Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 8398768, 15 pages
https://doi.org/10.1155/2022/8398768

Research Article

An Enhanced Moth-Flame Optimization with Multiple Flame Guidance Mechanism for Parameter Extraction of Photovoltaic Models

Zhenyu Wang,[1] Zijian Cao,[1] Chen Liu,[1] Haowen Jia,[1] Feng Tian,[2] and Fuxi Liu[3]

[1]*School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710021, China*
[2]*Bournemouth University, Poole, UK*
[3]*School of Mechanical and Electrical Engineering, Hunan Applied Technology University, Changde 415100, China*

Correspondence should be addressed to Fuxi Liu; liufx28@163.com

Received 25 November 2021; Revised 25 April 2022; Accepted 11 May 2022; Published 11 June 2022

Academic Editor: Diego Oliva

How to accurately and efficiently extract photovoltaic (PV) model parameters is the primary problem of photovoltaic system optimization. To accurately and efficiently extract the parameters of PV models, an enhanced moth-flame optimization (EMFO) with multiple flame guidance mechanism is proposed in this study. In EMFO, an adaptive flame number updating mechanism is used to adaptively control the flame number, which enhances the local and global exploration capabilities of MFO. Meanwhile, a multiple flame guidance mechanism is designed for the full use of the position information of flames, which enhances the global diversity of the population. The EMFO is evaluated with other variants of the MFO on 25 benchmark functions of CEC2005, 28 functions of CEC2017, and 5 photovoltaic model parameter extraction problems. Experimental results show that the EMFO has obtained a better performance than other compared algorithms, which proves the effectiveness of the proposed EMFO. The method proposed in this study provides MFO researchers with ideas for adaptive research and making full use of flame population information.

1. Introduction

The optimization problem is an application technology based on mathematics, applied to solve various engineering problems. The solution to most problems that can be transformed into optimization problems will be gained by using the optimization problem technology. However, some problems are nondeterministic polynomial (NP)-hard problems, and it is difficult to find the optimal solution by utilizing the optimization problem technology. To find the global optimal solution to the optimized problems, a series of intelligent algorithms have been proposed in the literature. In recent decades, optimization algorithms have been widely used to optimize NP-hard problems [1], which can be classified into four categories, i.e., evolutionary algorithms, physics-based algorithms, swarm intelligence-based algorithms, and human-based algorithms. In 1975, Holland [2] proposed a genetic algorithm (GA) based on the evolutionary process of organisms. In 1982, Kirkpatrick et al. [3] simulated the annealing process of metallurgy for designing the simulated annealing (SA) algorithm. In 1991, Colorni et al. [4] proposed the ant colony optimization (ACO), which is inspired from the colony behavior of ants' foraging. In 1995, Eberhart and Keendy et al. [5] proposed a particle swarm optimization (PSO) based on the foraging behavior of birds. In 1997, Storn and Price [6] proposed differential evolution (DE) based on GA. In the last decade, many excellent algorithms have been proposed to solve complex engineering problems [7], such as monarch butterfly optimization (MBO) [8, 9], kill herd [10], slime mold algorithm (SMA) [11], harris hawks optimization (HHO) [12], earthworm optimization algorithm (EWA) [13], elephant herding optimization (EHO) [14], moth search (MS) [15] and naked mole-rat (NMR) [16]. In addition, many

optimization algorithms have been utilized to successfully solve the real-life applications [17–23].

These algorithms have successfully optimized various complex optimization problems and have also been widely used to solve engineering, scientific, and image-processing problems. Although a lot of research has been performed on these algorithms, they still face some challenges. For example, PSOs are prone to premature when it is used to solve many multimodal problems. Therefore, it is still a challenge to develop a population-based heuristic search algorithm to effectively prevent falling into the local optimum and maintain a faster convergence rate.

The moth-flame optimization (MFO) is a natural heuristic algorithm proposed by Mirjalili [24] in 2015. The algorithm is based on a special navigation mechanism, in which moths use lateral positioning when flying around the moonlight at night. Since MFO is simple, flexible, and easily implemented, MFO has been widely used in engineering and scientific fields.

In literature [25], MFO was used to solve the optimal power-flow problem. The experimental results demonstrated that MFO is robust and superior to other algorithms. Sapre et al. [26] proposed the opposition-based moth-flame optimization (OMFO), in which the opposition-based learning mechanism, Cauchy mutation, and boundary constraint-handing method were integrated into the MFO algorithm. Experiment results showed that the OMFO has obtained high optimization ability and fast convergence speed, and can effectively jump out of the local optimum. Zhao et al. [27] proposed an ameliorated moth-flame optimization (AMFO), in which a crisscross mechanism was utilized to improve the search ability of MFO.

Xu et al. [28] proposed an enhanced moth-flame optimizer with a mutation strategy for global optimization, in which Gaussian mutation, Cauchy mutation, and Lévy mutation were combined with MFO (GMFO, CMFO, and LMFO), respectively. To optimize the pose of target, a new improved MFO combined with DE was designed by Li et al. [29]. In [30–32], and a chaotic map method was used to control different parameters of MFO. To effectively jump out of the local optimum, a Lévy flight-based moth-flame optimization (LMFO) was proposed by Li et al. [33]. Sayed and Hassanien [34] proposed that MFO is integrated into the SA algorithm (SA-MFO), in which the exploration ability of the SA and the learning mechanism of the MFO were utilized to balance the global exploration and local exploitation of the population. The PSO is embedded into MFO for enhancing the performance of the canonical MFO in solving real-world complex problems [35–37].

Nadimi-Shahraki et al. [38] proposed a migration-based MFO (M-MFO) algorithm for avoiding premature. In the M-MFO, a random migration operator was utilized to improve the position of the poor moth in the early iterations. Furthermore, new qualified solutions are separately stored in a guided archive to keep population diversity. Nadimi-Shahraki et al. [39] proposed a multitrial vector-based MFO (MTV-MFO) algorithm. In the MTV-MFO, the multitrial vector (MTV) mechanism is used to replace the movement strategy of MFO for the use of a combination of different movement strategies, each of which is adjusted to accomplish a particular behavior. Truong [40] proposed a new MFO algorithm, in which a new encoding scheme was used to help reduce the search domain and speed up the computing time by utilizing a shorter length binary vector.

Although those variants described above have improved the performance of the canonical MFO to a certain extent, they exhibit poor performance in solving nondifferentiable, nonlinear, and nonseparable multimodal problems and especially did not solve two important defects of MFO. For example, the number of flames linearly decreases, regardless of whether the population falls in the local optimal trap. This weakens the ability of the population to jump out of the local trap. Furthermore, the original flame-guiding mechanism only distributes the information of a flame to the moth so that the moth is easy to fall into local optimal traps. To solve the above problems, an enhanced MFO (EMFO) with multiple flame guidance is proposed in this study. In the EMFO, to improve the global search capability of the algorithm, the flame number is adaptively controlled according to the current population situation. Meanwhile, a multiple flame guidance mechanism is designed to fully utilize the information of the flames. To verify the performance of the EMFO, it is compared with the state-of-the-art algorithms on 25 benchmark functions of CEC 2005 and a real-world problem.

The main contributions of this study can be summarized as follows:

(1) An adaptive flame number updating mechanism is proposed to enhance the global exploration ability of the population by adaptively controlling the flame number according to the population situation.

(2) To enhance the search ability and make full use of the information of the flames, a multiple flame guidance mechanism is designed by utilizing the information of multiple flames.

(3) To systematically verify the superiority of the proposed EMFO algorithm, 25 complex benchmark functions and parameter extraction of photovoltaic models are utilized to estimate the overall performance of EMFO.

The section of this study is arranged as follows. Section 2 explains the basic concept of the MFO algorithm. Section 3 addresses a specific description of the EMFO algorithm. In Section 4, the experimental results are presented. The real-world problem simulation is described in Section 5. Section 6 summarizes the study and the prospects for future research.

2. Moth-Flame Optimization Algorithm

The MFO is a heuristic optimization algorithm proposed by Mirjalili in 2015. This algorithm is based on the special navigation mechanism of moths. Figure 1 shows the conceptual model of horizontal positioning of moths, which simulates the process that moths fly around the moonlight at a fixed angle. Since the moon is very far away from the moths, the moths can keep flying straight using this

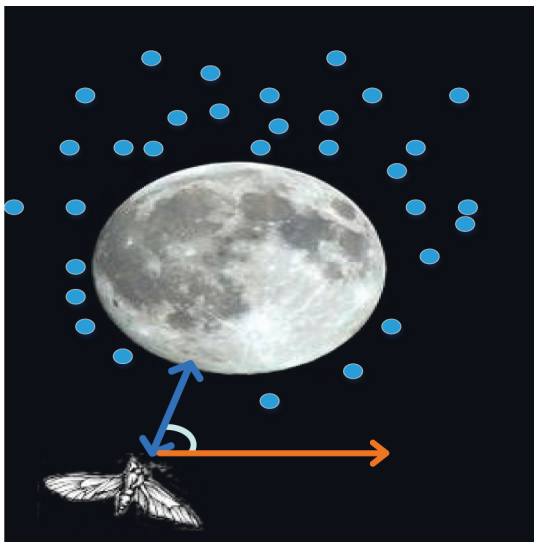


Figure 1: Horizontal positioning.

