mathematics

MDPI

Article

Elite Chaotic Manta Ray Algorithm Integrated with Chaotic Initialization and Opposition-Based Learning

Jianwei Yang [1], Zhen Liu [1], Xin Zhang [1] and Gang Hu [2,3,*]

[1] Design Art College, Xijing University, Xi'an 710123, China
[2] Department of Applied Mathematics, Xi'an University of Technology, Xi'an 710054, China
[3] School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China
* Correspondence: hugang@xaut.edu.cn

Abstract: The manta ray foraging optimizer (MRFO) is a novel nature-inspired optimization algorithm that simulates the foraging strategy and behavior of manta ray groups, i.e., chain, spiral, and somersault foraging. Although the native MRFO has revealed good competitive capability with popular meta-heuristic algorithms, it still falls into local optima and slows the convergence rate in dealing with some complex problems. In order to ameliorate these deficiencies of the MRFO, a new elite chaotic MRFO, termed the CMRFO algorithm, integrated with chaotic initialization of population and an opposition-based learning strategy, is developed in this paper. Fourteen kinds of chaotic maps with different properties are used to initialize the population. Thereby, the chaotic map with the best effect is selected; meanwhile, the sensitivity analysis of an elite selection ratio in an elite chaotic searching strategy to the CMRFO is discussed. These strategies collaborate to enhance the MRFO in accelerating overall performance. In addition, the superiority of the presented CMRFO is comprehensively demonstrated by comparing it with a native MRFO, a modified MRFO, and several state-of-the-art algorithms using (1) 23 benchmark test functions, (2) the well-known IEEE CEC 2020 test suite, and (3) three optimization problems in the engineering field, respectively. Furthermore, the practicability of the CMRFO is illustrated by solving a real-world application of shape optimization of cubic generalized Ball (CG-Ball) curves. By minimizing the curvature variation in these curves, the shape optimization model of CG-Ball ones is established. Then, the CMRFO algorithm is applied to handle the established model compared with some advanced meta-heuristic algorithms. The experimental results demonstrate that the CMRFO is a powerful and attractive alternative for solving engineering optimization problems.

Keywords: manta ray foraging optimizer; chaotic map; opposition-based learning; elite chaotic search; CG-Ball curves; shape optimization

MSC: 49K35; 68T20

Citation: Yang, J.; Liu, Z.; Zhang, X.; Hu, G. Elite Chaotic Manta Ray Algorithm Integrated with Chaotic Initialization and Opposition-Based Learning. *Mathematics* **2022**, *10*, 2960. https://doi.org/10.3390/math10162960

Academic Editors: Shi Qiang Liu, Erhan Kozan, Felix T. S. Chan and Weidong Li

Received: 25 July 2022
Accepted: 13 August 2022
Published: 16 August 2022

1. Introduction

Many complex problems to be solved in life can be described as optimization problems, and the research on high-precision algorithms for optimization problems has attracted many scholars. Traditional mathematical optimization (TMO) methods usually require the objective function of the optimization problem to satisfy convexity and differentiability. This requirement theoretically ensures that TMO methods can approach the optimal solution. However, since the objective functions of most optimization problems tend to be multimodal, discrete, non-differentiable, and non-convex, TMO cannot better handle complex optimization problems. Nowadays, swarm intelligence algorithms simulating organisms in nature are often adopted to solve optimization problems in order to achieve the desired purpose.

The concept of swarm intelligence was first introduced by Hackwood et al. [1] and originates from the research on the social behavior of swarm gregarious creatures (such as

birds, fish, and wolves). Swarm intelligence algorithms take advantage of the swarm evolution behavior of creatures. With the help of information sharing and competition mechanisms among populations, the swarm intelligence algorithm makes random exploration and exploitation to search for the optimal objective function solution. In addition, a swarm intelligence algorithm has no strict mathematical conditions on objective function compared with the traditional random search algorithm. When dealing with complex optimization problems, it can quickly search for the global optimal, is simple to operate, and has fast convergence speed. These advantages have made swarm intelligence algorithms popular in a short amount of time. The genetic algorithm (GA) introduced by scholar Holland in 1975 is an optimization technique inspired by natural evolution [2]. The birth of the genetic algorithm promoted the development of swarm intelligence algorithms, making the theoretical research of swarm intelligence algorithms a hot research field. After the development of biology and continuous research on intelligent optimization algorithms, a series of swarm intelligence optimization algorithms with their own characteristics have been put forward, and these algorithms have been widely used in data clustering [3], feature selection [4], economic emission dispatch [5], engineering problems [6], shape optimization [7], and many other application fields.

Since swarm intelligence optimization algorithms were often proposed and introduced with the help of the influence of natural flora and fauna, different swarm intelligence algorithms have focused on different strategies and solved different problems, showing differences and defects, and as they are a reliable means, the introduction of improvement strategies will help different algorithms in different optimization problems. Many scholars have modified different population intelligence algorithms with the help of some enlightening methods and solved some practical engineering problems [8–12]. For example, Elsisi et al. proposed a modified multitracker optimization algorithm by adding contrastive-based learning and quasi-OBL methods and applied it to nonlinear model predictive control [13]. Zheng et al. proposed an improved gray wolf optimization algorithm and applied it to solve Quintic generalized [14]. Elsisi and Abdelfattah proposed a new design of variable structure control based on the lightning search algorithm [15]. Hu et al. proposed an improved chimp optimization algorithm based on a combination of selective opposition and cuckoo search strategies and used for optimal degree reduction of Said-Ball curves [16]. Zhao et al. proposed an improved artificial hummingbird algorithm for solving complex multiobjective optimization problems [17]. In addition to the above, there have been many studies on improving optimization algorithms and their applications in solving practical problems [18–22].

As a representative of excellent meta-heuristic algorithms, the MRFO simulates the foraging strategy and behavior of manta ray groups [23]. In addition, the MRFO owns good global searching ability, high solving efficiency, and strong stability. Therefore, many scholars have used the MRFO to resolve complex optimization problems in practical application fields. For example, Houssein et al. used manta ray foraging optimization to solve the parameter extraction of the three-diode photovoltaic model [24]. Fathy et al. used the MRFO to deal with the robust global MPPT to mitigate partial shading of the triple-junction solar-cell-based system [25]. Ben et al. used the MRFO to deal with the Novel technique for interpreting gravity anomalies over geologic structures with idealized geometries and the Novel methodology for interpreting magnetic anomalies due to two-dimensional dipping dikes [26,27]. El-Hameed et al. used the MRFO to analyze and validate the three-diode model to characterize industrial solar generating units [28]. In addition, the MRFO has been applied to solve optimization of the support vector machine model [29], optimization of distributed generators [30], and parameter extraction of the artificial neural network model [31]. Some scholars have extended the MRFO to solve multiobjective problems. For example, Got et al. proposed an MOMRFO for solving multiobjective problems [32]. Zouache et al. and Abdelaziz proposed guided manta ray foraging optimization using epsilon dominance to solve multiobjective engineering problems [33]. Some scholars have improved it to solve the practical problem of optimal

power flow [34]. The MRFO also has room for further improvement. Many scholars have made relevant improvements to solve complex problems in their fields. There are many improvement methods: (1) Combining the MRFO with various improvement strategies: For example, Elaziz et al. [35,36] used fractional-order algorithms to enhance the MRFO for global optimization and image segmentation. Yousri et al. proposed a Novel memory-based fractional-order Caputo manta ray foraging optimizer [37]. Xu et al. proposed an improved MRFO algorithm for fire-use analysis and optimization of HT-PEMFC [38]. Jena et al. proposed an attacking manta-ray foraging optimization algorithm for handling multilevel thresholding of brain MR images based on maximum 3D Tsallis entropy [39]. Other studies have also worked on improving the MRFO [40–42]. (2) Combining the MRFO with other algorithms: For example, the SA algorithm was fused with the MRFO to obtain the SA–MRFO algorithm [43], the MRFO was combined with the GBO algorithm to obtain the MRFO–GBO [44], and the proposed ROA was combined with the MRFO to obtain the ROA–MRFO [45]. In this work, we develop a novel elite chaotic MRFO, termed the CMRFO, by integrating three different strategies: chaotic initialization of population, opposition-based learning, and elite chaotic searching. Through 23 test functions, the CEC 2020 test suite, three engineering examples, and one real-world application, the effectiveness of the CMRFO is examined by comparing it with the native MRFO, a modified MRFO, and well-known meta-heuristic algorithms.

Furthermore, the practicability of the CMRFO is verified by the shape optimization of parametric curves (i.e., CG-Ball curves). The classical Ball curves are a forceful tool for shape design in many geometric modeling fields, such as industrial design, manufacturing, and path planning. [36]. However, the shape of classical Ball curves is only defined by their control points. This paper constructs the local controlled CG-Ball curves to overcome these weaknesses by generalizing classical cubic Ball curves. Furthermore, a new mathematical model of shape optimization for CG-Ball ones based on minimum curvature variation is established and then the CMRFO is applied to deal with this optimization model to obtain the optimal shape of CG-Ball curves. The innovative points and main contributions of this paper are as follows:

(a) A new manta ray foraging optimizer (CMRFO) based on chaotic initialization, opposition-based learning, and elite chaotic searching is proposed.
(b) The effectiveness of the CMRFO is demonstrated by comparing it with the native MRFO, a modified MRFO, and several advanced algorithms on 23 classical benchmarks and IEEE CEC 2020, as well as three engineering design examples.
(c) A new optimization model of CG-Ball curves based on minimum curvature variation is established, and the CMRFO is adopted to solve this model to certify the superiority of the algorithm.

The rest of this paper is arranged as follows: a new elite chaotic CMRFO is proposed in Section 2. The effectiveness of the CMRFO is demonstrated by comparison with other optimization algorithms using 23 classical benchmarks and the IEEE CEC 2020 test suite in Section 3. Three real-world engineering application problems are presented to verify the superiority of the CMRFO in Section 4. In Section 5, the practicability of the CMRFO is verified by the shape optimization problem of CG-Ball curves. There is a summary of this paper in Section 6.

2. Proposed Chaotic MRFO

2.1. Overview of the MRFO

The MRFO is a new swarm intelligence optimization algorithm that simulates the behaviors of manta ray foraging for plankton, which has the following three foraging behaviors [23].

2.1.1. Chain Foraging (CF)

In the CF strategy, manta rays form a chain to swim straight for the plankton. Except for the first individual in the foraging chain, each individual is updated by the previous and best individual, respectively. The updating formula of $z_i^d(x)$ is given by:

$$\begin{cases} z_1^d(x+1) = z_1^d(x) + r\cdot\left[z_{best}^d(x) - z_1^d(x)\right] + \alpha\cdot\left[z_{best}^d(x) - z_1^d(x)\right], \\ z_i^d(x+1) = z_i^d(x) + r\cdot\left[z_{i-1}^d(x) - x_i^d(x)\right] + \alpha\cdot\left[z_{best}^d(x) - z_i^d(x)\right], \quad i = 2,\ 3,\ \ldots,\ M \\ \alpha = 2r\cdot\sqrt{|\log(r)|}, \end{cases} \tag{1}$$

where the random vector $r \in [0,\ 1]$, x represents the current number of iterations, and M represents the total number of individuals.

2.1.2. Spiral Foraging

In this behavior of manta rays, each individual moves both toward the previous individual and toward food in a spiral way. The updated position of $z_i^d(t)$ is defined by:

$$\begin{cases} z_1^d(x+1) = z_{best}^d(x) + r\cdot\left[z_{best}^d(x) - z_1^d(x)\right] + \beta\cdot\left[z_{best}^d(x) - z_1^d(x)\right], \\ z_i^d(x+1) = z_{best}^d(x) + r\cdot\left[z_{i-1}^d(x) - z_i^d(x)\right] + \beta\cdot\left[z_{best}^d(x) - z_i^d(x)\right],\ i = 2,\ 3,\ \ldots,\ M \\ \beta = 2e^{r_1\frac{X-t+1}{X}}\cdot\sin(2\pi r_1), \end{cases} \tag{2}$$

in which the random number $r_1 \in [0,\ 1]$ and X represents the maximum number of iterations.

The above spiral foraging behavior can also be improved for exploration. Then, the mathematical formula is given by:

$$\begin{cases} z_1^d(x+1) = z_{rand}^d(x) + r\cdot\left[z_{rand}^d(x) - z_1^d(x)\right] + \beta\cdot\left[z_{rand}^d(x) - z_1^d(x)\right], \\ z_i^d(x+1) = z_{rand}^d(x) + r\cdot\left[z_{i-1}^d(x) - z_i^d(x)\right] + \beta\cdot\left[z_{rand}^d(x) - z_i^d(x)\right],\ i = 2,\ 3,\ \ldots,\ M \\ z_{rand}^d = Lb^d + r\cdot(Ub^d - Lb^d), \end{cases} \tag{3}$$

where Lb^d and Ub^d represent the upper and lower bounds, respectively.

2.1.3. Somersault Foraging (SF)

In SF behavior of manta rays, each individual is updated only in relation to the best individual. The updated formula of $z_i^d(x)$ in SF is given by

$$z_i^d(x+1) = z_i^d(x) + S\cdot\left[r_2\cdot z_{best}^d(x) - r_3\cdot z_i^d(x)\right],\ i = 1,\ 2,\ \ldots,\ M \tag{4}$$

where $r_2,\ r_3 \in [0,\ 1]$ are random numbers and $S = 2$ is the somersault factor.

2.2. Chaotic MRFO

To heighten the overall performance of the MRFO algorithm, a novel elite chaotic manta ray algorithm, called the CMRFO, integrated with chaotic initialization and opposition-based learning is developed in the section.

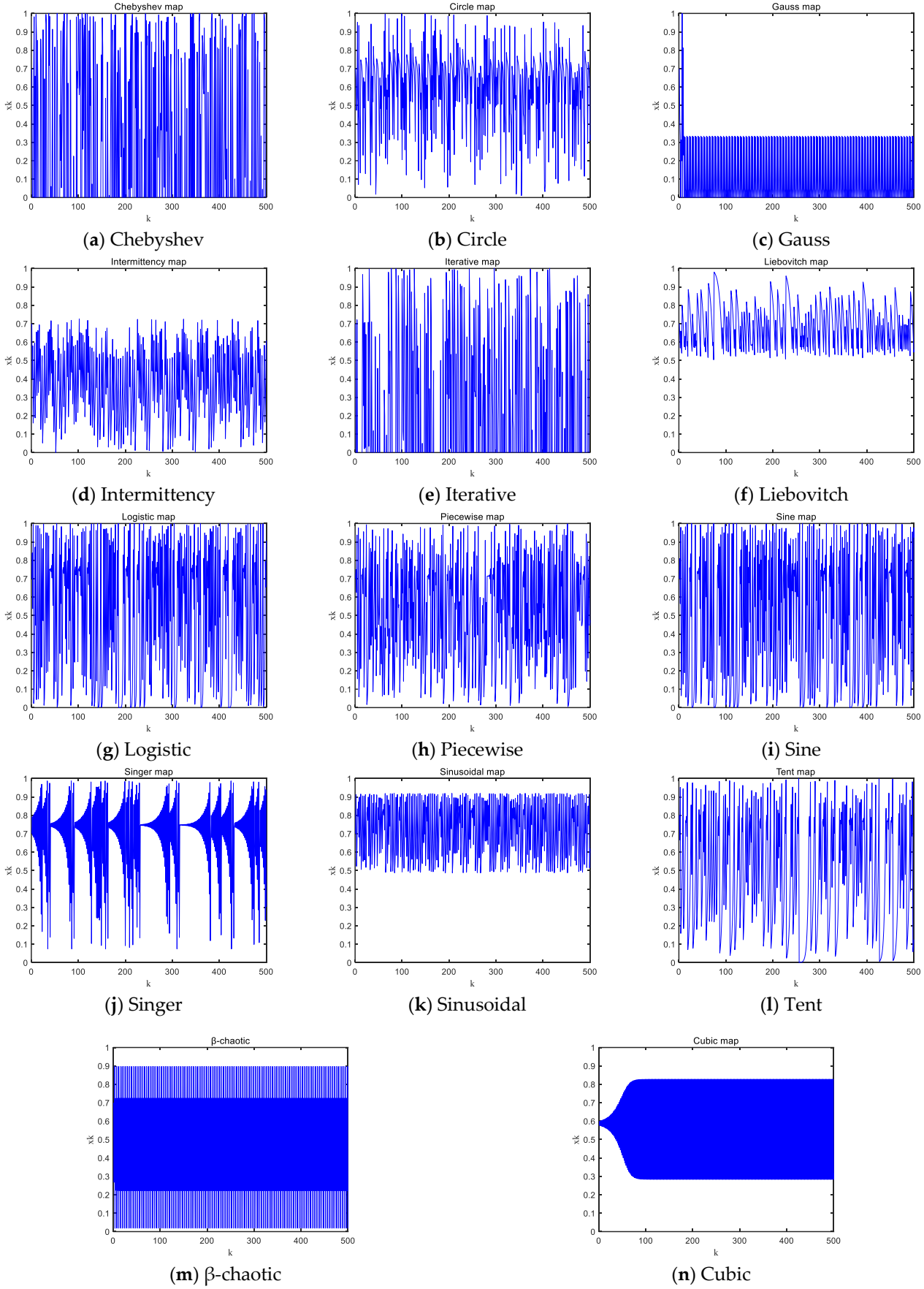
2.2.1. Chaotic Initialization of Population

The global convergence speed and convergence accuracy are affected by the quality of the initial population of optimization algorithms, and the high diversity of the initial population is conducive to enhancing the solution quality. The MRFO is known to initialize its population randomly, and the population cannot be uniformly distributed in the whole search space, which results in the reduction of efficiency in the search process. Nevertheless, a chaotic map owns the peculiarity of ergodicity and randomness, which can thoroughly

probe the search space in a range. In our paper, multiple chaotic maps are used to improve the MRFO algorithm. Table 1 describes 14 different one-dimensional maps, where k is the index and θ_k is the k-th number in the chaotic sequence. Figure 1 shows the corresponding 14 different one-dimensional map images.

Table 1. Fourteen chaotic maps.

