mathematics

MDPI

Article

Election Optimizer Algorithm: A New Meta-Heuristic Optimization Algorithm for Solving Industrial Engineering Design Problems

Shun Zhou [1,†], Yuan Shi [1,†], Dijing Wang [1], Xianze Xu [1,*], Manman Xu [2,*] and Yan Deng [3]

[1] School of Electronic Information, Wuhan University, Wuhan 430072, China; zhoushun1997@whu.edu.cn (S.Z.); shiyuan@whu.edu.cn (Y.S.); wangdijin@whu.edu.cn (D.W.)
[2] School of Mechanical Automation, Wuhan University of Science and Technology, Wuhan 430072, China
[3] School of Aeronautics and Intelligent Manufacturing, Hankou University, Wuhan 430072, China; 2063020001@hkxy.edu.cn
* Correspondence: xuxianze@whu.edu.cn (X.X.); xumanman@wust.edu.cn (M.X.)
† These author contributed equally to this work.

Abstract: This paper introduces the election optimization algorithm (EOA), a meta-heuristic approach for engineering optimization problems. Inspired by the democratic electoral system, focusing on the presidential election, EOA emulates the complete election process to optimize solutions. By simulating the presidential election, EOA introduces a novel position-tracking strategy that expands the scope of effectively solvable problems, surpassing conventional human-based algorithms, specifically, the political optimizer. EOA incorporates explicit behaviors observed during elections, including the party nomination and presidential election. During the party nomination, the search space is broadened to avoid local optima by integrating diverse strategies and suggestions from within the party. In the presidential election, adequate population diversity is maintained in later stages through further campaigning between elite candidates elected within the party. To establish a benchmark for comparison, EOA is rigorously assessed against several renowned and widely recognized algorithms in the field of optimization. EOA demonstrates superior performance in terms of average values and standard deviations across the twenty-three standard test functions and CEC2019. Through rigorous statistical analysis using the Wilcoxon rank-sum test at a significance level of 0.05, experimental results indicate that EOA consistently delivers high-quality solutions compared to the other benchmark algorithms. Moreover, the practical applicability of EOA is assessed by solving six complex engineering design problems, demonstrating its effectiveness in real-world scenarios.

Keywords: meta-heuristics; swarm intelligence algorithms; human-based optimization; industrial engineering design problems

MSC: 49K35

check for updates

Citation: Zhou, S.; Shi, Y.; Wang, D.; Xu, X.; Xu, M.; Deng, Y. Election Optimizer Algorithm: A New Meta-Heuristic Optimization Algorithm for Solving Industrial Engineering Design Problems. *Mathematics* **2024**, *12*, 1513. https://doi.org/10.3390/math12101513

Academic Editor: Andrea Scozzari

Received: 11 April 2024
Revised: 2 May 2024
Accepted: 9 May 2024
Published: 13 May 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, optimization problems have gained significant attention in various fields, including engineering, economics, and computer science [1]. For engineering optimization problems, it is of great potential to pursue the optimal or the best solution to enhance diverse targets, such as production safety, efficiency, and energy consumption. However, current engineering optimization problems exhibit distinctive characteristics of multi-constraints, non-linearities, and multi-modalities, making it increasingly challenging for traditional optimization methods to handle such problems [2–4]. Hence, the research of effective optimization strategies for engineering optimization problems is always a hot spot. Meta-heuristic algorithms possess strong adaptability, multimodal search, and global search capabilities compared to traditional optimization methods [5]. These attributes enable them

to effectively address the complexity and uncertainty of problems, offering high-quality solutions. Consequently, meta-heuristic algorithms are prone to solving high-dimensional and nonlinear problems in engineering.

As shown in Figure 1, meta-heuristic algorithms are mainly divided into four categories based on the design inspiration of the algorithms: swarm intelligence algorithms (SIs) [6], evolutionary algorithms (EAs) [7], physics-based algorithms (PBs) [8], and human-based algorithms (HBs) [9]. In the existing literature, SIs are proposed based on group behavior between organisms. The following are some common algorithms: particle swarm optimization (PSO) [10], honey badger algorithm (HBA) [11], arithmetic optimization algorithm (AOA) [12], whale optimization algorithm (WOA) [13], grey wolf optimizer (GWO) [14], slime mould algorithm (SMA) [15], and aquila optimizer (AO) [16]. EAs draw inspiration from biological evolution in nature, and the classical evolutionary algorithms mainly include genetic algorithm (GA) [17], evolution strategy (ES) [18], and differential evolution (DE) [19]. PBs solve optimization problems by simulating physical laws of operation and physical phenomena in the universe, such as simulated annealing (SA) [20], atomic orbital search (AOS) [21], electrostatic discharge algorithm (ESDA) [22], and gravitational search algorithm (GSA) [23]. The fourth category is mainly inspired by human behavior and human social laws. The representative HBs include soccer league competition (SLC) [24], social network search (SNS) [25], political optimizer (PO) [26], parliamentary optimization algorithm (POA) [27], and school based optimization (SBO) [28].

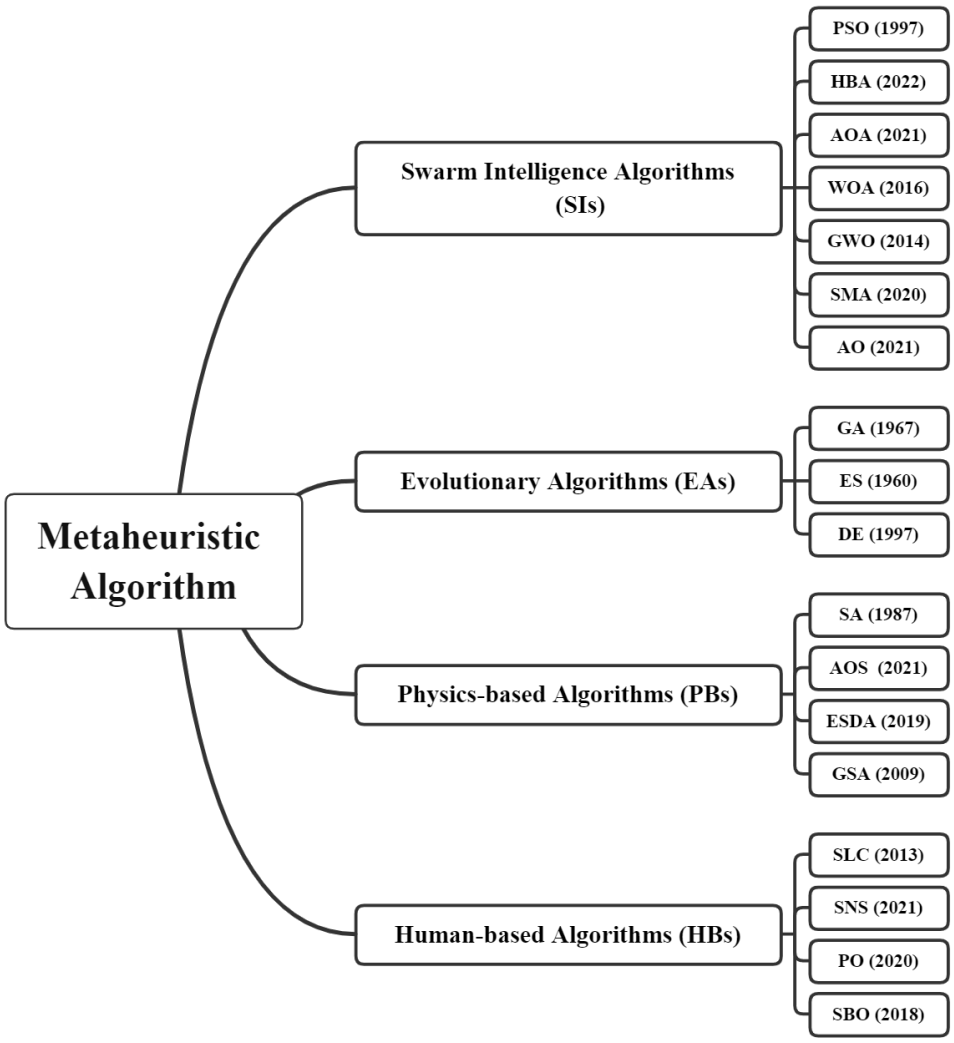


Figure 1. The classification of meta-heuristic algorithms.

As emphasized by the No Free Lunch Theorem, despite the existence and application of numerous meta-heuristic algorithms, no optimization algorithm can address all optimization problems; hence, there are still unresolved challenges. Therefore, it is necessary to further propose new algorithms to expand the range of solvable problems. To this end, this paper draws inspiration from the presidential election and presents a more efficient meta-heuristic algorithm, referred to as the election optimization algorithm (EOA). Existing political optimizers (POs) assign two identities to each candidate, in the party, and the constituency. Then POs update the relative position of the candidate concerning the party leader and the constituency winner. This position updating rule is beneficial for the algorithm to maintain a good convergence speed and exploration capability in the early iteration process. Compared to traditional POs, EOA divides the election process into party nomination and presidential campaign to simulate the alternating iterations of the exploration and exploitation phases. Furthermore, the two phases are subdivided. It is worth noting that a stochastic strategy reference factor is introduced in the party nomination phase to dynamically regulate the candidate's campaign strategy, preventing iteration from getting trapped in local optima. Additionally, a control factor is introduced in the presidential election phase to maintain population diversity while accelerating convergence speed. As a result, EOA can simulate the election process more comprehensively and thus achieve better convergence accuracy while maintaining good convergence speed and exploratory ability. In summary, the characteristics of EOA are summarized as follows:

- EOA imitates the complete process of the presidential election during the optimization process.
- EOA focuses on the explicit behaviors of candidates during the election process, such as strategy reference, innovative suggestions from the staff team, televised debates, and campaign speeches.
- EOA shows the superior effectiveness on each test and outperforms the existing HBs.

To evaluate the convergence speed, convergence precision, and diversity of EOA, we conducted rigorous experiments using twenty-three classical and ten IEEE CEC2019 benchmark functions. The results indicate that EOA outperforms other competing algorithms. Furthermore, we evaluate the effectiveness of EOA in solving real-world problems by using six engineering design problems, which are regarded as challenging test suites in the relevant literature. The findings demonstrate that EOA exhibits superior practical applicability for engineering problems.

The rest of the paper is structured as follows: Section 2 summarizes relevant works. Section 3 describes the proposed algorithm in detail. In Section 4, the proposed algorithm is simulated on benchmark functions to evaluate the performance. Section 5 applies the proposed algorithm to solve real engineering problems. Finally, Section 6 gives a conclusion to this paper.

2. Related Work

Currently, a large number of meta-heuristic algorithms have been proposed for solving engineering optimization problems. Wang et al. [29] proposed an artificial rabbit optimization method for solving the rolling bearing fault diagnosis problem. Li et al. [30] introduced an improved balanced optimizer based on multi-strategy optimization, which dynamically balances the exploration and development phases, demonstrating enhanced solving efficiency in engineering optimization problems. Xian et al. [31] presented a meerkat optimization algorithm that exhibits effectiveness and superiority in solving real engineering optimization problems with constraints. Abualigah et al. [12] introduced an arithmetic optimization algorithm utilizing mathematical modeling with the main arithmetic operators. The algorithm's performance has been tested in real engineering design problems, demonstrating its applicability in solving complex problems. Inspired by the foraging behavior of honeypots, Hashim et al. [11] developed a controllable and efficient search strategy that maintains population diversity throughout the search process. Optimization results on four engineering problems showed that the algorithm is particularly effective

in solving optimization problems in complex search spaces. Nadimi-Shahraki et al. [32] proposed a monkey king evolutionary algorithm based on multiple trial vectors, effectively avoiding the algorithm from falling into local optima by introducing a trial vector generator to balance the exploration and utilization ratio of the algorithm. Furthermore, the author provides a detailed summary of existing metaheuristic optimization algorithms and validates the effectiveness of the proposed algorithm by applying it to solve engineering optimization problems.

3. Election Optimizer Algorithm

3.1. Inspiration

Humans have created a splendid civilization with a rich history, in which political institutions play a vital role. Throughout history, various forms of political systems have emerged, including the slave system, monarchy, democratic election, etc. Democratic election, as a successful political system in the evolution of human politics, made great contributions to the development of modern human civilization. Democratic election refers to the system of electing political leaders and representatives through open, equal, and free voting procedures. In this system, all eligible citizens possess an equal opportunity to participate in the election, with the majority determining the voting outcomes. The elected candidates consequently represent the voters' will and assume public office. In essence, the goal of a democratic election is similar to that of an optimization algorithm, which seeks to identify the best solution within a given search space. A prominent example of a democratic election system is the presidential election. The optimization procedure employed by EOA mimics the presidential election process. Figure 2 illustrates the EOA optimization process, which can be illustrated by the following fundamental assumptions:

- The process of EOA is divided into two phases: party nomination and presidential election. The party nomination reflects the exploration phase, while the later one reflects the exploitation phase.
- In the party nomination phase, candidates win the nomination through two behaviors: strategy reference and innovative suggestions of the staff team.
- In the presidential election phase, candidates increase their poll approval rate via two approaches: televised debate and campaign speech.

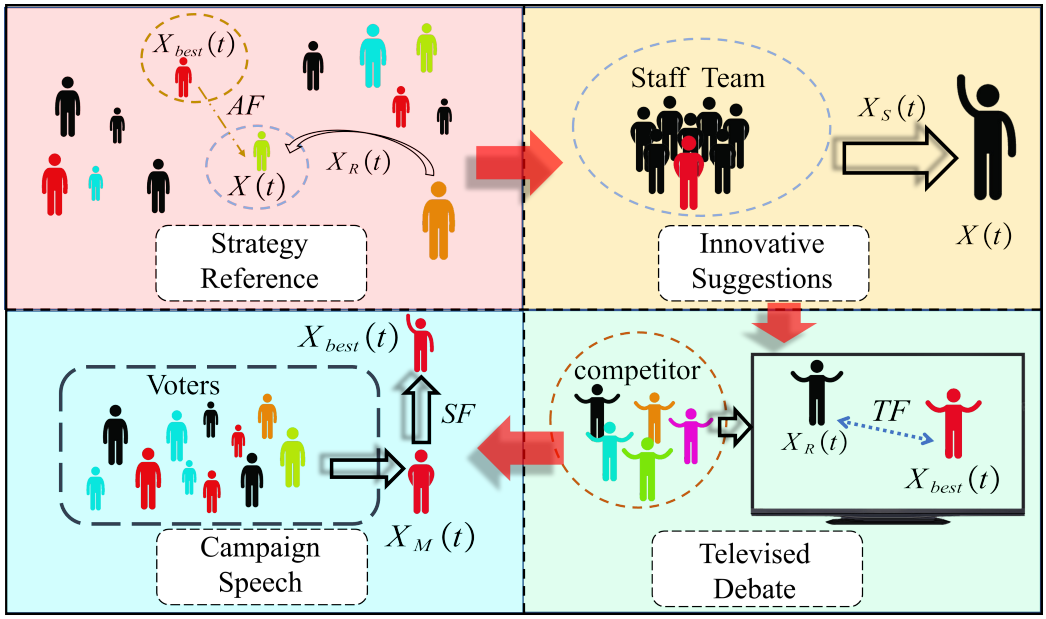


Figure 2. The overview of the EOA optimization process.

Although meta-heuristic algorithms adhere to various optimization rules, they all encompass exploration and exploitation phases. For EOA, the algorithm explores solution spaces to obtain diverse solutions during the party nomination phase, while simultaneously seeking the local optimum during the presidential election phase. The optimization process commences with a randomly generated set of candidate solutions. These solutions follow inherent optimization rules for updates and are assessed using a specific fitness function through iterative steps.

3.2. Initialization Model

In EOA, the initialization of candidates is expressed as follows: candidates are generated from a stochastic set within the strategy space's lower bound (*LB*) and upper bound (*UB*), which corresponds to the solution space of the given problem.

$$X = [X_1, \cdots, X_i, \cdots, X_N]^{\mathsf{T}} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,j} & \cdots & x_{1,Dim} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{i,1} & x_{i,2} & \cdots & x_{i,j} & \cdots & x_{i,Dim} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,j} & \cdots & x_{N,Dim} \end{bmatrix}, \tag{1}$$

where X_i refers to the i-th candidate in the set. N is the total number of candidates. Dim is the number of dimensions of candidate solutions, and $x_{i,j}$ denotes the j-th strategy of the candidate and is calculated as:

$$x_{i,j} = LB_j + \left(UB_j - LB_j\right) \times rand, \tag{2}$$

where $rand$ denotes a random number ranging from 0 to 1. LB_j and UB_j denote the lower and upper bounds for the j-th strategy within the strategy space, respectively.

3.3. Party Nomination

This section focuses on the party nomination behavior of EOA, which corresponds to the exploration phase of the optimization process. Candidates must first secure the party nominee status to participate in the presidential election. This can be achieved through two key behaviors: strategy reference (SR) and innovative suggestions from the staff team (ISST). These behaviors are specifically designed to enhance the competitiveness of candidates during the party nomination process.

3.3.1. Strategy Reference

This behavior entails candidates adjusting their campaign strategies by referencing other campaigners within the same party. To facilitate the SR behavior, the following position updating equation is proposed:

$$X_1(t+1) = X_R(t) + (X(t) - X_{best}(t)) \times AF, \tag{3}$$

where $X_1(t+1)$ represents the solution for the next iteration at the time $t+1$ and indicates the adjusted campaign strategies. $X_R(t)$ is a randomly selected candidate from the same party that serves as a reference for modifying the campaign strategy, and X_{best} represents the candidate with the highest support rating at the i-th iteration. AF is a stochastic strategy reference factor determined by the following equation:

$$AF = \alpha \times r_3 \cdot \exp\left(1 - \frac{t}{T}\right), \tag{4}$$

where α is a sensitive factor between 0 and 20. r_3 is a random number between 0 and 1. T represents the maximum iteration number, while t denotes the current iteration.

3.3.2. Innovative Suggestions of the Staff Team

This behavior entails each candidate having their own staff team, who propose innovative policy recommendations. Candidates enhance their poll support by dynamically adjusting their candidacy policies based on the innovative suggestions put forward by their staff team. The mathematical expression for this behavior is as follows:

$$X_2(t+1) = X(t) \times r_4 + r_5 \times X_S(t), \tag{5}$$

where r_4 and r_5 represent random numbers between 0 and 1. $X_S(t)$ denotes the innovative suggestions put forward by the staff team, which are calculated by using the following equation:

$$X_S(t) = (UB - LB) \times r_6 + LB, \tag{6}$$

where r_6 represents a random number between 0 and 1. Eligibility for the presidential election is attained upon securing the party nominee status. The subsequent phase of the iteration is described in detail below.