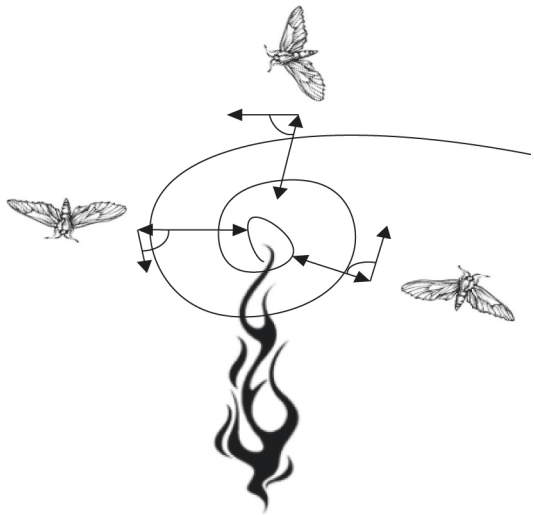


Figure 2: Spiral flight path around the light source.

approximate parallel light. Although this navigation mechanism is very effective for moths, there are many artificial or natural light sources in reality. This kind of light source is very close to the moth compared with the moon. The moths fly around the light source at a fixed angle, which leads to navigation failure and a spiral flight path, as shown in Figure 2.

In the MFO, there are two candidate solutions (moth and flame). The moth is a moving entity in the search space, utilized to find the optimal solution. The position of the best moth obtained so far is viewed as flame. The moth searches around the flame and updates its position to find a better solution in each iteration. The moth population M consists of n moths, as represented in the following formula:

$$M = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1d} \\ m_{21} & m_{22} & \cdots & m_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nd} \end{bmatrix}, \tag{1}$$

where n denotes the number of moths, and d stands for the dimensions of the problem. For all moths, the following set OM is used to store the fitness values.

$$OM = \begin{bmatrix} OM_1 \\ OM_2 \\ \vdots \\ OM_n \end{bmatrix}, \tag{2}$$

where OM_i denotes the fitness value of the i th moth.

The other core of the MFO is the flame population F, which is represented in the following formula:

$$F = \begin{bmatrix} F_{11} & F_{12} & \cdots & F_{1d} \\ F_{21} & F_{22} & \cdots & F_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ F_{n1} & F_{n2} & \cdots & F_{nd} \end{bmatrix}. \tag{3}$$

The set OF of fitness values is described in the following formula:

$$OF = \begin{bmatrix} OF_1 \\ OF_2 \\ \vdots \\ OF_n \end{bmatrix}, \tag{4}$$

where OF_i denotes the fitness value of the i th flame.

Both moths and flames are candidate solutions. The difference between them is the way that they are treated and updated in each iteration. In the search space, moths are the actual moving subjects, and flames are obtained the best position so far. Each moth searches around a flame, and the current optimal solution stored in the flame is updated when a better solution is found. Under this mechanism, the moth will not miss its optimal solution.

The position of each moth is updated according to the following formula:

$$M_i = S\left(M_i, F_j\right), \tag{5}$$

where M_i represents the i th moth, and F_j represents the j th flame. Besides, S represents the spiral function.

According to the behavior of the moth spiral approaching the flame, a logarithmic spiral search is defined as follows:

$$S\left(M_i, F_j\right) = D_i \cdot e^{bt} \cdot \cos\left(2\pi t\right) + F_j, \tag{6}$$

where b is a constant for defining the shape of the logarithmic spiral, and t is a random in the range $[-1, 1]$. D_i is the distance between moth and flame, which is expressed as follows:

$$D_i = |F_j - \mathrm{M_i}| \tag{7}$$

where M_i represents the i th moth, and F_j represents the j th flame. Meanwhile, D_i indicates the distance of the i th moth for the j th flame.

As shown in Figure 3, the t parameter in the spiral search defines how close the next position of the moth should be to the flame ($t = -1$ is the closest position to the flame, while $t = 1$ is the farthest position). To further enhance the exploitation global exploration capability and of the MFO, a parameter r is used to improve the search speed of moths and define the value range of t to $[r, 1]$. In the iteration process, the r linearly decreases from -1 to -2. The spiral search shows that the moths can surround the flames instead of just flying in the space between them, which balances the global exploration capability and local exploitation capability of MFO.

To increase the possibility of finding a better solution, the best solution obtained so far is considered to be the flame. Therefore, the matrix F always contains the best solution obtained so far. After each iteration, the flame position will be reordered according to the fitness value for updating the flame sequence, as shown in Figure 4. In the next generation, moths update their positions based on the flames, which are in the flame sequence corresponding to them.

If each position of n moths is updated based on n different positions in the solution space, the local exploitation capability of the algorithm will be greatly reduced. To solve this problem, a reduction mechanism of the flame number is proposed to time-varyingly reduce the number of flames in the iterative process, as presented in the following formula:

$$\mathrm{Flame\_no} = \mathrm{round}\left(n - l * \frac{n-1}{T}\right), \tag{8}$$

where n is the maximum number of flames, and l is the current iteration number. Besides, Flame_no stands for the number of flames, and T is the maximum iteration number.

3. Enhanced MFO with Multiple Flame Guidance Mechanism

In the basic MFO, the reduction in the number of flames is based on a fixed time-varying reduction mechanism, which makes MFO easy to fall into the local optimal solution trap. To solve this problem, an adaptive flame number updating mechanism is utilized to adaptively control the number of flames. In addition, a multiple flame guidance mechanism is used to enhance the search ability of the population by using simultaneously multiple flames to search.

3.1. Adaptive Flame Number Updating Mechanism. When the global optimal value does not change every ten generations, the flame quantity will be updated as follows:

$$\begin{cases} \mathrm{Flame\_no} = \mathrm{Flame\_no} + \max \mathrm{Success}, \\ \max \mathrm{Success} = \max \mathrm{Success} + \max \mathrm{Success} * \mathrm{adjust\ Success}. \end{cases} \tag{9}$$

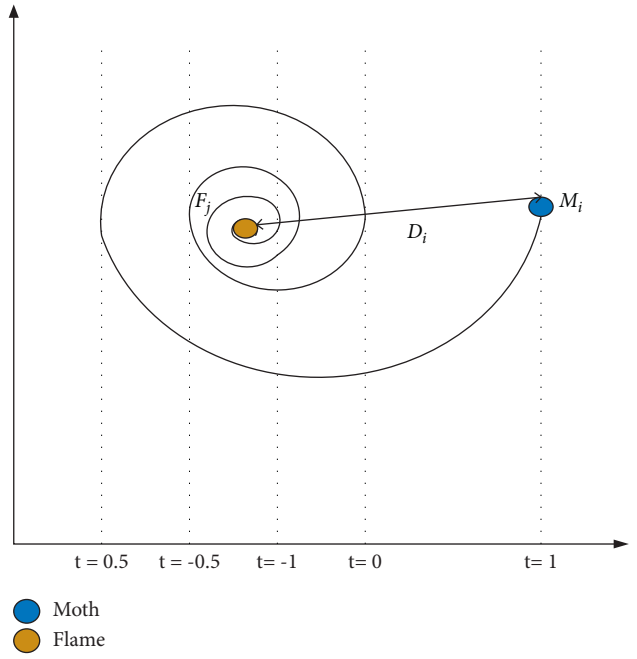


Figure 3: The position of the space around the logarithmic spiral flame relative to t.

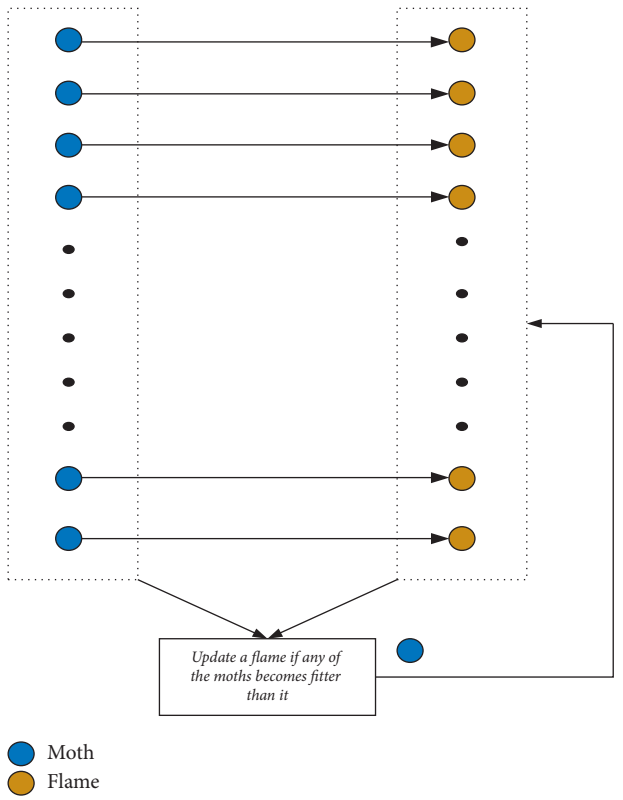


Figure 4: Flame assign mechanism.

On the contrary, the updating formula of flame's number is as follows:

$$\begin{cases} \text{Flame\_no} = \text{Flame\_no} - \max \text{Failures}, \\ \max \text{Failures} = \max \text{Failures} + \max \text{Failures} * \text{adjust Failures}, \end{cases} \tag{10}$$

where max Success and max Failures are the initial values of the updating step of flame's number, Flame_no denotes the number of flames. The adjust Success and adjust Failures are change rates, which are utilized to determine the degree of change of max Success and max Failures, respectively.

In this mechanism, the number of flames will exponentially increase when the global optimal solution is not updated for hundreds of generations, which can effectively solve the problem that the probabilistic stagnation of the algorithm leads to a drastic change in the flame's number. In addition, after the global optimal solution is updated for hundreds of generations, the reduction step of flame's number gradually decreases so that the population will maintain the diversity.

3.2. Multiple Flame Guidance Mechanism. Inspired by DE, the difference vector of the two flames randomly selected from the flame population is regarded as the direction vector that is used to guide another flame to search, which can increase the diversity of the population. To balance the multiple flame guidance mechanism and the spiral search mechanism of the MFO, a judgment parameter rc is added and randomly selected from the range [0, 1]. The multiple flame guidance mechanism is executed when $ran\,d < rc$ is not satisfied, as shown in formula (11). The schematic diagram in space of this mechanism is shown in Figure 5.

$$S\left(M_i, F_j\right) = \begin{cases} D_i \cdot e^{bt} \cdot \cos\left(2\pi t\right) + F_j, & if\ r \text{ and} < rc, \\ r \text{ and} * \left(F_m - F_p\right) + F_j, & \text{otherwise}, \end{cases} \tag{11}$$

where j, m, and p are distinct indexes. M_i represents the i th moth, and F_j represents the j th flame. $ran\,d$ is a random in interval [0, 1].