No.	Map Name	Map Equation
1	Chebyshev (M1)	$\theta_{k+1} = \cos\left[\frac{k}{\cos(\theta_k)}\right]$
2	Circle (M2)	$\theta_{k+1} = \theta_k - \frac{a}{2\pi}\sin(2\pi\theta_k)\mathrm{mod}(1) + b$
3	Gauss/mouse (M3)	$\theta_{k+1} = \begin{cases} 0 & , \theta_k = 0 \\ \frac{1}{\theta_k \mathrm{mod}(1)} & , \text{otherwise} \end{cases}$
4	Intermittency (M4)	$\theta_{k+1} = \begin{cases} \varepsilon + \theta_k + c\theta_k^n & , 0 < \theta_k \leq P \\ \frac{\theta_k - P}{1 - P} & , P < \theta_k < 1 \end{cases}$
5	Iterative (M5)	$\theta_{k+1} = \sin\left(\frac{a\pi}{\theta_k}\right),\ a \in (0,\ 1)$
6	Liebovitch (M6)	$\theta_{k+1} = \begin{cases} a\theta_k & , 0 < \theta_k \leq P_1 \\ \frac{P - \theta_k}{P_2 - P_1} & , P_1 < \theta_k \leq P_2 \\ 1 - \beta(1 - \theta_k) & , P_2 < \theta_k \leq 1 \end{cases}$
7	Logistic (M7)	$\theta_{k+1} = a\theta_k(1 - \theta_k)$
8	Piecewise (M8)	$\theta_{k+1} = \begin{cases} \theta_k P^{-1}, & 0 \leq \theta_k < P \\ (\theta_k - P)(0.5 - P)^{-1}, & P \leq \theta_k < 0.5 \\ (1 - P - \theta_k)(0.5 - P)^{-1}, & 0.5 \leq \theta_k < 1 - P \\ (1 - \theta_k)P^{-1}, & 1 - P \leq \theta_k < 1 \end{cases}$
9	Sine (M9)	$\theta_{k+1} = \frac{a}{4}\sin(\pi\theta_k),\ a \in (0,\ 4]$
10	Singer (M10)	$\theta_{k+1} = \mu(7.86\theta_k - 23.31\theta_k^2 + 28.75\theta_k^3 - 13.302875\theta_k^4)$
11	Sinusoidal (M11)	$\theta_{k+1} = a\theta_k^2\sin(\pi\theta_k)$
12	Tent (M12)	$\theta_{k+1} = \begin{cases} \theta_k/0.7\ , & \theta_k < 0.7 \\ 10(1 - \theta_k)/3, & \theta_k \geq 0.7 \end{cases}$
13	β-chaotic (M13)	$\theta_{k+1} = k\beta(\theta_k, \mu, \nu, \theta_1, \theta_2)$
14	Cubic (M14)	$\theta_{k+1} = \rho(1 - \theta_k^2)$



(**a**) Chebyshev (**b**) Circle (**c**) Gauss

(**d**) Intermittency (**e**) Iterative (**f**) Liebovitch

(**g**) Logistic (**h**) Piecewise (**i**) Sine

(**j**) Singer (**k**) Sinusoidal (**l**) Tent

(**m**) β-chaotic (**n**) Cubic

Figure 1. Fourteen different chaotic map images.

2.2.2. Opposition-Based Learning (OL)

High diversity of the population can emphasize the exploratory stage of the algorithm, and OL is one of the methods to improve population diversity. The size of the population is M. Suppose the individuals are denoted as $z_i(x) = (z_i{}^1(x), \ldots, z_i{}^d(x), \ldots, z_i{}^D(x))\ \forall\ i = 1, 2, \ldots, M$ and each dimension component satisfies $z_i^d(x) \in [Ub^d,\ Lb^d]$, $d = 1,\ 2,\ \ldots,\ D$. Then, the opposition-based solution of each dimension is calculated by Equation (5), where $\widetilde{z}_i(x) = (\widetilde{z}_i^1(x),\ \ldots,\ \widetilde{z}_i^d(x),\ \ldots,\ \widetilde{z}_i^D(x)),\ \forall\ i = 1,\ 2,\ \ldots,\ M$.

$$\widetilde{z}_i^d(x) = Ub^d + Lb^d - z_i^d(x) \tag{5}$$

The fitness values of $2M$ individuals composed of current and opposition-based individuals are calculated and sorted from small to large, and the first N individuals are selected as the new population. Opposition-based learning not only enhances the diversity of the population but also increases the ability of the MRFO to approach the global optimal solution.

2.2.3. Elite Chaotic Searching (ECS)

OL can improve the exploration ability of the proposed CMRFO. Then, elite chaotic searching is implemented to heighten the exploitation capacity of this algorithm, in which chaotic mutation is carried out on elite individuals to realize further renewal of elite individuals. The fitness values of the current population are calculated and sorted in ascending order, and then the first n ($n = p{\cdot}M$) individuals are selected as elite individuals, where p is the selection proportion and belongs to [0, 1]. The elite individuals are recorded as $\{ez_1(x),\ ez_2(x),\ \ldots,\ ez_m(x)\} \in \{z_1(x),\ z_2(x),\ \ldots,\ z_M(x)\}$, where the i-th elite individual is $ez_i(x) = (ez_i^1(x),\ ez_i^2(x),\ \ldots,\ ez_i^D(x))$ and its upper and lower bounds are as follows:

$$\begin{cases} eb^d(x) = \max\left\{ez_i^d(x),\ ez_2^d(x),\ \ldots,\ ez_n^d(x)\right\}, \\ ea^d(x) = \min\left\{ez_1^d(x),\ ez_2^d(x),\ \ldots,\ ez_n^d(x)\right\}. \end{cases} \tag{6}$$

The elite individual $ez_i(x)$ is mapped to the interval [0, 1], and the chaotic individual $c_i(x) = \left\{c_i^1(x),\ c_i^2(x),\ \ldots,\ c_i^D(x)\right\}$ is acquired, whose calculation is as follows:

$$c_i^d(x) = \frac{ez_i^d(x) - Lb^d(x)}{Ub^d(x) - Lb^d(x)},\ i = 1,\ 2,\ \ldots,\ n \tag{7}$$

Then, logistic chaotic maps on chaotic individual $c_i(x)$ are performed.

$$c_i^d(\kappa + 1) = \mu{\cdot}c_i^d(\kappa)[1 - c_i^d(\kappa)], \tag{8}$$

where the constant $\mu = 4$ and κ is the iteration number of chaotic maps.

When the maximum number of chaotic iterations $\kappa_{\max}$ is gained, the chaotic individuals are remapped into the interval $[ea^d(x),\ eb^d(x)]$. In this paper, $\kappa_{\max}$ is set to X. The i-th new elite individual $ec_i^d(x)$ is obtained as follows:

$$ec_i^d(x) = c_i^{d,\kappa_{\max}}(x){\cdot}[eb^d(x) - ea^d(x)] + ea^d(x) \tag{9}$$

Finally, a greedy choice is made between $ec_i(x)$ and $ez_i(x)$, that is

$$z_i(x+1) = \begin{cases} ez_i(x), & f(ez_i(x)) \le f(ec_i(x)) \\ ec_i(x). & f(ez_i(x)) > f(ec_i(x)) \end{cases} \tag{10}$$

With the increase in iterations, the upper and lower bounds of elite individuals are gradually reduced to the vicinity of the target solution. Hence the local search ability of the MRFO is enhanced.

The above three strategies are combined with the MRFO, and the chaotic manta ray foraging optimizer is proposed, which is recorded as the CMRFO. The pseudo-code and the flow chart for the CMRFO are given by Algorithm 1 and Figure 2, respectively.

Algorithm 1: CMRFO

Set the parameters: M, X, Ub, Lb, D, p
The chaotic map is used to generate the initial position of N manta rays. //Chaotic initialization of population
Calculate the fitness value of each individual, and save the best position.
While $x < X$
for i = 1: M
for d = 1 to D
if $rand < 0.5$ //Cyclone foraging
if $t/T < rand$

$$z_i^d(x+1) = \begin{cases} z_{rand}^d(x) + r\cdot(z_{rand}^d(x) - z_i^d(x)) + \beta\cdot(z_{rand}^d(x) - z_i^d(x)), & i = 1 \\ z_{rand}^d(x) + r\cdot(z_{i-1}^d(x) - z_i^d(x)) + \beta\cdot(z_{rand}^d(x) - z_i^d(x)). & i = 2,\ 3,\ \ldots,\ M \end{cases}$$

else

$$z_i^d(x+1) = \begin{cases} z_{best}^d(x) + r\cdot(z_{best}^d(x) - z_i^d(x)) + \beta\cdot(z_{best}^d(x) - z_i^d(x)), & i = 1 \\ z_{best}^d(x) + r\cdot(z_{i-1}^d(x) - z_i^d(x)) + \beta\cdot(z_{best}^d(x) - z_i^d(x)). & i = 2,\ 3,\ \ldots,\ M \end{cases}$$

end if
else //Chain foraging

$$z_i^d(x+1) = \begin{cases} z_i^d(x) + r\cdot(z_{best}^d(x) - z_i^d(x)) + \alpha\cdot(z_{best}^d(x) - z_i^d(x)), & i = 1 \\ z_i^d(x) + r\cdot(z_{i-1}^d(x) - x_i^d(x)) + \alpha\cdot(z_{best}^d(x) - z_i^d(x)). & i = 2,\ 3,\ \ldots,\ M \end{cases}$$

end if
Update the best position.
$z_i^d(x+1) = z_i^d(x) + S\cdot(r_2\cdot z_{best}^d(x) - r_3\cdot z_i^d(x)) \quad i = 1,\ 2,\ \ldots,\ M$ //Somersault foraging
Update the best position.
$\tilde{z}_i^d(x) = Ub^d + Lb^d - z_i^d(x)$ //Opposition-based learning
The first N individuals among current and opposition-based individuals are selected as the new population.
The fitness values of the current population are sorted in ascending order, and the first n individuals are selected as elite individuals. //Elite chaotic searching
for I = 1 to n
$c_i^d(x) = \frac{ez_i^d(x) - Lb^d(x)}{Ub^d(x) - Lb^d(x)}$
for k = 1 to X
$c_i^d(\kappa+1) = \mu\cdot c_i^d(\kappa)\cdot(1 - c_i^d(\kappa))$
end for
$ec_i^d(x) = c_i^{d,\kappa_{\max}}(x)\cdot(eb^d(x) - ea^d(x)) + ea^d(x)$
if $f(ec_i(x)) < f(z_i(x))$ then
$z_i(x+1) = ec_i(x)$
end if
end for
end for
end for
End while
Output the global best position.

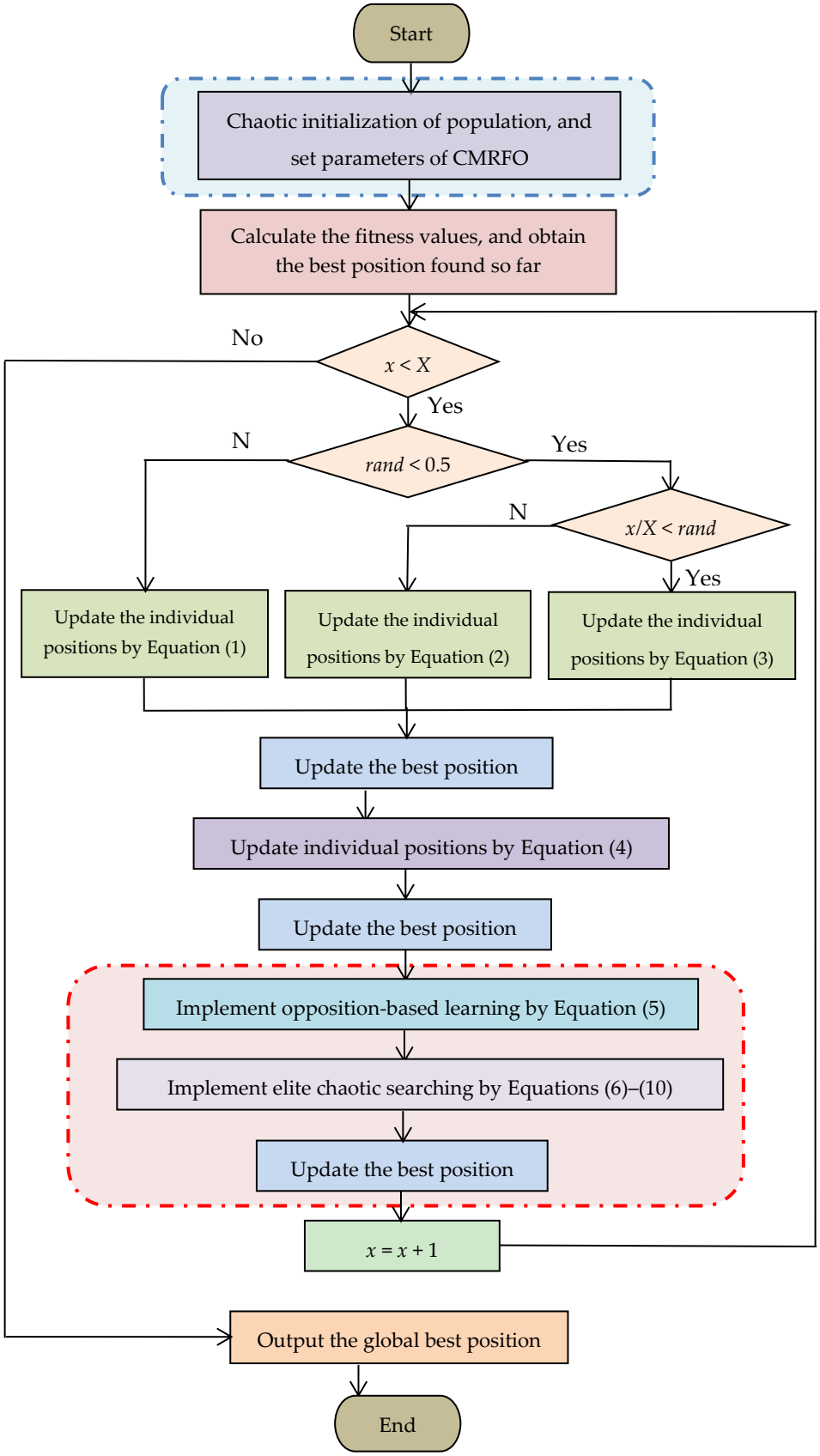


Figure 2. Flow chart of the CMRFO.

3. Experimental Results and Analysis

In this section, the capability and superiority of the CMRFO are comprehensively demonstrated by 23 classical test suites and the IEEE CEC2020 benchmark. The numerical experiments are implemented on Intel(R) Core(TM) i7-7700HQ, 2.80 GHz or 2.81 GHz, 8.00 GB, 512 GB, Windows 10, and Matlab 2018a. Here, the values of N and T are 50 and 1000, respectively. The obtained data are the results of each algorithm running independently 30 times. The 23 classical test functions of three different types are listed in the literature [7].

3.1. Performance of the CMRFO for the Initializing Population Based on Different Chaotic Maps

The CMRFO uses chaotic maps to generate the initial population, which can achieve a uniform distribution of the population and explore the search space more comprehensively within a certain range. This is conducive to heightening the performance and efficiency of intelligence algorithms in the search process. Unimodal functions can examine the local search ability of the CMRFO. Meanwhile, multimodal and fixed-dimensional multimodal functions are treated to be an intractable problem because they have numerous local extrema. So this comparison experiment is performed on 23 benchmark functions of three different types. Table 2 shows the statistical results of the CMRFO using 14 different chaotic maps to initialize the population, in which the boldface data represent the optimal values of 14 different chaotic maps.

Table 2. Results of the CMRFO using 14 different chaotic maps on 23 benchmark functions.

No.	Result	CMRFO						
		M1	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**
F1	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F2	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F3	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F4	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F5	Mean	8.70060	0.90144	1.76270	2.17×10^{-7}	7.80430	0.89795	0.88399
	Std	79.7595	16.2520	29.5025	8.89×10^{-13}	78.6172	16.1263	15.6288
	Rank	14	7	11	3	13	6	5
F6	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F7	Mean	3.26×10^{-5}	5.28×10^{-5}	4.33×10^{-5}	4.65×10^{-5}	5.44×10^{-5}	4.37×10^{-5}	3.88×10^{-5}
	Std	4.58×10^{-10}	1.29×10^{-9}	1.20×10^{-9}	1.71×10^{-9}	1.63×10^{-9}	1.87×10^{-9}	7.27×10^{-10}
	Rank	1	12	6	8	13	7	3
F8	Mean	−38,051.19	−12,569.49	−12,391.83	−9584.84	−39,165.73	−12,569.49	−12,569.49
	Std	0.0	2.35×10^{-23}	6.31×10^{5}	1.66×10^{6}	5.57×10^{-23}	1.41×10^{-23}	1.60×10^{-23}
	Rank	13	10	11	12	14	5	7
F9	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F10	Mean	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F11	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F12	Mean	1.88×10^{-30}	1.04×10^{-31}	3.61×10^{-31}	9.12×10^{-32}	2.06×10^{-30}	8.87×10^{-32}	2.68×10^{-31}
	Std	2.61×10^{-59}	5.64×10^{-62}	2.93×10^{-61}	6.34×10^{-62}	2.56×10^{-59}	2.65×10^{-62}	7.27×10^{-61}
	Rank	13	5	11	4	14	3	9

Table 2. *Cont.*

No.	Result	CMRFO						
		M1	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**
F13	Mean	1.38×10^{-29}	7.61×10^{-31}	1.15×10^{-29}	1.87×10^{-31}	5.49×10^{-4}	2.35×10^{-31}	5.92×10^{-31}
	Std	1.08×10^{-57}	3.98×10^{-60}	3.90×10^{-58}	1.77×10^{-61}	6.04×10^{-6}	7.95×10^{-62}	1.46×10^{-60}
	Rank	13	7	12	2	14	3	6
F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998	0.998
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F15	Mean	3.53×10^{-4}	3.07×10^{-4}	3.53×10^{-4}	3.07×10^{-4}	3.53×10^{-4}	3.07×10^{-4}	3.07×10^{-4}
	Std	4.19×10^{-8}	5.88×10^{-38}	4.19×10^{-8}	3.37×10^{-38}	4.19×10^{-8}	2.68×10^{-38}	2.26×10^{-38}
	Rank	11	10	11	9	11	3	2
F16	Mean	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316
	Std	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F17	Mean	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F18	Mean	3	3	3	3	3	3	3
	Std	4.05×10^{-31}	3.01×10^{-31}	5.19×10^{-32}	1.56×10^{-31}	0	7.78×10^{-31}	9.34×10^{-32}
	Rank	9	7	3	6	1	12	4
F19	Mean	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628
	Std	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F20	Mean	−3.2863	−3.3101	−3.2863	−3.3042	−3.3101	−3.2923	−3.3161
	Std	3.12×10^{-3}	1.34×10^{-3}	3.12×10^{-3}	1.90×10^{-3}	1.34×10^{-3}	2.79×10^{-3}	7.07×10^{-4}
	Rank	6	2	6	3	2	5	1
F21	Mean	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532
	Std	1.13×10^{-29}	1.33×10^{-29}	1.08×10^{-29}	1.08×10^{-29}	9.80×10^{-30}	1.08×10^{-29}	1.18×10^{-29}
	Rank	3	7	2	2	1	2	4
F22	Mean	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029
	Std	7.31×10^{-30}	8.30×10^{-30}	9.80×10^{-30}	8.30×10^{-30}	7.31×10^{-30}	9.30×10^{-30}	9.80×10^{-30}
	Rank	2	4	6	4	2	5	6
F23	Mean	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364
	Std	3.32×10^{-30}	3.65×10^{-30}	3.16×10^{-30}	3.32×10^{-30}	2.99×10^{-30}	3.32×10^{-30}	4.32×10^{-30}
	Rank	3	4	2	3	1	3	5
Mean Rank		4.3478	3.7826	4.0435	2.9565	4.2609	2.8696	2.7826
Result		13	8	10	5	12	4	3

Table 3 shows that in the CMRFO algorithm, the results of using different chaotic maps to initialize the initial population are slightly different. For all chaotic maps, according to the final ranking in the last line of Table 3, the performance of the CMRFO using M14 (cubic map) is significantly better than that on using other chaotic maps. Therefore, we choose M14 (cubic map) to initialize the population, and the M14-integrated CMRFO will be studied in detail in Section 3.2.

