Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 9752003, 20 pages
https://doi.org/10.1155/2022/9752003

Research Article

A Modified Reptile Search Algorithm for Numerical Optimization Problems

Qihang Yuan,[1] **Yongde Zhang**,[1,2] **Xuesong Dai**,[1] **and Shu Zhang**[2]

[1]*Key Laboratory of Advanced Manufacturing and Intelligent Technology, Ministry of Education, Harbin University of Science and Technology, Harbin 150080, China*
[2]*Foshan Baikang Robot Technology Co., Ltd, Nanhai, Foshan, Guangdong 528225, China*

Correspondence should be addressed to Yongde Zhang; zhangyd@hrbust.edu.cn

Received 18 June 2022; Revised 13 August 2022; Accepted 20 August 2022; Published 10 October 2022

Academic Editor: Heng Liu

The reptile search algorithm (RSA) is a swarm-based metaheuristic algorithm inspired by the encirclement and hunt mechanisms of crocodiles. Compared with other algorithms, RSA is competitive but still suffers from low population diversity, unbalanced exploitation and exploration, and the tendency to fall into local optima. To overcome these shortcomings, a modified variant of RSA, named MRSA, is proposed in this paper. First, an adaptive chaotic reverse learning strategy is employed to enhance the population diversity. Second, an elite alternative pooling strategy is proposed to balance exploitation and exploration. Finally, a shifted distribution estimation strategy is used to correct the evolutionary direction and improve the algorithm performance. Subsequently, the superiority of MRSA is verified using 23 benchmark functions, IEEE CEC2017 benchmark functions, and robot path planning problems. The Friedman test, the Wilcoxon signed-rank test, and simulation results show that the proposed MRSA outperforms other comparative algorithms in terms of convergence accuracy, convergence speed, and stability.

1. Introduction

The rapid advancement of technology has generated a large number of optimization problems that require solving. These optimization problems arise in various fields, such as finance, chemicals, electronics, machinery, and materials. Real-world optimization problems are often mixed with various unknown factors and have very complex solution spaces. These problems frequently have substantial computational efforts, complex nonlinear constraints, and large numbers of variables and constraints [1–6]. Traditional optimization methods have difficulty solving these nonproductivity discontinuity problems effectively because they cannot strike a balance between accuracy and time cost [7]. Metaheuristic optimization algorithms have demonstrated better performance in balancing the solution quality and time cost [8]. Due to a simple structure and no requirement for a problem to be continuously derivable, metaheuristic optimization algorithms have been widely used to solve complex optimization problems in natural and engineering fields [9–13].

In recent decades, metaheuristic algorithms have made great progress in memetic computing manner, balance of exploitation and exploration, self-adaption of hyperparameters, population structure evolution, and theoretical analysis of the search dynamics [14]. Memetic computing manner improves algorithm performance through metaheuristic algorithms incorporated with local search operator. Charin et al. used particle swarm optimization (PSO) combined with levy flight optimization (LFO) to track the maximum power point of a photovoltaic system [15]. Yu et al. showed that the combination of chaotic local search (CLS) and brain storm optimization (BSO) can significantly improve the performance of BSO [16]. How to balance the exploration and exploitation of the algorithm to improve the performance is a research hotspot of the metaheuristic algorithms. Many researchers use various operators or change the algorithm parameters to balance [17]. Cai et al. proposed an alternate search pattern strategy to balance the exploration and exploitation of BSO [18]. In the optimization process, the search performance of some metaheuristic algorithms is greatly affected by adjustable parameters such as crossover rate, mutation rate, and population size. In order to solve the problem of parameter value control at different stages in the optimization process, adaptive parameter control has been extensively studied by researchers [19]. Lei et al. proposed a variant of gravitational search algorithm (GSA) with a self-adaptive gravitational constant called ALGSA, which greatly improved the search performance of GSA [20]. Population structure evolution has great influence on the search performance of the metaheuristic algorithms. Zhong et al. proposed a variant to improve the performance of differential evolution (DE) algorithm called EHDE by incorporating elite elements into the hierarchical population structure [21]. Inspired by the two-layered structure GSA, Wang et al. proposed a four-layered GSA variant with stronger search capability called MLGSA [22]. In addition to the above factors, theoretical analysis of the search dynamics has recently attracted a great deal of attention from researchers [23].

In general, metaheuristic optimization algorithms can be classified into three categories [24]: evolutionary-based algorithms, physical-based algorithms, and swarm-based algorithms.

Evolutionary-based algorithms are inspired by the laws of natural evolution. Genetic algorithms are a typical example and their proposal was inspired by Darwinian evolutionary theory [25]. Genetic algorithms provide solutions through the concept of crossover and mutation of species in nature. In addition, other evolutionary-based algorithms have been proposed, including DE [26], evolutionary programming [27], and evolutionary strategies [28]. The second category is physics-based algorithms, which originate from natural physics laws. Simulated annealing [29] and GSA [30] are two common physics-based algorithms. They utilize the laws of thermodynamics and gravity for optimization. In addition, researchers have proposed other physics-based algorithms. Wei et al. proposed a nuclear reaction optimizer using the phenomenon of atomic nuclear reactions [31]. Inspired by the sine and cosine laws of mathematics, Mirjalili proposed the sine cosine algorithm [32]. Eskandar et al. proposed a water cycle algorithm based on the natural water cycle phenomena [33]. The third category is swarm-based algorithms, which build optimization models by emulating the social behavior of animal groups. PSO [34] and the ant colony algorithm [35] are two of the most common swarm-based algorithms. They provide solutions by sharing information about all individuals in the optimization process. Others include the grey wolf optimizer (GWO) [36], the whale optimization algorithm (WOA) [37], the butterfly optimization algorithm (BOA) [38], the firefly algorithm (FA) [39], the artificial bee colony (ABC) algorithm [40], the reptile search algorithm (RSA) [41], the Harris hawks optimizer (HHO) [42], the equilibrium optimizer (EO) [43], the tunicate swarm algorithm (TSA) [44], the salp swarm algorithm (SSA) [45], the Tasmanian devil optimization (TDO) [46], the arithmetic optimization algorithm (AOA) [47], and the pathfinder algorithm (PFA) [48].

The reptile search algorithm (RSA) is a novel swarm-based algorithm proposed by Abualigah. RSA is inspired by the encircling mechanism, hunting mechanism, and social behavior of crocodiles [41]. RSA has good performance but also has disadvantages such as diminished population diversity and unbalanced exploitation and exploration capabilities. To improve the performance of RSA and enhance the search capability, this paper proposes a modified variant of RSA, named MRSA. To improve the population diversity, an adaptive chaotic reverse learning strategy is proposed to optimize from the initialization and in each iteration update. To balance exploitation and exploration, an elite alternative pool strategy was developed. A shifted distribution estimation strategy was used to modify all the individuals and guide the evolutionary direction. To fully validate the performance of MRSA, 23 benchmark functions, IEEE CEC2017 benchmark functions, and robot path planning problems are used for testing. The superiority of the proposed algorithm is demonstrated by a convergence analysis, stability analysis, and statistical tests.

The rest of this paper is organized as follows. Section 2 provides a review of the basic RSA. The proposed MRSA is described in detail in Section 3. In Section 4, the effectiveness of the proposed improved strategy and the superiority of the modified algorithm are verified using classical test functions, IEEE CEC2017 benchmark functions, and robot path planning problems. Finally, Section 5 provides the conclusion and discusses future work.

2. Reptile Search Algorithm

In this section, the basic procedures of RSA are presented. RSA is a swarm-based metaheuristic algorithm inspired by the enveloping mechanism, hunting mechanism, and social behavior of crocodiles.

2.1. Initialization Phase. RSA is similar to other metaheuristics in that the initial solution is generated randomly in the solution space. The initialization formula is as follows:

$$X_i^1 = LB + \text{rand} \cdot (UB - LB), \tag{1}$$

where X_i^1 is the i^{th} initial individual and LB and UB are the upper and lower boundaries of the search space, respectively.

2.2. Encircling Phase (Exploration). Crocodiles perform high and sprawl walks during the global search phase. In RSA, the search strategy is determined by the number of current iterations. When $t \leq 0.25T$, RSA performs a high walk. When $t \leq 0.5T$ and $t > 0.25T$, the RSA performs a sprawl walk. The specific mathematical models of the mechanism are described as follows:

$$X_i^{t+1} = \begin{cases} X_{best}^t - \eta_i \times \beta - R_i^t \times rand, t \leq \dfrac{T}{4}, \\ \\ X_{best}^t \times X_{rand}^t \times ES \times rand, t \leq \dfrac{T}{2} \; and \; t > \dfrac{T}{4}, \end{cases} \tag{2}$$

$$\eta_i = X_{best}^t \times P_i, \tag{3}$$

```
Initialize RSA parameters and generate initial population randomly
While $t < T$
Calculate the Fitness of each solution
Find the Best solution so far
Update the ES using equation (5)
For each crocodile $X_i$ do
Update the $\eta$, $R$, and values using equations (3), (4), and (6), respectively
If $t < 0.25T$
Calculate the new position $X_i$ using equation (2)
Else if $t \leq 0.5T$ and $t > 0.25T$
Calculate the new position $X_i$ using equation (2)
Else if $t \leq 0.75T$ and $t > 0.5T$
Calculate the new position $X_i$ using equation (7)
Else
Calculate the new position $X_i$ using equation (7)
End if
Calculate the fitness and select the better one
End for
$t = t + 1$
End while
Return the best position and fitness
```

Algorithm 1: Pseudocode of the reptile search algorithm (RSA).

Table 1: Set of chaotic functions.

ID	Type	Function
1	Chebyshev map	$x_{i+1} = \cos\left(i \cos^{-1}(x_i)\right)$
2	Circle map	$x_{i+1} = \mathrm{mod}\left(x_i + b - (a/2\pi)\sin(2\pi x_i)\right), 1), a = 0.5\, and\, b = 0.2$
3	Gauss map	$x_{i+1} = \begin{cases} 1 & x_i = 0 \\ 1/\mathrm{mod}\,(x_i, 1) & \text{otherwise} \end{cases}$
4	Iterative map	$x_{i+1} = \sin(a\pi/x_i), a = 0.7$
5	Logistic map	$x_{i+1} = ax_i(1 - x_i), a = 4$
6	Pricewise map	$x_{i+1} = \begin{cases} x_i/p & 0 \leq \mathrm{x}_i < \mathrm{p} \\ (x_i - p)/(0.5 - p) & \mathrm{p} \leq \mathrm{x}_i < 0.5 \\ (1 - x_i - p)/(0.5 - p) & 0.5 \leq \mathrm{x}_i < 1 - \mathrm{p} \\ (1 - x_i)/(p) & 1 - \mathrm{p} \leq \mathrm{x}_i < 1 \end{cases}$
7	Sine map	$x_{i+1} = a/4 \cdot \sin(\pi x_i), a = 4$
8	Singer map	$x_{i+1} = \mu(7.86x_i - 23.32x_i^2 + 28.75x_i^3 - 13.301875x_i^4), \mu = 1.07$
9	Sinusoidal map	$x_{i+1} = ax_i^2 \sin(\pi x_i), a = 2.3$
10	Tent map	$x_{i+1} = \begin{cases} x_i/0.7 & x_i < 0.7 \\ 10/3 \times (1 - x_i) & x_i \geq 0.7 \end{cases}$

$$R_i = \frac{X_{best}^t - X_i^t}{X_{best}^t + \varepsilon}, \tag{4}$$

$$ES = 2 \times r_1 \times \left(1 - \frac{1}{T}\right), \tag{5}$$

$$P_i = \alpha + \frac{X_i^t - M\left(X_i^t\right)}{X_{\mathrm{best}}^t \times (UB - LB) + \varepsilon}, \tag{6}$$

where X_{best}^t is the current best solution, t is the current number of iterations, T is the maximum number of iterations, β is a constant taking the value of 0.1 to control the speed of exploration, $X_{\mathrm{ran\,d}}^t$ is a randomly chosen individual, ES is a random value decreasing in the interval $[-2, 2]$, ε is a minimal value to ensure that the denominator is not equal to 0, r_1 is a random number in the interval $[-1, 1]$, α is a constant taking the value of 0.1, and rand is a random number with values from 0 to 1.