Based on the above description, the pseudocode of the EMFO algorithm is presented in Algorithm 1.

4. Experimental Simulation and Results

In the simulation environment of this experiment, the CPU is Intel(R) Core (R) i7-7500. The CPU main frequency is 2.70 GHz, and memory is 12.00 GB. The MATLAB R2018b is selected as the development environment.

To verify the performance of the EMFO, some classical variants of the MFO and the original MFO are selected, for example, MFO with Gaussian mutation (GMFO) [28], MFO with Cauchy mutation (CMFO) [28], MFO with Lévy-flight mutation (LMFO) [33], ameliorated MFO (AMFO) [27], and opposition-based MFO (OMFO) [26]. A total of 25 benchmark functions of CEC 2005 [41] are used to test the performance of the EMFO and compared algorithms. In 25 benchmark functions, the functions F1-F5 are continuous unimodal functions while F6-F14 are multimodal and have a significant number of local minima. Besides, F15-F25 are hybrid composition functions.

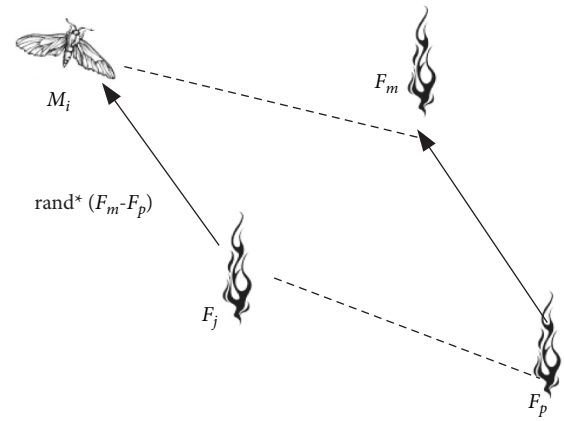


Figure 5: Schematic diagram of multiple flame guidance mechanism.

Additionally, to further verify the performance of the EMFO and the contribution of each component of the proposed algorithm, two state-of-the-art algorithms (EMFO [42] and SaDN [43]), MFO with adaptive flame number (MFO-AFN) updating mechanism, and MFO with multiple flame guidance (MFO-MFG) mechanism were added to the comparison experiment for solving 28 functions of CEC2017 [44]. A total of 28 benchmark functions were composed of 1 unimodal function F1, 7 simple multimodal functions (F3-F9), 10 hybrid functions (F10-F19), and 10 composition functions (F20-F29). It is noteworthy that F2 of CEC2017 is not used because it is an unstable problem.

The population size of all algorithms is set as 100, and the dimension of problems is 30 for all the 25 functions of CEC2005. The dimensions D are 10, 30, 50, and 100 for CEC2017, and the total number of function evaluations of each algorithm is set to $10000 * \mathrm{D}$. The parameters max Success, max Failures, adjust Success, and adjust Failures of the EMFO are set as 2.5, 2.5, 0.8, and 0.5, respectively. In addition, other parameters of compared algorithms follow their original studies.

In this simulation, the mean value and standard deviation of the function error value ($f(gbest) - f(X^*)$) are recorded for testing the performance of each algorithm, where gbest is the best solution found by the algorithm in a run, and X^* is the theoretical global optimum of the benchmark functions.

4.1. Comparison on Solution Accuracy. Table 1 shows the optimization results of the algorithm's solution accuracy. In each row of the table, the first part lists the mean value of 25 runs, and the second part is the standard deviation of the solutions. The P value and H value of the nonparametric statistical test with a significance level $\alpha = 0.05$ are presented in the third and fourth lines. The symbol "ǂ" is tagged in the back of the mean value yielded by the algorithm that is significantly worse than the EMFO. If the EMFO is worse than other algorithms, a "ξ" is added to the back of the mean

```
(1)  Begin
(2)    Initialize the parameters;
(3)    Initialize uniformly random population;
(4)    gen = 1;
(5)    while not meet termination condition
(6)      if mod(gen, 10) = = 0
(7)        Update the number of flames by using Equation (9) or Equation (10);
(8)      end if
(9)      Calculate the fitness values of moths;
(10)     if gen = 1
(11)       F = sort(M); OF = sort(OM);
(12)     else
(13)       F = sort(M_{t-1}, M_t); OF = sort(OM_{t-1}, OM_t);
(14)     end if
(15)     rc = rand (0,1);
(16)     for i = 1 to n do
(17)       Update the parameter r and t;
(18)       Update the moth's position by using equation (11);
(19)     end for
(20)     gen = gen + 1;
(21)   end while
(22) end
```

Algorithm 1: *Pseudocode of the EMFO.*

value of the corresponding algorithm. The symbol "∼" indicates that there is no significant difference between the EMFO and compared algorithm. The last row of the table presents a summary of a total number of "ǂ," "ξ," and "∼." In addition, the best results are bold to clearly show.

Table 1 clearly shows that the EMFO has obtained the best performance on 18 functions that consisted of 3 unimodal functions (F1, F3, and F4), 5 multimodal functions (F6, F9-F11, and F13), and 10 hybrid functions (F16-F25), which demonstrates that the EMFO is promising in solving unimodal functions. Meanwhile, the EMFO is worse than the AMFO on 2 functions (F5 and F7), which could be because the spiral search operator of the EMFO has poor search ability for the boundary neighborhood while the crisscross optimization method of the AMFO enhances the exploitation ability of the population to the boundary neighborhood. In addition, the MFO has gained the best performance on 4 functions (F2, F8, F12, and F24) and is better than the EMFO on 3 functions (F5, F8, and F12). The LMFO is best on 6 functions (F5, F7, F14, F15, F21, and F24). This may be because the multiple flame guidance mechanism decreases the search performance in solving the rotated problems. The GMFO and CMFO only have gained the best performance on function F24, and OMFO only is best on function F24.

In terms of the statistical results of Wilcoxon's rank-sum test, the EMFO is significantly better than the MFO, AMFO, GMFO, CMFO, LMFO, and OMFO on 14, 22, 16, 18, 15, and 16 out of 25 functions, and inferior to them on 3, 2, 3, 3, 5, and 2 out of 25 ones, respectively. Therefore, we can say that the EMFO produces the best results for these benchmark test functions.

Table 2 expresses the average ranks of the mean value of the function error value ($f(gbest) - f(X^*)$) with 51 runs, according to Friedman's test for the compared algorithms. The best ranks are shown in bold. Table 2 clearly shows that the EMFO is ranked first for all dimensions. Regarding mean ranking and summary rank, the EMFO has obtained the best performance, which demonstrates that the EMFO is promising and significate. In addition, the EMFO is better than the MFO-AFN and MFO-MFG for all dimensions using CEC2017, which indicates that there is a synergy between the adaptive flame number updating mechanism and the multiple flame guidance mechanism.

4.2. The Comparison Results of Convergence Speed. In Figure 6, the vertical axis is the nature logarithm of the mean value over independent 25 runs, and the horizontal axis is the sampling point where 30 sampling points are taken from FES = 1000 and mod (FES, 10000) = 0.

It can be clearly seen from Figure 6 that the EMFO obtains the best convergence speed on 10 functions (F4, F6, and F18-F25), which indicates that the adaptive flame number updating mechanism and the multiple flame guidance mechanism can improve the convergence speed of the canonical MFO algorithm. Although the EMFO gained poor convergence performance on 9 benchmark functions (F1-F3, F9-F11, F13, F16, and F17), it obtains strong global exploration ability and achieves the best convergence accuracy. This may be because the adaptive flame number updating mechanism enhances the ability of the population to jump out of the local optimal trap. The EMFO gains poor convergence speed on the other 6 functions (F5, F7, F8, F12, F14, and F15), which could be because the EMFO needs a longer time to jump out of the local optimum. In addition, the MFO outperforms the other 6 algorithms on 5 functions (F1-F3, F8, and F12), and the CMFO gains the best

Table 1: Solution accuracy of EMFO and compared algorithms.

