 | 3 | 4 | 1 |
| g4   | Mean   | 1.7713e+2 | 55.3556 | 2.3413e+2 | 85.196656 | **35.721213** | 46.10528 | 35.83705 |
|      | Std.   | 4.2182e+2 | 36.00807 | 2.1754e+2 | 10.8393 | **1.0050e+2** | 34.613162 | 34.745734 |
|      | Best   | **34.625063** | 34.6294 | 58.321733 | 34.867448 | 34.643162 | 34.625510 | 34.625510 |
|      | 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 |
4.4. Algorithm Performance Comparison in Constrained Engineering Optimization Problems. We are using six constrained engineering problems to test the capability of the ECSOA further. These constrained engineering problems are very popular in the literature. The penalty function is used to calculate the constrained problem. The parameters set for all of the heuristic algorithms still adopt the parameter setting in Table 4 of section 4.2.3. The formulations of these problems are available in the appendix.

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 11 [7]. Some of the algorithms are taken from other literature as follows: GSA [6], MFO [7], MVO [9], CPSO [45], and HS [46]. For the problem in this paper, the ECSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and SOA and provides the best-obtained values in Table 9.

In Table 9, the ECSOA is better than GSA, MFO, MVO, GA, CPSO, and HS algorithms in other literature. The ECSOA is also better than the PSO, SA_GA, GSA, SCA, MVO, and SOA. Therefore, the ECSOA can resolve the problem.

4.4.2. Pressure Vessel Design Problem. This is also the least fabrication cost problem of four parameters and four constraints. The parameters of the structural system are shown in Figure 12 [7]. Some of the algorithms are taken from other literature as follows: MFO [7], ES [47], DE [48], ACO [49], and GA [50]. For the problem, the ECSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and SOA and provides the best-obtained values in Table 10.
For the problem, the ECSOA is better than the MFO, ES, DE, ACO, and GA algorithms in other literature. The ECSOA is also better than the PSO, SA_GA, GSA, SCA, and MVO. There is not much difference between the optimal value of ESOA and that of SOA. Therefore, ECSOA can resolve the problem.

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 variables of constraints. The parameters of the structural system are shown in Figure 13 [7]. Some of the algorithms are taken from other literature as follows: MFO [7], CS [51], GCA [52], MMA [52], and SOS [53]. For the problem, the ECSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and SOA and provides the best-obtained values in Table 11.

In Table 11, the ECSOA proves to be better than the MFO, CS, GCA, MMA, and SOS algorithm in other literature. The ECSOA is also better than the PSO, SA_GA, GSA, SCA, and MVO. There is not much difference between the optimal value of ECSOA and that of SOA. Therefore, the ECSOA can resolve the problem.

4.4.4. Gear Train Design Problem. This is a minimum gear ratio problem, which has four variables and a scope of variables of constraints. Figure 14 is the schematic diagram [7]. Some of the algorithms are taken from other literature as follows: MFO [7], MVO [9], CS [51], ABC [54], and MBA [54]. For the problem in this paper, the ECSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and SOA and provides the best-obtained values in Table 12.

In Table 12, the ECSOA proves to be better than the MFO, MVO, CS, ABC, and MBA algorithm in other literature. Except for the SA_GA, GSA, and PSO, the ECSOA is also better than the SCA, the MVO, and the SOA. The result of the ECSOA has reached the theoretical best solution, although the optimum of the ECSOA is worse than that of the SA_GA, GSA, and PSO. The ECSOA finds a new value. Therefore, the ECSOA can resolve the problem.
4.4.5. Three-Bar Truss Design Problem. This is a minimize weight problem under stress, which has two variables and only applies to the scope of the variables of constraints. Figure 15 is the schematic diagram of the components [7]. Some of the algorithms are taken from other literature as follows: MFO [7], MVO [9], CS [51], MBA [54], and DEDS [55]. For the problem, the ECSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and SOA and provides the best values in Table 13. In Table 13, except for the MVO and the PSO, the ECSOA is better than the others. The best value of the ECSOA has reached the theoretical best solution, although the optimum of the ECSOA is worse than that of the MVO and the PSO. Therefore, the ECSOA can solve the problem.