2.3. Hunting Phase (Exploitation). In RSA, crocodiles use two strategies for foraging: hunting coordination and cooperation. When $t < 0.75T$ and $t \geq 0.5T$, the RSA performs hunting coordination. When $t < T$ and $t \geq 0.75T$, a hunting cooperation strategy is employed by the RSA. The formula for position updating in the hunting phase is as follows:

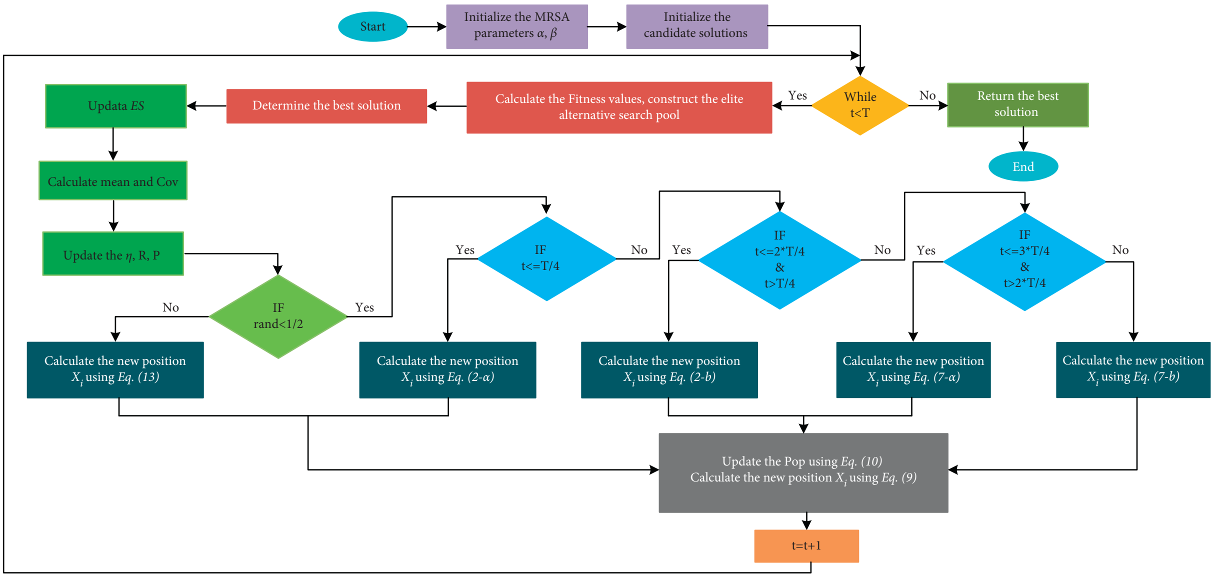
Start
Initialize the MRSA parameters α, β
Initialize the candidate solutions
While t<T
Yes
No
Return the best solution
End
Calculate the Fitness values, construct the elite alternative search pool
Determine the best solution
Updata ES
Calculate mean and Cov
Update the η, R, P
IF rand<1/2
No
Yes
IF t<=T/4
Yes
No
IF t<=2*T/4 & t>T/4
Yes
No
IF t<=3*T/4 & t>2*T/4
Yes
No
Calculate the new position X_i using Eq. (13)
Calculate the new position X_i using Eq. (2-a)
Calculate the new position X_i using Eq. (2-b)
Calculate the new position X_i using Eq. (7-a)
Calculate the new position X_i using Eq. (7-b)
Update the Pop using Eq. (10)
Calculate the new position X_i using Eq. (9)
t=t+1

FIGURE 1: Flow chart of MRSA.

```
Initialize RSA parameters and generate initial population randomly
While $t < T$
    Calculate the Fitness of each solution, construct the elite alternative search pool based on equation (12)
    Find the Best solution so far
    Update the ES using equation (5)
    Calculate mean and Cov based on equation (13)
    For each crocodile $X_i$ do
        Update the $\eta$, $R$, and values using equations (3), (4), and (6), respectively
        If rand < 0.5
            If $t \leq 0.25T$
                Calculate the new position $X_i$ using equation (2)
            Else if $t \leq 0.5T$ and $t > 0.25T$
                Calculate the new position $X_i$ using equation (2)
            Else if $t \leq 0.75T$ and $t > 0.5T$
                Calculate the new position $X_i$ using equation (7)
            Else
                Calculate the new position $X_i$ using equation (7)
            End if
        Else
            Calculate the new position $X_i$ using equations (13)
        End if
        Calculate the fitness and select the better one
    Update the Pop using equation (10)
    Calculate the new position $X_i$ using equation (9)
    Calculate the fitness and select the better one
    $t = t + 1$
End while
Return the best position and fitness
```

ALGORITHM 2: Pseudocode of the MRSA.

$$X_i^{t+1} = \begin{cases} X_{\text{best}}^t \times P_i \times \text{rand}, t \leq \dfrac{3T}{4} \text{ and } t > \dfrac{T}{2}, \\ X_{\text{best}}^t - \eta_i \times \varepsilon - R_i^t \times \text{rand}, t \leq T \text{ and } t > \dfrac{3T}{4}. \end{cases} \quad (7)$$

RSA generates the initial population randomly in the search space first and then chooses different search strategies depending on the number of iterations. The pseudocode for the RSA is shown in Algorithm 1.

Table 2: The classic benchmark functions (M: multimodal, U: unimodal, S: separable, N: nonseparable, Dim: dimension, Range: limits of search space, Optimum: global optimal value) [55].

Test function	Name	Type	*Dim*	Range	Optimum
$f_{01}(x)=\sum_{i=1}^{D}x_i^2$	Sphere	US	30	[−100, 100]	0
$f_{02}(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i-1}^{D}\lvert x_i\rvert$	Schwefel 2.22	UN	30	[−10, 10]	0
$f_{03}(x)=\sum_{i=1}^{D}(\sum_{j-1}^{D}x_i)^2$	Schwefel 1.2	UN	30	[−100, 100]	0
$f_{04}(x)=\max_i\{\lvert x_i\rvert,1\leq i\leq D\}$	Schwefel 2.21	US	30	[−100, 100]	0
$f_{05}(x)=\sum_{i=1}^{D}100(x_{i+1}^2-x_i^2)^2+(x_i-1)^2$	Rosenbrock	UN	30	[−30, 30]	0
$f_{06}(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	Step	US	30	[−100, 100]	0
$f_{07}(x)=\sum_{i=1}^{D}ix_i^4+\mathrm{random}[0,1)$	Quartic	US	30	[−1.28,1.28]	0
$f_{08}(x)=\sum_{i=1}^{D}-x_i\sin(\sqrt{\lvert x_i\rvert})$	Schwefel 2.26	MS	30	[−500, 500]	-418.9829D
$f_{09}(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	Rastrigin	MS	30	[−5.12, 5.12]	0
$f_{10}(x)=20+e-20\exp(-0.2\sqrt{1/D\sum_{i=1}^{D}x_i^2})-\exp(1/D\sum_{i=1}^{D}\cos(2\pi x_i))$	Ackley	MS	30	[−32, 32]	$8.8818e-16$
$f_{11}(x)=1/4000\sum_{i=1}^{D}(x_i^2)-(\prod_{i=1}^{D}\cos(x_i/\sqrt{i}))+1$	Griewank	MN	30	[−600, 600]	0
$f_{12}(x)=\pi/D\left\{10\sin(\pi y_i)+\sum_{i-1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)^2\right\}$ $+\sum_{i-1}^{D}u(x_i,10,100,4),\text{where}y_i=1+x_i+u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m\ x_i>a\\ 0-a<x_i<a\\ k(-x_i-a)^m x_i<a\end{cases}$	Penalized	MN	30	[−50, 50]	0
$f_{13}(x)=0.1(\sin^2(3\pi x_i)+\sum_{i=1}^{D}(x_i-1)^2[1+\sin^2(3\pi x_i)]+(x_D-1)^2 1+$ $\sin^2(2\pi x_D)]\}+\sum_{i-1}^{D}u(x_i,5,100,4)$	Penalized2	MN	30	[−50, 50]	0
$f_{14}(x)=(1/500+\sum_{j=1}^{25}1/j+\sum_{i=1}^{2}(x_i-a_{ij})^6)^{-1}$	Foxholes	MS	2	[−65.53,65.53]	0.998004
$f_{15}(x)=\sum_{i=1}^{11}(a_i-x_1(b_i^2+b_ix_2)/(b_i^2+b_ix_3+x_4))^{-1}$	Kowalik	MS	4	[−5, 5]	0.0003075
$f_{16}(x)=4x_1^2-2.1x_1^4+1/3x_1^6+x_1x_2-4x_2^2+x_2^4$	Six-hump camel back	MN	2	[−5, 5]	−1.03163
$f_{17}(x)=(x_2-5.1/4\pi^2*x_1^2+5/\pi*x_1-6)^2+10(1-1/8\pi)\cos x_1+10$	Branin	MS	2	[−5, 10] × [0, 15]	0.398
$f_{18}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]$ $\times[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	Goldstein-Price	MN	2	[−5, 5]	3
$f_{19}(x)=-\sum_{i=1}^{4}(c_i\exp-(\sum_{j-1}^{3}a_{ij}(x_j-p_{ij}{}^2))$	Hartman 3	MN	3	[0, 1]	−3.8628
$f_{20}(x)=-\sum_{i=1}^{4}(c_i\exp-(\sum_{j-1}^{6}a_{ij}(x_j-p_{ij}{}^2))$	Hartman 6	MN	6	[0, 1]	−3.32
$f_{21}(x)=-\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	Langermann 5	MN	4	[0, 10]	−10.1532
$f_{22}(x)=-\sum_{i=1}^{7}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	Langermann 7	MN	4	[0, 10]	−10.4029
$f_{23}(x)=-\sum_{i=1}^{10}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	Langermann 10	MN	4	[0, 10]	−10.5364

3. The Proposed RSA Variant

To enhance the performance of the basic RSA, three improvement strategies are proposed in this paper. An adaptive chaotic reverse learning strategy is first introduced to enhance the population diversity of RSA using the characteristics of chaotic mapping and reverse learning. Second, an elite alternative pooling strategy is used to balance the development and exploration of RSA. In addition, a distribution estimation strategy is used to modify the evolutionary direction. By sampling the dominant population information, the population direction is better guided, thus improving the algorithm's convergence efficiency. The three improvement strategies are described in detail in the following.

3.1. Adaptive Chaotic Reverse Learning Strategy.

One of the shortcomings of the metaheuristic algorithm is that the diversity of the algorithmic population continues to diminish as the optimization proceeds. To enhance the diversity of the algorithms, the researchers employ different approaches. The reverse learning strategy is a new technique that is widely used to improve population diversity. The reason for the popularity of reverse learning is that extensive literature shows that the probability of a reverse solution

Table 3: Results of 10 chaotic maps on all benchmark functions.

Function	Index	MRSA-C1	MRSA-C2	MRSA-C3	MRSA-C4	MRSA-C5	MRSA-C6	MRSA-C7	MRSA-C8	MRSA-C9	MRSA-C10	RSA [41]
F1	Mean	2.51E − 155	8.11E − 219	5.13E − 226	3.37E − 157	4.35E − 219	3.96E − 207	1.73E − 223	4.11E − 224	6.09E − 226	8.09E − 260	2.48E − 65
	Std	1.37E − 154	0.00E + 00	0.00E + 00	1.84E − 156	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	1.36E − 64
	Rank	10	7	2	9	6	8	5	4	3	**1**	11
F2	Mean	8.71E − 82	2.94E − 111	4.34E − 108	5.97E − 78	2.47E − 109	7.33E − 109	2.41E − 114	5.55E − 111	7.68E − 104	3.33E − 136	7.60E − 32
	Std	3.52E − 81	1.61E − 110	1.66E − 107	2.87E − 77	1.29E − 108	3.33E − 108	1.04E − 113	2.35E − 110	4.21E − 103	1.82E − 135	2.36E − 31
	Rank	9	3	7	10	5	6	2	4	8	**1**	11
F3	Mean	7.40E − 113	9.15E − 192	7.28E − 180	2.82E − 117	2.94E − 186	7.00E − 187	6.73E − 185	1.23E − 182	1.34E − 190	1.54E − 247	6.80E − 43
	Std	4.06E − 112	0.00E + 00	0.00E + 00	1.55E − 116	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	3.72E − 42
	Rank	10	2	8	9	5	4	6	7	3	**1**	11
F4	Mean	5.72E − 80	3.02E − 106	1.24E − 104	5.63E − 80	4.65E − 109	1.13E − 107	2.23E − 108	8.95E − 111	9.60E − 111	6.19E − 137	5.88E − 35
	Std	2.84E − 79	1.65E − 105	6.79E − 104	2.18E − 79	2.53E − 108	5.94E − 107	1.22E − 107	3.20E − 110	3.57E − 110	3.32E − 136	3.22E − 34
	Rank	10	7	8	9	4	6	5	2	3	**1**	11
F5	Mean	1.80E − 07	4.62E − 07	3.64E − 07	4.71E − 08	4.50E − 07	3.54E − 07	2.61E − 07	3.13E − 07	5.97E − 07	1.02E − 06	4.34E − 05
	Std	3.25E − 07	1.16E − 06	1.17E − 06	9.59E − 08	2.05E − 06	1.47E − 06	5.61E − 07	5.42E − 07	1.68E − 06	2.49E − 06	6.90E − 05
	Rank	2	8	6	**1**	7	5	3	4	9	10	11
F6	Mean	4.18E − 12	1.24E − 12	1.28E − 12	5.05E − 12	4.65E − 12	8.84E − 13	6.78E − 13	1.06E − 12	1.07E − 12	5.88E − 13	1.54E − 09
	Std	1.22E − 11	2.70E − 12	3.04E − 12	2.36E − 11	2.21E − 11	1.42E − 12	2.25E − 12	1.90E − 12	2.74E − 12	1.04E − 12	3.30E − 09
	Rank	8	6	7	10	9	3	2	4	5	**1**	11
F7	Mean	3.84E − 04	2.66E − 04	1.92E − 04	3.40E − 04	2.71E − 04	2.56E − 04	2.21E − 04	2.02E − 04	2.54E − 04	1.81E − 04	5.54E − 04
	Std	3.47E − 04	1.86E − 04	1.98E − 04	2.65E − 04	1.56E − 04	2.01E − 04	2.36E − 04	1.47E − 04	2.30E − 04	1.55E − 04	5.97E − 04
	Rank	10	7	2	9	8	6	4	3	5	**1**	11
F8	Mean	−7.63E + 03	−8.97E + 03	−8.67E + 03	−8.00E + 03	−8.39E + 03	−8.22E + 03	−8.96E + 03	−9.29E + 03	−8.99E + 03	−8.44E + 03	−8.29E + 03
	Std	7.50E + 02	1.73E + 03	1.64E + 03	1.04E + 03	9.94E + 02	1.02E + 03	1.45E + 03	1.74E + 03	1.47E + 03	5.81E + 02	5.78E + 02
	Rank	11	3	5	10	7	9	4	**1**	2	6	8
F9	Mean	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00
	Std	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F10	Mean	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16	8.88E − 16
	Std	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F11	Mean	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00
	Std	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F12	Mean	6.09E − 13	4.86E − 13	6.84E − 13	2.35E − 13	1.55E − 12	3.62E − 13	3.36E − 13	2.93E − 13	5.16E − 13	6.21E − 13	4.51E − 11
	Std	2.06E − 12	6.87E − 13	1.87E − 12	4.26E − 13	5.95E − 12	6.65E − 13	6.92E − 13	5.74E − 13	1.16E − 12	1.59E − 12	1.95E − 10
	Rank	7	5	9	**1**	10	4	3	2	6	8	11
F13	Mean	5.56E − 12	2.44E − 11	1.86E − 11	7.55E − 12	1.54E − 11	1.41E − 11	1.00E − 11	8.05E − 12	1.27E − 11	2.79E − 11	1.84E − 09
	Std	1.56E − 11	6.58E − 11	4.90E − 11	2.39E − 11	7.35E − 11	2.19E − 11	2.03E − 11	2.37E − 11	2.74E − 11	1.11E − 10	6.87E − 09
	Rank	**1**	9	8	2	7	6	4	3	5	10	11