Table 3. Results of CMRFO using 14 different chaotic maps to initialize the population (continued).

No.	Result	CMRFO						
		M8	**M9**	**M10**	**M11**	**M12**	**M13**	**M14**
F1	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F2	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F3	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0

Table 3. *Cont.*

No.	Result	CMRFO						
		M8	**M9**	**M10**	**M11**	**M12**	**M13**	**M14**
F4	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F5	Mean	0.91203	1.68030	0.91182	1.77×10^{-7}	7.79×10^{-7}	3.55540	1.62×10^{-8}
	Std	16.6361	26.7471	16.6282	2.01×10^{-13}	5.36×10^{-12}	53.3194	1.04×10^{-15}
	Rank	9	10	8	2	4	12	1
F6	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F7	Mean	4.89×10^{-5}	5.01×10^{-5}	4.30×10^{-5}	3.75×10^{-5}	6.05×10^{-5}	5.23×10^{-5}	4.14×10^{-5}
	Std	6.98×10^{-10}	2.06×10^{-9}	1.05×10^{-9}	1.08×10^{-9}	2.74×10^{-9}	1.30×10^{-9}	9.62×10^{-10}
	Rank	9	10	5	2	14	11	4
F8	Mean	−12,569.49	−12,569.49	−12,569.49	−12,569.49	−12,569.49	−12,569.49	−12,569.49
	Std	5.22×10^{-24}	1.36×10^{-23}	1.76×10^{-23}	1.95×10^{-23}	1.57×10^{-23}	1.22×10^{-23}	6.27×10^{-24}
	Rank	1	4	8	9	6	3	2
F9	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F10	Mean	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F11	Mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F12	Mean	2.29×10^{-31}	1.46×10^{-31}	7.63×10^{-32}	7.80×10^{-32}	2.86×10^{-31}	6.84×10^{-31}	1.56×10^{-31}
	Std	6.98×10^{-61}	5.94×10^{-62}	1.80×10^{-62}	9.70×10^{-63}	3.54×10^{-61}	2.90×10^{-60}	2.00×10^{-61}
	Rank	8	6	1	2	10	12	7
F13	Mean	1.21×10^{-30}	3.99×10^{-30}	3.19×10^{-30}	2.93×10^{-30}	4.16×10^{-31}	2.19×10^{-30}	1.80×10^{-31}
	Std	6.54×10^{-60}	5.57×10^{-59}	3.21×10^{-61}	8.34×10^{-59}	7.06×10^{-61}	1.42×10^{-59}	1.45×10^{-61}
	Rank	8	11	4	10	5	9	1
F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998	0.998
	Std	0	2.59×10^{-33}	0	0	0	0	0
	Rank	1	2	1	1	1	1	1
F15	Mean	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.53×10^{-4}	3.07×10^{-4}
	Std	2.17×10^{-38}	2.72×10^{-38}	2.80×10^{-38}	2.94×10^{-38}	2.71×10^{-38}	4.19×10^{-8}	3.12×10^{-38}
	Rank	1	5	6	7	4	11	8
F16	Mean	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316
	Std	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F17	Mean	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789
	Std	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F18	Mean	3	3	3	3	3	3	3
	Std	3.43×10^{-31}	7.68×10^{-31}	4.46×10^{-31}	9.34×10^{-32}	9.34×10^{-32}	1.35×10^{-31}	1.04×10^{-32}
	Rank	8	11	10	4	4	5	2
F19	Mean	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628
	Std	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}
	Rank	1.0	1.0	1.0	1.0	1.0	1.0	1.0
F20	Mean	−3.2982	−3.2923	−3.2982	−3.2923	−3.2744	−3.3042	−3.2923
	Std	2.38×10^{-3}	2.79×10^{-3}	2.38×10^{-3}	2.79×10^{-3}	3.57×10^{-3}	1.90×10^{-3}	2.79×10^{-3}
	Rank	4	5	4	5	7	3	5
F21	Mean	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532
	Std	1.23×10^{-29}	1.13×10^{-29}	1.28×10^{-29}	1.23×10^{-29}	1.23×10^{-29}	1.18×10^{-29}	1.23×10^{-29}
	Rank	5	3	6	5	5	4	5
F22	Mean	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029
	Std	5.31×10^{-30}	7.81×10^{-30}	9.30×10^{-30}	7.81×10^{-30}	7.81×10^{-30}	9.80×10^{-30}	7.31×10^{-30}
	Rank	1	3	5	3	3	6	2
F23	Mean	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364
	Std	3.16×10^{-30}	1.23×10^{-29}	3.16×10^{-30}	2.99×10^{-30}	3.32×10^{-30}	1.13×10^{-29}	3.32×10^{-30}
	Rank	2	7	2	1	3	6	3
Mean Rank		2.9565	3.8261	3.0870	2.6957	3.3478	4.0870	2.2609
Result		5	9	6	2	7	11	1

3.2. Elite Individual Proportion Analysis

In the CMRFO, the selection proportion p of elite individuals is the key to an elite chaotic searching strategy. A large value of p will cause premature algorithm convergence, while too small a value of p will have little impact on the algorithm. Therefore, this section discusses the influence of the parameter p on the performance of the CMRFO by simulation experiments. Table 4 illustrates the effects of the CMRFO based on different p-values on benchmark functions, where the boldface data represent the optimal values of different p-values.

Table 4. Results of the CMRFO based on different p-values.

No.	Result	The p-Value of CMRFO								
		0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
F1	Mean	0	0	0	0	0	0	0	0	0
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F2	Mean	0	0	0	0	0	0	0	0	0
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F3	Mean	0	0	0	0	0	0	0	0	0
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F4	Mean	0	4.94×10^{-324}	4.94×10^{-324}	4.94×10^{-324}	4.94×10^{-324}	4.94×10^{-324}	4.94×10^{-324}	4.94×10^{-324}	4.94×10^{-324}
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	2	2	2	2	2	2	2	2
F5	Mean	5.79×10^{-9}	1.29×10^{-8}	3.25×10^{-8}	1.76×10^{-8}	6.58×10^{-8}	3.11×10^{-7}	4.64×10^{-8}	1.23×10^{-6}	1.56×10^{-7}
	Std	2.35×10^{-16}	1.17×10^{-15}	1.64×10^{-14}	7.74×10^{-16}	1.24×10^{-14}	1.52×10^{-12}	1.59×10^{-14}	2.23×10^{-11}	4.22×10^{-13}
	Rank	1	2	4	3	6	8	5	9	7
F6	Mean	0	0	0	0	0	0	0	0	0
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F7	Mean	2.07×10^{-5}	2.40×10^{-5}	2.17×10^{-5}	2.44×10^{-5}	3.70×10^{-5}	2.70×10^{-5}	3.72×10^{-5}	5.33×10^{-5}	6.42×10^{-5}
	Std	2.88×10^{-10}	3.61×10^{-10}	6.50×10^{-10}	3.90×10^{-10}	7.30×10^{-10}	3.51×10^{-10}	1.03×10^{-9}	2.81×10^{-9}	3.66×10^{-9}
	Rank	1	3	2	4	6	5	7	8	9
F8	Mean	−12,569.49	−12,504.34	−12,563.56	−12,534.94	−12,569.49	−12,489.54	−12,569.49	−12,541.85	−12,532.97
	Std	6.62×10^{-24}	8.49×10^{-4}	7.01×10^{-2}	1.67×10^{-4}	6.62×10^{-24}	1.28×10^{5}	6.10×10^{-24}	7.60×10^{3}	2.67×10^{4}
	Rank	2	7	3	5	2	8	1	4	6
F9	Mean	0	0	0	0	0	0	0	0	0
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F10	Mean	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F11	Mean	0	0	0	0	0	0	0	0	0
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F12	Mean	1.57×10^{-32}	1.60×10^{-32}	7.76×10^{-32}	1.81×10^{-30}	2.38×10^{-30}	4.23×10^{-30}	1.07×10^{-29}	1.15×10^{-29}	8.80×10^{-30}
	Std	7.88×10^{-96}	1.23×10^{-67}	9.46×10^{-63}	1.13×10^{-59}	1.78×10^{-59}	2.93×10^{-59}	1.35×10^{-57}	5.77×10^{-58}	2.37×10^{-58}
	Rank	1	2	3	4	5	6	8	9	7
F13	Mean	1.36×10^{-32}	1.62×10^{-32}	2.21×10^{-30}	2.95×10^{-29}	1.01×10^{-29}	4.16×10^{-29}	1.29×10^{-28}	5.31×10^{-28}	1.24×10^{-28}
	Std	7.60×10^{-68}	4.79×10^{-65}	7.23×10^{-60}	4.54×10^{-57}	1.57×10^{-58}	5.71×10^{-57}	8.07×10^{-56}	4.54×10^{-54}	6.16×10^{-56}
	Rank	1	2	3	5	4	6	8	9	7
F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998	0.998	0.998	0.998
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F15	Mean	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.07×10^{-4}
	Std	1.87×10^{-38}	2.06×10^{-38}	2.72×10^{-38}	2.92×10^{-38}	3.96×10^{-38}	3.28×10^{-38}	1.89×10^{-38}	2.41×10^{-38}	2.10×10^{-38}
	Rank	1	3	6	7	9	8	2	5	4
F16	Mean	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316
	Std	2.55×10^{-20}	8.55×10^{-20}	5.83×10^{-41}	2.23×10^{-16}	1.03×10^{-18}	1.41×10^{-18}	1.02×10^{-19}	1.70×10^{-18}	1.58×10^{-12}
	Rank	1	2	8	7	4	5	3	6	9
F17	Mean	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789
	Std	0	0	0	0	0	0	0	0	0
	Rank	1	1	1	1	1	1	1	1	1
F18	Mean	3	3	3	3	3	3	3	3	3
	Std	1.35×10^{-31}	3.11×10^{-31}	5.19×10^{-32}	3.11×10^{-31}	2.70×10^{-31}	1.25×10^{-31}	5.29×10^{-31}	4.05×10^{-31}	3.63×10^{-31}
	Rank	3	5	1	5	4	2	8	7	6
F19	Mean	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628
	Std	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}
	Rank	1	1	1	1	1	1	1	1	1
F20	Mean	−3.2982	−3.2863	−3.2863	−3.2804	−3.2863	−3.2804	−3.2804	−3.2625	−3.2685
	Std	2.38×10^{-3}	3.12×10^{-3}	3.12×10^{-3}	3.39×10^{-3}	3.12×10^{-3}	3.39×10^{-3}	3.39×10^{-3}	3.72×10^{-3}	3.68×10^{-3}
	Rank	1	2	2	3	2	3	3	5	4

Table 4. *Cont.*

No.	Result	The p-Value of CMRFO								
		0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
F21	Mean	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532	−10.1532
	Std	1.18×10^{-29}	1.28×10^{-29}	1.33×10^{-29}	1.23×10^{-29}	1.23×10^{-29}	1.28×10^{-29}	1.33×10^{-29}	1.28×10^{-29}	1.28×10^{-29}
	Rank	1	3	4	2	2	3	4	3	3
F22	Mean	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029	−10.4029
	Std	5.81×10^{-30}	8.30×10^{-30}	7.31×10^{-30}	7.81×10^{-30}	5.81×10^{-30}	9.30×10^{-30}	9.30×10^{-30}	6.81×10^{-30}	9.30×10^{-30}
	Rank	1	5	3	4	1	6	6	2	6
F23	Mean	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364	−10.5364
	Std	3.16×10^{-30}	3.32×10^{-30}	3.32×10^{-30}	4.32×10^{-30}	3.32×10^{-30}	3.82×10^{-30}	3.32×10^{-30}	3.82×10^{-30}	3.32×10^{-30}
	Rank	1	2	2	4	2	3	2	3	2
Mean Rank		1.1304	2.1739	2.3043	2.8261	2.5652	3.2609	3.0000	3.5652	3.5652
Result		1	2	3	5	4	7	6	8	8

It is observed that when the different values of the selection proportion p are taken, CMRFO has different results on some functions, so the ECS is sensitive to the value of p. In conclusion, when p = 0.1, the CMRFO performs best. Therefore, in this paper, the value of p in the elite chaotic searching is set to 0.1.

3.3. Exploration–Exploitation Analysis

Exploration and exploitation are the two basic building blocks of a meta-heuristic optimization algorithm. The search phase can be explored in the search space of distant regions. However, in the exploitation phase, candidate solutions steadily exploit promising areas already identified in the previous step using local strategies. Thus, maintaining a good balance between these two phases is one way to ensure that an algorithm can guarantee optimal convergence.

In this paper, exploration and exploitation are obtained by the dimensional diversity measure. During the search process, the exploration capability can be measured by the increase in the average value of the distance within the population dimension. Alternatively, the decreasing mean can be considered as the stage of exploitation where the search agent is located in a concentrated area. The following equation shows that dimensional diversity is measured during the search process.

$$Div_j = \frac{1}{N}\sum_{i=1}^{N} Median(x_j) - x_{i,j} \tag{11}$$

$$Div(t) = \frac{1}{D}\sum_{j=1}^{D} Div_j,\ t = 1,\ 2,\ \ldots,\ T \tag{12}$$

where $x_{i,j}$ is the position of the i-th candidate solution in the j-th dim, Div_j is the average diversity in the j-th dimension, and $Median(x_j)$ is the median of the j-th dim of the candidate solution. N is the number of all populations, D is the dim, and T is the maximum number of iterations. The following equation calculates the percentage of exploration and exploitation:

$$Exploration\% = \frac{Div(t)}{\max(Div)} \times 100\% \tag{13}$$

$$Exploitation\% = \frac{|Div(t) - \max(Div)|}{\max(Div)} \times 100\% \tag{14}$$

where $max(Div)$ is the maximum diversity in T iterations.

We plotted the exploration and exploitation convergence behavior using some CEC2020 test functions. Figure 3 shows the exploration and exploitation behavior of the cec01, cec02, cec05, cec07, cec09, and cec10 functions. For these functions, the CMRFO demonstrates highly dynamic behavior. As seen from the figure, the CMRFO tends to start the iterative process with a high exploration rate and a low exploitation rate. However, it remains more

exploitative in the late iterations. The CMRFO maintains a tendency to balance exploration and exploitation during the search process.

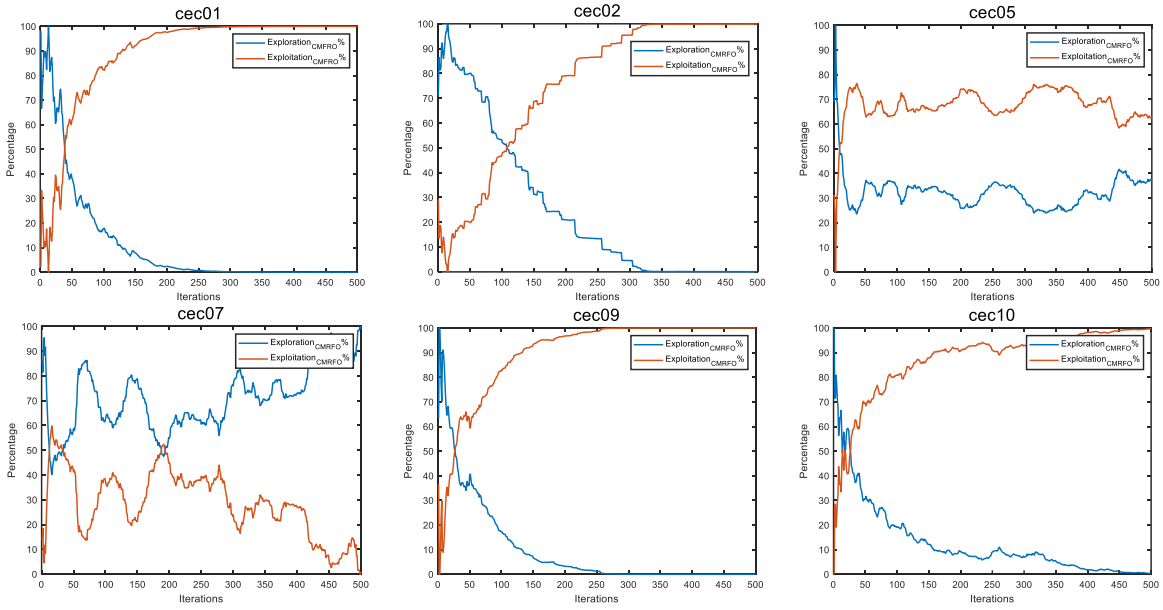


Figure 3. Exploration and exploitation phases in the CMRFO on the CEC2020 functions.

3.4. Comparison of the CMRFO with Other Optimizers on 23 Benchmark Functions

The CMRFO is compared with some optimizers using 23 classical test suites in this section. These algorithms are the original MRFO, an improved MRFO, classical algorithms, and recently proposed algorithms, including MRFO [23], MRFO–GBO [44], DMRFO [38], SA–MRFO [43], PSO [46], GWO [47], HHO [48], AOA [49], CHOA [50], and MPA [51]. For all improved MRFOs, the parameter settings of these algorithms remain unchanged. Table 5 provides the parameter settings for other optimizers. Numerical results of 11 comparison algorithms using 23 classical benchmarks are shown in Figure 4 and Table 6.

Table 5. Description of parameter settings.

