Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 7695596, 17 pages
https://doi.org/10.1155/2021/7695596

Research Article

A Tent Marine Predators Algorithm with Estimation Distribution Algorithm and Gaussian Random Walk for Continuous Optimization Problems

Chang-Jian Sun[1] **and Fang Gao**[2,3]

[1]*College of Electronic Science and Engineering, Jilin University, Changchun 130012, China*
[2]*College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[3]*Chang Guang Satellite Technology Co., Ltd, Changchun 130000, China*

Correspondence should be addressed to Fang Gao; gaofang@163.com

Received 10 November 2021; Accepted 3 December 2021; Published 28 December 2021

Academic Editor: Ahmed Mostafa Khalil

Copyright © 2021 Chang-Jian Sun and Fang Gao. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The marine predators algorithm (MPA) is a novel population-based optimization method that has been widely used in real-world optimization applications. However, MPA can easily fall into a local optimum because of the lack of population diversity in the late stage of optimization. To overcome this shortcoming, this paper proposes an MPA variant with a hybrid estimation distribution algorithm (EDA) and a Gaussian random walk strategy, namely, HEGMPA. The initial population is constructed using cubic mapping to enhance the diversity of individuals in the population. Then, EDA is adapted into MPA to modify the evolutionary direction using the population distribution information, thus improving the convergence performance of the algorithm. In addition, a Gaussian random walk strategy with medium solution is used to help the algorithm get rid of stagnation. The proposed algorithm is verified by simulation using the CEC2014 test suite. Simulation results show that the performance of HEGMPA is more competitive than other comparative algorithms, with significant improvements in terms of convergence accuracy and convergence speed.

1. Introduction

Solving optimization problems in engineering and scientific research is a common problem. An optimization problem is the process of finding the best value of a decision variable that satisfies the maximum or minimum objective value without violating the constraints. Traditional gradient-based deterministic algorithms show difficulty in solving practical problems [1]. With the development of science and technology today, the optimization problems we encounter are becoming more and more complex. These real-world optimization problems often involve many decision variables, complex nonlinear constraints and nonconvexity, dynamic objective functions, and expensive computational costs [2, 3]. Although these algorithms achieve faster processing speeds, they can easily fall into local optima. In addition, the performance of the algorithms depends heavily on the characteristics of the problem and the initial parameter values. However, metaheuristic algorithms, which do not depend on the characteristics of the problem, are simple in structure, flexible, and do not rely on gradient information and have therefore received widespread attention and flourished among scholars. As a result, it is widely used to solve various optimization problems, such as task planning [4, 5], feature selection [6, 7], parameter optimization [8, 9], and image segmentation [10, 11].

Over the past decades, many metaheuristic algorithms have been proposed. These algorithms can be divided into three categories: evolution-based algorithms, physics-based algorithms, and swarm-based algorithms. Evolutionary-based algorithms are a class of algorithms that simulate the laws of evolution in nature. Genetic algorithm (GA) [12] is a

widely used evolution-based algorithm proposed by Holland. It updates populations by simulating the natural law of superiority and inferiority. With the popularity of GA and GA variants, more and more evolutionary-based algorithms have been proposed, including differential evolution (DE) [13], genetic programming (GP) [14], and evolutionary strategy (ES) [15]. In addition to these evolutionary algorithms, new evolutionary-based algorithms have recently been proposed, such as the artificial algae algorithm (AAA) [16] and monkey king evolutionary (MKE) [17]. Physics-based algorithms simulate the laws of physics in nature or in the universe. Inspired by the phenomenon of annealing in metallurgy, simulated annealing (SA) [18] is one of the best-known physics-based algorithms. Other physics-based algorithms have been proposed, including the gravitational search algorithm (GSA) [19], nuclear reaction optimization (NRO) [20], water cycle algorithm (WCA) [21], and sine cosine algorithm (SCA) [22]. Population-based algorithms simulate the social behaviour of species such as self-organisation and division of labour. Particle swarm optimization (PSO) [23] and ant colony optimization (ACO) [24] are two classical swarm-based algorithms. Inspired by these two algorithms, an increasing number of scholars have studied this topic and proposed different swarm-based algorithms such as grey wolf optimizer (GWO) [25], whale optimization algorithm (WOA) [26], sparrow search algorithm (SSA) [27], firefly algorithm (FA) [28], artificial bee colony algorithm (ABC) [29], and tuna swarm optimization (TSO) [30].

The marine predators algorithm (MPA) is a novel population-based natural heuristic optimization algorithm proposed by Faramarzi et al. [31], which is mainly inspired by the different foraging strategies of marine predators and the optimal encounter rate strategies between predators and prey. Simulation results in the literature [31] show that MPA has better performance compared to GA, PSO, GSA, CS, SSA, and CMA-ES and thus has been widely used to tackle many practical engineering problems such as photovoltaics [32, 33], power systems [34], image classification [35], and task scheduling [36].

Although MPA has been applied in several fields, there is less analysis and improvement on the shortcomings of MPA. MPA mainly searches near the optimal individuals when performing population position update, without using the effective information of more individuals. The insufficient diversity leads the MPA into a local optimum. The FADs process is designed to enhance the diversity of the population, but it does not determine the superiority of the offspring and the parent, which is not conducive to the optimization and convergence of the algorithm to a certain extent.

Currently, there are three main approaches to improve the performance of natural heuristic optimization algorithms. The first is parameter tuning. Tang et al. [37] used chaos mapping to optimise key parameters of sparrow search algorithm. Ewees and Elaziz [38] investigated the effect of different chaotic mapping tools on the parameter settings of the algorithm. The second approach is to design different search operators. Zhang et al. [39] used a triangular variational strategy and a logarithmic spiral strategy to improve the development and exploration performance of the algorithm. Nor et al. [40] proposed an adaptive switching particle swarm algorithm based on a hybrid update sequence. The third approach is to introduce other techniques. The fractional order is an effective tool that has been used in other areas [41, 42]. Deep neural networks can also be combined [43, 44]. In addition, traditional gradient-based methods can be combined with metaheuristic algorithms. Inspired by these ideas, this paper proposes a hybrid MPA combined with the estimation of distribution algorithm for improving the basic MPA performance. And, we use Tent mapping and Gaussian random walk to further improve performance. The performance of HEGMPA was evaluated on the CEC2014 test suite and compared with five advanced algorithms. The superiority of the proposed algorithm is verified by numerical analysis, convergence analysis, stability analysis, and statistical analysis.

The left part of this paper is organized as follows: A review of the MPA is presented in Section 2. Section 3 shows a detailed description of the proposed algorithm. In Section 4, the effectiveness of the proposed improvement strategy is verified using CEC2014 test suite. Finally, we summarize this work in Section 4 and present directions for future research.

2. Marine Predators Algorithm (MPA)

In this section, the procedure of the basic MPA is presented. MPA is a novel swarm-based metaheuristic algorithm mainly inspired by the different foraging strategies of marine predators and the optimal encounter rate strategy between the predator and prey.

Similar to most metaheuristics, the initial solution of MPA is distributed as evenly as possible in the search space. The initialization formula is as follows:

$$X(1) = X_{\min} + r_1 \times (X_{\max} - X_{\min}), \tag{1}$$

where $X_{\min}$ and $X_{\max}$ denote the upper and lower bounds of the population variables, respectively. $r_1 \in (0, 1)$ is a random vector obeying a uniform distribution.

The MPA search process is divided into three phases based on different speed ratios: (1) a high-speed phase, where the prey speed is faster than the predator speed; (2) a unit speed ratio phase, where the prey speed and the predator speed are similar; and (3) a low-speed phase, where the prey speed is slower than the predator speed. In each stage, the movement of the predator and prey in nature is imitated separately.

Phase 1. The high-speed phase occurs at the beginning of the iteration, when the prey obeys Brownian motion and the predator performs mainly exploratory behaviour. The mathematical model for this phase is shown as follows:

$$\text{while iter} < \frac{1}{3}\text{iter}_{\max},$$

$$\overrightarrow{\mathbf{stepsize}_i} = \overrightarrow{\mathbf{R}_B} \otimes \left(\overrightarrow{\mathbf{Elite}_i} - \overrightarrow{\mathbf{R}_B} \otimes \overrightarrow{\mathbf{Prey}_i}\right), \quad i = 1, \ldots, n, \tag{2}$$

$$\overrightarrow{\mathbf{Prey}_i} = \overrightarrow{\mathbf{Prey}_i} + P \cdot \vec{\mathbf{R}} \otimes \overrightarrow{\mathbf{stepsize}_i}, \tag{3}$$

where $\overrightarrow{\mathbf{R}_B}$ is a random vector that follows a normal distribution based on Brownian motion. P is a constant taking the value 5. $\vec{\mathbf{R}} \in (0, 1)$ is a uniformly distributed random vector. iter denotes the number of current iterations. $\text{iter}_{\max}$ denotes the maximum number of iterations.

Phase 2. In the second stage, both exploitation and exploration are required, so the stock is divided into two parts. One part is used for exploitation, and the other for exploration. The prey is used for the exploitation process and the predator for the exploration process. The mathematical model of this stage is described as follows:

$$\text{while } \frac{1}{3}\text{iter}_{\max} \le \text{iter} < \frac{2}{3}\text{iter}_{\max}. \tag{4}$$

The first part of the population carries out exploitation behaviour.

$$\overrightarrow{\mathbf{stepsize}_i} = \overrightarrow{\mathbf{R}_L} \otimes \left(\overrightarrow{\mathbf{Elite}_i} - \overrightarrow{\mathbf{R}_L} \otimes \overrightarrow{\mathbf{Prey}_i}\right), \quad i = 1, \ldots, n, \tag{5}$$

$$\overrightarrow{\mathbf{Prey}_i} = \overrightarrow{\mathbf{Prey}_i} + P \cdot \vec{\mathbf{R}} \otimes \overrightarrow{\mathbf{stepsize}_i}, \tag{6}$$

where $\overrightarrow{\mathbf{R}_L}$ is a random vector obeying a Lévy distribution.