Func.	Results	EMFO	MFO	AMFO	GMFO	CMFO	LMFO	OMFO
F1	Mean	1.29*E*-24	4.10*E*-22~	8.36*E*+03‡	6.32*E*-11‡	1.17*E*-10‡	1.11*E*-10‡	1.62*E*-10‡
	Std.	1.86*E*-24	2.04*E*-21	9.74*E*+02	1.00*E*-10	1.34*E*-10	1.25*E*-10	2.08*E*-10
	P value	—	3.21*E*-01	6.03*E*-40	2.78*E*-03	6.57*E*-05	5.26*E*-05	3.07*E*-04
	H value	—	0	1	1	1	1	1
F2	Mean	3.17*E*-02	5.32*E*-02~	1.94*E*+04‡	3.14*E*+01‡	3.44*E*+01‡	3.05*E*+01‡	2.98*E*+01‡
	Std.	3.15*E*-02	5.24*E*-02	2.88*E*+03	2.07*E*+01	2.06*E*+01	1.75*E*+01	3.49*E*+01
	P value	—	8.46*E*-02	5.21*E*-35	8.73*E*-10	6.22*E*-11	2.18*E*-11	9.14*E*-05
	H value	—	0	1	1	1	1	1
F3	Mean	1.51*E*+06	1.84*E*+06~	1.30*E*+08‡	4.35*E*+06‡	4.26*E*+06‡	4.10*E*+06‡	3.89*E*+06‡
	Std.	9.36*E*+05	7.07*E*+05	2.54*E*+07	1.48*E*+06	1.95*E*+06	2.27*E*+06	1.58*E*+06
	P value	—	1.70*E*-01	1.93*E*-29	1.56*E*-10	7.29*E*-08	3.35*E*-06	4.76*E*-08
	H value	—	0	1	1	1	1	1
F4	Mean	1.91*E*+00	1.16*E*+03‡	2.47*E*+04‡	3.46*E*+03‡	4.01*E*+03‡	2.99*E*+03‡	3.29*E*+03‡
	Std.	2.70*E*+00	1.03*E*+03	3.74*E*+03	1.93*E*+03	2.42*E*+03	1.52*E*+03	2.72*E*+03
	P value	—	8.18*E*-07	1.32*E*-34	8.00*E*-12	8.39*E*-11	4.37*E*-13	2.12*E*-07
	H value	—	1	1	1	1	1	1
F5	Mean	2.90*E*+04	2.73*E*+04ξ	2.80*E*+04ξ	2.75*E*+04ξ	2.29*E*+04ξ	2.29*E*+04ξ	2.71*E*+04ξ
	Std.	2.88*E*+01	2.24*E*+03	1.66*E*+03	2.63*E*+03	3.40*E*+02	4.55*E*+02	2.55*E*+03
	P value	—	3.47*E*-04	2.40*E*-03	6.67*E*-03	4.62*E*-55	4.76*E*-49	5.31*E*-04
	H value	—	1	1	1	1	1	1
F6	Mean	5.03*E*+01	1.38*E*+02‡	6.14*E*+08‡	2.31*E*+02‡	3.05*E*+02~	2.23*E*+02~	1.60*E*+02‡
	Std.	3.75*E*+01	2.13*E*+02	1.23*E*+08	3.71*E*+02	8.97*E*+02	5.32*E*+02	2.65*E*+02
	P value	—	4.92*E*-02	3.82*E*-29	1.93*E*-02	1.63*E*-01	1.12*E*-01	4.52*E*-02
	H value	—	1	1	1	0	0	1
F7	Mean	4.70*E*+03	4.70*E*+03~	4.61*E*+03ξ	4.69*E*+03~	3.92*E*+03ξ	3.90*E*+03ξ	4.65*E*+03~
	Std.	2.83*E*-12	2.42*E*-12	1.13*E*+02	3.61*E*+01	1.75*E*+02	1.95*E*+02	1.53*E*+02
	P value	—	1.00*E*+00	2.33*E*-04	3.22*E*-01	6.08*E*-27	1.94*E*-25	1.61*E*-01
	H value	—	0	1	0	1	1	0
F8	Mean	2.09*E*+01	2.02*E*+01ξ	2.10*E*+01~	2.09*E*+01~	2.09*E*+01~	2.09*E*+01~	2.09*E*+01~
	Std.	6.02*E*-02	7.48*E*-02	6.03*E*-02	4.58*E*-02	7.21*E*-02	4.95*E*-02	6.25*E*-02
	P value	—	1.02*E*-38	3.72*E*-01	7.15*E*-01	8.44*E*-01	6.48*E*-01	9.54*E*-01
	H value	—	1	0	0	0	0	0
F9	Mean	2.13*E*+01	4.92*E*+01‡	2.33*E*+02‡	5.10*E*+01‡	4.82*E*+01‡	5.32*E*+01‡	4.94*E*+01‡
	Std.	3.95*E*+00	1.16*E*+01	1.62*E*+01	1.59*E*+01	1.20*E*+01	1.87*E*+01	1.60*E*+01
	P value	—	2.83*E*-15	5.38*E*-48	5.37*E*-12	3.53*E*-14	6.58*E*-11	3.42*E*-11
	H value	—	1	1	1	1	1	1
F10	Mean	3.46*E*+01	1.13*E*+02‡	2.96*E*+02‡	1.23*E*+02‡	1.13*E*+02‡	1.26*E*+02‡	1.13*E*+02‡
	Std.	9.31*E*+00	2.89*E*+01	1.42*E*+01	3.59*E*+01	3.17*E*+01	3.10*E*+01	2.82*E*+01
	P value	—	3.50*E*-17	6.81*E*-52	7.04*E*-16	6.26*E*-16	9.43*E*-19	1.28*E*-17
	H value	—	1	1	1	1	1	1
F11	Mean	6.14*E*+00	2.64*E*+01‡	3.93*E*+01‡	2.61*E*+01‡	3.97*E*+01‡	2.75*E*+01‡	2.39*E*+01‡
	Std.	1.78*E*+00	4.25*E*+00	1.25*E*+00	4.33*E*+00	1.20*E*+00	5.33*E*+00	3.95*E*+00
	P value	—	1.17*E*-26	8.48*E*-52	3.52*E*-26	2.74*E*-52	5.63*E*-24	2.20*E*-25
	H value	—	1	1	1	1	1	1
F12	Mean	8.78*E*+05	7.81*E*+03ξ	1.02*E*+06‡	1.15*E*+04ξ	9.42*E*+05‡	3.15*E*+04ξ	9.38*E*+03ξ
	Std.	8.30*E*+04	5.31*E*+03	1.51*E*+05	7.89*E*+03	1.24*E*+05	6.18*E*+04	5.85*E*+03
	P value	—	5.52*E*-44	1.25*E*-04	7.60*E*-44	3.61*E*-02	5.78*E*-39	6.14*E*-44
	H value	—	1	1	1	1	1	1
F13	Mean	2.67*E*+00	7.29*E*+00‡	3.16*E*+01‡	7.06*E*+00‡	7.11*E*+00‡	7.17*E*+00‡	7.48*E*+00‡
	Std.	4.69*E*-01	2.23*E*+00	1.76*E*+00	2.59*E*+00	2.78*E*+00	2.20*E*+00	2.00*E*+00
	P value	—	1.54*E*-13	1.27*E*-52	6.73*E*-11	3.17*E*-10	2.34*E*-13	1.15*E*-15
	H value	—	1	1	1	1	1	1
F14	Mean	1.29*E*+01	1.33*E*+01‡	1.34*E*+01‡	1.28*E*+01ξ	1.33*E*+01‡	1.27*E*+01ξ	1.30*E*+01~
	Std.	2.28*E*-01	3.10*E*-01	1.45*E*-01	2.85*E*-01	1.85*E*-01	3.16*E*-01	2.49*E*-01
	P value	—	5.94*E*-06	1.54*E*-12	2.01*E*-02	8.44*E*-07	2.59*E*-02	1.03*E*-01
	H value	—	1	1	1	1	1	0

Table 1: Continued.

Func.	Results	EMFO	MFO	AMFO	GMFO	CMFO	LMFO	OMFO
F15	Mean	3.88*E*+02	3.61*E*+02~	5.70*E*+02‡	3.96*E*+02~	2.68*E*+02ξ	2.63*E*+02ξ	4.00*E*+02~
	Std.	7.84*E*+01	6.62*E*+01	2.95*E*+01	8.08*E*+01	7.77*E*+01	9.45*E*+01	8.73*E*+01
	P value	—	1.85*E*-01	1.77*E*-14	7.51*E*-01	1.79*E*-06	5.95*E*-06	6.25*E*-01
	H value	—	0	1	0	1	1	0
F16	Mean	6.02*E*+01	2.05*E*+02‡	3.29*E*+02‡	1.83*E*+02‡	1.66*E*+02‡	1.83*E*+02‡	1.59*E*+02‡
	Std.	2.48*E*+01	1.18*E*+02	1.42*E*+01	9.51*E*+01	8.01*E*+01	8.56*E*+01	6.62*E*+01
	P value	—	2.36*E*-07	7.77*E*-42	1.00*E*-07	7.85*E*-08	1.15*E*-08	7.20*E*-09
	H value	—	1	1	1	1	1	1
F17	Mean	6.83*E*+01	1.81*E*+02‡	3.64*E*+02‡	1.79*E*+02‡	1.73*E*+02‡	1.55*E*+02‡	1.49*E*+02‡
	Std.	2.79*E*+01	1.04*E*+02	2.18*E*+01	8.75*E*+01	8.97*E*+01	6.53*E*+01	7.23*E*+01
	P value	—	3.16*E*-06	2.29*E*-39	2.43*E*-07	1.08*E*-06	1.82*E*-07	3.79*E*-06
	H value	—	1	1	1	1	1	1
F18	Mean	9.04*E*+02	9.11*E*+02‡	1.01*E*+03‡	9.11*E*+02‡	9.13*E*+02‡	9.13*E*+02‡	9.11*E*+02‡
	Std.	1.04*E*+00	3.37*E*+00	8.57*E*+00	3.36*E*+00	4.18*E*+00	5.38*E*+00	3.04*E*+00
	P value	—	1.18*E*-13	1.99*E*-46	4.57*E*-12	2.29*E*-13	6.98*E*-10	2.79*E*-13
	H value	—	1	1	1	1	1	1
F19	Mean	9.04*E*+02	9.13*E*+02‡	1.00*E*+03‡	9.12*E*+02‡	9.10*E*+02‡	9.11*E*+02‡	9.13*E*+02‡
	Std.	8.55*E*-01	4.32*E*+00	1.07*E*+01	3.03*E*+00	3.27*E*+00	4.05*E*+00	3.32*E*+00
	P value	—	2.87*E*-13	2.02*E*-41	4.50*E*-16	2.41*E*-12	1.73*E*-10	2.52*E*-16
	H value	—	1	1	1	1	1	1
F20	Mean	9.04*E*+02	9.12*E*+02‡	1.00*E*+03‡	9.12*E*+02‡	9.12*E*+02‡	9.12*E*+02‡	9.12*E*+02‡
	Std.	9.78*E*-01	3.26*E*+00	9.76*E*+00	3.56*E*+00	5.70*E*+00	2.68*E*+00	3.27*E*+00
	P value	—	2.76*E*-15	1.34*E*-43	3.77*E*-14	1.31*E*-08	6.84*E*-19	2.52*E*-15
	H value	—	1	1	1	1	1	1
F21	Mean	5.00*E*+02	5.27*E*+02~	1.13*E*+03‡	5.79*E*+02~	5.48*E*+02‡	5.00*E*+02~	5.77*E*+02~
	Std.	8.29*E*-14	1.33*E*+02	3.30*E*+01	2.19*E*+02	1.12*E*+02	1.41*E*-10	1.95*E*+02
	P value	—	3.22*E*-01	1.94*E*-56	7.67*E*-02	3.76*E*-02	7.98*E*-02	5.39*E*-02
	H value	—	0	1	0	1	0	0
F22	Mean	8.76*E*+02	9.56*E*+02‡	1.07*E*+03‡	9.67*E*+02‡	9.60*E*+02‡	9.65*E*+02‡	9.46*E*+02‡
	Std.	1.87*E*+01	3.75*E*+01	1.69*E*+01	5.35*E*+01	4.56*E*+01	3.40*E*+01	3.50*E*+01
	P value	—	1.52*E*-12	1.95*E*-37	2.36*E*-10	3.58*E*-11	2.97*E*-15	1.28*E*-11
	H value	—	1	1	1	1	1	1
F23	Mean	5.34*E*+02	5.77*E*+02~	1.12*E*+03‡	5.84*E*+02~	5.77*E*+02~	5.77*E*+02~	5.50*E*+02~
	Std.	4.29*E*-04	1.49*E*+02	2.97*E*+01	1.34*E*+02	1.51*E*+02	1.51*E*+02	8.06*E*+01
	P value	—	1.57*E*-01	3.04*E*-57	6.91*E*-02	1.60*E*-01	1.64*E*-01	3.22*E*-01
	H value	—	0	1	0	0	0	0
F24	Mean	2.00*E*+02	2.00*E*+02~	1.12*E*+03‡	2.00*E*+02~	2.00*E*+02~	2.00*E*+02~	2.00*E*+02~
	Std.	6.21*E*-13	6.21*E*-13	1.99*E*+01	1.29*E*-10	8.46*E*-11	6.37*E*-11	1.56*E*-10
	P value	—	4.58*E*-21	8.07*E*-75	3.42*E*-03	8.09*E*-04	1.46*E*-05	6.54*E*-03
	H value	—	1	1	1	1	1	1
F25	Mean	9.75*E*+02	9.85*E*+02‡	1.27*E*+03‡	9.86*E*+02‡	9.86*E*+02‡	9.86*E*+02‡	9.87*E*+02‡
	Std.	4.74*E*+00	9.61*E*+00	1.60*E*+01	8.71*E*+00	7.81*E*+00	7.50*E*+00	1.05*E*+01
	P value	—	2.07*E*-05	1.38*E*-54	4.71*E*-07	2.94*E*-07	6.27*E*-08	1.64*E*-06
	H value	—	1	1	1	1	1	1
‡/ξ/~		—	14/3/8	22/2/1	16/3/6	18/3/4	15/5/5	16/2/7