Algorithms	Parameter Values
MRFO	$S = 2$
CMRFO	$S = 2$, $p = 0.1$
PSO	$P_1 = P_2 = 2$; ω: linearly decreases from 0.8 to 0.2
GWO	α: the value range of α is [0, 2]; increases linearly
HHO	E_0: [−1 1]
AOA	$P_1 = 2$, $P_2 = 6$, $P_3 = 1$, $P_4 = 2$
CHOA	f: non-linearly decreases from 2.5 to 0; chaotic map: tent map
MPA	$F = 0.2$, $P = 0.5$

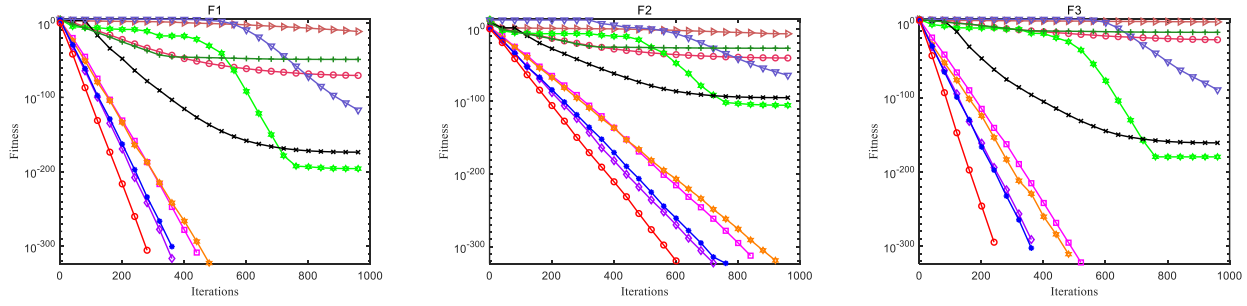


Figure 4. *Cont.*

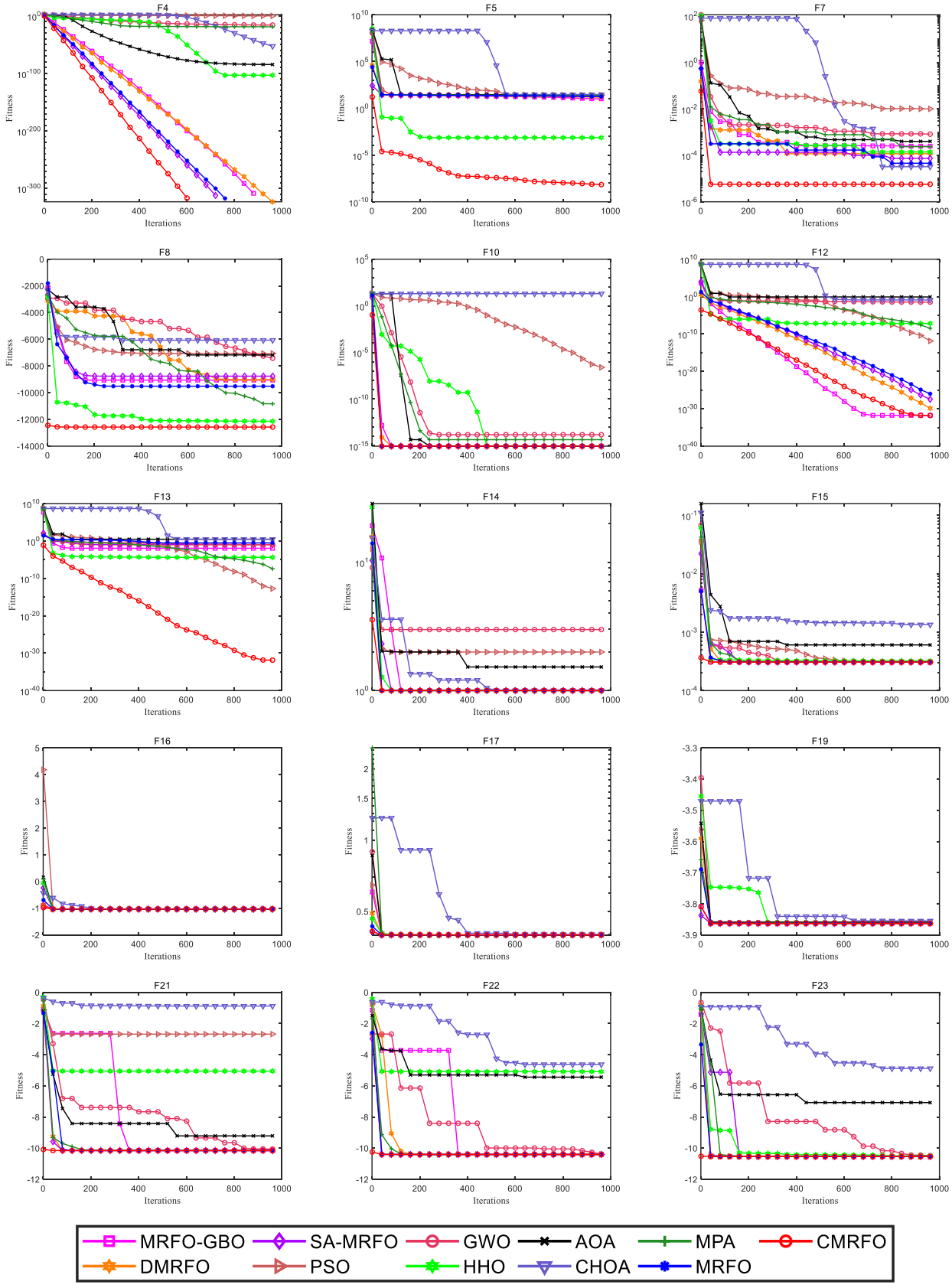


Figure 4. Convergence curves of 11 algorithms.

Table 6. Comparison results of all 11 optimizers on 23 classical benchmarks.

No.	Result	Algorithm										
		MRFO	**CMRFO**	**MRFO–GBO**	**DMRFO**	**SA–MRFO**	**PSO**	**GWO**	**HHO**	**AOA**	**CHOA**	**MPA**
F1	Mean	0	0	0	0	0	1.31×10^{-10}	4.08×10^{-70}	3.36×10^{-187}	6.99×10^{-181}	1.73×10^{-125}	1.28×10^{-49}
	Std	0	0	0	0	0	2.91×10^{-19}	8.72×10^{-139}	0	0	1.24×10^{-249}	8.54×10^{-98}
	Rank	1	1	1	1	1	7	5	2	3	4	6
F2	Mean	0	0	0	0	0	1.42×10^{-6}	6.14×10^{-41}	1.16×10^{-100}	2.80×10^{-91}	2.07×10^{-66}	1.29×10^{-27}
	Std	0	0	0	0	0	6.47×10^{-12}	5.88×10^{-81}	1.19×10^{-199}	1.56×10^{-180}	5.94×10^{-131}	9.56×10^{-54}
	Rank	1	1	1	1	1	7	5	2	3	4	6
F3	Mean	0	0	0	0	0	43.72	4.72×10^{-21}	1.40×10^{-160}	5.16×10^{-142}	1.05×10^{-99}	3.04×10^{-13}
	Std	0	0	0	0	0	292.39	1.17×10^{-40}	3.94×10^{-319}	5.31×10^{-282}	2.08×10^{-197}	6.34×10^{-25}
	Rank	1	1	1	1	1	7	5	2	3	4	6
F4	Mean	0	0	0	4.94×10^{-324}	0	1.0411	1.47×10^{-17}	1.74×10^{-100}	5.26×10^{-79}	3.93×10^{-55}	1.94×10^{-19}
	Std	0	0	0	0	0	9.58×10^{-2}	3.01×10^{-34}	2.14×10^{-199}	2.95×10^{-156}	1.79×10^{-108}	2.08×10^{-38}
	Rank	1	1	1	2	1	8	7	3	4	5	6
F5	Mean	17.3485	9.10×10^{-9}	10.4979	16.526	17.3679	37.8739	26.5877	1.44×10^{-3}	28.8316	28.9262	22.2968
	Std	2.49×10^{-1}	2.83×10^{-16}	4.5822	4.82×10^{-1}	3.12×10^{-1}	1.48×10^{3}	7.74×10^{-1}	4.99×10^{-6}	8.71×10^{-3}	8.71×10^{-3}	2.12×10^{-1}
	Rank	5	1	3	4	6	11	8	2	9	10	7
F6	Mean	0	0	0	0	0	5.00×10^{-2}	0	0	0	0	0
	Std	0	0	0	0	0	5.00×10^{-2}	0	0	0	0	0
	Rank	1	1	1	1	1	2	1	1	1	1	1
F7	Mean	5.98×10^{-5}	1.54×10^{-5}	1.08×10^{-4}	5.41×10^{-5}	3.37×10^{-5}	8.66×10^{-3}	5.20×10^{-4}	3.60×10^{-5}	2.56×10^{-4}	6.50×10^{-5}	5.71×10^{-4}
	Std	2.13×10^{-9}	1.21×10^{-10}	3.99×10^{-9}	1.52×10^{-9}	9.95×10^{-10}	5.97×10^{-6}	1.28×10^{-7}	7.04×10^{-10}	2.10×10^{-8}	3.16×10^{-9}	1.10×10^{-7}
	Rank	5	1	7	4	2	11	8	3	8	6	10
F8	Mean	−8432.83	−12,569.49	−9533.46	−9485.30	−8554.22	−6758.13	−5962.18	−12,569.41	-5.83×10^{7}	−5866.21	−10,161.35
	Std	7.61×10^{5}	7.14×10^{-24}	4.30×10^{5}	1.29×10^{5}	6.33×10^{5}	4.85×10^{5}	4.73×10^{5}	1.09×10^{-2}	3.24×10^{16}	3.87×10^{3}	1.05×10^{5}
	Rank	7	1	4	5	6	8	9	2	11	10	3
F9	Mean	0	0	0	0	0	43.6786	0.1592	0	6.1970	4.0521	0
	Std	0	0	0	0	0	145.0417	0.5071	0	768.0535	98.1056	0
	Rank	1	1	1	1	1	5	2	1	4	3	1
F10	Mean	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	2.58×10^{-7}	1.44×10^{-14}	8.88×10^{-16}	10.9812	19.9597	3.38×10^{-15}
	Std	0	0	0	0	0	3.54×10^{-13}	6.11×10^{-30}	0	1.04×10^{-2}	1.42×10^{-6}	2.79×10^{-30}
	Rank	1	1	1	1	1	4	3	1	5	6	2
F11	Mean	0	0	0	0	0	1.73×10^{-2}	0	0	0	0	0
	Std	0	0	0	0	0	3.12×10^{-4}	0	0	0	0	0
	Rank	1	1	1	1	1	2	1	1	1	1	1
F12	Mean	7.81×10^{-29}	1.57×10^{-32}	1.71×10^{-32}	5.32×10^{-31}	1.03×10^{-28}	1.04×10^{-2}	2.45×10^{-2}	4.59×10^{-7}	7.34×10^{-1}	1.25×10^{-1}	5.85×10^{-11}
	Std	1.06×10^{-56}	8.73×10^{-70}	9.10×10^{-67}	1.87×10^{-60}	7.17×10^{-56}	1.02×10^{-3}	3.25×10^{-4}	3.69×10^{-13}	1.54×10^{-2}	3.84×10^{-4}	1.34×10^{-21}
	Rank	4	1	2	3	5	8	9	7	11	10	6

Table 6. *Cont.*

No.	Result	Algorithm										
		MRFO	**CMRFO**	**MRFO–GBO**	**DMRFO**	**SA–MRFO**	**PSO**	**GWO**	**HHO**	**AOA**	**CHOA**	**MPA**
F13	Mean	2.3948	1.41×10^{-23}	1.47×10^{-2}	1.7660	1.5257	1.65×10^{-3}	3.74×10^{-1}	2.75×10^{-5}	2.8802	2.9816	9.50×10^{-10}
	Std	1.3760	2.95×10^{-66}	8.60×10^{-4}	1.8956	2.1871	1.62×10^{-5}	2.79×10^{-2}	8.72×10^{-10}	1.092	1.58×10^{-3}	3.87×10^{-19}
	Rank	9	1	5	8	7	4	6	3	10	11	2
F14	Mean	0.998	0.998	0.998	0.998	0.998	2.0349	3.6456	0.998	1.0509	0.9981	0.998
	Std	0	0	0	0	0	3.5231	14.0541	1.38×10^{-21}	5.00×10^{-2}	3.47×10^{-8}	0
	Rank	1	1	1	1	1	5	6	2	4	3	1
F15	Mean	3.53×10^{-4}	3.07×10^{-4}	3.07×10^{-4}	3.53×10^{-4}	4.05×10^{-4}	2.44×10^{-3}	2.36×10^{-3}	3.20×10^{-4}	5.27×10^{-4}	1.35×10^{-3}	3.07×10^{-4}
	Std	4.19×10^{-8}	6.81×10^{-39}	2.23×10^{-38}	4.19×10^{-8}	7.90×10^{-8}	3.77×10^{-5}	3.80×10^{-5}	2.95×10^{-10}	1.78×10^{-7}	3.81×10^{-9}	5.82×10^{-37}
	Rank	5	1	2	5	6	10	9	4	7	8	3
F16	Mean	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316
	Std	5.19×10^{-32}	2.44×10^{-20}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	5.19×10^{-32}	1.01×10^{-17}	3.40×10^{-26}	9.69×10^{-11}	6.32×10^{-9}	4.67×10^{-32}
	Rank	2	4	2	2	2	2	5	3	6	7	1
F17	Mean	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39789	0.39791	0.39856	0.39789
	Std	0	0	0	0	0	0	1.68×10^{-14}	9.53×10^{-17}	3.85×10^{-9}	3.40×10^{-7}	0
	Rank	1	1	1	1	1	1	3	2	4	5	1
F18	Mean	3	3	3	3	3	3	3	3	3.0353	3	3
	Std	4.26×10^{-31}	1.35×10^{-31}	1.04×10^{-32}	2.59×10^{-31}	4.67×10^{-31}	5.61×10^{-31}	1.57×10^{-11}	1.31×10^{-17}	5.48×10^{-3}	1.50×10^{-10}	1.06×10^{-30}
	Rank	4	2	1	3	5	6	9	8	11	10	7
F19	Mean	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628	−3.8616	−3.8626	−3.8580	−3.8539	−3.8628
	Std	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	5.19×10^{-30}	7.18×10^{-6}	3.93×10^{-7}	4.30×10^{-5}	8.75×10^{-7}	5.19×10^{-30}
	Rank	1	1	1	1	1	1	3	2	4	5	1
F20	Mean	−3.2566	−3.2923	−3.2566	−3.2863	−3.2625	−3.2863	−3.2472	−3.1740	−3.0823	−2.6656	−3.322
	Std	3.6×10^{-3}	2.79×10^{-3}	3.68×10^{-3}	3.12×10^{-3}	3.72×10^{-3}	3.12×10^{-3}	7.71×10^{-3}	4.50×10^{-3}	1.38×10^{-2}	1.86×10^{-1}	1.18×10^{-29}
	Rank	5	2	5	3	4	3	6	7	8	9	1
F21	Mean	−8.8787	−10.1532	−8.8787	−9.8983	−9.6434	−5.7660	−9.6453	−5.5628	−7.4155	−3.1803	−10.1532
	Std	5.1295	1.23×10^{-29}	5.1295	1.2995	2.4622	11.7623	2.4401	2.4432	3.8455	4.2111	4.32×10^{-30}
	Rank	6	2	6	3	5	8	4	9	7	10	1
F22	Mean	−9.8714	−10.4029	−9.6057	−10.3412	−9.3399	−9.4930	−10.1369	−5.3528	−7.4643	−3.2238	−10.4029
	Std	2.6765	7.81×10^{-30}	3.7917	7.61×10^{-2}	4.7582	5.1374	1.4124	1.4087	3.8869	3.9396	1.36×10^{-29}
	Rank	5	1	6	3	8	7	4	10	9	11	2
F23	Mean	−9.4548	−10.5364	−9.7252	−10.2660	−10.5364	−6.4475	−10.5361	−5.6641	−8.7643	−4.0086	−10.5364
	Std	4.9256	3.32×10^{-30}	3.9251	1.4623	3.8230	14.947	2.48×10^{-8}	2.7207	4.2842	2.6005	4.98×10^{-30}
	Rank	7	1	6	5	2	9	4	10	8	11	3
Mean Rank		3.2609	1.2609	2.6087	2.6087	3.0000	5.9130	5.3043	3.7826	6.1304	6.6957	3.3913
Result		4	1	2	2	3	8	7	6	9	10	5

To study the convergence characteristics of the CMRFO, Figure 4 shows the convergence curves of 11 comparison methods for 23 classical test suites. From Figure 4, we can see that the CMRFO, the MRFO, and other improved MRFOs can reach the optimal value on unimodal functions F1–F4, while the optimization ability of other comparison algorithms is not strong. Among them, the CMRFO has the fastest convergence speed. Particularly, for functions F5 and F7, the CMRFO obtains a great ameliorate in the field of convergence accuracy and speed compared with the MRFO, and the CMRFO is also obviously superior to other comparison algorithms. For multimodal functions, the CMRFO is greatly improved compared with the MRFO and the CMRFO performs better than all comparison optimizers. The results on fixed-dimensional multimodal functions for all 11 algorithms have little difference. Compared with 10 comparison optimizers, the CMRFO possesses a good convergence rate at the initial stage of iteration and does not fall into the local optimum. The CMRFO algorithm shows better convergence accuracy and speed. In general, Figure 4 illustrates that the proposed CMRFO has obvious improvements in convergence characteristics and has strong competitiveness.

The statistical results obtained by 11 methods on 23 classical benchmarks are given in Table 6. We can see that the CMRFO ranks first on 19 functions, showing the good optimization ability of the CMRFO. Except for functions F16, F18, F20, and F21, the CMRFO ranks second, and it performs best on the remaining functions. From the final rank in Table 6, the CMRFO is number one and the MRFO is number four, which shows that the CMRFO algorithm has obvious improvement and also shows the advantages of the CMRFO. In addition, the standard deviation of the CMRFO on 19 functions is small, which indicates that the CMRFO is relatively stable on benchmark functions. To sum up, the CMRFO displays excellent performance on 23 classical benchmarks.

The rank sum test (RST) is usually used to verify the distinction of different intelligent optimization algorithms. Table 7 gives the *p*-values of the RST of each algorithm based on the CMRFO at a 95% significance level ($\alpha = 0.05$) on 23 benchmark functions. "+/=/−" is the statistical result using the *p*-value and the rank on each function, which, respectively, represent that the CMRFO is significantly worse than/equal to/better than the comparison optimizer. Note that the data in bold are *p*-values greater than 0.05 in Table 7.

