mathematics

Article

MFO-SFR: An Enhanced Moth-Flame Optimization Algorithm Using an Effective Stagnation Finding and Replacing Strategy

Mohammad H. Nadimi-Shahraki [1,2,*], Hoda Zamani [1,2], Ali Fatahi [1,2] and Seyedali Mirjalili [3,4,*]

[1] Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad 8514143131, Iran
[2] Big Data Research Center, Najafabad Branch, Islamic Azad University, Najafabad 8514143131, Iran
[3] Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Brisbane 4006, Australia
[4] Yonsei Frontier Lab, Yonsei University, Seoul 03722, Republic of Korea
* Correspondence: nadimi@iaun.ac.ir (M.H.N.-S.); ali.mirjalili@torrens.edu.au (S.M.)

Abstract: Moth-flame optimization (MFO) is a prominent problem solver with a simple structure that is widely used to solve different optimization problems. However, MFO and its variants inherently suffer from poor population diversity, leading to premature convergence to local optima and losses in the quality of its solutions. To overcome these limitations, an enhanced moth-flame optimization algorithm named MFO-SFR was developed to solve global optimization problems. The MFO-SFR algorithm introduces an effective stagnation finding and replacing (SFR) strategy to effectively maintain population diversity throughout the optimization process. The SFR strategy can find stagnant solutions using a distance-based technique and replaces them with a selected solution from the archive constructed from the previous solutions. The effectiveness of the proposed MFO-SFR algorithm was extensively assessed in 30 and 50 dimensions using the CEC 2018 benchmark functions, which simulated unimodal, multimodal, hybrid, and composition problems. Then, the obtained results were compared with two sets of competitors. In the first comparative set, the MFO algorithm and its well-known variants, specifically LMFO, WCMFO, CMFO, ODSFMFO, SMFO, and WMFO, were considered. Five state-of-the-art metaheuristic algorithms, including PSO, KH, GWO, CSA, and HOA, were considered in the second comparative set. The results were then statistically analyzed through the Friedman test. Ultimately, the capacity of the proposed algorithm to solve mechanical engineering problems was evaluated with two problems from the latest CEC 2020 test-suite. The experimental results and statistical analysis confirmed that the proposed MFO-SFR algorithm was superior to the MFO variants and state-of-the-art metaheuristic algorithms for solving complex global optimization problems, with 91.38% effectiveness.

Keywords: global optimization problems; metaheuristic algorithms; moth-flame optimization; premature convergence; population diversity

MSC: 68T20

check for updates

Citation: Nadimi-Shahraki, M.H.; Zamani, H.; Fatahi, A.; Mirjalili, S. MFO-SFR: An Enhanced Moth-Flame Optimization Algorithm Using an Effective Stagnation Finding and Replacing Strategy. *Mathematics* **2023**, *11*, 862. https://doi.org/10.3390/math11040862

Academic Editor: Jian Dong

Received: 2 December 2022
Revised: 22 January 2023
Accepted: 3 February 2023
Published: 8 February 2023

1. Introduction

Global optimization problems are complex and characterized by various properties, for instance, they can be non-linear, non-separable, symmetric, asymmetrical, smooth with narrow ridges, unimodal, and multimodal, and can involve non-differentiable functions and high dimensionality [1,2]. These properties create challenges for existing optimization algorithms, and finding the global optimum is one of the long-standing goals in this area of study. To overcome such challenges, a series of metaheuristic algorithms have been introduced using various innovative approaches. Metaheuristic algorithms have exhibited impressive performance in exploring the problem space and approximating the promising regions in reasonable timeframes. They have been widely improved upon and adapted to solve optimization problems in diverse fields such as computer science [3,4], engineering [5,6], and medicine [7–9]. Metaheuristic algorithms can be classified into two groups: single-solution-based and population-based algorithms [10,11]. Single-solution-based metaheuristic algorithms are more oriented towards exploitation searches and they manipulate a single solution during the optimization process, which increases its potential to easily become stuck in local optima [12]. To solve this challenge, population-based metaheuristic algorithms were developed to be more exploration-oriented and to share the information in order to promote significant diversification in the search space [13,14]. Based on the source of inspiration, these algorithms can be classified as evolutionary-based, physics-based, human-based, and swarm intelligence-based algorithms [15,16].

Evolutionary-based algorithms involve a heuristic approach inspired by the biological evolution of species, such as animals, insects, and plants in nature [17,18]. Some prominent optimizers in this group are genetic algorithms [19], differential evolution [20], and the evolution strategy [21]. Physics-based algorithms are defined based on the main concepts of mathematics and physics, such as quantum physics [22–24], gravity [25,26], and optics [27], with the aim of performing a meaningful search in the problem space. Human-based algorithms simulate various human activities in order to generate innovative solutions in solving optimization problems. The imperialist competitive algorithm [28], the harmony search algorithm [29], teaching learning-based optimization [30], brain storm optimization (BSO) [31], the soccer league competition algorithm [32], the volleyball premier league algorithm [33], poor and rich optimization (PRO) [34], and past present future (PPF) [35] are some of the state-of-the-art optimizers in this group. Swarm intelligence-based optimization algorithms originated from the collective and self-organized behavior of unsophisticated agents such as insects, terrestrial, fish, and birds [36,37]. Ant colony optimization [38] and particle swarm optimization [39] were the most successful swarm intelligence-based optimization algorithms proposed in the 1990s. From the 21st century onwards, some new algorithms have been put forward in this group, such as artificial bee colony (ABC) [40], cuckoo search (CS) [41], the whale optimization algorithm (WOA) [42], elephant herding optimization (EHO) [43], moth-flame optimization (MFO) [44], the horse herd optimization algorithm (HOA) [45], the quantum-based avian navigation optimizer algorithm (QANA) [46], the African vultures optimization algorithm [47], farmland fertility [48], dwarf mongoose optimization (DMO) [49], the starling murmuration optimizer (SMO) [50], and the artificial gorilla troops optimizer [51].

Most population-based metaheuristic algorithms lack mechanisms that can maintain population diversity and the imbalance between search strategies and premature convergence problems. Hence, many effective mechanisms have been proposed to alleviate the weaknesses of these algorithms [52,53]. The artificial bee colony algorithm (ABC) is a prominent population-based metaheuristic algorithm that suffers from poor local search performance. Hence, Zhu et al. [54] proposed the Gbest-guided ABC (GABC) algorithm to incorporate information on the global best solution into the search strategy in order to improve the ability to exploit the algorithm. Other algorithms that have achieved significant performance improvements in terms of their local search ability are the quick artificial bee colony (qABC), best-so-far ABC [55], and grey artificial bee colony (GABC) algorithms [56]. Nadimi-Shahraki et al. [57] introduced a diversity-maintained multi-trial vector-differential evolution algorithm to increase population diversity and suspend the risk of premature convergence during the evolutionary process.

The moth-flame optimization (MFO) algorithm was inspired by the navigation behavior of moths toward a light source in nature and is used to solve global optimization problems. The MFO algorithm benefits from having a straightforward structure and a small number of control parameters, which increases its versatility. However, the MFO algorithm suffers from problems related to low population diversity [58], which leads it to become stuck in unpromising regions and to achieve low-quality solutions. Many MFO variants have been developed by introducing and hybridizing different search strategies and operators to overcome such challenges. Kaur et al. [59] proposed an enhanced moth flame optimization (E-MFO) method to solve global optimization problems. The E-MFO algorithm applied a Cauchy distribution function and the influence of the best flame parameter to enhance its exploration and exploitation capabilities, respectively. Moreover, an adaptive step size and division of iterations were proposed to balance search strategies. Li et al. [60] presented the Lévy-flight moth-flame optimization (LMFO) algorithm to prevent premature convergence into local optima and enable a trade-off between the algorithm's exploration and exploitation abilities during the search process. Khalilpourazari et al. [61] introduced the WCMFO algorithm, which is a hybridized form of two algorithms, the water cycle and moth-flame optimization algorithms, to increase the exploitation ability of MFO and the exploration ability of the water cycle algorithm. To cope with the weaknesses of MFO, Hongwei et al. [62] proposed chaos-enhanced moth-flame optimization (CMFO) using ten chaotic maps. The chaotic maps are applied in population initialization, boundary handling, and the tuning of the distance parameter. Other variants of MFO are sine-cosine moth-flame optimization (SMFO) [63], combining MFO with Gaussian, Cauchy, and Lévy mutations (LGCMFO) [64], the enhancement of the local search mechanism based on shuffled frog leaping and a death mechanism with MFO (ODSFMFO) [65], and the chaotic local search and Gaussian mutation-enhanced MFO (CLSGMFO) approach [66].

Although the mentioned MFO variants have attained effective modifications in performance, they may still suffer from poor population diversity, which leads to premature convergence to local optima and a decrease in the quality of the algorithms' solutions when tackling complex optimization problems. Moreover, due to the approximate nature of metaheuristic algorithms, there is always an opportunity for improvement in their search strategies. Therefore, this study was devoted to proposing an enhanced moth-flame optimization algorithm named MFO-SFR with the aim of solving global optimization problems. The proposed MFO-SFR algorithm is equipped with an effective stagnation finding and replacing (SFR) strategy to establish diversity throughout the search process and overcome the drawbacks of previous MFO approaches. Moreover, the boundary handling of the MFO algorithm is rectified by generating new random solutions in the range of the problem space. Overall, the main contributions of this study can be summarized as follows.

- We propose the MFO-SFR algorithm, boosting the performance and enriching the diversity of the canonical MFO;
- We introduce an effective stagnation finding and replacing (SFR) strategy to boost the performance of the search process; and
- We introduce an archive to incorporate the representative and the global best flames throughout the search process in order to enrich the diversity.

The performance of the proposed MFO-SFR algorithm was assessed with the CEC 2018 test functions [67] in 30 and 50 dimensions. Then, the MFO-SFR algorithm was compared with two sets of MFO variants and well-known optimizers. In the first set of contender algorithms, the canonical MFO [44] and its variants— Lévy-flight moth-flame optimization LMFO [60], an efficient hybrid algorithm based on the water cycle and moth-flame algorithms (WCMFO) [61], chaos-enhanced moth-flame optimization (CMFO) [62], death mechanism-based moth–flame optimization (ODSFMFO) [65], the synthesis of the moth-flame optimizer with sine cosine mechanisms (SMFO) [63], and the hybrid of whale and moth-flame optimization (WMFO) [68]—were selected. In the second set, the particle swarm optimization (PSO) [39], krill herd (KH) [69], grey wolf optimization (GWO) [70], the crow search algorithm (CSA) [71], and the horse herd optimization algorithm (HOA) [45] were considered. Furthermore, the results obtained using the proposed and contender algorithms were statistically analyzed using the Friedman test. Ultimately, two well-known mechanical engineering problems from the CEC 2020 test suite [72] were considered to assess the applicability of MFO-SFR in solving real-world optimization problems. The experimental results indicated that the proposed MFO-SFR algorithm boosted the performance of the canonical MFO by using an effective stagnation finding and replacing (SFR) strategy and an archive construction mechanism. Moreover, the statistical analysis revealed that the performance of the proposed MFO-SFR algorithm was superior to that of the contender algorithms.

The structure of the paper is as follows. Section 2 contains a review of related literature. Section 3 presents the MFO algorithm. In Section 4, the proposed MFO-SFR algorithm is explained in detail. Section 5 thoroughly evaluates the MFO-SFR's performance in addressing CEC 2018 benchmark test functions. Section 6 evaluates the applicability of the proposed MFO-SFR using two real-world mechanical engineering problems from the latest CEC 2020 test suite. Finally, Section 6 summarizes the results and outlines possible future directions of research.

2. Related Works

MFO variants used to solve different optimization problems are reviewed in this section.

Li et al. [60] boosted the performance of the canonical MFO by using the Lévy-flight strategy. Nadimi-Shahraki et al. [73] proved that the canonical MFO suffers from premature convergence, low population diversity, and an imbalance between search strategies in solving global optimization problems. Therefore, they proposed an improved moth-flame optimization (I-MFO) algorithm to cope with the abovementioned deficiencies. The I-MFO algorithm is equipped with the adapted wandering-around search strategy to maintain population diversity and escape from local optima. The chaos-enhanced MFO (CMFO) [62] algorithm was proposed to improve the performance of the MFO algorithm by incorporating chaos maps into population initialization, boundary handling, and parameter tuning. Pelusi et al. [74] proposed the improved moth-flame optimization (IMFO) algorithm using a hybrid phase, a dynamic crossover mechanism, and a fitness-dependent weight factor. The hybrid phase achieved a good trade-off between the exploration and exploitation phases, the dynamic crossover mechanism enhanced the population diversity, and the fitness-dependent weight factor improved the exploitation phase.

Xu et al. [64] proposed a series of MFO variants by combining the standard MFO algorithm with Gaussian mutation, Cauchy mutation, and Lévy mutation. Gaussian mutation was employed to improve its neighborhood-informed capability, Cauchy mutation was used to enhance its global exploration ability, and the Lévy mutation was employed to increase the randomness in the search process. Li et al. [65] proposed the ODSFMFO algorithm, which consists of an improved flame generation mechanism based on opposition-based learning and the differential evolution algorithm, an enhanced local search mechanism based on the shuffled frog leaping algorithm and a death mechanism. This algorithm maintained the quality of the population through opposition-based learning, population diversity using the differential evolution algorithm, the global search ability through the use of the shuffled frog leaping algorithm, and provided an escape from local optima via the use of the death mechanism. Nadimi-Shahraki et al. [75] proposed a migration-based moth–flame optimization (M-MFO) algorithm with a random migration operator, a guided migration operator, and a guiding archive to alleviate the low population diversity and poor exploration ability of MFO.

Ma et al. [76] developed an improved moth-flame optimization algorithm to prevent premature convergence to local minima. This algorithm uses the inertia weight of diversity feedback control to strike a balance between search strategies and maintain population diversity. Moreover, the mutation probability was added to improve the optimization performance. To enhance the diversity in the position of flames and the search strategy used for moths, Zhao et al. [77] developed an improved MFO (IMFO) algorithm. In this algorithm, the flames are generated through orthogonal opposition-based learning, and their positions are updated using a linear search and a mutation operator. Sapre et al. [78] introduced an opposition-based moth flame optimization method with Cauchy mutation and evolutionary boundary constraint handling (OMFO) to bypass the local optima and accelerate the convergence speed towards promising areas. Sahoo et al. [79] proposed a modified dynamic-opposite-learning-based MFO algorithm named m-MFO, using a modified dynamic-opposite learning strategy to enrich the performance of MFO in solving optimization problems. Other MFO variants include the double-evolutionary learning MFO algorithm (DELMFO) [80], the improved moth-flame optimization algorithm (IMFO) [81], the hybrid MFO and hill climbing (MFOHC) method [82], an enhanced MFO algorithm integrated with orthogonal learning and the Broyden–Fletcher–Goldfarb–Shanno (BFGSOLMFO) method [83], and quantum-behaved simulated annealing algorithm-based moth-flame optimization (QSMFO) [84].

Due to the simple structure of MFO and its low number of control parameters, it has great potential to solve real-world applications. However, the canonical MFO critically suffers from local optimum trapping and premature convergence during the optimization process, which results in low-quality solutions [85–87]. Therefore, many improved and hybrid variants have been developed to overcome these challenges. Sayed et al. [88] presented the SA-MFO algorithm, a hybrid of the MFO approach and the simulated annealing (SA) algorithm, to escape from local optima using SA and accelerate the search process using MFO. Many researchers have applied the MFO algorithm to solve the optimal power flow (OPF) problem [89–91]. An effective hybridization of the whale optimization algorithm and a modified moth-flame optimization algorithm named WMFO [68] was proposed to solve diverse scales of the OPF problem. Sahoo et al. [92] proposed a hybrid MFO and butterfly optimization algorithm (h-MFOBOA) to overcome shortcomings such as a slow convergence speed and poor exploitation ability in both optimizers. Sattar Khan et al. [93] adapted the MFO algorithm for an integrated power plant system containing stochastic wind. MFO has been applied to the solution of problems related to fuel cells in a renewable active distribution network [94], the identification of parameters for photovoltaic modules [95], and fuel consumption in variable-cycle engines [96], with promising results.

3. Moth-Flame Optimization (MFO) Algorithm

Nocturnal moths use celestial light sources to navigate over long distances accurately. They fly in a straight line with a constant angle toward the Moon or stars, and this behavior is called transverse orientation. However, when a moth flies toward a nearby artificial light, it thinks it is a star or the Moon. Therefore, the moth continually changes its flight angle to keep going in a straight line toward the light, resulting in a spiral motion around the artificial light. In 2015, this behavior was mathematically modeled in the moth-flame optimization algorithm [44] developed by Mirjalili to solve the global optimization problem, described in detail as follows.

