A Modernized Moth Flame Optimization Algorithm for Higher Dimensional Problems

Saroj Kumar Sahoo[1*], Apu Kumar Saha[2]

[1,2]*Department of Mathematics, National Institute of Technology Agartala, India*
**Corresponding author: Email: sarojlipu.gugle@gmail.com*

ABSTRACT
Moth flame optimization (MFO) algorithm is a relatively new nature-inspired optimization algorithm based on the moth's movement towards the moon. Premature convergence and convergence to local optima are the main demerits of the basic MFO algorithm. To avoid these drawbacks, a new variant of MFO algorithm, namely a modernized MFO (M-MFO) algorithm is presented in this paper. Firstly, we added a self-adaptive levy distribution method before the position update phase of the MFO algorithm to enhance the search region. Secondly, we introduce a new type of parameter to strike a better balance between diversification and intensification. Third, we incorporate the Fibonacci search technique into the MFO algorithm after the position update phase to get around the problem of local optimal solutions and speed up convergence. The proposed M-MFO is verified by testing it on fifteen benchmark functions in higher dimensions (1000, for example), undergoing statistical experiments, and solving engineering design problems, and then comparing the results to those obtained by using other, more conventional optimization algorithms. The experimental results show that the proposed M-MFO algorithm outperforms competing stochastic algorithms in terms of solution quality and convergence rate. This encourages further research into topics such as multi-objective optimization, vehicle routing, job shop planning, and image segmentation.

Keywords: *Moth Flame Optimization Algorithm, Swarm Intelligence, Fibonacci Search Method; Benchmark Functions.*

1. INTRODUCTION

The challenge of finding the better solution in optimization problems is an interesting topic of research due to its importance in both academia and industry. The difficulty of optimization issues grows in proportion to the number of available optimization factors. In recent decades, researchers are very much interested in machine learning and artificial intelligence techniques as real-life applications, which are either constrained or unconstrained, linear or nonlinear, continuous or discontinuous, can easily tackle by the help of these techniques [1]. Due to the aforesaid characteristics, there are various level of difficulties to handle such types of problems using conventional techniques with numerical or mathematical programming [2]. Many studies [3] have shown experimentally that traditional approaches cannot deal with non-continuous, non-differentiable, and practical multimodal problems. Due to their simplicity and widespread applicability, meta-heuristics algorithms have proven indispensable in the fight against these problems. Few of them are particle swarm optimization (PSO) [4], differential evolution (DE) [5], flower pollination algorithm (FPA) [6] butterfly optimization algorithm (BOA) [7], moth flame optimization (MFO) [8]**,** etc. These algorithms typically begin with some randomly selected set of initial solutions and iteratively refine those solutions until they yield the globally optimal solutions to the objective functions.

The MFO algorithm is the subject of this article. In 2015, Mirjalili discovered MFO, a swarm intelligence-based algorithm. Transverse orientation, used by moths to navigate in the wild, served as an inspiration for MFO. Spiral flight search (SFS) and simple flame generation (SFG) are two of the most important MFO tactics. In the SFG technique, the best moths and flames collected thus far can be used to directly manufacture flames. By following moths' transverse direction, the SFS technique allows moths to

spiral toward the flames to update their own positions in an iterative process. The key advantage of MFO above other standard algorithms is its ability to tackle several tough issues involving confined and unknown search spaces. Optical network unit placement [9], automatic generation control problem [10], image segmentation [11], feature selection [12], medical diagnoses [13] etc.