3.4. Presidential Election

Once the exploration phase is complete, elite candidates in party races compete further and maintain population diversity by controlling for the control factor SF. This section introduces the exploitation phase of EOA, which corresponds to the behavior observed during the presidential election. In this phase, candidates aim to boost their poll support through various campaigning activities, including televised debates (TD) and campaign speeches (CS). The process of the presidential election phase in EOA is modeled by incorporating these two behavioral traits, which will be described in detail below.

3.4.1. Televised Debate

In this behavior, candidates participate in televised debates with other party rivals to raise their approval ratings. During the debate, candidates modify their campaign strategies by consulting their opponents' policies. Its mathematical expression is presented below:

$$X_3(t+1) = X_{best}(t) + r_7 \times (X_{best}(t) - TF \times X_R(t)), \tag{7}$$

where r_7 is a random value between 0 and 1. X_{best} represents the optimal election strategies, while X_R denotes the strategies of the opponent from the rival party. TF is a random value between 0 to 2, which indicates the extent of reference to the opponent's campaign policies.

3.4.2. Campaign Speech

Candidates convey their campaign promises to voters through campaign speeches and adapt their campaign strategies based on the political aspirations of the electorate. The behavior of a candidate delivering a campaign speech can be expressed as follows:

$$X_4(t+1) = X_{best}(t) + (X_M(t) - X_{best}(t)) \cdot SF \times r_8, \tag{8}$$

where $X_M(t)$ denotes the mean value of the current solutions associated with the t-th iteration and is calculated using Equation (9). r_8 is a random value between 0 and 1. The control factor SF is determined by the equation $SF = (t+1)^{-\frac{2 \cdot r_9 + 1}{(T+1)^2}}$, where T is the maximum iteration number and r_9 is a random value between 0 and 1.

$$X_M(t) = \frac{1}{N} \sum_{i=1}^{N} X_i(t) \tag{9}$$

3.5. Pseudo-Code of the EOA

In summary, the optimization process of EOA begins by generating a randomized set of solutions that represent the candidates for election. Each candidate's solution is updated

by simulating the process of electing a president in the election. The iterative process is divided into two phases: party nomination and presidential election, which correspond to the exploration and exploitation phases, respectively. During the party nomination phase, candidates strive to secure party nominations through the SR and ISST behaviors. In the presidential election phase, the nominees from each party increase their polling support through the TB and CS behaviors. Finally, the EOA process terminates when the specified end criteria are met. The flowchart of EOA is illustrated in Figure 3, while the pseudocode of EOA is presented in Algorithm 1.

Algorithm 1: Election Optimizer Algorithm

Input: N, T, UB, Lb, Dim
Output: X_{best}
Initialize the population X and the parameters of EOA
while *not done* **do**
  Calculate the fitness function values and $X_{best}(t)$
  for *each iteration* **do**
    Update the mean value of the current solution $X_M(t)$
    if $t < \frac{1}{2}T$ **then**
      Party nomination:
      if $r_1 < 0.5$ **then**
        Step 1: Strategy reference
        Update the current solution using Equation (3)
        $X_{new}(t+1) = X_1(t+1)$
      else
        Step 2: Innovative suggestions of the staff team
        Update the current solution using Equation (5)
        $X_{new}(t+1) = X_2(t+1)$
      end
      if Fitness$(X_{new}(t+1)) <$ Fitness$(X(t))$ **then**
        $X(t) = X_{new}(t+1)$
      end
    else
      Presidential election:
      if $rand \leq 0.5$ **then**
        Step 3: Televised debate
        Update the current solution using Equation (7)
        $X_{new}(t+1) = X_3(t+1)$
      else
        Step 4: Campaign speech
        Update the current solution using Equation (8)
        $X_{new}(t+1) = X_4(t+1)$
      end
      if $Fitness(X_{new}(t+1)) < Fitness(X(t))$ **then**
        $X(t) = X_{new}(t+1)$
      end
    end
  end
end
return the best solution $X_{best}(t)$

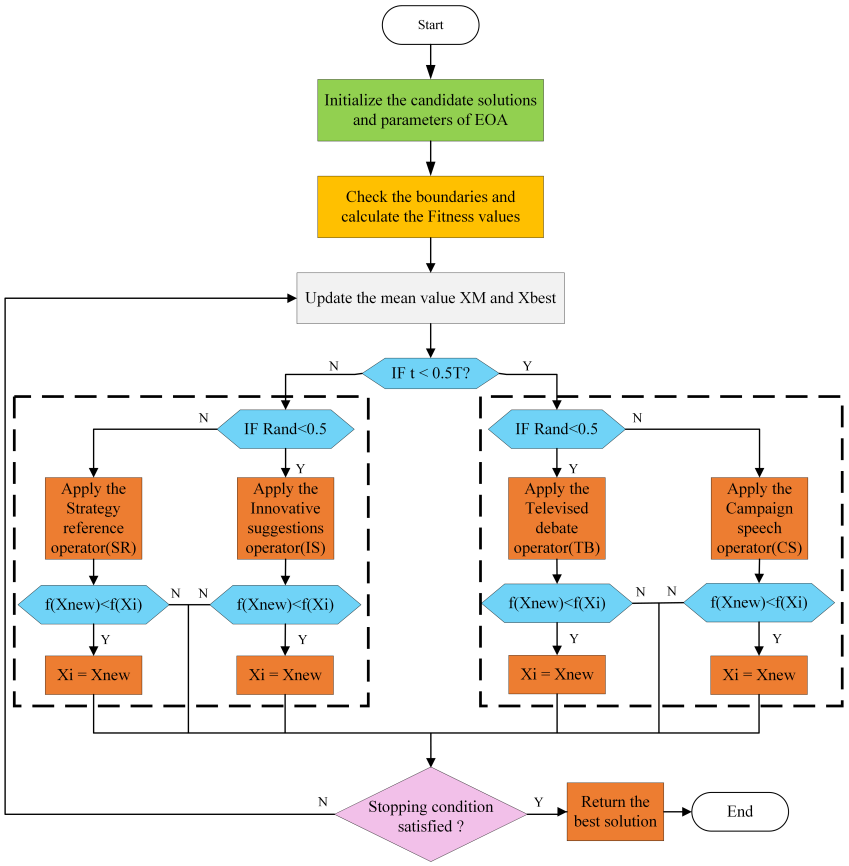


Figure 3. Flowchart of EOA.

3.6. Computational Complexity of EOA

This section focuses on calculating the computational complexity of EOA, which is typically determined by three factors: initialization, fitness calculation, and solution updating. Assuming that there are N solutions, the computational complexity of the initialization process is $\mathcal{O}(N)$. The complexity of fitness calculation is problem-dependent and is not discussed here. Finally, the complexity of updating solutions is $\mathcal{O}(T \cdot N) + \mathcal{O}(T \cdot N \cdot Dim)$, where Dim indicates the dimension of the problem, and T is the number of iterations. Therefore, the overall computational complexity of EOA is $\mathcal{O}(N \cdot (TDim + 1))$.

4. Experimental Results and Discussions

In this section, twenty-three classical functions and ten IEEE CEC2019 test functions are used to evaluate the performance of EOA. To ensure the fairness of comparison, all algorithms have been implemented for the same number of iterations and population size of 500 and 30, respectively. All experiments are carried out using the MATLAB 2019b platform on a computer equipped with an Intel(R) Core(TM) i7-9750H CPU @ 2.60 GHz (from Intel Corporation, Santa Clara, CA, USA) and 16 GB of RAM.

4.1. Experiment 1: Twenty-Three Benchmark Test Functions

The twenty-three benchmark functions are categorized into three main types: unimodal functions (F1–F7, Table 1), multimodal functions (F8–F13, Table 2), and fixed-dimension multimodal functions (F14–F23, Table 3). Initially, we investigate the impact of different parameter values of EOA on its performance using the F14–F23 functions. Subsequently, the F1–F7 functions are employed to evaluate the exploitation capability of EOA, while the F8–F23 functions are utilized to assess its exploration capability. Furthermore, experiments are conducted on the F1–F13 functions under dimensions of 10, 50, 100, and 500 to evaluate the performance of EOA. Finally, the computational time

consumption of EOA is evaluated. All experiments are compared against the following nine well-known algorithms:

- Aquila Optimizer (AO) [16];
- Particle Swarm Optimization (PSO) [10];
- Sine Cosine Algorithm (SCA) [33];
- Whale Optimization Algorithm (WOA) [13];
- Grey Wolf Optimizer (GWO) [14];
- Honey Badger Algorithm (HBA) [11];
- Sparrow Search Algorithm (SSA) [34];
- Social Network Search (SNS) [25];
- Bald Eagle Search (BES) [35].

Table 1. Unimodal benchmark functions.

Fun	Dimensions	Range	Fmin
$f_1(x) = \sum_{i=1}^{n} x_i^2$	10, 50 , 100, 500	$[-100, 100]$	0
$f_2(x) = \sum_{i=0}^{n} \lvert x_i \rvert + \prod_{i=0}^{n} \lvert x_i \rvert$	10, 50, 100, 500	$[-10, 10]$	0
$f_3(x) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j \right)^2$	10, 50, 100, 500	$[-100, 100]$	0
$f_4(x) = \max_i \{ \lvert x_i \rvert, 1 \leqslant i \ll n \}$	10, 50, 100, 500	$[-100, 100]$	0
$f_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_i^2 - x_{i+1} \right)^2 + (1 - x_i)^2 \right]$	10, 50, 100, 500	$[-30, 30]$	0
$f_6(x) = \sum_{i=1}^{n} \left([x_i + 0.5] \right)^2$	10, 50, 100, 500	$[-100, 100]$	0
$f_7(x) = \sum_{i=0}^{n} i x_i^4 + random[0, 1)$	10, 50, 100, 500	$[-128, 128]$	0

Table 2. Multimodal benchmark functions.

Fun	Dimensions	Range	Fmin
$f_8(x) = \sum_{i=1}^{n} \left(-x_i \sin\left(\sqrt{\lvert x_i \rvert} \right) \right)$	10, 50, 100, 500	$[-500, 500]$	$-418.9829n$
$f_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	10, 50, 100, 500	$[-5.12, 5.12]$	0
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$	10, 50, 100, 500	$[-32, 32]$	0
$f_{11}(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right)$	10, 50, 100, 500	$[-600, 600]$	0
$f_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_1)\} + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1}) + \sum_{i=1}^{n} u(x_i, 10, 100, 4)]$, $where\ y_i = 1 + \frac{x_i + 1}{4}, u(x_i, a, k, m) = \begin{cases} K(x_i - a)^m, & if\ x_i > a \\ 0, -a \leqslant x_i \geqslant a \\ K(-x_i - a)^m, & -a \leqslant x_i \end{cases}$	10, 50, 100, 500	$[-50, 50]$	0
$f_{13}(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_1 + 1)] + (x_n - 1)^2 1 + \sin^2(2\pi x_n)) + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	10, 50, 100, 500	$[-50, 50]$	0

Table 4 presents the main control parameters of the comparison algorithms. The performance of these algorithms is compared based on the mean and standard deviation (STD) of 30 independent runs. Further details can be found at the bottom of each table. The first row displays three symbols (W | L | T) indicating whether EOA's performance is superior (win), inferior (lose), or undifferentiated (tie) compared to other algorithms across all functions. The second row presents the Friedman mean, while the third row displays the final ranking value of the algorithm.

Table 3. Fixed-dimension multimodal benchmark functions.

Fun	Dimensions	Range	Fmin
$f_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})}\right)^{-1}$	2	$[-65, 65]$	1
$f_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	$[-5, 5]$	0.0003
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 5]$	0.398
$f_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_i + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	2	$[-2, 2]$	3
$f_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{i=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	$[-1, 2]$	-3.86
$f_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{i=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6	$[0, 1]$	-0.32
$f_{21}(x) = -\sum_{i=1}^{5} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 1]$	-10.1532
$f_{22}(x) = -\sum_{i=1}^{7} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 1]$	-10.4028
$f_{23}(x) = -\sum_{i=1}^{10} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 1]$	-10.5363

Table 4. Parameters for the comparative algorithms.

Algorithm	Parameters
AO	$U = 0.00565; r_1 = 10; \omega = 0.005; \alpha = 0.1; \delta = 0.1$
PSO	$c_1 = 2; c_2 = 2; v_{\max} = 6$
SCA	-
WOA	$a = [2, 0]; b = 2$
GWO	$a = [2, 0]$
HBA	$C = 2; \beta = 6$
SSA	-
SNS	-
BES	-

4.1.1. The Parameter Sensitivity Analysis of EOA

Firstly, we conducted a preliminary analysis to examine the impact of varying sensitive factor α values on the performance of EOA. Table 5 presents the statistical results obtained by applying different parameter values to the classical test functions (F14–F23).

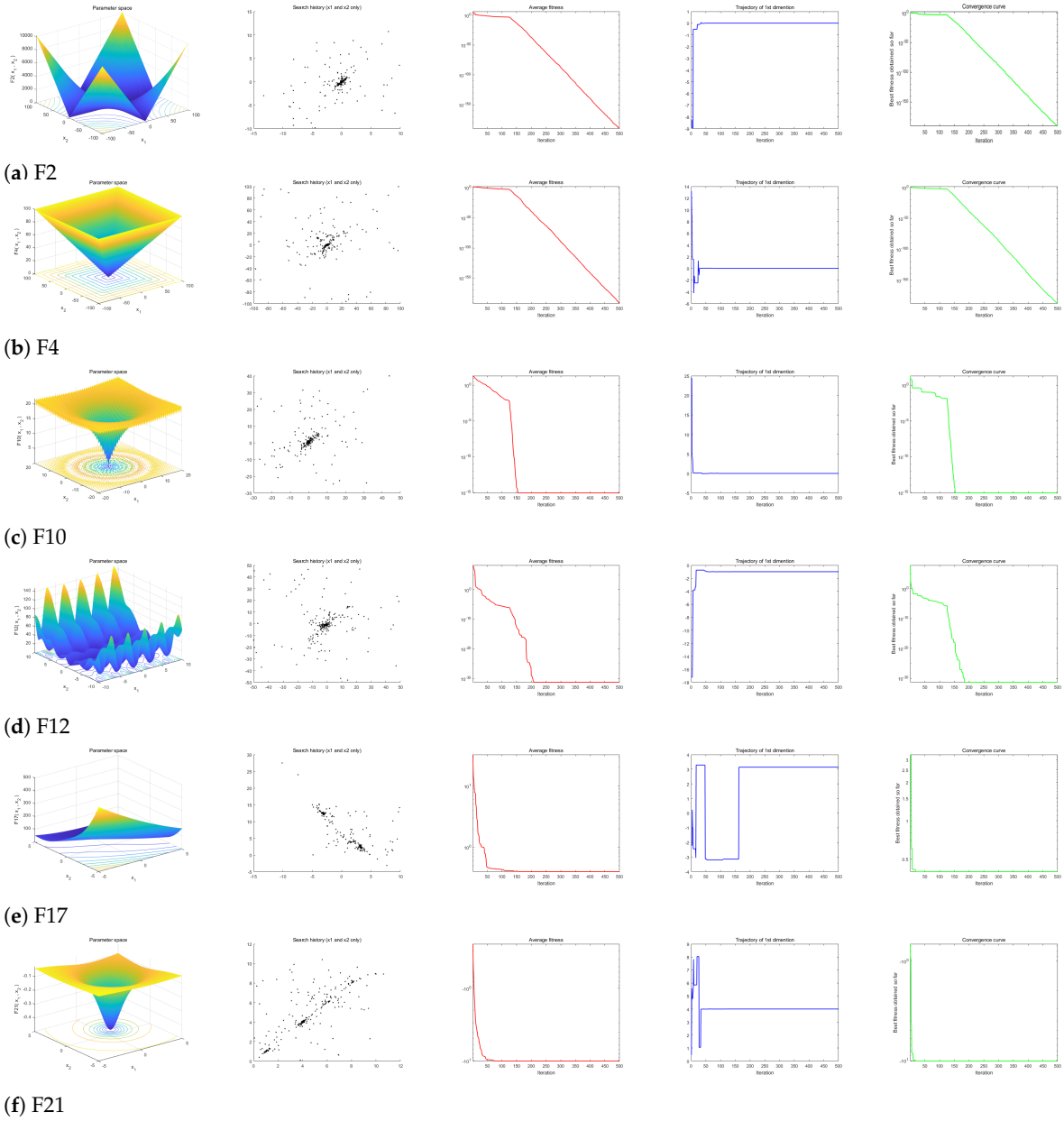
These test functions encompass various dimensions and provide a comprehensive evaluation of EOA's performance across multiple aspects. The experimental findings indicate that the optimal results are achieved when α is set to 2. Moreover, for the ten classical test functions, EOA exhibits similar performance when α is set to 5 or 15. Conversely, the poorest results are observed when α is set to 0.1. As a result, the sensitivity factor of $\alpha = 2$ is chosen in the subsequent experiments.

Table 5. The influence of the EOA parameter (i.e., α) on classical test functions (F14–F23).