In this approach to solving the optimization problem, the positions of moths evolve during predefined iterations. In the first iteration, moths are randomly distributed in the problem space using Equation (1), where X_{id} denotes the dth dimension of the ith moth position and the parameters Ub_d and Lb_d are the upper and lower boundaries for the dth dimension, respectively.

$$X_{id} = rand_{i,d} \times (Ub_d - Lb_d) + Lb_d, \qquad 1 \le d \le D \tag{1}$$

For the rest of the iterations, their new positions are updated based on the position of the flame. Therefore, the flame number (R) is computed using Equation (2), where the parameters N and *MaxIterations* denote the number of moths and the maximum number of iterations, respectively. Then, the positions of the flames are determined based on the stepwise procedure denoted in Table 1.

$$R = round\left(N - t \times \frac{N-1}{MaxIterations}\right) \tag{2}$$

Ultimately, for the flame number (R), each moth can update its position using the two different trials denoted in Equation (3), where $X_i\ (t+1)$ is the new position of the ith moth, $D_i{}'(t)$ is computed using Equation (4), b is the constant value, k is calculated using Equations (5) and (6), and $F_i\ (t)$ denotes the ith flame.

$$RX_i(t+1) = \begin{cases} D_i{}'(t) \times e^{bk} \times cos(2\pi k) + F_i(t) & i \le R \\ D_i{}''(t) \times e^{bk} \times cos(2\pi k) + F_R(t) & i > R \end{cases} \tag{3}$$

$$D_i{}'(t) = |F_i(t) - X_i(t)| \tag{4}$$

$$k = (a-1) \times rand(0,1) + 1 \tag{5}$$

$$a = -1 + t \times \left(\frac{-1}{MaxIterations} \right) \tag{6}$$

In the second trial (when $i > R$), the parameter $D_i{}''\ (t)$ is computed using Equation (7), and $F_R\ (t)$ is the current position of the Rth flame.

$$D_i{}'(t) = |F_R(t) - X_i(t)| \tag{7}$$

Table 1. Flame construction procedure.

Input: X: the positions of moths, Fit: the fitness values of moths, F: the position of the flame, and OF: the fitness values of flames.
Flame construction in the first iteration when t = 1.
1. Sort the vector Fit in ascending order and extract the sorted index in $\{j_1, j_2, \ldots, j_N\}$.
2. Construct the flame matrix $F\ (t) = \{F_1 \leftarrow X_{j1}, F_2 \leftarrow X_{j2}, \ldots, F_N \leftarrow X_{jN}\}$.
Flame construction for the rest iteration when t > 1.
1. Construct matrix $dual_{Pop}$ by combining matrices $F(t)$ and $X\ (t-1)$.
2. Construct vector $dual_{Fit}$ by combining vectors $OF(t)$ and $Fit\ (t-1)$.
3. Sort the vector $\mathrm{dual}_{\mathrm{Fit}}$ in ascending order and extract the sorted index in $\{j_1, j_2, \ldots, j_{2N}\}$.
4. Construct the flame matrix $F\ (t) = \{F_1 \leftarrow X_{j1}, F_2 \leftarrow X_{j2}, \ldots, F_N \leftarrow X_{jN}\}$.

4. The Proposed MFO-SFR Algorithm

According to the literature, the canonical MFO lacks an efficient operator to maintain population diversity. The search process may be biased by the best solutions obtained in each iteration [97]. This deficiency leads to premature convergence into unpromising regions, local optimum stagnation, and a decrease in the solution quality when solving complex problems. Hence, in this study, we were motivated to propose an enhanced moth-flame optimization algorithm named MFO-SFR to effectively maintain population diversity and mitigate the deficiencies mentioned above by introducing an effective stagnation finding and replacing (SFR) strategy.

Stagnation finding and replacing (SFR) strategy: Suppose that the matrix $X\ (t) = \{X_{1D}\ (t), \ldots, X_{iD}\ (t), \ldots, X_{ND}\ (t)\}$ denotes a moth population in the current iteration t in a D-dimensional search space. Each vector $X_{iD}\ (t)$ denotes the position of the ith moth in the problem space. The matrix $X\ (t)$ is initialized for the first iteration using a uniform random distribution. For the rest of the iterations (when $t \geq 2$), the new positions of the moths are determined using Equation (8), where $D_i^{\alpha}(t)$ and $D_i^{\beta}(t)$ are the main elements of the SFR strategy, which is computed using Equations (9) and (10), respectively. A constant b expresses the shape of the logarithmic spiral, and τ is a random number between the intervals -1 and 1. $F_j(t)$ and $F_R(t)$ are the positions of the jth flame and the Rth flame such that the parameter R is computed using Equation (2). In Equation (9), vector $M_i(t)$ is determined using Definition 1. To find the stagnant solutions, the mean of the distance or φ_i is calculated using Equation (11), where X_{iq} is the qth dimension of the ith moth. F_{jq} is the qth dimension of the jth flame in which the index j is determined by Equation (12), which sorts the results obtained from Equation (11) in descending order to obtain the indexes, then applies them as flame indexes in Equation (10).

$$X_i(t+1) = \begin{cases} D_i^{\alpha}(t) \times e^{b\tau} \times cos(2\pi t) + F_j(t) & if\ i \leq R(t) \\ \\ D_i^{\beta}(t) \times e^{b\tau} \times cos(2\pi t) + F_R(t) & else \end{cases} \tag{8}$$

$$D_i^{\alpha}(t) = \left|F_j(t) - M_i(t)\right| \tag{9}$$

$$D_i^{\beta}(t) = \begin{cases} \left|F_j(t) - X_i(t)\right| & \varphi_i > 0 \\ Selecting\ a\ random\ position\ from\ the\ Arc & \varphi_i = 0 \end{cases} \tag{10}$$

$$\{\varphi_1, \ldots, \varphi_i, \ldots, \varphi_N\} \leftarrow \varphi_i = \frac{1}{D} \times \sum_{q=1}^{D} \left|F_{jq}(t) - X_{iq}(t)\right| \tag{11}$$

$$\{\varphi_1, \ldots, \varphi_j, \ldots, \varphi_N\} \leftarrow \text{Sort}(\varphi_1, \ldots, \varphi_i, \ldots, \varphi_N) \tag{12}$$

Definition 1. (*Archive construction*): *The main idea behind archive construction is to enrich the population diversity by preserving the generated representative flame and boost the convergence of solutions toward promising areas by preserving the best solutions in each iteration. To construct the archive Arc, consider the matrix* $M = \{M_1, \ldots, M_i, \ldots, M_\kappa\}$ *as the memory of the Arc with predefined* κ. *Each* $M_i = [m_{i1}, m_{i2}, \ldots, m_{iD}]$ *denotes this memory's vector position, which is generated using Algorithm 1. First,* $dual_{Pop}$ *and* $dual_{Fit}$ *are created based on the flame construction process described in* Table 1. *Then, the representative flame (RF) with the average of flames' positions is computed using Equation (13), where C is the total number of considered moths and* F_{id} *denotes the dth dimension of the ith flame. Finally, the global best flame and RF position are archived as two new entries in the memory M. In regard to inserting these new entries; they are randomly replaced with two existing entries if the memory is full.*

$$RF_d(t) = \frac{1}{C} \sum_{i=1}^{C} F_{id}(t) \tag{13}$$

In addition, MFO-SFR checks the feasibility of the position of the new moths to return those that have violated the problem space boundaries by generating random positions in the range of the problem space.

Algorithm 1. The pseudocode of the archive construction process.

```
Input: $C$: Number of considered flames, and $\kappa$: the maximum size of the archive $Arc$.
Output: Returns the archive $Arc$.
1.  begin
2.  $dual_{Pop}$ and $dual_{Fit}$ are created based on flame construction defined in Table 1.
3.  $Fit_{Best}$ = Ascending order of the vector $dual_{Fit}$ and selecting the best $N$ values.
4.  $Pop_{Best}$ = The corresponding positions of vector $Fit_{Best}$.
5.  Computing $RF$ using Equation (13) for C number of considered flames.
6.  If the current memory size < $\kappa-1$.
7.      Inserting $RF$ and the global best flame into the Arc.
8.  else
9.      Replacing $RF$ and the global best flame with two existing memory entries.
10. end if
11. end
```

Complexity Analysis

Regarding the pseudocode of MFO-SFR shown in Algorithm 2, the MFO-SFR algorithm consists of six distinct phases: initialization, flame construction, archive construction, movement, correcting the violated positions, and updating the positions. In the initialization phase, N moths are randomly distributed in a D-dimensional search space with an O(ND) computational complexity. In the flame construction phase, flames are constructed differently with the computational complexity of O(N^2), considering the worst case for the quicksort algorithm. The computational complexity of the archive construction phase using Algorithm 1 is O(N^2 + ND), because lines 2−4 have the complexity of O(N^2) with respect to the original paper's definition of MFO, and Equation (13) has O(ND) in the worst case. The cost of the movement phase is O(ND), using either Equations (8) and (9) when $i \leq R$ or using Equations (8) and (10) when $i > R$. Then, the feasibility of the new positions is checked to correct the violated positions with the computational complexity of O(ND). Finally, the updating phase is performed with O(ND) computational complexity. Therefore, considering T iterations, the computational complexity of MFO-SFR is O(ND + N^2 + T(2N^2 + 4ND) or O(TN^2 + TND). In the same fashion, the space complexity is O(N + ND + κ), considering that the memory is reusable and the size of the memory is κ. Thus, the space complexity of MFO-SFR is O(ND + $D^2 log\ N$).

Algorithm 2. The pseudocode of the proposed MFO-SFR algorithm.

```
Input: $N$: Number of moths, $MaxIterations$: Maximum iterations, and $D$: Dimension size.
Output: Returns the position of the global best flame and its fitness value.
1.  Begin
2.  Initiating matrix $X$ ($t$) using a uniform random distribution in the $D$-dimensional search space.
3.  Computing the fitness value of $X$ ($t$) and storing them in vector $OX$ ($t$).
4.  Constructing the flame fitness value $OF$ by ascending order of the vector $OM$ ($t$).
5.  Constructing the flame positions $F$ based on their obtained vector $OF$.
6.  While $t \leq MaxIterations$
7.      Updating $F$ and $OF$ by the best $N$ moths from $F$ and current $X$.
8.      Computing the flame number $R$ using Equation (2).
9.      Archiving using Algorithm 1.
10.     For $i$ = 1: $N$
11.         If $i \leq R$
12.             Computing the distance between flame $F_i$ ($t$) and $M_i$ ($t$) using Equation (9).
13.             Updating the position of $X_i$ ($t$) using Equation (8).
14.         else
15.             Computing the distance using Equation (10).
16.             Updating the position of $X_i$ ($t$) using Equation (8).
17.         End if
18.         Checking the feasibility and correcting the new position.
19.         Computing the fitness value of the new position.
20.     End for
21.     Updating the global best flame.
22. End while
```

5. Evaluation of the Proposed MFO-SFR Algorithm

In this section we present our evaluation of the performance of the proposed MFO-SFR algorithm in solving global optimization problems from the CEC 2018 benchmark test suite [67]. This test suite is suitable for evaluating the proposed algorithm in terms of its local optimum avoidance ability and the diversity of solutions as it consists of 29 test functions with different characteristics, such as unimodal, multimodal, and hybrid functions, as well as compositions with various dimensions (D), specifically, 30 and 50 dimensions. Moreover, in this section, we also present two separate sets of experiments conducted to extensively assess and compare the performance of the proposed MFO-SFR algorithm with several well-known optimization methods. The proposed algorithm was compared to the original MFO and its variants in the first set, and then, in the second experimental set, it was compared to other prominent and recent optimizers. In both experiment sets, all comparative algorithms' control parameter values were adjusted to match those in their original articles, as depicted in Table 2. All of the algorithms were executed 20 times on a laptop with an Intel Core i7-10750H CPU (2.60 GHz), 24 GB of memory, and MATLAB R2022a with a maximum of $(D \times 10^4)/N$ iterations, where D represents the dimension size of the problem and N is the population size, which was set to 100 in this study.

Table 2. Parameter values for the optimization algorithms.

Alg.	Parameter Settings
MFO	$b = 1$, a decreased linearly from -1 to -2.
LMFO	$\beta = 1.5$, μ and v are normal distributions, Γ is the gamma function.
WCMFO	The number of rivers and seas = 4.
CMFO	$b = 1$, a decreased linearly from -1 to -2, chaotic map = Singer.
ODSFMFO	$m = 6$, $pc = 0.5$, $\gamma = 5$, $\alpha = 1$, $l = 10$, $b = 1$, $\beta = 1.5$.
SMFO	r_4 = random number between the interval (0, 1).
WMFO	α decreased linearly from 2 to 0, $b = 1$.
PSO	$c_1 = c_2 = 2$, $vmax = 6$, $w = 0.9$.
KH	$V_f = 0.02$, $D_{max} = 0.005$, $N_{max} = 0.01$, $Sr = 0$.
GWO	The parameter a decreased linearly from 2 to 0.
CSA	$AP = 0.1$, $fl = 2$.
HOA	$w = 1$, $\delta_D = 0.02$, $\delta_I = 0.02$, $g_\delta = 1.5$, $h_\beta = 0.9$, $h_\gamma = 0.5$, $s_\beta = 0.2$, $s_\gamma = 0.1$, $i_\gamma = 0.3$, $d_\alpha = 0.5$, $d_\beta = 0.2$, $d_\gamma = 0.1$, $r_\delta = 0.1$, $r_\gamma = 0.05$
MFO-SFR	$b = 1$, a decreased linearly from -1 to -2, κ = round $(D^2 \times (log\ N))$, $C = N/5$.

To investigate the impact of the archive introduced in Equation (10), a numerical pretest was performed on the canonical MFO algorithm using the CEC 2018 benchmark test suite on dimension 30. In this pre-test percentage of situations when the parameter φ_i was equal to zero is computed and reported in Table A1 of Appendix A. The results reported in Table A1 in Appendix A showed that for some test functions, especially hybrid and composition ones, the percentage of stagnant solutions was high enough to affect the quality of the generated solutions.

5.1. Comparing the Proposed MFO-SFR Algorithm with MFO Variants

In this set of experiments, we compared the proposed MFO-SFR algorithm with moth-flame optimization (MFO) [44] and its variants, including LMFO [60], WCMFO [61], CMFO [62], ODSFMFO [65], SMFO [63], and WMFO [68]. Table 3 compares the results of the proposed MFO-SFR algorithm with those of MFO and its variants in solving the CEC 2018 test functions with 30 dimensions. The results acquired from the unimodal test functions F_1 and F_3 demonstrated that MFO-SFR had an acceptable exploitation potential compared to the other algorithms. The results from multimodal test functions F_4–F_{10} indicated that the proposed algorithm was able to efficiently search the problem space and find the unvisited areas by maintaining its population diversity throughout the optimization process. The overall results of the hybrid and composition functions F_{11}–F_{30} confirmed that MFO-SFR avoided local optimum solutions by striking a balance between exploration and exploitation abilities. Moreover, the final rows of Tables 3 and 4 reveal that according to the Friedman test [98], the proposed MFO-SFR algorithm ranked first among the algorithms, including MFO and the other investigated variants.

Table 3. Comparison of MFO-SFR with MFO variants for CEC 2018 test functions with D = 30.