As a new population-based optimization method, MFO's performance still needs to be improved and studied in some dimensions, including convergence speed and global search capabilities. Various academics have come up with a variety of ways to fix the MFO algorithm's flaws, and a few of these are presented here. In order to tackle the shortcomings of the MFO method, Hongwei et al. [14] presented a new form of the algorithm called chaos-enhanced MFO, which incorporates a chaotic map. The authors [15] proposed a series of new MFO algorithm variants by integrating MFO with Gaussian mutation, Cauchy mutation, levy mutation, or a combination of the three mutations to reduce the disadvantages of the MFO algorithm and improve the basic MFO algorithm's capacity for diversification and intensification. To achieve a more stable balance between diversity and intensification in the MFO algorithm by embedding Gaussian mutation and chaotic local search, authors [16] developed CLSGMFO. Three additional adjustments were introduced to the MFO algorithm by Kaur et al. [17] to maintain a favorable trade-off between diversification and intensification, boost exploration and exploitation, respectively. For the prediction of software errors, Tumar et al. [18] implemented a modified MFO method and presented an extended binary moth-flame optimization algorithm (EBMFO). To strike a compromise between global and local search capabilities, Gu and Xiang [19] suggested a new modified MFO algorithm termed "multi operator MFO algorithm" (MOMFO). An updated version of the MFO algorithm was created by Ma and colleagues [20] to alleviate the MFO's shortcomings, including delayed convergence and convergence to a local minimum. A new version of the MFO, namely EMFO, based on the mutualism phase of symbiotic organism search, has been proposed by Sahoo et al [21]. Few researchers have developed modified method of MFO algorithm for higher dimensional problems such as Kaur et al. [17] proposed an upgraded version of the MFO algorithm named an enhanced moth flame optimization by using three modifications and its performance was checked over benchmark functions with five different dimensions like 30, 50, 100, 200 and 500. Yu et al. [22] proposed an improved version of MFO algorithm named a quantum-behaved simulated annealing (SA) enhanced MFO method (QSMFO) by integrating SA strategy into the exploitation phase and quantum rotation gate into the exploration phase of the basic MFO algorithm and evaluated the performance of the QSMFO by the help of applied on four benchmark functions over 30, 50 and 100 dimensions.

Many swarm intelligence (SI)-based algorithms achieves excellent results in dealing with lower dimensional (2-100) problems but relatively weak performance in case of higher dimensional problems as with the increase in dimensions, search area increases exponentially leads problem to be more complicated. But, in literature, there is hardly any study where the MFO has been utilized to solve these large-scale optimization problems. This fact has motivated us to develop an improved variant of MFO for large scale optimization problems. To achieve a good trade-off between diversification and intensification for higher dimensional problems, in this article, an upgraded version of MFO algorithm (M-MFO) with the help few modifications in the MFO algorithm is suggested. The significant contributions of the present work are summarized below:

A self-adaptive Levy mutation is used before the position update phase of MFO to enhance the searching efficacy, non-linear function is employed in MFO algorithm to handle global and local search of the suggested MFO algorithm and finally Fibonacci principle is used after the position update phase of MFO to enhance the solution quality. The proposed M-MFO has been applied to solve higher dimensional problems with 1000-dimensions and the results are compared with five well-known metaheuristics and two MFO variants. The Friedman rank test is used to measure the performance of the suggested M-MFO statistically and the suggested M-MFO has used to solve one real-world engineering design problem.

This article has been organized as follows: In Section 2, we provide a brief overview of the MFO algorithm. In Section 3, the proposed M-MFO algorithm is presented. In Section 4, we detail our experimental setups, our simulation results, our statistical tests, and how we've applied these to real-world problems. Finally, Sect. 6 discusses the implications for future research.

2. Classical MFO algorithm

This section presents the origin of the MFO algorithm and its working process with the mathematical formulation is presented below.

All moths can be expressed as a set of candidates in the simplest form of MFO. A vector of decision variables representing the location of all moths. Let's take a look at the following moth-specific matrix.

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n-1} & x_{1,n} \\ x_{2,1} & \ddots & \cdots & \cdots & x_{2,n} \\ \vdots & \cdots & \ddots & \cdots & \vdots \\ x_{N-1,1} & \cdots & \cdots & \ddots & x_{N-1,n} \\ x_{N,1} & x_{N,2} & \cdots & x_{N,n-1} & x_{N,n} \end{bmatrix} \quad (1)$$

where, $X_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,n}], i \in \{1, 2, \ldots, N\}$.
N indicates moths' number at initial population and n as variable numbers. The flame matrix (FMx) is the second key point of the MFO algorithm and presented below

$$FMx = \begin{bmatrix} FMx_1 \\ FMx_2 \\ \vdots \\ FMx_N \end{bmatrix} = \begin{bmatrix} Fm_{1,1} & Fm_{1,2} & \cdots & Fm_{1,n-1} & Fm_{1,n} \\ Fm_{2,1} & \ddots & \cdots & \cdots & Fm_{2,n} \\ \vdots & \cdots & \ddots & \cdots & \vdots \\ Fm_{N-1,1} & \cdots & \cdots & \ddots & Fm_{N-1,n} \\ Fm_{N,1} & Fm_{N,2} & \cdots & Fm_{N-1} & Fm_{N,n} \end{bmatrix} \quad (2)$$