Fun	Measure	α = 0.1	α = 2	α = 5	α = 10	α = 15	α = 20
F14	Average	9.98×10^{-1}	9.98×10^{-1}	9.98×10^{-1}	9.98×10^{-1}	9.98×10^{-1}	9.98×10^{-1}
	STD	4.16×10^{-10}	1.70×10^{-16}	1.80×10^{-16}	1.87×10^{-13}	1.54×10^{-16}	1.84×10^{-16}
	Rank	6	2	3	5	1	4
F15	Average	1×10^{-3}	3.41×10^{-4}	4.41×10^{-4}	4.36×10^{-4}	4.78×10^{-4}	4.74×10^{-4}
	STD	4.37×10^{-4}	7.15×10^{-5}	2.77×10^{-4}	1.65×10^{-4}	2.52×10^{-4}	2.58×10^{-4}
	Rank	6	1	5	2	3	4
F16	Average	-1.02×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}
	STD	3.04×10^{-2}	2.82×10^{-5}	6.02×10^{-5}	2.95×10^{-4}	5.74×10^{-4}	6.45×10^{-4}
	Rank	6	1	2	3	4	5
F17	Average	4.33×10^{-1}	3.98×10^{-1}	3.99×10^{-1}	4.00×10^{-1}	4.01×10^{-1}	4.01×10^{-1}
	STD	9.26×10^{-2}	4.37×10^{-4}	4.54×10^{-3}	4.48×10^{-3}	5.30×10^{-3}	6.83×10^{-3}
	Rank	6	1	3	2	4	5
F18	Average	6.76×10^{0}	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	3.02×10^{0}
	STD	8.38×10^{0}	5.87×10^{-10}	4.15×10^{-5}	2.78×10^{-3}	7.21×10^{-3}	2.67×10^{-2}
	Rank	6	1	2	5	3	4
F19	Average	-3.75×10^{0}	-3.86×10^{0}	-3.86×10^{0}	-3.86×10^{0}	-3.86×10^{0}	-3.86×10^{0}
	STD	1.08×10^{-1}	8.62×10^{-8}	3.58×10^{-5}	7.25×10^{-4}	1.70×10^{-3}	3.15×10^{-3}
	Rank	6	1	2	3	4	5
F20	Average	-2.46×10^{0}	-3.27×10^{0}	-3.24×10^{0}	-3.18×10^{0}	-3.16×10^{0}	-3.13×10^{0}
	STD	3.92×10^{-1}	6.91×10^{-2}	7.91×10^{-2}	1.08×10^{-1}	1.10×10^{-1}	1.90×10^{-1}
	Rank	6	1	2	3	4	5
F21	Average	-1.02×10^{1}	-1.02×10^{1}	-1.02×10^{1}	-1.02×10^{1}	-1.02×10^{1}	-1.02×10^{1}
	STD	8.40×10^{-4}	2.88×10^{-7}	1.85×10^{-7}	2.03×10^{-7}	1.45×10^{-7}	1.95×10^{-7}
	Rank	6	5	2	4	1	3
F22	Average	-1.04×10^{1}	-1.04×10^{1}	-1.04×10^{1}	-1.04×10^{1}	-1.04×10^{1}	-1.04×10^{1}
	STD	1.94×10^{-3}	1.84×10^{-5}	1.04×10^{-5}	3.25×10^{-5}	2.49×10^{-5}	1.58×10^{-5}
	Rank	6	2	5	4	3	1
F23	Average	-1.05×10^{1}	-1.05×10^{1}	-1.05×10^{1}	-1.05×10^{1}	-1.05×10^{1}	-1.05×10^{1}
	STD	9.68×10^{-4}	2.24×10^{-5}	3.65×10^{-5}	3.32×10^{-5}	3.52×10^{-5}	2.78×10^{-5}
	Rank	6	1	5	3	4	2
Mean	Rank	6	1.6	3.1	3.4	3.1	3.8
Final	Rank	5	1	2	3	2	4

4.1.2. Qualitative Analysis for Convergence of EOA

Several experiments were conducted to qualitatively analyze the convergence behavior of EOA. The results of these experiments are presented in Figure 4, where the first to fifth columns correspond to the parameter space of the function, search history, average fitness, trajectory, and convergence curve, respectively. The parameter space of the function describes the topology of the search space. The search history depicts the candidates and the behaviors in which candidates run for president. In unimodal functions, the mode represents the movement of the candidate in the vicinity of the optimal point, while in multimodal functions, it represents the scattering characteristics of the candidate. The search history results demonstrate that EOA is capable of rapidly identifying the optimal solution in both unimodal and multimodal functions. Furthermore, the average fitness and convergence curve exhibit large initial values that gradually decrease with each iteration. Notably, the solutions display high amplitude and frequency in the early iterations, which diminishes in the final iterations. This pattern suggests that EOA has a high exploration ability during the early stages and good exploitation ability in the later stages, thereby increasing its likelihood of reaching the optimal solution. Qualitative results indicate that EOA possesses strong development and exploration capabilities throughout the iteration process.



(**a**) F2

(**b**) F4

(**c**) F10

(**d**) F12

(**e**) F17

(**f**) F21

Figure 4. Qualitative results for the studied problems.

Additionally, Figure 5 shows the convergence curve for the EOA and the other compared algorithms on the 23 benchmark functions. For unimodal functions, the convergence rate of EOA is faster than the other algorithms, and its final convergence accuracy is better as well. This confirms that EOA has a strong exploitation capacity and reliability. For multimodal functions, the EOA achieves a high balance between the exploration and exploitation stages. For F1–F7, EOA approaches the positions around the optimal solutions and exploits them efficiently to obtain high-precision final solutions. For F8–F23, the EOA gradually approaches the optimal solutions and updates the positions to confirm the final solution as the iteration progresses. Therefore, the EOA has strong local optimum escape capability and can escape the local optimum after multiple stagnations.

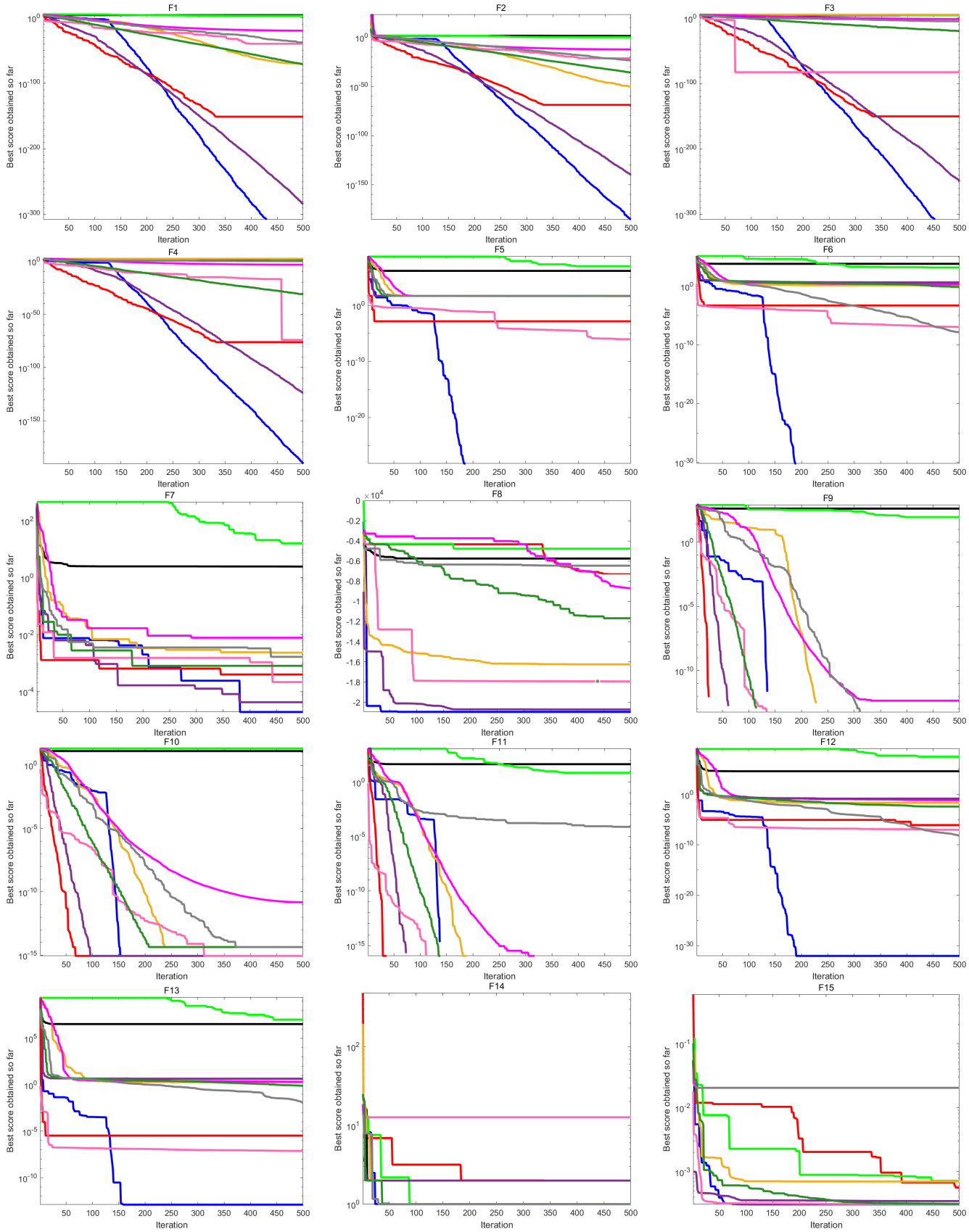


Figure 5. *Cont.*

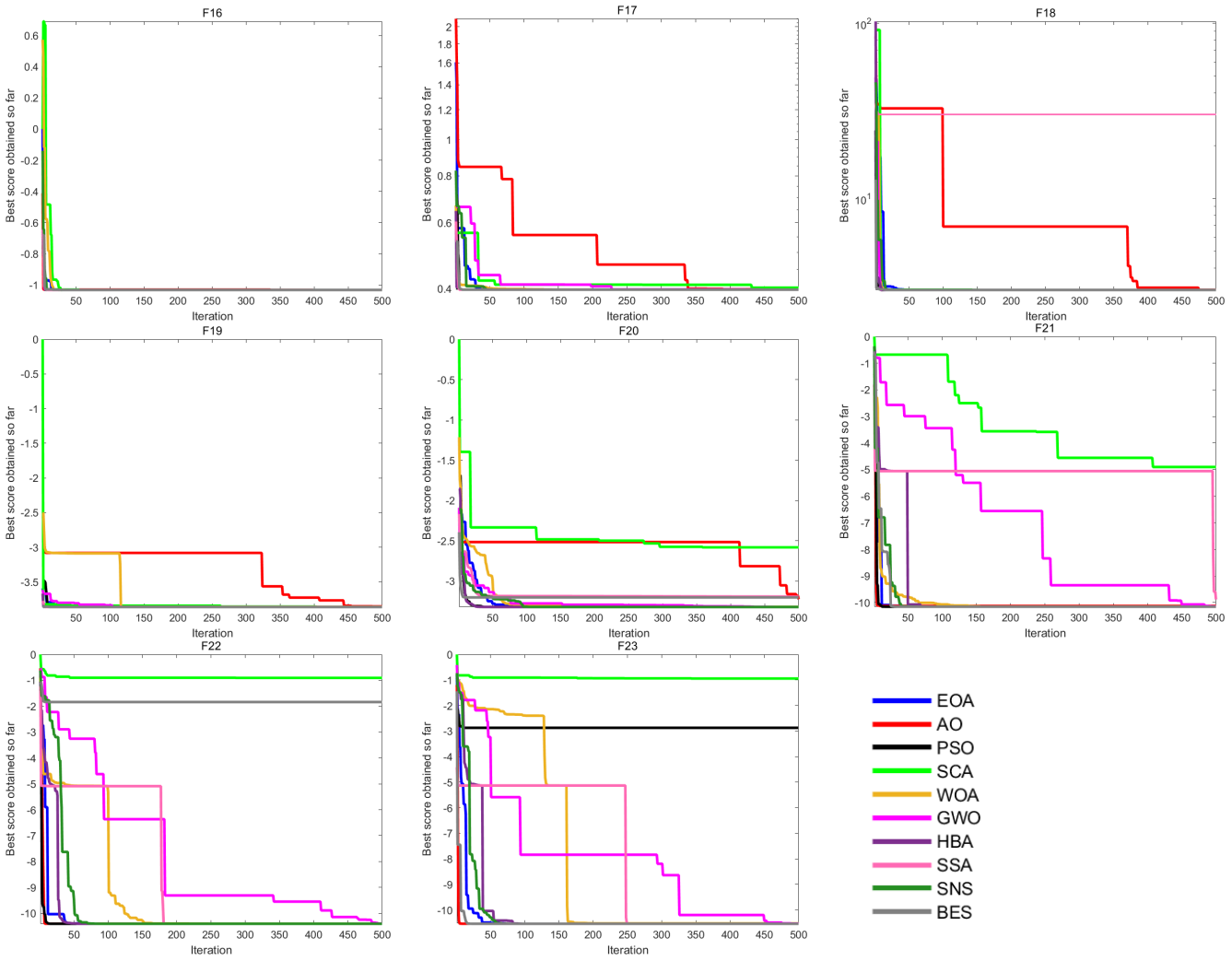


Figure 5. Convergence behavior of the comparative algorithms on classical test functions (F1–F23).

4.1.3. Exploration and Exploitation Evaluation of EOA

Exploration and exploitation are the two core aspects of meta-heuristic optimization algorithms, which help to assess the overall performance of the algorithm. Unimodal functions have only one globally optimal solution and are used to assess the exploitation capability. Multimodal functions have multiple locally optimal solutions and are therefore used to assess the exploration capability. In this section, the exploration and exploitation performance of EOA will be evaluated using these two types of functions and compared against the comparison algorithms, the relevant details of which are given below:

As shown in Table 6, EOA significantly outperforms the other comparison algorithms (PSO, SCA, WOA, GWO) in all test metrics on the unimodal function of dimension 10. On F9 and F11, the performance of EOA is similar to AO, HBA, and SSA. The results of the multimodal functions in Table 7 show that EOA outperforms the other algorithms in most cases when solving multimodal global optimization problems. These are mainly attributed to the stochastic reference coefficient AF in the party nomination stage, which provides more flexibility and reliability and enables it to effectively avoid local optimization. Tables 8–10 further test the unimodal function on dimensions 50, 100, and 500, and, finally, obtain similar experimental results as when the dimension is 10. This proves that the EOA can effectively solve the multi-dimensional unimodal function optimization problem, benefiting from the presidential election stage of EOA, which further filters the party's elite candidates, who have obtained the nomination, and effectively regulates the control factors.

Table 6. Results of the comparative algorithms on classical test functions (F1–F13); the dimension is fixed to 10.

Fun No. Measure	Comparative Algorithms									
	EOA	**AO**	**PSO**	**SCA**	**WOA**	**GWO**	**HBA**	**SSA**	**SNS**	**BES**
F1										
Average	0.0000×10^{0}	1.1533×10^{-104}	1.1392×10^{1}	1.9470×10^{-12}	4.0398×10^{-78}	3.9699×10^{-56}	3.5547×10^{-312}	5.6831×10^{-29}	1.4416×10^{-83}	1.3155×10^{-83}
STD	0.0000×10^{0}	6.1682×10^{-104}	9.4332×10^{1}	7.7160×10^{-12}	1.6210×10^{-77}	1.9941×10^{-55}	0.0000×10^{0}	2.3456×10^{-28}	3.5612×10^{-83}	4.0797×10^{-83}
F2										
Average	5.7485×10^{-167}	9.6411×10^{-57}	2.0966×10^{0}	1.5043×10^{-9}	3.9642×10^{-53}	5.9832×10^{-33}	3.2575×10^{-160}	2.1668×10^{-17}	3.8990×10^{-43}	1.9158×10^{-43}
STD	2.4509×10^{-159}	5.2412×10^{-56}	1.5699×10^{0}	2.9623×10^{-9}	9.4221×10^{-53}	9.5273×10^{-33}	1.7796×10^{-159}	1.0738×10^{-16}	9.9081×10^{-43}	2.4764×10^{-43}
F3										
Average	4.6951×10^{-320}	6.0686×10^{-103}	3.8131×10^{2}	2.8358×10^{-2}	2.1617×10^{2}	1.8913×10^{-24}	1.4818×10^{-287}	8.7829×10^{-25}	3.1956×10^{-43}	4.8278×10^{-44}
STD	0.0000×10^{0}	3.3239×10^{-102}	3.7641×10^{2}	1.2450×10^{-1}	2.5478×10^{2}	8.4211×10^{-24}	0.0000×10^{0}	4.7366×10^{-24}	1.4400×10^{-43}	1.1647×10^{-43}
F4										
Average	9.7840×10^{-169}	1.2838×10^{-54}	1.0636×10^{1}	6.3871×10^{-4}	3.9778×10^{0}	4.7137×10^{-18}	5.2818×10^{-151}	4.7386×10^{-14}	5.2853×10^{-39}	7.0078×10^{-39}
STD	0.0000×10^{0}	6.7668×10^{-54}	4.1526×10^{0}	1.1645×10^{-3}	1.1226×10^{1}	7.2618×10^{-18}	1.8091×10^{-150}	1.6950×10^{-13}	7.0512×10^{-39}	1.0665×10^{-38}
F5										
Average	2.1875×10^{-11}	2.0125×10^{-3}	6.5201×10^{3}	7.4097×10^{0}	6.8185×10^{0}	6.4806×10^{0}	5.2809×10^{0}	1.0456×10^{-6}	7.3818×10^{0}	9.2718×10^{-1}
STD	7.4311×10^{-11}	4.7637×10^{-3}	9.9818×10^{3}	4.0683×10^{-1}	1.2453×10^{0}	5.8892×10^{-1}	6.5485×10^{-1}	2.6620×10^{-6}	4.0359×10^{-1}	1.1280×10^{0}
F6										
Average	1.3423×10^{-11}	8.5519×10^{-5}	1.3352×10^{2}	4.6986×10^{-1}	1.1743×10^{-3}	8.3961×10^{-3}	5.9259×10^{-8}	7.5730×10^{-11}	8.3101×10^{-7}	4.1087×10^{-34}
STD	7.2641×10^{-11}	2.0687×10^{-4}	1.3565×10^{2}	1.3641×10^{-1}	1.1235×10^{-3}	4.5968×10^{-2}	2.4121×10^{-7}	1.9638×10^{-11}	3.8576×10^{-6}	1.7606×10^{-33}
F7										
Average	1.0781×10^{-4}	1.3029×10^{-4}	5.9725×10^{-2}	2.7296×10^{-3}	2.7883×10^{-3}	5.7299×10^{-4}	4.4690×10^{-4}	4.9487×10^{-4}	4.1342×10^{-4}	5.5327×10^{-4}
STD	9.5072×10^{-5}	1.3546×10^{-4}	4.1444×10^{-2}	2.1231×10^{-3}	5.3831×10^{-3}	3.5644×10^{-4}	3.7913×10^{-4}	5.0986×10^{-4}	3.1656×10^{-4}	5.2212×10^{-4}
F8										
Average	-4.1898×10^{3}	-3.3454×10^{3}	-2.5558×10^{3}	-2.1411×10^{3}	-3.2459×10^{3}	-2.6275×10^{3}	-4.0256×10^{3}	-2.6666×10^{3}	-4.1416×10^{3}	-2.8871×10^{3}
STD	2.7751×10^{-12}	8.5102×10^{2}	3.5420×10^{2}	1.2876×10^{2}	6.2299×10^{2}	3.7201×10^{2}	5.1765×10^{2}	5.2729×10^{2}	7.9068×10^{1}	3.2829×10^{2}
F9										
Average	0.0000×10^{0}	0.0000×10^{0}	2.8584×10^{1}	1.1033×10^{0}	8.3123×10^{-1}	4.9397×10^{-1}	0.0000×10^{0}	0.0000×10^{0}	8.2802×10^{-13}	4.1047×10^{0}
STD	0.0000×10^{0}	0.0000×10^{0}	1.0608×10^{1}	5.9009×10^{0}	4.5528×10^{0}	1.3669×10^{0}	0.0000×10^{0}	0.0000×10^{0}	4.5352×10^{-12}	6.1534×10^{0}
F10										
Average	8.8818×10^{-16}	8.8818×10^{-16}	7.1716×10^{0}	2.7198×10^{-7}	4.5593×10^{-15}	7.4015×10^{-15}	8.8818×10^{-16}	9.4147×10^{-15}	8.8818×10^{-16}	4.0856×10^{-15}
STD	0.0000×10^{0}	0.0000×10^{0}	1.8229×10^{0}	3.5323×10^{-7}	3.0208×10^{-15}	2.1035×10^{-15}	0.0000×10^{0}	2.4589×10^{-14}	0.0000×10^{0}	1.0840×10^{-15}
F11										
Average	0.0000×10^{0}	0.0000×10^{0}	2.2826×10^{0}	1.2227×10^{-1}	8.8558×10^{-2}	4.3030×10^{-2}	0.0000×10^{0}	0.0000×10^{0}	4.1070×10^{-4}	9.4813×10^{-2}
STD	0.0000×10^{0}	0.0000×10^{0}	1.4692×10^{0}	2.0183×10^{-1}	1.6438×10^{-1}	1.3956×10^{-1}	0.0000×10^{0}	0.0000×10^{0}	2.2495×10^{-3}	1.2187×10^{-1}
F12										
Average	2.0345×10^{-9}	4.1045×10^{-6}	5.9767×10^{0}	1.0529×10^{-1}	9.1024×10^{-3}	2.6685×10^{-3}	2.2066×10^{-2}	2.2423×10^{-8}	6.7544×10^{-9}	5.1830×10^{-2}
STD	5.8925×10^{-9}	7.4447×10^{-6}	4.8914×10^{0}	3.6317×10^{-2}	1.6703×10^{-2}	6.9166×10^{-3}	8.1391×10^{-2}	3.5304×10^{-8}	1.8283×10^{-8}	1.6503×10^{-1}
F13										
Average	4.2187×10^{-9}	2.7660×10^{-5}	6.2226×10^{1}	3.0794×10^{-1}	4.5687×10^{-2}	2.8888×10^{-2}	1.1409×10^{-1}	9.6459×10^{-8}	3.6644×10^{-4}	5.1221×10^{-3}
STD	1.2566×10^{-8}	3.5506×10^{-5}	2.6656×10^{2}	7.2912×10^{-2}	6.0818×10^{-2}	5.2371×10^{-2}	8.9435×10^{-2}	1.8620×10^{-7}	2.0060×10^{-3}	1.2796×10^{-2}
W|L|T		10|0|3	13|0|0	13|0|0	13|0|0	13|0|0	10|0|3	10|1|2	12|0|1	12|0|1
Mean Rank	1.1538×10^{0}	2.8461×10^{0}	9.9231×10^{0}	8.7692×10^{0}	6.9231×10^{0}	6.6923×10^{0}	3.2307×10^{0}	4.4615×10^{0}	4.2307×10^{0}	5.3846×10^{0}
Final Rank	1	2	10	9	8	7	3	5	4	6