The second part of the population performs exploratory behaviour.

$$\overrightarrow{\mathbf{stepsize}_i} = \overrightarrow{\mathbf{R}_B} \otimes \left(\overrightarrow{\mathbf{R}_B} \otimes \overrightarrow{\mathbf{Elite}_i} - \overrightarrow{\mathbf{Prey}_i}\right), \quad i = 1, \ldots, n, \tag{7}$$

$$\overrightarrow{\mathbf{Prey}_i} = \overrightarrow{\mathbf{Elite}_i} + P \cdot \text{CF} \otimes \overrightarrow{\mathbf{stepsize}_i}, \tag{8}$$

where $\text{CF} = (1 - (\text{iter}/\text{iter}_{\max}))^{2(\text{iter}/\text{iter}_{\max})}$ is an adaptive parameter that controls the predator step size.

Phase 3. As the last phase, the exploitation process is mainly carried out and the mathematical model of this phase is described as follows:

$$\text{while iter} > \frac{2}{3}\text{iter}_{\max}, \tag{9}$$

$$\overrightarrow{\mathbf{stepsize}_i} = \overrightarrow{\mathbf{R}_L} \otimes \left(\overrightarrow{\mathbf{R}_L} \otimes \overrightarrow{\mathbf{Elite}_i} - \overrightarrow{\mathbf{Prey}_i}\right),$$

$$\overrightarrow{\mathbf{Prey}_i} = \overrightarrow{\mathbf{Elite}_i} + P \cdot \text{CF} \otimes \overrightarrow{\mathbf{stepsize}_i}. \tag{10}$$

In addition, environmental issues can cause changes in predator behaviour, and fish aggregating devices (FADs) are a factor that affects predator behaviour and are seen as a local optimum in this search area, assuming that the local optimum can be jumped out of by longer steps. The effect of FADs can be expressed mathematically as follows:

$$\overrightarrow{\mathbf{Prey}_i} = \begin{cases} \overrightarrow{\mathbf{Prey}_i} + \text{CF}\left[\overrightarrow{\mathbf{X}_{\min}} + \vec{\mathbf{R}} \otimes \left(\overrightarrow{\mathbf{X}_{\max}} - \overrightarrow{\mathbf{X}_{\min}}\right)\right] \otimes \vec{U}, & \text{if } r \le \text{FADs}, \\ \overrightarrow{\mathbf{Prey}_i} + [\text{FADs}(1 - r) + r]\left(\overrightarrow{\mathbf{Prey}_{r1}} - \overrightarrow{\mathbf{Prey}_{r2}}\right), & \text{if } r > \text{FADs}, \end{cases} \tag{11}$$

where FADs = 0.2 denotes the probability that FADs affect the optimization process. $\vec{U}$ is a binary vector including 0 or 1. When a random vector from 0 to 1 is generated and is less than 0.2, the array is changed to 0, and vice versa. $r \in (0, 1)$ is a uniformly distributed random number. $\overrightarrow{\mathbf{Prey}_{r1}}$ and $\overrightarrow{\mathbf{Prey}_{r2}}$ are two randomly selected individuals. The pseudo-code for the MPA is shown in Algorithm 1 and Figure 1.

3. The Proposed MPA Variant

The basic MPA uses only the best individuals for iterative search, not making full use of valid information from the remaining individuals, resulting in reduced population diversity. The FADs process is performed at each iteration, which increases the computational cost. We use three strategies to improve the performance of the algorithm. Firstly, we take advantage of the good traversal and randomness of chaotic mapping to generate the initial solution of the population and increase the population diversity. Secondly, we use EDA to sample the dominant population information and correct the evolutionary direction. A Gaussian wandering strategy is used to enhance the population diversity when the algorithm stalls, helping the algorithm to jump out of the local optimum. Finally, a greedy strategy is used to ensure that the algorithm converges efficiently.

3.1. Population Initialization Based on Cubic Mapping. The initial population of most current intelligent optimization algorithms is randomly generated in the search space, and the quality of the initialized population has a great impact on the efficiency of the optimization algorithm. A uniformly distributed population is conducive to expanding the search range and thus improving the convergence speed and accuracy of the algorithm.

(1) Initialize search agents (Prey) populations
(2) While termination criteria are not met
(3) Calculate the fitness, construct the Elite matrix, and accomplish memory saving
(4) if iter < $iter_{max}/3$
(5) Update prey based on equations (2) and (3)
(6) else if $iter_{max}/3$ < iter < $2 \times iter_{max}/3$
(7) For the first half of the populations ($i = 1, \ldots, NP/2$)
(8) Update prey based on equation (6)
(9) For the other half of the populations
(10) Update prey based on equation (8)
(11) else if iter > $2 \times iter_{max}/3$
(12) Update prey based on equation (10)
(13) end if
(14) Accomplish memory saving and Elite update
(15) Applying FADs effect and update based on equation (11)
(16) end while

Algorithm 1: Pseudo-code of MPA.

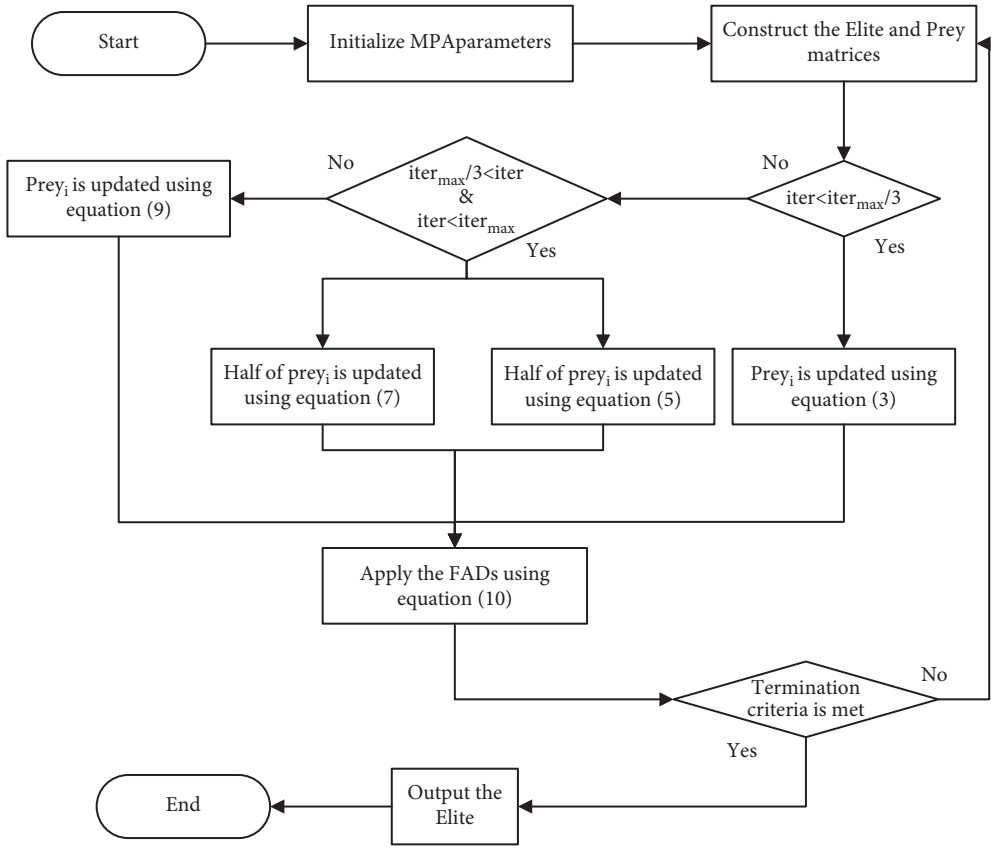

Figure 1: Flow chart of MPA.

MPA, like other intelligent algorithms, suffers from a reduction in population diversity late in the iteration when solving complex problems, which can easily fall into local optima leading to premature maturity resulting in poor convergence accuracy. To improve its global search capability and avoid the reduction of population diversity in the postiteration period, the chaos operator is used to initialize the population, considering that it has the characteristics of randomness and regularity and can traverse all states within a certain range without repetition, so the cubic mapping chaos operator is used. Figure 2 shows the effect of cubic mapping and logistic mapping.

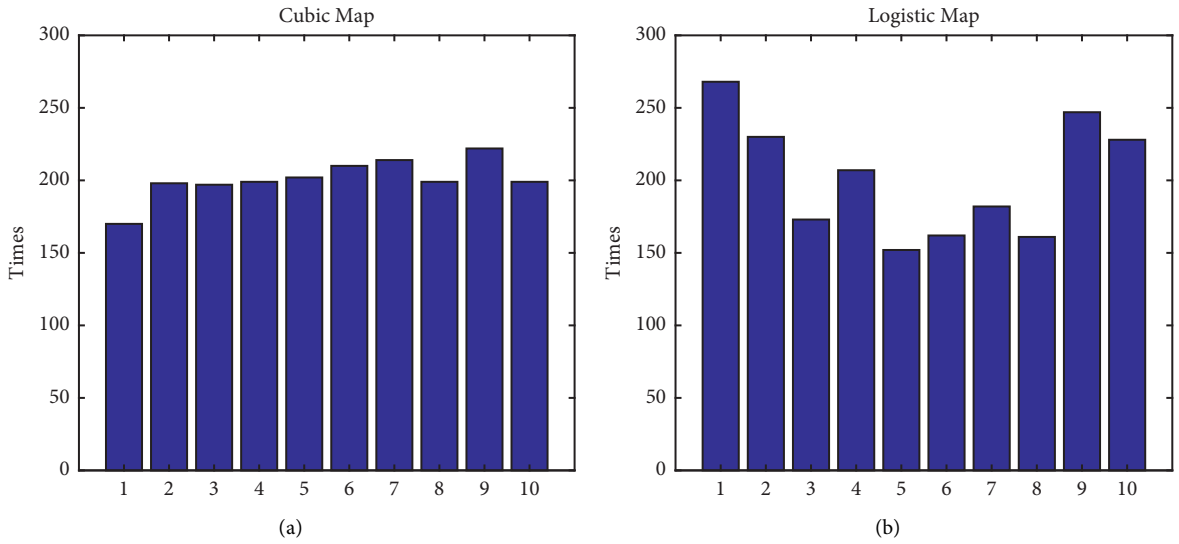

Figure 2: Mapping contrast diagram. (a) Logistic mapping values. (b) Cubic mapping values.