As the moth moves in a spiral manner, therefore, the author of MFO has defined a spiral function which is represented in the following equation:

$$x_i^{K+1} = \begin{cases} \delta_i \cdot e^{bt} \cdot cos(2\pi t) + Fm_i(k), & i \le N.FM \\ \delta_i \cdot e^{bt} \cdot cos(2\pi t) + Fm_{N.FM}(k), & i \ge N.FM \end{cases} \quad (3)$$

where, $\delta_i = \left| x_i^K - Fm_i \right|$ represents distance of moth at i^{th} place and its specific flame (Fm_i), b is a constant which is equal to 1 and $'t'$ be any random number between –1 and 1 and represented as follows

$$r = -1 + current_{iter}\left(\frac{-1}{max_{iter}}\right) \quad (4)$$

$$t = (r - 1) \times rand + 1 \quad (5)$$

where max_{iter} denotes the number of maximum iterations, and 'r' is the convergence constant, which falls linearly from (−1) to (−2) in the MFO method, demonstrating that both diversification and intensification occur. The author used the following formula can obtain the number of flames ($N.FM$) that has been reduced over the iteration.

$$N.FM = round\left(N.FM_{Lst\ it} - crnt.it\frac{(N.FM_{Lst\ it}-1)}{max\ it}\right) \quad (6)$$

3. PROPOSED METHOD

The spiral movement of moths around the flame provides diversification and intensification in MFO and which are better understood by the exponent factor $'t'$. The parameter 't' is taken from r to 1 in the conventional MFO algorithm, where r is a linearly decreasing function from -1 to 2 during the length of iteration but in the proposed M-MFO, we embedded a non-linear decreasing function from -1 to 2 to maintain an equilibrium between global and local search, which first explore the search space and then slowly reduce and then exploit the region. The mathematical formulation of the parameter 'r' is as follows:

$$r = -2 + e^{\left(\frac{-iteration}{k \times maxiter}\right)^3} \quad (7)$$

where, k is a constant and its value is 0.55 which is suitable chosen so that it helps in both global and local search and represented in Fig. 1.

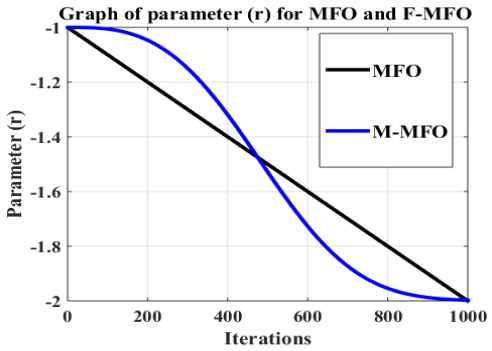

Figure 1. non-linear adaption curve

SELF-ADAPTIVE LEVY MUTATION

Mathematically, the random step lengths of Levy flight described in the following equation which are generated using a Levy distribution

$$L(s) \sim |s|^{-1-\beta} \quad (8)$$

where, s is the step length and $\beta \in (0 < \beta \leq 2)$. This study uses a Mantegna method [23] for a symmetric Levy stable distribution to generate random step sizes. If the step size can be either positive or negative, we say that it is symmetric. The step size s of the Mantegna algorithm can be determined by solving Eqn. 9.

$$s = \frac{u}{|v|^{\frac{1}{\beta}}} \quad (9)$$

where, u and v are taken from normal distributions. That is

$$u \sim N(0,\ \sigma_u^2),\ v \sim N(0, \sigma_v^2)$$

where, $\sigma_u = \left\{ \frac{\Gamma(1+\beta) + \sin\left(\frac{\pi\beta}{2}\right)}{\beta\Gamma\left[\frac{1+\beta}{2}\right]2^{\frac{\beta-1}{2}}} \right\}^{1/\beta}$, $\sigma_v = 1$ (10)

Mathematically, the Gamma functions Γ (.) is as follows:

$$\Gamma(1+\beta) = \int_0^\infty t^\beta e^{-t} dt \quad (11)$$

In order to make efficient use of the search space, the step sizes in the proposed method are generated according to a Levy distribution.

$$step_{size(t)} = 0.001 * s(t) * SLC \quad (12)$$

where, $s(t)$ is determined using the Levy distribution as given in Eqn. (8), SLC is the global search algorithm's social learning component and t is the local search strategy's iteration counter. The step sizes in Levy flight are too large, and they frequently generate new solutions beyond the domain. As a result, in Eqn. (14) a multiplier of 0.001 is employed to minimize the step size.
In the newly developed M-MFO, the updated equation is as follows

$$X_{i+1} = X_{i+1} + (r \times sign(r - 0.5) * levy(1, D, \beta) \quad (13)$$

where β= 1+$e^{(k/T)}$

k =current iteration and T = Maximum iteration D is the dimension, $sign$ represents signum function.