F.	Metrics	MFO	LMFO	WCMFO	CMFO	ODSFMFO	SMFO	WMFO	MFO-SFR
F_1	Avg	6.278×10^9	2.544×10^7	1.317×10^4	1.078×10^8	6.016×10^6	3.119×10^{10}	3.822×10^3	1.791×10^3
	Min	1.027×10^9	1.899×10^7	1.924×10^3	3.760×10^6	9.949×10^5	1.734×10^{10}	1.013×10^2	1.017×10^2
F_3	Avg	9.453×10^4	3.473×10^3	1.541×10^3	5.059×10^4	3.050×10^4	8.300×10^4	3.909×10^2	1.312×10^4
	Min	1.203×10^4	1.499×10^3	3.111×10^2	2.945×10^4	1.631×10^4	7.186×10^4	3.007×10^2	7.513×10^3
F_4	Avg	8.558×10^2	4.919×10^2	4.846×10^2	6.960×10^2	5.356×10^2	5.612×10^3	4.810×10^2	4.914×10^2
	Min	4.991×10^2	4.742×10^2	4.009×10^2	5.139×10^2	4.985×10^2	2.322×10^3	4.249×10^2	4.700×10^2

Table 3. *Cont.*

F.	Metrics	MFO	LMFO	WCMFO	CMFO	ODSFMFO	SMFO	WMFO	MFO-SFR
F_5	Avg	6.740×10^2	6.300×10^2	6.721×10^2	6.073×10^2	5.506×10^2	8.725×10^2	6.739×10^2	5.227×10^2
	Min	6.114×10^2	5.816×10^2	6.126×10^2	5.736×10^2	5.270×10^2	8.105×10^2	6.234×10^2	5.109×10^2
F_6	Avg	6.260×10^2	6.030×10^2	6.236×10^2	6.189×10^2	6.037×10^2	6.814×10^2	6.366×10^2	6.000×10^2
	Min	6.113×10^2	6.018×10^2	6.137×10^2	6.086×10^2	6.010×10^2	6.571×10^2	6.143×10^2	6.000×10^2
F_7	Avg	1.007×10^3	8.716×10^2	9.050×10^2	9.430×10^2	8.099×10^2	1.359×10^3	1.056×10^3	7.669×10^2
	Min	8.538×10^2	8.311×10^2	8.045×10^2	8.684×10^2	7.824×10^2	1.198×10^3	9.248×10^2	7.460×10^2
F_8	Avg	9.895×10^2	9.375×10^2	9.839×10^2	9.097×10^2	8.528×10^2	1.093×10^3	9.539×10^2	8.209×10^2
	Min	9.126×10^2	8.978×10^2	9.344×10^2	8.645×10^2	8.343×10^2	1.052×10^3	8.547×10^2	8.090×10^2
F_9	Avg	6.219×10^3	9.256×10^2	8.623×10^3	2.331×10^3	1.118×10^3	9.431×10^3	4.543×10^3	9.038×10^2
	Min	3.323×10^3	9.074×10^2	5.118×10^3	1.476×10^3	9.647×10^2	7.359×10^3	1.675×10^3	9.005×10^2
F_{10}	Avg	5.259×10^3	4.240×10^3	4.848×10^3	5.005×10^3	4.332×10^3	8.272×10^3	5.192×10^3	4.062×10^3
	Min	4.231×10^3	3.205×10^3	4.003×10^3	4.204×10^3	3.570×10^3	7.449×10^3	3.759×10^3	2.461×10^3
F_{11}	Avg	3.967×10^3	1.314×10^3	1.363×10^3	1.985×10^3	1.284×10^3	5.799×10^3	1.248×10^3	1.143×10^3
	Min	1.370×10^3	1.180×10^3	1.252×10^3	1.206×10^3	1.204×10^3	2.547×10^3	1.170×10^3	1.107×10^3
F_{12}	Avg	9.043×10^7	5.251×10^6	1.416×10^6	2.113×10^7	2.157×10^6	4.342×10^9	1.014×10^5	1.508×10^5
	Min	7.305×10^4	1.578×10^6	3.718×10^4	7.171×10^5	2.328×10^5	2.607×10^9	6.932×10^3	2.035×10^4
F_{13}	Avg	4.593×10^6	4.072×10^5	9.457×10^4	9.006×10^3	1.184×10^4	7.405×10^8	6.660×10^3	6.405×10^3
	Min	1.003×10^4	1.634×10^5	1.150×10^4	2.446×10^3	1.596×10^3	1.145×10^8	1.400×10^3	1.690×10^3
F_{14}	Avg	6.942×10^4	2.500×10^4	1.872×10^4	3.941×10^4	5.651×10^4	1.715×10^6	1.406×10^4	8.200×10^3
	Min	5.450×10^3	2.821×10^3	4.075×10^3	6.379×10^3	4.686×10^3	7.879×10^4	3.027×10^3	2.021×10^3
F_{15}	Avg	3.090×10^4	8.218×10^4	3.207×10^4	5.756×10^3	5.070×10^3	4.161×10^7	1.157×10^4	5.614×10^3
	Min	5.117×10^3	4.614×10^4	2.547×10^3	1.707×10^3	1.703×10^3	1.868×10^6	1.609×10^3	1.515×10^3
F_{16}	Avg	2.956×10^3	2.564×10^3	2.867×10^3	2.709×10^3	2.366×10^3	4.223×10^3	2.662×10^3	1.855×10^3
	Min	2.398×10^3	2.101×10^3	2.267×10^3	2.241×10^3	1.965×10^3	3.565×10^3	2.068×10^3	1.617×10^3
F_{17}	Avg	2.349×10^3	2.192×10^3	2.315×10^3	2.056×10^3	1.985×10^3	2.788×10^3	2.234×10^3	1.745×10^3
	Min	1.975×10^3	1.925×10^3	1.942×10^3	1.818×10^3	1.764×10^3	2.359×10^3	1.958×10^3	1.727×10^3
F_{18}	Avg	2.830×10^6	2.674×10^5	1.804×10^5	7.780×10^5	8.975×10^5	5.330×10^7	8.188×10^4	1.493×10^5
	Min	7.725×10^4	3.452×10^4	4.774×10^4	7.998×10^4	9.364×10^4	2.825×10^6	6.883×10^3	4.305×10^4
F_{19}	Avg	4.261×10^6	7.040×10^4	3.083×10^4	2.505×10^4	7.822×10^3	7.588×10^7	1.560×10^4	6.534×10^3
	Min	1.293×10^4	3.487×10^4	2.168×10^3	3.280×10^3	1.968×10^3	5.192×10^6	2.310×10^3	1.910×10^3
F_{20}	Avg	2.537×10^3	2.398×10^3	2.528×10^3	2.402×10^3	2.287×10^3	2.837×10^3	2.690×10^3	2.091×10^3
	Min	2.215×10^3	2.117×10^3	2.103×10^3	2.185×10^3	2.053×10^3	2.454×10^3	2.294×10^3	2.004×10^3
F_{21}	Avg	2.472×10^3	2.439×10^3	2.485×10^3	2.384×10^3	2.351×10^3	2.630×10^3	2.462×10^3	2.321×10^3
	Min	2.420×10^3	2.378×10^3	2.430×10^3	2.338×10^3	2.331×10^3	2.363×10^3	2.389×10^3	2.312×10^3
F_{22}	Avg	6.353×10^3	4.878×10^3	6.611×10^3	2.380×10^3	2.319×10^3	8.681×10^3	5.292×10^3	2.300×10^3
	Min	3.223×10^3	2.325×10^3	5.330×10^3	2.319×10^3	2.305×10^3	5.677×10^3	2.300×10^3	2.300×10^3
F_{23}	Avg	2.811×10^3	2.754×10^3	2.796×10^3	2.797×10^3	2.722×10^3	3.273×10^3	2.861×10^3	2.671×10^3
	Min	2.740×10^3	2.724×10^3	2.749×10^3	2.734×10^3	2.697×10^3	3.027×10^3	2.763×10^3	2.654×10^3
F_{24}	Avg	2.979×10^3	2.924×10^3	2.972×10^3	2.948×10^3	2.872×10^3	3.482×10^3	3.003×10^3	2.844×10^3
	Min	2.926×10^3	2.888×10^3	2.927×10^3	2.887×10^3	2.848×10^3	3.217×10^3	2.912×10^3	2.828×10^3
F_{25}	Avg	3.181×10^3	2.888×10^3	2.894×10^3	3.011×10^3	2.925×10^3	3.972×10^3	2.900×10^3	2.887×10^3
	Min	2.895×10^3	2.885×10^3	2.884×10^3	2.935×10^3	2.890×10^3	3.467×10^3	2.884×10^3	2.887×10^3
F_{26}	Avg	5.650×10^3	4.854×10^3	5.538×10^3	4.465×10^3	4.415×10^3	9.093×10^3	5.841×10^3	3.903×10^3
	Min	4.921×10^3	4.504×10^3	5.074×10^3	3.113×10^3	2.876×10^3	5.057×10^3	4.741×10^3	3.739×10^3
F_{27}	Avg	3.233×10^3	3.223×10^3	3.229×10^3	3.285×10^3	3.244×10^3	3.754×10^3	3.276×10^3	3.219×10^3
	Min	3.206×10^3	3.194×10^3	3.204×10^3	3.238×10^3	3.218×10^3	3.538×10^3	3.220×10^3	3.208×10^3
F_{28}	Avg	3.756×10^3	3.270×10^3	3.192×10^3	3.376×10^3	3.294×10^3	5.462×10^3	3.199×10^3	3.216×10^3
	Min	3.263×10^3	3.211×10^3	3.100×10^3	3.265×10^3	3.271×10^3	4.419×10^3	3.122×10^3	3.196×10^3

Table 3. *Cont.*

F.	Metrics	MFO	LMFO	WCMFO	CMFO	ODSFMFO	SMFO	WMFO	MFO-SFR
F_{29}	Avg	4.014×10^3	3.764×10^3	3.949×10^3	4.040×10^3	3.691×10^3	5.639×10^3	4.068×10^3	3.414×10^3
	Min	3.499×10^3	3.410×10^3	3.574×10^3	3.629×10^3	3.475×10^3	4.728×10^3	3.545×10^3	3.323×10^3
F_{30}	Avg	2.524×10^5	1.426×10^5	3.318×10^4	7.742×10^5	1.803×10^4	2.326×10^8	1.079×10^4	7.835×10^3
	Min	7.219×10^3	6.606×10^4	1.642×10^4	5.609×10^4	7.769×10^3	2.468×10^7	5.674×10^3	6.362×10^3
Average rank		6.06	3.95	4.51	4.57	3.28	7.91	4.18	1.55
Total rank		7	3	5	6	2	8	4	1

Table 4. Comparison of MFO-SFR with MFO variants for CEC 2018 test functions with D = 50.

F.	Metrics	MFO	LMFO	WCMFO	CMFO	ODSFMFO	SMFO	WMFO	MFO-SFR
F_1	Avg	3.036×10^{10}	1.091×10^8	6.099×10^4	1.323×10^9	2.624×10^8	7.209×10^{10}	4.264×10^3	3.463×10^4
	Min	7.064×10^3	8.074×10^7	8.054×10^2	1.287×10^8	3.470×10^7	5.140×10^{10}	1.054×10^2	9.385×3
F_3	Avg	1.540×10^5	3.139×10^4	1.413×10^4	1.004×10^5	9.325×10^4	1.770×10^5	9.948×10^2	5.495×10^4
	Min	1.176×10^4	1.960×10^4	2.418×10^3	7.381×10^4	6.538×10^4	1.457×10^5	3.217×10^2	4.223×10^4
F_4	Avg	4.178×10^3	5.786×10^2	5.431×10^2	1.182×10^3	7.401×10^2	1.835×10^4	5.385×10^2	5.867×10^2
	Min	1.187×10^3	5.296×10^2	4.286×10^2	5.419×10^2	6.608×10^2	1.005×10^4	4.961×10^2	5.196×10^2
F_5	Avg	9.086×10^2	8.080×10^2	9.240×10^2	8.065×10^2	6.269×10^2	1.125×10^3	8.608×10^2	5.624×10^2
	Min	7.996×10^2	7.226×10^2	7.743×10^2	6.742×10^2	5.862×10^2	1.032×10^3	7.209×10^2	5.318×10^2
F_6	Avg	6.455×10^2	6.078×10^2	6.395×10^2	6.356×10^2	6.075×10^2	6.888×10^2	6.513×10^2	6.001×10^2
	Min	6.270×10^2	6.035×10^2	6.165×10^2	6.239×10^2	6.041×10^2	6.780×10^2	6.291×10^2	6.000×10^2
F_7	Avg	1.728×10^3	1.081×10^3	1.139×10^3	1.207×10^3	9.855×10^2	1.937×10^3	1.461×10^3	8.682×10^2
	Min	1.119×10^3	1.011×10^3	1.023×10^3	1.031×10^3	8.795×10^2	1.769×10^3	1.204×10^3	8.099×10^2
F_8	Avg	1.217×10^3	1.107×10^3	1.213×10^3	1.055×10^3	9.213×10^2	1.406×10^3	1.131×10^3	8.610×10^2
	Min	1.050×10^3	1.021×10^3	1.096×10^3	9.983×10^2	8.625×10^2	1.315×10^3	1.021×10^3	8.318×10^2
F_9	Avg	1.651×10^4	1.737×10^3	2.097×10^4	6.212×10^3	1.717×10^3	3.051×10^4	1.190×10^4	9.243×10^2
	Min	8.748×10^3	9.529×10^2	1.190×10^4	3.607×10^3	1.299×10^3	1.925×10^4	5.498×10^3	9.066×10^2
F_{10}	Avg	8.426×10^3	7.527×10^3	7.974×10^3	7.980×10^3	7.490×10^3	1.387×10^4	7.755×10^3	6.534×10^3
	Min	6.288×10^3	6.340×10^3	6.303×10^3	6.040×10^3	5.766×10^3	1.198×10^4	6.397×10^3	5.135×10^3
F_{11}	Avg	5.571×10^3	1.594×10^3	1.469×10^3	2.114×10^3	1.865×10^3	1.495×10^4	1.299×10^3	1.259×10^3
	Min	1.574×10^3	1.439×10^3	1.291×10^3	1.417×10^3	1.394×10^3	8.985×10^3	1.200×10^3	1.146×10^3
F_{12}	Avg	2.581×10^9	4.574×10^7	6.833×10^6	1.705×10^8	1.999×10^7	3.109×10^{10}	5.722×10^5	1.873×10^6
	Min	6.409×10^7	2.731×10^7	1.384×10^6	1.878×10^6	7.129×10^6	1.600×10^{10}	1.341×10^5	1.078×10^6
F_{13}	Avg	2.561×10^8	2.672×10^6	9.255×10^4	1.340×10^5	1.729×10^4	1.251×10^{10}	8.898×10^3	5.362×10^3
	Min	1.454×10^5	1.614×10^6	3.011×10^4	5.781×10^3	9.648×10^3	1.435×10^9	2.611×10^3	1.749×10^3
F_{14}	Avg	9.567×10^5	1.375×10^5	6.855×10^4	1.073×10^5	4.132×10^5	2.622×10^7	3.670×10^4	4.016×10^4
	Min	1.246×10^4	3.771×10^4	2.161×10^4	9.772×10^3	7.234×10^4	8.185×10^5	1.148×10^4	1.202×10^4
F_{15}	Avg	1.078×10^7	5.308×10^5	6.616×10^4	8.967×10^3	6.016×10^3	1.341×10^9	7.075×10^3	2.964×10^3
	Min	4.298×10^4	3.335×10^5	1.422×10^4	1.884×10^3	2.543×10^3	1.237×10^8	1.943×10^3	1.534×10^3
F_{16}	Avg	4.104×10^3	3.570×10^3	3.769×10^3	3.335×10^3	2.915×10^3	6.808×10^3	3.575×10^3	2.614×10^3
	Min	3.133×10^3	2.836×10^3	2.788×10^3	2.616×10^3	2.404×10^3	5.302×10^3	2.509×10^3	2.148×10^3
F_{17}	Avg	3.846×10^3	3.218×10^3	3.787×10^3	3.151×10^3	2.690×10^3	4.919×10^3	3.478×10^3	2.474×10^3
	Min	3.034×10^3	2.568×10^3	3.044×10^3	2.615×10^3	2.084×10^3	3.399×10^3	2.827×10^3	2.018×10^3
F_{18}	Avg	4.168×10^6	1.053×10^6	3.688×10^5	2.670×10^6	1.816×10^6	5.905×10^7	1.937×10^5	1.247×10^6
	Min	1.543×10^5	2.843×10^5	1.381×10^5	2.302×10^5	1.304×10^5	5.306×10^6	3.375×10^4	1.067×10^5
F_{19}	Avg	2.346×10^6	2.754×10^5	2.368×10^4	6.213×10^4	1.682×10^4	9.590×10^8	1.541×10^4	1.276×10^4
	Min	5.030×10^3	1.841×10^5	2.700×10^3	5.247×10^3	2.057×10^3	2.437×10^7	2.172×10^3	2.447×10^3
F_{20}	Avg	3.529×10^3	2.999×10^3	3.311×10^3	3.108×10^3	2.830×10^3	3.923×10^3	3.317×10^3	2.485×10^3
	Min	3.116×10^3	2.429×10^3	2.655×10^3	2.572×10^3	2.495×10^3	3.475×10^3	2.534×10^3	2.081×10^3

Table 4. *Cont.*

F.	Metrics	MFO	LMFO	WCMFO	CMFO	ODSFMFO	SMFO	WMFO	MFO-SFR
F_{21}	Avg	2.682×10^3	2.604×10^3	2.720×10^3	2.503×10^3	2.408×10^3	3.071×10^3	2.635×10^3	2.360×10^3
	Min	2.575×10^3	2.528×10^3	2.590×10^3	2.445×10^3	2.379×10^3	2.938×10^3	2.518×10^3	2.335×10^3
F_{22}	Avg	1.028×10^4	9.106×10^3	9.780×10^3	7.972×10^3	5.227×10^3	1.605×10^4	9.538×10^3	8.339×10^3
	Min	8.688×10^3	7.734×10^3	8.346×10^3	2.497×10^3	2.436×10^3	1.474×10^4	8.203×10^3	6.317×10^3
F_{23}	Avg	3.133×10^3	3.009×10^3	3.095×10^3	3.137×10^3	2.888×10^3	3.969×10^3	3.183×10^3	2.793×10^3
	Min	3.013×10^3	2.945×10^3	2.974×10^3	2.979×10^3	2.822×10^3	3.594×10^3	3.056×10^3	2.761×10^3
F_{24}	Avg	3.197×10^3	3.135×10^3	3.224×10^3	3.217×10^3	3.030×10^3	4.292×10^3	3.274×10^3	2.970×10^3
	Min	3.098×10^3	3.071×10^3	3.101×10^3	3.098×10^3	2.978×10^3	3.875×10^3	3.095×10^3	2.931×10^3
F_{25}	Avg	5.123×10^3	3.062×10^3	3.048×10^3	3.889×10^3	3.242×10^3	1.069×10^4	3.061×10^3	3.070×10^3
	Min	3.031×10^3	2.994×10^3	2.964×10^3	3.242×10^3	3.176×10^3	7.290×10^3	3.021×10^3	2.985×10^3
F_{26}	Avg	8.137×10^3	6.855×10^3	8.205×10^3	8.456×10^3	5.513×10^3	1.577×10^4	8.342×10^3	4.406×10^3
	Min	6.910×10^3	6.234×10^3	7.239×10^3	5.759×10^3	4.905×10^3	1.445×10^4	2.900×10^3	4.051×10^3
F_{27}	Avg	3.538×10^3	3.403×10^3	3.489×10^3	4.237×10^3	3.525×10^3	5.612×10^3	3.759×10^3	3.307×10^3
	Min	3.407×10^3	3.297×10^3	3.361×10^3	3.897×10^3	3.448×10^3	4.453×10^3	3.460×10^3	3.266×10^3
F_{28}	Avg	7.554×10^3	3.555×10^3	3.296×10^3	4.481×10^3	3.749×10^3	9.637×10^3	3.300×10^3	3.378×10^3
	Min	4.720×10^3	3.268×10^3	3.259×10^3	3.882×10^3	3.472×10^3	8.008×10^3	3.259×10^3	3.310×10^3
F_{29}	Avg	5.133×10^3	4.380×10^3	4.681×10^3	5.199×10^3	4.191×10^3	1.608×10^4	4.870×10^3	3.545×10^3
	Min	4.271×10^3	3.944×10^3	3.587×10^3	4.366×10^3	3.748×10^3	8.290×10^3	4.292×10^3	3.289×10^3
F_{30}	Avg	2.924×10^7	5.428×10^6	2.810×10^6	2.564×10^7	1.768×10^6	2.271×10^9	1.204×10^6	1.144×10^6
	Min	2.389×10^6	3.442×10^6	1.262×10^6	8.984×10^6	9.999×10^5	2.782×10^8	6.441×10^5	9.567×10^5
Average rank		6.29	3.82	4.25	4.78	3.34	7.93	3.79	1.79
Total rank		7	3	5	6	2	8	4	1

Table 4 presents the average and minimum fitness values obtained from the proposed MFO-SFR algorithms, MFO, and its six variants in solving the CEC 2018 benchmark test functions with 50 dimensions. Overall, the results showed that the proposed MFO-SFR algorithm provided competitive results for most test functions, and it ranked first according to the Friedman test results, which are reported in the final row of the table. Additionally, an exploratory data analysis is depicted in Figure 1 to show the ranking of algorithms for each function. Overall, it can be seen that the proposed MFO-SFR algorithm surrounds the center of the radar chart for most test functions in 30 and 50 dimensions. For instance, for F_1, the proposed MFO-SFR algorithm was ranked first in 30 dimensions and third in 50 dimensions, whereas WMFO and the canonical MFO algorithms were ranked second and seventh for 30 and 50 dimensions, respectively. For F_{12}, it can be seen that MFO-SFR was ranked second, MFO was ranked seventh, and WMFO was ranked first for 30 dimensions, and these three algorithms were ranked second, seventh, and first, respectively, for 50 dimensions. For F_{27}, MFO-SFR and WMFO were ranked first and sixth in both 30 and 50 dimensions, whereas the canonical MFO algorithm was ranked fourth in 30 dimensions and fifth in 50 dimensions.