The cubic mapping formula is shown as follows:

$$y(n+1) = 4y(n)^3 - 3y(n), \quad -1 < y(n) < 1, y(n) \neq 0, n = 0, 1, 2, \ldots. \tag{12}$$

The cubic mapping is used to initialize the prey population by generating a random vector of −1 to 1 in each dimension as the first individual, then iterating over each dimension of the first individual to obtain the remaining $M-1$ individuals using equation (12), and finally mapping the values of the variables generated by the cubic mapping onto the prey individuals using

$$\text{Prey}_{i,j} = X_{\min} + \left(X_{\max} - X_{\min}\right) \times \frac{\left(y_{ij} + 1\right)}{2}. \tag{13}$$

3.2. Estimation of Distribution Algorithm. The estimation of distribution algorithm is an algorithm that uses probability models to represent relationships between individuals. EDA has been employed for hybridization with other algorithms and has achieved better results [45, 46]. It uses the current dominant population to calculate a probability distribution model and generates a new population of children based on the probability distribution model sampling, thus iterating continuously to eventually obtain the optimal solution. In this paper, the distribution model is estimated using a weighted maximum likelihood estimation method and the top one-half population with better performance is taken as the dominant population. The mathematical model of the algorithm is described as follows:

$$\overrightarrow{\mathbf{Prey}_{\text{mean}}} = \sum_{i=1}^{\text{NP}/2} \omega_i \times \overrightarrow{\mathbf{Prey}_i}, \tag{14}$$

$$\omega_i = \frac{\ln(\text{NP}/2 + 0.5) - \ln(i)}{\sum_{i=1}^{\text{NP}/2} (\ln(\text{NP}/2 + 0.5) - \ln(i))}, \tag{15}$$

$$\mathbf{Cov}(i) = \frac{1}{\text{NP}/2} \sum_{i=1}^{\text{NP}/2} \left(\overrightarrow{\mathbf{Prey}_i} - \overrightarrow{\mathbf{Prey}_{\text{mean}}}\right) \times \left(\overrightarrow{\mathbf{Prey}_i} - \overrightarrow{\mathbf{Prey}_{\text{mean}}}\right)^T, \tag{16}$$

$$\overrightarrow{\mathbf{Prey}_i} = \text{Gaussian}\left(\overrightarrow{\mathbf{Prey}_{\text{mean}}}, \mathbf{Cov}(i)\right) + \text{rand} \times \left(\overrightarrow{\mathbf{Prey}_{\text{mean}}} - \overrightarrow{\mathbf{Prey}_i}\right), \tag{17}$$

where $\overrightarrow{\mathbf{Prey}}_{\text{mean}}$ denotes the weighted mean of the dominant population. NP is the population size. ω_i denotes the weight coefficients in the dominant population in the descending order of fitness values. **Cov** is the weighted covariance matrix of the dominant population.

3.3. Medium Solution Gaussian Random Walk. A random walk strategy is used to help the algorithm jump out of stagnation and enhance its exploration capabilities when it falls into a local optimum late in the iteration. A stalled algorithm is considered to have stalled if the average fitness of the top half of the dominant population does not change in two consecutive iterations, and the random walk strategy is then used to update the population. The random walk strategy is a probabilistic model that simulates the random movement of organisms in nature and is widely used in the design and improvement of various optimization algorithms. In this paper, we propose a Gaussian random walk strategy for constructing new offspring using a medium population.

As the vector of differences between dominant and intermediate populations can improve the diversity of populations, information on intermediate populations is considered in this paper. The sampling points are related to the relative positions of the medium and dominant populations, extending the search range and providing a strong exploration capability. The mathematical model of the strategy is described as follows:

$$\begin{aligned}\overrightarrow{\mathbf{Prey}}_i &= \text{Gaussian}\left(\overrightarrow{\mathbf{Prey}}_i, \sigma\right) + \text{rand} \times \overrightarrow{\mathbf{Prey}}_{r_4}^{\text{best}} \\ &\quad - \text{rand} \times \overrightarrow{\mathbf{Prey}}_{r_3}^{\text{medium}}, \\ \sigma &= \left|\overrightarrow{\mathbf{Prey}}_{r_4}^{\text{best}} - \overrightarrow{\mathbf{Prey}}_{r_3}^{\text{medium}}\right|,\end{aligned} \tag{18}$$

where $\overrightarrow{\mathbf{Prey}}^{\text{best}}$ and $\overrightarrow{\mathbf{Prey}}^{\text{medium}}$ represent randomly selected individuals from the dominant and intermediate populations, respectively.

At the end of each iteration, HEGMPA used a greedy strategy to retain the best NP individuals in the parent and offspring, thus forming a new population, which facilitated the global convergence ability of HEGMPA. In summary, the flow chart of the improved algorithm proposed in this paper is shown in the following. The pseudo-code for the HEGMPA is shown in Algorithm 2.

4. Simulation Experiments and Analysis of Results

To comprehensively validate the performance of the improved algorithms, we first verify the effectiveness of different improvement strategies and then verify the superiority and competitiveness of the improved algorithms by comparing them with recently proposed ones.

The CEC2014 test suite contains 30 test functions, which can be divided into four categories of test functions according to different characteristics: F1–F3 for single-peaked functions, F4–F16 for multipeaked functions, F17–F22 for mixed functions, and F23–F30 for combined functions. The definitions and optimal values of the functions are shown in Table 1. In the CEC2014 test, the maximum number of iterations is 600 and the population size is 500. All algorithms were run independently 51 times to record statistical values. The program was run on a MATLAB 2016b platform.

4.1. Comparison of HEGMPA Improvement Strategies. HEGMPA was compared with MPA-1 using cubic mapping to initialize the population, MPA-2 with fused EDA, MPA-3 using a moderately solved Gaussian random wandering strategy, and basic MPA to verify the effectiveness of the different improvement strategies. Table 2 records the mean error, standard deviation, and ranking of the different algorithms for solving the test functions. The last column shows the average ranking of each algorithm.

From Table 2, HEGMPA with a full search strategy has the best search performance, while the basic MPA ranks last. Specifically, for unimodal test functions F1–F3, MPA-2, which only incorporates the EDA algorithm, performs similarly to HEGMPA in unimodal functions and far outperforms the other compared algorithms, indicating that incorporating the EDA strategy can effectively improve the development capability of the algorithm. For the multimodal functions F4–F16, HEGMPA and EDA-2 also ranked in the top two positions, while MPA was the least effective, suggesting that using dominant population information to generate offspring is beneficial in enhancing the diversity of individuals in the population. For the combinatorial functions F17–F30, HEGMPA only underperformed MPA-2 on F21, F24, and F26, indicating that the Tent chaotic mapping and moderately solved Gaussian random walk strategies can improve the algorithm's performance in solving complex combinatorial functions and effectively help the algorithm to jump out of the local optimum when the algorithm stalls. In summary, the improvement strategy proposed in this paper can effectively improve the MPA optimality finding performance.

4.2. An Analysis of HEGMPA Compared with Other Algorithms. To further illustrate the superiority of the improved algorithms, five algorithms, TLMPA, VCS [47], MMPA [48], CPIJADE [49], and HFPSO [50], are selected for comparison with HEGMPA. CPIJADE is an improved JADE algorithm using a new framework. HFPSO is an improved particle swarm algorithm mixed with the firefly algorithm. To ensure fairness, the parameters of each algorithm are referred to the original literature, as shown in Table 3. In the experiments, NP = 500, dim = 30, and the maximum number of evaluations is 300,000. Table 4 records the average error and ranking of each algorithm in each test function for 51 independent runs.

The analysis in Table 4 shows that for single-peaked test functions F1–F3, HEGMPA outperforms all the comparative algorithms and can consistently find the optimal values of these three test functions, demonstrating the

(1) Initialize search agents (Prey) populations
(2) While termination criteria are not met
(3) Calculate the fitness, construct the Elite matrix and accomplish memory saving
(4) if stagnation
(5) Update prey based on equation (18)
(4) else if rand < 0.5
(6) if iter < $\text{iter}_{\max}/3$
(7) Update prey based on equations (2) and (3)
(8) else if $\text{iter}_{\max}/3 < \text{iter} < 2 \times \text{iter}_{\max}/3$
(9) For the first half of the populations ($i = 1, \ldots, \text{NP}/2$)
(10) Update prey based on equation (6)
(11) For the other half of the populations
(12) Update prey based on equation (8)
(13) else if iter > $2 \times \text{iter}_{\max}/3$
(14) Update prey based on equation (10)
(15) end if
(16) else
(17) Update prey based on equation (17)
(18) Accomplish memory saving and Elite update
(19) end while

Algorithm 2: Pseudo-code of HEGMPA.

Table 1: Descriptions of CEC2014 test suite.

Type	No.	Function name	$F_i^* = F_i(x^*)$
Unimodal functions	1	Rotated high conditioned elliptic function	100
	2	Rotated bent cigar function	200
	3	Rotated discus function	300
Multimodal functions	4	Shifted and rotated Rosenbrock's function	400
	5	Shifted and rotated Ackley's function	500
	6	Shifted and rotated Weierstrass function	600
	7	Shifted and rotated Griewank's function	700
	8	Shifted Rastrigin's function	800
	9	Shifted and rotated Rastrigin's function	900
	10	Shifted Schwefel's function	1000
	11	Shifted and rotated Schwefel's function	1100
	12	Shifted and rotated Katsuura function	1200
	13	Shifted and rotated HappyCat function	1300
	14	Shifted and rotated HGBat function	1400
	15	Shifted and rotated expanded Griewank's plus Rosenbrock's function	1500
	16	Shifted and rotated expanded Schaffer's F6 function	1600
Hybrid functions	17	Hybrid function 1 ($N = 3$)	1700
	18	Hybrid function 2 ($N = 3$)	1800
	19	Hybrid function 3 ($N = 4$)	1900
	20	Hybrid function 4 ($N = 4$)	2000
	21	Hybrid function 5 ($N = 5$)	2100
	22	Hybrid function 6 ($N = 5$)	2200
Composite functions	23	Composition function 1 ($N = 5$)	2300
	24	Composition function 2 ($N = 3$)	2400
	25	Composition function 3 ($N = 3$)	2500
	26	Composition function 4 ($N = 5$)	2600
	27	Composition function 5 ($N = 5$)	2700
	28	Composition function 6 ($N = 5$)	2800
	29	Composition function 7 ($N = 3$)	2900
	30	Composition function 8 ($N = 3$)	3000
Search range: $[-100,100]^D$			

Table 2: Statistics of the results of five algorithms in CEC2014 30D testing.