Fibonacci search method (FSM):

The FSM is a mathematical procedure that uses Fibonacci numbers to determine the lowest or greatest value of a function by shifting and narrowing down the search range. The direction of shifting is based on the values of that function at two experiment points. The FSM is based on the Fibonacci numbers which is defined as follows

$Fib = [F_1, F_2, F_3, \ldots\ldots, F_n]$, where, $F_i \; \forall \; i = 1,2,..n$ are Fibonacci numbers and generated by the following equation

$$F_0 = 1 = F_1, F_{no} = F_{no-1} + F_{no-2}, no = 2,3,4, \ldots . n \quad (14)$$

Let for any iteration T, for each moth ($X_{i,j}$); i=1,2,…, N (number of population) and j = 1, 2, … , D (dimension), Let $X = (x_1, x_2, x_3, \ldots x_D)$ and $Y = (y_1, y_2, y_3, \ldots y_D)$ are two distinct search agents of any finite length of interval $[a_j, b_j]$. Take two x_1 and y_1 are two experimental members from X and Y which are calculated as follows:

$$x_1 = a_j + \left(\frac{F_{k-2}}{F_k}\right)(b_j - a_j), \; y_1 = b_j - \left(\frac{F_{k-2}}{F_k}\right)(b_j - a_j) \quad (15)$$

The range is moved to the right because the function's value at y_1 is greater than that at x_1, and to the left if x_1 is greater than that at y_1. The new value of x_2 and y_2 are generated using the Fibonacci search formula as, $x_2 = x_1$ and $y_2 = y_1$. If two functional values are unequal then, only one (x_2 or y_2) will be considered as a new experimental point, whereas the other will be same as either of x_1 and y_1 depending on the contracting direction. In the case of two equal function values, then, both x_2 and y_2 are formed new experimental points and continue the process until the stopping criterion condition and at final iteration the new solution can be obtained by taking average of last two experimental numbers. Due to the high computational efficiency of FSM, the author of [24] used a modified Fibonacci search method for the partially shaded solar PV array. Recently, Yazici et al. [25] applied modified Fibonacci search method for wind energy conversion systems. We embedded the concept of Fibonacci search method after the position update phase of the MFO algorithm. The pseudocode of the proposed M-MFO algorithm is presented in Algorithm 1.

4. RESULTS AND DISCUSSION

It should be noted that the code for the proposed M-MFO algorithm has also been written in the MATLAB environment and run on a 1.80 GHz i5 8th generation computer with 8.00GB of RAM and MATLAB R2015a. The selected parameters of all population-based optimization algorithms involve the maximum iteration and number of populations 1000 and 30, respectively. To find the optimum set of parameters for M-MFO and all other algorithms, 30 trials are performed for each possible set of parameters and mean (M) and standard deviation (S) of all considered algorithms have recorded in Table 1. On fifteen benchmark functions, comprising unimodal and multimodal functions for 1000 dimensions, the simulation results of our proposed M-MFO have been compared with DE, PSO, FPA, BOA, SMFO [26], E-MFO [27], and basic MFO, as shown in Table-1.

Compared to the other optimization techniques, the recommended M-MFO has superior performance, as shown in Table 1. When compared to the PSO and the FPA algorithm, the proposed M-MFO is always the more effective. DE provides best results than M-MFO for F4 and F8 functions and provides similar results for F15. When we compared our proposed M-MFO with BOA algorithm, it provides best results except F4 and F8 functions. In addition to both SMFO and E-MFO, Except for F4, M-MFO yields far superior outcomes across the board. Furthermore, the newly developed M-MFO provides worse results for F4 and similar results for F14 as compared to MFO algorithm.