Table 7. Results of the comparative algorithms on classical test functions (F14–F23).

Fun No. Measure	Comparative Algorithms									
	EOA	**AO**	**PSO**	**SCA**	**WOA**	**GWO**	**HBA**	**SSA**	**SNS**	**BES**
F14										
Average	9.9800×10^{-1}	3.1502×10^{0}	3.1309×10^{0}	1.5948×10^{0}	2.6394×10^{0}	5.9632×10^{0}	5.6931×10^{0}	9.7894×10^{0}	9.9800×10^{-1}	9.9800×10^{-1}
STD	1.7494×10^{-16}	3.8581×10^{0}	2.8343×10^{0}	9.2377×10^{-1}	2.5862×10^{0}	4.3870×10^{0}	4.5277×10^{0}	4.8834×10^{0}	5.8312×10^{-14}	1.2150×10^{-11}
F15										
Average	3.5032×10^{-4}	4.8363×10^{-4}	7.1338×10^{-3}	8.9559×10^{-4}	7.4251×10^{-4}	5.8200×10^{-3}	5.9400×10^{-3}	3.5665×10^{-4}	3.6128×10^{-4}	3.2629×10^{-3}
STD	1.6656×10^{-4}	9.0528×10^{-5}	8.7476×10^{-3}	3.5297×10^{-4}	5.8247×10^{-4}	8.9263×10^{-3}	8.8796×10^{-3}	1.6523×10^{-4}	1.7249×10^{-4}	6.8342×10^{-3}
F16										
Average	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}	-1.0316×10^{0}
STD	5.7788×10^{-6}	4.5126×10^{-4}	6.5843×10^{-16}	3.3581×10^{-5}	2.0495×10^{-9}	1.9173×10^{-8}	4.6436×10^{-8}	1.6976×10^{-15}	6.7122×10^{-13}	1.2856×10^{-7}
F17										
Average	3.9791×10^{-1}	3.9812×10^{-1}	3.9789×10^{-1}	3.9993×10^{-1}	3.9790×10^{-1}	3.9789×10^{-1}	3.9789×10^{-1}	3.9789×10^{-1}	3.9789×10^{-1}	3.9789×10^{-1}
STD	2.4840×10^{-5}	3.0975×10^{-4}	0.0000×10^{0}	1.9941×10^{-3}	1.8505×10^{-5}	2.3984×10^{-4}	4.1755×10^{-12}	2.8303×10^{-10}	0.0000×10^{0}	1.3252×10^{-6}
F18										
Average	3.0000×10^{0}	3.0457×10^{0}	3.0000×10^{0}	3.0001×10^{0}	3.0001×10^{0}	3.0000×10^{0}	1.0200×10^{1}	3.0000×10^{0}	3.0000×10^{0}	3.0000×10^{0}
STD	1.1683×10^{-10}	5.8557×10^{-2}	1.1247×10^{-15}	7.0373×10^{-5}	1.3265×10^{-4}	3.4588×10^{-5}	1.7271×10^{1}	3.6350×10^{-15}	1.1367×10^{-15}	2.2020×10^{-15}
F19										
Average	-3.8628×10^{0}	-3.8552×10^{0}	-3.8632×10^{0}	-3.8544×10^{0}	-3.8512×10^{0}	-3.8628×10^{0}	-3.8628×10^{0}	-3.8370×10^{0}	-3.8628×10^{0}	-3.8628×10^{0}
STD	1.6534×10^{-12}	7.4057×10^{-3}	1.9996×10^{-3}	2.5992×10^{-3}	2.3682×10^{-2}	3.0209×10^{-3}	8.5426×10^{-8}	1.4113×10^{-1}	2.6684×10^{-15}	2.7101×10^{-15}
F20										
Average	-3.2854×10^{0}	-3.1626×10^{0}	-3.2342×10^{0}	-2.8637×10^{0}	-3.2405×10^{0}	-3.2574×10^{0}	-3.2705×10^{0}	-3.2236×10^{0}	-3.2861×10^{0}	3.2586×10^{0}
STD	6.4936×10^{-2}	8.9475×10^{-2}	7.1977×10^{-2}	3.9953×10^{-1}	9.1794×10^{-2}	8.0283×10^{-2}	5.9923×10^{-2}	9.6034×10^{-2}	5.5415×10^{-2}	6.0328×10^{-2}
F21										
Average	-1.0153×10^{1}	-1.0143×10^{1}	-6.6515×10^{0}	-2.6894×10^{0}	-8.7614×10^{0}	-9.4778×10^{0}	-9.1501×10^{0}	-9.4208×10^{0}	-9.9833×10^{0}	-9.1372×10^{0}
STD	1.7358×10^{-7}	1.3443×10^{-2}	3.6279×10^{0}	1.8340×10^{0}	2.2756×10^{0}	1.7470×10^{0}	2.6009×10^{0}	1.7652×10^{0}	9.3076×10^{-1}	2.3462×10^{0}
F22										
Average	-1.0403×10^{1}	-1.0393×10^{1}	-5.7534×10^{0}	-3.1574×10^{0}	-6.9636×10^{0}	-1.0146×10^{1}	-9.6402×10^{0}	-9.6662×10^{0}	-9.8260×10^{0}	-8.4611×10^{0}
STD	2.7246×10^{-7}	1.1445×10^{-2}	3.6563×10^{0}	1.8034×10^{0}	3.1176×10^{0}	1.3940×10^{0}	2.3404×10^{0}	1.8330×10^{0}	1.7726×10^{0}	3.2817×10^{0}
F23										
Average	-1.0536×10^{1}	-1.0522×10^{1}	-7.1053×10^{0}	-4.1089×10^{0}	-7.4850×10^{0}	-1.0293×10^{1}	-8.9545×10^{0}	-9.8878×10^{0}	-1.0132×10^{1}	-9.0547×10^{0}
STD	8.5162×10^{-7}	2.4584×10^{-2}	3.7756×10^{0}	1.5194×10^{0}	3.3958×10^{0}	1.3199×10^{0}	3.2288×10^{0}	1.7070×10^{0}	1.5454×10^{0}	3.0303×10^{0}
W|L|T		9|0|1	8|0|2	9|0|1	9|0|1	7|0|3	8|0|2	8|0|2	5|1|4	6|0|4
Mean Rank	1.1000×10^{0}	5.1000×10^{0}	6.0000×10^{0}	7.5000×10^{0}	5.8000×10^{0}	3.7000×10^{0}	5.7000×10^{0}	4.9000×10^{0}	2.2000×10^{0}	4.4000×10^{0}
Final Rank	1	6	9	10	8	3	7	5	2	4

Table 8. Results of the comparative algorithms on classical test functions (F1–F13); the dimension is fixed to 50.

Fun No. Measure	Comparative Algorithms									
	EOA	**AO**	**PSO**	**SCA**	**WOA**	**GWO**	**HBA**	**SSA**	**SNS**	**BES**
F1										
Average	1.4822×10^{-323}	9.0906×10^{-106}	6.1034×10^{3}	9.2672×10^{2}	1.5100×10^{-74}	1.2943×10^{-19}	5.8825×10^{-261}	3.0820×10^{-30}	9.8001×10^{-70}	1.1520×10^{-31}
STD	0.0000×10^{0}	4.9791×10^{-105}	1.5981×10^{3}	1.2821×10^{3}	3.5563×10^{-74}	1.8411×10^{-19}	0.0000×10^{0}	1.6865×10^{-29}	1.7934×10^{-69}	6.0095×10^{-31}
F2										
Average	4.3112×10^{-161}	4.6638×10^{-50}	4.9594×10^{1}	7.2155×10^{-1}	1.7873×10^{-50}	2.3815×10^{-12}	3.2588×10^{-139}	4.9445×10^{-19}	2.9212×10^{-36}	4.0220×10^{-23}
STD	1.5473×10^{-160}	2.5545×10^{-49}	1.0662×10^{1}	9.8103×10^{-1}	6.4021×10^{-50}	1.3421×10^{-12}	1.7479×10^{-138}	2.2251×10^{-18}	3.7035×10^{-36}	7.9305×10^{-23}
F3										
Average	7.1382×10^{-288}	2.2220×10^{-102}	1.9219×10^{4}	4.8881×10^{4}	1.9279×10^{5}	2.5573×10^{-1}	4.6693×10^{-238}	1.8691×10^{-24}	2.7912×10^{-19}	3.1045×10^{-2}
STD	0.0000×10^{0}	1.2170×10^{-101}	9.4500×10^{3}	1.9570×10^{4}	5.2258×10^{4}	3.5917×10^{-1}	0.0000×10^{0}	1.0232×10^{-23}	8.2795×10^{-19}	1.3629×10^{-1}
F4										
Average	4.3611×10^{-166}	5.0262×10^{-58}	2.6473×10^{1}	7.0185×10^{1}	6.3534×10^{1}	3.8917×10^{-4}	1.2279×10^{-125}	2.5437×10^{-16}	1.2060×10^{-31}	5.7561×10^{-1}
STD	0.0000×10^{0}	2.7530×10^{-57}	4.5059×10^{0}	6.2980×10^{0}	2.9083×10^{1}	3.9622×10^{-4}	6.7246×10^{-125}	1.3438×10^{-15}	1.0834×10^{-31}	2.8488×10^{-1}
F5										
Average	5.7287×10^{-8}	1.2994×10^{-2}	1.8450×10^{6}	6.3368×10^{6}	4.8187×10^{1}	4.7443×10^{1}	4.7545×10^{1}	2.9119×10^{-5}	4.7962×10^{1}	4.4470×10^{1}
STD	3.0658×10^{-7}	2.5341×10^{-2}	9.1430×10^{5}	6.3167×10^{5}	4.4626×10^{-1}	8.8040×10^{-1}	9.1984×10^{-1}	5.0827×10^{-5}	2.9966×10^{-1}	6.9957×10^{-1}
F6										
Average	3.2879×10^{-12}	9.5046×10^{-5}	5.9567×10^{3}	9.2932×10^{2}	1.3683×10^{0}	2.7157×10^{0}	3.7948×10^{0}	3.0357×10^{-7}	1.1565×10^{0}	1.7984×10^{-7}
STD	1.8008×10^{-11}	1.6032×10^{-4}	1.7419×10^{3}	1.0251×10^{3}	4.8648×10^{-1}	5.5837×10^{-1}	8.8555×10^{-1}	4.7941×10^{-7}	4.6361×10^{-1}	3.6741×10^{-7}
F7										
Average	1.5484×10^{-4}	1.1181×10^{-4}	4.2520×10^{0}	3.0551×10^{0}	2.6522×10^{-3}	2.8556×10^{-3}	3.8706×10^{-4}	4.0252×10^{-4}	5.5709×10^{-4}	2.6192×10^{-3}
STD	1.2637×10^{-4}	1.0889×10^{-4}	2.0897×10^{0}	3.9337×10^{0}	3.9819×10^{-3}	1.3493×10^{-3}	5.1336×10^{-4}	3.3771×10^{-4}	3.4100×10^{-4}	2.2192×10^{-3}
F8										
Average	-2.0949×10^{4}	-8.1241×10^{3}	-7.3046×10^{3}	-4.7242×10^{3}	-1.8037×10^{4}	-8.8450×10^{3}	-2.0770×10^{4}	-1.8555×10^{4}	-1.1462×10^{4}	-7.0156×10^{3}
STD	3.7002×10^{-12}	5.1387×10^{3}	9.0850×10^{2}	3.2939×10^{2}	2.7581×10^{3}	1.9535×10^{3}	2.7562×10^{2}	1.8144×10^{3}	9.0469×10^{2}	1.7343×10^{3}
F9										
Average	0.0000×10^{0}	0.0000×10^{0}	3.4992×10^{2}	1.0762×10^{2}	0.0000×10^{0}	3.6886×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	3.4620×10^{1}
STD	0.0000×10^{0}	0.0000×10^{0}	3.6073×10^{1}	5.4411×10^{1}	0.0000×10^{0}	3.1416×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	9.1441×10^{1}
F10										
Average	8.8818×10^{-16}	8.8818×10^{-16}	1.2857×10^{1}	1.5827×10^{1}	3.8488×10^{-15}	3.5618×10^{-11}	6.6556×10^{-1}	8.5857×10^{-15}	4.3225×10^{-15}	1.3410×10^{-1}
STD	0.0000×10^{0}	0.0000×10^{0}	1.1853×10^{0}	6.9001×10^{0}	2.6526×10^{-15}	1.4568×10^{-11}	3.6454×10^{0}	3.7599×10^{-14}	6.4863×10^{-16}	4.2408×10^{-1}
F11										
Average	0.0000×10^{0}	0.0000×10^{0}	6.2256×10^{1}	1.1223×10^{1}	1.5907×10^{-2}	4.4474×10^{-3}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	6.9897×10^{-5}
STD	0.0000×10^{0}	0.0000×10^{0}	1.2718×10^{1}	1.7808×10^{1}	6.0792×10^{-2}	7.7996×10^{-3}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	3.4986×10^{-4}
F12										
Average	2.3693×10^{-11}	2.1369×10^{-6}	1.2920×10^{4}	1.1884×10^{7}	3.7176×10^{-2}	1.1158×10^{-1}	1.2212×10^{-1}	3.5536×10^{-8}	4.1769×10^{-3}	6.2213×10^{-3}
STD	1.2161×10^{-10}	3.2573×10^{-6}	3.2688×10^{4}	1.3991×10^{7}	2.3827×10^{-2}	8.6899×10^{-2}	6.7479×10^{-2}	1.2075×10^{-7}	1.9560×10^{-3}	1.8979×10^{-2}
F13										
Average	4.5120×10^{-10}	2.2072×10^{-5}	1.0466×10^{6}	1.3319×10^{7}	1.2155×10^{0}	2.1698×10^{0}	4.1832×10^{0}	2.2710×10^{-7}	2.8202×10^{-1}	1.4115×10^{-1}
STD	1.7750×10^{-9}	3.2179×10^{-5}	9.5909×10^{5}	1.3555×10^{7}	4.9467×10^{-1}	4.1950×10^{-1}	4.0524×10^{-1}	3.3006×10^{-7}	1.2227×10^{-1}	1.8676×10^{-1}
W|L|T		9|1|3	13|0|0	13|0|0	12|0|1	13|0|0	11|0|2	11|0|2	11|0|2	13|0|0
Mean Rank	1.0769×10^{0}	2.8461×10^{0}	9.2307×10^{0}	9.4615×10^{0}	5.7692×10^{0}	7.0000×10^{0}	4.0000×10^{0}	3.5384×10^{0}	4.2307×10^{0}	5.9230×10^{0}
Final Rank	1	2	9	10	6	8	4	3	5	7

Table 9. Results of the comparative algorithms on classical test functions (F1–F13); the dimension is fixed to 100.