Function	MPA			MPA-1			MPA-2			MPA-3			HEGMPA		
	Mean	Std	Rank	Mean	Std	Rank	Mean	Std	Rank	Mean	Std	Rank	Mean	Std	Rank
F1	6.02E+07	3.69E+07	5	5.08E+07	3.45E+07	4	0.00E+00	0.00E+00	1	1.44E+07	8.57E+06	3	0.00E+00	0.00E+00	1
F2	3.86E+09	3.19E+09	5	3.67E+09	2.61E+09	4	0.00E+00	0.00E+00	1	9.32E+07	5.60E+08	3	0.00E+00	0.00E+00	1
F3	3.72E+04	1.14E+04	5	3.20E+04	1.03E+04	4	0.00E+00	0.00E+00	1	4.84E+03	3.14E+03	3	0.00E+00	0.00E+00	1
F4	4.10E+02	1.63E+02	4	4.24E+02	1.61E+02	5	2.01E−14	9.15E−14	1	1.56E+02	3.80E+01	3	6.02E−14	1.94E−13	2
F5	2.00E+01	4.53E−02	4	2.00E+01	4.77E−02	5	2.00E+01	1.70E−02	2	2.00E+01	3.82E−02	3	2.00E+01	5.64E−03	1
F6	2.58E+01	2.53E+00	5	2.50E+01	2.65E+00	4	2.79E−05	9.58E−05	1	1.57E+01	2.23E+00	3	3.00E−02	2.10E−01	2
F7	3.21E+01	2.01E+01	5	2.99E+01	1.87E+01	4	0.00E+00	0.00E+00	1	6.45E−01	4.73E−01	3	0.00E+00	0.00E+00	1
F8	1.27E+02	2.53E+01	5	1.22E+02	2.78E+01	4	1.38E+01	5.45E+00	2	1.52E+01	7.02E+00	3	1.35E+01	5.94E+00	1
F9	1.62E+02	2.59E+01	5	1.56E+02	3.04E+01	4	3.49E+01	8.50E+00	1	1.12E+02	1.66E+01	3	3.74E+01	1.09E+01	2
F10	2.09E+03	6.43E+02	5	1.83E+03	4.49E+02	4	3.53E+02	1.80E+02	2	2.99E+02	1.82E+02	1	3.79E+02	1.87E+02	3
F11	3.34E+03	5.12E+02	5	3.17E+03	5.23E+02	4	2.45E+03	5.18E+02	2	2.16E+03	4.71E+02	1	2.46E+03	4.13E+02	3
F12	1.76E−01	7.58E−02	4	1.84E−01	9.38E−02	5	1.13E−02	7.21E−03	1	6.78E−02	3.21E−02	3	1.36E−02	1.04E−02	2
F13	9.15E−01	3.99E−01	4	9.32E−01	3.70E−01	5	1.68E−01	3.62E−02	2	4.33E−01	8.89E−02	3	1.66E−01	3.27E−02	1
F14	5.12E+00	6.14E+00	5	4.23E+00	5.98E+00	4	2.18E−01	3.74E−02	2	2.64E−01	7.52E−02	3	2.05E−01	4.06E−02	1
F15	1.73E+03	2.53E+03	5	1.25E+03	1.78E+03	4	3.01E+00	7.14E−01	1	1.11E+01	4.99E+00	3	3.06E+00	8.33E−01	2
F16	1.17E+01	5.54E−01	4	1.18E+01	4.79E−01	5	9.98E+00	5.98E−01	2	1.05E+01	4.45E−01	3	9.75E+00	5.74E−01	1
F17	1.05E+06	8.70E+05	4	1.24E+06	1.13E+06	5	1.58E+02	1.35E+02	2	2.32E+05	1.36E+05	3	1.51E+02	1.16E+02	1
F18	6.31E+03	6.69E+03	5	5.34E+03	4.39E+03	4	2.20E+01	7.42E+00	2	1.20E+03	4.22E+02	3	2.18E+01	6.38E+00	1
F19	3.77E+01	2.06E+01	5	3.50E+01	2.08E+01	4	2.11E+00	6.50E−01	2	1.63E+01	1.28E+01	3	2.09E+00	6.36E−01	1
F20	1.09E+04	6.82E+03	5	1.07E+04	6.90E+03	4	6.96E+00	2.48E+00	2	4.58E+03	3.85E+03	3	6.79E+00	2.45E+00	1
F21	2.94E+05	2.69E+05	4	3.15E+05	2.92E+05	5	8.89E+01	1.01E+02	1	9.14E+04	4.73E+04	3	1.08E+02	1.32E+02	2
F22	4.90E+02	1.89E+02	4	5.08E+02	2.00E+02	5	1.99E+02	9.35E+01	2	3.76E+02	1.40E+02	3	1.58E+02	3.03E+01	1
F23	3.54E+02	1.45E+01	4	3.56E+02	1.77E+01	5	3.02E+02	3.75E+01	2	3.17E+02	1.25E+00	3	2.75E+02	5.56E+01	1
F24	2.70E+02	9.95E+00	5	2.57E+02	9.11E+00	4	2.00E+02	2.26E−07	1	2.00E+02	4.02E−05	3	2.00E+02	2.56E−07	2
F25	2.16E+02	4.76E+00	5	2.16E+02	3.91E+00	4	2.02E+02	7.02E−01	2	2.00E+02	2.85E−13	1	2.02E+02	5.08E−01	3
F26	1.01E+02	2.88E−01	2	1.03E+02	1.41E+01	3	1.00E+02	2.98E−02	1	1.06E+02	2.37E+01	4	1.14E+02	3.47E+01	5
F27	4.38E+02	2.12E+01	4	4.49E+02	3.29E+01	5	3.29E+02	4.60E+01	2	4.06E+02	2.70E+00	3	3.08E+02	3.37E+01	1
F28	1.32E+03	2.27E+02	5	1.30E+03	2.15E+02	4	7.82E+02	1.45E+02	2	1.13E+03	3.09E+02	3	7.37E+02	1.70E+02	1
F29	1.48E+04	4.26E+03	4	4.52E+05	2.23E+06	5	1.23E+02	2.63E+01	2	2.52E+03	1.02E+03	3	1.21E+02	2.49E+01	1
F30	2.99E+04	2.69E+04	4	3.12E+04	2.62E+04	5	4.14E+02	2.97E+01	2	6.65E+03	2.77E+03	3	4.12E+02	4.33E+01	1
Average rank		4.50			4.37			1.60			2.83			1.56	

Table 3: Algorithm parameter setting.

Algorithm	Parameter setting
TLMPA	$r_3 \in (0,1), r_4 \in (0,1), r_5 \in (0,1)$
VCS	$\lambda = 0.5, \sigma = 0.3$
HFPSO	$c_1 = c_2 = 1.49445, a = 0.2, B_0 = 2$ $\gamma = 1, w_i = 0.9, w_f = 0.5$
MMPA	$jr = 0.3, w = 10$
CPI-JADE	$Cr = 0.5, F = 0.5, c = 0.1, p = 0.5$

Table 4: Statistics of the results of six algorithms in CEC2014 30D testing.

Function	TLMPA		VCS		HFPSO		MMPA		CPIJADE		HEGMPA	
	Mean	Rank	Mean	Rank	Mean	Rank	Mean	Rank	Mean	Rank	Mean	Rank
F1	$9.88E-03$	3	$6.12E+06$	6	$8.59E+05$	4	$8.82E+05$	5	$1.10E-04$	2	$0.00E+00$	1
F2	$3.18E-04$	3	$4.90E+05$	6	$1.76E+03$	5	$4.96E+02$	4	$1.57E-09$	2	$0.00E+00$	1
F3	$1.27E-05$	3	$2.77E+03$	6	$1.94E+02$	5	$3.50E+00$	4	$5.74E-11$	2	$0.00E+00$	1
F4	$5.62E-04$	2	$4.61E+01$	4	$4.83E+01$	5	$5.78E+01$	6	$6.21E-01$	3	$1.56E-14$	1
F5	$2.09E+01$	6	$2.07E+01$	4	$2.01E+01$	2	$2.09E+01$	5	$2.06E+01$	3	$2.00E+01$	1
F6	$1.09E-01$	2	$1.59E+01$	5	$8.20E+00$	3	$1.73E+01$	6	$1.24E+01$	4	$1.98E-05$	1
F7	$1.60E-09$	3	$6.85E-01$	6	$1.30E-02$	4	$1.62E-02$	5	$0.00E+00$	1	$0.00E+00$	1
F8	$4.11E+01$	3	$5.81E+01$	5	$5.64E+01$	4	$7.35E+01$	6	$2.83E+01$	2	$1.43E+01$	1
F9	$4.71E+01$	2	$1.26E+02$	6	$7.58E+01$	3	$8.49E+01$	5	$7.93E+01$	4	$3.60E+01$	1
F10	$1.75E+03$	4	$2.09E+03$	5	$1.04E+03$	2	$2.93E+03$	6	$1.17E+03$	3	$4.03E+02$	1
F11	$2.90E+03$	3	$4.13E+03$	5	$2.73E+03$	2	$3.48E+03$	4	$4.14E+03$	6	$2.49E+03$	1
F12	$1.42E+00$	5	$1.23E-01$	2	$2.81E-01$	3	$1.82E+00$	6	$9.10E-01$	4	$1.36E-02$	1
F13	$2.49E-01$	2	$4.03E-01$	6	$2.88E-01$	4	$3.65E-01$	5	$2.51E-01$	3	$1.60E-01$	1
F14	$2.52E-01$	4	$3.07E-01$	6	$3.07E-01$	5	$2.35E-01$	2	$2.39E-01$	3	$2.05E-01$	1
F15	$3.82E+00$	2	$1.15E+01$	6	$4.25E+00$	3	$7.97E+00$	4	$8.72E+00$	5	$3.01E+00$	1
F16	$1.01E+01$	2	$1.16E+01$	5	$1.08E+01$	3	$1.16E+01$	6	$1.13E+01$	4	$9.84E+00$	1
F17	$2.56E+02$	2	$3.83E+05$	6	$1.73E+05$	5	$5.10E+03$	4	$7.58E+02$	3	$1.25E+02$	1
F18	$3.95E+01$	2	$1.50E+04$	6	$3.67E+03$	5	$2.44E+02$	4	$4.25E+01$	3	$2.26E+01$	1
F19	$2.75E+00$	2	$1.32E+01$	6	$1.01E+01$	4	$1.07E+01$	5	$4.48E+00$	3	$2.10E+00$	1
F20	$1.80E+01$	3	$5.96E+03$	6	$8.32E+02$	5	$1.78E+02$	4	$1.67E+01$	2	$5.90E+00$	1
F21	$2.51E+02$	2	$1.58E+05$	6	$8.22E+04$	5	$2.78E+03$	4	$4.21E+02$	3	$9.34E+01$	1
F22	$1.99E+02$	3	$3.14E+02$	6	$2.91E+02$	5	$2.13E+02$	4	$1.07E+02$	1	$1.74E+02$	2
F23	$2.00E+02$	3	$2.00E+02$	1	$3.14E+02$	5	$2.00E+02$	1	$3.15E+02$	6	$2.81E+02$	4
F24	$2.00E+02$	4	$2.00E+02$	1	$2.23E+02$	6	$2.00E+02$	2	$2.22E+02$	5	$2.00E+02$	3
F25	$2.02E+02$	3	$2.00E+02$	1	$2.04E+02$	6	$2.00E+02$	1	$2.03E+02$	5	$2.02E+02$	4
F26	$1.00E+02$	2	$1.00E+02$	5	$1.02E+02$	6	$1.00E+02$	4	$1.00E+02$	3	$1.00E+02$	1
F27	$3.65E+02$	5	$2.16E+02$	2	$4.65E+02$	6	$2.08E+02$	1	$3.59E+02$	4	$3.25E+02$	3
F28	$5.51E+02$	4	$2.21E+02$	2	$4.09E+02$	3	$2.00E+02$	1	$8.44E+02$	6	$7.87E+02$	5
F29	$1.41E+02$	2	$2.07E+02$	3	$5.31E+02$	4	$2.64E+04$	6	$7.16E+02$	5	$1.16E+02$	1
F30	$4.34E+02$	2	$4.72E+02$	3	$7.67E+02$	4	$1.95E+03$	6	$9.30E+02$	5	$4.15E+02$	1
Average rank	2.93		4.56		4.2		4.2		3.5		1.5	