The convergence comparison of the proposed MFO-SFR algorithm and the other studied algorithms is shown in Figure 2. For F_1 in 30 dimensions, it can be seen that although MFO-SFR exhibited prolonged convergence, it provided the best solution compared to the other algorithms. In 50 dimensions, however, it ranked second after WMFO. For multimodal functions F_5 and F_7, the convergence trend of MFO-SFR continued up to the final iterations, whereas most of the competitors were flattened in local optimum zones. As evidence of the adequate balance between exploration and exploitation, for hybrid functions F_{10} and F_{16}, MFO-SFR exhibited sharp movements in the first half of the iterations and relatively modest fluctuations in the second half. Ultimately, for composition test functions F_{21}, F_{26}, and F_{30}, MFO-SFR exhibited a gradual trend toward the optimum solutions after beginning its convergence with a sharply descending slope. This behavior indicates the capacity of MFO-SFR to bypass the local optimum and avoid premature convergence.

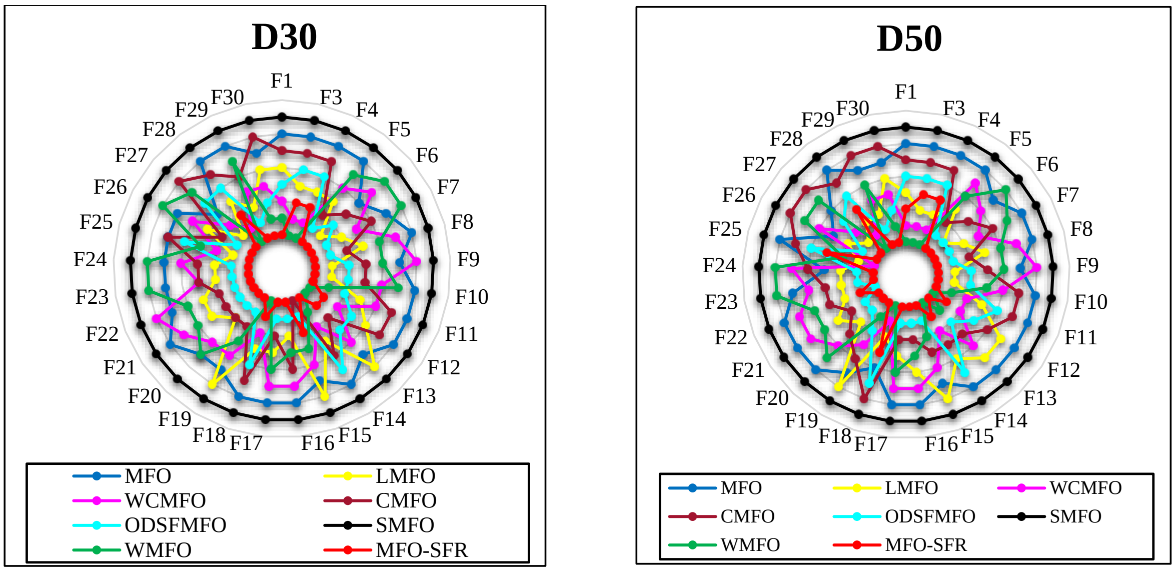


Figure 1. Exploratory data analysis of MFO-SFR, MFO, and its variants on CEC 2018 with 30 and 50 dimensions.

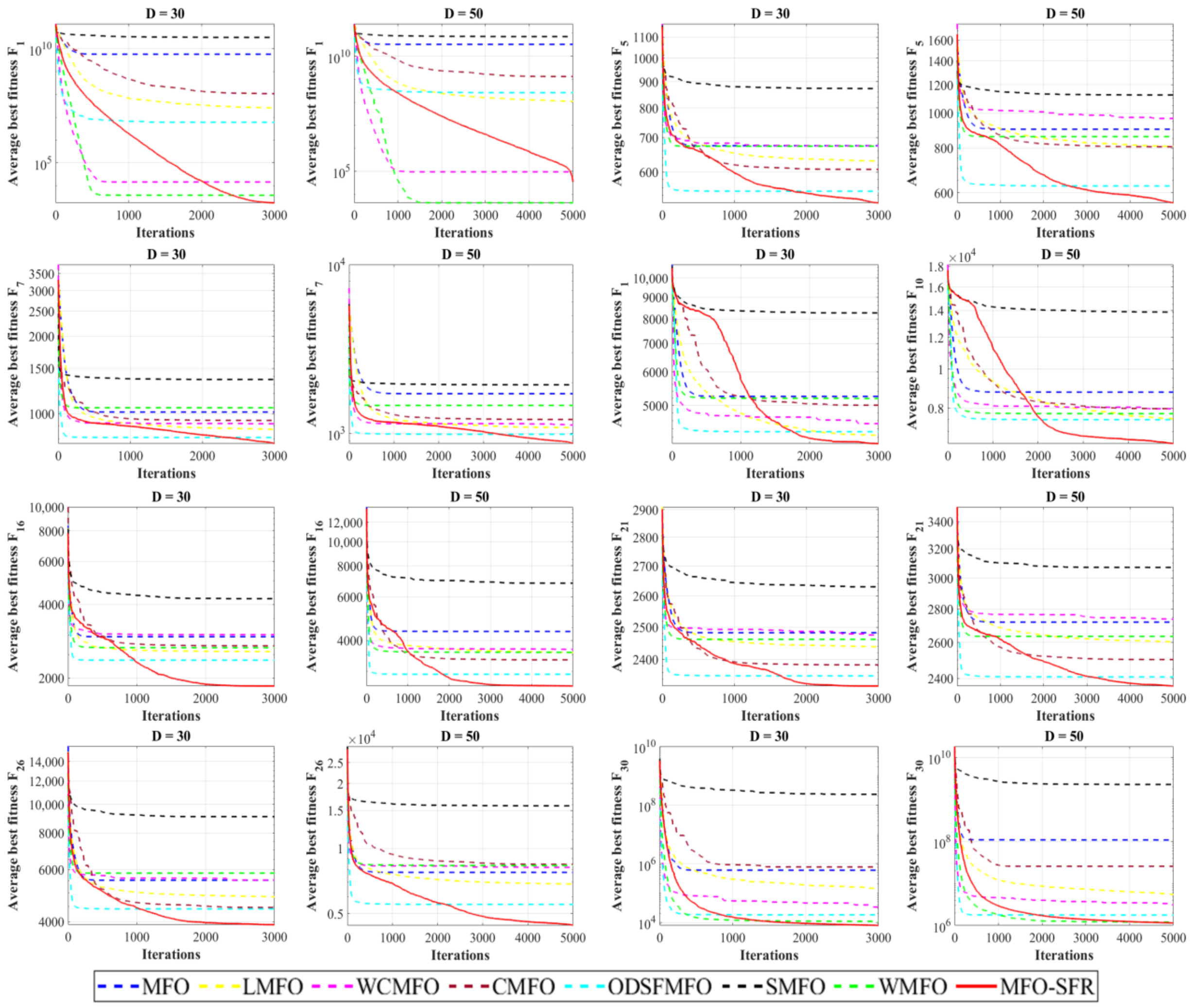


Figure 2. Convergence comparison of MFO-SFR, MFO, and its variants on CEC 2018 with D = 30 and 50.

5.2. Comparing the Proposed MFO-SFR Algorithm with Other Well-Known Optimization Algorithms

The second set of experiments, we compared the performance of the proposed MFO-SFR algorithm with the well-known representative metaheuristic algorithms presented in the literature, including particle swarm optimization (PSO) [39], krill herd (KH) [69], grey wolf optimization (GWO) [70], the crow search algorithm (CSA) [71], and the horse herd optimization algorithm (HOA) [45]. The algorithms' source codes were gathered from publicly available resources, and their parameter values were the same ones considered in the original papers, as reported in Table 2. Tables 5 and 6 compare the average and minimum fitness values produced by the proposed MFO-SFR algorithm and the other algorithms for 30 and 50 dimensions. The results of the test functions F_1 and F_3–F_{10} for both numbers of dimensions demonstrated that MFO-SFR exhibited impressive exploitation and exploration capabilities and generated better solutions while dealing with unimodal and multimodal tests. The results of test functions F_{11}–F_{30} demonstrated that the MFO-SFR avoided local optimum trapping and balanced the trade-off between exploration and exploitation abilities. Furthermore, the final two rows present the results of the Friedman test for each algorithm, in which MFO-SFR ranked first among the comparative algorithms for both 30 and 50 dimensions.

Table 5. Comparison of MFO-SFR with well-known algorithms for CEC 2018 test functions with D = 30.

F.	Metrics	PSO	KH	GWO	CSA	HOA	MFO-SFR
F_1	Avg	5.907×10^{10}	1.963×10^4	1.145×10^9	3.246×10^{10}	3.003×10^9	1.791×10^3
	Min	3.270×10^{10}	7.354×10^3	1.583×10^8	2.130×10^{10}	2.203×10^9	1.017×10^2
F_3	Avg	1.308×10^5	4.403×10^4	2.987×10^4	9.600×10^4	3.075×10^4	1.312×10^4
	Min	1.039×10^5	2.051×10^4	1.476×10^4	5.473×10^4	2.032×10^4	7.513×10^3
F_4	Avg	1.183×10^4	4.965×10^2	5.369×10^2	5.726×10^3	1.047×10^3	4.914×10^2
	Min	7.302×10^3	4.041×10^2	4.933×10^2	3.185×10^3	8.889×10^2	4.700×10^2
F_5	Avg	9.635×10^2	6.454×10^2	5.950×10^2	8.509×10^2	7.938×10^2	5.227×10^2
	Min	8.845×10^2	6.115×10^2	5.511×10^2	8.017×10^2	7.612×10^2	5.109×10^2
F_6	Avg	6.921×10^2	6.418×10^2	6.047×10^2	6.683×10^2	6.605×10^2	6.000×10^2
	Min	6.828×10^2	6.303×10^2	6.009×10^2	6.554×10^2	6.496×10^2	6.000×10^2
F_7	Avg	2.511×10^3	8.400×10^2	8.512×10^2	1.730×10^3	1.029×10^3	7.669×10^2
	Min	2.201×10^3	7.960×10^2	7.932×10^2	1.552×10^3	9.979×10^2	7.460×10^2
F_8	Avg	1.220×10^3	9.054×10^2	8.709×10^2	1.134×10^3	1.061×10^3	8.209×10^2
	Min	1.163×10^3	8.647×10^2	8.450×10^2	1.105×10^3	1.041×10^3	8.090×10^2
F_9	Avg	1.735×10^4	3.138×10^3	1.360×10^3	1.040×10^4	4.292×10^3	9.038×10^2
	Min	1.271×10^4	2.368×10^3	9.830×10^2	7.223×10^3	2.668×10^3	9.005×10^2
F_{10}	Avg	8.218×10^3	4.797×10^3	3.874×10^3	8.279×10^3	8.340×10^3	4.062×10^3
	Min	7.661×10^3	3.165×10^3	3.030×10^3	7.738×10^3	7.745×10^3	2.461×10^3
F_{11}	Avg	1.018×10^4	1.711×10^3	1.408×10^3	4.700×10^3	1.797×10^3	1.143×10^3
	Min	7.488×10^3	1.304×10^3	1.236×10^3	3.395×10^3	1.699×10^3	1.107×10^3
F_{12}	Avg	6.824×10^9	2.220×10^6	3.441×10^7	2.979×10^9	3.763×10^8	1.508×10^5
	Min	3.870×10^9	7.468×10^5	2.122×10^6	1.516×10^9	2.821×10^8	2.035×10^4
F_{13}	Avg	3.156×10^9	3.457×10^4	1.505×10^6	9.478×10^8	1.051×10^8	6.405×10^3
	Min	5.760×10^8	1.430×10^4	4.674×10^4	5.211×10^8	3.296×10^7	1.690×10^3
F_{14}	Avg	7.227×10^5	2.910×10^5	1.926×10^5	4.342×10^5	1.340×10^5	8.200×10^3
	Min	1.041×10^5	1.873×10^4	2.446×10^4	1.482×10^5	5.216×10^4	2.021×10^3
F_{15}	Avg	2.025×10^8	1.788×10^4	1.956×10^5	7.286×10^7	3.143×10^7	5.614×10^3
	Min	1.064×10^7	9.433×10^3	1.435×10^4	2.378×10^7	8.141×10^6	1.515×10^3

Table 5. *Cont.*

F.	Metrics	PSO	KH	GWO	CSA	HOA	MFO-SFR
F_{16}	Avg	4.452×10^3	2.884×10^3	2.385×10^3	3.989×10^3	3.786×10^3	1.855×10^3
	Min	3.827×10^3	2.377×10^3	1.949×10^3	3.147×10^3	3.419×10^3	1.617×10^3
F_{17}	Avg	3.298×10^3	2.277×10^3	1.943×10^3	2.628×10^3	2.361×10^3	1.745×10^3
	Min	2.755×10^3	1.804×10^3	1.778×10^3	2.256×10^3	2.102×10^3	1.727×10^3
F_{18}	Avg	6.473×10^6	4.258×10^5	8.049×10^5	7.890×10^6	1.212×10^6	1.493×10^5
	Min	6.593×10^5	4.192×10^4	6.880×10^4	2.022×10^6	4.281×10^5	4.305×10^4
F_{19}	Avg	2.509×10^8	9.593×10^4	7.374×10^5	1.358×10^8	4.426×10^7	6.534×10^3
	Min	3.341×10^7	1.081×10^4	3.279×10^3	6.263×10^7	1.774×10^7	1.910×10^3
F_{20}	Avg	2.847×10^3	2.624×10^3	2.362×10^3	2.759×10^3	2.681×10^3	2.091×10^3
	Min	2.574×10^3	2.303×10^3	2.146×10^3	2.476×10^3	2.487×10^3	2.004×10^3
F_{21}	Avg	2.707×10^3	2.416×10^3	2.379×10^3	2.625×10^3	2.575×10^3	2.321×10^3
	Min	2.615×10^3	2.359×10^3	2.351×10^3	2.587×10^3	2.539×10^3	2.312×10^3
F_{22}	Avg	8.759×10^3	3.018×10^3	4.411×10^3	6.831×10^3	4.513×10^3	2.300×10^3
	Min	6.900×10^3	2.300×10^3	2.406×10^3	5.558×10^3	2.705×10^3	2.300×10^3
F_{23}	Avg	3.239×10^3	2.880×10^3	2.729×10^3	3.143×10^3	3.133×10^3	2.671×10^3
	Min	3.101×10^3	2.807×10^3	2.678×10^3	3.061×10^3	3.059×10^3	2.654×10^3
F_{24}	Avg	3.539×10^3	3.107×10^3	2.890×10^3	3.319×10^3	3.188×10^3	2.844×10^3
	Min	3.253×10^3	2.994×10^3	2.849×10^3	3.206×10^3	3.122×10^3	2.828×10^3
F_{25}	Avg	7.655×10^3	2.911×10^3	2.958×10^3	4.890×10^3	3.137×10^3	2.887×10^3
	Min	5.843×10^3	2.887×10^3	2.916×10^3	4.344×10^3	3.069×10^3	2.887×10^3
F_{26}	Avg	8.709×10^3	5.651×10^3	4.483×10^3	8.661×10^3	4.738×10^3	3.903×10^3
	Min	6.500×10^3	2.800×10^3	3.473×10^3	7.772×10^3	3.736×10^3	3.739×10^3
F_{27}	Avg	3.827×10^3	3.400×10^3	3.230×10^3	3.690×10^3	3.720×10^3	3.219×10^3
	Min	3.591×10^3	3.283×10^3	3.212×10^3	3.537×10^3	3.616×10^3	3.208×10^3
F_{28}	Avg	6.851×10^3	3.228×10^3	3.356×10^3	5.474×10^3	3.519×10^3	3.216×10^3
	Min	5.611×10^3	3.198×10^3	3.283×10^3	4.541×10^3	3.468×10^3	3.196×10^3
F_{29}	Avg	5.426×10^3	4.194×10^3	3.642×10^3	5.253×10^3	4.726×10^3	3.414×10^3
	Min	4.907×10^3	3.858×10^3	3.439×10^3	4.952×10^3	4.454×10^3	3.323×10^3
F_{30}	Avg	2.946×10^8	1.043×10^6	2.842×10^6	1.084×10^8	2.610×10^7	7.835×10^3
	Min	8.913×10^7	8.260×10^4	5.249×10^5	4.274×10^7	1.221×10^7	6.362×10^3
Average rank		5.84	2.68	2.45	5.00	3.93	1.09
Total rank		6	3	2	5	4	1

Table 6. Comparison of MFO-SFR with well-known algorithms for CEC 2018 test functions with D = 50.