Table 2 displays the frequency with which average MFO's performance exceeds, meets, or falls short of that of the competing approaches. As can be seen in Table 2, M-MFO performs better than DE, PSO, FPA, BOA, SMFO, E-MFO, and MFO on 12, 15, 15, 13, 14, 14, and 13 benchmark functions; on 1, 0, 0, 0, 0, 0, and 1 occasions; and on 2, 0, 0, 2, 1, 1 and 1 occasions, M-MFO performs worse.

The Friedman test is a non-parametric statistic that was developed by Milton Friedman [28]. Throughout a series of test iterations, it can help identify any discrepancies in the treatment. In this study, we compare the average performance of the algorithms on each benchmark function using the Friedman rank test (with the help of IBM's SPSS programme). According to Table 3, M-MFO performs better than the other algorithms because it ranks lowest. Figure 2 displays M-very MFO's competitive convergence performance in comparison to other methods.

Algorithm 1. Pseudocode of M-MFO algorithm.

1. Input: Objective function $f(X)$, $X = (X_1\, X_2 \ldots\ldots X_{dim})$, Number of moths in the population (N), Maximum iteration ($Maximum_{iter}$), Flame number $(N.FM)$, b, and other related parameters are determined;
2. for $i = 1: N$
3. for $j = 1: D$
4. Generate N organism solutions $X_{i,j}$ ($i = 1, 2, \ldots., N$) randomly;
5. Find fitness;
6. check boundaries;
7. end for
8. end for
9. While $Current_{iter} < Maximum_{iter} + 1$
10. if Iteration == 1
11. Enter *N.FM* = N in initial population;
12. else
Employ Eqn. (8);
13. end if
14. Update the individuals by using Eqn. (13)
15. FM = Fitness Function f(X);
16. if Iteration == 1
17. Sort the moths according to FM;
18. Update the Flames;
19. Iteration = 0;
20. else
Sort the moths based on FM from last iteration;
21. Update the Flames;
22. end if
23. Reduce the convergence constant;
24. for $j = 1: N$
25. for $k = 1: n$
26. Update *r*, *t* and moths position as to their particular flame using Eqn. (3-5);

27. end for
28. end for
29. Apply Fibonacci approach using Eqn. (14-15) and check boundaries;
30. $Current_{iter} = Current_{iter} + 1$
31. end while
32. Output: The best solution with the minimum fitness function value in the ecosystem;

Table 1. Comparison of M-MFO with basic and MFO variants algorithms for D=1000.