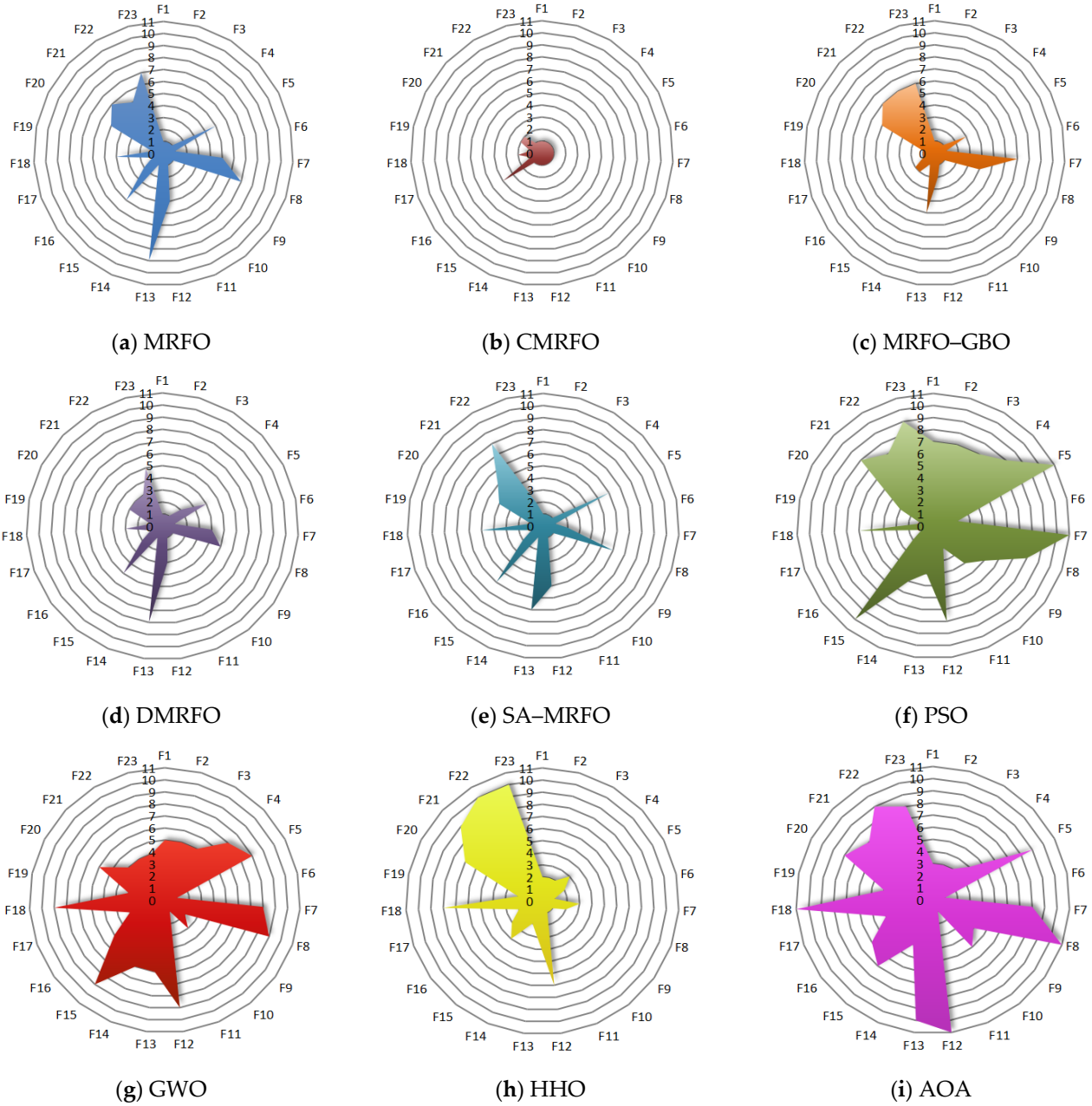
Table 7. Statistical results of each algorithm based on the CMRFO.

No.	Algorithm									
	MRFO	**MRFO–GBO**	**DMRFO**	**SA–MRFO**	**PSO**	**GWO**	**HHO**	**AOA**	**CHOA**	**MPA**
F1	NaN	NaN	NaN	NaN	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}
F2	NaN	NaN	NaN	NaN	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}
F3	NaN	NaN	NaN	NaN	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}
F4	5.2×10^{-2}	3.3×10^{-4}	3.3×10^{-4}	2.1×10^{-2}	4.2×10^{-8}	4.2×10^{-8}	4.2×10^{-8}	4.2×10^{-8}	4.2×10^{-8}	4.2×10^{-8}
F5	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}	6.8×10^{-8}
F6	NaN	NaN	NaN	NaN	3.4×10^{-1}	NaN	NaN	NaN	NaN	NaN
F7	2.3×10^{-5}	3.0×10^{-7}	4.2×10^{-5}	3.6×10^{-2}	6.8×10^{-8}	6.8×10^{-8}	8.4×10^{-3}	1.1×10^{-7}	6.6×10^{-5}	6.8×10^{-8}
F8	3.7×10^{-8}	3.7×10^{-8}	3.7×10^{-8}	3.7×10^{-8}	3.7×10^{-8}	3.7×10^{-8}	3.7×10^{-8}	2.8×10^{-2}	3.7×10^{-8}	3.7×10^{-8}
F9	NaN	NaN	NaN	NaN	8.0×10^{-9}	4.0×10^{-2}	NaN	3.4×10^{-1}	8.1×10^{-2}	NaN
F10	NaN	NaN	NaN	NaN	8.0×10^{-9}	3.7×10^{-9}	NaN	6.7×10^{-5}	8.0×10^{-9}	5.0×10^{-6}
F11	NaN	NaN	NaN	NaN	8.0×10^{-9}	NaN	NaN	NaN	NaN	NaN
F12	1.9×10^{-8}	1.1×10^{-7}	1.9×10^{-8}	1.9×10^{-8}	1.9×10^{-8}	1.9×10^{-8}	1.9×10^{-8}	1.9×10^{-8}	1.9×10^{-8}	1.9×10^{-8}
F13	1.1×10^{-8}	1.5×10^{-8}	1.4×10^{-8}	1.4×10^{-8}	1.5×10^{-8}	1.5×10^{-8}	1.5×10^{-8}	1.5×10^{-8}	1.5×10^{-8}	1.5×10^{-8}
F14	NaN	NaN	NaN	NaN	2.0×10^{-3}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	NaN
F15	2.1×10^{-3}	5.5×10^{-3}	1.4×10^{-1}	4.3×10^{-2}	6.1×10^{-8}	6.1×10^{-8}	6.1×10^{-8}	6.1×10^{-8}	6.1×10^{-8}	4.7×10^{-7}
F16	2.9×10^{-8}	2.9×10^{-8}	2.9×10^{-8}	2.9×10^{-8}	2.9×10^{-8}	1.2×10^{-7}	4.2×10^{-1}	9.1×10^{-8}	6.7×10^{-8}	1.2×10^{-7}
F17	NaN	NaN	NaN	NaN	NaN	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	NaN
F18	9.5×10^{-1}	9.1×10^{-2}	6.2×10^{-1}	3.1×10^{-1}	7.1×10^{-1}	1.5×10^{-8}	5.0×10^{-8}	1.5×10^{-8}	1.5×10^{-8}	5.7×10^{-2}
F19	NaN	NaN	NaN	NaN	NaN	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	NaN
F20	4.5×10^{-2}	4.5×10^{-2}	5.8×10^{-1}	8.5×10^{-2}	5.8×10^{-1}	3.5×10^{-5}	6.8×10^{-7}	1.6×10^{-7}	3.4×10^{-8}	2.7×10^{-2}
F21	2.1×10^{-2}	1.6×10^{-1}	6.9×10^{-4}	5.9×10^{-1}	5.9×10^{-6}	1.5×10^{-8}	1.5×10^{-8}	1.5×10^{-8}	1.5×10^{-8}	6.4×10^{-7}
F22	3.5×10^{-2}	1.9×10^{-1}	3.2×10^{-1}	4.1×10^{-1}	1.9×10^{-1}	4.1×10^{-8}	4.1×10^{-8}	4.1×10^{-8}	4.1×10^{-8}	5.7×10^{-4}
F23	2.0×10^{-2}	8.1×10^{-2}	8.1×10^{-2}	3.4×10^{-1}	6.6×10^{-5}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	8.0×10^{-9}	5.4×10^{-7}
+/=/−	1/12/10	1/14/8	1/15/7	1/15/7	1/6/16	0/2/21	0/5/18	0/3/20	0/3/20	3/7/13

The last row of Table 7 has a statistical result of 1/12/10, 1/14/8, 1/15/7, 1/15/7, 1/6/16, 0/2/21, 0/5/18, 0/3/20, 0/3/20, and 3/7/13. The experimental results indicate

that the CMRFO is superior to the original MRFO on 10 functions. For other improved MRFOs, the CMRFO is significantly better than the MRFO–GBO on eight functions and significantly superior to the DMRFO and the SA–MRFO on seven functions, which fully shows that the CMRFO has obvious advantages over other competitors. To sum up, the capability of the CMRFO is better than that of other comparison algorithms.

Figure 5 displays the radar charts of all 11 algorithms based on the rank on 23 functions. Obviously, the CMRFO has the smallest shadow region, demonstrating its superiority over other MRFOs. By comparing radar charts of the MRFO and other algorithms, we can also see which test functions the CMRFO has improved and performs well.

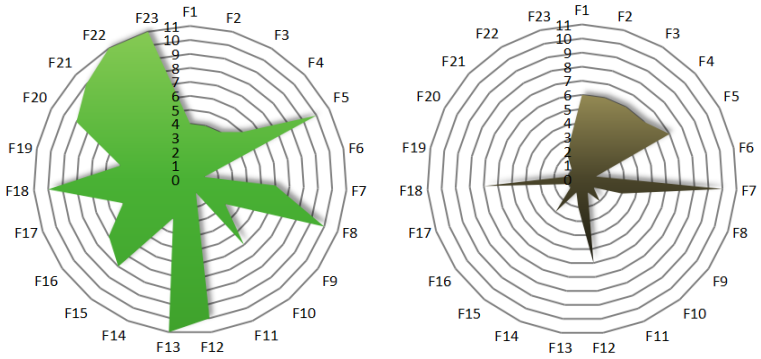


(**a**) MRFO (**b**) CMRFO (**c**) MRFO–GBO

(**d**) DMRFO (**e**) SA–MRFO (**f**) PSO

(**g**) GWO (**h**) HHO (**i**) AOA

Figure 5. *Cont.*



(**j**) CHOA (**k**) MPA

Figure 5. Radar charts of all 11 algorithms.

3.5. Comparison of the CMRFO with Other Optimizers on CEC2020

To further test the optimization capability of the CMRFO, the CMRFO is further tested on the CEC2020 test suite in this section. The comparison algorithms are the MRFO [23], the PSO [46], the SCA [52], the WOA [53], the HHO [48], the AOA [49], the CHOA [50], the SSA [54], and the SOA [55]. Table 8 provides the parameter setting for all optimizers. Numerical results of 10 comparison algorithms on IEEE CEC2020 are enumerated in Figure 6 and Table 9.

Table 8. Parameter settings of algorithms.

Algorithm	Parameter
MRFO	$S = 2$
CMRFO	$S = 2$, $p = 0.1$
PSO	$F_1 = 2$, $F_2 = 2$; s: linearly decreases from 0.8 to 0.2
SCA	$\alpha = 2$
WOA	α: the value range of α is [0, 2]; increases linearly; $b = 1$
HHO	E_0: [−1 1]
CHOA	f: non-linearly decreases from 2.5 to 0; chaotic map: tent map
AOA	$F_1 = F_4 = 2$, $F_2 = 6$, $F_3 = 1$
SSA	$v_0 = 0$
SOA	A: linearly decreases from 2 to 0; $fc = 0$

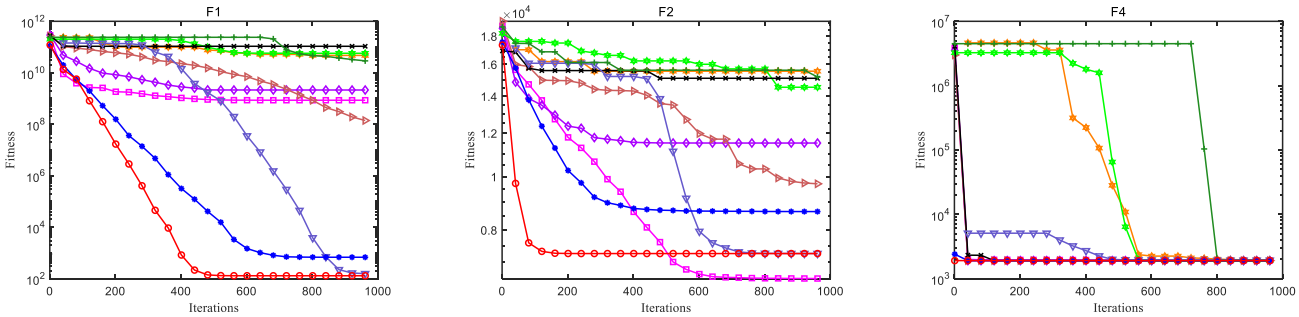


Figure 6. *Cont.*

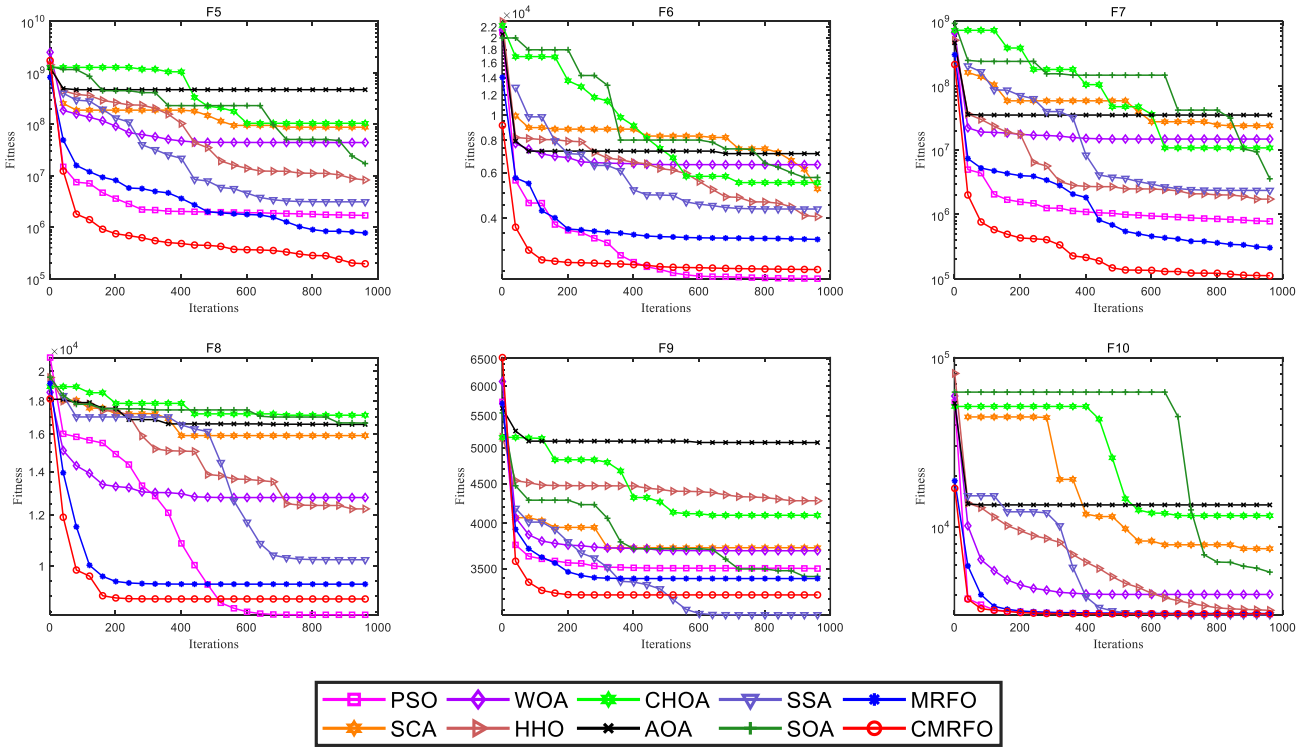


Figure 6. Convergence curves for 10 comparison algorithms (50-dimensional IEEE CEC2020).

Table 9. Results of all 10 algorithms on the 50-dimensional CEC2020 test suite.

No.	Result	Algorithm									
		MRFO	**CMRFO**	**PSO**	**SCA**	**WOA**	**HHO**	**CHOA**	**AOA**	**SSA**	**SOA**
F1	Mean	6.80×10^{3}	2.56×10^{3}	6.68×10^{8}	5.45×10^{10}	3.48×10^{9}	1.48×10^{8}	5.69×10^{10}	1.05×10^{11}	8.03×10^{3}	3.70×10^{10}
	Std	3.75×10^{7}	7.12×10^{6}	9.35×10^{17}	6.69×10^{19}	2.92×10^{18}	1.08×10^{16}	2.53×10^{18}	7.30×10^{19}	5.92×10^{7}	6.81×10^{19}
	Rank	2	1	5	8	6	4	9	10	3	7
F2	Mean	8.58×10^{3}	8.57×10^{3}	7.46×10^{3}	1.52×10^{4}	1.15×10^{4}	1.03×10^{4}	1.56×10^{4}	1.51×10^{4}	8.61×10^{3}	1.24×10^{4}
	Std	1.19×10^{6}	9.48×10^{5}	6.85×10^{5}	1.80×10^{5}	1.13×10^{6}	1.42×10^{6}	1.49×10^{5}	2.58×10^{5}	9.92×10^{5}	1.14×10^{5}
	Rank	3	2	1	9	6	5	10	8	4	7
F3	Mean	1.45×10^{3}	1.51×10^{3}	9.29×10^{2}	1.76×10^{3}	1.79×10^{3}	1.83×10^{3}	1.75×10^{3}	1.99×10^{3}	1.21×10^{3}	1.54×10^{3}
	Std	1.79×10^{4}	6.04×10^{4}	4.08×10^{3}	9.22×10^{3}	1.41×10^{4}	7.42×10^{3}	1.65×10^{3}	4.92×10^{3}	2.55×10^{4}	9.80×10^{3}
	Rank	3	4	1	7	8	9	6	10	2	5
F4	Mean	1.90×10^{3}	1.90×10^{3}	1.91×10^{3}	2.04×10^{3}	1.90×10^{3}	1.90×10^{3}	1.90×10^{3}	1.90×10^{3}	1.93×10^{3}	1.90×10^{3}
	Std	0	0	3.77	1.75×10^{4}	0	0	0	0	1.04×10^{2}	0
	Rank	1	1	2	4	1	1	1	1	3	1
F5	Mean	7.44×10^{5}	4.10×10^{5}	1.76×10^{6}	7.65×10^{7}	1.18×10^{8}	1.01×10^{7}	5.96×10^{7}	4.27×10^{8}	2.62×10^{6}	1.28×10^{7}
	Std	2.25×10^{11}	4.32×10^{10}	6.46×10^{11}	1.60×10^{15}	3.92×10^{15}	2.67×101^{3}	2.09×10^{14}	1.08×10^{16}	1.65×10^{12}	6.12×10^{13}
	Rank	2	1	3	8	9	5	7	10	4	6
F6	Mean	3.36×10^{3}	3.17×10^{3}	2.85×10^{3}	6.34×10^{3}	5.77×10^{3}	4.55×10^{3}	5.79×10^{3}	7.61×10^{3}	3.86×10^{3}	4.22×10^{3}
	Std	2.07×10^{5}	1.15×10^{5}	1.04×10^{5}	4.45×10^{5}	9.15×10^{5}	1.81×10^{5}	1.08×10^{5}	8.27×10^{5}	1.83×10^{5}	2.67×10^{5}
	Rank	3	2	1	9	7	6	8	10	4	5
F7	Mean	3.94×10^{5}	2.36×10^{5}	1.52×10^{6}	2.06×10^{7}	1.33×10^{7}	4.26×10^{6}	2.35×10^{7}	3.72×10^{7}	2.16×10^{6}	7.55×10^{6}
	Std	6.89×10^{10}	1.79×10^{10}	1.43×10^{12}	7.40×10^{13}	3.48×10^{13}	6.53×10^{12}	4.31×10^{13}	2.76×10^{14}	1.88×10^{12}	1.73×10^{13}
	Rank	2	1	3	8	7	5	9	10	4	6
F8	Mean	9356.9042	8852.8621	8545.6711	16798.8094	13060.8894	11558.1408	17340.3701	16235.7414	9446.7043	13085.0313
	Std	6.33×10^{6}	9.47×10^{6}	3.22×10^{6}	2.79×10^{5}	1.15×10^{6}	1.29×10^{6}	1.07×10^{5}	5.20×10^{5}	5.59×10^{5}	1.39×10^{6}
	Rank	3	2	1	9	6	5	10	8	4	7
F9	Mean	3324.5494	3304.7364	3399.3970	3799.0453	3801.2554	4208.3563	4063.366	4906.9563	3155.6406	3306.2197
	Std	1.78×10^{4}	1.07×10^{4}	1.62×10^{4}	4.87×10^{3}	2.79×10^{4}	6.15×10^{4}	6.99×10^{3}	1.48×10^{5}	3.20×10^{3}	4.83×10^{3}
	Rank	4	2	5	6	7	9	8	10	1	3
F10	Mean	3075.667	3059.2648	3077.4469	7425.5938	3592.3986	3209.6626	11,968.0231	14507.0025	3092.3489	5653.4024
	Std	5.95×10^{2}	1.07×10^{3}	2.48×10^{3}	5.82×10^{5}	3.10×10^{4}	1.43×10^{3}	5.49×10^{5}	1.59×10^{6}	6.87×10^{2}	8.37×10^{5}
	Rank	2	1	3	8	6	5	9	10	4	7
Mean Rank		2.50	1.70	2.50	7.60	6.30	5.40	7.70	8.70	3.30	5.40
Result		2	1	2	6	5	4	7	8	3	4

Figure 6 illustrates the convergence curves for 10 comparison algorithms on the 50-dimensional IEEE CEC2020 benchmark. Compared with the native MRFO, the convergence accuracy and speed of the CMRFO are also significantly improved on the 50-dimensional CEC2020 test suite. Compared with other algorithms, the CMRFO can obtain the optimum value quickly at the beginning of the iteration and it can avert running into the local optimum solution at the end of the iteration, which shows excellent search performance. Overall, the CMRFO possesses powerful competitiveness.