advantage of HEGMPA in solving pathological functions and once again verifying that the improved strategy can effectively improve the development capability of the algorithm. For the multipeaked test functions F4–F16, HEGMPA also outperforms all the comparative algorithms, indicating that the improved algorithms can maintain good enough population diversity to avoid falling into local optima. For complex combinatorial functions, each algorithm has its own advantages and disadvantages. HEGMPA obtains optimal solutions on F17–F22, F26, and F29–F30. VCS achieves better solutions on F23–F25. MMPA outperforms the other comparative algorithms on F23, F25, and F27–F28. CPIJADE performs best on F22. F17–F30 are more complex combinatorial test functions, and HEGMPA achieves the best results on eight of them, providing better evidence of HEGMPA's potential to solve complex optimization problems in the real world.

To further illustrate the convergence performance of the algorithms, Figure 3 shows the average error convergence curves of the six algorithms for solving the CEC2014 test set. HEGMPA has better convergence accuracy and faster convergence on F1–F4, F6–F7, F13, and F16–F21. In solving F5, F8–F10, and F12, HEGMPA converges faster and with better convergence accuracy in the later part of the iteration, although the convergence speed is slower in the early part of the iteration. In summary, HEGMPA outperforms the comparison algorithms in terms of convergence accuracy and convergence speed.

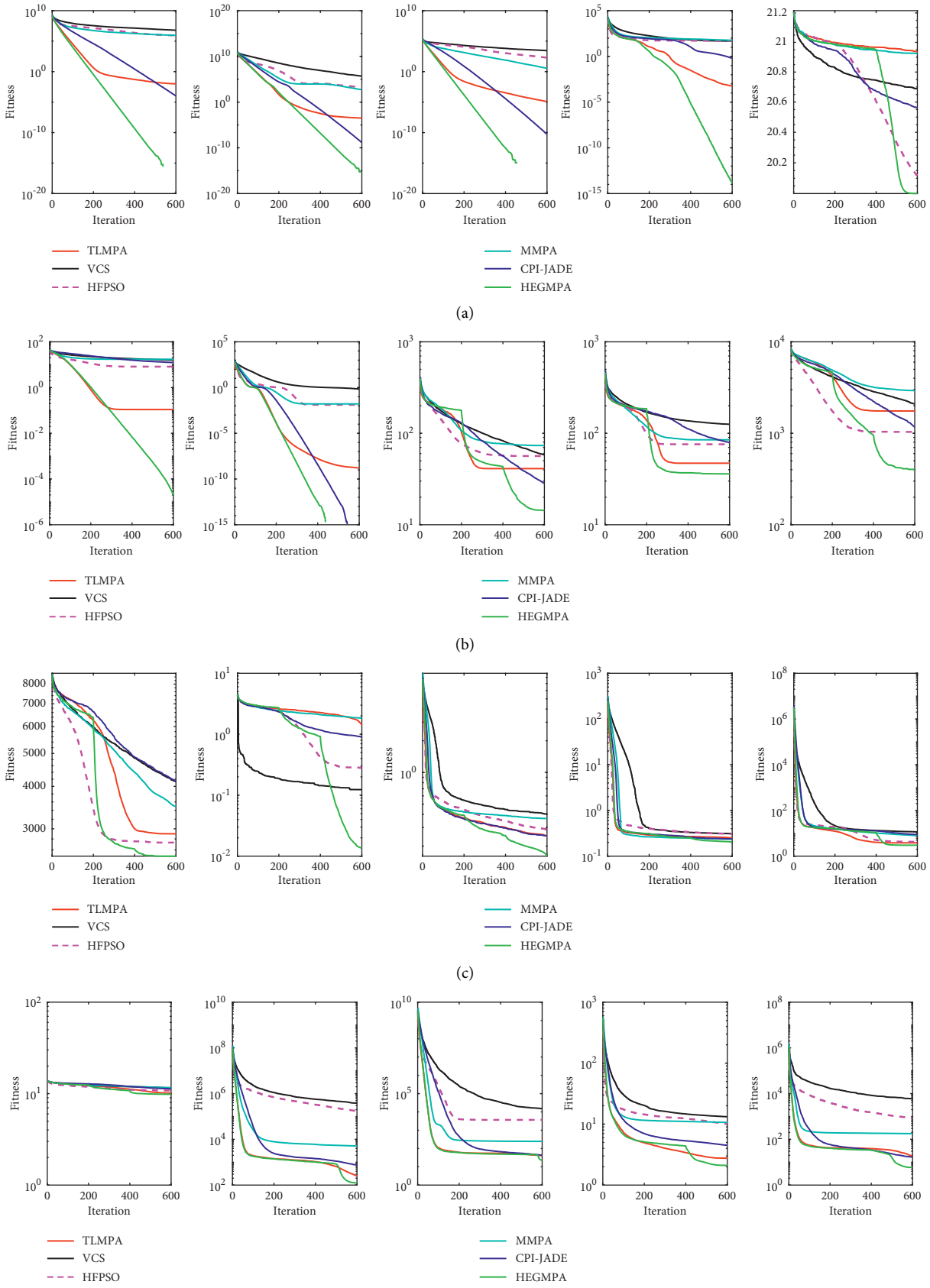

Figure 3: Continued.

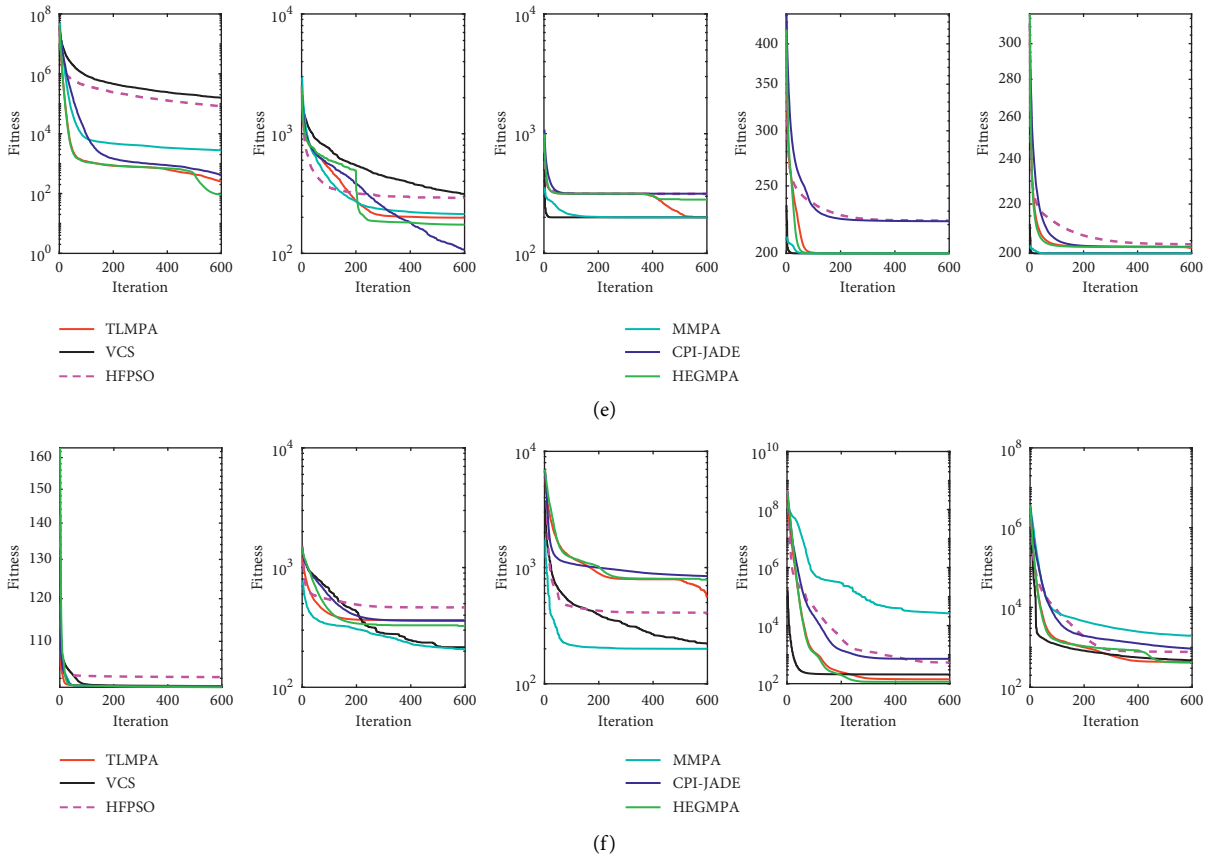

Figure 3: Convergence curves of CEC2014. (a) F1–F5. (b) F6–F10. (c) F11–F15. (d) F16–F20. (e) F21–F25. (f) F26–F30.