Fun No. Measure	Comparative Algorithms									
	EOA	**AO**	**PSO**	**SCA**	**WOA**	**GWO**	**HBA**	**SSA**	**SNS**	**BES**
F1										
Average	0.0000×10^{0}	4.0029×10^{-103}	1.6345×10^{4}	1.4079×10^{4}	4.5370×10^{-69}	1.6596×10^{-12}	1.1187×10^{-205}	1.2068×10^{-30}	5.8683×10^{-67}	8.2747×10^{-27}
STD	0.0000×10^{0}	2.1833×10^{-102}	3.7598×10^{3}	9.2173×10^{3}	2.4806×10^{-68}	1.3188×10^{-12}	0.0000×10^{0}	6.5834×10^{-30}	2.2150×10^{-66}	2.9283×10^{-26}
F2										
Average	4.4575×10^{-179}	3.9562×10^{-62}	1.1433×10^{2}	8.9434×10^{0}	1.3137×10^{-50}	4.3921×10^{-8}	1.5550×10^{-101}	2.8551×10^{-17}	2.4448×10^{-35}	1.5440×10^{-20}
STD	0.0000×10^{0}	2.1669×10^{-61}	2.2027×10^{1}	7.6503×10^{0}	3.7053×10^{-50}	1.5525×10^{-8}	8.5168×10^{-101}	1.5386×10^{-16}	2.3098×10^{-35}	6.8796×10^{-20}
F3										
Average	0.0000×10^{0}	7.1007×10^{-98}	7.7511×10^{4}	2.4156×10^{5}	1.0452×10^{6}	5.5610×10^{2}	6.5896×10^{-180}	6.8440×10^{-29}	1.1490×10^{-13}	5.6474×10^{-1}
STD	0.0000×10^{0}	3.6289×10^{-97}	2.4621×10^{4}	5.9337×10^{4}	3.1330×10^{5}	5.6977×10^{2}	0.0000×10^{0}	2.0530×10^{-28}	4.5494×10^{-13}	7.9921×10^{-1}
F4										
Average	6.5445×10^{-188}	7.1813×10^{-52}	3.0709×10^{1}	8.9072×10^{1}	8.1691×10^{1}	7.3059×10^{-1}	2.2174×10^{-101}	1.8440×10^{-16}	2.1152×10^{-30}	6.6873×10^{-1}
STD	0.0000×10^{0}	3.9333×10^{-51}	3.6557×10^{0}	3.2725×10^{0}	1.5473×10^{1}	6.5283×10^{-1}	1.2142×10^{-100}	6.4903×10^{-16}	3.0160×10^{-30}	3.9586×10^{-1}
F5										
Average	1.3070×10^{-8}	1.8336×10^{-2}	5.7684×10^{6}	1.0114×10^{8}	9.8220×10^{1}	9.7810×10^{1}	9.8234×10^{1}	5.1900×10^{-4}	9.8069×10^{1}	9.6142×10^{1}
STD	4.0499×10^{-8}	2.8904×10^{-2}	2.4303×10^{6}	5.3817×10^{7}	2.0138×10^{-1}	7.1535×10^{-1}	5.1158×10^{-1}	1.4436×10^{-3}	3.2945×10^{-1}	6.2463×10^{-1}
F6										
Average	2.7170×10^{-9}	4.1435×10^{-4}	1.6947×10^{5}	1.5260×10^{5}	4.4421×10^{0}	1.0075×10^{1}	1.3259×10^{1}	1.0928×10^{-6}	7.5049×10^{0}	2.7399×10^{-1}
STD	1.2497×10^{-8}	7.5504×10^{-4}	3.2440×10^{3}	9.4995×10^{3}	1.2959×10^{0}	1.1010×10^{0}	1.6297×10^{0}	2.6478×10^{-6}	8.4381×10^{-1}	2.1308×10^{-1}
F7										
Average	2.0726×10^{-4}	3.0684×10^{-4}	2.4187×10^{1}	1.7268×10^{2}	4.0517×10^{-3}	7.0693×10^{-3}	3.8552×10^{-4}	3.1925×10^{-4}	6.1021×10^{-4}	3.1469×10^{-3}
STD	2.8478×10^{-4}	1.1522×10^{-4}	1.0245×10^{1}	7.5362×10^{1}	6.7313×10^{-3}	2.6871×10^{-3}	4.4644×10^{-4}	2.2516×10^{-4}	3.5386×10^{-4}	2.5790×10^{-3}
F8										
Average	-4.1898×10^{4}	-1.0651×10^{4}	-1.0656×10^{4}	-6.8232×10^{3}	-3.4382×10^{4}	-1.6772×10^{4}	-4.0638×10^{4}	-3.5984×10^{4}	-1.8761×10^{4}	-1.0391×10^{4}
STD	7.4003×10^{-12}	2.7049×10^{3}	1.4484×10^{3}	4.7495×10^{2}	5.7776×10^{3}	2.1652×10^{3}	2.5206×10^{3}	6.2480×10^{3}	3.2023×10^{3}	3.2092×10^{3}
F9										
Average	0.0000×10^{0}	0.0000×10^{0}	8.3431×10^{2}	2.5606×10^{2}	0.0000×10^{0}	8.8446×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	4.6131×10^{1}
STD	0.0000×10^{0}	0.0000×10^{0}	5.6033×10^{1}	1.0143×10^{2}	0.0000×10^{0}	6.5023×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	1.7540×10^{2}
F10										
Average	8.8818×10^{-16}	8.8818×10^{-16}	1.3735×10^{1}	2.0089×10^{1}	4.2040×10^{-15}	1.2828×10^{-7}	8.8818×10^{-16}	5.2699×10^{-15}	4.4409×10^{-15}	1.0112×10^{-1}
STD	0.0000×10^{0}	0.0000×10^{0}	1.4630×10^{0}	2.5688×10^{0}	2.0723×10^{-15}	5.6815×10^{-8}	0.0000×10^{0}	2.3337×10^{-14}	0.0000×10^{0}	3.5835×10^{-1}
F11										
Average	0.0000×10^{0}	0.0000×10^{0}	1.5147×10^{2}	9.6103×10^{1}	1.9673×10^{-2}	3.7803×10^{-3}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}
STD	0.0000×10^{0}	0.0000×10^{0}	3.2492×10^{1}	5.5808×10^{1}	7.5018×10^{-2}	9.9841×10^{-3}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}
F12										
Average	2.1641×10^{-13}	1.1293×10^{-6}	3.0333×10^{5}	3.1645×10^{8}	4.7689×10^{-2}	3.1307×10^{-1}	3.1627×10^{-1}	3.3212×10^{-8}	4.5410×10^{-2}	3.2880×10^{-2}
STD	1.0227×10^{-12}	1.2994×10^{-6}	4.4216×10^{5}	1.5897×10^{8}	1.8146×10^{-2}	7.8459×10^{-2}	1.0157×10^{-1}	6.8878×10^{-8}	1.0533×10^{-2}	1.6346×10^{-1}
F13										
Average	8.8580×10^{-10}	1.6830×10^{-5}	5.3445×10^{6}	5.4656×10^{8}	2.9100×10^{0}	6.6822×10^{0}	9.4644×10^{0}	1.2370×10^{-6}	4.8412×10^{0}	3.6917×10^{0}
STD	3.3831×10^{-9}	2.5911×10^{-5}	3.1394×10^{6}	3.1606×10^{8}	9.9665×10^{-1}	6.2134×10^{-1}	3.0280×10^{-1}	2.7566×10^{-6}	1.0811×10^{0}	9.7761×10^{-1}
W|L|T		10|0|3	13|0|0	13|0|0	12|0|1	13|0|0	10|0|3	11|0|2	11|0|2	12|0|1
Mean Rank	1.0000×10^{0}	2.8461×10^{0}	9.0000×10^{0}	9.6153×10^{0}	5.6153×10^{0}	7.0000×10^{0}	3.7692×10^{0}	3.3846×10^{0}	4.5384×10^{0}	5.6923×10^{0}
Final Rank	1	2	9	10	6	8	4	3	5	7

Table 10. Results of the comparative algorithms on classical test functions (F1–F13; the dimension is fixed to 500.

Fun No. Measure	Comparative Algorithms									
	EOA	**AO**	**PSO**	**SCA**	**WOA**	**GWO**	**HBA**	**SSA**	**SNS**	**BES**
F1										
Average	0.0000×10^{0}	9.9737×10^{-103}	8.9842×10^{5}	2.0913×10^{6}	2.3139×10^{-67}	1.6451×10^{-3}	5.4499×10^{-120}	7.0179×10^{-34}	3.7880×10^{-65}	4.5159×10^{-18}
STD	0.0000×10^{0}	5.4628×10^{-102}	1.1450×10^{4}	7.3718×10^{4}	1.2431×10^{-66}	6.1988×10^{-4}	2.9851×10^{-119}	3.8220×10^{-33}	5.7597×10^{-65}	2.3128×10^{-17}
F2										
Average	2.0683×10^{-178}	6.6208×10^{-58}	6.0664×10^{2}	9.5981×10^{1}	1.1698×10^{-48}	1.1211×10^{-2}	1.9076×10^{-79}	3.3064×10^{-19}	1.4023×10^{-33}	3.4771×10^{-17}
STD	0.0000×10^{0}	3.6264×10^{-57}	7.4849×10^{1}	4.5975×10^{1}	3.2010×10^{-48}	1.7375×10^{-3}	7.2597×10^{-79}	1.7911×10^{-18}	1.0193×10^{-33}	4.5700×10^{-17}
F3										
Average	1.0079×10^{-320}	2.5500×10^{-97}	2.0125×10^{6}	6.8571×10^{6}	3.1479×10^{7}	3.3106×10^{5}	6.7872×10^{-114}	1.1341×10^{-30}	1.5434×10^{-8}	3.7061×10^{1}
STD	0.0000×10^{0}	1.2342×10^{-96}	8.2112×10^{5}	1.4497×10^{6}	1.0174×10^{7}	8.9074×10^{4}	3.7175×10^{-113}	6.2116×10^{-30}	4.9467×10^{-8}	3.6087×10^{1}
F4										
Average	1.9379×10^{-189}	1.0819×10^{-53}	4.1305×10^{1}	9.9091×10^{1}	8.0649×10^{1}	6.3834×10^{1}	1.4333×10^{-58}	9.4338×10^{-17}	1.4785×10^{-28}	8.1384×10^{-1}
STD	0.0000×10^{0}	5.8415×10^{-53}	3.2120×10^{0}	3.2245×10^{-1}	2.1819×10^{1}	5.2083×10^{0}	5.4696×10^{-58}	5.1615×10^{-16}	1.3175×10^{-28}	4.8306×10^{-1}
F5										
Average	6.3303×10^{-7}	1.0933×10^{-1}	3.6932×10^{7}	1.9230×10^{9}	4.9615×10^{2}	4.9810×10^{2}	4.9832×10^{2}	4.6623×10^{-3}	4.9801×10^{2}	5.0008×10^{2}
STD	3.4672×10^{-6}	2.0082×10^{-1}	1.0165×10^{7}	5.2059×10^{8}	3.4532×10^{-1}	5.2390×10^{-1}	2.5199×10^{-1}	1.4692×10^{-2}	1.0777×10^{-1}	1.6650×10^{1}
F6										
Average	7.4937×10^{-12}	8.6370×10^{-4}	9.4331×10^{4}	2.0149×10^{5}	3.2597×10^{1}	9.1699×10^{1}	1.0805×10^{2}	2.9849×10^{-5}	9.8335×10^{1}	5.7510×10^{1}
STD	3.1014×10^{-11}	1.4888×10^{-3}	1.3924×10^{4}	7.0713×10^{4}	8.7655×10^{0}	2.2078×10^{0}	4.5238×10^{0}	1.3696×10^{-4}	1.9667×10^{0}	3.4930×10^{0}
F7										
Average	1.7138×10^{-4}	9.2675×10^{-5}	8.4313×10^{2}	1.5005×10^{4}	3.8356×10^{-3}	5.0364×10^{-2}	6.9244×10^{-4}	3.7404×10^{-4}	6.1745×10^{-4}	3.2118×10^{-3}
STD	1.2709×10^{-4}	8.4577×10^{-5}	3.1069×10^{2}	2.6098×10^{3}	4.4600×10^{-3}	1.1971×10^{-2}	1.0003×10^{-3}	3.4538×10^{-4}	4.0468×10^{-4}	3.0253×10^{-3}
F8										
Average	-2.0949×10^{5}	-3.8466×10^{4}	-2.5106×10^{4}	-1.5125×10^{4}	-1.6746×10^{5}	-5.6390×10^{4}	-1.9600×10^{5}	-1.7714×10^{5}	-6.8100×10^{4}	-2.4742×10^{4}
STD	8.8804×10^{-11}	9.9789×10^{3}	2.5637×10^{3}	1.1739×10^{3}	3.1949×10^{4}	9.2617×10^{3}	2.1626×10^{4}	3.7390×10^{4}	1.1335×10^{4}	1.0765×10^{4}
F9										
Average	0.0000×10^{0}	0.0000×10^{0}	5.0287×10^{3}	1.2286×10^{3}	0.0000×10^{0}	6.5433×10^{1}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}
STD	0.0000×10^{0}	0.0000×10^{0}	1.7426×10^{2}	5.7668×10^{2}	0.0000×10^{0}	1.7021×10^{1}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}
F10										
Average	8.8818×10^{-16}	8.8818×10^{-16}	1.4442×10^{1}	2.0182×10^{1}	4.3225×10^{-15}	1.8316×10^{-3}	8.8818×10^{-16}	8.8818×10^{-16}	4.4409×10^{-15}	2.3365×10^{-2}
STD	0.0000×10^{0}	0.0000×10^{0}	2.6537×10^{0}	2.2691×10^{0}	2.5523×10^{-15}	2.8332×10^{-4}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	9.2834×10^{-2}
F11										
Average	0.0000×10^{0}	0.0000×10^{0}	8.5233×10^{2}	1.7330×10^{3}	0.0000×10^{0}	2.0784×10^{-2}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}
STD	0.0000×10^{0}	0.0000×10^{0}	1.2425×10^{2}	7.0905×10^{2}	0.0000×10^{0}	3.5608×10^{-2}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}	0.0000×10^{0}
F12										
Average	4.0536×10^{-28}	6.6002×10^{-7}	4.4407×10^{6}	5.8968×10^{9}	8.7271×10^{-2}	7.5684×10^{-1}	8.8390×10^{-1}	7.4980×10^{-9}	6.0864×10^{-1}	7.9732×10^{-1}
STD	2.2202×10^{-27}	1.1542×10^{-6}	2.6724×10^{6}	1.0678×10^{9}	3.6072×10^{-2}	5.5654×10^{-2}	7.8785×10^{-2}	1.7203×10^{-8}	3.2435×10^{-2}	4.2288×10^{-1}
F13										
Average	7.6645×10^{-9}	4.1604×10^{-4}	4.6253×10^{7}	9.8283×10^{9}	1.8706×10^{1}	5.1046×10^{1}	4.9520×10^{1}	2.4322×10^{-6}	4.7515×10^{1}	5.1708×10^{1}
STD	4.1156×10^{-8}	9.0268×10^{-4}	2.0882×10^{7}	1.5833×10^{9}	6.0539×10^{0}	1.6554×10^{0}	6.0628×10^{-1}	6.8656×10^{-6}	5.9600×10^{-1}	5.1483×10^{0}
W|L|T		9|1|3	13|0|0	13|0|0	11|0|2	13|0|0	10|0|3	10|0|3	11|0|2	11|0|2
Mean Rank	1.0769×10^{0}	2.6923×10^{0}	8.8461×10^{0}	9.7692×10^{0}	4.6923×10^{0}	7.1538×10^{0}	3.6153×10^{0}	3.0000×10^{0}	4.4615×10^{0}	6.0769×10^{0}
Final Rank	1	2	9	10	6	8	4	3	5	7

Simple analyses of the convergence curves and the final results fail to provide a comprehensive understanding of the algorithm's search process. As a result, we measure the exploration and exploitation percentages of EOA when solving optimization problems on three distinct functions: F2, F11, and F18. The experimental results are shown in Figure 6, which shows that in solving the unimodal problem, the EOA first performs the exploration behavior at a higher percentage and then converges towards the global optimum by increasing the exploitation percentage. In solving F11, the EOA maintains a high exploitation rate initially and gradually increases its exploratory behavior as iterations progress, facilitating the escape from local optima. When solving F18, EOA commences with high exploration and low exploitation rates, subsequently transitioning to high exploitation rates. This adaptive adjustment of exploration and exploitation ratios during the search process enhances convergence accuracy. Furthermore, EOA demonstrates the ability to achieve a relative balance between exploration and exploitation after 100 iterations across all three test functions. This observation directly confirms the algorithm's superior adaptability and faster convergence speed.

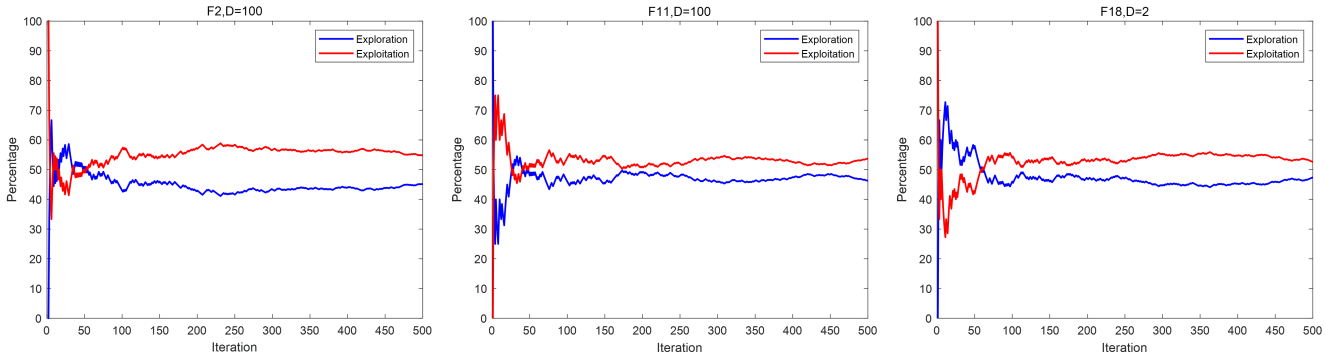


Figure 6. Exploration and exploitation phases in EOA.