performance on 2 functions (F5 and F7). The LMFO is best on 3 test functions (F5, F7, and F15).

In terms of convergence speed, the EMFO is significantly speeder than MFO, AMFO, GMFO, CMFO, LMFO, and OMFO on 15, 21, 20, 20, 18, and 21 out of 25 functions. Meanwhile, the EMFO is clearly slower than MFO, AMFO, GMFO, CMFO, LMFO, and OMFO on 7, 2, 3, 3, 5, and 2 out of 25 ones, respectively. Therefore, the results of convergence speed indicate that the EMFO generates a promising convergence speed for these benchmark test functions.

4.3. Comparison Results of Time Complexity. The total comparisons of the average time complexity of 25 functions in one iteration of about eight algorithms are shown in Figure 7 in the form of a bar plot. In Figure 7, it is clearly shown that the mean CPU time of the OMFO is the best and that of the CMFO is worst. The mean CPU time of the EMFO is the fourth and only 16.476 seconds longer than the OMFO. Although the time complexity of the EMFO is not the best, it can achieve acceptable convergence speed and promising convergence accuracy.

Table 2: Average ranks for all algorithms across 28 problems and all dimensions using CEC2017.

Algorithms	10D	30D	50D	100D	Mean ranking	Rank
EMFO	1.84	2.21	1.96	1.89	1.98	1
MFO	6	5.68	4.93	5.32	5.48	5
AMFO	9.48	10.71	10.93	11	10.53	11
CMFO	5.30	5.21	5.11	5.71	5.33	4
GMFO	4.75	5.14	5.36	5.25	5.13	3
LMFO	5.11	5.53	6.57	5.25	5.62	6
OMFO	6.29	6.14	6.04	5.79	6.07	7
EMFO	5.39	5.92	6.68	7.68	6.42	8
SaDN	10.61	10.11	9.61	9.43	9.94	10
MFO-AFN	7.86	6.71	6.79	6.36	6.93	9
MFO-MFG	3.39	2.60	2.04	2.32	2.59	2

5. The Application of Parameter Extraction of Photovoltaic Models

It is well known that solar energy is the most important renewable energy resource [45], and its main application is photovoltaic (PV) power generation [46], which is because solar energy can be directly converted into electricity by using the PV system. However, the conversion efficiency of the PV model is significantly affected by model parameters. To solve this problem, the EMFO is utilized to extract the parameters of the PV models.

5.1. PV Models and Fitness Functions. The single-diode and double-diode models [47] are the most widely used, which can explain the current-voltage features of the PV systems. The single-diode model, the diode model, the PV module, and the fitness function are addressed as follows.

5.1.1. Single-Diode Model. The output current and voltage of the single-diode model are calculated by using the following formula [48]:

$$I = I_{ph} - I_d\left[\exp\left(\frac{V + IR_s}{aV_t}\right) - 1\right] - \frac{V + IR_s}{R_{sh}}, \tag{12}$$

where I and V are the output current and the output voltage of the single-diode model, respectively. R_s and R_{sh} denote the series resistance and the shunt resistance, respectively, and a stands for the diode ideal factor. I_{ph} and I_d are the photo-generated current and the diode current, respectively. V_t represents the junction thermal voltage, generated by using the following formula:

$$V_t = \frac{k \cdot T}{q}, \tag{13}$$

where k denotes the Boltzmann constant ($1.3806503 \times 10^{-23} J/K$) and T stands for the temperature of junction in Kelvin. q represents the electron charge ($1.60217646 \times 10^{-19} C$).

The unknown parameters (R_s, R_{sh}, a, I_{ph}, and I_d) of the single-diode model will be extracted.

5.1.2. Double-Diode Model. The relationship between output current and voltage of the double-diode model is described in the following formula:

$$I = I_{ph} - I_{d1}\left[\exp\left(\frac{V + IR_s}{a_1V_t}\right) - 1\right] - I_{d2}\left[\exp\left(\frac{V + IR_s}{a_2V_t}\right) - 1\right] - \frac{V + IR_s}{R_{sh}}, \tag{14}$$

where $I_{di}, i \in \{1, 2\}$ denotes the current i th diode, and $a_i, i \in \{1, 2\}$ represents the i th diode ideal factor. I and V are the output current and the output voltage.

The unknown parameters (R_s, R_{sh}, a_1, a_2 , I_{ph}, I_{d1}, and I_{d2}) of the single-diode model will be extracted.

5.1.3. PV Module. The output current I and voltage V of the PV module are described in the following formula:

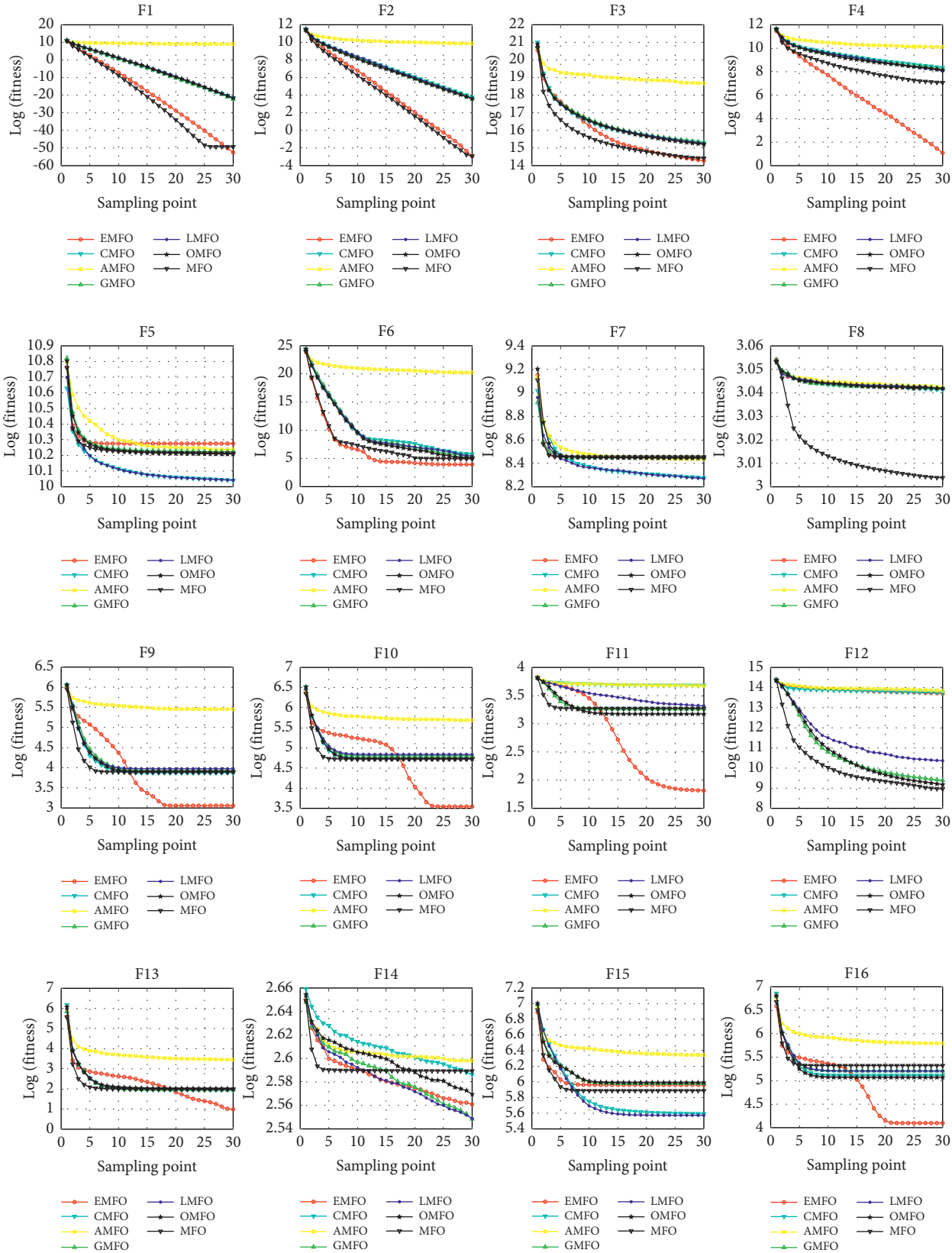
$$I = I_{ph}N_p - I_dN_p\left[\exp\left(\frac{V + IR_sN_s/N_p}{aN_sV_t}\right) - 1\right] - \frac{V + IR_sN_s/N_p}{R_{sh}N_s/N_p}, \tag{15}$$