4.4.6. I-Beam Design Problem. This is a minimize vertical deflection problem that has four variables and a constraint. Figure 16 is the design diagram [7]. Some of the algorithms are taken from other literature as follows: MFO [7], CS [51], SOS [53], IARSM [56], and ARSM [56]. For the problem, the ECSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and SOA and provides the best-obtained values in Table 14. In Table 14, except for the MFO, GSA, SOA, and SA_GA, the ECSOA is better than the others. The fitness of the MFO is the best. Although the most minor vertical deviation of the ECSOA is not as good as that of the GSA, the SOA, and the SA_GA, it is very close to other relative optimal values. Therefore, the ECSOA is an effective and feasible solution to the I-beam design optimization problem.

In brief, the ECSOA proves to be better than the other algorithms in most actual studies. The ECSOA can resolve these practical problems.
Table 9: Comparison results of the welded beam design problem.

| Algorithm   | Optimal values for variables | Optimal cost | Rank |
|-------------|------------------------------|--------------|------|
| 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 [45]   | 0.202369 3.544214 9.048210   | 0.205723     | 1.72802 | 7    |
| HS [46]     | 0.2442 6.2231 8.2915         | 0.2443       | 2.3807  | 12   |
| PSO         | 0.20437461682 3.27746206207 9.03907307954 0.20573458197 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 3.23348182622 9.40574225336 0.20795790595 1.76704865429 | 8    |
| MVO         | 0.20397627841 3.28970350716 9.03536739179 0.20582407425 1.69811381975 | 4    |
| SOA         | 0.19348578918 3.489546622637 9.027790656861 0.20615302629 1.69714450048 | 3    |
| ECSOA       | 0.18588842973 3.68013189994 9.06091584266 0.20569607018 1.69934872979 | 1    |

Table 10: Comparison results for pressure vessel design problem.

| Algorithm   | Optimal values for variables | Optimal cost | Rank |
|-------------|------------------------------|--------------|------|
| MFO [9]     | 0.8125 0.4375 42.098445 176.636596 | 6059.7143 | 8    |
| ES [47]     | 0.8125 0.4375 42.098087 176.640518 | 6059.7456 | 10   |
| DE [48]     | 0.8125 0.4375 42.098411 176.637690 | 6059.7340 | 9    |
| ACO [49]    | 0.8125 0.4375 42.103624 176.572656 | 6059.0888 | 7    |
| GA [50]     | 0.8125 0.4375 42.097398 176.654050 | 6059.9463 | 11   |
| PSO         | 0.93627266112 0.41391783346 47.19018939907 123.0285131625 | 6317.0167340514 | 12   |
| SA_GA       | 0.8380497369 0.41223740796 45.10610463950 142.6407851569 | 5931.2868373440 | 5    |
| GSA         | 0.89533101776 0.43654377356 47.8964096198 115.9627925902 | 6057.9305553133 | 6    |
| SCA         | 0.7116523790 0.39215740603 40.39056304889 200.0000000000 | 5903.0336988822 | 4    |
| MVO         | 0.75462696023 0.37830685291 40.94839768196 191.64503059607 | 5764.4347452930 | 3    |
| SOA         | 0.76961590364 41.5284631287 0.388196715944 183.8417207932 | 5735.135960512 | 1    |
| ECSOA       | 1.08847662958 0.52082333958 57.69587709181 47.007639941708 | 5736.531519781 | 2    |

Figure 12: Pressure vessel design problem.

Figure 13: Cantilever beam design problem.
### Table 11: Comparison results for cantilever beam design problem.