Table 3: Continued.

Function	Index	MRSA-C1	MRSA-C2	MRSA-C3	MRSA-C4	MRSA-C5	MRSA-C6	MRSA-C7	MRSA-C8	MRSA-C9	MRSA-C10	RSA [41]
F14	Mean	1.06E+00	1.79E+00	1.59E+00	1.52E+00	1.62E+00	2.17E+00	1.46E+00	2.76E+00	1.20E+00	9.98E−01	8.98E+00
	Std	3.62E−01	1.90E+00	1.86E+00	1.85E+00	1.90E+00	2.97E+00	1.83E+00	3.68E+00	6.05E−01	1.54E−16	5.34E+00
	Rank	2	8	6	5	7	9	4	10	3	**1**	11
F15	Mean	3.07E−04	3.07E−04	3.07E−04	3.07E−04	3.07E−04	3.07E−04	3.07E−04	3.07E−04	3.07E−04	3.07E−04	3.35E−04
	Std	2.19E−11	3.38E−11	1.30E−10	2.78E−10	1.02E−10	7.23E−11	6.56E−12	2.16E−12	5.91E−12	7.98E−12	1.03E−04
	Rank	5	6	9	10	8	7	4	**1**	2	3	11
F16	Mean	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00
	Std	5.98E−16	5.68E−16	6.12E−16	5.76E−16	5.61E−16	5.22E−16	5.53E−16	5.83E−16	5.68E−16	5.68E−16	5.68E−16
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F17	Mean	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F18	Mean	4.80E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	6.60E+00
	Std	6.85E+00	1.62E−15	1.51E−15	1.79E−15	1.95E−15	1.74E−15	7.71E−11	2.00E−15	1.10E−15	1.28E−15	9.34E+00
	Rank	10	6	2	4	8	4	9	6	**1**	2	11
F19	Mean	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00
	Std	2.37E−15	2.42E−15	2.43E−15	2.34E−15	2.46E−15	2.42E−15	2.45E−15	2.46E−15	2.46E−15	2.39E−15	2.42E−15
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F20	Mean	−3.27E+00	−3.27E+00	−3.26E+00	−3.30E+00	−3.28E+00	−3.27E+00	−3.28E+00	−3.27E+00	−3.27E+00	−3.27E+00	−3.28E+00
	Std	5.92E−02	5.99E−02	6.03E−02	4.84E−02	5.70E−02	5.99E−02	5.83E−02	5.92E−02	5.99E−02	5.92E−02	5.83E−02
	Rank	5	8	11	**1**	2	8	3	5	8	5	3
F21	Mean	−1.02E+01	−1.02E+01	−1.02E+01	−1.02E+01	−1.02E+01	−1.02E+01	−1.02E+01	−1.02E+01	−1.02E+01	−1.02E+01	−8.96E+00
	Std	5.63E−15	5.56E−15	5.51E−15	5.44E−15	5.78E−15	5.56E−15	5.63E−15	5.46E−15	5.55E−15	5.46E−15	2.19E+00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	11
F22	Mean	−1.04E+01	−1.04E+01	−1.04E+01	−1.04E+01	−1.04E+01	−1.04E+01	−1.04E+01	−1.04E+01	−1.04E+01	−1.04E+01	−9.16E+00
	Std	8.73E−16	8.73E−16	8.08E−16	4.66E−16	8.08E−16	8.73E−16	7.38E−16	9.33E−16	7.38E−16	1.09E−15	2.29E+00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	11
F23	Mean	−1.05E+01	−1.05E+01	−1.05E+01	−1.05E+01	−1.05E+01	−1.05E+01	−1.05E+01	−1.05E+01	−1.05E+01	−1.05E+01	−9.45E+00
	Std	2.11E−15	2.24E−15	1.92E−15	1.78E−15	1.75E−15	2.03E−15	1.75E−15	2.09E−15	2.01E−15	1.78E−15	2.20E+00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	11

approximating the global optimum is approximately fifty percent higher than the current original solution, and reverse learning strategies have been used to improve other algorithms with success [49–51]. The mathematical model of the reverse learning strategy is described as follows:

$$X_i^o = LB + UB - X_i^t, \tag{8}$$

where X_i^o is the inverse solution corresponding to X_i^t. The population diversity is related to the distribution of individuals in the search space. The more uniform the distribution of the individuals, the better the diversity. Chaotic mappings are characterized by random selection and ergodicity, which can help RSA generate new solutions and avoid premature convergence, and chaotic mappings have been successfully used to improve other algorithms [52]. Therefore, this paper combines a reverse learning strategy with chaotic mappings, called the chaotic reverse learning strategy, and it is given as follows:

$$X_i^{co} = LB + UB - \lambda_i X_i^t, \tag{9}$$

where X_i^{co} denotes the solution generated by the chaotic reverse learning mechanism corresponding to the i^{th} individual in the population. λ_i is the corresponding chaotic mapping value. There are ten common chaotic mappings, with the formulas and numerical distributions shown in Table 1.

For swarm-based algorithms, the quality of the initial population has a significant impact on the algorithm's performance. Therefore, the initial population is first generated using COBL to improve the population quality and to increase the algorithm's convergence accuracy. Second, during each iteration, the corresponding reverse population is generated using COBL and evaluated separately to retain the dominant individuals in the next generation.

In addition, as the algorithm proceeds, there will be many useless searches using the chaotic inverse learning strategy for all the individuals, which increases the computational cost and is not conducive to the convergence of the algorithm, so this paper proposes using the linear decreasing population strategy. As the iteration proceeds, the number of individuals using the chaotic inverse learning strategy is gradually reduced, and the specific mathematical formula is as follows:

$$Pop = \text{round}\left(\frac{(pop_{\min} - pop_{\max}) \cdot t}{T} + pop_{\max}\right), \tag{10}$$

where Pop denotes the number of populations using the chaotic backward learning strategy and $pop_{\max}$ and $pop_{\min}$ denote the maximum number and minimum number of populations, respectively.

3.2. Elite Alternative Pool Strategy.

RSA performs position updates by following the best individual. This facilitates a faster convergence of the algorithm but diminishes population diversity and tends to trap local optimums. To maintain a balance between the exploitation and exploration of the algorithm, an elite alternative pooling strategy is proposed in this section. We place the current best three individuals into a pool as shown in the following equation:

Table 4: Parameter setting for comparison algorithm.

Algorithm	Parameters
HHO [42]	$\beta = 1.5, E_0 \in [-1, 1]$
EO [43]	$a_1 = 2, a_2 = 1$
TSA [44]	$P_{\max} = 4, P_{\min} = 1$
GWO [36]	$a = 2$ (linearly decreased over iterations)
SSA [45]	$c_1 = rand$, $c_2 = rand$
WOA [37]	$a_1 = 2$ (linearly decreased over iterations)
RSA [41]	$\alpha = 0.1, \beta = 0.1$

$$X_{eap} = \left\{X_{eap1}, X_{eap2}, X_{eap3}\right\}, \tag{11}$$

where X_{eap1}, X_{eap2}, and X_{eap3} are the three best individuals in the population thus far. The food source is chosen randomly from these three individuals each time. By using the elite alternative pooling strategy, the position of the food source changes from the best individual to one of the best three individuals. This goes some way to avoiding the premature convergence of the algorithm due to the best individual falling into a local optimum. To better balance the development and exploration of the algorithm, we also put the globally optimal individuals into the elite alternative pool to ensure that each individual has the opportunity to move closer to the optimal individual and ensure the convergence efficiency of the algorithm. Thus, the final mathematical model of the elite alternative pooling strategy is described as follows:

$$\mathbf{X}_{eap} = \left\{\mathbf{X}_{eap1}, \mathbf{X}_{eap2}, \mathbf{X}_{eap3}, \mathbf{X}_{\text{best}}\right\}. \tag{12}$$

3.3. Shifted Distribution Estimation Strategy.

RSA searches by following the optimal individuals, ignoring valid information from other individuals. To make full use of the position information of the dominant population, some scholars use a distribution estimation strategy for implementation [53, 54]. This strategy uses the current dominant population to calculate a probability distribution model, generates a new offspring population based on the sampling of the probability distribution model, and eventually obtains the optimal solution through continuous iteration. In addition to using the dominant population, this paper considers a modification of it by introducing information about the optimal individual and its own position and proposes a shifted distribution estimation strategy. The mathematical model is as follows:

$$\begin{aligned}
X_i^{t+1} &= \mathbf{mean} + \mathbf{y}, \mathbf{y} \sim N(0, \mathbf{Cov}),\\
\mathbf{mean} &= \frac{\left(X_{\text{best}}^t + X_{\text{mean}}^t + X_i^t\right)}{3},\\
\mathbf{Cov}(i) &= \frac{1}{NP/2}\sum_{i=1}^{NP/2}\left(X_i^{t+1} - X_{\text{mean}}^t\right) \times \left(X_i^t - X_{\text{mean}}^t\right)^T,\\
X_{\text{mean}}^t &= \sum_{i=1}^{NP/2} \omega_i \times X_i^t,\\
\omega_i &= \frac{\ln\,(0.5NP + 0.5) - \ln(i)}{\sum_{i=1}^{NP/2}(\ln(0.5NP + 0.5) - \ln i)}
\end{aligned} \tag{13}$$

Table 5: Test results of different algorithms for F1–F13 (Dim = 30).

30D	MRSA	HHO [42]	EO [43]	TSA [44]	GWO [36]	SSA [45]	WOA [37]	RSA [41]
F1	**0.00E + 00**	1.06E − 187	1.68E − 72	6.24E − 27	2.09E − 34	1.14E − 02	2.32E − 166	2.10E − 182
F2	**1.29E − 248**	4.21E − 100	2.03E − 42	1.12E − 17	6.77E − 21	1.43E + 01	5.11E − 107	8.28E − 93
F3	**0.00E + 00**	5.81E − 135	5.43E − 06	7.48E + 02	7.43E − 01	1.91E + 04	7.02E + 05	6.13E − 116
F4	**7.53E − 256**	3.88E − 96	1.87E − 11	2.95E + 01	6.86E − 05	2.05E + 01	7.27E + 01	4.55E − 109
F5	**3.89E − 07**	3.67E − 03	9.41E + 01	9.80E + 01	9.72E + 01	6.74E + 02	9.73E + 01	4.69E − 05
F6	**4.26E − 10**	3.74E − 05	9.21E − 02	1.36E + 01	7.86E + 00	1.64E − 02	5.45E − 01	4.76E − 08
F7	6.11E − 05	**4.18E − 05**	7.93E − 04	1.32E − 02	1.76E − 03	8.03E − 01	8.11E − 04	2.38E − 04
F8	−3.60E + 04	**−4.19E + 04**	−2.92E + 04	−1.42E + 04	−1.61E + 04	−2.38E + 04	−4.01E + 04	−2.47E + 04
F9	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	9.12E + 02	2.23E − 01	1.34E + 02	3.79E − 15	**0.00E + 00**
F10	**8.88E − 16**	**8.88E − 16**	7.99E − 15	4.96E − 14	6.97E − 14	5.14E + 00	4.91E − 15	**8.88E − 16**
F11	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	2.04E − 03	2.57E − 03	1.01E − 01	**0.00E + 00**	**0.00E + 00**
F12	**9.70E − 12**	3.36E − 07	4.13E − 04	1.01E + 01	1.83E − 01	1.15E + 01	5.96E − 03	3.51E − 10
F13	**3.42E − 10**	1.51E − 05	2.30E + 00	1.17E + 01	5.60E + 00	1.61E + 02	7.29E − 01	2.92E − 08

Table 6: Test results of different algorithms for F1–F13 (Dim = 100).