Sl. No.		DE	PSO	FPA	BOA	SMFO	EMFO	MFO	M-MFO
1	M	2.73e+06	2.02e+06	5.12e+05	1.85e-02	7.09e-01	2.27e+00	1.89e+00	2.87e-80
	S	7.45e+04	2.04e+05	1.26e+06	7.75e-04	3.87e+00	4.47e-01	1.03e+01	1.57e-79
2	M	2.88e-57	4.95e-02	5.79	7.71e-04	1.02e-13	1.81e-08	2.93e-13	8.29e-132
	S	8.17e-57	6.45e-02	-9.99	2.29e-04	5.61e-13	1.32e-08	1.57e-12	3.99e-131
3	M	4.95e-175	1.48e-16	8.29e+01	2.52e-03	1.07e-12	1.55e-08	9.72e-13	1.22e-202
	S	0	8.10e-16	2.64e-09	9.40e-04	5.32e-12	1.80e-08	3.91e-12	0
4	M	-3.79e-03	1.41e-01	6.69e-02	-3.53e-03	-3.79e-03	-3.79e-03	-3.78e-03	-2.15e-03
	S	1.76e-18	1.44e-01	-3.78e-03	6.51e-04	2.31e-06	2.87e-07	2.67e-05	1.91e-03
5	M	3.36e+22	8.21e+22	1.56e+22	3.54e+00	1.05e+05	1.81e+02	3.58e+02	7.86e-14
	S	3.61e+21	1.50e+22	4.54e+22	3.91e+00	5.51e+05	4.22e+02	1.10e+03	4.28e-13
6	M	1.53e+11	1.36e+11	2.90e+10	2.23e-02	5.73e+01	8.62e+04	1.20e+03	3.69e-65
	S	3.21e+09	1.68e+10	1.13e+11	1.57e-03	3.13e+02	1.41e+04	6.56e+03	2.02e-64
7	M	6.46e+16	1.13e+17	2.69e+18	8.53e-03	4.60e-01	6.47e+00	1.07e-01	1.02e-76
	S	2.83e+16	2.40e+17	1.69e+13	4.77e-04	1.47e+00	1.33e+00	3.02e-01	5.61e-76
8	M	4.11e-34	3.44e+00	2.54e+02	1.78e-03	9.35e-03	1.77e-02	9.94e-02	9.25e-03
	S	2.25e-33	3.08e+00	2.76e-02	5.69e-03	1.14e-02	1.92e-02	2.44e-01	2.29e-02
9	M	5.74e+08	4.26e+08	9.03e+07	2.16e-02	5.28e+00	1.06e+03	6.95e-02	1.73e-84
	S	1.52e+07	3.74e+07	3.54e+08	1.54e-03	2.37e+01	1.61e+02	3.10e-01	9.46e-84
10	M	4.94e+06	9.99e+06	2.45e+06	1.97e-02	7.41e+00	1.13e+03	4.62e+01	1.41e-70
	S	1.01e+05	9.14e+05	7.83e+06	1.06e-03	4.06e+01	2.57e+02	2.53e+02	7.72e-70
11	M	9.54e+00	9.63e+00	5.51e-01	4.85e-02	2.43e-03	4.78e-01	2.46e-03	3.50e-06
	S	2.97e-02	1.67e-01	8.13e+00	2.17e-03	7.52e-03	7.15e-02	8.90e-03	1.43e-05
12	M	3.80e+09	7.89e+09	3.27e+09	9.99e+02	9.97e+02	1.25e+03	1.00e+03	9.99e+02
	S	9.95e+07	1.44e+09	3.93e+09	1.99e-02	2.92e+00	4.55e+01	1.14e+01	6.28e-02
13	M	1.17e+15	2.13e+15	4.88e+14	2.34e-02	4.18e+08	2.36e+09	7.91e+05	5.66e-77
	S	2.34e+13	2.20e+14	1.64e+15	1.90e-03	1.81e+09	3.42e+08	2.77e+06	3.10e-76
14	M	1.17e+06	2.05e+06	4.87e+05	3.10e+00	2.33e-01	3.33e-02	0	0
	S	2.62e+04	2.32e+05	1.48e+06	1.27e+00	1.28e+00	1.83e-01	0	0
15	M	0	1.77e+02	2.72e+02	3.35e-01	1.19e-08	6.12e-06	5.96e-12	0
	S	0	2.69e+02	1.78e-02	1.82e-01	4.22e-08	6.15e-06	2.04e-11	0

Table 2. Performance Assessment of M-MFO with considered algorithms for 1000 dimensions.

Proposed algorithm	DE	PSO	FPA	BOA	SMFO	E-MFO	MFO
Superior to	12	15	15	13	14	14	13
Similar to	1	0	0	0	0	0	1
Inferior to	2	0	0	2	1	1	1

Table 3. Friedman Rank Test between M-MFO and basic metaheuristic algorithms for D=500.

Algorithm	Mean rank	rank	P-value
M-MFO	**1.50**	**1**	With a P-value of, Ho is not supported at the 95% confidence level (0.0000.01). This means that there is a statistically significant difference in performance between methods at the 1% level of significance.
DE	4.57	6	
PSO	6.44	8	
FPA	5.48	7	
BOA	3.06	2	
SMFO	3.65	4	
MFO	4.11	5	
E-MFO	3.22	3	

REAL-LIFE PROBLEM: CANTILEVER BEAM DESIGN PROBLEM

The cantilever beam design problem is used to solve by the M-MFO algorithm and compared with few metaheuristic algorithms named found in the literature [21]**.** In Table 4, the best values of the above algorithms with M-MFO are presented and from Table 4, it can be observed that M-MFO has proved superiority among other algorithms.

Table 4. M-MFO vs. other cantilever beam design techniques.

Algorithm	Optimal variables					Optimal weight
	x_1	x_2	x_3	x_4	x_5	
M-MFO	**5.89794**	**5.38340**	**4.12496**	**3.72437**	**2.32918**	**1.33959**
ALO	6.01812	5.31142	4.48836	3.49751	2.158329	1.33995
SOS	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996
CS	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999
MMA	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA-I	6.0100	5.30400	4.4900	3.4980	2.1500	1.3400
GCA-II	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400

5. Conclusion

This paper presents an upgraded variety of the classic MFO algorithm namely, a modernized MFO (M-MFO) which uses a non-linear adaption formula, self-adaptive levy mutation and Fibonacci search concept and to improve the MFO algorithm and make a proper balance between diversification and intensification. To evaluate the performance of M-MFO, fifteen benchmark functions over 1000 dimensions have considered for experimentations and compared with the five basic and two variants of MFO algorithm. Friedman test is used to measure the effectiveness of the suggested M-MFO algorithm. It

has also been used to solve one engineering issues, providing a better result than other algorithms, to validate the proposed M-MFO.