| Algorithm   | $x_1$       | $x_2$       | $x_3$       | $x_4$       | $x_5$       | Optimum weight | Rank |
|-------------|-------------|-------------|-------------|-------------|-------------|----------------|------|
| MFO [7]     | 5.9848717732 | 5.3167269243 | 4.4973325858 | 3.5136164677 | 2.1616202934 | 1.339988086    | 6    |
| CS [51]     | 6.0089      | 5.3049      | 4.5023      | 3.5077      | 2.1504      | 1.33999        | 7    |
| GCA [52]    | 6.0100      | 5.3000      | 4.4900      | 3.4900      | 2.1500      | 1.3400         | 8    |
| MMA [52]    | 6.0100      | 5.3000      | 4.4900      | 3.4900      | 2.1500      | 1.3400         | 8    |
| SOS [53]    | 6.011878    | 5.30344     | 4.9587      | 3.49896     | 2.15564     | 1.3396         | 3    |
| PSO         | 6.007219438 | 5.311747232 | 4.505611438 | 3.4904346887 | 2.158626706 | 1.339963522    | 4    |
| SA_GA       | 6.251285023 | 5.460509756 | 4.149903106 | 3.8032391760 | 1.974102742 | 1.350285757    | 11   |
| GSA         | 6.020285873 | 5.305304583 | 4.512114944 | 3.493972220 | 2.142187864 | 1.339969652    | 5    |
| SCA         | 5.801308754 | 5.589807963 | 4.97563735  | 3.4994713866 | 2.262668613 | 1.351011196    | 12   |
| MBA [54]    | 6.017949991 | 5.336576175 | 4.93102726  | 3.4797461041 | 2.146292918 | 1.340024388    | 10   |
| SOA         | 6.014092415 | 5.315532928 | 4.484154000 | 3.503360363 | 2.156331174 | 1.33957510     | 2    |
| ECSOA       | 5.993351697 | 5.332166494 | 4.470567257 | 3.5421087592 | 2.137284902 | 1.33957510     | 2    |

### Table 12: Comparison results of the gear train design problem.

| Algorithm   | $n_A$ | $n_B$ | $n_C$ | $n_D$ | Optimal gear ratio | Rank |
|-------------|-------|-------|-------|-------|--------------------|------|
| MFO [7]     | 43    | 19    | 16    | 49    | 2.709e-012         | 7    |
| MVO [9]     | 43    | 16    | 19    | 49    | 2.709e-012         | 7    |
| CS [51]     | 43    | 16    | 19    | 49    | 2.709e-012         | 7    |
| ABC [54]    | 49    | 16    | 19    | 49    | 2.709e-012         | 7    |
| MBA [54]    | 43    | 16    | 19    | 49    | 2.709e-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 | 46.9161662381 | 5.431797718e-12   | 12   |
| MVO         | 60.0000000000 | 12.0000000000 | 41.864783833  | 58.0329758032 | 2.334953506e-16   | 5    |
| SOA         | 60.0000000000 | 12.0000000000 | 43.283502093  | 60.0000000000 | 2.567448245e-16   | 6    |
| ECSOA       | 58.3864369781 | 18.85988647288 | 21.2811710819 | 47.6451311622 | 8.240738329e-19   | 4    |

Figure 14: Gear train design problem.

Figure 15: Three-bar truss design problem.
5. Conclusion

An ECSOA is presented, with a completely elastic collision, completely inelastic collision, and non-completely elastic collision method. According to the four-phase test of the ECSOA from different perspectives, it improved the SOA, the benchmark function optimization, the PID control parameter optimization problems, and the constrained engineering problems.

In the first phase, the SOA is improved in seven different ways: the SOA based on parameter change (PCSOA), the SOA based on parameter adaptive Gaussian transform (PAGTSOA), the SOA based on Levy variation (LVSOA), the SOA based on refraction reverse learning mechanism (RRLOOA), the SOA based on mutual benefit factor strategy (MBFSOA), the SOA based on Cauchy variation (CVSOA), and the SOA based on elastic collision (ECSOA). Each improved algorithm was optimized for the fifteen functions. The result is that the ECSOA is feasible in the benchmark functions. In this phase, we consider the ranking values of 30 independent runs between the ECSOA mean values, the standard deviation values, the best fitness values, the best fitness values rank, the convergence curves, and the variance tests for the global minimum values.

In the second phase, fifteen benchmark function optimization problems are used to test the ECSOA further. The ECSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and SOA for verification. It was observed that the ECSOA is feasible and competitive in benchmark functions. The second test phase is also about the ranking values of 30 in independent runs between the ECSOA mean values, standard deviation values, the best fitness values, the best fitness values rank, convergence curves, and variance tests for the global minimum values.