where N_s denotes the number of solar cells that are connected in series, and N_p is the number of solar cells that are connected in parallel. In the simulation, N_p is set to 1, because the used PV modules are all in series. Therefore, the output current I and voltage V of the used PV module are represented in the following formula:

$$I = I_{ph} - I_dN_p\left[\exp\left(\frac{V + IR_sN_s}{aN_sV_t}\right) - 1\right] - \frac{V + IR_sN_s}{R_{sh}N_s}. \tag{16}$$

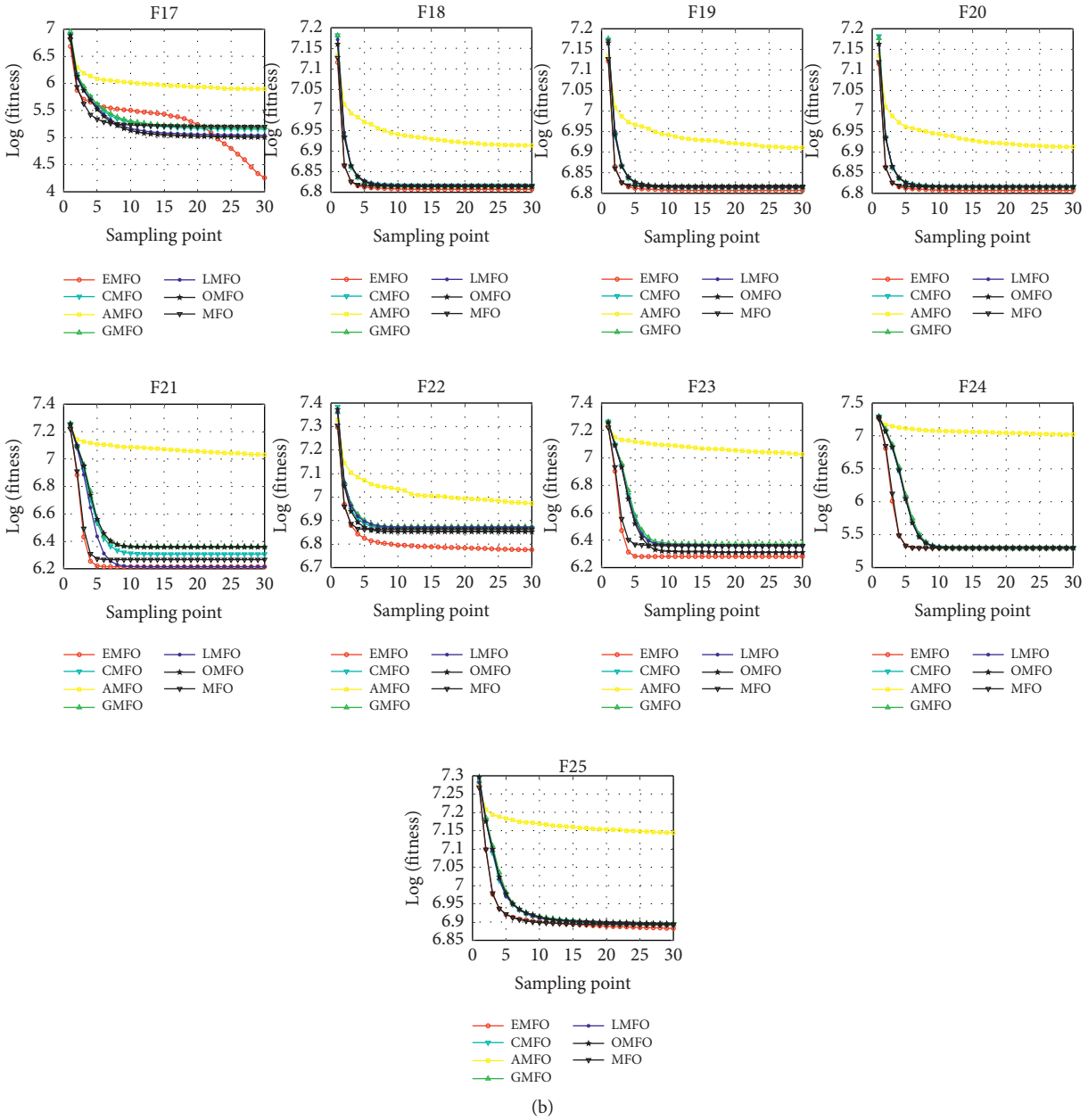
The unknown parameters (R_s, R_{sh}, a, I_{ph}, and I_d) of the PV module will be extracted.

5.1.4. Fitness Functions. The purpose of parameter extraction of the PV models is to minimize the error between simulated and measured current data I. The absolute error current (AEC) of the individuals is calculated as follows [49].



(a)

Figure 6: Continued.



(b)

Figure 6: Convergence performance of the seven compared algorithms on 25 functions of CEC 2005.

The AEC of the single-diode model is generated by the following formula:

$$AEC = \left| I_{ph} - I_d \left[\exp\left(\frac{V + IR_s}{aV_t} \right) - 1 \right] - \frac{V + IR_s}{R_{sh}} - I \right|. \quad (17)$$

The AEC of the double-diode model is calculated by the following formula:

$$AEC = \left| I_{ph} - I_{d1} \left[\exp\left(\frac{V + IR_s}{a_1 V_t} \right) - 1 \right] - I_{d2} \left[\exp\left(\frac{V + IR_s}{a_2 V_t} \right) - 1 \right] - \frac{V + IR_s}{R_{sh}} - I \right|. \quad (18)$$

The AEC of the PV module is described in the following formula:

$$AEC = \left| I_{ph} - I_d N_p \left[\exp\left(\frac{V + IR_s N_s}{a N_s V_t} \right) - 1 \right] - \frac{V + IR_s N_s}{R_{sh} N_s} - I \right|. \quad (19)$$

To quantify the overall error between the simulated and measured current, the root mean square error is utilized as the fitness function, which is represented as the following formula:

$$f(x) = \sqrt{1/N \sum_{i=1}^{N} AEC^2}, \quad (20)$$

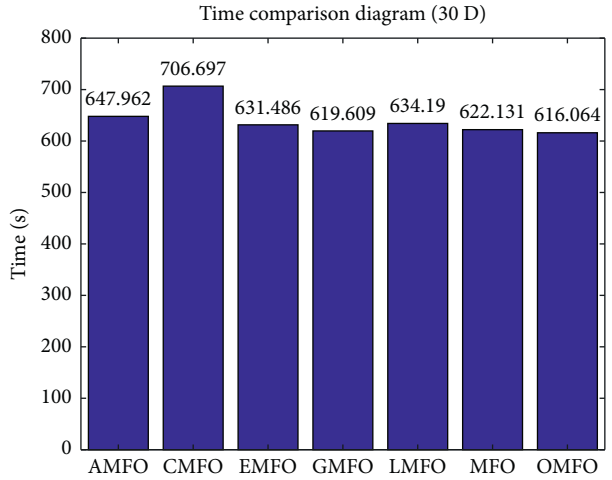


Figure 7: Mean CPU time of the EMFO and compared algorithms.

Table 3: The feasible range of parameters to be extracted.

Parameters	PV-F1/F2		PV-F3		PV-F4		PV-F5	
	LB	UB	LB	UB	LB	UB	LB	UB
I_{ph}	0	1	0	2	0	2	0	8
I_d, I_{d1}, I_{d2}	0	1	0	50	0	50	0	50
R_s	0	0.5	0	2	0	0.36	0	0.36
R_{sh}	0	100	0	2000	0	1000	0	1500
a, a_1, a_2	1	2	1	50	1	60	1	50

Table 4: Solution accuracy of the EMFO and compared algorithms on the PV problems.