By quantitatively evaluating the exploration and exploitation percentages, EOA's dynamic adjustment capabilities and its ability to achieve a balance between the two search strategies become evident. These findings contribute to a deeper understanding of EOA's search behavior and highlight its potential for efficient optimization.

4.1.4. Wilcoxon Rank-Sum Test Analysis of EOA

To demonstrate the robustness and fairness of EOA, statistical tests are employed to evaluate its superiority over other algorithms. Table 11 gives the Wilcoxon rank-sum test results with a significance level at 0.05 among nine comparative algorithms for twenty-three classical test functions (F1–F23). The rejection of the null hypothesis is considered strong when $p < 0.05$. In addition, the NaN mark indicates comparable algorithm performance.

Based on the results presented in Table 11, it is evident that EOA exhibits superior performance to the classical AO algorithm across all test functions except for F7 and F10. Furthermore, EOA and AO show the same results on F9 and F11, indicating that there is no significant difference between the two algorithms on these test functions. Similar experimental results are also revealed on the remaining comparison algorithms, proving that EOA has significantly superior convergence performance over the remaining comparison algorithms.

Table 11. Wilcoxon rank-sum test between EOA and other algorithms using the classical test functions (F1–F23).

Fun	Dimensions	Comparative Algorithms								
		AO	PSO	SCA	WOA	GWO	HBA	SSA	SNS	BES
		p-Value	*p*-Value	*p*-Value	*p*-Value	*p*-Value	*p*-Value	*p*-Value	*p*-Value	*p*-Value
F1	10	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	8.8658×10^{-7}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}
F2	10	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	5.5727×10^{-10}	3.0199×10^{-11}	3.0199×10^{-11}
F3	10	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}
F4	10	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}
F5	10	6.4789×10^{-12}	6.4789×10^{-12}	6.4789×10^{-12}	6.4789×10^{-12}	6.4789×10^{-12}	6.4789×10^{-12}	1.8692×10^{-11}	6.4789×10^{-12}	6.4789×10^{-12}
F6	10	1.9256×10^{-12}	1.7203×10^{-12}	1.7203×10^{-12}	1.7203×10^{-12}	1.7203×10^{-12}	1.9256×10^{-12}	4.1040×10^{-11}	3.0159×10^{-12}	3.3371×10^{-1}
F7	10	3.0418×10^{-1}	3.0199×10^{-11}	2.8716×10^{-10}	4.8011×10^{-7}	1.0907×10^{-5}	2.3399×10^{-1}	1.3832×10^{-2}	4.0330×10^{-3}	5.5611×10^{-4}
F8	10	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}	1.2118×10^{-12}
F9	10	*NaN*	1.2118×10^{-12}	1.6560×10^{-11}	1.6080×10^{-1}	4.1926×10^{-2}	*NaN*	*NaN*	8.1523×10^{-2}	6.2470×10^{-10}
F10	10	3.3371×10^{-1}	1.2118×10^{-12}	1.2118×10^{-12}	2.8987×10^{-8}	1.1772×10^{-13}	*NaN*	4.1911×10^{-2}	*NaN*	5.4183×10^{-12}
F11	10	*NaN*	1.2118×10^{-12}	1.2118×10^{-12}	6.6096×10^{-5}	3.4526×10^{-7}	*NaN*	*NaN*	*NaN*	5.8522×10^{-9}
F12	10	2.3982×10^{-11}	2.3982×10^{-11}	2.3982×10^{-11}	2.3982×10^{-11}	2.3982×10^{-11}	1.4363×10^{-10}	5.5063×10^{-10}	7.7953×10^{-8}	7.3512×10^{-4}
F13	10	7.1391×10^{-11}	2.6286×10^{-11}	2.6286×10^{-11}	2.6286×10^{-11}	2.6286×10^{-11}	3.9302×10^{-11}	3.7900×10^{-7}	2.4219×10^{-4}	3.5146×10^{-1}
F14	2	1.3369×10^{-11}	2.8637×10^{-2}	1.3369×10^{-11}	1.3369×10^{-11}	1.3369×10^{-11}	1.0114×10^{-10}	1.1480×10^{-11}	4.7792×10^{-2}	1.3125×10^{-1}
F15	4	3.4742×10^{-10}	1.0934×10^{-6}	4.0772×10^{-11}	1.0937×10^{-10}	6.5261×10^{-7}	4.4205×10^{-6}	1.4067×10^{-4}	5.2650×10^{-5}	2.3388×10^{-1}
F16	2	5.4941×10^{-11}	3.1507×10^{-12}	5.5727×10^{-10}	1.3367×10^{-5}	4.1191×10^{-1}	2.3043×10^{-10}	4.0988×10^{-12}	1.7203×10^{-12}	6.9047×10^{-8}
F17	2	1.2870×10^{-9}	1.2118×10^{-12}	3.0199×10^{-11}	9.5873×10^{-1}	9.8231×10^{-1}	3.2975×10^{-9}	1.6332×10^{-10}	1.2118×10^{-12}	4.1748×10^{-6}
F18	2	3.0199×10^{-11}	2.0575×10^{-11}	3.0199×10^{-11}	8.1014×10^{-10}	3.0199×10^{-11}	1.1706×10^{-4}	5.5270×10^{-10}	1.9558×10^{-11}	1.7203×10^{-12}
F19	3	3.0199×10^{-11}	6.0253×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	9.9765×10^{-7}	3.1118×10^{-1}	3.1507×10^{-12}	1.2118×10^{-12}
F20	6	1.7290×10^{-6}	5.5923×10^{-1}	3.6897×10^{-11}	1.7649×10^{-2}	9.2344×10^{-1}	2.1566×10^{-3}	5.7460×10^{-2}	9.1863×10^{-9}	1.8101×10^{-3}
F21	4	2.9747×10^{-11}	2.6882×10^{-2}	2.9747×10^{-11}	2.9747×10^{-11}	2.9747×10^{-11}	1.0845×10^{-8}	5.5907×10^{-1}	2.1653×10^{-10}	1.5919×10^{-3}
F22	4	3.0199×10^{-11}	6.6060×10^{-1}	3.0199×10^{-11}	3.0199×10^{-11}	3.0199×10^{-11}	1.7613×10^{-1}	2.1959×10^{-7}	9.7566×10^{-11}	2.5317×10^{-2}
F23	4	4.9752×10^{-11}	1.8506×10^{-1}	3.0199×10^{-11}	3.3384×10^{-11}	3.0199×10^{-11}	3.7742×10^{-2}	3.8307×10^{-5}	1.3974×10^{-11}	4.2813×10^{-6}
W|L|T		19|2|2	18|5|0	23|0|0	21|2|0	20|3|0	18|2|3	18|3|2	20|1|2	19|4|0

4.1.5. Time Consuming Test of EOA

Running time is calculated to analyze the efficiency of EOA. Each algorithm is executed in 10 dimensions, and the time is recorded. Table 12 presents the results, indicating that the EOA's running time is longer than that of PSO, SCA, WOA, and GWO but shorter than that of other algorithms. Given its superior performance, the time consumption of EOA can be considered acceptable.

Table 12. Computation time results of algorithms on 23 benchmark functions.

Fun	Comparative Methods									
	EOA	AO	PSO	SCA	WOA	GWO	HBA	SSA	SNS	BES
F1	6.8115×10^{-2}	9.8085×10^{-2}	3.4956×10^{-2}	5.4717×10^{-2}	4.3282×10^{-2}	5.9570×10^{-2}	8.9540×10^{-2}	9.0899×10^{-2}	1.3116×10^{-1}	2.1016×10^{-1}
F2	6.4422×10^{-2}	9.9208×10^{-2}	3.4124×10^{-2}	5.3613×10^{-2}	4.3075×10^{-2}	6.9087×10^{-2}	1.2605×10^{-1}	9.3343×10^{-2}	1.5505×10^{-1}	2.1274×10^{-1}
F3	1.2644×10^{-1}	1.8611×10^{-1}	6.1275×10^{-2}	7.2501×10^{-2}	6.6503×10^{-2}	9.1233×10^{-2}	1.5829×10^{-1}	1.7288×10^{-1}	1.6094×10^{-1}	3.3838×10^{-1}
F4	6.7713×10^{-2}	9.7462×10^{-2}	2.2772×10^{-2}	5.9037×10^{-2}	4.4468×10^{-2}	6.4078×10^{-2}	1.2530×10^{-1}	9.6689×10^{-2}	1.4745×10^{-1}	2.1220×10^{-1}
F5	1.0952×10^{-1}	1.5159×10^{-1}	3.9064×10^{-2}	7.3286×10^{-2}	5.2367×10^{-2}	6.1165×10^{-2}	1.2314×10^{-1}	1.4017×10^{-1}	1.3594×10^{-1}	2.6024×10^{-1}
F6	6.1147×10^{-2}	1.0082×10^{-1}	2.1587×10^{-2}	3.8285×10^{-2}	4.2003×10^{-2}	7.7522×10^{-2}	1.5042×10^{-1}	1.0823×10^{-1}	1.1205×10^{-1}	2.4893×10^{-1}
F7	1.0970×10^{-1}	1.4743×10^{-1}	4.5540×10^{-2}	6.2368×10^{-2}	6.4391×10^{-2}	6.8581×10^{-2}	1.3976×10^{-1}	1.3761×10^{-1}	1.4567×10^{-1}	2.8342×10^{-1}
F8	8.1905×10^{-2}	1.1927×10^{-1}	3.2974×10^{-2}	6.7624×10^{-2}	5.7245×10^{-2}	6.1309×10^{-2}	1.4809×10^{-1}	1.2963×10^{-1}	1.4778×10^{-1}	2.6745×10^{-1}
F9	6.5281×10^{-2}	1.0431×10^{-1}	3.2066×10^{-2}	6.3449×10^{-2}	3.8757×10^{-2}	5.6972×10^{-2}	9.6648×10^{-2}	8.5491×10^{-2}	1.2795×10^{-1}	2.1911×10^{-1}
F10	5.8755×10^{-2}	1.0672×10^{-1}	2.9341×10^{-2}	5.9860×10^{-2}	4.8606×10^{-2}	5.3515×10^{-2}	1.3242×10^{-1}	9.2782×10^{-2}	1.4505×10^{-1}	2.1673×10^{-1}
F11	8.3763×10^{-2}	1.4699×10^{-1}	5.0108×10^{-2}	5.6213×10^{-2}	5.4075×10^{-2}	7.9552×10^{-2}	1.5825×10^{-1}	1.3460×10^{-1}	1.3522×10^{-1}	2.6654×10^{-1}
F12	2.4026×10^{-1}	3.1589×10^{-1}	1.5856×10^{-1}	1.2824×10^{-1}	1.3854×10^{-1}	1.3809×10^{-1}	2.9596×10^{-1}	1.8116×10^{-1}	2.7592×10^{-1}	5.0326×10^{-1}
F13	2.3316×10^{-1}	3.0596×10^{-1}	1.2143×10^{-1}	1.2665×10^{-1}	1.2966×10^{-1}	1.3382×10^{-1}	2.4408×10^{-1}	2.1936×10^{-1}	2.5295×10^{-1}	5.0905×10^{-1}
F14	6.7698×10^{-1}	7.5053×10^{-1}	3.4351×10^{-1}	3.3985×10^{-1}	3.3099×10^{-1}	3.5265×10^{-1}	4.3041×10^{-1}	4.3446×10^{-1}	5.1826×10^{-1}	1.2380×10^{0}
F15	6.1532×10^{-2}	9.3392×10^{-2}	3.6226×10^{-2}	4.2978×10^{-2}	3.8608×10^{-2}	3.8113×10^{-2}	1.1031×10^{-1}	9.0620×10^{-2}	1.2725×10^{-1}	2.6246×10^{-1}
F16	5.9045×10^{-2}	1.0336×10^{-1}	3.3799×10^{-2}	3.9366×10^{-2}	3.6385×10^{-2}	4.0252×10^{-2}	1.0379×10^{-1}	7.7384×10^{-2}	1.2831×10^{-1}	2.1716×10^{-1}
F17	6.6089×10^{-2}	1.0461×10^{-1}	1.9708×10^{-2}	3.9950×10^{-2}	3.2535×10^{-2}	3.7021×10^{-2}	9.5610×10^{-2}	6.3065×10^{-2}	1.3259×10^{-1}	2.0606×10^{-1}
F18	8.3514×10^{-2}	1.1229×10^{-1}	3.1262×10^{-2}	4.5131×10^{-2}	3.1597×10^{-2}	2.9165×10^{-2}	1.1249×10^{-1}	8.4600×10^{-2}	1.0763×10^{-1}	2.1768×10^{-1}
F19	7.3697×10^{-2}	1.2476×10^{-1}	2.8396×10^{-2}	4.4830×10^{-2}	4.2574×10^{-2}	3.9099×10^{-2}	1.1571×10^{-1}	9.0114×10^{-2}	1.4465×10^{-1}	2.6988×10^{-1}
F20	7.3176×10^{-2}	1.0807×10^{-1}	2.8218×10^{-2}	5.8219×10^{-2}	3.6605×10^{-2}	4.6614×10^{-2}	1.3891×10^{-1}	1.1751×10^{-1}	1.3354×10^{-1}	2.3362×10^{-1}
F21	8.7999×10^{-2}	1.5877×10^{-1}	3.9831×10^{-2}	5.5214×10^{-2}	4.8524×10^{-2}	6.4906×10^{-2}	1.4992×10^{-1}	9.4011×10^{-2}	1.4039×10^{-1}	2.6573×10^{-1}
F22	1.0196×10^{-1}	1.8355×10^{-1}	5.1664×10^{-2}	6.7354×10^{-2}	5.9536×10^{-2}	6.7559×10^{-2}	1.5836×10^{-1}	1.1122×10^{-1}	1.6278×10^{-1}	3.8579×10^{-1}
F23	1.1882×10^{-1}	1.8635×10^{-1}	6.3300×10^{-2}	7.5518×10^{-2}	6.1244×10^{-2}	7.9163×10^{-2}	2.1806×10^{-1}	1.2640×10^{-1}	2.1946×10^{-1}	4.3389×10^{-1}

4.2. Experiment 2: IEEE CEC2019 Test Suite

To further validate the effectiveness of EOA, we employ the CEC2019 standard test functions to confirm its superiority. The explicit introduction of the IEEE CEC2019 test suite is provided in Table 13. All experiments are compared against seven other algorithms,

namely, sine cosine algorithm (SCA [33]), lion swarm optimization (LSO [36]), AOA, PSO, reptile search algorithm (RSA [37]), WOA, and sparrow search algorithm (SSA [34]). The results are shown in Table 14. Since the average and STD of EOA are the least among all algorithms for more than half of the functions, we can confidently conclude that the performance of EOA is excellent.

Table 13. IEEE CEC2019 test suite.

No.	Functions	Fmin	D	Search Range
1	Storn's Chebyshev Polynomial Fitting Problem	1	9	[−8192, 8192]
2	Inverse Hilbert Matrix Problem	1	16	[−16,384, 16,384]
3	Lennard-Jones Minimum Energy Cluster	1	18	[−4, 4]
4	Rastrigin's Function	1	10	[−100, 100]
5	Griewangk's Function	1	10	[−100, 100]
6	Weierstrass Function	1	10	[−100, 100]
7	Modified Schwefel's Function	1	10	[−100, 100]
8	Expanded Schaffer's F6 Function	1	10	[−100, 100]
9	Happy Cat Function	1	10	[−100, 100]
10	Ackley Function	1	10	[−100, 100]

Table 14. Results of the comparative methods on IEEE CEC2019 test suite.

Fun No. Measure	Comparative Methods							
	EOA	SCA	AOA	LSO	PSO	RSA	WOA	SSA
F1								
Average	1.0000×10^{0}	4.6261×10^{2}	1.0000×10^{0}	1.0000×10^{0}	1.5491×10^{3}	1.0000×10^{0}	2.4689×10^{3}	1.0000×10^{0}
STD	0.0000×10^{0}	6.5011×10^{2}	1.4760×10^{-4}	0.0000×10^{0}	3.0365×10^{3}	0.0000×10^{0}	3.5679×10^{3}	0.0000×10^{0}
F2								
Average	4.7423×10^{0}	2.6214×10^{1}	4.8790×10^{0}	5.0000×10^{0}	2.0953×10^{1}	5.0000×10^{0}	4.5502×10^{1}	5.0000×10^{0}
STD	2.2596×10^{-1}	1.1409×10^{1}	1.6112×10^{-1}	0.0000×10^{0}	1.0093×10^{1}	0.0000×10^{0}	1.9774×10^{1}	8.4630×10^{-6}
F3								
Average	5.5860×10^{0}	1.2715×10^{1}	5.6322×10^{0}	5.9753×10^{0}	1.2068×10^{1}	8.9564×10^{0}	7.2275×10^{1}	1.0498×10^{1}
STD	1.1182×10^{0}	1.8979×10^{-1}	1.7676×10^{0}	9.2398×10^{-1}	1.1722×10^{0}	1.0917×10^{0}	2.5775×10^{0}	2.8290×10^{0}
F4								
Average	4.8776×10^{1}	5.2948×10^{1}	6.0218×10^{1}	7.5242×10^{1}	5.8290×10^{1}	8.5709×10^{1}	5.8762×10^{1}	8.4331×10^{1}
STD	1.1907×10^{1}	8.1644×10^{0}	1.6815×10^{1}	2.2697×10^{1}	1.9287×10^{1}	1.5773×10^{1}	2.4084×10^{1}	1.9615×10^{1}
F5								
Average	1.8158×10^{1}	1.0765×10^{1}	8.1016×10^{1}	8.0803×10^{1}	1.0118×10^{1}	9.5532×10^{1}	2.5452×10^{0}	6.7188×10^{1}
STD	1.8227×10^{1}	4.0485×10^{0}	2.4097×10^{1}	3.0927×10^{1}	7.9447×10^{0}	2.3995×10^{1}	5.9997×10^{-1}	2.8603×10^{1}
F6								
Average	6.7141×10^{0}	8.1202×10^{0}	1.0973×10^{1}	9.4200×10^{0}	8.9996×10^{0}	1.1144×10^{1}	9.2302×10^{0}	1.1809×10^{1}
STD	1.7583×10^{0}	1.3545×10^{0}	1.2479×10^{0}	8.9244×10^{-1}	1.8231×10^{0}	1.1472×10^{0}	2.1539×10^{0}	1.6148×10^{0}
F7								
Average	1.5222×10^{3}	1.6382×10^{3}	1.2810×10^{3}	1.7133×10^{3}	1.4719×10^{3}	1.9188×10^{3}	1.4682×10^{3}	1.6061×10^{3}
STD	2.8147×10^{2}	1.8822×10^{2}	2.6387×10^{2}	3.0674×10^{2}	3.2844×10^{2}	1.9653×10^{2}	3.1903×10^{2}	3.7153×10^{2}
F8								
Average	4.5748×10^{0}	4.5615×10^{0}	4.7911×10^{0}	4.7051×10^{0}	4.7021×10^{0}	5.0206×10^{0}	4.6447×10^{0}	4.9544×10^{0}
STD	2.6585×10^{-1}	2.3974×10^{-1}	3.1398×10^{-1}	3.1656×10^{-1}	3.1930×10^{-1}	1.6797×10^{-1}	2.8375×10^{-1}	3.5998×10^{-1}
F9								
Average	1.4741×10^{0}	1.6507×10^{0}	3.0125×10^{0}	3.5019×10^{0}	1.5483×10^{0}	3.0824×10^{0}	1.4248×10^{0}	2.6399×10^{0}
STD	2.3896×10^{-1}	1.2800×10^{-1}	7.3992×10^{-1}	5.8104×10^{-1}	4.3488×10^{-1}	4.3308×10^{-1}	2.3794×10^{-1}	9.1971×10^{-1}
F10								
Average	2.1197×10^{1}	2.1291×10^{1}	2.1103×10^{1}	2.1466×10^{1}	2.1187×10^{1}	2.1456×10^{1}	2.1288×10^{1}	2.1024×10^{1}
STD	1.1478×10^{0}	1.1327×10^{0}	1.4680×10^{-1}	1.0705×10^{-1}	1.9119×10^{-1}	1.0788×10^{-1}	1.4676×10^{-1}	1.0726×10^{-1}
W|L|T		8|2|0	7|2|1	9|0|1	7|3|0	9|1|0	7|3|0	7|2|1
Mean Rank	2.1000×10^{0}	4.4000×10^{0}	3.8000×10^{0}	5.2000×10^{0}	3.7000×10^{0}	6.1000×10^{0}	4.3000×10^{0}	4.7000×10^{0}
Final Rank	1.0000×10^{0}	5.0000×10^{0}	3.0000×10^{0}	7.0000×10^{0}	2.0000×10^{0}	8.0000×10^{0}	4.0000×10^{0}	6.0000×10^{0}