100D	MRSA	HHO [42]	EO [43]	TSA [44]	GWO [36]	SSA [45]	WOA [37]	RSA [41]
F1	**0.00E + 00**	3.63E − 193	7.51E − 73	1.25E − 27	3.22E − 34	1.03E − 02	4.14E − 166	9.57E − 183
F2	**3.08E − 251**	4.29E − 100	2.29E − 42	1.10E − 17	7.27E − 21	1.23E + 01	3.27E − 106	2.05E − 101
F3	**0.00E + 00**	1.47E − 142	2.51E − 06	1.03E + 03	9.92E − 01	2.04E + 04	6.76E + 05	1.78E − 122
F4	**1.01E − 254**	4.63E − 93	2.04E − 11	2.98E + 01	4.34E − 04	2.07E + 01	7.47E + 01	8.33E − 93
F5	**1.86E − 07**	4.44E − 03	9.39E + 01	9.81E + 01	9.72E + 01	5.52E + 02	9.72E + 01	3.67E − 05
F6	**2.52E − 11**	2.76E − 05	1.56E − 01	1.38E + 01	7.65E + 00	6.61E − 03	5.85E − 01	4.85E − 08
F7	7.56E − 05	**3.24E − 05**	8.65E − 04	1.63E − 02	1.75E − 03	7.38E − 01	1.66E − 03	2.54E − 04
F8	−3.36E + 04	**−4.19E + 04**	−2.92E + 04	−1.45E + 04	−1.62E + 04	−2.38E + 04	−3.72E + 04	−2.50E + 04
F9	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	9.28E + 02	4.94E − 01	1.29E + 02	3.79E − 15	**0.00E + 00**
F10	**8.88E − 16**	**8.88E − 16**	7.88E − 15	8.02E − 02	6.93E − 14	5.04E + 00	4.56E − 15	**8.88E − 16**
F11	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	2.25E − 03	1.61E − 03	1.07E − 01	2.41E − 03	**0.00E + 00**
F12	**9.43E − 13**	2.55E − 07	2.02E − 04	9.78E + 00	1.80E − 01	1.13E + 01	5.40E − 03	7.39E − 10
F13	**2.25E − 10**	1.12E − 05	2.07E + 00	1.19E + 01	5.72E + 00	1.57E + 02	6.03E − 01	1.13E − 08

Table 7: Test results of different algorithms for F1–F13 (Dim = 500).

500D	MRSA	HHO [42]	EO [43]	TSA [44]	GWO [36]	SSA [45]	WOA [37]	RSA [41]
F1	**0.00E + 00**	2.61E − 195	1.13E − 72	7.72E − 28	1.82E − 34	6.70E − 03	1.27E − 165	7.20E − 175
F2	**1.13E − 250**	3.07E − 99	2.64E − 42	9.40E − 18	7.22E − 21	1.31E + 01	1.54E − 108	7.77E − 103
F3	**0.00E + 00**	3.88E − 121	3.93E − 05	5.72E + 02	3.37E − 01	1.67E + 04	7.11E + 05	5.41E − 136
F4	**1.47E − 259**	3.18E − 98	9.92E − 03	3.19E + 01	6.68E − 05	2.01E + 01	8.02E + 01	2.27E − 105
F5	**4.99E − 07**	3.65E − 03	9.40E + 01	9.83E + 01	9.71E + 01	5.06E + 02	9.72E + 01	3.68E − 05
F6	**8.10E − 11**	6.80E − 05	1.33E − 01	1.36E + 01	7.81E + 00	7.64E − 03	5.83E − 01	5.17E − 08
F7	7.12E − 05	**5.01E − 05**	7.65E − 04	1.54E − 02	1.62E − 03	7.27E − 01	1.79E − 03	2.23E − 04
F8	−3.40E + 04	**−4.19E + 04**	−2.90E + 04	−1.44E + 04	−1.63E + 04	−2.45E + 04	−3.99E + 04	−2.46E + 04
F9	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	8.69E + 02	7.10E − 02	1.37E + 02	**0.00E + 00**	**0.00E + 00**
F10	**8.88E − 16**	**8.88E − 16**	7.99E − 15	1.78E − 01	6.98E − 14	4.97E + 00	3.85E − 15	**8.88E − 16**
F11	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	3.16E − 03	3.28E − 04	1.10E − 01	**0.00E + 00**	**0.00E + 00**
F12	**6.72E − 12**	3.29E − 07	9.68E − 04	1.03E + 01	1.85E − 01	1.15E + 01	5.97E − 03	2.03E − 10
F13	**9.35 E − 11**	8.81E − 06	2.23E + 00	1.17E + 01	5.63E + 00	1.55E + 02	6.80E − 01	3.90E − 08

where X^t_{mean} denotes the weighted position of the dominant population and ω_i denotes the weight coefficient in the dominant population in descending order of fitness values. **Cov** is the weighted covariance matrix of the dominant populations. The flow chart of MRSA is shown in Figure 1 and its pseudocode is in Algorithm 2.

3.4. Time Complexity. The time complexity determines the operating efficiency of the algorithm. In RSA, the computational complexity of the initialization process is $O(N)$, where N is the population size. The computational complexity of the update process is $O(T \times N) + O(T \times N \times D)$, where D is the problem's dimensionality and T is the

Table 8: Test results of different algorithms for F1–F13 (Dim = 1000).

1000D	MRSA	HHO [42]	EO [43]	TSA [44]	GWO [36]	SSA [45]	WOA [37]	RSA [41]
F1	**0.00E + 00**	7.31E − 188	6.27E − 56	3.27E − 08	5.24E − 10	1.19E + 05	8.86E − 163	1.80E − 186
F2	**2.83E − 236**	6.65E − 99	2.70E − 33	3.03E − 07	2.64E − 05	8.48E + 02	2.51E − 105	1.52E − 86
F3	**0.00E + 00**	6.88E − 83	2.98E + 04	3.99E + 06	5.80E + 05	2.34E + 06	1.03E + 08	1.38E − 157
F4	**4.64E − 266**	1.59E − 94	8.56E + 01	9.95E + 01	7.27E + 01	3.63E + 01	8.08E + 01	1.09E − 97
F5	**5.37E − 05**	3.62E − 02	9.96E + 02	9.91E + 03	9.97E + 02	3.03E + 07	9.91E + 02	4.88E − 04
F6	**4.95E − 07**	3.61E − 04	1.69E + 02	2.06E + 02	2.02E + 02	1.18E + 05	2.32E + 01	5.34E − 06
F7	7.24E − 05	**5.54E − 05**	1.59E − 03	1.92E + 00	1.30E − 02	4.38E + 02	1.67E − 03	3.87E − 04
F8	−3.82E + 05	**−4.19E + 05**	−1.57E + 05	−4.98E + 04	−1.08E + 05	−1.30E + 05	−4.04E + 05	−1.60E + 05
F9	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	1.10E + 04	1.09E + 01	5.61E + 03	1.21E − 13	**0.00E + 00**
F10	**8.88E − 16**	**8.88E − 16**	1.07E − 14	7.83E − 06	7.54E − 07	1.29E + 01	3.97E − 15	**8.88E − 16**
F11	**0.00E + 00**	**0.00E + 00**	8.88E − 17	1.77E − 02	3.16E − 03	1.08E + 03	3.70E − 18	**0.00E + 00**
F12	**1.75E − 10**	1.50E − 07	5.43E − 01	3.82E + 06	8.37E − 01	6.96E + 04	1.89E − 02	1.82E − 09
F13	**8.09E − 08**	4.45E − 05	9.90E + 01	4.51E + 05	9.50E + 01	1.41E + 07	1.04E + 01	7.07E − 07

Table 9: Test results of different algorithms for F14–F23.

	MRSA	HHO [42]	EO [43]	TSA [44]	GWO [36]	SSA [45]	WOA [37]	RSA [41]
F14	1.13E + 00	1.06E + 00	**9.98E − 01**	8.28E + 00	3.00E + 00	**9.98E − 01**	1.78E + 00	5.34E + 00
F15	**3.07E − 04**	3.46E − 04	3.68E − 03	5.17E − 03	3.04E − 03	7.66E − 04	6.61E − 04	3.21E − 04
F16	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**
F17	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**
F18	**3.00E + 00**	**3.00E + 00**	**3.00E + 00**	8.40E + 00	**3.00E + 00**	**3.00E + 00**	**3.00E + 00**	5.70E + 00
F19	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**
F20	**−3.27E + 00**	−3.17E + 00	−3.26E + 00	−3.24E + 00	**−3.27E + 00**	−3.22E + 00	−3.22E + 00	−3.26E + 00
F21	**−1.02E + 01**	−5.22E + 00	−9.05E + 00	−6.23E + 00	−9.48E + 00	−8.47E + 00	−9.56E + 00	−9.81E + 00
F22	**−1.04E + 01**	−5.44E + 00	**−1.04E + 01**	−7.62E + 00	−1.00E + 01	−9.97E + 00	−9.08E + 00	**−1.04E + 01**
F23	**−1.05E + 01**	−5.67E + 00	−9.91E + 00	−6.05E + 00	−1.04E + 01	−1.04E + 01	−9.64E + 00	−1.04E + 01

maximum number of iterations, so the computational complexity of RSA is $O(N \times (T \times D + T + 1))$.

The computational complexity of MRSA is determined by six main factors (initialization process, solution update, number of fitness evaluations, chaotic reverse learning strategy, elite alternative pooling strategy, and shifted distribution estimation strategy). The computational complexity of the MRSA initialization process is $O(N)$. The computational complexity of the update process is $O(T \times N) + O(2 \times T \times N \times D)$. Therefore, the computational complexity of MRSA is $O(N \times (2 \times T \times D + T + 1))$. The introduction of three improved strategies causes the computational complexity of MRSA to increase slightly compared to RSA. RSA and MRSA can be considered to have similar levels of operating efficiency.

4. Experimental Results and Discussion

In this section, we first evaluate various chaotic mapping combination algorithms using benchmark test functions and then determine which chaotic mapping sequence to use in combination with the adaptive reverse learning strategy. The performance of MRSA is then evaluated, and 23 benchmark functions, IEEE CEC2017 benchmark functions, and robot path planning problems are compared with other state-of-the-art algorithms.

4.1. Benchmark Test Functions. This section uses 23 benchmark test functions that are commonly found in the literature. These benchmark test functions include seven unimodal functions, six multimodal functions, and ten fixed dimensional functions [55]. Unimodal functions F1–F7 have only one global optimum and are primarily used to test the local exploitation capabilities of the algorithms. The multimodal functions have multiple local minimums and can be used to check the global exploration capability and local optimum avoidance capability of the algorithm. Details of the benchmark test functions are shown in Table 2.