Func.	Results	EMFO	MFO	AMFO	GMFO	CMFO	LMFO	OMFO
PV-F1	Mean	1.28*E*-03	1.74*E*-03‡	3.11*E*-03‡	1.74*E*-03‡	1.94*E*-03‡	2.06*E*-03‡	1.59*E*-03‡
	Std.	2.50*E*-04	3.73*E*-04	5.82*E*-04	4.83*E*-04	3.70*E*-04	4.05*E*-04	3.76*E*-04
	P value		5.87*E*-06	4.55*E*-19	9.74*E*-05	2.05*E*-09	9.41*E*-11	1.02*E*-03
	H value		1	1	1	1	1	1
PV-F2	Mean	1.27*E*-03	1.81*E*-03‡	6.99*E*-03‡	1.50*E*-03‡	2.28*E*-03‡	1.83*E*-03‡	1.44*E*-03‡
	Std.	2.21*E*-04	4.19*E*-04	2.27*E*-03	3.71*E*-04	3.92*E*-04	8.30*E*-04	2.16*E*-04
	P value		7.34*E*-07	9.23*E*-17	1.36*E*-02	5.57*E*-15	2.06*E*-03	9.98*E*-03
	H value		1	1	1	1	1	1
PV-F3	Mean	2.44*E*-03	2.51*E*-03‡	2.60*E*-03~	2.65*E*-03‡	1.62*E*-03ξ	2.43*E*-03~	2.56*E*-03‡
	Std.	1.71*E*-05	1.07*E*-04	1.46*E*-03	3.38*E*-04	3.82*E*-05	1.18*E*-03	2.02*E*-04
	P value		1.79*E*-03	5.86*E*-01	3.27*E*-03	5.56*E*-57	9.59*E*-01	6.34*E*-03
	H value		1	0	1	1	0	1
PV-F4	Mean	3.24*E*-03	5.64*E*-03‡	2.23*E*-04ξ	3.32*E*-03~	1.73*E*-04ξ	7.50*E*-04ξ	3.69*E*-03‡
	Std.	6.34*E*-04	2.30*E*-03	2.57*E*-05	4.88*E*-04	2.83*E*-05	1.59*E*-03	5.59*E*-04
	P value		7.30*E*-06	3.44*E*-28	6.05*E*-01	1.67*E*-28	2.94*E*-09	1.03*E*-02
	H value		1	1	0	1	1	1
PV-F5	Mean	3.38*E*-02	3.97*E*-02‡	2.39*E*-04ξ	3.14*E*-02~	1.91*E*-02~	2.95*E*-02~	3.83*E*-02‡
	Std.	5.46*E*-03	1.30*E*-02	1.19*E*-04	1.33*E*-02	9.48*E*-02	9.25*E*-02	5.06*E*-03
	P value	-	4.17*E*-02	3.41*E*-33	4.13*E*-01	4.43*E*-01	8.18*E*-01	3.88*E*-03
	H value	-	1	1	0	0	0	1
‡/ξ/~		-	5/0/0	2/2/1	3/0/2	2/2/1	2/1/2	5/0/0

where x denotes the feasible solution consisted of the unknown parameters to be extracted, and N indicates the number of measured current data. Furthermore, $f(x)$ is the fitness value of feasible solution x. It is clearly shown that the smaller the fitness value, the more accurate the extracted parameters.

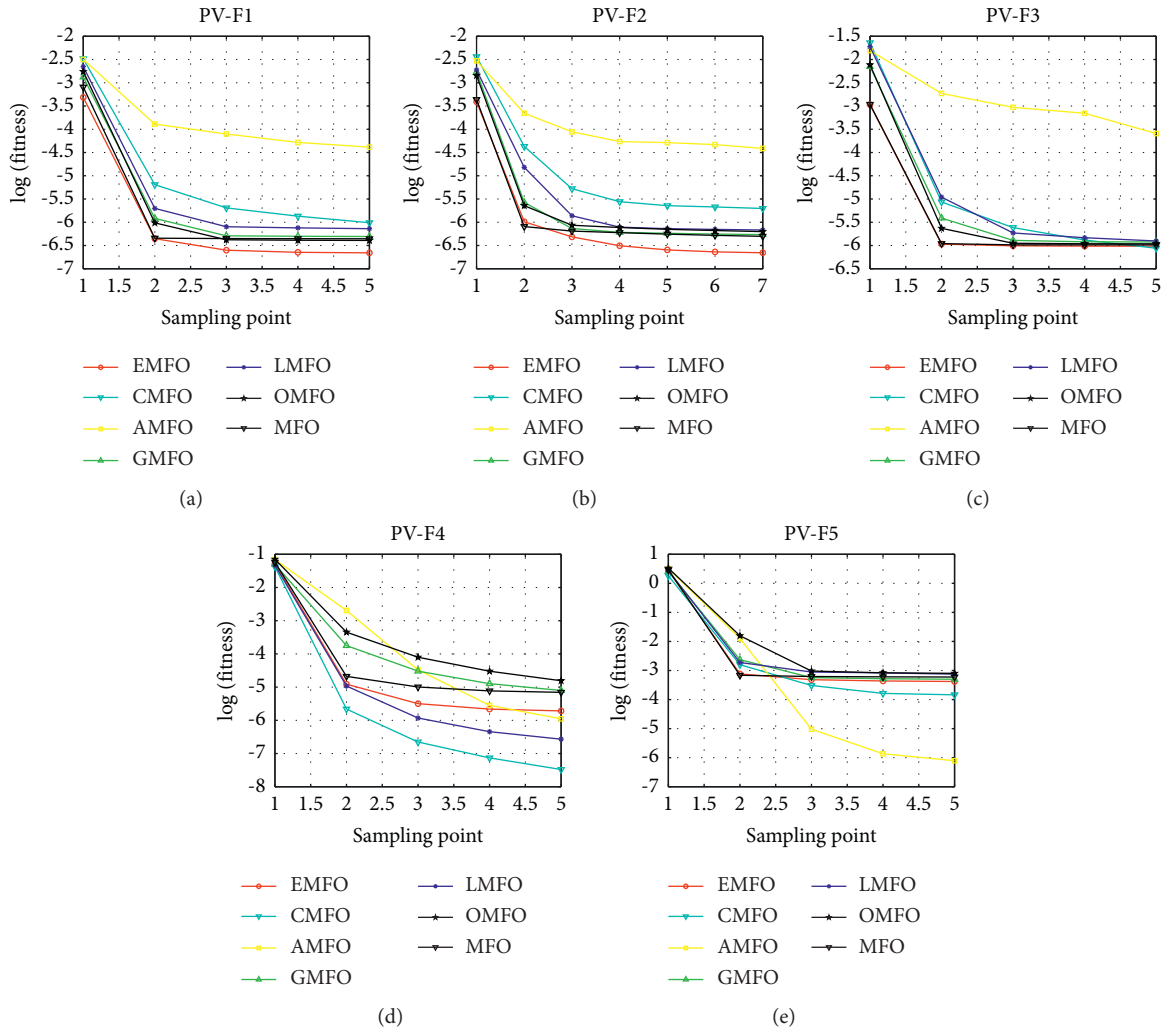


Figure 8: Convergence performance of the seven compared algorithms on the 5 PV models. (a) (PV-F1). (b) (PV-F2). (c) (PV-F3). (d) (PV-F4). (e) (PV-F5).

5.2. Parameter Setting. The current-voltage data of single- (PV-F1) and double (PV-F2)-diode models are gained from [50], measured on a 57 mm diameter commercial silicon R.T.C. France solar cell under 1000 W/m^2 at 33 ℃. Three different PV modules are utilized, i.e., polycrystalline Photowatt-PWP201 (PV-F3), monocrystalline STM6-40/36 (PV-F4), and polycrystalline STP6-120/36 (PV-F5). The polycrystalline Photowatt-PWP201 is measured under 1000W/m^2 at 45°C [50]. The monocrystalline STM6-40/36 and the polycrystalline STP6-120/36 are measured under 51°C and 55°C, respectively, whose current-voltage data are gained from [51, 52]. The feasible range of parameters to be extracted is shown in Table 3. In addition, the parameter setting of the simulation is consistent with that in Section 4.

5.3. Comparison of Solution Accuracy. The basic settings of Table 4 are the same as Table 1. From Table 4, it can be clearly seen that the EMFO outperforms 6 compared algorithms on 2 functions (PV-F1 and PV-F2) and provides the second-best result on 2 functions (PV-F3 and PV-F5). Additionally, the EMFO obtains the fourth-best result on function PV-F4. The simulation results demonstrate that the EMFO is effective to accurately extract parameters of the PV models.

5.4. The Comparison Results of Convergence Speed. In Figure 8, the vertical axis is the nature logarithm of the mean value over independent 25 runs, and the horizontal axis is the sampling point where sampling points are taken from FES = 1000 and mod (FES, 10000) = 0.

It can be clearly observed from Figure 8 that the EMFO obtains the best convergence performance on 3 functions (PV-F1–PV-F3) and gains the third-best convergence speed on 2 functions (PV-F4 and PV-F5). The results described above indicate that the EMFO is promising and significant in solving the parameters' extract problems of the PV models.

6. Conclusions

To enhance the exploration ability and avoid to fall into the local trap, an enhanced MFO with multiple flame guidance

mechanism is proposed in this study, named EMFO for short. In EMFO, an adaptive flame number updating mechanism is utilized to adaptively adjust flame's number for helping the moth population escape the local optimal trap. Besides, a multiple flame guidance mechanism is designed to fully use the position information of flames, which enhances the diversity of flame guidance information and avoids premature. To verify the performance of the EMFO, it is used to optimize 25 benchmark functions of the CEC 2005, 28 test functions of CEC2017, and a real-world optimization problem, compared with the state-of-the-art algorithms. The results show that the EMFO has obtained promising performance, and gained higher convergence accuracy and faster convergence speed than the compared algorithms.

In future work, the EMFO will be further tested on newer, more complex test functions and more complex application problems to verify its robustness.

Data Availability

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

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Authors' Contributions

Zhenyu Wang and Zijian Cao conceptualized the study; Zijian Cao developed methodology; Chen Liu and Haowen Jia investigated the study; Zhenyu Wang and Chen Liu prepared the original draft; Zijian Cao reviewed and edited the manuscript; Zijian Cao and Fuxi Liu contributed to funding acquisition. All authors have read and agreed to the published version of the manuscript. In particular, Feng Tian worked to publish the whole paper.