5. EOA for Solving Industrial Engineering Design Problems

This section presents the evaluation of the proposed algorithm on six real-world engineering problems with inequality constraints. These problems include the design of a speed reducer, pressure vessel, Three-bar trusses, welded beams, tension/compression springs, and cantilever beams. To handle the constraints, a penalty function approach is

used, whereby a significant penalty value is assigned if any constraint is violated during the optimization process. The same parameter settings used in the previous experiments are employed to ensure consistency and comparability of results across different problems. By testing the proposed algorithm on these diverse engineering problems, we can assess its robustness and effectiveness in handling complex constrained optimization tasks.

5.1. Speed Reducer Design Problem

The objective of this problem is to minimize the total weight of a gearbox through the optimization of seven variables, which include the gear teeth curvature stress, transverse deflections of the shafts, stresses in the shafts, and surface stress. Figure 7 depicts the design of the problem, and its mathematical formulation is presented as Equations (10) and (11).

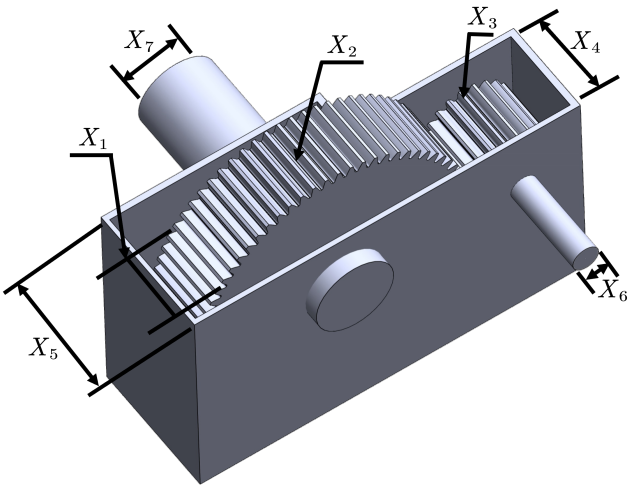


Figure 7. Speed reducer problem.

The performance of EOA is compared against several other optimization algorithms, including AOA, AO, GWO, PSO, SCA, GA, MDA, MFO, FA, HS, and AAO. The results of this comparison are presented in Table 15. From the results, EOA outperformed all other methods and obtained the highest rank in solving the problem. AAO ranked second, followed by GWO. These results demonstrate the effectiveness of EOA in tackling complex optimization problems.

$$
\begin{aligned}
\textit{Minimize}: \quad & f(\vec{x}) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0943\right) - 1.508x_1\left(x_6^2 + x_7^2\right) \\
& + 7.4777\left(x_6^3 + x_7^3\right) \\
\textit{Subject to}: \quad & g_1(\vec{x}) = \frac{27}{x_1x_2^2x_3} - 1 \le 0 \quad g_2(\vec{x}) = \frac{397.5}{x_1x_2^2x_2^2} - 1 \le 0 \\
& g_3(\vec{x}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0 \quad g_4(\vec{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0 \\
& g_5(\vec{x}) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6}}{110x_6^3} - 1 \le 0 \\
& g_6(\vec{x}) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 157.5 \times 10^6}}{85x_6^3} - 1 \le 0 \\
& g_7(\vec{x}) = \frac{x_2x_3}{40} - 1 \le 0 \quad g_8(\vec{x}) = \frac{5x_2}{x_1} - 1 \le 0 \quad g_9(\vec{x}) = \frac{x_1}{12x_2} - 1 \le 0 \\
& g_{10}(\vec{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0 \quad g_{11}(\vec{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0
\end{aligned}
\tag{10}
$$

$$\begin{aligned} \textit{where}: \quad & 2.6 \le x_1 \le 3.6, \quad 0.7 \le x_2 \le 0.8, \quad 17 \le x_3 \le 28, \quad 7.3 \le x_4 \le 8.3, \\ & 7.8 \le x_5 \le 8.3, \quad 2.9 \le x_6 \le 3.9, \quad 5 \le x_7 \le 5.5 \end{aligned} \tag{11}$$

Table 15. Results of the comparative algorithms for solving the speed reducer design problem.

Algorithm	Optimal Values for Variables							Optimum	Ranking
	x_1	x_2	x_3	x_4	x_5	x_6	x_7		
EOA	3.5	0.7	17	7.317	7.80572	3.350572	5.286685	2996.691188	1
AOA [12]	3.50384	0.7	17	7.3	7.72933	3.35649	5.2867	2997.9157	4
AO [16]	3.5021	0.7	17	7.3099	7.7476	3.3641	5.2994	3007.7328	6
GWO [14]	3.501	0.7	17	7.3	7.811013	3.350704	5.287411	2997.81965	3
PSO [10]	3.5001	0.7	17.0002	7.5177	7.7832	3.3508	5.2867	3145.922	12
SCA [33]	3.508755	0.7	17	7.3	7.8	3.46102	5.289213	3030.563	9
GA [17]	3.510253	0.7	17	8.35	7.8	3.362201	5.287723	3067.561	11
MDA [38]	3.5	0.7	17	7.3	7.670396	3.542421	5.245814	3019.583365	8
MFO [39]	3.49745	0.7	17	7.82775	7.71245	3.35178	5.28635	2998.9408	5
FA [40]	3.507495	0.7001	17	7.71967	8.08085	3.35151	5.28705	3010.137492	7
HS [41]	3.520124	0.7	17	8.37	7.8	3.36697	5.288719	3029.002	10
AAO [42]	3.499	0.6999	17	7.3	7.8	3.3502	5.2872	2996.783	2

5.2. Pressure Vessel Design Problem

The objective of this design problem is to minimize the manufacturing cost of pressure vessels. The problem involves optimizing four design variables, namely, the thickness of the shell (T_s) and head (T_h), the length of the cylindrical section (L), and the inner radius (R), while satisfying four constraints, as shown in Figure 8. The mathematical formulation is in Equation (12). The experimental results, presented in Table 16, demonstrate that the proposed EOA outperformed other state-of-the-art algorithms, achieving the optimal solution for this problem. These findings highlight the effectiveness of EOA in tackling complex optimization problems in the field of engineering design.

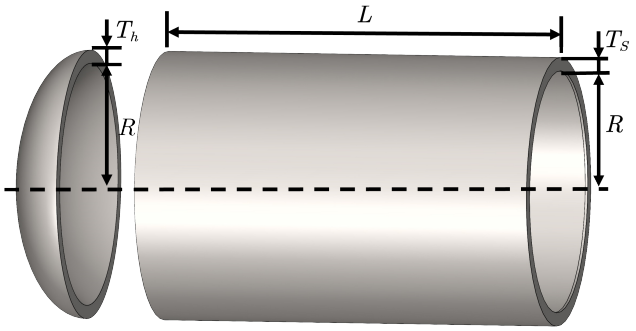


Figure 8. Pressure vessel design problem.

$$\begin{aligned} \textit{Consider}: \quad & \vec{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L] \\ \textit{Minimize}: \quad & f(\vec{x}) = 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2 + 3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3 \\ \textit{Subject to}: \quad & g_1(\vec{x}) = -x_1 + 0.0193 x_3 \le 0 \\ & g_2(\vec{x}) = -x_2 + 0.00954 x_3 \le 0 \\ & g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \\ & g_4(\vec{x}) = x_4 - 240 \le 0 \\ \textit{Variable range}: \quad & 0 \le x_1 \le 99, \\ & 0 \le x_2 \le 99, \\ & 10 \le x_3 \le 200, \\ & 10 \le x_4 \le 200 \end{aligned} \tag{12}$$

Table 16. Results of the comparative algorithms for solving the pressure vessel design problem.

Algorithm	Optimal Values for Variables				Optimum	Ranking
	T_s	T_h	R	L		
EOA	0.7787543	0.3858478	40.32629	199.9165	5892.3459	1
AOA [12]	0.8303737	0.4162057	42.75127	169.3454	6048.7844	4
AO [16]	1.054	0.182806	59.6219	38.805	5949.2258	2
WOA [13]	0.8125	0.4375	42.0982699	176.638998	6059.741	6
SMA [15]	0.7931	0.3932	40.6711	196.2178	5994.1857	3
GWO [14]	0.8125	0.4345	42.0892	176.7587	6051.5639	5
PSO-SCA [33]	0.8125	0.4375	42.098446	176.6366	6059.71433	8
MVO [43]	0.8125	0.4375	42.090738	176.73869	6060.8066	11
GA [17]	0.8125	0.4375	42.097398	176.65405	6059.94634	10
HPSO [44]	0.8125	0.4375	42.0984	176.6366	6059.7143	7
ES [18]	0.8125	0.4375	42.098087	176.640518	6059.7456	9

5.3. Three-Bar Truss Design Problem

This engineering design problem requires creating a truss composed of three bars to minimize its weight. The problem has a highly constrained search space, with the structural parameters illustrated in Figure 9 and mathematically formulated in Equation (13). Table 17 presents the results of solving the three-bar truss design problem using EOA and compares its performance against several other state-of-the-art optimization algorithms reported in the literature. The experimental results show that EOA outperformed other well-known optimization techniques, achieving the minimum weight of the three-bar truss. These findings highlight the effectiveness of EOA in tackling complex engineering design problems with highly constrained search spaces.

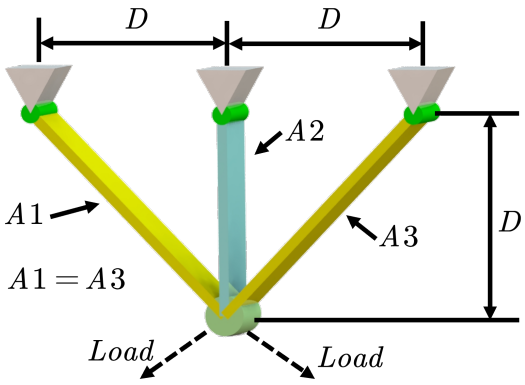


Figure 9. Three-bar truss design problem.

Table 17. Results of the comparative algorithms for solving the Three-bar truss design problem.

Algorithm	Optimal Values for Variables		Optimum	Ranking
	x_1	x_3		
EOA	0.788576562	0.408197	263.8628605	1
AOA [12]	0.79369	0.39426	263.9154	8
Ray and Sain [45]	0.795	0.395	264.3	10
AO [16]	0.7926	0.3966	263.8684	2
SSA [34]	0.78866541	0.408275784	263.89584	3
MBA [46]	0.788565	0.4085597	263.89585	5
PSO-DE [47]	0.7886751	0.4082482	263.89584	3
CS [48]	0.78867	0.40902	263.9716	9
GOA [49]	0.788897556	0.40761957	263.8958815	6
MFO [39]	0.788244771	0.409466906	263.8959797	7

$$
\begin{aligned}
&\textit{Consider}: && \vec{x} = [x_1, x_2] = [A_1, A_2] \\
&\textit{Minimize}: && f(\vec{x}) = \left(2\sqrt{2x_1} + x_2\right) \times l \\
&\textit{Subject to}: && g_1(\vec{x}) = \frac{\sqrt{2x_1} + x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \leq 0 \\
& && g_2(\vec{x}) = \frac{x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \leq 0 \\
& && g_3(\vec{x}) = \frac{1}{\sqrt{2x_2} + x_1} P - \sigma \leq 0 \\
& && 0 \leq x_1, x_2 \leq 1 \\
&\textit{where}: && l = 100\ cm \\
& && P = 2\ KN/cm^2 \\
& && \sigma = 2\ KN/cm^2
\end{aligned}
\tag{13}
$$

5.4. Welded Beam Design Problem

The objective of this problem is to minimize the fabrication cost by optimizing four variables: the thickness of the bar (b), thickness of weld (h), height of the bar (t), and length of the attached part of the bar (l), subject to seven constraints, as shown in Figure 10. The formulation of this problem is presented in Equation (14). EOA is applied for solving the welded beam design and compared with the results of several algorithms published in the literature, and the experimental results are presented in Table 18. The results demonstrate that EOA achieved superior performance in solving this problem, highlighting its effectiveness in tackling complex engineering optimization tasks.

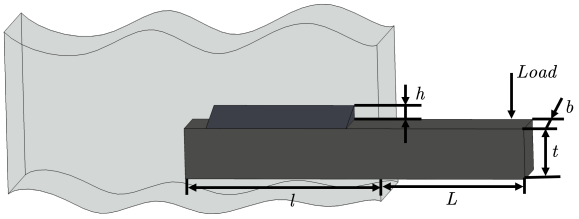


Figure 10. Welded beam design problem.

Table 18. Results of the comparative algorithms for solving the welded beam design problem.

Algorithm	Optimal Values for Variables				Optimum	Ranking
	h	***l***	***t***	***b***		
EOA	0.20572584	3.4706	9.036535	0.2057338	1.724879	1
GA [17]	0.2489	6.173	8.1789	0.2533	2.43	8
CPSO [50]	0.202369	3.544214	9.04821	0.205723	1.72802	3
WOA [13]	0.205396	3.484293	9.037426	0.206276	1.730499	4
MVO [43]	0.205463	3.473193	9.044502	0.205695	1.72645	2
DAVID [51]	0.2434	6.2552	8.2915	0.2444	2.3841	7
APPROX [51]	0.2444	6.2189	8.2915	0.2444	2.3815	6
HS [52]	0.2442	6.2231	8.2915	0.24	2.3807	5
SIMPLEX [51]	0.2792	5.6256	7.7512	0.2796	2.5307	9

$$
\begin{aligned}
&\textit{Consider}: && \vec{x} = [x_1, x_2, x_3, x_4] = [h, l, tt, bb] \\
&\textit{Minimize}: && f(\vec{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2) \\
&\textit{Subject to}: && g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \leq 0 \\
& && g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \leq 0 \\
& && g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \leq 0 \\
& && g_4(\vec{x}) = x_1 - x_4 \leq 0 \\
& && g_5(\vec{x}) = P - P_c(\vec{x}) \leq 0 \\
& && g_6(\vec{x}) = 0.125 - x_1 \leq 0 \\
& && g_7(\vec{x}) = 1.10471 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \leq 0 \\
&\textit{Variable range}: && 0.1 \leq x_1 \leq 2, 0.1 \leq x_2 \leq 10, 0.1 \leq x_3 \leq 10, 0.1 \leq x_4 \leq 2 \\
&\textit{where}: && \tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \quad \tau' = \frac{p}{\sqrt{2x_1x_2}}, \tau'' = \frac{MR}{J}, \\
& && M = P\left(L + \frac{x_2}{2}\right), \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \\
& && J = 2\left\{\sqrt{2x_1x_2}\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}, \delta(\vec{x}) = \frac{6PL^3}{Ex_3^2x_4} \\
& && P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), P = 6000lb, l = 14in., \delta_{\max} = 0.25in., \\
& && E = 30 \times 1^6 psi, G = 12 \times 10^6 psi, \tau_{\max} = 13600psi, \sigma_{\max} = 30000psi
\end{aligned}
\tag{14}
$$

5.5. Tension/Compression Spring Design Problem

The objective of this problem is to minimize the weight of a tension/compression spring. As depicted in Figure 11, the problem involves optimizing three variables, namely, the mean coil diameter D, the wire diameter d, and the number of coils N, subject to four constraints; the mathematical formulation is in Equation (15). To evaluate the performance of the proposed EOA, we compare it with several state-of-the-art optimization algorithms, including AOA, PSO, OBSCA, ES, WOA, RO, MVO, GSA, CPSO, and CC. The results of the experiments, shown in Table 19, indicate that EOA outperformed all other methods, demonstrating its effectiveness in tackling this challenging optimization problem.

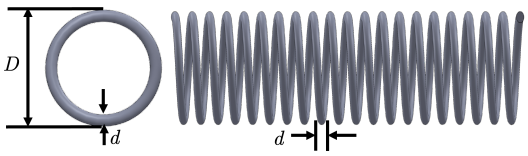


Figure 11. Tension/compression spring design problem.

Table 19. Results of the comparative algorithms for solving the tension/compression spring design problem.