In future we can extend it to multi-objective optimization, apply it to solve higher constraint optimization problem like car-side crash problem, Robot gripper problem, welded beam design problem etc. We can generate an efficient metaheuristic algorithm by hybridizing our suggested approach M-MFO with any other meta-heuristic algorithm. Also, simply we apply our M-MFO to solve combined economical emission design problem or apply in medical disease problem like melanoma detection in human body, X-ray image segmentation of Covid-19.

Compliance with ethical standards

Conflict of interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval: Any of the authors' investigations with human participants or animals are not included in this article.

Appendix

Sl.no	Functions	Formulation of objective functions	D	Fmin	Search space
Unimodal Benchmark Functions					
F1	Sphere	$f(x) = \sum_{j=1}^{d} x_j^2$	30	0	[-100, 100]
F2	Matyas	$f(x) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$	2	0	[-10, 10]
F3	Sumsquare	$f(x) = \sum_{i=1}^{D} x_i^2 \times i$	30	0	[-10, 10]
F4	Zettl	$f(x) = (x - 1^2 + x - 2^2 - 2x_1)^2 + 0.25x_1$	2	-0.00379	[-1, 5]
F5	Zakhrov	$f(x) = \sum_{j=1}^{d} x_j^2 + \left(0.5\sum_{j=1}^{d} jx_j\right)^2 + \left(0.5\sum_{j=1}^{d} jx_j\right)^4$	2	0	[-5, 10]
F6	High conditioned Eliptic function	$f(x) = \sum_{j=1}^{d} \left(1000000^{\left(\frac{j-1}{D-1}\right)} \times x_j^2\right)$	30	0	[-100, 100]
F7	Brown	$f(x) = \sum_{j=1}^{n-1} (x_j^2)^{(x_{j+1}^2+1)} + \left((x_{j+1}^2)^{(x_j^2+1)}\right)$	30	0	[-1, 4]
F8	Cube	$f(x) = 100(x_2 - x_1^3)^2 + (1 - x_1)^2$		0	[-10, 10]
F9	Rotated hyper elipsoid	$f(x) = \sum_{j=1}^{d} (d + 1 - j)x_j^2$		0	[-100, 100]

F10	Schwefel 1.2	$f(x) = \sum_{j=1}^{d} \sum_{k=1}^{j} x_j^2$	30	0	[-100, 100]
Multimodal Benchmark Functions					
F11	Bohachevsky	$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.3$	2	0	[-100, 100]
F12	Bohachevsky 3	$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.3$	2	0	[-50, 50]
F13	Levy	$f(x) = sin^2(\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2[1 + 10sin^2(\pi x_i + 1)] + (x_D - 1)^2[1 + sin^2(2\pi x_D)]$ Where, $x_i = 1 + \frac{1}{4}(x_i - 1),\ i = 1,2, \dots \dots \dots D$	30	0	[-10, 10]
F14	Alpine	$f(x) = \sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	30	0	[-10, 10]
F15	Schaffers	$f(x) = 0.5 + \frac{sin^2(x_1^2 + x_2^2) - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2}$	2	0	[-100, 100]