F.	Metrics	PSO	KH	GWO	CSA	HOA	MFO-SFR
F_1	Avg	1.526×10^{11}	1.703×10^5	4.506×10^9	9.609×10^{10}	1.149×10^{10}	3.463×10^4
	Min	8.451×10^{10}	1.932×10^4	7.183×10^8	8.123×10^{10}	8.346×10^9	9.385×10^3
F_3	Avg	2.549×10^5	1.192×10^5	7.730×10^4	2.070×10^5	8.230×10^4	5.495×10^4
	Min	1.957×10^5	7.643×10^4	4.662×10^4	1.774×10^5	6.990×10^4	4.223×10^4
F_4	Avg	3.307×10^4	5.428×10^2	8.017×10^2	1.712×10^4	2.589×10^3	5.867×10^2
	Min	1.811×10^4	4.765×10^2	6.224×10^2	1.286×10^4	2.058×10^3	5.196×10^2
F_5	Avg	1.367×10^3	7.662×10^2	6.775×10^2	1.211×10^3	1.047×10^3	5.624×10^2
	Min	1.256×10^3	7.090×10^2	6.316×10^2	1.145×10^3	1.011×10^3	5.318×10^2

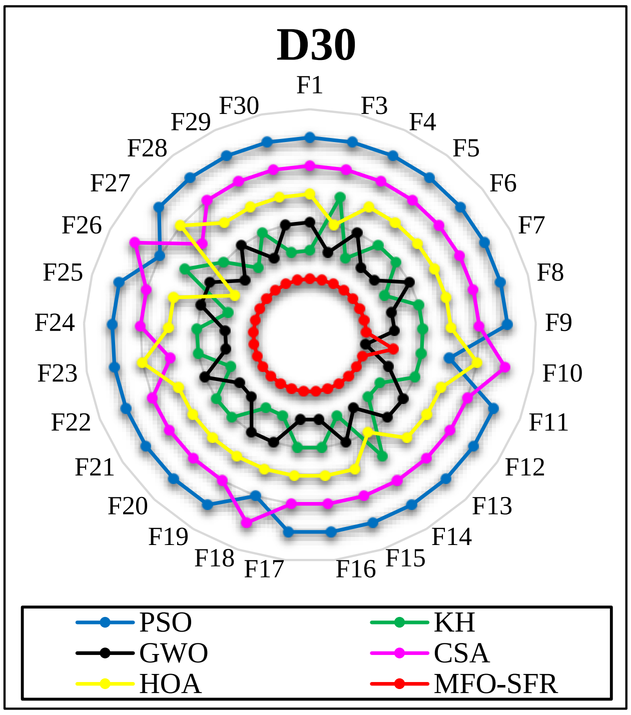
Table 6. *Cont.*

F.	Metrics	PSO	KH	GWO	CSA	HOA	MFO-SFR
F_6	Avg	7.114×10^2	6.499×10^2	6.112×10^2	6.862×10^2	6.745×10^2	6.001×10^2
	Min	6.991×10^2	6.393×10^2	6.075×10^2	6.776×10^2	6.645×10^2	6.000×10^2
F_7	Avg	4.585×10^3	1.052×10^3	9.899×10^2	3.206×10^3	1.338×10^3	8.682×10^2
	Min	4.142×10^3	9.353×10^2	9.057×10^2	2.735×10^3	1.289×10^3	8.099×10^2
F_8	Avg	1.687×10^3	1.059×10^3	9.862×10^2	1.499×10^3	1.354×10^3	8.610×10^2
	Min	1.575×10^3	1.033×10^3	9.423×10^2	1.423×10^3	1.283×10^3	8.318×10^2
F_9	Avg	5.170×10^4	9.873×10^3	4.844×10^3	3.622×10^4	2.153×10^4	9.243×10^2
	Min	3.909×10^4	7.955×10^3	2.552×10^3	3.021×10^4	1.416×10^4	9.066×10^2
F_{10}	Avg	1.441×10^4	7.948×10^3	5.919×10^3	1.429×10^4	1.412×10^4	6.534×10^3
	Min	1.356×10^4	6.701×10^3	4.473×10^3	1.342×10^4	1.324×10^4	5.135×10^3
F_{11}	Avg	2.610×10^4	4.813×10^3	2.766×10^3	1.573×10^4	3.782×10^3	1.259×10^3
	Min	1.947×10^4	3.034×10^3	1.652×10^3	1.157×10^4	3.239×10^3	1.146×10^3
F_{12}	Avg	4.300×10^{10}	1.287×10^7	3.406×10^8	2.210×10^{10}	2.417×10^9	1.873×10^6
	Min	2.464×10^{10}	4.289×10^6	3.715×10^7	1.421×10^{10}	1.705×10^9	1.078×10^6
F_{13}	Avg	1.683×10^{10}	5.864×10^4	1.026×10^8	6.132×10^9	5.696×10^8	5.362×10^3
	Min	5.672×10^9	2.099×10^4	8.150×10^4	3.709×10^9	4.310×10^8	1.749×10^3
F_{14}	Avg	6.142×10^6	5.701×10^5	3.453×10^5	4.140×10^6	8.316×10^5	4.016×10^4
	Min	1.659×10^6	1.615×10^5	2.928×10^4	1.829×10^6	2.222×10^5	1.202×10^4
F_{15}	Avg	5.029×10^9	1.956×10^4	4.078×10^6	1.190×10^9	2.294×10^8	2.964×10^3
	Min	2.307×10^9	9.499×10^3	2.722×10^4	4.163×10^8	9.908×10^7	1.534×10^3
F_{16}	Avg	7.306×10^3	3.250×10^3	2.896×10^3	6.252×10^3	5.184×10^3	2.614×10^3
	Min	6.699×10^3	2.463×10^3	2.326×10^3	5.731×10^3	4.749×10^3	2.148×10^3
F_{17}	Avg	1.334×10^4	3.359×10^3	2.661×10^3	5.190×10^3	3.888×10^3	2.474×10^3
	Min	5.938×10^3	2.849×10^3	2.264×10^3	4.547×10^3	3.183×10^3	2.018×10^3
F_{18}	Avg	3.800×10^7	2.330×10^6	3.051×10^6	3.471×10^7	8.478×10^6	1.247×10^6
	Min	1.430×10^7	1.131×10^6	3.848×10^5	1.224×10^7	4.500×10^6	1.067×10^5
F_{19}	Avg	2.060×10^9	1.719×10^5	1.231×10^6	5.231×10^8	7.924×10^7	1.276×10^4
	Min	6.897×10^8	2.586×10^4	9.261×10^3	2.060×10^8	2.979×10^7	2.447×10^3
F_{20}	Avg	4.010×10^3	3.276×10^3	2.743×10^3	3.903×10^3	3.675×10^3	2.485×10^3
	Min	3.712×10^3	2.764×10^3	2.380×10^3	3.680×10^3	3.261×10^3	2.081×10^3
F_{21}	Avg	3.164×10^3	2.556×10^3	2.473×10^3	2.994×10^3	2.850×10^3	2.360×10^3
	Min	3.046×10^3	2.455×10^3	2.422×10^3	2.896×10^3	2.767×10^3	2.335×10^3
F_{22}	Avg	1.609×10^4	1.049×10^4	8.211×10^3	1.603×10^4	1.527×10^4	8.339×10^3
	Min	1.492×10^4	9.115×10^3	6.990×10^3	1.493×10^4	4.474×10^3	6.317×10^3
F_{23}	Avg	4.077×10^3	3.379×10^3	2.916×10^3	3.777×10^3	3.749×10^3	2.793×10^3
	Min	3.763×10^3	3.052×10^3	2.826×10^3	3.595×10^3	3.533×10^3	2.761×10^3
F_{24}	Avg	4.174×10^3	3.643×10^3	3.109×10^3	3.983×10^3	3.744×10^3	2.970×10^3
	Min	3.943×10^3	3.406×10^3	2.991×10^3	3.738×10^3	3.597×10^3	2.931×10^3
F_{25}	Avg	2.556×10^4	3.092×10^3	3.355×10^3	1.580×10^4	4.331×10^3	3.070×10^3
	Min	1.693×10^4	3.052×10^3	3.146×10^3	1.384×10^4	3.997×10^3	2.985×10^3
F_{26}	Avg	1.697×10^4	9.335×10^3	5.804×10^3	1.506×10^4	5.800×10^3	4.406×10^3
	Min	1.280×10^4	3.159×10^3	4.993×10^3	1.326×10^4	5.170×10^3	4.051×10^3
F_{27}	Avg	5.456×10^3	4.354×10^3	3.516×10^3	5.185×10^3	5.049×10^3	3.307×10^3
	Min	4.582×10^3	3.985×10^3	3.402×10^3	4.636×10^3	4.657×10^3	3.266×10^3
F_{28}	Avg	1.242×10^4	3.346×10^3	3.927×10^3	1.039×10^4	4.740×10^3	3.378×10^3
	Min	9.606×10^3	3.311×10^3	3.545×10^3	8.991×10^3	4.469×10^3	3.310×10^3

Table 6. *Cont.*

F.	Metrics	PSO	KH	GWO	CSA	HOA	MFO-SFR
F_{29}	Avg	1.018×10^4	5.360×10^3	4.202×10^3	8.981×10^3	6.514×10^3	3.545×10^3
	Min	8.653×10^3	4.196×10^3	3.826×10^3	7.451×10^3	6.113×10^3	3.289×10^3
F_{30}	Avg	2.508×10^9	4.241×10^7	7.032×10^7	1.287×10^9	3.337×10^8	1.144×10^6
	Min	1.281×10^9	1.429×10^7	3.629×10^7	6.453×10^8	2.374×10^8	9.567×10^5
Average rank		5.93	2.70	2.29	4.98	3.92	1.18
Total rank		6	3	2	5	4	1

The exploratory data analysis shown in Figure 3 was conducted to investigate the ranking of algorithms for each test function. Overall, it can be noted that the proposed MFO-SFR algorithm was ranked first among the other compared algorithms for all test functions, except for F_{10} in 30 dimensions. For 50 dimensions, it is notable that MFO-SFR was ranked first for all test functions except for F_4, F_{10}, F_{22}, and F_{28}.



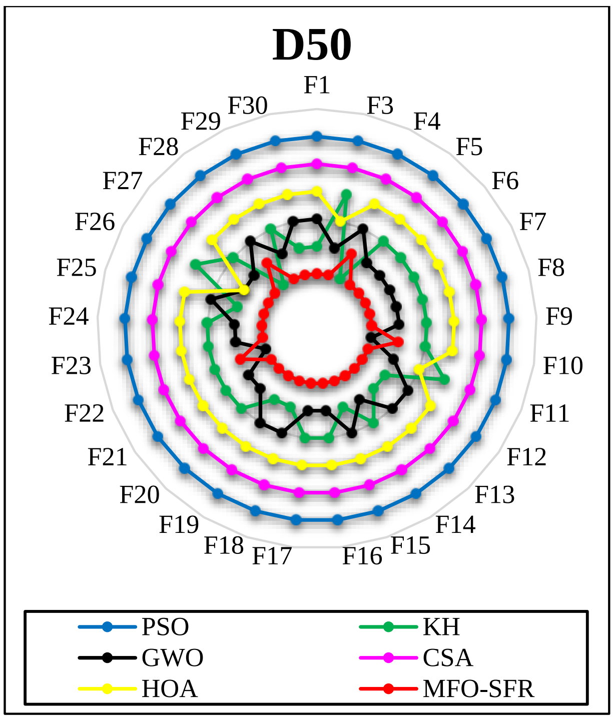


Figure 3. Exploratory data analysis of MFO-SFR and other well-known MFO algorithms on CEC 2018 with 30 and 50 dimensions.

As shown in Figure 4, we analyzed MFO-SFR's convergence behavior and compared it with that of the other algorithms. Overall, it can be seen that the proposed MFO-SFR algorithm was able to converge toward more accurate solutions by avoiding local optimum solutions and striking a balance between its search abilities. It is also notable that the proposed MFO-SFR algorithm maintained its solution accuracy by enhancing the number of dimensions, which demonstrates the scalability of the proposed algorithm.

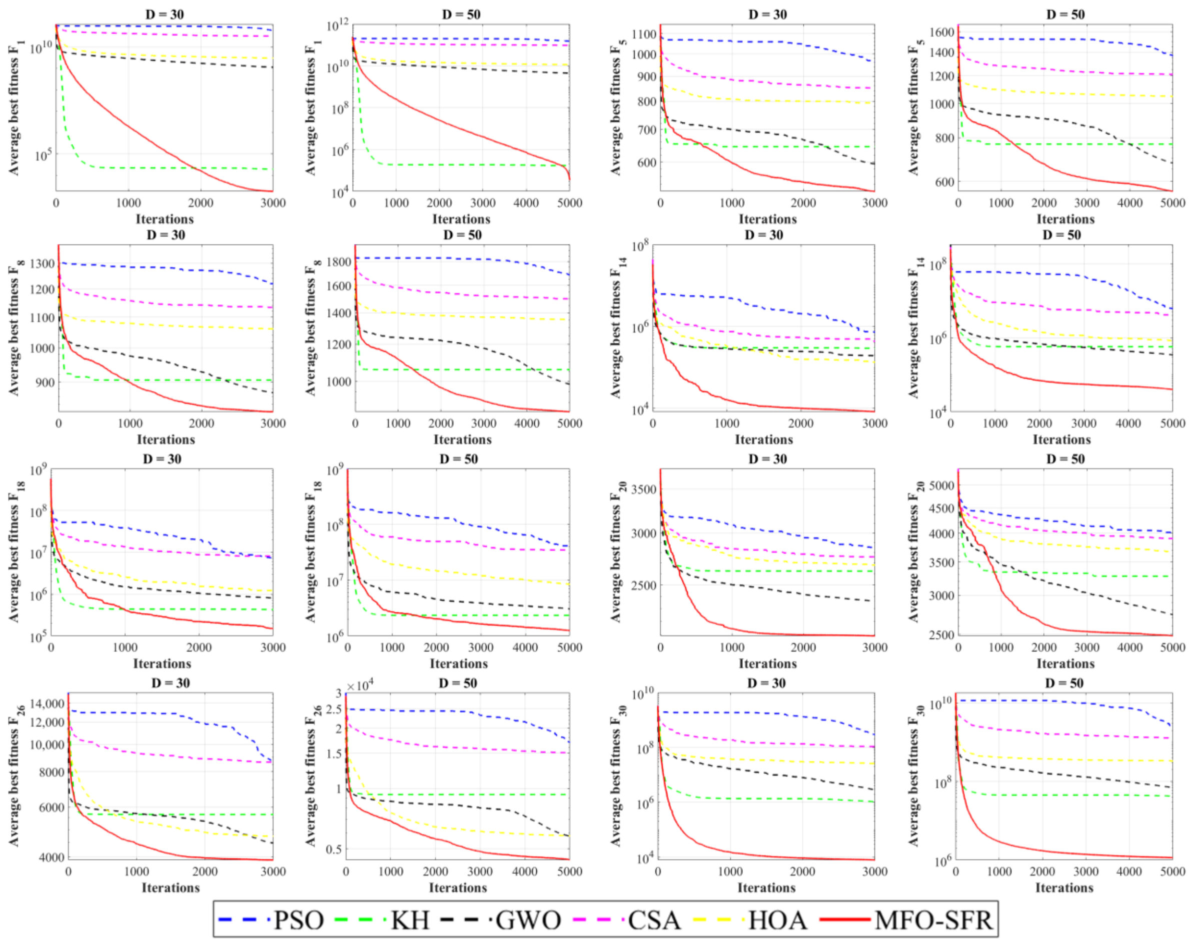


Figure 4. Comparison of the convergence behavior of MFO-SFR and well-known algorithms for CEC 2018 test functions with 30 and 50 dimensions.