To analyse the distributional properties of the solutions solved by the improved algorithms, box plots were drawn based on the results of 51 independent solutions for each algorithm, as shown in Figure 4. For each algorithm, the centre marker of each box indicates the median of the results of 51 solved functions, the bottom and top edges of the box indicate first- and third-degree points, and the symbol "+" indicates bad values that are not inside the box. As can be seen from Figure 4, HEGMPA has no outliers when solving 17 of the test functions (F1–F3, F7–F9, F11, F13–F15, F18, F21, F23–F24, F26–F27, and F29), indicating that the distribution solved by HEGMPA is very concentrated; meanwhile, for the other test functions where bad values exist, the HEGMPA has a small median, indicating that the quality of HEGMPA's solutions is relatively better. Therefore, the improved algorithm proposed in this paper has a strong robustness.

To avoid chance in testing, this paper uses the Wilcoxon signed-rank test to verify whether the improved algorithms are statistically significantly different from the comparison algorithms. Table 5 presents the results of the Wilcoxon signed-rank sum test for each algorithm and HEGMPA. In the table, "+" indicates that HEGMPA outperforms the comparison algorithm in terms of optimization results, "−" indicates poorer results, "=" indicates similar results, and the symbol "$R+$" is a positive rank value indicating the extent to which HEGMPA is better than the comparison algorithm and "$R-$" indicates the opposite result. As can be seen in Table 4, HEGMPA outperformed the basic MPA in at least 23 of the 30 tested functions due to all comparison algorithms and in all tested functions, which statistically validates the excellent performance of the improved algorithm.

The computational efficiency of the algorithm is also another important aspect in evaluating the performance of the algorithm. Table 6 lists the average time taken by each algorithm in solving the test function, and the last column lists the average ranking of each algorithm. We can learn that HEGMPA takes more time to compute, ranking only fourth, while MPA ranks third, due to the increased computational cost caused by the introduction of EDA, and the computation of the covariance matrix based on the Gaussian distribution model increases the computational time taken by the improved algorithm. Although the introduction of the improved strategy leads to an increase in the computational time consumed by the basic MPA, the performance improvement it brings is significant and therefore the computational time consumed by the HEGMPA proposed in this paper is acceptable.

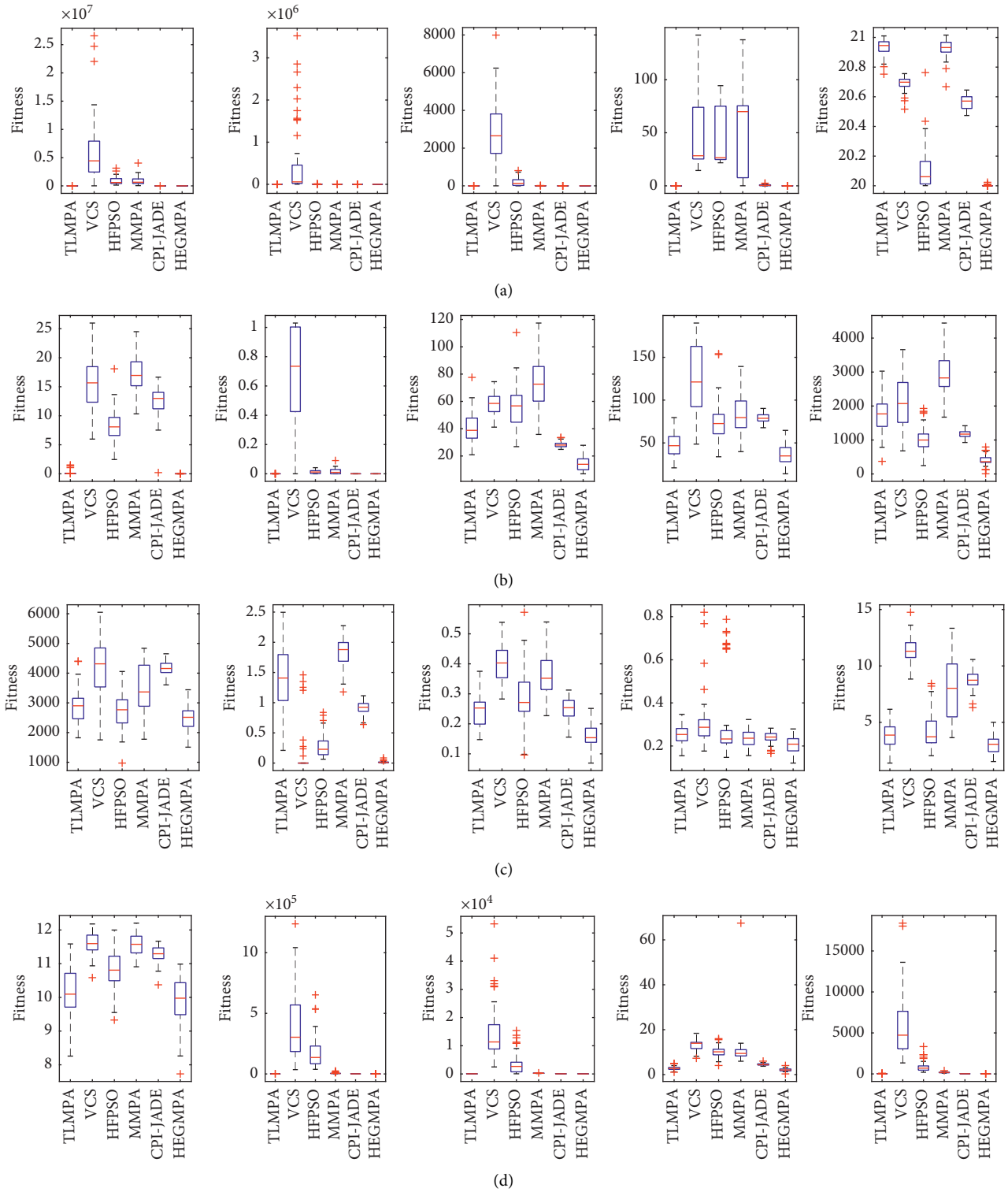

Figure 4: Continued.

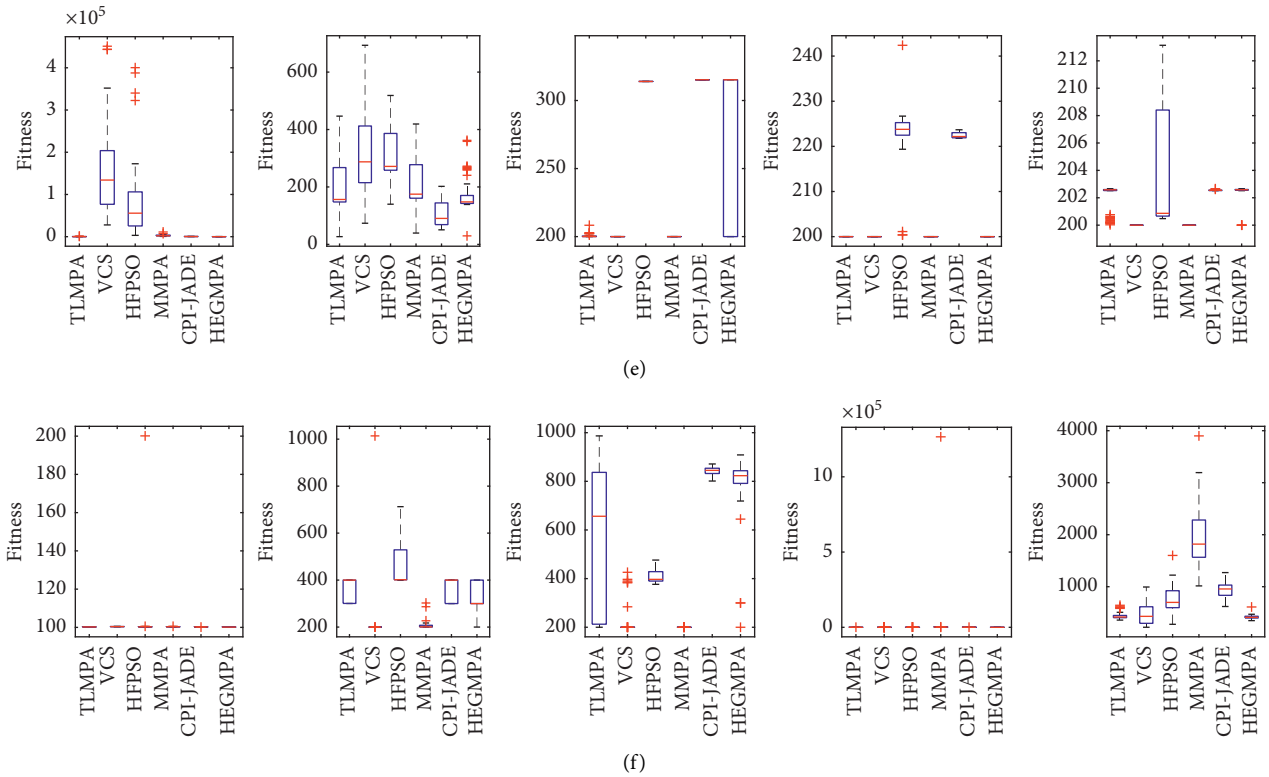

Figure 4: Box diagram of CEC2014. (a) F1–F5. (b) F6–F10. (c) F11–F15. (d) F16–F20. (e) F21–F25. (f) F26–F30.