4.2. Chaos Mapping Selection Test. The adaptive chaotic reverse learning strategy proposed in this paper combines a chaotic mapping and a reverse learning mechanism. To verify which chaotic mapping is employed, each of the 10 chaotic mappings is combined with a reverse learning mechanism. The MRSA using the chaotic mapping with ID 1 is called MRSA-C1. The rest of the MRSA algorithms using chaotic mappings are named similarly. For a fair comparison, the number of populations was set to 50, and the maximum number of iterations was set to 300 on the same experimental platform. All the algorithms were programmed using MATLAB R2016b, the computer operating system was Windows 10, and the processor was AMD R5 3600 × 16 GB. Table 3 shows the statistical results of each algorithm run

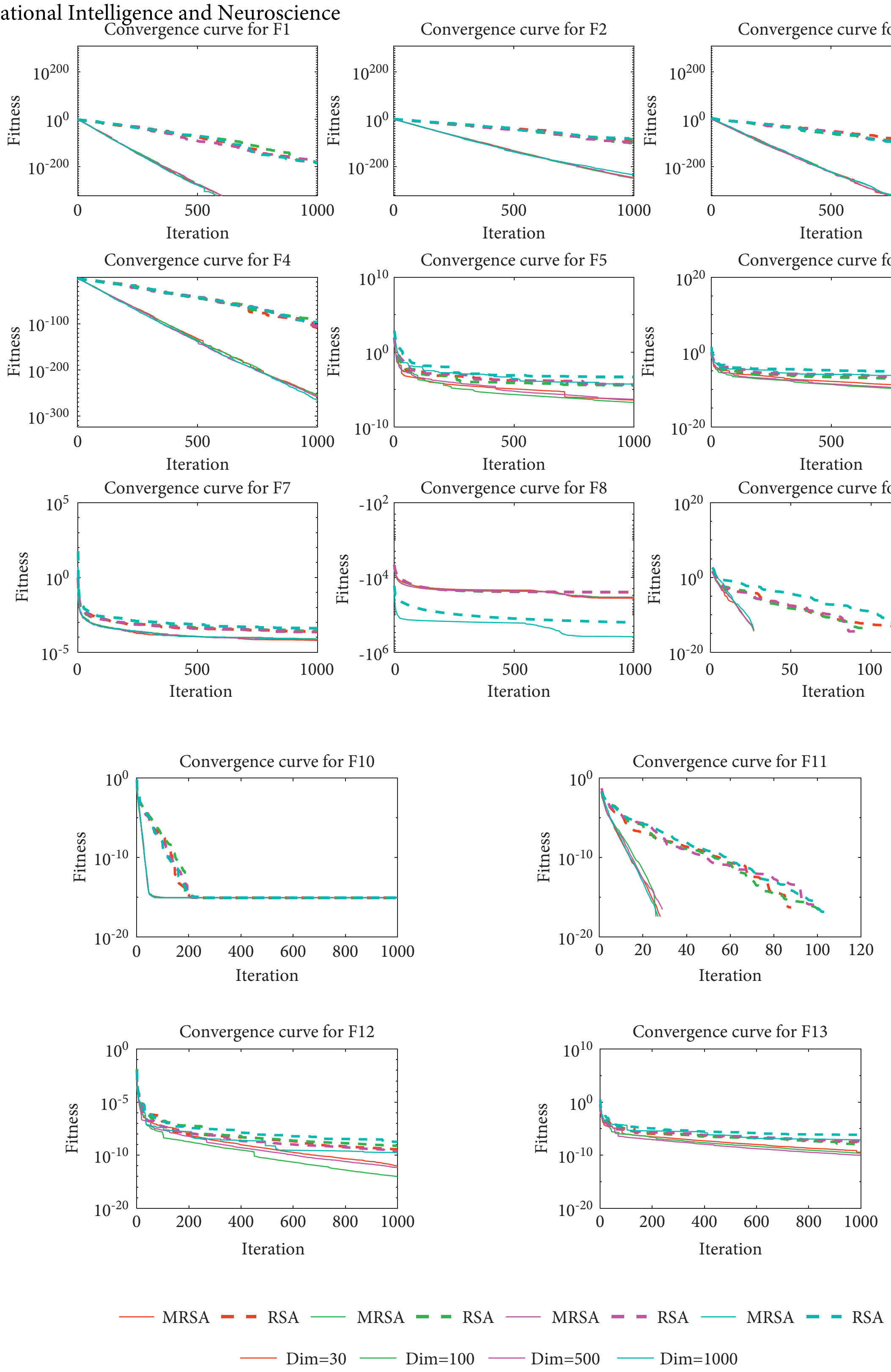

Figure 2: Convergence curves for different dimensional functions.

independently 30 times. In presenting the simulation results, “avg” is the average of the best candidate solutions obtained, and “std” is the standard deviation of these values.

As shown in Table 3, MRSA-C1 (Chebyshev map) and MRSA-C9 (Sinusoidal map) outperformed RSA on 10 out of 23 functions. MRSA-C2 (Circle map), MRSA-C3 (Gauss map), MRSA-C5 (Logistic map), MRSA-C6 (Pricewise map), and MRSA-C7 (Sine map) outperformed RSA in 9 functions. MRSA-C4 (Iterative map) outperformed RSA in 12 functions. MRSA-C8 (Singer map) performed better than

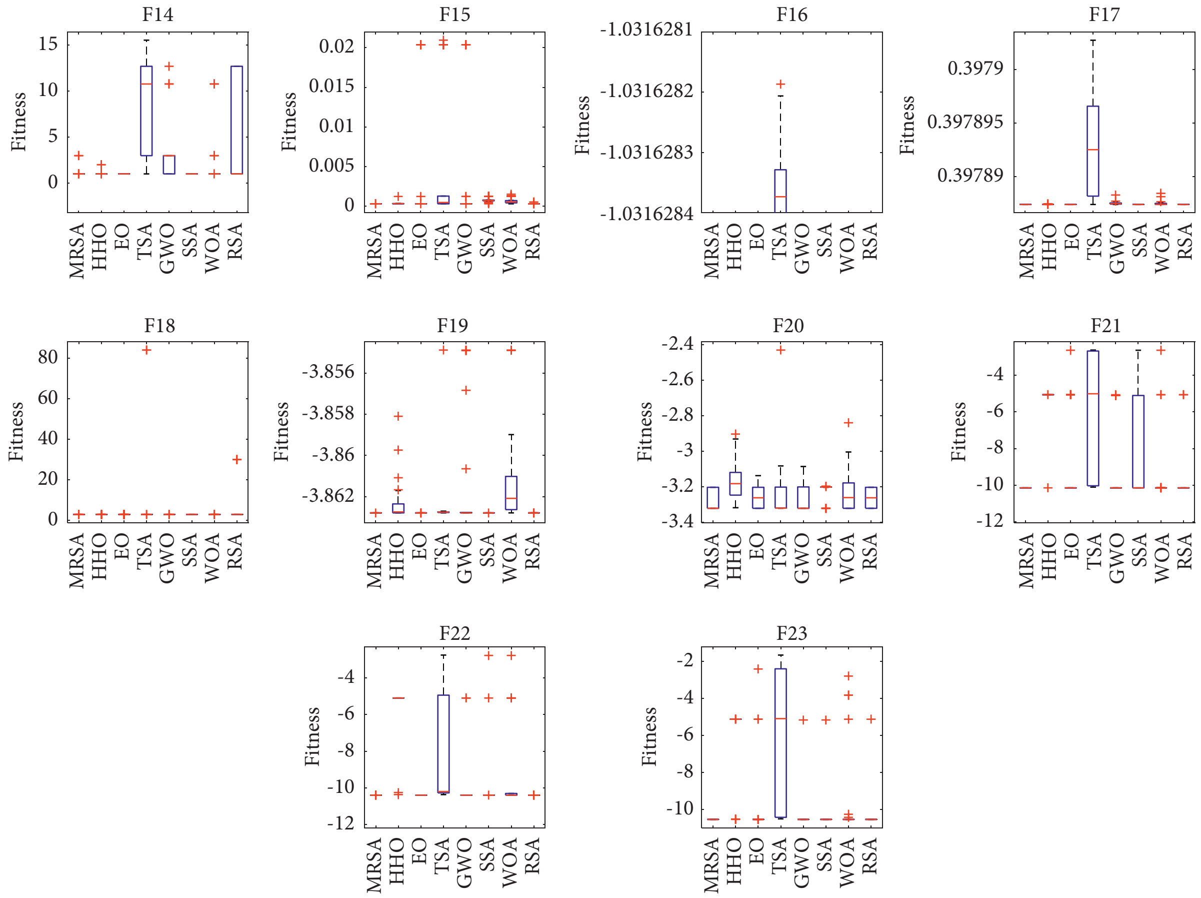

Figure 3: Box plot of different test functions.

Table 10: The Friedman test results for different algorithms.

Function	Type	MRSA	HHO [42]	EO [43]	TSA [44]	GWO [36]	SSA [45]	WOA [37]	RSA [41]
F1–F13	Dim = 30	1.58	2.42	4.19	7.00	6.08	7.23	4.77	2.73
	Dim = 100	1.54	2.38	4.15	7.08	5.85	7.23	5.08	2.69
	Dim = 500	1.62	2.62	4.31	7.15	5.77	7.23	4.69	2.62
	Dim = 1000	1.50	2.42	4.88	7.23	5.92	7.23	4.31	2.50
F14–F23	Fixed dim	1.75	5.80	3.05	7.10	5.20	4.20	5.50	3.40
All cases		**1.59**	3.00	4.17	7.11	5.79	6.74	4.84	2.76

RSA in 11 functions. MRSA-C10 (Tent map) outperformed RSA on 16 out of 23 functions. Remarkably, all the improved algorithms perform no worse than RSA in at least 22 functions, indicating that the improved strategies proposed in this paper effectively improve the algorithm's performance. Furthermore, the best of the ten chaotic mapping combination algorithms is MRSA-C10. Therefore, the MRSA-C10 algorithm was used to participate in the tests in the comparison that followed.

4.3. *Performance Comparison Tests of MRSA with Other Advanced Algorithms on 23 Benchmark Functions.*

To verify the performance of the MRSA algorithm, the modified algorithm was compared with the original RSA [41], HHO [42], EO [43], TSA [44], GWO [36], SSA [45], and WOA [37]. The parameters of all the algorithms were set according to the original paper to ensure the performance of the comparison algorithms, as shown in Table 4. Given that F1–F13 are the multidimensional functions used in this section, the thirteen functions were solved under Dim = 30, 100, 500, and 1000. The means obtained by these algorithms are recorded, as shown in Tables 5–8.

The results in Tables 5–8 show that MRSA achieves better results in most of the functions. Specifically, MRSA obtains satisfactory results for the unimodal functions F1–F7 in both the low and high dimensions. MRSA achieves a stable optimal value of 0 when solving for F1 and F3 and remains so as the dimensionality increases. For the other unimodal functions, HHO outperforms MRSA in solving F7. The unimodal function results show that MRSA outperforms RSA in all the functions, and MRSA does not show

Table 11: The Wilcoxon signed-rank test results for different algorithms.

Functions type	Comparison	p value	$\alpha = 0.05$
F1–F13 (Dim = 30)	MRSA versus HHO [42]	**0.182338**	No
	MRSA versus EO [43]	0.001871	Yes
	MRSA versus TSA [44]	0.001306	Yes
	MRSA versus GWO [36]	0.001306	Yes
	MRSA versus SSA [45]	0.001306	Yes
	MRSA versus WOA [37]	0.02537	Yes
	MRSA versus RSA [41]	0.002873	Yes
F1–F13 (Dim = 100)	MRSA versus HHO [42]	**0.209427**	No
	MRSA versus EO [43]	0.001871	Yes
	MRSA versus TSA [44]	0.001306	Yes
	MRSA versus GWO [36]	0.001306	Yes
	MRSA versus SSA [45]	0.001306	Yes
	MRSA versus WOA [37]	0.017496	Yes
	MRSA versus RSA [41]	0.002873	Yes
F1–F13 (Dim = 500)	MRSA versus HHO [42]	**0.182338**	No
	MRSA versus EO [43]	0.001871	Yes
	MRSA versus TSA [44]	0.001306	Yes
	MRSA versus GWO [36]	0.001306	Yes
	MRSA versus SSA [45]	0.001306	Yes
	MRSA versus WOA [37]	0.02313	Yes
	MRSA versus RSA [41]	0.002873	Yes
F1–F13 (Dim = 1000)	MRSA versus HHO [42]	**0.157939**	No
	MRSA versus EO [43]	0.001944	Yes
	MRSA versus TSA [44]	0.001306	Yes
	MRSA versus GWO [36]	0.001306	Yes
	MRSA versus SSA [45]	0.001306	Yes
	MRSA versus WOA [37]	0.017496	Yes
	MRSA versus RSA [41]	0.002873	Yes
F14–F23	MRSA versus HHO [42]	**0.083131**	No
	MRSA versus EO [43]	**0.192518**	No
	MRSA versus TSA [44]	0.010862	Yes
	MRSA versus GWO [36]	0.032969	Yes
	MRSA versus SSA [45]	**0.126279**	No
	MRSA versus WOA [37]	0.019059	Yes
	MRSA versus RSA [41]	0.019059	Yes

a significant decrease in performance as the dimensionality increases, which indicates that the improvement strategy proposed in this paper greatly improves the development capability of RSA. For the variable dimensional multimodal functions F8–F13, MRSA, RSA, HHO, and EO consistently achieve their respective optimal solutions in different dimensions when solving F9–F11. HHO and WOA outperform MRSA in solving F8. MRSA achieves a stable optimal value of 0 when solving for F9 and F11 and remains so as the dimensionality increases. MRSA shows the best performance in solving F12 and F13, outperforming all the compared algorithms. It is worth noting that MRSA does not perform any less than RSA in all the multimodal functions in the different dimensions and has significant improvements in three of the six variable dimensional multimodal functions, which indicates that MRSA has a better global search capability, and the improvement strategy proposed in this paper is well suited to enhance the population diversity and to expand the search range of the population, thus improving the exploration capability of the algorithm.