5.3. Population Diversity Analysis

Maintaining population diversity is essential in metaheuristic algorithms since low diversity among search agents may cause the algorithm to become stuck at local optimum areas. In this experiment, the population diversity of MFO-SFR and five representatives of comparative algorithms was investigated on several CEC 2018 benchmark test suites with 30 and 50 dimensions. The population diversity curves presented in Figure 5 were calculated by measuring the moment of inertia (I_c) [99], where I_c denotes the spreading of each individual from their centroid, which was determined by Equation (14), and the centroid c_j for j = 1, 2, ... D was calculated using Equation (15). Comparing the population diversity curves with the convergence curves plotted in Figures 2 and 4, it can be noted that the proposed MFO-SFR algorithm effectively maintained diversification among solutions until the near-optimal solution was met. This behavior occurred mainly because of the introduced SFR strategy, which identified stagnant solutions using a distance-based technique and replaced them with a solution selected from the archive constructed from the previous solutions. The introduced archive was able to maintain not only the diversification of solutions by preserving the generated representative flame but also the convergence of solutions toward promising areas by preserving the best solutions in each iteration.

$$I_c = \sum_{i=1}^{D}\sum_{j=1}^{N}\left(M_{ij} - c_i\right)^2 \tag{14}$$

$$c_i = \frac{1}{N}\sum_{j=1}^{N} M_{ij} \tag{15}$$

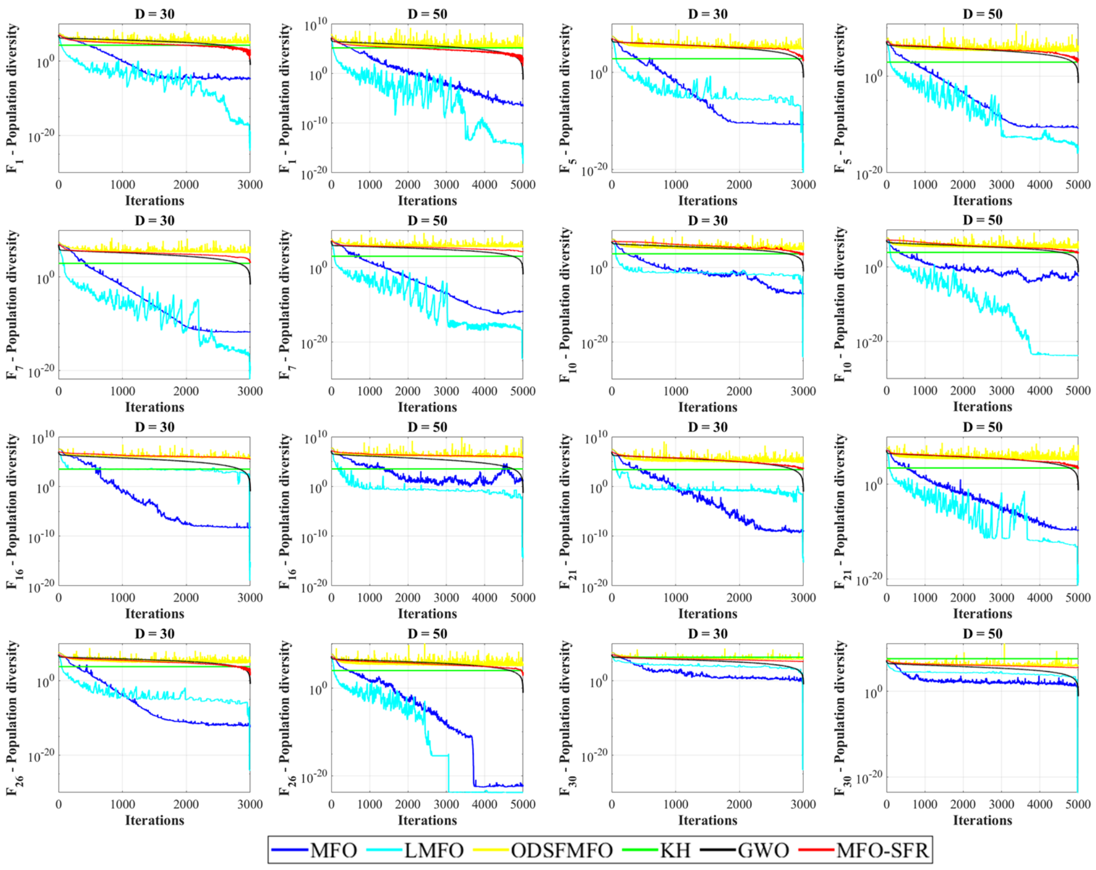


Figure 5. Population diversity of MFO-SFR and comparative algorithms on CEC 2018 test functions.

5.4. The Overall Effectiveness of MFO-SFR

The overall effectiveness (OE) achieved by the proposed MFO-SFR algorithm in solving test functions with 30 and 50 dimensions was computed using Equation (16) and the results are reported in Tables 7 and 8. OE_i indicates the overall effectiveness of the i-th algorithm, L_i is the total number of test functions that the i-th algorithm lost, and TF is the total number of test functions. Table 7 compares the OE achieved by the proposed MFO-SFR with the other MFO variants, showing that MFO-SFR attained the highest OE value, equal to 74.14%. Moreover, Table 8 shows that MFO-SFR achieved a higher OE value of 91.38% compared to other well-known optimization algorithms.

$$OE_i(\%) = \frac{TF - L_i}{TF} \tag{16}$$

Table 7. The overall effectiveness of MFO-SFR and MFO variants.

Algorithms	**30 (W | T | L)**	**50 (W | T | L)**	**Total (W | T | L)**	**OE**
MFO	0 | 0 | 29	0 | 0 | 29	0 | 0 | 58	0%
LMFO	0 | 0 | 29	0 | 0 | 29	0 | 0 | 58	0%
WCMFO	1 | 0 | 28	2 | 0 | 27	3 | 0 | 55	5.17%
CMFO	0 | 0 | 29	0 | 0 | 29	0 | 0 | 58	0%
ODSFMFO	1 | 0 | 28	1 | 0 | 28	2 | 0 | 56	3.45%
SMFO	0 | 0 | 29	0 | 0 | 29	0 | 0 | 58	0%
WMFO	4 | 0 | 25	6 | 0 | 23	10 | 0 | 48	17.24%
MFO-SFR	23 | 0 | 6	20 | 0 | 9	43 | 0 | 15	74.14%

Table 8. The overall effectiveness of MFO-SFR and contender algorithms.

Algorithms	30 (W | T | L)	50 (W | T | L)	Total (W | T | L)	OE
PSO	0 | 0 | 29	0 | 0 | 29	0 | 0 | 58	0%
KH	0 | 0 | 29	2 | 0 | 27	2 | 0 | 56	3.45%
GWO	1 | 0 | 28	2 | 0 | 27	3 | 0 | 55	5.17%
CSA	0 | 0 | 29	0 | 0 | 29	0 | 0 | 58	0%
HOA	0 | 0 | 29	0 | 0 | 29	0 | 0 | 58	0%
MFO-SFR	28 | 0 | 1	25 | 0 | 4	53 | 0 | 5	91.38%

6. Applicability of MFO-SFR to Solving Mechanical Engineering Problems

There is a growing interest in using optimization algorithms in mechanical and engineering systems to improve performance, cost, and product lifespan [50,100]. Therefore, in this section we assessed the applicability of MFO-SFR using two challenging real-world optimization issues from the most recent CEC 2020 test suite [72]. The constraints of the problems were handled using a death penalty function. The maximum number of iterations for MFO-SFR and the variants of MFO was $(D \times 10^4)/N$, where D is the number of decision variables and N is the number of search agents, which was set to 20.

6.1. Welded Beam Design (WBD) Problem

The WBD [101], stated in Equation (17), is a well-known optimization issue in constrained engineering problems. The primary goal of this task, as indicated in Figure 6, is to minimize the total fabrication cost of a welded beam by determining the best design parameters for the clamped bar length (l), weld thickness (h), bar thickness (b), and bar height (t). The results tabulated in Table 9 indicate that the proposed MFO-SFR exhibited superior performance compared with the other algorithms.

$$\begin{array}{ll}
\text{Consider} & \vec{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b], \\
\text{Min} & f\left(\vec{x}\right) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2), \\
\text{Subject to} & g_1\left(\vec{x}\right) = \tau\left(\vec{x}\right) - \tau_{max} \le 0, \\
& g_2\left(\vec{x}\right) = \sigma\left(\vec{x}\right) - \sigma_{max} \le 0, \\
& g_3\left(\vec{x}\right) = x_1 - x_4 \le 0, \\
& g_4\left(\vec{x}\right) = 1.10471 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \le 0, \\
& g_5\left(\vec{x}\right) = 0.125 - x_1 \le 0, \\
& g_6\left(\vec{x}\right) = \delta\left(\vec{x}\right) - \delta_{max} \le 0, \\
& g_7\left(\vec{x}\right) = P - P_c \le 0, \\
\text{Variable range} & 0.1 \le x_i \le 2, i = 1, 4, \\
& 0.1 \le x_i \le 10, i = 2, 3. \\
\text{where} & \tau\left(\vec{x}\right) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2}, \tau' = \frac{P}{\sqrt{2} x_1 x_2}, \tau'' = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right), \\
& R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \quad J = 2\left\{\sqrt{2} x_1 x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \sigma\left(\vec{x}\right) = \frac{6PL}{x_4 x_3^2}, \\
& \delta\left(\vec{x}\right) = \frac{6PL^3}{E x_3^2 x_4}, P_c\left(\vec{x}\right) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{E x_3^2 x_4}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), P = 6000 lb, L = 14 in., \\
& E = 30 \times 10^6 psi, G = 12 \times 10^6 psi, \tau_{max} = 13,600 psi, \sigma_{max} = 30,000 psi, \\
& \delta_{max} = 0.25\ in.
\end{array} \quad (17)$$

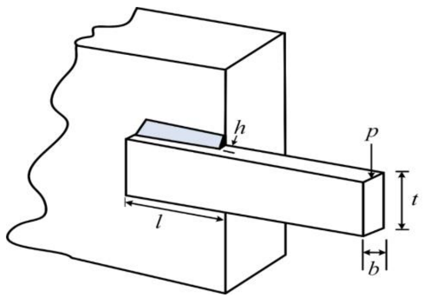


Figure 6. Schematic of the welded beam design problem [101].

Table 9. Comparison of the results obtained for the welded beam design problem.

Algorithms	**Optimal Values for Variables**				**Optimum Cost**
	h	***l***	***t***	***b***	
MFO	0.20576	3.47017	9.03582	0.20577	1.72499
LMFO	0.20739	3.43588	9.25131	0.20807	1.77799
WCMFO	0.20573	3.47035	9.03670	0.20573	1.72489
CMFO	0.18276	4.49027	8.98308	0.20819	1.82934
ODSFMFO	0.20569	3.47128	9.03662	0.20573	1.72490
SMFO	0.20836	3.46324	8.99144	0.20853	1.74135
WMFO	0.20722	3.45341	8.99834	0.20748	1.73151
MFO-SFR	0.20573	3.47056	9.03662	0.20573	1.72486

6.2. The Four-Stage Gearbox Problem

The design of a four-stage gearbox [102] was the second engineering design optimization problem examined in this study. To reduce the weight of the gearbox, the mathematical model specified in Equation (18) was used, together with 86 non-linear constraints and 22 discrete decision variables. According to the results reported in Table 10, the MFO-SFR algorithm outperformed the other algorithms in terms of the quality of its solution.

$$\text{Minimize}: \ F\left(\bar{x}\right) = \left(\frac{\pi}{1000}\right)\sum_{i=1}^{4} \frac{b_i c_i^2\left(N_{pi}^2 + N_{gi}^2\right)}{\left(N_{pi} + N_{gi}\right)^2}, \ i = (1,2,3,4) \tag{18}$$

Table 10. Comparison of the results obtained for the four-stage gearbox problem.

Variables	**MFO**	**LMFO**	**WCMFO**	**CMFO**	**ODSFMFO**	**SMFO**	**WMFO**	**MFO-SFR**
x_1	21.9311	26.0390	13.2709	14.4470	19.9522	7.4896	11.6081	19.7537
x_2	56.2686	59.6014	38.0428	35.9087	42.7879	36.8603	36.5289	51.3053
x_3	6.5100	23.4280	47.0038	13.7051	16.7363	9.6905	32.5497	17.3633
x_4	21.2349	62.1487	64.7345	27.4891	34.3685	35.0066	48.2554	35.7343
x_5	17.8415	41.7704	6.9925	19.5000	15.1662	15.4976	18.6551	17.4063
x_6	37.9201	50.8023	16.3730	32.6744	27.1178	36.7195	35.1674	30.5308
x_7	23.1300	18.5328	16.3790	13.5190	22.4386	37.1973	15.6177	21.5071
x_8	27.5979	49.6525	33.9110	31.5029	56.4417	15.5455	38.4569	44.0394
x_9	0.5100	0.9809	0.7443	1.3021	0.9727	2.0702	1.4277	0.8917
x_{10}	3.3914	0.5964	0.8894	2.4919	0.8552	0.7589	1.3920	1.2137
x_{11}	0.5100	0.7173	4.2285	1.4999	1.2069	0.8063	0.9781	0.6197
x_{12}	0.5100	0.5100	1.2452	1.4996	0.9356	1.3444	1.0893	1.1821
x_{13}	7.8792	5.4735	6.3799	3.4649	2.3679	0.9250	1.6757	4.1070
x_{14}	5.9533	4.8956	5.5138	6.4560	5.1076	4.7899	5.7668	6.8698
x_{15}	5.3993	4.9521	5.8834	5.1835	5.0748	0.5867	5.6259	2.6797
x_{16}	6.9070	4.8722	2.7391	5.1577	4.1023	4.6470	5.0989	2.2119
x_{17}	6.7122	4.6078	5.4174	6.4936	5.1441	4.0477	4.6631	4.2234
x_{18}	7.5214	7.5663	8.3727	6.4751	5.1403	3.3638	3.5762	1.5645
x_{19}	4.7338	3.7869	3.5741	4.4972	3.9364	3.6366	3.7593	3.7526
x_{20}	5.3254	4.3966	3.3184	3.4597	4.8508	4.6303	3.5222	5.4510
x_{21}	4.8923	3.3528	5.7764	4.4278	2.8171	2.9203	2.6439	4.0416
x_{22}	5.6086	4.0724	4.8985	4.4763	2.7781	3.9267	5.6805	5.2131
Optimum Weight	7.2632×10^1	6.6228×10^1	2.1631×10^{12}	5.2157×10^1	3.6565×10^1	3.4380×10^{17}	2.6870×10^{14}	3.6555×10^1