In the benchmark function optimization problems, the complexity analysis of the ECSOA is

**Table 13: Comparison results of the three-bar truss design problem.**

| Algorithm  | $x_1$          | $x_2$          | Optimum weight | Rank |
|------------|----------------|----------------|----------------|------|
| MFO [7]    | 0.788244770931922 | 0.409466905784741 | 263.895979682 | 10   |
| MVO [9]    | 0.78860276 | 0.40845307 | 263.8958499 | 8    |
| CS [51]    | 0.78867 | 0.40902 | 263.9716 | 11   |
| MBA [54]   | 0.7885650 | 0.4085597 | 263.895822 | 9    |
| DEDS [55]  | 0.78867513 | 0.40824828 | 263.8958434 | 7    |
| PSO        | 0.788425434690935 | 0.4080596065985 | 263.8523456301364 | 2    |
| SA_GA      | 0.787321758116231 | 0.4112143996852 | 263.853291023197 | 5    |
| GSA        | 0.761893501005708 | 0.493138841375638 | 264.809905788021 | 12   |
| SCA        | 0.789922169362555 | 0.403817724788810 | 263.854185836347 | 6    |
| MVO        | 0.788407496115311 | 0.408135122885127 | 263.852346859033 | 1    |
| SOA        | 0.788530250484097 | 0.4079127972805087 | 263.852349457003 | 3    |

**Table 14: Comparison results for I-beam design problem.**

| Algorithm  | $b$    | $h$    | $t_w$    | $t_f$    | Optimum vertical deflection | Rank |
|------------|--------|--------|----------|----------|-----------------------------|------|
| MFO [7]    | 50     | 80     | 1.7647   | 5.0000   | 0.0066259                   | 1    |
| CS [51]    | 50     | 80     | 0.95     | 2.32167  | 0.0130747                   | 9    |
| SOS [53]   | 50     | 80     | 0.9      | 2.32179  | 0.0130741                   | 8    |
| IARSM [56] | 48.42  | 79.99  | 0.9      | 2.40     | 0.131                       | 11   |
| ARSM [56]  | 37.05  | 80     | 1.71     | 2.31     | 0.0157                      | 10   |
| PSO        | 29.23495045988 | 77.7790428198 | 5.0000000000 | 3.5987373218 | 0.014625320 | 12   |
| SA_GA      | 34.9999839459 | 79.999964294 | 5.0000000000 | 4.9999823841 | 0.0078637302 | 5    |
| GSA        | 35.0000000001 | 80.0000000000 | 5.0000000000 | 5.0000000000 | 0.0078636959 | 2    |
| SCA        | 34.9878089422 | 80.0000000000 | 5.0000000000 | 5.0000000000 | 0.0078636963 | 6    |
| MVO        | 34.9999614894 | 80.0000000000 | 5.0000000000 | 5.0000000000 | 0.0078636963 | 3    |
| SOA        | 34.9999002914 | 80.0000000000 | 5.0000000000 | 5.0000000000 | 0.0078636963 | 3    |
| ECSOA      | 34.99997548457 | 79.9999999921 | 4.9999999961 | 4.9999999961 | 0.0078636983 | 3    |

**Figure 16: I-beam design problem.**

---

**Table 15: Comparison results of I-beam design problem.**

| Algorithm  | $b$    | $h$    | $t_w$    | $t_f$    | Optimum weight | Rank |
|------------|--------|--------|----------|----------|----------------|------|
| MFO [7]    | 50     | 80     | 1.7647   | 5.0000   | 263.8523456301364 | 2    |
| CS [51]    | 50     | 80     | 0.95     | 2.32167  | 263.853291023197 | 5    |
| SOS [53]   | 50     | 80     | 0.9      | 2.32179  | 264.809905788021 | 12   |
| IARSM [56] | 48.42  | 79.99  | 0.9      | 2.40     | 263.852349457003 | 3    |
researched, and the overall calculational complexity of the ECSOA is almost the same as that of the basic SOA. Wilcoxon’s rank-sum test is studied, and the ECSOA proves to be better than the other six algorithms. Based on the run time comparison of seven algorithms in benchmark functions, the ECSOA has relatively more program running time, and it is not optimal in terms of running time. From the results of the performance ratios of the average solution for the seven algorithms, the optimization probability of the ECSOA is the highest.

In the third phase, the four PID control parameter optimization models were used to test the ECSOA 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 micro delay, 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 ECSOA mean values, standard deviation values, best fitness values, best fitness values rank of 30 independent runs, convergence curves, and ANOVA. From the results of PID parameter optimization problems, the ECSOA was compared to various algorithms. The results show that the ECSOA is effective and feasible in practical problems.