Table 9 presents the test results when different algorithms solve the fixed dimensional multimodal function. The comparison shows that HHO, EO, and SSA outperform MRSA on F14. For F15–F23, MRSA performs best in all the tested functions. In particular, MRSA provides better solutions in all the test functions compared to RSA. Since fixed-dimension functions are usually used to test the ability of an algorithm to maintain a balance between development and exploration, the above analysis shows that the MRSA proposed in this paper is able to balance the development and exploration capabilities effectively and has a strong local optimum avoidance capability.

The convergence speed and convergence accuracy are important indicators of the performance of the algorithm. Figure 2 shows the mean convergence curves of MRSA and RSA when solving the test functions in different dimensions. It can be seen that MRSA has a faster convergence speed and a better convergence accuracy in different dimensions. Moreover, the convergence speed and convergence accuracy of MRSA do not decrease much with increasing

Table 12: Summary of CEC2017 benchmark functions [41].

Type	Number	Function name	$f_i(x^*)$
Unimodal	3	Shifted and rotated Zakharov function	300
Multimodal	4	Shifted and rotated Rosenbrock's function	400
	5	Shifted and rotated Rastrigin's function	500
	6	Shifted and rotated expanded Scaffer's F6 function	600
	7	Shifted and rotated Lunacek bi-Rastrigin function	700
	8	Shifted and rotated noncontinuous Rastrigin's function	800
	9	Shifted and rotated Levy function	900
	10	Shifted and rotated Schwefel's function	1000
Hybrid	11	Hybrid function 1 ($N=3$)	1100
	12	Hybrid function 2 ($N=3$)	1200
	13	Hybrid function 3 ($N=3$)	1300
	14	Hybrid function 4 ($N=4$)	1400
	15	Hybrid function 5 ($N=4$)	1500
	16	Hybrid function 6 ($N=4$)	1600
	17	Hybrid function 6 ($N=5$)	1700
	18	Hybrid function 6 ($N=5$)	1800
	19	Hybrid function 6 ($N=5$)	1900
	20	Hybrid function 6 ($N=6$)	2000
Composition	21	Composition function 1 ($N=3$)	2100
	22	Composition function 2 ($N=3$)	2200
	23	Composition function 3 ($N=4$)	2300
	24	Composition function 4 ($N=4$)	2400
	25	Composition function 5 ($N=5$)	2500
	26	Composition function 6 ($N=5$)	2600
	27	Composition function 7 ($N=6$)	2700
	28	Composition function 8 ($N=6$)	2800
	29	Composition function 9 ($N=3$)	2900
	30	Composition function 10 ($N=3$)	3000

dimensionality, which indicates that the improvement strategy proposed in this paper can effectively improve the convergence ability of RSA and thus obtain better optimization results.

To analyze the distribution characteristics of each algorithm in the fixed dimensional test function, box plots were drawn based on the results obtained by solving F14–F23, as shown in Figure 3. For each algorithm, the center mark of each box indicates the median of the results of 30 runs, and the bottom and top edges of each box indicate the trivial and quartiles, respectively. “The “+” sign indicates bad values that are not inside the box. As seen from Figure 3, MRSA has no bad values for F17 and F21–F23, which indicates that the distribution of the solutions obtained by MRSA is more concentrated and MRSA is more stable. For the other test functions with some bad values, MRSA outperforms the comparison algorithm in terms of maximum, minimum, and median values, and the distribution of the solutions obtained by MRSA is more concentrated compared to the comparison algorithm. Therefore, MRSA solves the test function with better stability compared to the other comparison algorithms.

Apart from the convergence and stability analysis, to further analyze the experimental results, the Friedman test and Wilcoxon's signed-rank test were used for multiple comparisons in this paper. Table 10 is the Friedman test showing the average ranking results of each algorithm. The overall ranking value of MRSA is 1.59, which ranks first among all the algorithms. The remaining seven algorithms are ranked as follows: RSA, HHO, EO, WOA, GWO, SSA, and TSA. In solving F1–F13 in different dimensions, MRSA is ranked first, and HHO and RSA are ranked second and third, respectively. For fixed dimensions F14–F23, MRSA, EO, and RSA ranked in the top three. In either case, MRSA ranks better than RSA. The results of Wilcoxon's signed-rank test are shown in Table 11. In the case of F1–F13 (Dim = 30, 100, 500, and 1000), MRSA outperformed EO, TSA, GWO, SSA, WOA, and RSA at the 0.05 significance level, but there was no significant difference between MRSA and HHO. In the case of F14–F23, MRSA outperformed TSA, GWO, WOA, and RSA at the 0.05 significance level, but there was no significant difference between MRSA and HHO, EO, or SSA, which statistically proves that the improvement strategy proposed in this paper can effectively help MRSA balance the exploitation and exploration capabilities and has a better local optimal avoidance ability.

Table 13: Parameters setting.

Algorithm	Parameters setting
BOA [38]	$a = 0.1, c = 0.01, p = 0.6$
HHO [42]	$\beta = 1.5, E_0 \in [-1, 1]$
AOA [47]	$Mop_{\max} = 1, Mop_{\min} = 0.2, C = 1, a = 5, Mu = 0.499$
SSA [45]	$c_1 = \text{rand}, c_2 = \text{rand}$
PFA [48]	$u_1 = -1 + 2\text{rand}, u_2 = -1 + 2\text{rand}$
TDO [46]	~

Table 14: Statistical results of seven algorithms in the CEC2017 test.

		BOA [38]	HHO [42]	AOA [47]	SSA [45]	PFA [48]	TDO [46]	MRSA
F3	Mean	3.82*E* + 04	1.68*E* + 03	6.91*E* + 04	8.40*E* + 04	4.66*E* + 04	3.77*E* + 04	3.01*E* − 06
	Std	6.97*E* + 03	7.95*E* + 02	1.15*E* + 04	6.59*E* + 03	1.22*E* + 04	3.65*E* + 03	5.54*E* − 07
	Rank	4	2	6	7	5	3	**1**
F4	Mean	9.33*E* + 03	1.23*E* + 02	7.61*E* + 03	1.44*E* + 03	9.80*E* + 01	4.96*E* + 02	7.24*E* + 01
	Std	1.29*E* + 03	3.33*E* + 01	2.45*E* + 03	1.09*E* + 03	1.77*E* + 01	2.16*E* + 01	2.96*E* + 01
	Rank	7	3	6	5	2	4	**1**
F5	Mean	3.49*E* + 02	2.05*E* + 02	2.95*E* + 02	3.50*E* + 02	1.14*E* + 02	6.19*E* + 02	8.24*E* + 01
	Std	2.16*E* + 01	3.62*E* + 01	3.20*E* + 01	4.40*E* + 01	3.11*E* + 01	1.56*E* + 01	2.43*E* + 01
	Rank	5	3	4	6	2	7	**1**
F6	Mean	6.63*E* + 01	5.62*E* + 01	6.21*E* + 01	8.06*E* + 01	1.47*E* + 01	6.00*E* + 02	2.03*E* + 01
	Std	5.76*E* + 00	5.92*E* + 00	6.71*E* + 00	8.84*E* + 00	4.99*E* + 00	3.05*E* − 02	5.92*E* + 00
	Rank	5	3	4	6	**1**	7	2
F7	Mean	5.57*E* + 02	4.98*E* + 02	6.00*E* + 02	7.12*E* + 02	1.34*E* + 02	8.49*E* + 02	1.46*E* + 02
	Std	3.17*E* + 01	6.57*E* + 01	5.66*E* + 01	6.85*E* + 01	3.12*E* + 01	1.47*E* + 01	3.38*E* + 01
	Rank	4	3	5	6	**1**	7	2
F8	Mean	2.93*E* + 02	1.40*E* + 02	2.25*E* + 02	2.72*E* + 02	9.97*E* + 01	9.22*E* + 02	7.56*E* + 01
	Std	1.54*E* + 01	2.13*E* + 01	2.67*E* + 01	4.31*E* + 01	2.66*E* + 01	1.58*E* + 01	2.31*E* + 01
	Rank	6	3	4	5	2	7	**1**
F9	Mean	6.82*E* + 03	4.69*E* + 03	4.50*E* + 03	9.35*E* + 03	2.28*E* + 02	9.01*E* + 02	4.00*E* + 02
	Std	8.69*E* + 02	8.28*E* + 02	7.24*E* + 02	1.85*E* + 03	1.83*E* + 02	8.01*E* − 01	2.11*E* + 02
	Rank	6	5	4	7	**1**	3	2
F10	Mean	7.33*E* + 03	4.35*E* + 03	5.51*E* + 03	7.05*E* + 03	4.98*E* + 03	5.15*E* + 03	3.90*E* + 03
	Std	2.85*E* + 02	7.25*E* + 02	5.83*E* + 02	7.45*E* + 02	9.01*E* + 02	3.62*E* + 02	6.06*E* + 02
	Rank	7	2	5	6	3	4	**1**
F11	Mean	2.19*E* + 03	1.61*E* + 02	1.72*E* + 03	3.91*E* + 03	1.91*E* + 02	1.17*E* + 03	5.60*E* + 01
	Std	6.72*E* + 02	4.86*E* + 01	9.74*E* + 02	1.64*E* + 03	5.28*E* + 01	2.47*E* + 01	2.69*E* + 01
	Rank	6	2	5	7	3	4	**1**
F12	Mean	2.08*E* + 09	7.61*E* + 06	6.27*E* + 09	4.69*E* + 08	1.88*E* + 06	1.85*E* + 05	8.83*E* + 02
	Std	7.43*E* + 08	4.21*E* + 06	2.56*E* + 09	3.76*E* + 08	1.97*E* + 06	9.78*E* + 04	7.27*E* + 02
	Rank	6	4	7	5	3	2	**1**
F13	Mean	3.15*E* + 08	1.51*E* + 05	3.80*E* + 04	8.55*E* + 07	7.54*E* + 04	1.21*E* + 04	2.24*E* + 02
	Std	2.10*E* + 08	9.05*E* + 04	1.71*E* + 04	4.66*E* + 08	4.12*E* + 04	5.52*E* + 03	1.60*E* + 02
	Rank	7	5	3	6	4	2	**1**
F14	Mean	1.19*E* + 05	3.82*E* + 04	5.72*E* + 04	1.50*E* + 06	3.00*E* + 04	2.97*E* + 03	4.32*E* + 01
	Std	7.62*E* + 04	4.25*E* + 04	4.92*E* + 04	1.21*E* + 06	2.94*E* + 04	6.93*E* + 02	1.13*E* + 01
	Rank	6	4	5	7	3	2	**1**
F15	Mean	1.82*E* + 06	6.86*E* + 04	2.35*E* + 04	1.83*E* + 07	3.35*E* + 04	2.40*E* + 03	2.98*E* + 01
	Std	1.46*E* + 06	4.86*E* + 04	1.22*E* + 04	2.37*E* + 07	1.77*E* + 04	5.08*E* + 02	1.67*E* + 01
	Rank	6	5	3	7	4	2	**1**
F16	Mean	3.18*E* + 03	1.55*E* + 03	1.98*E* + 03	2.74*E* + 03	1.00*E* + 03	2.40*E* + 03	1.07*E* + 03
	Std	4.12*E* + 02	3.56*E* + 02	5.09*E* + 02	5.38*E* + 02	2.63*E* + 02	1.34*E* + 02	3.20*E* + 02
	Rank	7	3	4	6	**1**	5	2
F17	Mean	1.22*E* + 03	7.48*E* + 02	9.12*E* + 02	1.20*E* + 03	3.77*E* + 02	1.88*E* + 03	4.41*E* + 02
	Std	2.49*E* + 02	2.19*E* + 02	2.67*E* + 02	3.85*E* + 02	1.71*E* + 02	3.61*E* + 01	2.10*E* + 02
	Rank	6	3	4	5	**1**	7	2
F18	Mean	9.60*E* + 05	6.90*E* + 05	1.29*E* + 06	1.51*E* + 07	2.75*E* + 05	6.11*E* + 04	3.11*E* + 01
	Std	6.22*E* + 05	8.77*E* + 05	1.60*E* + 06	1.51*E* + 07	2.82*E* + 05	2.10*E* + 04	5.54*E* + 00
	Rank	5	4	6	7	3	2	**1**
F19	Mean	4.61*E* + 06	1.46*E* + 05	1.08*E* + 06	4.23*E* + 07	4.45*E* + 04	5.14*E* + 03	2.29*E* + 01
	Std	4.06*E* + 06	1.42*E* + 05	1.39*E* + 05	1.23*E* + 08	3.91*E* + 04	9.69*E* + 02	5.65*E* + 00
	Rank	6	4	5	7	3	2	**1**
F20	Mean	7.29*E* + 02	6.71*E* + 02	6.94*E* + 02	8.59*E* + 02	4.61*E* + 02	2.29*E* + 03	5.30*E* + 02
	Std	9.88*E* + 01	2.01*E* + 02	1.54*E* + 02	2.42*E* + 02	1.52*E* + 02	4.33*E* + 01	1.62*E* + 02
	Rank	5	3	4	6	**1**	7	2

Table 14: Continued.