Algorithm	Optimal Values for Variables			Optimum	Ranking
	d	*D*	*N*		
EOA	0.051073	0.3420839	11.4717	0.012021	1
AOA [12]	0.05	0.349809	11.8637	0.012124	2

Table 19. *Cont.*

Algorithm	Optimal Values for Variables			Optimum	Ranking
	d	*D*	*N*		
PSO [10]	0.051728	0.357644	11.244543	0.0126747	5
PO [26]	0.052482	0.375940	10.245091	0.0126720	3
OBSCA [53]	0.0523	0.31728	12.54854	0.012625	4
ES [18]	0.051643	0.35536	11.397926	0.012698	8
WOA [13]	0.051207	0.345215	12.004032	0.0126763	6
RO [54]	0.05137	0.349096	11.76279	0.0126788	7
MVO [43]	0.05251	0.37602	10.33513	0.01279	10
GSA [23]	0.050276	0.32368	13.52541	0.0127022	9
CPSO [50]	0.051728	0.357644	11.244543	0.0126747	5
CC [55]	70.05	0.3159	14.25	0.0128334	11

$$
\begin{aligned}
&\textit{Consider}: && \vec{x} = [x_1, x_2, x_3] = [d, D, N] \\
&\textit{Minimize}: && f(\vec{x}) = (x_3 + 2) x_2 x_1^2 \\
&\textit{Subject to}: && g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0 \\
& && g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} \leq 0 \\
& && g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0 \\
& && g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \\
&\textit{Variable range}: && 0.05 \leq x_1 \leq 2 \\
& && 0.25 \leq x_2 \leq 1.30 \\
& && 2.00 \leq x_3 \leq 15
\end{aligned}
\tag{15}
$$

5.6. Cantilever Beam Design Problem

This problem is a concrete engineering optimization problem aimed at minimizing the total weight of a cantilever beam by optimizing the parameters of a hollow square cross-section. Figure 12 depicts the design of the problem, and its mathematical formulation is presented in Equation (16).

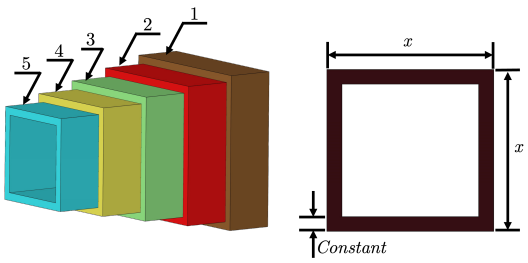


Figure 12. Cantilever beam design problem.

$$
\begin{aligned}
&\textit{Consider}: && \vec{x} = [x_1, x_2, x_3, x_4, x_5] \\
&\textit{Minimize}: && f(\vec{x}) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5) \\
&\textit{Subject to}: && g(\vec{x}) = \frac{60}{x_1^3} + \frac{27}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0 \\
&\textit{Variable range}: && 70.01 \leq x_1, x_2, x_3, x_4, x_5 \leq 100
\end{aligned}
\tag{16}
$$

To evaluate the performance of EOA in solving the cantilever beam design problem, we compare it with several algorithms: SMA, MFO, ALO, MMA, and SOS. From the

experimental results listed in Table 20, it can be concluded that EOA came in the first rank compared with all other algorithms, which proves that EOA has the ability to solve the engineering problem with significant advantages.

Table 20. Results of the comparative algorithms for solving the cantilever beam design problem.

Algorithm	Optimal Values for Variables					Optimum Weight	Ranking
	x_1	x_2	x_3	x_4	x_5		
EOA	6.023716	5.303997	4.4954247	3.49728	2.15328	1.33994	1
SMA [15]	6.017757	5.310892	4.493758	3.501106	2.150159	1.33996	3
MFO [39]	5.983	5.3167	4.4973	3.5136	2.1616	1.33998	6
ALO [56]	6.01812	5.31142	4.48836	3.49751	2.158329	1.33995	2
MMA [57]	6.01	5.3	4.49	3.49	2.15	1.34	5
SOS [58]	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996	3

6. Conclusions

Compared to traditional optimization algorithms, meta-heuristic algorithms offer distinct advantages in tackling challenging optimization problems, such as high-dimensional and multimodal optimization problems. However, these algorithms are not general and require further improvement in optimization accuracy and speed for certain problems. In this paper, a novel meta-heuristic algorithm, referred to as EOA, is proposed. EOA consists of two phases: party nomination and presidential campaign, corresponding to exploratory and exploitative phases, respectively. In contrast to the classical algorithms, EOA incorporates the random factors in both the exploration and exploitation phases and thus achieves a balance between exploration and exploitation.

To evaluate the performance of the proposed algorithm, we conduct experiments using twenty-three classical test functions and the CEC2019 test set, comparing its performance to that of other classical algorithms. Qualitative analyses demonstrate that EOA exhibits higher amplitudes and frequencies in early iterations, gradually decreasing in subsequent iterations, indicating its strong exploration and exploitation capabilities throughout the optimization process. For unimodal functions, the EOA notably outperforms other comparison algorithms in terms of convergence speed and accuracy, highlighting its high exploitability and reliability. This superiority can be attributed to the mediation capability of the control factors in the presidential election phase, which accelerates convergence speed and narrows down the search region. In the case of multimodal functions, the EOA generally outperforms other comparison algorithms, primarily due to the stochastic strategy reference factor employed in the party nomination phase. This factor provides greater flexibility to EOA and helps EOA to effectively escape local optima. Furthermore, the adaptability of EOA is demonstrated through multidimensional tests, showcasing its efficacy in handling high-dimensional and complex problems. Statistical results from the Wilcoxon rank-sum test analysis indicate that EOA not only outperforms other algorithms, but also maintains a balance between exploration and exploitation. Finally, we compare EOA and related engineering application algorithms on six classical engineering problems, which confirms the applicability of EOA in real engineering scenarios.

In future work, we will consider introducing relevant balancing strategies to further harmonize the exploration and exploitation phases of EOA and ensure solution accuracy while achieving overall computational time improvements.

Author Contributions: Conceptualization, S.Z. and Y.S.; methodology, S.Z.; software, Y.S.; validation, S.Z., Y.S., and D.W.; formal analysis, Y.D.; investigation, Y.S. and D.W.; resources, D.W.; data curation, Y.S.; writing—original draft preparation, S.Z. and Y.S.; writing—review and editing, S.Z. and Y.S.; visualization, M.X.; supervision, X.X. and M.X.; project administration, X.X. and D.W.; funding acquisition, X.X. and M.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Key Research and Development Project of Hubei Province under Grant 2023BAB050, Expert Workstations of the Yunnan Province: 202205AF150061,

Key Research and Development Project of Yangzhou under Grant YZ2022013, Wuhan East Lake New Technology Development Zone Key Research and Development Project under Grant 2022KJB129, The Fundamental Research Funds for the Central Universities under Grant 2042023kf1016 and 2042023kf0184, and The Natural Science Foundation of Hubei Province under Grant 2022CFB276 and 2022CFB998.

Data Availability Statement: The data presented in this study are openly available in [12].

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Yao, J.; Sha, Y.; Chen, Y.; Zhao, X. A Novel ensemble of arithmetic optimization algorithm and Harris hawks optimization for solving industrial engineering optimization problems. *Machines* **2022**, *10*, 602. [CrossRef]
2. Alnowibet, K.A.; Mahdi, S.; El-Alem, M.; Abdelawwad, M.; Mohamed, A.W. Guided hybrid modified simulated annealing algorithm for solving constrained global optimization problems. *Mathematics* **2022**, *10*, 1312. [CrossRef]
3. Yang, X.; Wang, R.; Zhao, D.; Yu, F.; Huang, C.; Heidari, A.A.; Cai, Z.; Bourouis, S.; Algarni, A.D.; Chen, H. An adaptive quadratic interpolation and rounding mechanism sine cosine algorithm with application to constrained engineering optimization problems. *Expert Syst. Appl.* **2023**, *213*, 119041. [CrossRef]
4. Dehghani, M.; Montazeri, Z.; Dhiman, G.; Malik, O.; Morales-Menendez, R.; Ramirez-Mendoza, R.A.; Dehghani, A.; Guerrero, J.M.; Parra-Arroyo, L. A spring search algorithm applied to engineering optimization problems. *Appl. Sci.* **2020**, *10*, 6173. [CrossRef]
5. Tian, A.Q.; Liu, F.F.; Lv, H.X. Snow Geese Algorithm: A novel migration-inspired meta-heuristic algorithm for constrained engineering optimization problems. *Appl. Math. Model.* **2024**, *126*, 327–347. [CrossRef]
6. Kalpana, C.; Booba, B. Bio-Inspired Firefly Algorithm A Methodical Survey–Swarm Intelligence Algorithm. In Proceedings of the 2021 International Conference on Computational Intelligence and Computing Applications (ICCICA), Maharashtra, India, 18–19 June 2021; IEEE: Piscataway Township, NJ, USA, 2021; pp. 1–7.
7. Vikhar, P.A. Evolutionary algorithms: A critical review and its future prospects. In Proceedings of the 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), Jalgaon, India, 22–24 December 2016; IEEE: Piscataway Township, NJ, USA, 2016; pp. 261–265.
8. Su, H.; Zhao, D.; Heidari, A.A.; Liu, L.; Zhang, X.; Mafarja, M.; Chen, H. RIME: A physics-based optimization. *Neurocomputing* **2023**, *532*, 183–214. [CrossRef]
9. Naik, A.; Satapathy, S.C. Past present future: A new human-based algorithm for stochastic optimization. *Soft Comput.* **2021**, *25*, 12915–12976. [CrossRef]
10. Marini, F.; Walczak, B. Particle swarm optimization (PSO). A tutorial. *Chemom. Intell. Lab. Syst.* **2015**, *149*, 153–165. [CrossRef]
11. Hashim, F.A.; Houssein, E.H.; Hussain, K.; Mabrouk, M.S.; Al-Atabany, W. Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems. *Math. Comput. Simul.* **2022**, *192*, 84–110. [CrossRef]
12. Abualigah, L.; Diabat, A.; Mirjalili, S.; Abd Elaziz, M.; Gandomi, A.H. The arithmetic optimization algorithm. *Comput. Methods Appl. Mech. Eng.* **2021**, *376*, 113609. [CrossRef]
13. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
14. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
15. Zhao, J.; Gao, Z.M.; Sun, W. The improved slime mould algorithm with Levy flight. *J. Phys. Conf. Ser.* **2020**, *1617*, 012033. [CrossRef]
16. Abualigah, L.; Yousri, D.; Abd Elaziz, M.; Ewees, A.A.; Al-Qaness, M.A.; Gandomi, A.H. Aquila optimizer: A novel meta-heuristic optimization algorithm. *Comput. Ind. Eng.* **2021**, *157*, 107250. [CrossRef]
17. Coello, C.A.C. Use of a self-adaptive penalty approach for engineering optimization problems. *Comput. Ind.* **2000**, *41*, 113–127. [CrossRef]
18. Mezura-Montes, E.; Coello, C.A.C. An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *Int. J. Gen. Syst.* **2008**, *37*, 443–473. [CrossRef]
19. Li, S.; Gu, Q.; Gong, W.; Ning, B. An enhanced adaptive differential evolution algorithm for parameter extraction of photovoltaic models. *Energy Convers. Manag.* **2020**, *205*, 112443. [CrossRef]
20. Ghannadi, P.; Kourehli, S.S.; Mirjalili, S. A review of the application of the simulated annealing algorithm in structural health monitoring (1995–2021). *Frat. Integrità Strutt.* **2023**, *17*, 51–76. [CrossRef]
21. Ali, F.; Sarwar, A.; Bakhsh, F.I.; Ahmad, S.; Shah, A.A.; Ahmed, H. Parameter extraction of photovoltaic models using atomic orbital search algorithm on a decent basis for novel accurate RMSE calculation. *Energy Convers. Manag.* **2023**, *277*, 116613. [CrossRef]
22. Fallah, A.M.; Ghafourian, E.; Shahzamani Sichani, L.; Ghafourian, H.; Arandian, B.; Nehdi, M.L. Novel Neural Network Optimized by Electrostatic Discharge Algorithm for Modification of Buildings Energy Performance. *Sustainability* **2023**, *15*, 2884. [CrossRef]
23. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]

24. Moosavian, N.; Roodsari, B.K. Soccer league competition algorithm: A novel meta-heuristic algorithm for optimal design of water distribution networks. *Swarm Evol. Comput.* **2014**, *17*, 14–24. [CrossRef]
25. Talatahari, S.; Bayzidi, H.; Saraee, M. Social Network Search for Global Optimization. *IEEE Access* **2021**, *9*, 92815–92863. [CrossRef]
26. Askari, Q.; Younas, I.; Saeed, M. Political Optimizer: A novel socio-inspired meta-heuristic for global optimization. *Knowl.-Based Syst.* **2020**, *195*, 105709. [CrossRef]
27. Altunbey, F.; Alatas, B. Overlapping community detection in social networks using parliamentary optimization algorithm. *Int. J. Comput. Netw. Appl.* **2015**, *2*, 12–19.
28. Farshchin, M.; Maniat, M.; Camp, C.V.; Pezeshk, S. School based optimization algorithm for design of steel frames. *Eng. Struct.* **2018**, *171*, 326–335. [CrossRef]
29. Wang, L.Y.; Cao, Q.J.; Zhang, Z.X.; Mirjalili, S.; Zhao, W.G. Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105082. [CrossRef]
30. Li, Y.; Liang, X.; Liu, J.; Zhou, H. Multi-strategy Equilibrium Optimizer: An improved meta-heuristic tested on numerical optimization and engineering problems. *PLoS ONE* **2022**, *17*, e0276210. [CrossRef]
31. Xian, S.; Feng, X. Meerkat optimization algorithm: A new meta-heuristic optimization algorithm for solving constrained engineering problems. *Expert Syst. Appl.* **2023**, *231*, 120482. [CrossRef]
32. Nadimi-Shahraki, M.H.; Taghian, S.; Zamani, H.; Mirjalili, S.; Elaziz, M.A. MMKE: Multi-trial vector-based monkey king evolution algorithm and its applications for engineering optimization problems. *PLoS ONE* **2023**, *18*, e0280006. [CrossRef]
33. Mirjalili, S. SCA: A sine cosine algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133. [CrossRef]
34. Xue, J.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control. Eng.* **2020**, *8*, 22–34. [CrossRef]
35. Alsattar, H.A.; Zaidan, A.; Zaidan, B. Novel meta-heuristic bald eagle search optimisation algorithm. *Artif. Intell. Rev.* **2020**, *53*, 2237–2264. [CrossRef]
36. Liu, J.; Li, D.; Wu, Y.; Liu, D. Lion swarm optimization algorithm for comparative study with application to optimal dispatch of cascade hydropower stations. *Appl. Soft Comput.* **2020**, *87*, 105974. [CrossRef]
37. Abualigah, L.; Abd Elaziz, M.; Sumari, P.; Geem, Z.W.; Gandomi, A.H. Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer. *Expert Syst. Appl.* **2022**, *191*, 116158. [CrossRef]
38. Lu, S.; Kim, H.M. A regularized inexact penalty decomposition algorithm for multidisciplinary design optimization problems with complementarity constraints. *J. Mech. Des. Apr.* **2010**, *13*, 041005. [CrossRef]
39. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [CrossRef]
40. Baykasoğlu, A.; Ozsoydan, F.B. Adaptive firefly algorithm with chaos for mechanical design optimization problems. *Appl. Soft Comput.* **2015**, *36*, 152–164. [CrossRef]
41. Geem, Z.W.; Kim, J.H.; Loganathan, G.V. A new heuristic optimization algorithm: Harmony search. *Simulation* **2001**, *76*, 60–68. [CrossRef]
42. Czerniak, J.M.; Zarzycki, H.; Ewald, D. AAO as a new strategy in modeling and simulation of constructional problems optimization. *Simul. Model. Pract. Theory* **2017**, *76*, 22–33. [CrossRef]
43. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* **2016**, *27*, 495–513. [CrossRef]
44. He, Q.; Wang, L. A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. *Appl. Math. Comput.* **2007**, *186*, 1407–1422. [CrossRef]
45. Ray, T.; Saini, P. Engineering design optimization using a swarm with an intelligent information sharing among individuals. *Eng. Optim.* **2001**, *33*, 735–748. [CrossRef]
46. Sadollah, A.; Bahreininejad, A.; Eskandar, H.; Hamdi, M. Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Appl. Soft Comput.* **2013**, *13*, 2592–2612. [CrossRef]
47. Liu, H.; Cai, Z.; Wang, Y. Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. *Appl. Soft Comput.* **2010**, *10*, 629–640. [CrossRef]
48. Gandomi, A.H.; Yang, X.S.; Alavi, A.H. Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Eng. Comput.* **2013**, *29*, 17–35. [CrossRef]
49. Saremi, S.; Mirjalili, S.; Lewis, A. Grasshopper optimisation algorithm: Theory and application. *Adv. Eng. Softw.* **2017**, *105*, 30–47. [CrossRef]
50. He, Q.; Wang, L. An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Eng. Appl. Artif. Intell.* **2007**, *20*, 89–99. [CrossRef]
51. Ragsdell, K.; Phillips, D. Optimal design of a class of welded structures using geometric programming. *J. Eng. Ind. Aug.* **1976**, *98*, 1021–1025. [CrossRef]
52. Mahdavi, M.; Fesanghary, M.; Damangir, E. An improved harmony search algorithm for solving optimization problems. *Appl. Math. Comput.* **2007**, *188*, 1567–1579. [CrossRef]
53. Abd Elaziz, M.; Oliva, D.; Xiong, S. An improved opposition-based sine cosine algorithm for global optimization. *Expert Syst. Appl.* **2017**, *90*, 484–500. [CrossRef]
54. Kaveh, A.; Khayatazad, M. A new meta-heuristic method: Ray optimization. *Comput. Struct.* **2012**, *112*, 283–294. [CrossRef]

55. Arora, J. *Introduction to Optimum Design*; Elsevier: Amsterdam, The Netherlands, 2004.
56. Mirjalili, S. The ant lion optimizer. *Adv. Eng. Softw.* **2015**, *83*, 80–98. [CrossRef]
57. Chickermane, H.; Gea, H.C. Structural optimization using a new local approximation method. *Int. J. Numer. Methods Eng.* **1996**, *39*, 829–846. [CrossRef]
58. Cheng, M.Y.; Prayogo, D. Symbiotic organisms search: A new metaheuristic optimization algorithm. *Comput. Struct.* **2014**, *139*, 98–112. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.