Subject to:

$$g_1\left(\bar{x}\right)=\left(\frac{366000}{\pi\omega_1}+\frac{2c_1N_{p1}}{N_{pi}+N_{g1}}\right)\left(\frac{\left(N_{p1}+N_{g1}\right)^2}{4b_1c_1^2N_{p1}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0$$
$$g_2\left(\bar{x}\right)=\left(\frac{366000N_{g1}}{\pi\omega_1N_{p1}}+\frac{2c_2N_{p2}}{N_{p2}+N_{g2}}\right)\left(\frac{\left(N_{p2}+N_{g2}\right)^2}{4b_2c_2^2N_{p2}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0$$
$$g_3\left(\bar{x}\right)=\left(\frac{366000N_{g1}N_{g2}}{\pi\omega_1N_{p1}N_{p2}}+\frac{2c_3N_{p3}}{N_{p3}+N_{g3}}\right)\left(\frac{\left(N_{p3}+N_{g3}\right)^2}{4b_3c_3^2N_{p3}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0$$
$$g_4\left(\bar{x}\right)=\left(\frac{366000N_{g1}N_{g2}N_{g3}}{\pi\omega_1N_{p1}N_{p2}N_{p3}}+\frac{2c_4N_{p4}}{N_{p4}+N_{g4}}\right)\left(\frac{\left(N_{p4}+N_{g4}\right)^2}{4b_4c_4^2N_{p4}}\right)-\frac{\sigma_N J_R}{0.0167WK_0K_m}\le 0$$
$$g_5\left(\bar{x}\right)=\left(\frac{366000}{\pi\omega_1}+\frac{2c_1N_{p1}}{N_{p1}+N_{g1}}\right)\left(\frac{\left(N_{p1}+N_{g1}\right)^3}{4b_1c_1^2N_{g1}N_{p1}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{sin(\varnothing)cos(\varnothing)}{0.0334WK_0K_m}\right)\le 0$$
$$g_6\left(\bar{x}\right)=\left(\frac{366000N_{g1}}{\pi\omega_1N_{p1}}+\frac{2c_2N_{p2}}{N_{p2}+N_{g2}}\right)\left(\frac{\left(N_{p2}+N_{g2}\right)^3}{4b_2c_2^2N_{g2}N_{p2}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{sin(\varnothing)cos(\varnothing)}{0.0334WK_0K_m}\right)\le 0$$
$$g_7\left(\bar{x}\right)=\left(\frac{366000N_{g1}N_{g2}}{\pi\omega_1N_{p1}N_{p2}}+\frac{2c_3N_{p3}}{N_{p3}+N_{g3}}\right)\left(\frac{\left(N_{p3}+N_{g3}\right)^3}{4b_3c_3^2N_{g3}N_{p3}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{sin(\varnothing)cos(\varnothing)}{0.0334WK_0K_m}\right)\le 0$$
$$g_8\left(\bar{x}\right)=\left(\frac{366000N_{g1}N_{g2}N_{g3}}{\pi\omega_1N_{p1}N_{p2}N_{p3}}+\frac{2c_4N_{p4}}{N_{p4}+N_{g4}}\right)\left(\frac{\left(N_{p4}+N_{g4}\right)^3}{4b_4c_4^2N_{g4}N_{p4}^2}\right)-\left(\frac{\sigma_H}{C_p}\right)^2\left(\frac{sin(\varnothing)cos(\varnothing)}{0.0334WK_0K_m}\right)\le 0$$
$$g_{9\text{–}12}\left(\bar{x}\right)=-N_{pi}\sqrt{\frac{\sin^2(\varnothing)}{4}-\frac{1}{N_{pi}}+\left(\frac{1}{N_{pi}}\right)^2}+N_{gi}\sqrt{\frac{sin^2(\varnothing)}{4}-\frac{1}{N_{gi}}+\left(\frac{1}{N_{gi}}\right)^2}+$$
$$\frac{sin(\varnothing)\left(N_{pi}+N_{gi}\right)}{2}+CR_{min}\pi\cos(\varnothing)\le 0$$
$$g_{13\text{–}16}\left(\bar{x}\right)=d_{min}-\frac{2c_iN_{pi}}{N_{pi}+N_{gi}}\le 0$$
$$g_{17\text{–}20}\left(\bar{x}\right)=d_{min}-\frac{2c_iN_{gi}}{N_{pi}+N_{gi}}\le 0$$
$$g_{21}\left(\bar{x}\right)=x_{p1}+\left(\frac{\left(N_{p1}+2\right)c_1}{N_{p1}+N_{g1}}\right)-L_{max}\le 0$$
$$g_{22\text{–}24}\left(\bar{x}\right)=-L_{max}+\left(\frac{\left(N_{pi}+2\right)c_i}{N_{pi}+N_{gi}}\right)_{i=2,3,4}+x_{g(i-1)}\le 0$$
$$g_{25}\left(\bar{x}\right)=-x_{p1}+\left(\frac{\left(N_{p1}+2\right)c_1}{N_{p1}+N_{g1}}\right)\le 0$$
$$g_{26\text{–}28}\left(\bar{x}\right)=\left(\frac{\left(N_{pi}+2\right)c_i}{N_{pi}+N_{gi}}-x_{g(i-1)}\right)_{i=2,3,4}\le 0$$
$$g_{29}\left(\bar{x}\right)=y_{p1}+\frac{\left(N_{p1}+2\right)c_1}{N_{p1}+N_{g1}}-L_{max}\le 0$$
$$g_{30\text{–}32}\left(\bar{x}\right)=-L_{max}+\left(\frac{c_i\left(2+N_{pi}\right)}{N_{pi}+N_{gi}}-y_{g(i-1)}\right)_{i=2,3,4}\le 0$$
$$g_{33}\left(\bar{x}\right)=\frac{\left(2+N_{p1}\right)c_1}{N_{p1}+N_{g1}}-y_{p1}\le 0$$
$$g_{34\text{–}36}\left(\bar{x}\right)=\left(\frac{c_i\left(2+N_{pi}\right)}{N_{pi}+N_{gi}}-y_{g(i-1)}\right)_{i=2,3,4}\le 0$$
$$g_{37\text{–}40}\left(\bar{x}\right)=-L_{max}+\frac{\left(2+N_{gi}\right)c_1}{N_{pi}+N_{gi}}+x_{gi}\le 0$$
$$g_{41\text{–}44}\left(\bar{x}\right)=-x_{gi}+\left(\frac{\left(N_{gi}+2\right)c_i}{N_{pi}+N_{gi}}\right)+x_{gi}\le 0$$
$$g_{45\text{–}48}\left(\bar{x}\right)=-y_{gi}+\left(\frac{\left(N_{gi}+2\right)c_i}{N_{pi}+N_{gi}}\right)-L_{max}\le 0$$
$$g_{49\text{–}52}\left(\bar{x}\right)=-y_{gi}+\left(\frac{\left(N_{gi}+2\right)c_i}{N_{pi}+N_{gi}}\right)\le 0$$
$$g_{53\text{–}56}\left(\bar{x}\right)=(b_i-8.255)(b_i-5.715)(b_i-12.70)(-N_{pi}+0.945c_i-N_{gi})(-1)\le 0$$
$$g_{57\text{–}60}\left(\bar{x}\right)=(b_i-8.255)(b_i-3.175)(b_i-12.70)(-N_{pi}+0.646c_i-N_{gi})\le 0$$
$$g_{61\text{–}64}\left(\bar{x}\right)=(b_i-5.715)(b_i-3.175)(b_i-12.70)(-N_{pi}+0.504c_i-N_{gi})\le 0$$
$$g_{65\text{–}68}\left(\bar{x}\right)=(b_i-5.715)(b_i-3.175)(b_i-8.255)(0c_i-N_{gi}-N_{pi})\le 0$$

$$
\begin{aligned}
&g_{69\text{–}72}\left(\bar{x}\right) = (b_i - 8.255)(b_i - 5.715)(b_i - 12.70)(N_{gi} + N_{pi} - 1.812c_i) \le 0 \\
&g_{73\text{–}76}\left(\bar{x}\right) = (b_i - 8.255)(b_i - 3.175)(b_i - 12.70)(-0.945c_i + N_{pi} + N_{gi}) \le 0 \\
&g_{77\text{–}80}\left(\bar{x}\right) = (b_i - 5.715)(b_i - 3.175)(b_i - 12.70)(-0.646c_i + N_{pi} + N_{gi})(-1) \le 0 \\
&g_{81\text{–}84}\left(\bar{x}\right) = (b_i - 5.715)(b_i - 3.175)(b_i - 8.255)(N_{pi} + N_{gi} - 0.504c_i) \le 0 \\
&g_{85}\left(\bar{x}\right) = \omega_{min} + \frac{\omega_1\left(N_{p1}N_{p2}N_{p3}N_{p4}\right)}{\left(N_{g1}N_{g2}N_{g3}N_{g4}\right)} \le 0 \\
&g_{86}\left(\bar{x}\right) = \frac{\omega_1\left(N_{p1}N_{p2}N_{p3}N_{p4}\right)}{\left(N_{g1}N_{g2}N_{g3}N_{g4}\right)} - \omega_{min} \le 0
\end{aligned}
\tag{1}
$$

where

$$
\begin{aligned}
&\bar{x} = \{N_{p1}, N_{g1}, N_{p2}, N_{g2} \ldots b_1, b_2 \ldots x_{p1}, x_{g1}, x_{g2} \ldots y_{p1}, y_{g1}, y_{g2} \ldots y_{g4}\} \\
&c_i = \sqrt{\left(y_{gi} - y_{pi}\right)^2 + \left(x_{gi} - x_{pi}\right)^2}, K_0 = 1.5, d_{min} = 25, J_R = 0.2, \varnothing = 120^\circ, W = 55.9, \\
&K_M = 1.6, CR_{min} = 1.4, \\
&L_{max} = 127, C_p = 464, \sigma_H = 3290, \omega_{max} = 255, \omega_1 = 5000, \sigma_N = 2090, \omega_{min} = 245.
\end{aligned}
$$

with bounds:

$$
\begin{aligned}
&b_1 \in \{3.175, 12.7, 8.255, 5.715\} \\
&y_{p1}, x_{p1}, y_{gi}, x_{gi} \in \{12.7, 38.1, 25.4, 50.8, 76.2, 63.5, 88.9, 114.3, 101.6\} \\
&7 \le N_{gi}, N_{pi} \le 76 \in integer.
\end{aligned}
$$

7. Conclusions and Future Works

MFO is a prominent metaheuristic algorithm, inspired by the nighttime convergent behavior of moths in relation to a light source. A large part of MFO's popularity in recent years has been attributed to its straightforward construction. However, due to its rapid loss of population diversity and inadequate exploration ability, the MFO algorithm often encounters local optimum entrapment and premature convergence. In this study, an enhanced moth-flame optimization (MFO-SFR) algorithm was proposed to tackle these weaknesses. MFO-SFR introduces an effective stagnation finding and replacing (SFR) strategy to effectively maintain population diversity by finding stagnant solutions using a distance-based technique and replacing them with a solution selected from the archive constructed on the basis of previous solutions.

The performance of the proposed MFO-SFR algorithm was evaluated on global optimization problems using the CEC 2018 benchmark test suite in two different sets of experiments. In the first set of experiments, the performance of MFO-SFR was benchmarked by conducting the CEC 2018 benchmark functions with 30 and 50 dimensions. The obtained results were compared to those obtained using MFO and its six recent variants, including Lévy-flight moth-flame optimization (LMFO), an efficient hybrid algorithm based on the water cycle and moth-flame (WCMFO), chaos-enhanced moth-flame optimization (CMFO), death mechanism-based moth-flame optimization (ODSFMFO), the synthesis of the moth-flame optimizer with sine cosine mechanisms (SMFO), and the hybrid of whale and moth-flame optimization (WMFO). In the second set of experiments, the results obtained using MFO-SFR were compared with the results of five well-known swarm intelligence algorithms, including particle swarm optimization (PSO), krill herd (KH), the grey wolf optimizer (GWO), the crow search algorithm (CSA), and the horse herd optimization algorithm (HOA) in 30 and 50 dimensions. Furthermore, the results of the two sets of experiments were statistically analyzed and ranked based on their average fitness values. To further analyze the performance of the proposed algorithms, convergence and population diversity results were plotted and compared with those of the other studied algorithms. The plotted curves showed that MFO-SFR could avoid premature convergence and local optimum solutions by maintaining its population diversity throughout the optimization process. To verify the viability of MFO-SFR in solving real-world optimization problems, two well-known mechanical engineering problems from the CEC 2020 dataset were considered. For future studies, solving the problem of improving the exploitation ability of MFO-SFR without degrading its exploration ability is a worthwhile direction of research. Furthermore, the SFR strategy could be considered as a reference in solving the issue of low population diversity for those metaheuristic algorithms that suffer from this problem. Moreover, alternative methods to construct an archive, such as history-based methods, as used in SHADE [103], can be investigated in future studies.

Author Contributions: Conceptualization, M.H.N.-S., A.F. and H.Z.; methodology, M.H.N.-S., A.F. and H.Z.; software, M.H.N.-S., A.F. and H.Z.; validation, M.H.N.-S., H.Z. and S.M.; formal analysis, M.H.N.-S., H.Z., A.F. and S.M.; investigation, M.H.N.-S., A.F. and H.Z.; resources, M.H.N.-S., H.Z. and S.M.; data curation, M.H.N.-S., A.F. and H.Z.; writing, M.H.N.-S., H.Z. and A.F.; original draft preparation, M.H.N.-S., A.F. and H.Z.; writing—review and editing, M.H.N.-S., H.Z., A.F. and S.M.; visualization, M.H.N.-S., A.F. and H.Z.; supervision, M.H.N.-S., H.Z. and S.M.; project administration, M.H.N.-S. and S.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data and code used in the research may be obtained from the corresponding author upon request.

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

Appendix A

Table A1 provides the results of the pretest conducted on the canonical MFO in 30 dimensions to investigate the average and maximum percentages of situations when φ_i was equal to 0.

Table A1. The analysis of situations where φ_i was equal to zero with D = 30.

#F	Max Percentage	Average Percentage	#F	Max Percentage	Average Percentage	#F	Max Percentage	Average Percentage
F_1	2.03	0.40	F_{12}	26.05	3.32	F_{22}	22.13	3.51
F_3	0.00	0.00	F_{13}	1.16	0.16	F_{23}	0.12	0.01
F_4	0.90	0.13	F_{14}	6.85	0.34	F_{24}	3.41	0.42
F_5	0.60	0.16	F_{15}	2.12	0.21	F_{25}	0.27	0.03
F_6	0.00	0.00	F_{16}	0.10	0.01	F_{26}	2.53	0.65
F_7	4.25	0.34	F_{17}	0.02	0.00	F_{27}	3.40	0.28
F_8	14.97	0.84	F_{18}	0.37	0.02	F_{28}	11.15	0.64
F_9	0.01	0.00	F_{19}	2.00	0.13	F_{29}	28.22	1.46
F_{10}	12.05	1.22	F_{20}	5.24	0.81	F_{30}	9.30	0.47
F_{11}	1.25	0.31	F_{21}	0.01	0.00			

References

1. Zabinsky, Z.B. Stochastic methods for practical global optimization. *J. Glob. Optim.* **1998**, *13*, 433–444. [CrossRef]
2. Pardalos, P.M.; Romeijn, H.E.; Tuy, H. Recent developments and trends in global optimization. *J. Comput. Appl. Math.* **2000**, *124*, 209–228. [CrossRef]
3. Hosseinzadeh, M.; Masdari, M.; Rahmani, A.M.; Mohammadi, M.; Aldalwie, A.H.M.; Majeed, M.K.; Karim, S.H.T. Improved butterfly optimization algorithm for data placement and scheduling in edge computing environments. *J. Grid Comput.* **2021**, *19*, 1–27. [CrossRef]
4. Hassan, B.A.; Rashid, T.A.; Mirjalili, S. Formal context reduction in deriving concept hierarchies from corpora using adaptive evolutionary clustering algorithm star. *Complex Intell. Syst.* **2021**, *7*, 2383–2398. [CrossRef]

5. Hassan, B.A. CSCF: A chaotic sine cosine firefly algorithm for practical application problems. *Neural Comput. Appl.* **2021**, *33*, 7011–7030. [CrossRef]
6. Yi, H.; Duan, Q.; Liao, T.W. Three improved hybrid metaheuristic algorithms for engineering design optimization. *Appl. Soft Comput.* **2013**, *13*, 2433–2444. [CrossRef]
7. Nadimi-Shahraki, M.H.; Asghari Varzaneh, Z.; Zamani, H.; Mirjalili, S. Binary Starling Murmuration Optimizer Algorithm to Select Effective Features from Medical Data. *Appl. Sci.* **2022**, *13*, 564. [CrossRef]
8. Piri, J.; Mohapatra, P.; Acharya, B.; Gharehchopogh, F.S.; Gerogiannis, V.C.; Kanavos, A.; Manika, S. Feature Selection Using Artificial Gorilla Troop Optimization for Biomedical Data: A Case Analysis with COVID-19 Data. *Mathematics* **2022**, *10*, 2742. [CrossRef]
9. Nadimi-Shahraki, M.H.; Fatahi, A.; Zamani, H.; Mirjalili, S. Binary Approaches of Quantum-Based Avian Navigation Optimizer to Select Effective Features from High-Dimensional Medical Data. *Mathematics* **2022**, *10*, 2770. [CrossRef]
10. Talbi, E.-G. *Metaheuristics: From Design to Implementation*; John Wiley & Sons: Hoboken, NJ, USA, 2009.
11. Siddiqi, U.F.; Shiraishi, Y.; Dahb, M.; Sait, S.M. A memory efficient stochastic evolution based algorithm for the multi-objective shortest path problem. *Appl. Soft Comput.* **2014**, *14*, 653–662. [CrossRef]
12. Kavoosi, M.; Dulebenets, M.A.; Abioye, O.; Pasha, J.; Theophilus, O.; Wang, H.; Kampmann, R.; Mikijeljević, M. Berth scheduling at marine container terminals: A universal island-based metaheuristic approach. *Marit. Bus. Rev.* **2019**, *5*, 30–66. [CrossRef]
13. Dokeroglu, T.; Sevinc, E.; Kucukyilmaz, T.; Cosar, A. A survey on new generation metaheuristic algorithms. *Comput. Ind. Eng.* **2019**, *137*, 106040. [CrossRef]
14. Agushaka, J.O.; Ezugwu, A.E. Initialisation Approaches for Population-Based Metaheuristic Algorithms: A Comprehensive Review. *Appl. Sci.* **2022**, *12*, 896. [CrossRef]
15. Singh, A.; Kumar, A. Applications of nature-inspired meta-heuristic algorithms: A survey. *Int. J. Adv. Intell. Paradig.* **2021**, *20*, 388–417. [CrossRef]
16. Fister Jr, I.; Yang, X.-S.; Fister, I.; Brest, J.; Fister, D. A brief review of nature-inspired algorithms for optimization. *arXiv* **2013**, arXiv:1307.4186.
17. Dehghani, M.; Mardaneh, M.; Malik, O.P.; NouraeiPour, S.M. DTO: Donkey theorem optimization. In Proceedings of the 2019 27th Iranian Conference on Electrical Engineering (ICEE), Yazd, Iran, 30 April–2 May 2019; pp. 1855–1859.
18. Fard, E.S.; Monfaredi, K.; Nadimi, M.H. An Area-Optimized Chip of Ant Colony Algorithm Design in Hardware Platform Using the Address-Based Method. *Int. J. Electr. Comput. Eng.* **2014**, *4*, 989–998. [CrossRef]
19. Holland, J.H. Genetic algorithms. *Sci. Am.* **1992**, *267*, 66–73. [CrossRef]
20. Storn, R.; Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
21. Beyer, H.-G.; Schwefel, H.-P. Evolution strategies–a comprehensive introduction. *Nat. Comput.* **2002**, *1*, 3–52. [CrossRef]
22. Jiao, L.; Li, Y.; Gong, M.; Zhang, X. Quantum-inspired immune clonal algorithm for global optimization. *IEEE Trans. Syst. Man Cybern. Part B* **2008**, *38*, 1234–1253. [CrossRef]
23. Lu, T.-C.; Juang, J.-C. Quantum-inspired space search algorithm (QSSA) for global numerical optimization. *Appl. Math. Comput.* **2011**, *218*, 2516–2532. [CrossRef]
24. Arpaia, P.; Maisto, D.; Manna, C. A Quantum-inspired Evolutionary Algorithm with a competitive variation operator for Multiple-Fault Diagnosis. *Appl. Soft Comput.* **2011**, *11*, 4655–4666. [CrossRef]
25. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
26. Hatamlou, A. Black hole: A new heuristic optimization approach for data clustering. *Inf. Sci.* **2013**, *222*, 175–184. [CrossRef]
27. Kashan, A.H. A new metaheuristic for optimization: Optics inspired optimization (OIO). *Comput. Oper. Res.* **2015**, *55*, 99–125. [CrossRef]
28. Atashpaz-Gargari, E.; Lucas, C. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 4661–4667.
29. Geem, Z.W.; Kim, J.H.; Loganathan, G.V. A new heuristic optimization algorithm: Harmony search. *Simulation* **2001**, *76*, 60–68. [CrossRef]
30. Rao, R.V.; Savsani, V.J.; Vakharia, D. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput. -Aided Des.* **2011**, *43*, 303–315. [CrossRef]
31. Shi, Y. Brain storm optimization algorithm. In Proceedings of the International Conference in Swarm Intelligence, Chongqing, China, 12–15 June 2011; pp. 303–309.
32. Moosavian, N.; Roodsari, B.K. Soccer league competition algorithm: A novel meta-heuristic algorithm for optimal design of water distribution networks. *Swarm Evol. Comput.* **2014**, *17*, 14–24. [CrossRef]
33. Moghdani, R.; Salimifard, K. Volleyball premier league algorithm. *Appl. Soft Comput.* **2018**, *64*, 161–185. [CrossRef]
34. Moosavi, S.H.S.; Bardsiri, V.K. Poor and rich optimization algorithm: A new human-based and multi populations algorithm. *Eng. Appl. Artif. Intell.* **2019**, *86*, 165–181. [CrossRef]
35. Naik, A.; Satapathy, S.C. Past present future: A new human-based algorithm for stochastic optimization. *Soft Comput.* **2021**, *25*, 12915–12976. [CrossRef]
36. Chakraborty, A.; Kar, A.K. Swarm intelligence: A review of algorithms. *Nat. -Inspired Comput. Optim.* **2017**, *10*, 475–494.

37. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. CCSA: Conscious neighborhood-based crow search algorithm for solving global optimization problems. *Appl. Soft Comput.* **2019**, *85*, 105583. [CrossRef]
38. Dorigo, M.; Birattari, M.; Stutzle, T. Ant colony optimization. *IEEE Comput. Intell. Mag.* **2006**, *1*, 28–39. [CrossRef]
39. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; pp. 1942–1948.
40. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
41. Gandomi, A.H.; Yang, X.-S.; Alavi, A.H. Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Eng. Comput.* **2013**, *29*, 17–35. [CrossRef]
42. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
43. Wang, G.-G.; Deb, S.; Coelho, L.d.S. Elephant herding optimization. In Proceedings of the 2015 3rd International Symposium on Computational and Business Intelligence (ISCBI), Bali, Indonesia, 7–9 December 2015; pp. 1–5.
44. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl. -Based Syst.* **2015**, *89*, 228–249. [CrossRef]
45. MiarNaeimi, F.; Azizyan, G.; Rashki, M. Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems. *Knowl. -Based Syst.* **2021**, *213*, 106711. [CrossRef]
46. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. QANA: Quantum-based avian navigation optimizer algorithm. *Eng. Appl. Artif. Intel.* **2021**, *104*, 104314. [CrossRef]
47. Abdollahzadeh, B.; Gharehchopogh, F.S.; Mirjalili, S. African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. *Comput. Ind. Eng.* **2021**, *158*, 107408. [CrossRef]
48. Shayanfar, H.; Gharehchopogh, F.S. Farmland fertility: A new metaheuristic algorithm for solving continuous optimization problems. *Appl. Soft Comput.* **2018**, *71*, 728–746. [CrossRef]
49. Agushaka, J.O.; Ezugwu, A.E.; Abualigah, L. Dwarf mongoose optimization algorithm. *Comput. Method Appl. M* **2022**, *391*, 114570. [CrossRef]
50. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. Starling murmuration optimizer: A novel bio-inspired algorithm for global and engineering optimization. *Comput. Methods Appl. Mech. Eng.* **2022**, *392*, 114616. [CrossRef]
51. Abdollahzadeh, B.; Soleimanian Gharehchopogh, F.; Mirjalili, S. Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. *Int. J. Intell. Syst.* **2021**, *36*, 5887–5958. [CrossRef]
52. Pandey, H.M.; Chaudhary, A.; Mehrotra, D. A comparative review of approaches to prevent premature convergence in GA. *Appl. Soft Comput.* **2014**, *24*, 1047–1077. [CrossRef]
53. Chaitanya, K.; Somayajulu, D.; Krishna, P.R. Memory-based approaches for eliminating premature convergence in particle swarm optimization. *Appl. Intell.* **2021**, *51*, 4575–4608. [CrossRef]
54. Zhu, G.; Kwong, S. Gbest-guided artificial bee colony algorithm for numerical function optimization. *Appl. Math. Comput.* **2010**, *217*, 3166–3173. [CrossRef]
55. Banharnsakun, A.; Achalakul, T.; Sirinaovakul, B. The best-so-far selection in artificial bee colony algorithm. *Appl. Soft Comput.* **2011**, *11*, 2888–2901. [CrossRef]
56. Xiang, W.-L.; Li, Y.-Z.; Meng, X.-L.; Zhang, C.-M.; An, M.-Q. A grey artificial bee colony algorithm. *Appl. Soft Comput.* **2017**, *60*, 1–17. [CrossRef]
57. Nadimi-Shahraki, M.H.; Zamani, H. DMDE: Diversity-maintained multi-trial vector differential evolution algorithm for non-decomposition large-scale global optimization. *Expert Syst. Appl.* **2022**, *198*, 116895. [CrossRef]
58. Wang, F.; Liao, X.; Fang, N.; Jiang, Z. Optimal Scheduling of Regional Combined Heat and Power System Based on Improved MFO Algorithm. *Energies* **2022**, *15*, 3410. [CrossRef]
59. Kaur, K.; Singh, U.; Salgotra, R. An enhanced moth flame optimization. *Neural Comput. Appl.* **2020**, *32*, 2315–2349. [CrossRef]
60. Li, Z.; Zhou, Y.; Zhang, S.; Song, J. Lévy-flight moth-flame algorithm for function optimization and engineering design problems. *Math. Probl. Eng.* **2016**, *2016*, 1–22. [CrossRef]
61. Khalilpourazari, S.; Khalilpourazary, S. An efficient hybrid algorithm based on Water Cycle and Moth-Flame Optimization algorithms for solving numerical and constrained engineering optimization problems. *Soft Comput.* **2019**, *23*, 1699–1722. [CrossRef]
62. Hongwei, L.; Jianyong, L.; Liang, C.; Jingbo, B.; Yangyang, S.; Kai, L. Chaos-enhanced moth-flame optimization algorithm for global optimization. *J. Syst. Eng. Electron.* **2019**, *30*, 1144–1159.
63. Chen, C.; Wang, X.; Yu, H.; Wang, M.; Chen, H. Dealing with multi-modality using synthesis of Moth-flame optimizer with sine cosine mechanisms. *Math. Comput. Simul.* **2021**, *188*, 291–318. [CrossRef]
64. Xu, Y.; Chen, H.; Luo, J.; Zhang, Q.; Jiao, S.; Zhang, X. Enhanced Moth-flame optimizer with mutation strategy for global optimization. *Inf. Sci.* **2019**, *492*, 181–203. [CrossRef]
65. Li, Z.; Zeng, J.; Chen, Y.; Ma, G.; Liu, G. Death mechanism-based moth–flame optimization with improved flame generation mechanism for global optimization tasks. *Expert Syst. Appl.* **2021**, *183*, 115436. [CrossRef]
66. Xu, Y.; Chen, H.; Heidari, A.A.; Luo, J.; Zhang, Q.; Zhao, X.; Li, C. An efficient chaotic mutative moth-flame-inspired optimizer for global optimization tasks. *Expert Syst. Appl.* **2019**, *129*, 135–155. [CrossRef]

67. Awad, N.; Ali, M.; Liang, J.; Qu, B.; Suganthan, P. *Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization*; Technical Report; Nanyang Technological University: Singapore, 2016.
68. Nadimi-Shahraki, M.H.; Fatahi, A.; Zamani, H.; Mirjalili, S.; Oliva, D. Hybridizing of Whale and Moth-Flame Optimization Algorithms to Solve Diverse Scales of Optimal Power Flow Problem. *Electronics* **2022**, *11*, 831. [CrossRef]
69. Gandomi, A.H.; Alavi, A.H. Krill herd: A new bio-inspired optimization algorithm. *Commun. Non-Linear Sci. Numer. Simul.* **2012**, *17*, 4831–4845. [CrossRef]
70. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
71. Askarzadeh, A. A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Comput. Struct.* **2016**, *169*, 1–12. [CrossRef]
72. Kumar, A.; Wu, G.; Ali, M.Z.; Mallipeddi, R.; Suganthan, P.N.; Das, S. A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. *Swarm Evol. Comput.* **2020**, *56*, 100693. [CrossRef]
73. Nadimi-Shahraki, M.H.; Fatahi, A.; Zamani, H.; Mirjalili, S.; Abualigah, L. An improved moth-flame optimization algorithm with adaptation mechanism to solve numerical and mechanical engineering problems. *Entropy* **2021**, *23*, 1637. [CrossRef] [PubMed]
74. Pelusi, D.; Mascella, R.; Tallini, L.; Nayak, J.; Naik, B.; Deng, Y. An Improved Moth-Flame Optimization algorithm with hybrid search phase. *Knowl. -Based Syst.* **2020**, *191*, 105277. [CrossRef]
75. Nadimi-Shahraki, M.H.; Fatahi, A.; Zamani, H.; Mirjalili, S.; Abualigah, L.; Abd Elaziz, M. Migration-based moth-flame optimization algorithm. *Processes* **2021**, *9*, 2276. [CrossRef]
76. Ma, L.; Wang, C.; Xie, N.-g.; Shi, M.; Ye, Y.; Wang, L. Moth-flame optimization algorithm based on diversity and mutation strategy. *Appl. Intell.* **2021**, *51*, 5836–5872. [CrossRef]
77. Zhao, X.; Fang, Y.; Liu, L.; Li, J.; Xu, M. An improved moth-flame optimization algorithm with orthogonal opposition-based learning and modified position updating mechanism of moths for global optimization problems. *Appl. Intell.* **2020**, *50*, 4434–4458. [CrossRef]
78. Sapre, S.; Mini, S. Opposition-based moth flame optimization with Cauchy mutation and evolutionary boundary constraint handling for global optimization. *Soft Comput.* **2019**, *23*, 6023–6041. [CrossRef]
79. Sahoo, S.K.; Saha, A.K.; Nama, S.; Masdari, M. An improved moth flame optimization algorithm based on modified dynamic opposite learning strategy. *Artif. Intell. Rev.* **2022**, 1–59. [CrossRef]
80. Li, C.; Niu, Z.; Song, Z.; Li, B.; Fan, J.; Liu, P.X. A double evolutionary learning moth-flame optimization for real-parameter global optimization problems. *IEEE Access* **2018**, *6*, 76700–76727. [CrossRef]
81. Li, Y.; Zhu, X.; Liu, J. An improved moth-flame optimization algorithm for engineering problems. *Symmetry* **2020**, *12*, 1234. [CrossRef]
82. Shehab, M.; Alshawabkah, H.; Abualigah, L.; AL-Madi, N. Enhanced a hybrid moth-flame optimization algorithm using new selection schemes. *Eng. Comput.* **2021**, *37*, 2931–2956. [CrossRef]
83. Zhang, H.; Li, R.; Cai, Z.; Gu, Z.; Heidari, A.A.; Wang, M.; Chen, H.; Chen, M. Advanced orthogonal moth flame optimization with Broyden–Fletcher–Goldfarb–Shanno algorithm: Framework and real-world problems. *Expert Syst. Appl.* **2020**, *159*, 113617. [CrossRef]
84. Yu, C.; Heidari, A.A.; Chen, H. A quantum-behaved simulated annealing algorithm-based moth-flame optimization method. *Appl. Math. Model.* **2020**, *87*, 1–19. [CrossRef]
85. Alzaqebah, M.; Alrefai, N.; Ahmed, E.A.; Jawarneh, S.; Alsmadi, M.K. Neighborhood search methods with moth optimization algorithm as a wrapper method for feature selection problems. *Int. J. Electr. Comput. Eng.* **2020**, *10*, 3672. [CrossRef]
86. Xu, L.; Li, Y.; Li, K.; Beng, G.H.; Jiang, Z.; Wang, C.; Liu, N. Enhanced moth-flame optimization based on cultural learning and Gaussian mutation. *J. Bionic Eng.* **2018**, *15*, 751–763. [CrossRef]
87. Helmi, A.; Alenany, A. An enhanced Moth-flame optimization algorithm for permutation-based problems. *Evol. Intell.* **2020**, *13*, 741–764. [CrossRef]
88. Sayed, G.I.; Hassanien, A.E. A hybrid SA-MFO algorithm for function optimization and engineering design problems. *Complex Intell. Syst.* **2018**, *4*, 195–212. [CrossRef]
89. Buch, H.; Trivedi, I.N.; Jangir, P. Moth flame optimization to solve optimal power flow with non-parametric statistical evaluation validation. *Cogent Eng.* **2017**, *4*, 1286731. [CrossRef]
90. Trivedi, I.N.; Jangir, P.; Parmar, S.A.; Jangir, N. Optimal power flow with voltage stability improvement and loss reduction in power system using Moth-Flame Optimizer. *Neural Comput. Appl.* **2018**, *30*, 1889–1904. [CrossRef]
91. Jangir, P.; Jangir, N. Optimal power flow using a hybrid particle Swarm optimizer with moth flame optimizer. *Glob. J. Res. Eng.* **2017**, *17*, 15–32.
92. Sahoo, S.K.; Saha, A.K. A hybrid moth flame optimization algorithm for global optimization. *J. Bionic Eng.* **2022**, *19*, 1522–1543. [CrossRef]
93. Khan, B.S.; Raja, M.A.Z.; Qamar, A.; Chaudhary, N.I. Design of moth flame optimization heuristics for integrated power plant system containing stochastic wind. *Appl. Soft Comput.* **2021**, *104*, 107193. [CrossRef]
94. Singh, P.; Bishnoi, S. Modified moth-Flame optimization for strategic integration of fuel cell in renewable active distribution network. *Electr. Power Syst. Res.* **2021**, *197*, 107323. [CrossRef]

95. Zhang, H.; Heidari, A.A.; Wang, M.; Zhang, L.; Chen, H.; Li, C. Orthogonal Nelder-Mead moth flame method for parameters identification of photovoltaic modules. *Energy Convers. Manag.* **2020**, *211*, 112764. [CrossRef]
96. Cui, Z.; Li, C.; Huang, J.; Wu, Y.; Zhang, L. An improved moth flame optimization algorithm for minimizing specific fuel consumption of variable cycle engine. *IEEE Access* **2020**, *8*, 142725–142735. [CrossRef]
97. Khurma, R.A.; Aljarah, I.; Sharieh, A. A simultaneous moth flame optimizer feature selection approach based on levy flight and selection operators for medical diagnosis. *Arab. J. Sci. Eng.* **2021**, *46*, 8415–8440. [CrossRef]
98. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
99. Morrison, R.W. *Designing Evolutionary Algorithms for Dynamic Environments*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 178.
100. Altabeeb, A.M.; Mohsen, A.M.; Abualigah, L.; Ghallab, A. Solving capacitated vehicle routing problem using cooperative firefly algorithm. *Appl. Soft Comput.* **2021**, *108*, 107403. [CrossRef]
101. Ragsdell, K.; Phillips, D. Optimal design of a class of welded structures using geometric programming. *Eng. Ind.* **1976**, *98*, 1021–1025. [CrossRef]
102. Yokota, T.; Taguchi, T.; Gen, M. A solution method for optimal weight design problem of the gear using genetic algorithms. *Comput. Ind. Eng.* **1998**, *35*, 523–526. [CrossRef]
103. Tanabe, R.; Fukunaga, A. Success-history based parameter adaptation for differential evolution. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013; pp. 71–78.