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

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

(1) The elastic collision strategy includes the completely elastic collision, the completely inelastic collision, and the noncomplete elastic collision. The three different situations of elastic collision strategy tend to generate random seekers, increase the diversity of the seeker, increase the search space, and avoid premature convergence.

(2) Among the eight improved algorithms (PCSQA, PAGTSAO, ECSAO, LVSOA, RRLOOA, MBFSOA, CVSOA, and ECSOA), the ECSOA performed best in the benchmark functions test.

(3) Among the seven algorithms (PSO, SA_GA, GSA, SCA, MVO, SOA, and ECSOA), the ECSOA optimization benchmark function has the highest optimization capability.

(4) The ECSOA optimization benchmark functions have almost the same calculational complexity as the SOA.

(5) The running time of the ECSOA optimization benchmark function is relatively high. Among the seven algorithms compared, the running time is only better than that of the SA_GA.

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

(7) Further improving and application can be incorporated into future studies. The improved SOA and the heuristic algorithms based on those improved strategies can be applied not only to engineering optimization problems, but also to path planning problems, pattern recognition, intelligent control and other fields, and many practical application optimization problems that cannot be solved by traditional methods. Except the methods used in the paper, some of representative computational intelligence algorithms can be used to solve the problems, such as the MBO, EHO, MS, SMA, and HHO.

Appendix

A. Welded Beam Design Problem

Consider $\mathbf{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b]$, and minimize $f(\mathbf{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2)$, subject to

$g_1(\mathbf{x}) = \tau(\mathbf{x}) - \tau_{\text{max}} \leq 0,$ (A.1)

$g_2(\mathbf{x}) = \sigma(\mathbf{x}) - \sigma_{\text{max}} \leq 0,$ (A.2)

$g_3(\mathbf{x}) = x_1 - x_4 \leq 0,$ (A.3)

$g_4(\mathbf{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \leq 0,$ (A.4)

$g_5(\mathbf{x}) = 0.125 - x_1 \leq 0,$ (A.5)

$g_6(\mathbf{x}) = \delta(\mathbf{x}) - \delta_{\text{max}} \leq 0,$ (A.6)

$g_7(\mathbf{x}) = P - P_c(\mathbf{x}) \leq 0.$ (A.7)

Variable ranges are $0.1 \leq x_1 \leq 2$, $0.1 \leq x_2 \leq 10$, $0.1 \leq x_3 \leq 10$, and $0.1 \leq x_4 \leq 2$, where

$\tau(\mathbf{x}) = \sqrt{(r')^2 + 2r'\tau''(x_2/2R) + (\tau'')^2}$

$r' = (P/\sqrt{2x_1x_2}), r'' = (MP/\tau M = P(L + (x_2/2))$,

$R = \sqrt{(x_2/4) + (x_1 + x_2/2)^2}$

$J = 2\sqrt[3]{2x_1x_2[[x_2^3/4) + (x_1 + x_3/2)^3]}$,

$\sigma(\mathbf{x}) = (6PLx_3)^2$,

$\delta(\mathbf{x}) = (6PL^3/E_1x_3^3)$,

$P_c(\mathbf{x}) = (4.013E/\sqrt{3}\sqrt{36/L^2})(1 - (x_3/2L))\sqrt{(E/4G)}$.

$P = 6000 \text{ lb}, L = 14 \text{ in}, E = 30 \times 10^6 \text{ psi}, G = 12 \times 10^6 \text{ psi},$

$\tau_{\text{max}} = 13600 \text{ psi}, \sigma_{\text{max}} = 30000 \text{ psi}, \delta_{\text{max}} = 0.25 \text{ in}$.

B. Pressure Vessel Design Problem

Consider $\mathbf{x} = [x_1, x_2, x_3] = [T, T', R, L]$, and minimize $f(\mathbf{x}) = 0.6224x_1x_2x_4 + 1.7781x_2x_3 + 3.1661x_1^2x_4 + 19.84x_2x_3$, subject to
\[ g_1(x) = -x_1 + 0.0193x_3 \leq 0, \quad (B.1) \]
\[ g_2(x) = -x_2 + 0.00954x_3 \leq 0, \quad (B.2) \]
\[ g_3(x) = -\pi x_2^2x_4 - \frac{4}{3}\pi x_4^3 + 129600 \leq 0, \quad (B.3) \]
\[ g_4(x) = x_4 - 240 \leq 0. \quad (B.4) \]

Variable ranges are 0 \leq x_1 < 99, 0 \leq x_2 < 100, 10 \leq x_3 < 200, 10 \leq x_4 < 200.