The experimental results of 10 comparison algorithms on the 50-dimensional IEEE CEC2020 test suite are given in Table 8. From the last line of Table 8, we can see that the comprehensive rank of the CMRFO is first, which further verifies the superiority and effectiveness of the CMRFO for solving the 50-dimensional CEC2020 test suite.

Figure 7 further plots the boxplots of 10 comparison algorithms on the CEC2020 test suite. Apparently, the CMRFO possesses the narrowest boxplot without outliers with regard to F4, F5, F7, and F10, indicating that the CMRFO has excellent performance and stability for solving these functions. With respect to functions F1 and F8, although CMRFO results have few outliers, the results are more competitive. As for F2, F6, as well as F9, the positions of the CMRFO boxplot are much lower than that of the original MRFO and the median is smaller, indicating that the result of the CMRFO is closer to the optimal value. Based on the above analysis, it can be concluded that the proposed CMRFO is also effective for solving the 50-dimensional CEC2020 test suite.

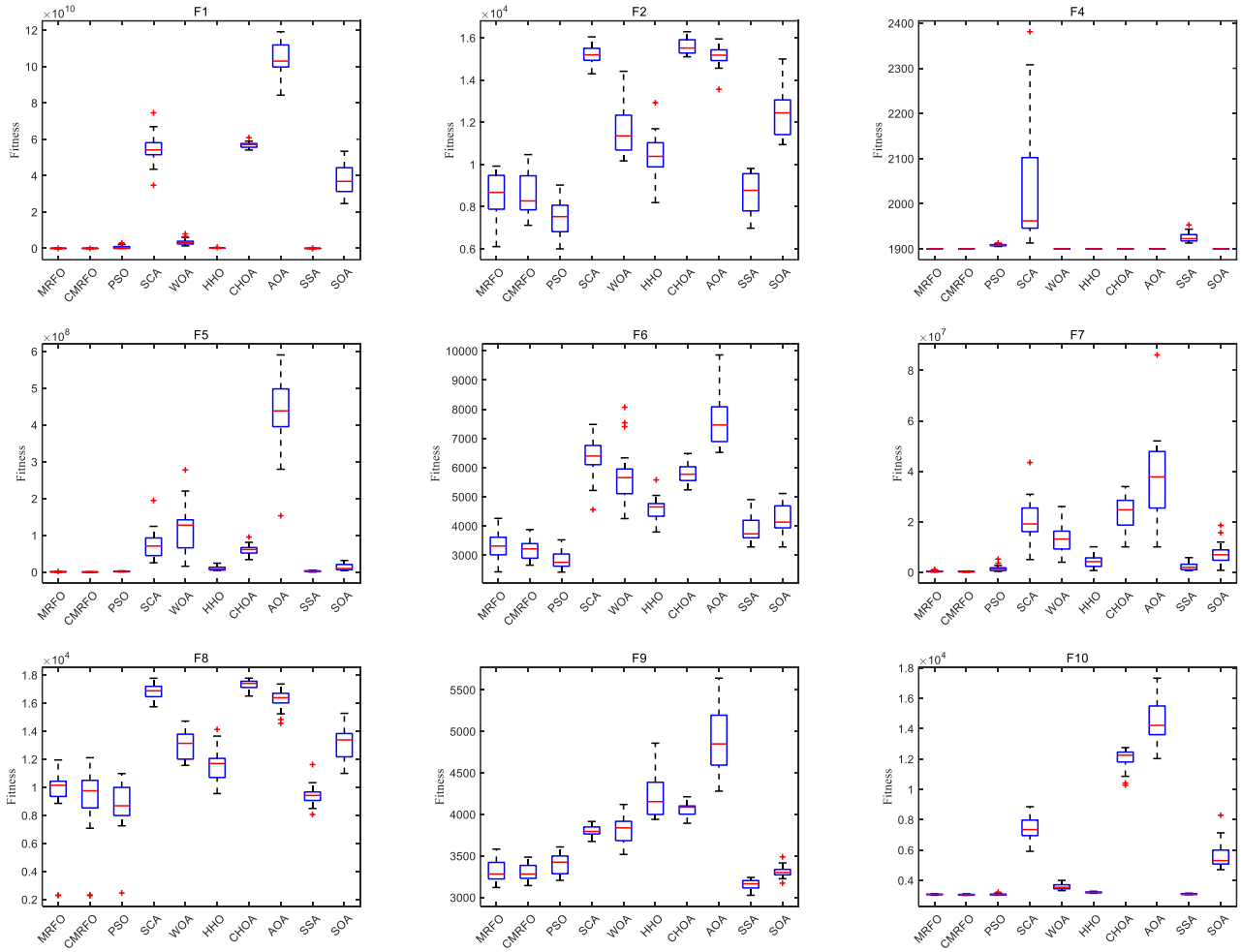


Figure 7. Boxplots of all 10 algorithms on the 50-dimensional CEC2020 test suite.

4. Practical Engineering Application

Next, the CMRFO is used to deal with three practical applications with nonlinear constrained optimization. The result of the CMRFO is compared with other comparison algorithms, including MRFO [23], AO [56], AOA [49], CHOA [50], TSA [57], SCA [52], SOA [55], GWO [47], HHO [48], JS [58], and MPA [51]. For the execution of the engineering application, we guaranteed a population of 50 for all comparison algorithms and a maximum number of iterations of 1000 and ensured that all algorithms were executed 30 times. In general, the practical engineering application with minimization constraints is defined as:

$$\text{Minimize: } f\ (X),\ X = [x_1, x_2, \ldots, x_n] \tag{15}$$

$$\text{Subject to : } \begin{cases} g_i(X) \le 0,\ i = 1,\ \ldots,\ m. \\ h_j(X) = 0,\ j = 1,\ \ldots,\ n. \end{cases} \tag{16}$$

where m is the number of multiple constraints and l is the number of balance constraints [6]. Thus, the engineering optimization equation after constraint weighting is described as:

$$f(X) = f(X) + \alpha \sum_{i=1}^{m} \max\{g_i(X),\ 0\} + \beta \sum_{j=1}^{n} \max\{h_j(X),\ 0\}. \tag{17}$$

where α is the weight of multiple constraints and β is the weight of balance constraints.

4.1. Pressure Vessel (PV) Design

Figure 8 shows graphically the structure of PV with four variables [51]. The objective function (OF) of the corresponding optimization task is the entire cost of PV. The objective function needs to be minimized by optimizing four design variables.

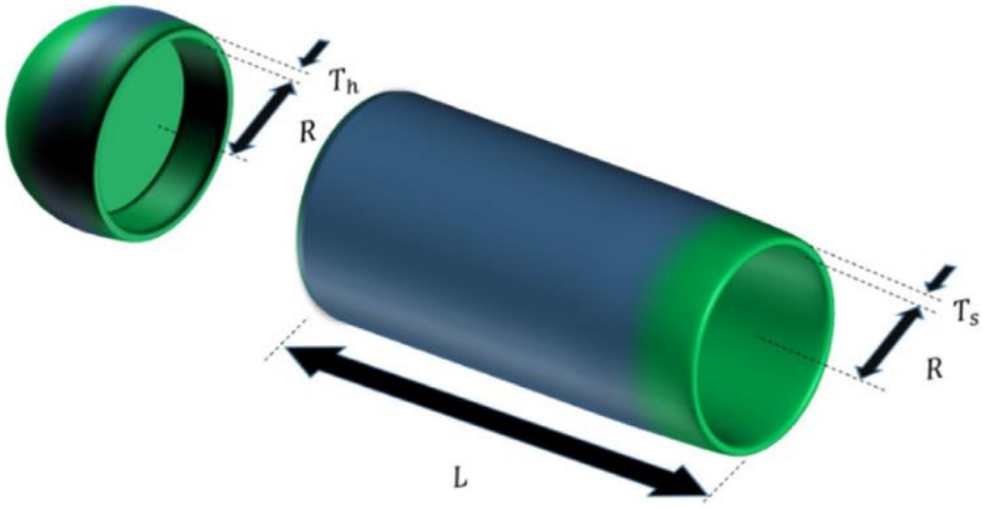


Figure 8. Structure of the pressure vessel.

Let $\boldsymbol{W} = [w_1, w_2, w_3, w_4] = [T_s, T_h, R, L]$. The optimization model of the PV design task can be mathematically described in detail as follows:

$$\begin{aligned} &\min\ PV(\boldsymbol{W}) = 0.6224 w_1 w_3 w_4 + 1.7781 w_2 w_3^2 + 3.1661 w_1^2 w_4 + 19.84 w_1^2 w_3 \\ &s.t. \begin{cases} c_1(\boldsymbol{W}) = 0.0193 w_3 - w_1 \le 0,\ c_2(\boldsymbol{W}) = 0.00954 w_3 - w_2 \le 0, \\ c_3(\boldsymbol{W}) = -\pi w_3^2 w_4 - \frac{4}{3}\pi w_3^3 + 1{,}296{,}000 \le 0,\ c_4(\boldsymbol{W}) = w_4 - 240 \le 0, \end{cases} \end{aligned} \tag{18}$$

where $0 \le w_1, w_2 \le 99,\ 10 \le w_3, w_4 \le 200$.

Table 10 gives the minimum cost and design variables of 11 algorithms to solve the PV optimization problem. Table 11 gives the PV optimization statistical analyses of all

11 algorithms running 30 times. The simulation results show that the obtained value of the CMRFO is the smallest and the CMRFO is more effective than 10 comparison algorithms for solving PV optimization design.

Table 10. Results of all 11 algorithms.

Algorithm	Variable				Minimum Cost
	z_1	z_2	z_3	z_4	
MRFO	0.7745007	0.3832242	40.31993	199.9957	5870.1394
CMRFO	0.7745476	0.3832055	40.31962	200.0000	5870.1240
AO	0.8585610	0.4232512	44.60451	149.0881	6061.6264
AOA	1.0039500	4.4602500	53.50010	72.7940	27,253.6871
CHOA	1.3046100	0.6133140	66.01640	10.0000	7843.5750
TSA	0.7724745	0.3829585	40.36756	200.0000	5895.4548
SCA	0.8249876	0.4522591	41.32788	187.4612	6313.5757
SOA	0.8284806	0.4054728	42.92673	166.8016	5984.1297
GWO	0.7891866	0.3899311	41.06881	189.8338	5895.9798
HHO	0.9126808	0.4497941	47.41997	120.2297	6150.9174
JS	0.7745259	0.3831937	40.32009	199.9941	5870.1564

Table 11. Statistical analyses of all 11 algorithms after 30 runs.

Algorithm	Best	Worst	Mean	Std
MRFO	5870.1245	5871.2569	5870.2136	6.22×10^{-2}
CMRFO	5870.1240	5870.1240	5870.1240	4.62×10^{-11}
AO	5988.6403	7564.7221	6666.1339	2.34×10^{5}
AOA	6662.4718	142,714.8674	33,101.3667	1.69×10^{9}
CHOA	7528.6200	362,414.1763	77,881.9124	1.45×10^{10}
TSA	5873.3548	6418.7976	5956.9796	1.75×10^{4}
SCA	6171.7885	6859.0257	6406.6324	3.74×10^{4}
SOA	5878.9664	753,596.6961	80,183.5623	2.93×10^{10}
GWO	5870.1956	7171.1804	5943.8058	8.42×10^{4}
HHO	6015.6406	7482.8086	6684.3887	1.50×10^{5}
JS	5870.1240	5873.7386	5871.1915	1.34

4.2. Tension/Compression Spring (TCS) Optimization Problem

The design task on TCS mathematically is an optimization problem with three optimization variables, and its structure is illustrated in Figure 9 [51]. The optimization model of the design task is given in Equation (16), whose objective function with four constraints is the total weight of TCS. The total weight is minimized by optimizing the parameters in the model.

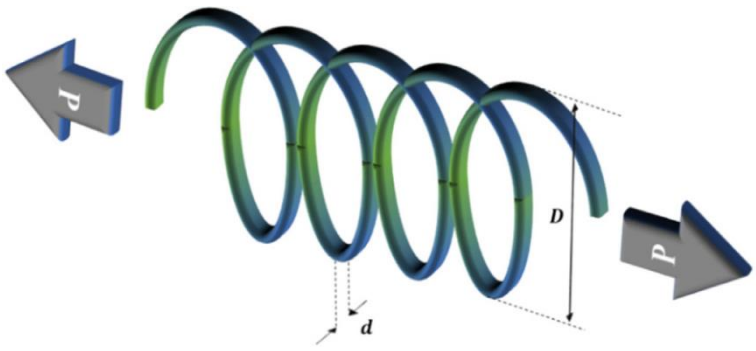


Figure 9. Structure of TCS.

$$\text{Consider } \mathbf{W} = [w_1, w_2, w_3] = [d, D, N],$$

$$\min spring(\mathbf{W}) = (w_3 + 2)w_2 w_1^2$$
$$s.t. \begin{cases} g_1(\mathbf{W}) = \frac{4w_2^2 - w_1 w_2}{12{,}566(w_2 w_1^3 - w_1^4)} + \frac{1}{5108 w_1^2} - 1 \le 0,\ g_2(\mathbf{W}) = 1 - \frac{140.45 w_1}{w_2^2 w_3} \le 0, \\ g_3(\mathbf{W}) = 1 - \frac{w_2^3 w_3}{71{,}785 w_1^4} \le 0,\ g_4(\mathbf{W}) = \frac{w_1 + w_2}{1.5} - 1 \le 0, \end{cases} \tag{19}$$

where $w_1 \in [0.05,\ 2]$, $w_2 \in [0.25,\ 1.3]$, $w_3 \in [2,\ 15]$.

Table 12 gives the minimum weight and design variables of 11 algorithms to finish this optimization design task. Meanwhile, Table 13 gives the comparison results of all 11 algorithms after 30 runs, where the data in bold are the optimal values of the 11 algorithms. From Tables 12 and 13, we can see that the simulation results of all 11 comparison methods are not significantly different, but the CMRFO can still provide a better solution to this optimization problem.

Table 12. Results of all 11 algorithms.

Algorithm	Variable			Minimum Weight
	z_1	z_2	z_3	
MRFO	0.0517557	0.358265	11.2078	0.012675
CMRFO	0.0516888	0.356712	11.2893	0.012665
AO	0.0572140	0.504390	6.5055	0.014044
AOA	0.0638360	0.725450	3.1224	0.015143
CHOA	0.0500000	0.316611	14.2597	0.012870
TSA	0.0526618	0.379585	10.1016	0.012739
SCA	0.0500000	0.316202	14.2037	0.012809
SOA	0.0500000	0.317083	14.0790	0.012746
GWO	0.0506104	0.331238	12.9618	0.012694
HHO	0.0564420	0.482210	6.4974	0.013053
JS	0.0520076	0.364425	10.8523	0.012668

Table 13. Comparison results of all 11 algorithms after 30 runs.