Acknowledgments

This research was partially funded by the Shaanxi Natural Science Basic Research Project (grant no. 2020JM-565).

References

[1] N. Panagant, N. Pholdee, and S. Bureerat, "A comparative study of recent multi-objective metaheuristics for solving constrained truss optimisation problems," *Archives of Computational Methods in Engineering*, vol. 28, no. 5, pp. 4031–4047, 2021.

[2] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, The MIT press, London, 1975.

[3] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[4] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," in *Proceedings of the First European Conference on Artificial Life*, vol. 142, pp. 134–142, 1991.

[5] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, vol. 4, pp. 39–43, Nagoya, Japan, October 1995.

[6] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[7] W. Li, G. G. Wang, and A. H. Gandomi, "A survey of learning-based intelligent optimization algorithms," *Archives of Computational Methods in Engineering*, pp. 1–19, 2021.

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

[9] Y. Feng, S. Deb, G. G. Wang, and A. H Alavi, "Monarch butterfly optimization: a comprehensive review," *Expert Systems with Applications*, vol. 168, Article ID 114418, 2021.

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

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

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

[13] G. G. Wang, S. Deb, and L. D. S. Coelho, "Earthworm optimisation algorithm: a bio-inspired metaheuristic algorithm for global optimisation problems," *International Journal of Bio-Inspired Computation*, vol. 12, no. 1, pp. 1–22, 2018.

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

[15] G. G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Computing*, vol. 10, no. 2, pp. 151–164, 2018.

[16] R. Salgotra and U. Singh, "The naked mole-rat algorithm," *Neural Computing & Applications*, vol. 31, no. 12, pp. 8837–8857, 2019.

[17] D. Oliva, M. Abd El Aziz, and A. E. Hassanien, "Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm," *Applied Energy*, vol. 200, pp. 141–154, 2017.

[18] B. S. Yıldız, "Optimal design of automobile structures using moth-flame optimization algorithm and response surface methodology," *Materials Testing*, vol. 62, no. 4, pp. 371–377, 2020.

[19] B. S. Yıldız, V. Patel, N. Pholdee, B Sujin, S Sadiq, and R. Y Ali, "Conceptual comparison of the ecogeography-based algorithm, equilibrium algorithm, marine predators algorithm and slime mold algorithm for optimal product design," *Materials Testing*, vol. 63, no. 4, pp. 336–340, 2021.

[20] B. S. Yıldız, N. Pholdee, S. Bureerat, U. E Mehmet, R. Y Ali, and S Sadiq, "Comparision of the political optimization algorithm, the Archimedes optimization algorithm and the Levy flight algorithm for design optimization in industry," *Materials Testing*, vol. 63, no. 4, pp. 356–359, 2021.

[21] A. R. Yıldız, H. Özkaya, M. Yıldız, B Sujin, Y Betül, and S Sadiq, "The equilibrium optimization algorithm and the response surface-based metamodel for optimal structural

design of vehicle components," *Materials Testing*, vol. 62, no. 5, pp. 492–496, 2020.

[22] A. B. S. Yıldız, N. Pholdee, S. Bureerat, and S Sadiq, "Sine-cosine optimization algorithm for the conceptual design of automobile components," *Materials Testing*, vol. 62, no. 7, pp. 744–748, 2020.

[23] N. Panagant, N. Pholdee, S. Bureerat, K Khon, R. Y Ali, and M. S Sadiq, "Seagull optimization algorithm for solving real-world design optimization problems," *Materials Testing*, vol. 62, no. 6, pp. 640–644, 2020.

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

[25] B. Bentouati, L. Chaib, and S. Chettih, "Optimal power flow using the moth flam optimizer: a case study of the Algerian power system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 1, no. 3, pp. 431–445, 2016.

[26] S. Sapre and S. Mini, "Opposition-based moth flame optimization with Cauchy mutation and evolutionary boundary constraint handling for global optimization," *Soft Computing*, vol. 23, no. 15, pp. 6023–6041, 2019.

[27] X. Zhao, Y. Fang, Z. Ma, and M Xu, "An Ameliorated moth-flame optimization algorithm," in *Proceedings of the 2018 37th Chinese Control Conference (CCC)*, pp. 2372–2377, Wuhan, China, July 2018.

[28] Y. Xu, H. Chen, J. Luo, Z Qian, J Shan, and Z Xiaoqin, "Enhanced Moth-flame optimizer with mutation strategy for global optimization," *Information Sciences*, vol. 492, pp. 181–203, 2019.

[29] Y. Li, Z. Wang, Y. Cheng, Y Tang, and Z Shang, "A hybrid improved moth-flame optimization with differential evolution with global and local neighborhoods algorithm for pose optimization on a space manipulator," *Measurement Science and Technology*, vol. 30, no. 12, Article ID 125012, 2019.

[30] E. Emary and H. M. Zawbaa, "Impact of chaos functions on modern swarm optimizers," *PLoS One*, vol. 11, no. 7, Article ID e0158738, 2016.

[31] M. Wang, H. Chen, B. Yang et al., "Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses," *Neurocomputing*, vol. 267, pp. 69–84, 2017.

[32] U. Guvenc, S. Duman, and Y. Hınıslıoglu, "Chaotic moth swarm algorithm," in *Proceedings of the 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pp. 90–95, Gdynia, Poland, July 2017.

[33] Z. Li, Y. Zhou, S. Zhang, and J Song, "Lévy-flight moth-flame algorithm for function optimization and engineering design problems," *Mathematical Problems in Engineering*, vol. 2016, Article ID 1423930, 22 pages, 2016.

[34] G. I. Sayed and A. E. Hassanien, "A hybrid SA-MFO algorithm for function optimization and engineering design problems," *Complex & Intelligent Systems*, vol. 4, no. 3, pp. 195–212.

[35] R. H. Bhesdadiya, I. N. Trivedi, P. Jangir, A Kumar, N Jangir, and R Totlani, "A novel hybrid approach particle swarm optimizer with moth-flame optimizer algorithm," *Advances in Computer and Computational Sciences*, Springer, Singapore, pp. 569–577, 2017.

[36] M. Anfal and H. Abdelhafid, "Optimal placement of PMUs in Algerian network using a hybrid particle swarm–moth flame optimizer (PSO-MFO)," Electrotehnica, Electronica," *Automatica*, vol. 65, no. 3, 2017.

[37] P. Jangir, "Optimal power flow using a hybrid particle Swarm optimizer with moth flame optimizer," *Global Journal of Researches in Engineering*, vol. 17, no. 5, pp. 15–32, 2017.

[38] M. H. Nadimi-Shahraki, A. Fatahi, H. Zamani, S Mirjalili, L Abualigah, and M. E. A Elaziz, "Migration-based moth-flame optimization algorithm," *Processes*, vol. 9, no. 12, p. 2276, 2021.

[39] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, A. A. E Ewees, L Abualigah, and M. E. A Elaziz, "MTV-MFO: multi-trial vector-based moth-flame optimization algorithm," *Symmetry*, vol. 13, no. 12, p. 2388, 2021.

[40] T. K. Truong, "A new moth-flame optimization algorithm for discounted {0-1} knapsack problem," *Mathematical Problems in Engineering*, vol. 2021, Article ID 5092480, 15 pages, 2021.

[41] P. N. Suganthan, N. Hansen, J. J. Liang et al., "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," *Kangal report*, vol. 2005005, p. 2005, 2005.

[42] K. Kaur, U. Singh, and R. Salgotra, "An enhanced moth flame optimization," *Neural Computing & Applications*, vol. 32, no. 7, pp. 2315–2349, 2020.

[43] R. Salgotra, U. Singh, G. Singh, M Nitin, and H. G Amir, "A self-adaptive hybridized differential evolution naked mole-rat algorithm for engineering optimization problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 383, Article ID 113916, 2021.

[44] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization,"Technical Report 2017, Singapore.

[45] S. Xu and Y. Wang, "Parameter estimation of photovoltaic modules using a hybrid flower pollination algorithm," *Energy Conversion and Management*, vol. 144, pp. 53–68, 2017.

[46] W. C. Yeh, C. L. Huang, P. Lin, Z Chen, Y Jiang, and B Sun, "Simplex simplified swarm optimization for the efficient optimization of parameter identification for solar cell models," *IET Renewable Power Generation*, vol. 12, no. 1, pp. 45–51, 2018.

[47] A. Askarzadeh and A. Rezazadeh, "Parameter identification for solar cell models using harmony search-based algorithms," *Solar Energy*, vol. 86, no. 11, pp. 3241–3249, 2012.

[48] M. R. AlRashidi, M. F. AlHajri, K. M. El-Naggar, and A. K Al-Othman, "A new estimation approach for determining the I–V characteristics of solar cells," *Solar Energy*, vol. 85, no. 7, pp. 1543–1550, 2011.

[49] S. Li, W. Gong, X. Yan et al., "Parameter extraction of photovoltaic models using an improved teaching-learning-based optimization," *Energy Conversion and Management*, vol. 186, pp. 293–305, 2019.

[50] T. Easwarakhanthan, J. Bottin, I. Bouhouch, and C Boutrit, "Nonlinear minimization algorithm for determining the solar cell parameters with microcomputers," *International Journal of Solar Energy*, vol. 4, no. 1, pp. 1–12, 1986.

[51] T. Tong and W. Pora, "A parameter extraction technique exploiting intrinsic properties of solar cells," *Applied Energy*, vol. 176, pp. 104–115, 2016.

[52] X. Gao, Y. Cui, J. Hu et al., "Parameter extraction of solar cell models using improved shuffled complex evolution algorithm," *Energy Conversion and Management*, vol. 157, pp. 460–479, 2018.