### C. Cantilever Design Problem

Consider \( \vec{x} = [x_1, x_2, x_3, x_4, x_5] \), and minimize \( f(\vec{x}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5) \), subject to \( g(\vec{x}) = 61/x_1^3 + 37/x_3^2 + 19/x_3^2 + 7/x_1^2 + 4\pi x_1^2 - 1 \leq 0 \).

Variable ranges are 0.01 \leq x_1, x_2, x_3, x_4, x_5 \leq 100.

### D. Gear Train Design Problem

Consider \( \vec{x} = [x_1, x_2, x_3, x_4] = [n_A, n_B, n_C, n_D] \), and minimize \( f(\vec{x}) = (1/6.931 - x_2/x_3)^2 \).

Variable ranges are 12 \leq x_1, x_2, x_3, x_4 \leq 60.

### E. Three-Bar Truss Design Problem

Consider \( \vec{x} = [x_1, x_2] = [A_1, A_2] \), and minimize \( f(\vec{x}) = (2\sqrt{2}x_1 + x_2) + l \), subject to
\[
\begin{align*}
g_1(\vec{x}) &= \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} - p - \sigma \leq 0, \quad (E.1) \\
g_2(\vec{x}) &= \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} - p - \sigma \leq 0, \quad (E.2) \\
g_3(\vec{x}) &= \frac{1}{\sqrt{2}x_2 + x_1} - p - \sigma \leq 0. \quad (E.3)
\end{align*}
\]

Variable ranges are 0 \leq x_1, x_2 \leq 1, l = 100cm, P = 2KN/cm², \( \sigma = 2KN/cm² \).

### F. I-Beam Design Problem

Consider \( \vec{x} = [x_1, x_2, x_3, x_4] = [h, h, t_{ws}, t_f] \), and minimize \( f(\vec{x}) = 5000/(x_1(x_2 - 2x_3)) + 1/12 + x_1x_2^2/6 + 2x_2x_4(x_2 - x_4)/2 \), subject to \( g(\vec{x}) = 2x_1x_3 - x_4(x_2 - 2x_4) \leq 0 \).

Variable ranges are 10 \leq x_1 \leq 50, 10 \leq x_2 \leq 80, 0.9 \leq x_3 \leq 5, 0.9 \leq x_4 \leq 5.

### 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 work was supported by the National Natural Science Foundation of China (Grant nos. 51766005 and 52166001) and the 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 6th International Symposium on 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: a new introduction," Statistica Neerlandica, vol. 43, no. 1, pp. 31–52, 1989.

[5] Z. W. Zong Woo Geem, J. H. Joong Hoon Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," Simulation, vol. 76, no. 2, pp. 60–68, 2001.

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

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

[8] S. Mirjalilii, "A Sine Cosine Algorithm for solving optimization problems," Knowledge-Based Systems, vol. 27, pp. 1–14, 2016.

[9] S. Mirjalilii, 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] J. Ji, S. Song, C. Tang, 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.

[12] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," Communications in Nonlinear Science and Numerical Simulation, vol. 17, no. 12, pp. 4831–4845, 2012.

[13] G.-G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," Neural Computing & Applications, vol. 31, no. 7, pp. 1995–2014, 2019.

[14] G.-G. Wang, S. Deb, and L. dos S. Coelho, "Elephant herding optimization," in Proceedings of the 2015 3rd International Symposium on Computational and Business Intelligence, pp. 1–5, Bali, Indonesia, December 2015.

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

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

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

[18] B. S. Yildiz and A. R. Yildiz, “The Harris hawks optimization algorithm, salp swarm algorithm, grasshopper optimization algorithm and dragonfly algorithm for structural design optimization of vehicle components,” *Materials Testing*, vol. 61, pp. 744–748, 2019.

[19] H. Jia, Z. Wei, and X. He, “Path planning based on improved particle swarm optimization algorithm,” *Transactions of the Chinese Society for Agricultural Machinery*, vol. 49, no. 12, pp. 371–377, 2018.