		BOA [38]	HHO [42]	AOA [47]	SSA [45]	PFA [48]	TDO [46]	MRSA
F21	Mean	1.97E + 02	4.06E + 02	4.87E + 02	5.06E + 02	2.90E + 02	2.41E + 03	2.83E + 02
	Std	3.01E + 01	3.51E + 01	5.23E + 01	5.36E + 01	2.62E + 01	9.32E + 00	2.52E + 01
	Rank	**1**	4	5	6	3	7	2
F22	Mean	4.71E + 02	2.39E + 03	5.13E + 03	4.18E + 03	2.08E + 02	2.30E + 03	1.02E + 02
	Std	7.76E + 01	2.37E + 03	1.21E + 03	1.88E + 03	7.61E + 02	8.09E + 00	2.05E + 00
	Rank	3	5	7	6	2	4	**1**
F23	Mean	6.97E + 02	7.05E + 02	9.68E + 02	8.60E + 02	4.85E + 02	2.72E + 03	5.11E + 02
	Std	5.59E + 01	7.35E + 01	9.10E + 01	1.00E + 02	4.09E + 01	1.33E + 01	4.21E + 01
	Rank	3	4	6	5	**1**	7	2
F24	Mean	1.10E + 03	8.26E + 02	1.14E + 03	8.99E + 02	5.24E + 02	2.88E + 03	5.83E + 02
	Std	1.68E + 02	7.42E + 01	1.09E + 02	1.37E + 02	3.72E + 01	1.03E + 01	5.29E + 01
	Rank	5	3	6	4	**1**	7	2
F25	Mean	1.75E + 03	4.11E + 02	1.67E + 03	7.84E + 02	3.97E + 02	2.89E + 03	3.93E + 02
	Std	2.01E + 02	1.87E + 01	4.55E + 02	1.30E + 02	1.72E + 01	1.09E + 01	1.35E + 01
	Rank	6	3	5	4	2	7	**1**
F26	Mean	5.21E + 03	3.94E + 03	6.40E + 03	6.30E + 03	2.12E + 03	3.07E + 03	2.38E + 03
	Std	1.49E + 03	1.10E + 03	7.22E + 02	1.11E + 03	7.10E + 02	4.75E + 02	4.59E + 02
	Rank	5	4	7	6	**1**	3	2
F27	Mean	8.14E + 02	6.05E + 02	1.34E + 03	9.56E + 02	5.46E + 02	3.21E + 03	5.98E + 02
	Std	9.81E + 01	4.00E + 01	2.14E + 02	1.65E + 02	2.85E + 01	6.74E + 00	3.98E + 01
	Rank	4	3	6	5	**1**	7	2
F28	Mean	3.28E + 03	4.62E + 02	2.95E + 03	1.06E + 03	4.31E + 02	3.23E + 03	4.09E + 02
	Std	3.99E + 02	2.60E + 01	6.15E + 02	3.22E + 02	1.90E + 01	2.10E + 01	3.06E + 01
	Rank	7	3	5	4	2	6	**1**
F29	Mean	3.04E + 03	1.32E + 03	2.43E + 03	2.64E + 03	1.03E + 03	3.67E + 03	9.64E + 02
	Std	4.72E + 02	2.56E + 02	5.22E + 02	6.35E + 02	2.35E + 02	6.51E + 01	2.18E + 02
	Rank	6	3	4	5	2	7	**1**
F30	Mean	3.98E + 07	1.01E + 06	1.47E + 07	4.85E + 07	4.09E + 05	6.59E + 03	2.14E + 03
	Std	2.31E + 07	6.08E + 05	1.01E + 07	3.75E + 07	4.56E + 05	3.95E + 02	1.05E + 02
	Rank	6	4	5	7	3	2	**1**

Table 15: The Friedman test results for different algorithms.

Algorithm	Ranking
MRSA	**1.3929**
PFA [48]	2.2143
HHO [42]	3.5714
TDO [46]	5.2857
AOA [47]	5.6429
BOA [38]	6
SSA [45]	6.6071

Table 16: The Wilcoxon signed-rank test results for different algorithms.

Comparison	p value	$\alpha = 0.05$
MRSA versus BOA [38]	0.000004	YES
MRSA versus HHO [42]	0.000004	YES
MRSA versus AOA [47]	0.000004	YES
MRSA versus SSA [45]	0.000004	YES
MRSA versus PFA [48]	0.039321	YES
MRSA versus TDO [46]	0.000004	YES

4.4. Performance Comparison Tests of MRSA with Other Advanced Algorithms on CEC2017.

To further verify the superior performance of the MRSA algorithm, the algorithm was tested using the IEEE CEC2017 [41] single objective test function defined in Table 12. In this section, six recently proposed algorithms were evaluated for comparison with MRSA. These state-of-the-art algorithms are BOA [38], HHO [42], AOA [47], SSA [45], PFA [48], and TDO [46]. For a fair comparison, all the algorithm parameters are set the same as those used by the authors of the original literature, as shown in Table 13. The dimension of the CEC2017 benchmark functions was set to 30 on the same experimental platform. Table 14 shows the statistical results of each algorithm run independently 51 times.

From the analysis in Table 14, we know that, for the unimodal test function F3, MRSA outperformed all the comparison algorithms, and although MRSA could not stably obtain the optimal solution, it performed the best among all the comparison algorithms, indicating that MRSA has a stronger exploitation ability. For the multi-peaked test functions F4–F10, MRSA performs best among the four test functions (F4, F5, F8, and F10), while PFA achieves the best results on F6, F7, and F9, with MRSA ranking second in all cases. The performance of MRSA on

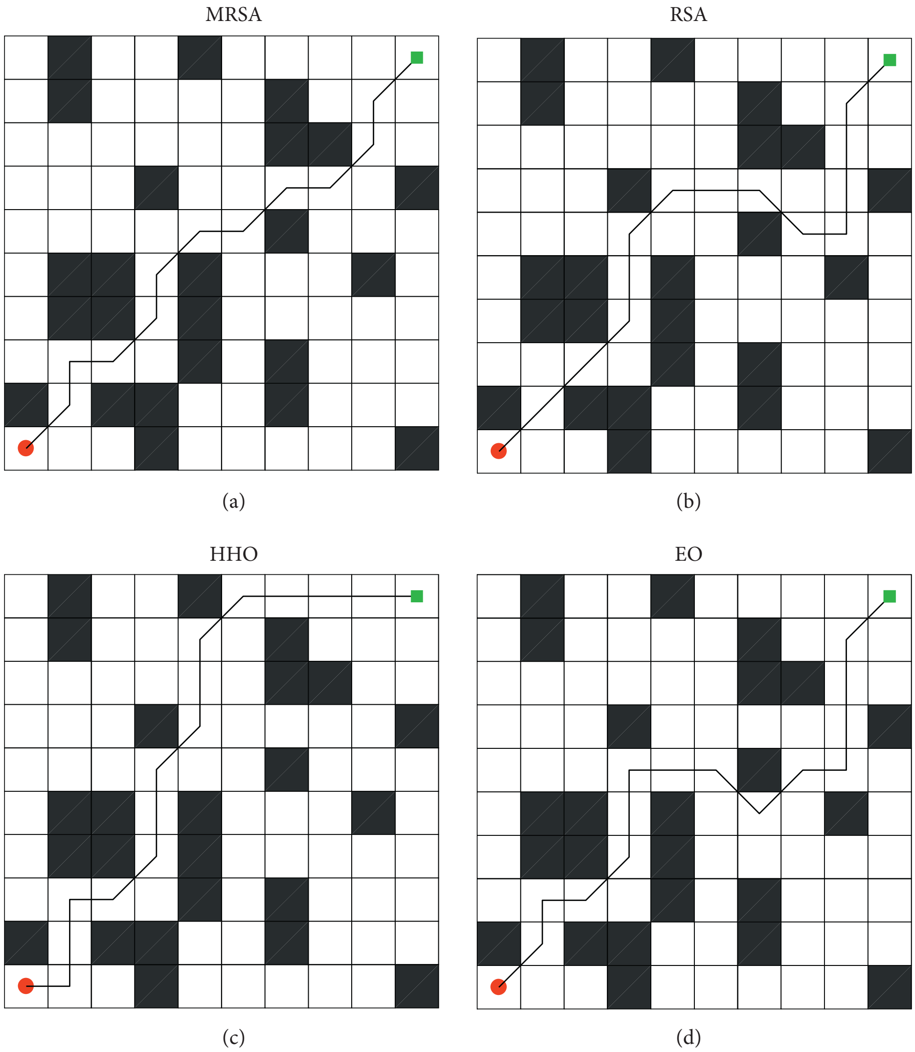

Figure 4: Path planning diagram: (a) MRSA. (b) RSA. (c) HHO. (d) EO.

the multipeaked functions indicates that the improved algorithm can maintain sufficient population diversity to avoid falling into local optima. For complex and combinatorial functions, each algorithm has its advantages and disadvantages. MRSA obtains optimal solutions on F11–F15, F18-F19, F22, F25, and F28–F30. PFA achieves better solutions on F16-F17, F20, F23-F24, and F26-F27. BOA outperforms the other comparison algorithms on F21. Overall, MRSA achieves the top two results in both the complex and combinatorial functions, better demonstrating the potential of MRSA to solve complex optimization problems in the real world.

To perform a statistical analysis on the performance of MRSA and the six competing algorithms, the Friedman test and Wilcoxon's signed-rank test were used for multiple comparisons in this paper. Table 15 is the Friedman test results showing the average ranking of each algorithm. The overall ranking value of MRSA is 1.3929, which ranks first among all the algorithms. The remaining six algorithms are ranked as follows: PFA, HHO, TDO, AOA, BOA, and SSA. The results of Wilcoxon's signed-rank test are shown in Table 16. MRSA outperformed BOA, AOA, SSA, PFA, and TDO at the 0.05 significance level, which statistically proves that the improvement strategy proposed in this paper can effectively help MRSA balance the exploitation and exploration capabilities and has a better local optimal avoidance ability.

4.5. Robot Path Planning Based on MRSA. To verify the performance of the improved strategy, MRSA is applied to solve the robot path planning in this paper. Each crocodile represents a possible path. It is assumed that there are N possible paths, and the dimension D is determined by the number of connections from the starting point to the destination point. The environment is modeled using the raster method, and the raster values are used to equate to the obstacles at the location. The robot's working environment is equated to a plane, similar to a lattice effect, and then the feasible and obstacle zones are determined based on the raster values. The grid number 0 is defined as the feasible area, and 1 is defined as the obstacle area. The robot can walk

Table 17: Robot root planning results.

Index	MRSA	RSA [41]	HHO [42]	EO [43]
Best	12.7279	15.5563	12.7279	15.5563
Mean	**14.4250**	16.4049	15.2735	17.8191
Worst	15.5563	21.2132	18.3848	21.2132
Std	1.4606	1.9090	1.6055	2.2311

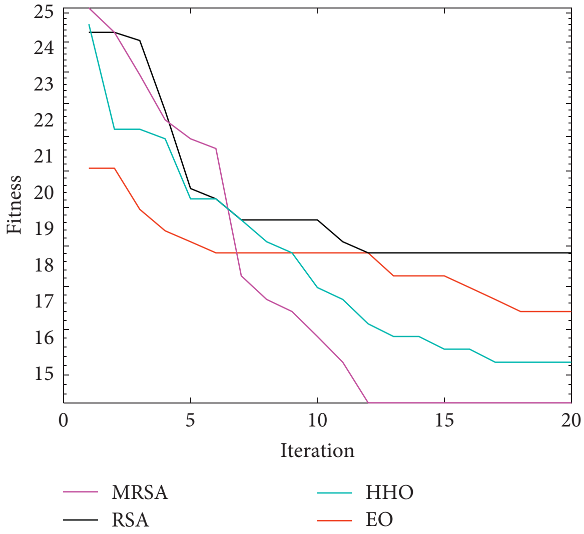

Figure 5: Convergence curves of four algorithms.

on the grid designated as 0. The cost function for the i^{th} crocodile is shown below:

$$\cos t_i = \sum_{j=1}^{D-1} \sqrt{\left(x_{j+1} + x_j\right)^2 + \left(y_{j+1} + y_j\right)^2}, \qquad (14)$$

where j denotes the j^{th} dimension of each crocodile. In robot path planning, the population size is 100, and the number of iterations is 20. RSA [41], HHO [42], and EO [43] are used as competitors. Each algorithm works in a 10 × 10 model, and the optimal route is shown in Figure 4. To eliminate chance, each algorithm was run 10 times, and the mean, optimal, and worst values of each algorithm were recorded. The statistical results of each algorithm are shown in Table 17.

As shown in Figure 4, MRSA has the shortest route, followed by HHO, while EO and RSA are clearly trapped in a local optimum. As seen from Table 17, MRSA is the best among all the algorithms in terms of best cost, mean cost, and worst cost. This indicates that MRSA can consistently provide excellent solutions. Figure 5 shows the convergence curves of the four algorithms. MRSA has the fastest convergence speed and a higher convergence accuracy. Therefore, the introduction of multiple strategies makes the algorithm more comprehensive in its search, which greatly improves the search capability of MRSA and plans the least costly route.