Table 5: Wilcoxon signed-rank sum test results for CEC2014.

No.	MPA				TLMPA				VCS			
	P value	$R+$	$R-$	Win	P value	$R+$	$R-$	Win	P value	$R+$	$R-$	Win
F1	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F2	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F3	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F4	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F5	$5.46E-10$	1325	1	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F6	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F7	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F8	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F9	$5.15E-10$	1326	0	+	$1.00E-04$	1078	248	+	$5.15E-10$	1326	0	+
F10	$5.15E-10$	1326	0	+	$5.46E-10$	1325	1	+	$5.15E-10$	1326	0	+
F11	$5.15E-10$	1326	0	+	$8.47E-04$	1019	307	+	$1.87E-09$	1304	22	+
F12	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$7.99E-03$	380	946	−
F13	$5.15E-10$	1326	0	+	$3.94E-09$	1291	35	+	$5.15E-10$	1326	0	+
F14	$5.15E-10$	1326	0	+	$6.77E-07$	1193	133	+	$4.17E-09$	1290	36	+
F15	$5.15E-10$	1326	0	+	$3.95E-04$	1041	285	+	$5.15E-10$	1326	0	+
F16	$5.15E-10$	1326	0	+	$5.35E-02$	869	457	=	$5.15E-10$	1326	0	+
F17	$5.15E-10$	1326	0	+	$3.45E-03$	975	351	+	$5.15E-10$	1326	0	+
F18	$5.15E-10$	1326	0	+	$5.35E-05$	1094	232	+	$5.15E-10$	1326	0	+
F19	$5.15E-10$	1326	0	+	$7.45E-06$	1141	185	+	$5.15E-10$	1326	0	+
F20	$5.15E-10$	1326	0	+	$3.20E-08$	1253	73	+	$5.15E-10$	1326	0	+
F21	$5.15E-10$	1326	0	+	$7.34E-05$	1086	240	+	$5.15E-10$	1326	0	+
F22	$5.15E-10$	1326	0	+	$1.29E-01$	825	501	=	$3.56E-08$	1251	75	+
F23	$5.15E-10$	1326	0	+	$3.58E-07$	120	1206	−	$1.97E-09$	0	666	−
F24	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	0	1326	−
F25	$5.15E-10$	1326	0	+	$1.15E-01$	495	831	=	$1.63E-09$	0	1176	−
F26	$5.15E-10$	1326	0	+	$1.14E-08$	1272	54	+	$5.15E-10$	1326	0	+
F27	$5.15E-10$	1326	0	+	$1.32E-06$	1179	147	+	$1.42E-08$	50	1225	−

Table 5: Continued.

No.	MPA				TLMPA				VCS			
	P value	*R*+	*R*−	Win	*P* value	*R*+	*R*−	Win	*P* value	*R*+	*R*−	Win
F28	$5.15E-10$	1326	0	+	$7.94E-05$	242	1084	−	$5.46E-10$	1	1325	−
F29	$5.15E-10$	1326	0	+	$7.94E-05$	1084	242	+	$5.15E-10$	1326	0	+
F30	$5.15E-10$	1326	0	+	$1.29E-01$	825	501	=	$1.74E-01$	808	518	=
+/−/=		30/0/0				24/2/4				23/6/1		
No.	HFPSO				MMPA				CPI-JAD*E*			
	P value	*R*+	R−	Win	P value	*R*+	R−	Win	P value	*R*+	R−	Win
F1	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F2	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F3	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F4	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F5	$7.80E-10$	1319	7	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F6	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F7	$5.10E-10$	1326	0	+	$5.15E-10$	1326	0	+	1.00*E*+00	0	0	=
F8	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.80E-10$	1324	2	+
F9	$5.80E-10$	1324	2	+	$5.80E-10$	1324	2	+	$5.15E-10$	1326	0	+
F10	$1.32E-09$	1310	16	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F11	$3.49E-02$	888	438	+	$2.32E-08$	1259	67	+	$5.15E-10$	1326	0	+
F12	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F13	$1.18E-09$	1312	14	+	$5.15E-10$	1326	0	+	$8.77E-10$	1317	9	+
F14	$3.82E-04$	1042	284	+	$2.76E-04$	1051	275	+	$7.78E-06$	1140	186	+
F15	$6.24E-06$	1145	181	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F16	$2.41E-07$	1214	112	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F17	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F18	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.80E-10$	1324	2	+
F19	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F20	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F21	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$1.25E-09$	1311	15	+
F22	$4.94E-09$	1287	39	+	$1.27E-02$	929	397	+	$1.31E-07$	100	1226	−
F23	$9.78E-01$	660	666	=	$1.97E-09$	0	666	−	$6.10E-05$	120	0	+
F24	$5.15E-10$	1326	0	+	$5.15E-10$	0	1326	−	$5.15E-10$	1326	0	+
F25	$7.50E-01$	629	697	=	$1.63E-09$	0	1176	−	$1.98E-03$	333	993	−
F26	$1.25E-09$	1311	15	+	$5.15E-10$	1326	0	+	$6.53E-10$	1322	4	+
F27	$5.80E-10$	1324	2	+	$7.35E-10$	6	1320	−	$1.92E-05$	1119	207	+
F28	$7.35E-10$	6	1320	−	$5.15E-10$	0	1326	−	$2.06E-04$	1059	267	+
F29	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
F30	$1.67E-09$	1306	20	+	$5.15E-10$	1326	0	+	$5.15E-10$	1326	0	+
+/−/=		27/1/2				25/5/0				27/2/1		

Table 6: Time cost of seven algorithms in CEC2014 30D testing.

	MPA	TLMPA	VCS	HFPSO	MMPA	CPIJADE	HEGMPA
F1	$2.80E+00$	$2.19E+00$	$2.52E+00$	$1.32E+00$	$2.71E+01$	$3.51E+00$	$3.24E+00$
F2	$2.25E+00$	$1.69E+00$	$1.97E+00$	8.31*E*-01	$2.58E+01$	$3.09E+00$	$2.63E+00$
F3	$2.26E+00$	$1.73E+00$	$1.94E+00$	8.23*E*-01	$2.65E+01$	$3.15E+00$	$2.68E+00$
F4	$2.26E+00$	$1.74E+00$	$1.97E+00$	8.22*E*-01	$2.75E+01$	$3.18E+00$	$2.61E+00$
F5	$2.57E+00$	$1.83E+00$	$2.92E+00$	$1.10E+00$	$2.69E+01$	$3.11E+00$	$2.94E+00$
F6	$2.69E+01$	$2.78E+01$	$3.25E+01$	$2.52E+01$	$6.61E+01$	$2.83E+01$	$2.74E+01$
F7	$2.59E+00$	$1.95E+00$	$2.25E+00$	$1.08E+00$	$2.61E+01$	$3.40E+00$	$2.91E+00$
F8	$2.12E+00$	$1.49E+00$	$1.80E+00$	6.31*E*-01	$2.54E+01$	$2.64E+00$	$2.40E+00$
F9	$2.54E+00$	$1.99E+00$	$2.17E+00$	$1.06E+00$	$2.73E+01$	$3.12E+00$	$2.87E+00$
F10	$2.90E+00$	$1.87E+00$	$2.28E+00$	$1.03E+00$	$2.66E+01$	$3.03E+00$	$2.73E+00$
F11	$3.28E+00$	$2.35E+00$	$3.18E+00$	$1.58E+00$	$2.79E+01$	$3.52E+00$	$3.15E+00$
F12	$7.98E+00$	$8.13E+00$	$6.31E+00$	$6.52E+00$	$3.65E+01$	$9.59E+00$	$8.32E+00$
F13	$2.28E+00$	$1.54E+00$	$1.93E+00$	8.16*E*-01	$2.65E+01$	$2.87E+00$	$2.62E+00$
F14	$2.26E+00$	$1.43E+00$	$1.91E+00$	7.92*E*-01	$2.74E+01$	$2.84E+00$	$2.59E+00$
F15	$2.60E+00$	$2.05E+00$	$2.25E+00$	$1.11E+00$	$2.62E+01$	$3.22E+00$	$2.99E+00$
F16	$2.62E+00$	$2.18E+00$	$2.46E+00$	$1.16E+00$	$2.61E+01$	$3.24E+00$	$3.24E+00$

Table 6: Continued.

	MPA	TLMPA	VCS	HFPSO	MMPA	CPIJADE	HEGMPA
F17	3.02$E+00$	1.99$E+00$	2.57$E+00$	1.25$E+00$	2.63$E+01$	3.50$E+00$	3.17$E+00$
F18	2.41$E+00$	1.41$E+00$	2.13$E+00$	9.30E-01	2.58$E+01$	3.14$E+00$	2.85$E+00$
F19	7.42$E+00$	7.55$E+00$	7.28$E+00$	5.93$E+00$	3.36$E+01$	8.91$E+00$	7.88$E+00$
F20	2.47$E+00$	1.56$E+00$	2.44$E+00$	1.02$E+00$	2.61$E+01$	3.17$E+00$	2.83$E+00$
F21	2.57$E+00$	1.75$E+00$	2.44$E+00$	1.11$E+00$	2.64$E+01$	3.31$E+00$	2.94$E+00$
F22	3.19$E+00$	2.78$E+00$	3.15$E+00$	1.67$E+00$	2.70$E+01$	4.06$E+00$	3.55$E+00$
F23	4.76$E+00$	4.47$E+00$	4.46$E+00$	3.27$E+00$	2.97$E+01$	6.25$E+00$	5.16$E+00$
F24	4.60$E+00$	3.32$E+00$	3.29$E+00$	2.11$E+00$	2.78$E+01$	4.51$E+00$	6.73$E+00$
F25	5.49$E+00$	4.42$E+00$	4.23$E+00$	3.10$E+00$	2.95$E+01$	5.87$E+00$	4.93$E+00$
F26	3.10$E+01$	3.09$E+01$	4.76$E+01$	2.86$E+01$	7.00$E+01$	3.17$E+01$	3.08$E+01$
F27	3.01$E+01$	3.10$E+01$	4.45$E+01$	2.88$E+01$	6.96$E+01$	3.15$E+01$	3.06$E+01$
F28	5.96$E+00$	6.00$E+00$	6.32$E+00$	4.39$E+00$	3.09$E+01$	7.00$E+00$	6.34$E+00$
F29	9.24$E+00$	9.67$E+00$	1.31$E+01$	7.47$E+00$	3.62$E+01$	1.06$E+01$	9.45$E+00$
F30	4.68$E+00$	4.66$E+00$	5.09$E+00$	3.18$E+00$	2.94$E+01$	5.76$E+00$	5.06$E+00$
Average rank	3.73	2.50	3.36	1.03	7.00	5.76	4.60

5. Conclusions

In this paper, we propose a variant of MPA, called HEGMPA. The performance of the algorithm is improved using Tent mapping, distribution estimation strategy, and Gaussian random walk. To evaluate the effectiveness of the improved strategy and the superiority of HEGMPA, it was validated using the CEC2014 test suite. It was compared with five state-of-the-art algorithms through numerical analysis, convergence analysis, stability analysis, and statistical tests. The simulation results show that the HEGMPA algorithm balances development and exploration and is competitive with other algorithms. On the other hand, there is still room for improvement in HEGMPA. The effect of initialization of small populations needs to be investigated. The calculation of the covariance matrix increases the computational cost. Therefore, how to reduce the computational cost while maintaining performance is an issue that needs to be further investigated.