Algorithm	Best	Worst	Mean	Std
MRFO	0.012667	0.012754	0.012688	5.43×10^{-10}
CMRFO	0.012666	0.012695	0.012679	7.95×10^{-11}
AO	0.012983	0.020436	0.015930	5.25×10^{-6}
AOA	0.012907	0.016086	0.013938	9.99×10^{-7}
CHOA	0.012856	0.017668	0.014543	3.28×10^{-6}
TSA	0.012713	0.013048	0.012801	5.45×10^{-9}
SCA	0.012742	0.013197	0.012982	1.01×10^{-8}
SOA	0.012730	0.012798	0.012758	3.83×10^{-10}
GWO	0.012669	0.012766	0.012708	6.44×10^{-10}
HHO	0.012805	0.014597	0.013365	3.30×10^{-7}
JS	0.012671	0.012770	0.012710	1.02×10^{-9}

4.3. Pressure Vessel (PV) Design

The OF of the mathematical model for WB optimization design is the entire cost of WB, which minimizes the objective function of the model by finding a set of feasible problem variables [51]. The structure of this design is shown graphically in Figure 10. The mathematical model of WB design is defined in Equation (17), where $W = [w_1, w_2, w_3, w_4] = [h, l, t, b]$.

$$\begin{aligned} &\min Weidedbeam(\boldsymbol{W}) = 1.10471w_1^2w_2 + 0.04811w_3w_4(14.0 + w_2) \\ &s.t.\begin{cases} h_1(\boldsymbol{W}) = \tau(\boldsymbol{W}) - \tau_{\max} \le 0,\ h_2(\boldsymbol{W}) = \sigma(\boldsymbol{W}) - \sigma_{\max} \le 0, \\ h_3(\boldsymbol{W}) = \delta(\boldsymbol{W}) - \delta_{\max} \le 0,\ h_4(\boldsymbol{W}) = w_1 - w_4 \le 0,\ h_5(\boldsymbol{W}) = P - P_c(\boldsymbol{W}) \le 0, \\ h_6(\boldsymbol{W}) = 0.125 - w_1 \le 0,\ h_7(\boldsymbol{W}) = 1.10471w_1^2 + 0.04811w_3w_4(14.0 + w_2) - 5.0 \le 0, \end{cases} \end{aligned} \tag{20}$$

in which $0.1 \le w_1, w_4 \le 2.0$, $0.1 \le w_2, w_3 \le 10.0$, τ_{max} = 136,000 *psi*, σ_{max} = 36,600 *psi*, δ_{max} = 0.25 *in*, and P = 6000 *lb*. The formulas for $\tau(\boldsymbol{W})$, $\sigma(\boldsymbol{W})$, $P_c(\boldsymbol{W})$, and $\delta(\boldsymbol{W})$. are:

$$\tau(\boldsymbol{W}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{w_2}{2R} + (\tau'')^2},\ \tau' = \frac{P}{\sqrt{2}w_1w_2},\ \tau'' = \frac{MR}{J},\ M = P(L + \frac{w_2}{2}),\ R = \sqrt{\frac{w_2^2}{4} + (\frac{w_1 + w_3}{2})^2},$$

$$J = 2\sqrt{2}w_1w_2\left[\frac{w_2^2}{4} + (\frac{w_1 + w_3}{2})^2\right],\ \sigma(\boldsymbol{W}) = \frac{6PL}{w_4w_3^2},\ \delta(\boldsymbol{W}) = \frac{6PL^3}{Ew_3^2w_4},\ P_c(\boldsymbol{W}) = \frac{4.013E\sqrt{\frac{w_3^2w_4^6}{36}}}{L^2}(1 - \frac{w_3}{2L}\sqrt{\frac{E}{4G}}),$$

where $L = 14in$ and $E = 30 \times 10^6$.

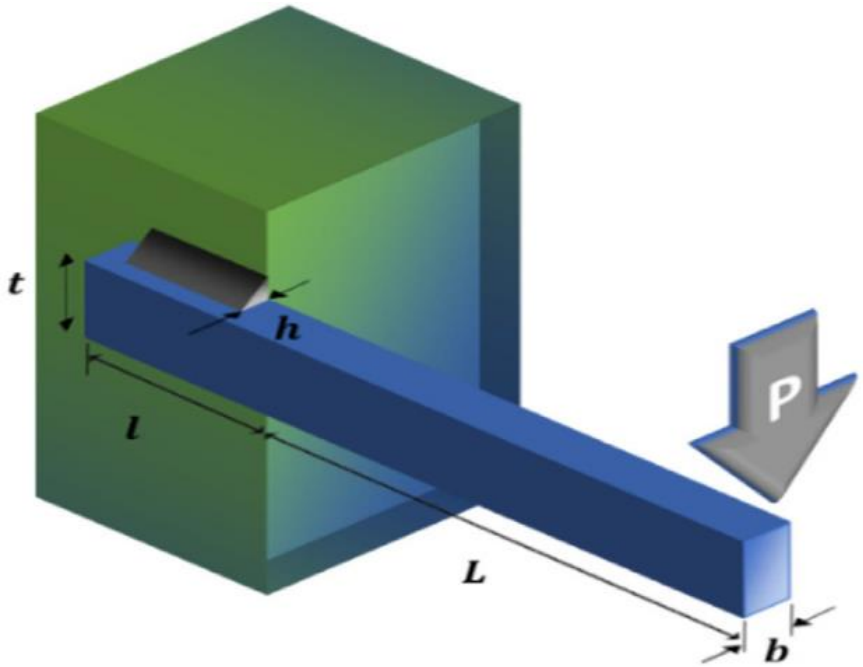


Figure 10. Structure of WB.

Table 14 shows the simulation results of 11 algorithms to solve the WB optimization problem. Table 15 shows the WB optimization statistical analyses of 11 algorithms after 30 runs. From the results of the experiment, the CMRFO, the MRFO, JS, and MPA, all perform well and can obtain the same minimum value. However, Table 15 shows that the CMRFO has a smaller standard deviation, indicating that the CMRFO is more stable in solving welded beam optimization problems.

Table 14. Results of all 11 algorithms.

Algorithm	Variable				Minimum Cost
	z_1	z_2	z_3	z_4	
MRFO	0.20573	3.2531	9.0366	0.20573	1.6952
CMRFO	0.20573	3.2531	9.0366	0.20573	1.6952
AO	0.16706	7.5426	8.9751	0.21036	2.1893
CHOA	0.13745	5.2800	8.9767	0.21607	1.9093
TSA	0.20558	3.2838	9.0614	0.20599	1.7054
SCA	0.18648	3.6064	9.1512	0.20602	1.7355
SOA	0.19182	3.5352	9.0470	0.20578	1.7143
GWO	0.20408	3.2834	9.0390	0.20573	1.6973
HHO	0.19402	3.6019	8.8153	0.21619	1.7636
JS	0.20573	3.2531	9.0366	0.20573	1.6952
MPA	0.20573	3.2531	9.0366	0.20573	1.6952

Table 15. Results of all 11 algorithms after 30 runs.

Algorithm	Best	Worst	Mean	Std
MRFO	1.6952	1.6952	1.6952	1.48×10^{-19}
CMRFO	1.6952	1.6952	1.6952	6.49×10^{-21}
AO	1.7456	2.2250	1.8935	1.32×10^{-2}
CHOA	1.7621	1.9147	1.8597	1.48×10^{-3}
TSA	1.7013	1.7209	1.7105	2.57×10^{-5}
SCA	1.7755	1.9083	1.8190	1.23×10^{-3}
SOA	1.6996	1.7465	1.7100	1.33×10^{-4}
GWO	1.6955	1.6994	1.6971	1.52×10^{-6}
HHO	1.7248	1.8616	1.7949	1.66×10^{-3}
JS	1.6952	1.6952	1.6952	3.81×10^{-20}
MPA	1.6952	1.6952	1.6952	1.06×10^{-15}

From what has been discussed above, the numerical results of three practical engineering applications show that the CMRFO is more effective than comparison optimizers in dealing with practical engineering applications.

5. Real-World Application: Construction of CG-Ball Curves with Optimal Shape

Then, the CMRFO is used to address a challenging real-world optimization problem further. An optimization model of CG-Ball curves based on minimum curvature variation in curves is established in this section. CG-Ball curves are a new kind of parametric curve containing three shape control parameters, which are generalized cubic Ball curves. What is more, its advantage is that the shape of the curve can be freely changed by using the control parameters.

5.1. Shape Optimization Model: Minimum Curvature Variation in Curves

Given four control points $\boldsymbol{P}_i$ $(i = 0,\ 1,\ 2,\ 3) \in \mathbf{R}^2$ or $\mathbf{R}^3$, the following parametric equation

$$\boldsymbol{P}(s;\boldsymbol{\Omega}) = \sum_{i=0}^{3} \boldsymbol{P}_i b_{i,4}(s) \tag{21}$$

is for cubic generalized Ball (CG-Ball) curves [36], where $t \in [0,\ 1]$ and $\boldsymbol{\Omega} = (\alpha, \beta, \gamma)$ are three shape parameters. The basic functions $B_{i,4}(s)$, $i = 0,\ 1,\ 2,\ 3$ are defined as follows:

$$\begin{cases} b_{0,4}(s) = [1 - \alpha s(1-s)](1-s)^2, \\ b_{1,4}(s) = (2 + \alpha - \alpha s - \beta s)s(1-s)^2, \\ b_{2,4}(s) = (2 + \gamma\, s + \beta - \beta s)s^2(1-s), \\ b_{3,4}(s) = (1 - \gamma\, s + \gamma\, s^2)\, s^2, \end{cases} \tag{22}$$

where $-2 \le \alpha,\ \gamma \le 4,\ -2 \le \beta \le 2$.

The curvature variation (CV) of CG-Ball curves is given by Equation (23).

$$CV(\mathbf{\Omega}) = \int_0^1 \left\| \boldsymbol{P}^{(3)}(s;\mathbf{\Omega}) \right\|^2 \mathrm{d}s \tag{23}$$

Further calculation shows that

$$\begin{aligned} \left\| \boldsymbol{P}^{(3)}(s;\mathbf{\Omega}) \right\|^2 &= \left\| b_{0,4}^{(3)}(s)\boldsymbol{P}_0 \right\|^2 + \left\| b_{1,4}^{(3)}(s)\boldsymbol{P}_1 \right\|^2 + \left\| b_{2,4}^{(3)}(s)\boldsymbol{P}_2 \right\|^2 + \left\| b_{3,4}^{(3)}(s)\boldsymbol{P}_3 \right\|^2 \\ &+2b_{0,4}^{(3)}(s)b_{1,4}^{(3)}(s)\boldsymbol{P}_0\boldsymbol{P}_1 + 2b_{0,4}^{(3)}(s)b_{2,4}^{(3)}(s)\boldsymbol{P}_0\boldsymbol{P}_2 + 2b_{0,4}^{(3)}(s)b_{3,4}^{(3)}(s)\boldsymbol{P}_0\boldsymbol{P}_3 \\ &+2b_{1,4}^{(3)}(s)b_{2,4}^{(3)}(s)\boldsymbol{P}_1\boldsymbol{P}_2 + 2b_{1,4}^{(3)}(s)b_{3,4}^{(3)}(s)\boldsymbol{P}_1\boldsymbol{P}_3 + 2b_{2,4}^{(3)}(s)b_{3,4}^{(3)}(s)\boldsymbol{P}_2\boldsymbol{P}_3. \end{aligned} \tag{24}$$

Substituting Equation (24) into Equation (23), we have

$$\begin{aligned} CV(\mathbf{\Omega}) &= \int_0^1 \left\| \boldsymbol{P}^{(3)}(s;\mathbf{\Omega}) \right\|^2 \mathrm{d}s \\ &= l_0\|\boldsymbol{P}_0\|^2 + l_1\|\boldsymbol{P}_1\|^2 + l_2\|\boldsymbol{P}_2\|^2 + l_3\|\boldsymbol{P}_3\|^2 \\ &+ l_4\boldsymbol{P}_0\boldsymbol{P}_1 + l_5\boldsymbol{P}_0\boldsymbol{P}_2 + l_6\boldsymbol{P}_0\boldsymbol{P}_3 + l_7\boldsymbol{P}_1\boldsymbol{P}_2 + l_8\boldsymbol{P}_1\boldsymbol{P}_3 + l_9\boldsymbol{P}_2\boldsymbol{P}_3 \end{aligned} \tag{25}$$

where

$l_0 = 84\alpha^2$, $l_1 = 84\alpha^2 + 96\alpha\beta + 144\alpha + 48\beta^2 + 144$,
$l_2 = 48\beta^2 - 96\beta\gamma + 84\gamma^2 + 144\gamma + 144$, $l_3 = 84\gamma^2$,
$l_4 = -12\alpha(7\alpha + 4\beta + 6)$, $l_5 = 12\alpha(4\beta - \gamma + 6)$,
$l_6 = 12\alpha\gamma$, $l_7 = 12\alpha\gamma - 72\gamma - 48\alpha\beta - 72\alpha + 48\beta\gamma - 48\beta^2 - 144$,
$l_8 = -12\gamma(\alpha + 4\beta - 6)$, $l_9 = -12\gamma(7\gamma - 4\beta + 6)$.

Finally, the shape optimization model based on minimum curvature variation in curves is established, and the specific formula is:

$$\begin{aligned} &\arg\min C(\mathbf{\Omega}) = \int_0^1 \left\| \boldsymbol{P}^{(3)}(s;\mathbf{\Omega}) \right\|^2 \mathrm{d}s = l_0\|\boldsymbol{P}_0\|^2 + l_1\|\boldsymbol{P}_1\|^2 + l_2\|\boldsymbol{P}_2\|^2 + l_3\|\boldsymbol{P}_3\|^2 \\ &\qquad + l_4\boldsymbol{P}_0\boldsymbol{P}_1 + l_5\boldsymbol{P}_0\boldsymbol{P}_2 + l_6\boldsymbol{P}_0\boldsymbol{P}_3 + l_7\boldsymbol{P}_1\boldsymbol{P}_2 + l_8\boldsymbol{P}_1\boldsymbol{P}_3 + l_9\boldsymbol{P}_2\boldsymbol{P}_3, \\ &\text{s.t. } \alpha,\ \gamma \in [-2,\ 4],\ \beta \in [-2,\ 2] \end{aligned} \tag{26}$$

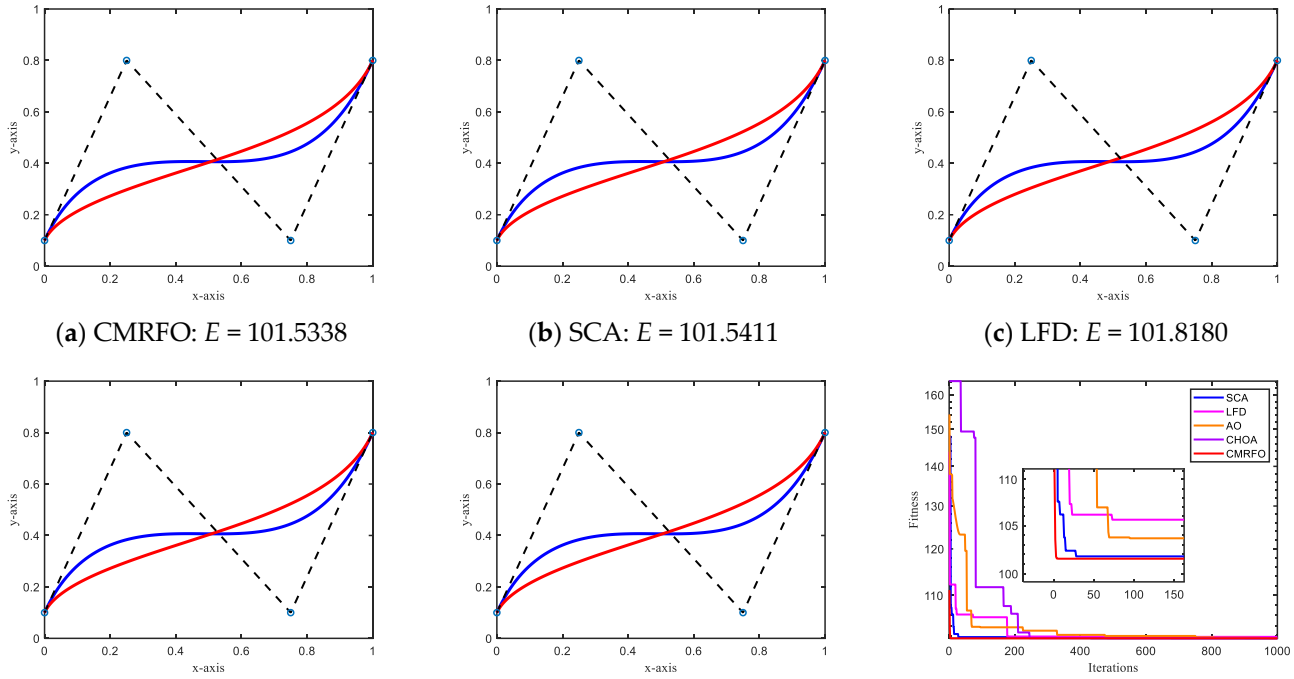
5.2. Modeling Examples

This section will optimize the CG-Ball curves shape according to the proposed optimization model. Two examples are given to prove the effectiveness of the CMRFO in solving the established optimization model in Equation (26).

Example 1. *The shape optimization of plane-S-type CG-Ball curves and the convergence curve of the CMRFO are given in this example. The control points of CG-Ball curves are*

$$\boldsymbol{P}_0 = (0, 0.1),\ \boldsymbol{P}_1 = (0.25, 0.8),\ \boldsymbol{P}_2 = (0.75, 0.1),\ \boldsymbol{P}_3 = (1, 0.8)$$

Figure 11 shows the optimized CG-Ball curves with the minimum curvature variation obtained by five algorithms. Figure 11a–e shows the CG-Ball curves with optimal shape obtained by the proposed CMRFO, SCA [52], LFD [59], AO [56], and CHOA [50] algorithms, respectively. The red curve is the curve with optimized shape parameter values, and all shape parameters of the blue curve have the value of 1, which demonstrates that CG-Ball curves have flexible shape adjustability. The convergence curves of the five algorithms are illustrated in Figure 11f. The optimal values of shape parameters and objective function obtained by all five algorithms are shown in Table 16. According to the experimental results, the CMRFO has the best effect on solving the optimization model and the curvature variation in CG-Ball curves obtained by the CMRFO is minimum among all five algorithms (its value is 101.5338). Meanwhile, the CMRFO has the fastest convergence speed compared with other algorithms.



(**a**) CMRFO: E = 101.5338 (**b**) SCA: E = 101.5411 (**c**) LFD: E = 101.8180

(**d**) AO: E = 101.5378 (**e**) CHOA: E = 101.5374 (**f**) Convergence curves

Figure 11. Shape optimization of CG-Ball curves.

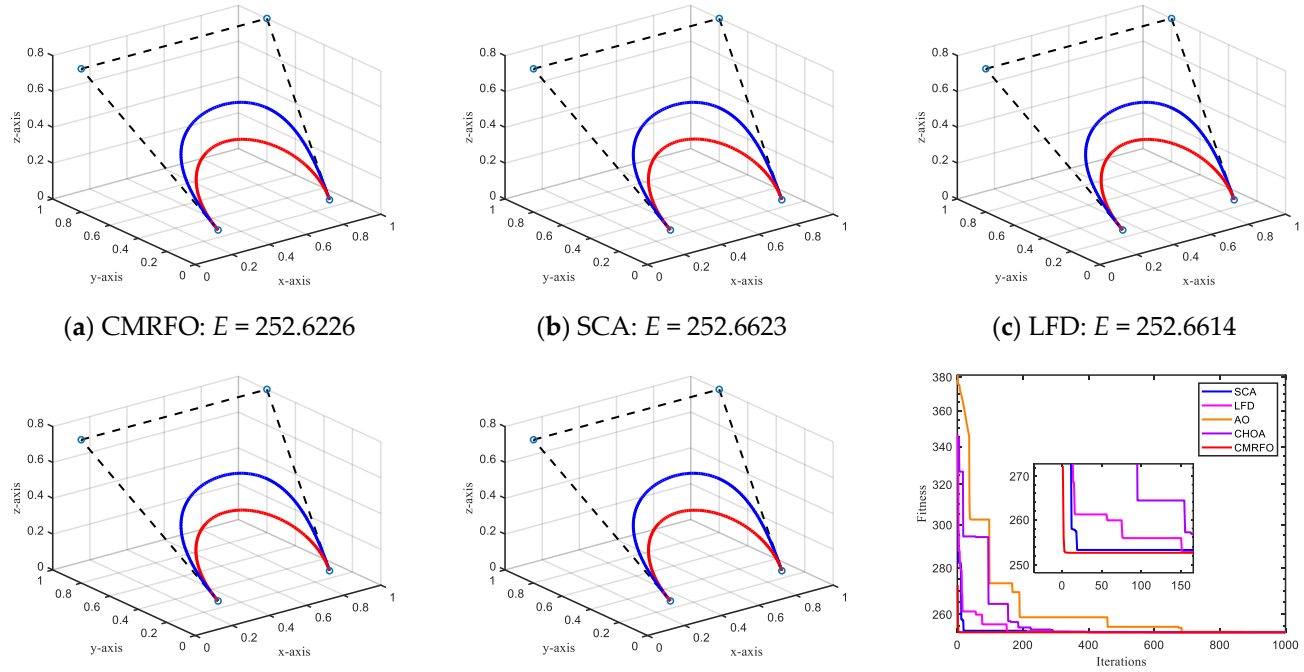
Table 16. Optimal values of shape parameters and corresponding objective function.