5. Conclusion

This paper proposes a novel variant of the reptile search algorithm, called MRSA. First, the adaptive chaotic reverse learning strategy combines the advantages of the reverse learning mechanism and chaotic mapping to enhance the population diversity. Second, the elite alternative pool strategy balances the exploitation and exploration capabilities by controlling the reference points followed by the population. Finally, the shifted distribution estimation strategy makes full use of the dominant population information to guide the direction of individual evolution, thus improving the performance of RSA. The superiority of MRSA was verified in 23 benchmark functions, IEEE CEC2017 benchmark functions, and robot path planning problems. The experimental results show that the adaptive chaotic reverse learning strategy can effectively improve the population diversity, among which tent mapping is the most effective. The MRSA outperforms the comparison algorithm in terms of convergence accuracy, convergence speed, and stability. The results of the multimodal functions F8–F23 among the 23 benchmark functions show that the elite alternative pool strategy balances algorithm exploitation and exploration effectively and prevents the algorithm from falling into a local optimum. The adaptive chaotic reverse learning strategy enhances the population diversity. The shifted distribution estimation strategy enhances the convergence speed and convergence accuracy of the algorithm by learning information about the dominant populations. In addition, the test results were analyzed using the Friedman test and the Wilcoxon signed-rank test. The statistical results show that MRSA is significantly more effective than the comparison algorithm.

In a subsequent study, we plan to examine the following issues: First, the shifted distribution estimation strategy increases the computational cost of MRSA. Optimizing the algorithm structure and performance needs further investigation and discussion. Second, the capacity and composition of the elite replacement pool need to be further analyzed. Additionally, MRSA can be extended to multiobjective and binary versions. We will consider solving problems in image processing, industry, neural networks, text, and data mining as real-world optimization problems.

Data Availability

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

Conflicts of Interest

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

Acknowledgments

This work was supported in part by the Reserve Leader Funding Project of Leading Talent Echelon of Heilongjiang Province of China under Grant 2501050628 and in part by the Science and Technology Innovation Team Project of Foshan City of China under Grant 2018IT100302.

References

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

[2] R. Abbassi, A. Abbassi, A. A. Heidari, and S. Mirjalili, “An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models,” *Energy Conversion and Management*, vol. 179, pp. 362–372, 2019.

[3] Y. Zhou, H. X. Wang, and H. Liu, “Generalized function projective synchronization of incommensurate fractional-order chaotic systems with inputs saturation,” *International Journal of Fuzzy Systems*, vol. 21, no. 3, pp. 823–836, 2019.

[4] Y. Zhou, H. Liu, J. D. Cao, and S. G. Li, “Composite learning fuzzy synchronization for incommensurate fractional-order chaotic systems with time-varying delays,” *International Journal of Adaptive Control and Signal Processing*, vol. 33, no. 12, pp. 1739–1758, 2019.

[5] B. Jin, L. Cruz, and N. Goncalves, “Deep facial diagnosis: deep transfer learning from face recognition to facial diagnosis,” *IEEE Access*, vol. 8, pp. 123649–123661, 2020.

[6] B. Jin, L. Cruz, and N. Goncalves, “Face depth prediction by the scene depth,” in *Proceedings of the 2021 IEEE/ACIS 19th International Conference on Computer and Information Science (ICIS)*, Shanghai, China, June 2021.

[7] X. Y. Zhou, J. X. Lu, J. H. Huang, M. S. Zhong, and M. W. Wang, “Enhancing artificial bee colony algorithm with multi-elite guidance,” *Information Sciences*, vol. 543, pp. 242–258, 2021.

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

[9] 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.

[10] M. Alweshah, “Solving feature selection problems by combining mutation and crossover operations with the monarch butterfly optimization algorithm,” *Applied Intelligence*, vol. 51, no. 6, pp. 4058–4081, 2021.

[11] M. Alweshah, S. A. 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*, vol. 34, no. 14, pp. 11267–11281, 2020.

[12] X. K. Lin and Y. H. Wu, “Parameters identification of photovoltaic models using niche-based particle swarm optimization in parallel computing architecture,” *Energy*, vol. 196, Article ID 117054, 2020.

[13] A. D. Tang, T. Han, H. Zhou, and L. Xie, “An improved equilibrium optimizer with application in unmanned aerial vehicle path planning,” *Sensors*, vol. 21, no. 5, p. 1814, 2021.

[14] Z. Xu, H. C. Yang, J. Y. Li, X. Y. Zhang, B. Lu, and S. C. Gao, “Comparative study on single and multiple chaotic maps incorporated grey wolf optimization algorithms,” *IEEE Access*, vol. 9, pp. 77416–77437, 2021.

[15] C. Charin, D. Ishak, M. A. A. Mohd Zainuri, B. Ismail, and M. K. Mohd Jamil, “A hybrid of bio-inspired algorithm based on levy flight and particle swarm optimizations for photovoltaic system under partial shading conditions,” *Solar Energy*, vol. 217, pp. 1–14, 2021.

[16] Y. Yu, S. C. Gao, S. Cheng, Y. R. Wang, S. Y. Song, and F. G. Yuan, “CBSO: a memetic brain storm optimization with chaotic local search,” *Memetic Computing*, vol. 10, no. 4, pp. 353–367, 2018.

[17] B. Morales-Castaneda, D. Zaldivar, E. Cuevas, F. Fausto, and A. Rodriguez, “A better balance in metaheuristic algorithms: does it exist,” *Swarm and Evolutionary Computation*, vol. 54, 2020.

[18] Z. H. Cai, S. C. Gao, X. Yang, G. Yang, S. Cheng, and Y. H. Shi, “Alternate search pattern-based brain storm optimization,” *Knowledge-Based Systems*, vol. 238, Article ID 107896.

[19] A. Aleti and I. Moser, “A systematic literature review of adaptive parameter control methods for evolutionary algorithms,” *ACM Computing Surveys*, vol. 49, no. 3, pp. 1–35, 2017.

[20] Z. Y. Lei, S. C. Gao, S. Gupta, J. J. Cheng, and G. Yang, “An aggregative learning gravitational search algorithm with self-adaptive gravitational constants,” *Expert Systems with Applications*, vol. 152, Article ID 113396, 2020.

[21] X. X. Zhong and P. Cheng, “An elite-guided hierarchical differential evolution algorithm,” *Applied Intelligence*, vol. 51, no. 7, pp. 4962–4983, 2021.

[22] Y. R. Wang, S. C. Gao, M. C. Zhou, and Y. Yu, “A multi-layered gravitational search algorithm for function optimization and real-world problems,” *IEEE-CAA Journal of Automatica Sinica*, vol. 8, no. 1, pp. 94–109, 2021.

[23] S. Gao, Y. R. Wang, J. H. Wang, and J. J. Cheng, “Understanding differential evolution: a Poisson law derived from population interaction network,” *Journal of Computational Science*, vol. 21, pp. 140–149, 2017.

[24] L. Xie, T. Han, H. Zhou, Z. R. Zhang, B. Han, and A. D. 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.

[25] A. Yaghoubzadeh-Bavandpour, O. Bozorg-Haddad, M. Rajabi, B. Zolghadr-Asli, and X. F. Chu, “Application of swarm intelligence and evolutionary computation algorithms for optimal reservoir operation,” *Water Resources Management*, vol. 36, no. 7, pp. 2275–2292, 2022.

[26] X. W. Xia, L. Gui, Y. L. Zhang et al., “A fitness-based adaptive differential evolution algorithm,” *Information Sciences*, vol. 549, pp. 116–141, 2021.

[27] S. Kumar and A. Sikander, “Optimum mobile robot path planning using improved artificial bee colony algorithm and evolutionary programming,” *Arabian Journal for Science and Engineering*, vol. 47, no. 3, pp. 3519–3539, 2022.

[28] M. M. Rosso, R. Cucuzza, A. Aloisio, and G. C. Marano, “Enhanced multi-strategy particle swarm optimization for constrained problems with an evolutionary-strategies-based unfeasible local search operator,” *Applied Sciences*, vol. 12, no. 5, p. 2285, 2022.

[29] K. P. Wang, X. Y. Li, L. Gao, P. G. Li, and S. M. Gupta, “A genetic simulated annealing algorithm for parallel partial disassembly line balancing problem,” *Applied Soft Computing*, vol. 107, Article ID 107404.

[30] D. Pelusi, R. Mascella, L. Tallini, J. Nayak, B. Naik, and Y. Deng, “Improving exploration and exploitation via a hyperbolic gravitational search algorithm,” *Knowledge-Based Systems*, vol. 193, Article ID 105404, 2020.

[31] Z. L. Wei, C. Q. Huang, X. F. Wang, T. Han, and Y. T. Li, “Nuclear reaction optimization: a novel and powerful physics-based algorithm for global optimization,” *IEEE Access*, vol. 7, pp. 66084–66109, 2019.

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

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

[34] W. Li, X. Meng, Y. Huang, and Z. H. Fu, "Multipopulation cooperative particle swarm optimization with a mixed mutation strategy," *Information Sciences*, vol. 529, pp. 179–196, 2020.

[35] Q. Luo, H. B. Wang, Y. Zheng, and J. C. He, "Research on path planning of mobile robot based on improved ant colony algorithm," *Neural Computing & Applications*, vol. 32, no. 6, pp. 1555–1566, 2020.

[36] M. H. Nadimi-Shahraki, S. Taghian, and S. Mirjalili, "An improved grey wolf optimizer for solving engineering problems," *Expert Systems with Applications*, vol. 166, 2021.

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

[38] S. Arora and S. Singh, "Butterfly optimization algorithm: a novel approach for global optimization," *Soft Computing*, vol. 23, no. 3, pp. 715–734, 2019.

[39] A. M. Altabeeb, A. M. Mohsen, L. Abualigah, and A. Ghallab, "Solving capacitated vehicle routing problem using cooperative firefly algorithm," *Applied Soft Computing*, vol. 108, Article ID 107403.

[40] H. Wang, W. J. Wang, X. Y. Zhou et al., "Artificial bee colony algorithm based on knowledge fusion," *Complex & Intelligent Systems*, vol. 7, no. 3, pp. 1139–1152, 2021.

[41] L. Abualigah, M. A. Elaziz, P. Sumari, Z. W. Geem, and A. H. Gandomi, "Reptile search algorithm (RSA): a nature-inspired meta-heuristic optimizer," *Expert Systems with Applications*, vol. 191, Article ID 116158, 2022.

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

[43] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili, "Equilibrium optimizer: a novel optimization algorithm," *Knowledge-Based Systems*, vol. 191, Article ID 105190, 2020.

[44] S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, "Tunicate swarm algorithm: a new bio-inspired based metaheuristic paradigm for global optimization," *Engineering Applications of Artificial Intelligence*, vol. 90, 2020.

[45] M. Tubishat, S. Ja'afar, M. Alswaitti et al., "Dynamic salp swarm algorithm for feature selection," *Expert Systems with Applications*, vol. 164, Article ID 113873.

[46] M. Dehghani, S. Hubalovsky, and P. Trojovsky, "Tasmanian devil optimization: a new bio-inspired optimization algorithm for solving optimization algorithm," *IEEE Access*, vol. 10, pp. 19599–19620, 2022.

[47] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 376, Article ID 113609.

[48] H. Yapici and N. Cetinkaya, "A new meta-heuristic optimizer: pathfinder algorithm," *Applied Soft Computing*, vol. 78, pp. 545–568, 2019.

[49] A. D. Tang, S. Q. Tang, T. Han, H. Zhou, and L. Xie, "A modified slime mould algorithm for global optimization," *Computational Intelligence and Neuroscience*, vol. 2021, 22 pages, 2021.

[50] T. Y. Zheng and W. L. Luo, "An enhanced lightning attachment procedure optimization with quasi-opposition-based learning and dimensional search strategies," *Computational Intelligence and Neuroscience*, vol. 2019, pp. 1–24, Article ID 1589303, 2019.

[51] S. Wang, Q. X. Liu, Y. X. Liu et al., "A hybrid ssa and sma with mutation opposition-based learning for constrained engineering problems," *Computational Intelligence and Neuroscience*, vol. 2021, pp. 1–21, Article ID 6379469, 2021.

[52] L. Yang, S. C. Gao, H. C. Yang, Z. H. Cai, Z. Y. Lei, and Y. Todo, "Adaptive chaotic spherical evolution algorithm," *Memetic Computing*, vol. 13, no. 3, pp. 383–411, 2021.

[53] A. D. Tang, H. Zhou, T. Han, and L. Xie, "A modified manta ray foraging optimization for global optimization problems," *IEEE Access*, vol. 9, pp. 128702–128721, 2021.

[54] X. F. 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*, vol. 78, pp. 240–260, 2019.

[55] X. X. Han, Q. X. Xu, L. Yue, Y. C. Dong, G. Xie, and X. Y. Xu, "An improved crow search algorithm based on spiral search mechanism for solving numerical and engineering optimization problems," *IEEE Access*, vol. 8, pp. 1–92382, 2020.