In future work, we plan to further apply HEGMPA to medical image recognition detection. In addition, we plan to develop a multiobjective version of HEGMPA to address optimization problems in other domains.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors acknowledge funding received from the Key Scientific and Technological Research and Development Projects of Jilin (20200401093GX).

References

[1] G. Wu, W. Pedrycz, P. N. Suganthan, and R. Mallipeddi, "A variable reduction strategy for evolutionary algorithms handling equality constraints," *Applied Soft Computing Journal*, vol. 37, 2015.

[2] J. Katebi, M. Shoaei-parchin, M. Shariati, N. T. Trung, and M. Khorami, "Developed comparative analysis of metaheuristic optimization algorithms for optimal active control of structures," *Engineering with Computers*, vol. 36, 2020.

[3] K. Hussain, M. N. Mohd Salleh, S. Cheng, and Y. Shi, "Metaheuristic research: a comprehensive survey," *Artificial Intelligence Review*, vol. 52, 2019.

[4] A. Di Tang, T. Han, H. Zhou, and L. Xie, "An improved equilibrium optimizer with application in unmanned aerial vehicle path planning," *Sensors*, vol. 21, 2021.

[5] Y. Li, T. Han, H. Zhao, and H. Gao, "An adaptive whale optimization algorithm using Gaussian distribution strategies and its application in heterogeneous UCAVs task allocation," *IEEE Access*, vol. 7, 2019.

[6] M. Alweshah, S. Al Khalaileh, B. B. Gupta, A. Almomani, A. I. Hammouri, and M. A. Al-Betar, "The monarch butterfly optimization algorithm for solving feature selection problems," *Neural Computing & Applications*, 2020.

[7] O. S. Qasim, N. A. Al-Thanoon, and Z. Y. Algamal, "Feature selection based on chaotic binary black hole algorithm for data classification," *Chemometrics and Intelligent Laboratory Systems*, vol. 204, 2020.

[8] X. Lin and Y. Wu, "Parameters identification of photovoltaic models using niche-based particle swarm optimization in parallel computing architecture," *Energy*, vol. 196, 2020.

[9] Q. Hao, Z. Zhou, Z. Wei, and G. Chen, "Parameters identification of photovoltaic models using a multi-strategy success-history-based adaptive differential evolution," *IEEE Access*, vol. 8, 2020.

[10] M. Abd Elaziz, D. Yousri, M. A. A. Al-qaness, A. M. AbdelAty, A. G. Radwan, and A. A. Ewees, "A Grunwald–Letnikov based Manta ray foraging optimizer for global optimization and image segmentation," *Engineering Applications of Artificial Intelligence*, vol. 98, 2021.

[11] A. Wunnava, M. K. Naik, R. Panda, B. Jena, and A. Abraham, "A novel interdependence based multilevel thresholding

technique using adaptive equilibrium optimizer," *Engineering Applications of Artificial Intelligence*, vol. 94, 2020.

[12] H. J. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI, USA, 1992.

[13] R. A. Sarker, S. M. Elsayed, and T. Ray, "Differential evolution with dynamic parameters selection for optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, 2014.

[14] J. R. Koza and J. P. Rice, "Automatic programming of robots using genetic programming," in *Proceedings of the tenth national conference on Artificial intelligence*, San Jose, CA, USA, July 1992.

[15] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies–a comprehensive introduction," *Natural Computing*, vol. 1, 2002.

[16] S. A. Uymaz, G. Tezel, and E. Yel, "Artificial algae algorithm (AAA) for nonlinear global optimization," *Applied Soft Computing J.*vol. 32, 2015.

[17] Z. Meng and J. S. Pan, "Monkey King Evolution: a new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization," *Knowledge-Based Systems*, vol. 97, 2016.

[18] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 80, 1983.

[19] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Science*, vol. 179, 2009.

[20] Z. Wei, C. Huang, X. Wang, T. Han, and Y. Li, "Nuclear reaction optimization: a novel and powerful physics-based algorithm for global optimization," *IEEE Access*, vol. 7, 2019.

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

[22] S. Mirjalili, "SCA: a Sine Cosine Algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, 2016.

[23] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the ICNN'95-International Conference on Neural Networks*, Perth, WA, Australia, November 1995.

[24] M. Dorigo and G. Di Caro, "Ant colony optimization: a new meta-heuristic," in *Proceedings of the 1999 Congress on Evolutionary Computation*, Washington, DC, USA, July 1999.

[25] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, 2014.

[26] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, 2016.

[27] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," *Systems Science and Control Engineering*, vol. 8, 2020.

[28] X. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, London, UK, 2010.

[29] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, 2007.

[30] L. Xie, T. Han, H. Zhou, Z.-R. Zhang, B. Han, and A. Tang, "Tuna swarm optimization: a novel swarm-based metaheuristic algorithm for global optimization," *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 9210050, 22 pages, 2021.

[31] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: a nature-inspired metaheuristic," *Expert Systems with Applications*, vol. 152, 2020.

[32] M. A. Soliman, H. M. Hasanien, and A. Alkuhayli, "Marine predators algorithm for parameters identification of triple-diode photovoltaic models," *IEEE Access*, vol. 8, 2020.

[33] H. M. Ridha, "Parameters extraction of single and double diodes photovoltaic models using Marine Predators Algorithm and Lambert W function," *Solar Energy*, vol. 209, 2020.

[34] M. A. M. Shaheen, D. Yousri, A. Fathy, H. M. Hasanien, A. Alkuhayli, and S. M. Muyeen, "A novel application of improved marine predators algorithm and particle swarm optimization for solving the ORPD problem," *Energies*, vol. 13, 2020.

[35] N. Wang, J. S. Wang, L. F. Zhu, H. Y. Wang, and G. Wang, "A novel dynamic clustering method by integrating marine predators algorithm and particle swarm optimization algorithm," *IEEE Access*, vol. 9, 2021.

[36] M. Abdel-Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, "Energy-aware marine predators algorithm for task scheduling in IoT-based fog computing applications," *IEEE Transactions on Industrial Informatics*, vol. 17, 2020.

[37] A. Tang, H. Zhou, T. Han, and L. Xie, "A chaos sparrow search algorithm with logarithmic spiral and adaptive step for engineering problems," *Computer Modeling in Engineering and Sciences*, 2021.

[38] A. A. Ewees and M. A. Elaziz, "Performance analysis of Chaotic Multi-Verse Harris Hawks Optimization: a case study on solving engineering problems," *Engineering Applications of Artificial Intelligence*, vol. 88, 2020.

[39] Z. Zhang, H. Huang, C. Huang, and B. Han, "An improved TLBO with logarithmic spiral and triangular mutation for global optimization," *Neural Computing & Applications*, vol. 31, 2019.

[40] N. A. Nor, Z. Ibrahim, M. Mubin, S. W. Nawawi, and M. S. Mohamad, "Improving particle swarm optimization via adaptive switching asynchronous – synchronous update," *Applied Soft Computing-Journals*, vol. 72, 2018.

[41] H. Liu, Y. Pan, S. Li, and Y. Chen, "Synchronization for fractional-order neural networks with full/under-actuation using fractional-order sliding mode control," *Int. International Journal of Machine Learning and Cybernetics*, vol. 9, 2018.

[42] H. Liu, S. Li, G. Lia, and H. Wang, "Robust adaptive control for fractional-order financial chaotic systems with system uncertainties and external disturbances," *Information Technology and Control*, vol. 46, 2017.

[43] B. Jin, L. Cruz, and N. Goncalves, "Deep facial diagnosis: deep transfer learning from face recognition to facial diagnosis," *IEEE Access*, vol. 8, 2020.

[44] B. Jin, L. Cruz, and N. Gonçalves, "Face depth prediction by the scene depth," in *Proceedings of the 20th IEEE/ACIS International Conference on Computer and Information Science ICIS 2021-Summer*, Shanghai, China, June 2021.

[45] A. Tang, H. Zhou, T. Han, and L. Xie, "A modified manta ray foraging optimization for global optimization problems," *IEEE Access*, vol. 9, 2021.

[46] X. Wang, H. Zhao, T. Han, H. Zhou, and C. Li, "A grey wolf optimizer using Gaussian estimation of distribution and its application in the multi-UAV multi-target urban tracking problem," *Applied Soft Computing Journal*, vol. 78, 2019.

[47] M. D. Li, H. Zhao, X. W. Weng, and T. Han, "A novel nature-inspired algorithm for optimization: virus colony search," *Advances in Engineering Software*, vol. 92, 2016.

[48] Y. Xu, Z. Yang, X. Li, H. Kang, and X. Yang, "Dynamic opposite learning enhanced teaching–learning-based optimization," *Knowledge-Based Systems*, vol. 188, 2020.
[49] Y. Wang, Z. Z. Liu, J. Li, H. X. Li, and G. G. Yen, "Utilizing cumulative population distribution information in differential evolution," *Applied Soft Computing Journal*, vol. 48, 2016.
[50] İ. B. Aydilek, "A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems," *Applied Soft Computing Journal*, vol. 66, 2018.