REFERENCES

[1] Abbassi, R., Abbassi, A., Heidari, A. A., & Mirjalili, S. (2019). An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models. Energy Conversion and Management, 179, 362–372.
[2] McCarthy, J. F. (1989). Block-conjugate-gradient method. Physical Review D, 40(6), 2149.
[3] Wu, G., Pedrycz, W., Suganthan, P. N., & Mallipeddi, R. (2015). A variable reduction strategy for evolutionary algorithms handling equality constraints. Applied Soft Computing, 37, 774–786.
[4] Kennedy, J., Eberhart, R. Particle swarm optimization. In: *Proceedings of ICNN'95—international conference on neural networks*, Perth, Australia, 1995, 1942–1948.
[5] Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11(4), 341–359.
[6] Yang, X.-S. (2012). Flower pollination algorithm for global optimization. International Conference on Unconventional Computing and Natural Computation, 240–249.
[7] Arora, S., & Singh, S. (2015). Butterfly algorithm with levy flights for global optimization. *2015 International Conference on Signal Processing, Computing and Control (ISPCC)*, 220–224.
[8] Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowledge-Based Systems, 89, 228–249.
[9] Singh, P., & Prakash, S. (2019). Optical network unit placement in Fiber-Wireless (FiWi) access network by Whale Optimization Algorithm. Optical Fiber Technology, 52, 101965.
[10] Mohanty, B. (2019). Performance analysis of moth flame optimization algorithm for AGC system. International Journal of Modelling and Simulation, 39(2), 73–87.
[11] Khairuzzaman, A. K. M., & Chaudhury, S. (2020). Modified moth-flame optimization algorithm-based multilevel minimum cross entropy thresholding for image segmentation: international journal of swarm intelligence research, 11(4), 123–139.
[12] Gupta, D., Ahlawat, A. K., Sharma, A., & Rodrigues, J. J. P. C. (2020). Feature selection and evaluation for software usability model using modified moth-flame optimization. Computing, 102(6), 1503–1520.

[13] Muduli, D., Dash, R., & Majhi, B. (2020). Automated breast cancer detection in digital mammograms: A moth flame optimization-based ELM approach. Biomedical Signal Processing and Control, 59, 101912.

[14] Hongwei, L., Jianyong, L., Liang, C., Jingbo, B., Yangyang, S., & Kai, L. (2019). Chaos-enhanced moth-flame optimization algorithm for global optimization. Journal of Systems Engineering and Electronics, 30(6), 1144–1159.

[15] Xu, Y., Chen, H., Luo, J., Zhang, Q., Jiao, S., & Zhang, X. (2019). Enhanced Moth-flame optimizer with mutation strategy for global optimization. Information Sciences, 492, 181–203.

[16] Xu, Y., Chen, H., Heidari, A. A., Luo, J., Zhang, Q., Zhao, X., & Li, C. (2019). An efficient chaotic mutative moth-flame-inspired optimizer for global optimization tasks. Expert Systems with Applications, 129, 135–155.

[17] Kaur, K., Singh, U., & Salgotra, R. (2020). An enhanced moth flame optimization. Neural Computing and Applications, 32(7), 2315–2349.

[18] Tumar, I., Hassouneh, Y., Turabieh, H., & Thaher, T. (2020). Enhanced binary moth flame optimization as a feature selection algorithm to predict software fault prediction. IEEE Access, 8, 8041–8055.

[19] Gu, W., & Xiang, G. (2021). Improved Moth Flame Optimization with Multioperator for solving real-world optimization problems. *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 5, 2459–2462.

[20] Ma, L., Wang, C., Xie, N., Shi, M., Ye, Y., & Wang, L. (2021). Moth-flame optimization algorithm based on diversity and mutation strategy. Applied Intelligence.

[21] Sahoo, S. K., Saha, A. K., Sharma, S., Mirjalili, S., & Chakraborty, S. (2022). An enhanced moth flame optimization with mutualism scheme for function optimization. Soft Computing, 26(6), 2855–2882.

[22] Yu, C., Heidari, A. A., & Chen, H. (2020). A quantum-behaved simulated annealing algorithm-based moth-flame optimization method. Applied Mathematical Modelling, 87, 1–19.

[23] Yang, X.-S. (2010). Nature-inspired metaheuristic algorithms. Luniver press.

[24] Ramaprabha, R. (2012). Maximum power point tracking of partially shaded solar PV system using modified Fibonacci search method with fuzzy controller. 12.

[25] Yazıcı, İ., Yaylacı, E. K., Cevher, B., Yalçın, F., & Yüzkollar, C. (2021). A new MPPT method based on a modified Fibonacci search algorithm for wind energy conversion systems. Journal of Renewable and Sustainable Energy, 13(1), 013304.

[26] Chen, C. L., Wang, X., Yu, H., Wang, M., Chen, H. (2021). Dealing with multi-modality using synthesis of Moth-flame optimizer with sine cosine mechanisms. Mathematics and Computers in Simulation, 188, 291–318.

[27] Elsakaan, A. A., El-Sehiemy, R. A., Kaddah, S. S., & Elsaid, M. I. (2018). An enhanced moth-flame optimizer for solving non-smooth economic dispatch problems with emissions. Energy, 157, 1063–1078.

[28] Milton F (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association. 32: 675–701.