Algorithm	α	β	γ	Objective Function
CMRFO	−0.91829	0.37267	−0.24463	101.5338
SCA	−0.92874	0.38598	−0.24488	101.5411
LFD	−0.85734	0.29792	−0.28173	101.8180
AO	−0.92616	0.38264	−0.24147	101.5378
CHOA	−0.91621	0.36598	−0.25012	101.5374

Example 2. *The shape optimization of space-M-type CG-Ball curves and convergence curves of the optimization model when the curvature variation in curves converges to the optimal value are given graphically in this example. Take the control points of the curves as*

$$\boldsymbol{P}_0 = (0.2,\ 0.1,\ 0.1),\ \boldsymbol{P}_1 = (0,\ 0.8,\ 0.8), \boldsymbol{P}_2 = (1,\ 0.8,\ 0.8),\ \boldsymbol{P}_3 = (0.8,\ 0.1,\ 0.1)$$

Figure 12 displays the CG-Ball curves after optimization. Figure 12a–e displays the CG-Ball curves after shape optimization using five different algorithms, and the curvature variation in the curves is the smallest. The red curve is the curve with optimized shape parameters, and all shape parameters of the blue curve have the value of 1. Figure 12f shows the convergence of the curvature variation in CG-Ball curves. Table 17 gives the optimal shape parameters and corresponding objective function after shape optimization using five different algorithms. It can be seen that the curvature variation in the CG-Ball curve obtained by the CMRFO is the smallest. The optimal value of the objective function is 252.6226.



(**a**) CMRFO: $E = 252.6226$ (**b**) SCA: $E = 252.6623$ (**c**) LFD: $E = 252.6614$

(**d**) AO: $E = 252.6538$ (**e**) CHOA: $E = 252.6233$ (**f**) Convergence curves

Figure 12. Shape optimization of CG-Ball curves.

Table 17. Optimal values of shape parameters and corresponding fitness value.

Algorithm	α	β	γ	Objective Function
CMRFO	−0.44182	−0.17432	−0.58420	252.6226
SCA	−0.43369	−0.16005	−0.57312	252.6623
LFD	−0.46342	−0.16430	−0.58057	252.6614
AO	−0.42406	−0.17529	−0.58248	252.6538
CHOA	−0.44194	−0.17604	−0.58321	252.6233

6. Conclusions

This paper develops an improved manta ray foraging optimizer (CMRFO) that combines chaos initialization, opposition-based learning, and elite chaotic searching strategy with higher accuracy and a faster convergence rate. Various correction techniques are used in the MRFO. First, a chaos-based initialization strategy is introduced to help explore the search space comprehensively and improve the algorithm's efficiency during the search. In addition, an opposition-based learning strategy improves the solution quality of the solution, further improving the algorithm's solution quality. Finally, the elite chaotic searching strategy is introduced to achieve the update of elite individuals and enhance the optimization capability. Fourteen different chaotic maps are used to initialize the population. It has been proved that a cubic map can provide better searchability. In addition, the effect of critical parameters (elite selection proportion) on CMRFO sensitivity is discussed. The CMRFO is competitive compared with numerous advanced intelligent algorithms on 23 benchmark functions, the well-known IEEE CEC2020 test suite, and three practical engineering applications. In addition, a mathematical model of shape optimization for CG-Ball curves is established and the CMRFO is used to solve the established shape optimization model contrasted with four popular advanced algorithms. Numerical results further verify the effectiveness and practicability of the CMRFO in solving challenging optimization problems in the engineering field.

Future work can extend the proposed CMRFO to other examples of CG-Ball curve shape optimization. In addition, multiobjective optimization of CG-Ball curve shapes

can be considered. We also believe in the application of the proposed CMRFO in other engineering fields, for example feature selection, surface optimization, and path planning.

Supplementary Materials: The following supporting information can be downloaded at https://www.mdpi.com/article/10.3390/math10162960/s1: The Matlab codes of the proposed CMRFO algorithm and other optimization algorithms.

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

Funding: This work is supported by the Natural Science Foundation of Xijing University (Grant No. XJ190214).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data generated or analyzed during this study are included in this published article (and its Supplementary Information Files).

Acknowledgments: The authors are grateful to the reviewers for their insightful suggestions and comments, which helped us to improve the presentation and content of the paper.

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

References

1. Hackwood, S.; Beni, G. Self-organization of sensors for swarm intelligence. In Proceedings of the 1992 IEEE International Conference on Robotics and Automation, Nice, France, 12–14 May 1992; pp. 819–829.
2. Holland, J.H. Genetic algorithms. *Sci. Am.* **1992**, *267*, 66–73. [CrossRef]
3. Barshandeh, S.; Dana, R.; Eskandarian, P. A learning automata-based hybrid MPA and JS algorithm for numerical optimization problems and its application on data clustering. *Knowl. Based Syst.* **2021**, *236*, 107682. [CrossRef]
4. Hu, G.; Du, B.; Wang, X.F.; Wei, G. An enhanced black widow optimization algorithm for feature selection. *Knowl. Based Syst.* **2022**, *235*, 107638. [CrossRef]
5. Hassan, M.H.; Kamel, S.; Abualigah, L.; Eid, A. Development and application of slime mould algorithm for optimal economic emission dispatch. *Expert Syst. Appl.* **2021**, *182*, 115205. [CrossRef]
6. Hu, G.; Zhong, J.; Du, B.; Wei, G. An enhanced hybrid arithmetic optimization algorithm for engineering applications. *Comput. Methods Appl. Mech. Eng.* **2022**, *394*, 114901. [CrossRef]
7. Hu, G.; Zhu, X.N.; Wei, G.; Chang, C.T. An improved marine predators algorithm for shape optimization of developable Ball surfaces. *Eng. Appl. Artif. Intell.* **2021**, *105*, 104417. [CrossRef]
8. Elsisi, M.; Ebrahim, M. Optimal design of low computational burden model predictive control based on SSDA towards autonomous vehicle under vision dynamics. *Int. J. Intell. Syst.* **2021**, *36*, 6968–6987. [CrossRef]
9. Elsisi, M.; Ismail, M.; Bendary, A. Optimal design of battery charge management controller for hybrid system PV/wind cell with storage battery. *Int. J. Power Energy Convers.* **2020**, *11*, 412. [CrossRef]
10. Elsisi, M.; Mahmoud, K.; Lehtonen, M.; Darwish, M.M.F. Effective Nonlinear Model Predictive Control Scheme Tuned by Improved NN for Robotic Manipulators. *IEEE Access* **2021**, *9*, 64278–64290. [CrossRef]
11. Elsisi, M.; Soliman, M.; Aboelela, M.; Mansour, W. ABC based design of PID controller for two area load frequency control with nonlinearities. *Telkomnika Indones. J. Electr. Eng.* **2015**, *16*, 58–64. [CrossRef]
12. Elsisi, M.; Tran, M.-Q.; Hasanien, H.M.; Turky, R.A.; Albalawi, F.; Ghoneim, S.S.M. Robust Model Predictive Control Paradigm for Automatic Voltage Regulators against Uncertainty Based on Optimization Algorithms. *Mathematics* **2021**, *9*, 2885. [CrossRef]
13. Elsisi, M. Optimal design of nonlinear model predictive controller based on new modified multitracker optimization algorithm. *Int. J. Intell. Syst.* **2020**, *35*, 1857–1878. [CrossRef]
14. Zheng, J.Y.; Hu, G.; Ji, X.; Qin, X. Quintic generalized Hermite interpolation curves: Construction and shape optimization using an improved GWO algorithm. *Comput. Appl. Math.* **2022**, *41*, 115. [CrossRef]
15. Elsisi, M.; Abdelfattah, H. New design of variable structure control based on lightning search algorithm for nuclear reactor power system considering load-following operation. *Nucl. Eng. Technol.* **2020**, *52*, 544–551. [CrossRef]
16. Hu, G.; Dou, W.; Wang, X.; Abbas, M. An enhanced chimp optimization algorithm for optimal degree reduction of Said–Ball curves. *Math. Comput. Simul.* **2022**, *197*, 207–252. [CrossRef]

17. Zhao, W.; Zhang, Z.; Mirjalili, S.; Wang, L.; Khodadadi, N.; Mirjalili, S.M. An effective multi-objective artificial hummingbird algorithm with dynamic elimination-based crowding distance for solving engineering design problems. *Comput. Methods Appl. Mech. Eng.* **2022**, *398*, 115223. [CrossRef]
18. Xie, Z.; Zhang, C.; Shao, X.; Lin, W.; Zhu, H. An effective hybrid teaching–learning-based optimization algorithm for permutation flow shop scheduling problem. *Adv. Eng. Softw.* **2014**, *77*, 35–47. [CrossRef]
19. Hu, G.; Li, M.; Wang, X.; Wei, G.; Chang, C.-T. An enhanced manta ray foraging optimization algorithm for shape optimization of complex CCG-Ball curves. *Knowl. Based Syst.* **2022**, *240*, 108071. [CrossRef]
20. Elsisi, M. Improved grey wolf optimizer based on opposition and quasi learning approaches for optimization: Case study autonomous vehicle including vision system. *Artif. Intell. Rev.* **2022**, 1–24. [CrossRef]
21. Hu, G.; Du, B.; Li, H.N.; Wang, X.P. Quadratic interpolation boosted black widow spider-inspired optimization algorithm with wavelet mutation. *Math. Comput. Simul.* **2022**, *200*, 428–467. [CrossRef]
22. Zhao, W.; Shi, T.; Wang, L.; Cao, Q.; Zhang, H. An adaptive hybrid atom search optimization with particle swarm optimization and its application to optimal no-load PID design of hydro-turbine governor. *J. Comput. Des. Eng.* **2021**, *8*, 1204–1233. [CrossRef]
23. Zhao, W.G.; Zhang, Z.X.; Wang, L.Y. Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications. *Eng. Appl. Artif. Intell.* **2020**, *87*, 103300. [CrossRef]
24. Houssein, E.H.; Zaki, G.N.; Diab, A.A.Z.; Younis, E.M.G. An efficient manta ray foraging optimization algorithm for parameter extraction of three-diode photovoltaic model. *Comput. Electr. Eng.* **2021**, *94*, 107304. [CrossRef]
25. Fathy, A.; Rezk, H.; Yousri, D. A robust global MPPT to mitigate partial shading of triple-junction solar cell-based system using manta ray foraging optimization algorithm. *Sol. Energy* **2020**, *207*, 305–316. [CrossRef]
26. Ben, U.C.; Akpan, A.E.; Enyinyi, E.O.; Awak, E. Novel technique for the interpretation of gravity anomalies over geologic structures with idealized geometries using the manta ray foraging optimization. *J. Asian Earth Sci. X* **2021**, *6*, 100070. [CrossRef]
27. Ben, U.C.; Akpan, A.E.; Mbonu, C.C.; Ebong, E.D. Novel methodology for interpretation of magnetic anomalies due to two-dimensional dipping dikes using the manta ray foraging optimization. *J. Appl. Geophys.* **2021**, *192*, 104405. [CrossRef]
28. El-Hameed, M.A.; Elkholy, M.M.; El-Fergany, A.A. Three-diode model for characterization of industrial solar generating units using manta-rays foraging optimizer: Analysis and validations. *Energy Convers. Manag.* **2020**, *219*, 113048. [CrossRef]
29. Houssein, E.H.; Ibrahim, I.E.; Neggaz, N.; Hassaballah, M.; Wazery, Y.M. An efficient ECG arrhythmia classification method based on manta ray foraging optimization. *Expert Syst. Appl.* **2021**, *181*, 115131. [CrossRef]
30. Hemeida, M.G.; Ibrahim, A.A.; Mohamed, A.A.; Alkhalaf, S.; El-Dine, A.M.B. Optimal allocation of distributed generators DG based manta ray foraging optimization algorithm (MRFO). *Ain Shams Eng. J.* **2021**, *12*, 609–619. [CrossRef]
31. Elmaadawy, K.; Elaziz, M.A.; Elsheikh, A.H.; Moawad, A.; Liu, B.C.; Lu, S.F. Utilization of random vector functional link integrated with manta ray foraging optimization for effluent prediction of wastewater treatment plant. *J. Environ. Manag.* **2021**, *298*, 113520. [CrossRef]
32. Got, A.; Zouache, D.; Moussaoui, A. MOMRFO: Multi-objective manta ray foraging optimizer for handling engineering design problems. *Knowl. Based Syst.* **2022**, *237*, 107880. [CrossRef]
33. Zouache, D.; Abdelaziz, F.B. Guided manta ray foraging optimization using epsilon dominance for multi-objective optimization in engineering design. *Expert Syst. Appl.* **2022**, *189*, 116126. [CrossRef]
34. Kahraman, H.T.; Akbel, M.; Duman, S. Optimization of optimal power flow problem using multi-objective manta ray foraging optimizer. *Appl. Soft Comput.* **2022**, *116*, 108334. [CrossRef]
35. Elaziz, M.A.; Yousri, D.; Al-Qaness, M.A.A.; AbdelAty, A.M.; Radwan, A.G.; Ewees, A.A. A Grunwald–Letnikov based manta ray foraging optimizer for global optimization and image segmentation. *Eng. Appl. Artif. Intell.* **2021**, *98*, 104105. [CrossRef]
36. Hu, S.M.; Wang, G.Z.; Jin, T.G. Properties of two types of generalized Ball curves. *Comput. Aided Des.* **1996**, *28*, 125–133. [CrossRef]
37. Yousri, D.; AbdelAty, A.M.; Al-qaness, M.A.A.; Ewees, A.A.; Radwan, A.G.; Elaziz, M.A. Discrete fractional-order Caputo method to overcome trapping in local optima: Manta ray foraging optimizer as a case study. *Expert Syst. Appl.* **2022**, *192*, 116355. [CrossRef]
38. Xu, H.T.; Song, H.Q.; Xu, C.X.; Wu, X.W.; Yousefi, N. Exergy analysis and optimization of a HT-PEMFC using developed manta ray foraging optimization algorithm. *Int. J. Hydrog. Energy* **2020**, *45*, 30932–30941. [CrossRef]
39. Jena, B.; Naik, M.K.; Panda, R.; Abraham, A. Maximum 3D Tsallis entropy based multilevel thresholding of brain MR image using attacking manta ray foraging optimization. *Eng. Appl. Artif. Intell.* **2021**, *103*, 104293. [CrossRef]
40. Feng, J.Y.; Luo, X.G.; Gao, M.Z.; Abbas, A.; Xu, Y.P.; Pouramini, S. Minimization of energy consumption by building shape optimization using an improved manta-ray foraging optimization algorithm. *Energy Rep.* **2021**, *7*, 1068–1078. [CrossRef]
41. Liu, B.Z.; Wang, Z.Z.; Feng, L.; Jermsittiparsert, K. Optimal operation of photovoltaic/diesel generator/pumped water reservoir power system using modified manta ray optimization. *J. Clean. Prod.* **2021**, *289*, 125733. [CrossRef]
42. Sheng, B.Q.; Pan, T.H.; Luo, Y.; Jermsittiparsert, K. System identification of the PEMFCs based on balanced manta-ray foraging optimization algorithm. *Energy Rep.* **2020**, *6*, 2887–2896. [CrossRef]
43. Micev, M.; Ćalasan, M.; Ali, Z.M.; Hasanien, H.M.; Aleem, S.H.E.A. Optimal design of automatic voltage regulation controller using hybrid simulated annealing—Manta ray foraging optimization algorithm. *Ain Shams Eng. J.* **2021**, *12*, 641–657. [CrossRef]
44. Hassan, M.H.; Houssein, E.H.; Mahdy, M.A.; Kamel, S. An improved manta ray foraging optimizer for cost-effective emission dispatch problems. *Eng. Appl. Artif. Intell.* **2021**, *100*, 104155. [CrossRef]

45. Jain, S.; Indora, S.; Atal, D.K. Rider manta ray foraging optimization-based generative adversarial network and CNN feature for detecting glaucoma, Biomed. *Signal Process. Control* **2022**, *73*, 103425. [CrossRef]
46. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; pp. 1942–1948.
47. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
48. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H.L. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872. [CrossRef]
49. Hashim, F.A.; Hussain, K.; Houssein, E.H.; Mabrouk, M.S.; Al-Atabany, W. Archimedes optimization algorithm: A new metaheuristic algorithm for solving optimization problems. *Appl. Intell.* **2021**, *51*, 1531–1551. [CrossRef]
50. Khishe, M.; Mosavi, M.R. Chimp optimization algorithm. *Expert Syst. Appl.* **2020**, *149*, 113338. [CrossRef]
51. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine predators algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]
52. Mirjalili, S. SCA: A sine cosine algorithm for solving optimization problems. *Knowl. Based Syst.* **2016**, *96*, 120–133. [CrossRef]
53. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
54. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
55. Dhiman, G.; Kumar, V. Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowl. Based Syst.* **2019**, *165*, 169–196. [CrossRef]
56. Abualigah, L.; Yousri, D.; Elaziz, M.A.; Ewees, A.A.; Al-qaness, M.A.A.; Gandomi, A.H. Aquila optimizer: A novel meta-heuristic optimization algorithm. *Comput. Ind. Eng.* **2021**, *157*, 107250. [CrossRef]
57. Kaur, S.; Awasthi, L.K.; Sangal, A.L.; Dhiman, G. Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103541. [CrossRef]
58. Chou, J.S.; Truong, D.N. A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. *Appl. Math. Comput.* **2021**, *389*, 125535. [CrossRef]
59. Houssein, E.H.; Saad, M.R.; Hashim, F.A.; Shaban, H.; Hassaballah, M. Lévy flight distribution: A new metaheuristic algorithm for solving engineering optimization problems. *Eng. Appl. Artif. Intell.* **2020**, *94*, 103731. [CrossRef]