[20] L. d. 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.

[21] X. Song, L. Jian, and Y. Song, “A chunk updating LS-SVMs based on block Gaussian elimination method,” *Applied Soft Computing*, vol. 51, pp. 96–104, 2017.

[22] Y. Saji and M. Barkatou, “A discrete bat algorithm based on Levy flights for Euclidean traveling salesman problem,” *Expert Systems with Applications*, vol. 172, pp. 1–17, 2021.

[23] E. Shadkhani, “Cuckoo optimization algorithm in reverse logistics: a new tool for design for COVID-19 waste management,” *Waste Management & Research: The Journal for a Sustainable Circular Economy*, pp. 1–12, 2021.

[24] M. Kelidari and J. Hamidzadeh, “Feature selection by using chaotic cuckoo optimization algorithm with levy flight, opposition-based learning and disruption operator,” *Soft Computing*, vol. 25, no. 4, pp. 1–23, 2021.

[25] Y. Wang and Z. Ma, “Elite symbiotic organisms search algorithm based on subpopulation stretching operation,” *Control and Decision*, vol. 34, no. 7, pp. 1355–1364, 2019.

[26] X. Chu, S. Li, D. Gao, W. Zhao, J. Cui, and L. Huang, “A binary superior tracking artificial bee colony with dynamic Cauchy mutation for feature selection,” *Complexity*, vol. 2020, pp. 6–25, 2020.

[27] Z. Liu, S. Li, B. Li, and H. Wang, “New elastic collision optimization algorithm and its application in sensor cloud resource scheduling,” *Journal of Zhejiang University (Engineering Science)*, vol. 52, no. 8, pp. 1431–1443, 2018.

[28] C. Dai, Y. Zhu, and W. Chen, “Seeker optimization algorithm,” in *Proceedings of the 2006 International Conference Computational Intelligence and Security*, pp. 225–229, IEEE Press, Guangzhou, China, November 2006.

[29] C. Dai, Y. Zhu, and W. Chen, “Seeker optimization algorithm,” *Computational Intelligence and Security*, vol. 4456, pp. 167–176, 2007.

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

[31] 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.

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

[33] 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.

[34] C. Dai, W. Chen, 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.

[35] 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.

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

[37] L. Li, Y. Zhou, and J. Xie, “A free search krill herd algorithm for functions optimization,” *Mathematical Problems in Engineering*, vol. 2014, pp. 1–21, 2014.

[38] 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.

[39] M. Molga and C. Smutnicki, “Test functions for optimization needs,” 2005, http://www.robertmarks.org/Courses/ENGR5358/Papers/functions.pdf.

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

[41] H. Yu, H. Fang, P. Yao, and Y. 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.

[42] S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, *Algorithms*, pp. 15–18, China Machine Press, New York, NY, United States, 2009.

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

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

[45] R. A. Krohling and L. dos Santos 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, no. 6, pp. 1407–1416, 2006.

[46] 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, no. 36–38pp, 3902–3933, 2005.

[47] E. Mezura-Montes 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, no. 4, pp. 443–473, 2008.

[48] L. J. Li, Z. B. Huang, F. Liu, and Q. H. Wu, “A heuristic particle swarm optimizer for optimization of pin connected structures,” *Computers & Structures*, vol. 85, no. 7–8, pp. 340–349, 2007.

[49] A. Kaveh and S. Talatahari, “An improved ant colony optimization for constrained engineering design problems,” *Engineering Computations*, vol. 27, no. 1, pp. 155–182, 2010.

[50] K. Deb, “Optimal design of a welded beam via genetic algorithms,” *AIAA Journal*, vol. 29, no. 11, pp. 2013–2015, 1991.

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

[52] H. Chickermane and H. C. Gea, “Structural optimization using a new local approximation method,” International Journal for Numerical Methods in Engineering, vol. 39, no. 5, pp. 829–846, 1996.

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

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

[55] M. Zhang, W. Luo, and X. Wang, “Differential evolution with dynamic stochastic selection for constrained optimization,” Information Sciences, vol. 178, no. 15, pp. 3043–3074, 2008.

[56] G. G. Wang, “Adaptive response surface method using inherited Latin hypercube design points,” Journal of Mechanical Design, vol. 125, no. 2, pp. 210–220, 2003.