Int. J. Nonlinear Anal. Appl. In Press, 1–13
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2022.26929.3453

A novel algorithm to solve fixed points of various functions by the Bisection-Moth-Flame optimization algorithm

Sayyed Masood Zekavatmand, Javad Vahidi*

Department of Applied Mathematics, Iran University of Science and Technology, Tehran, Iran

(Communicated by Madjid Eshaghi Gordji)

Abstract

The essential purpose of this paper is to obtain the fixed point of different functions by using a modern repetitive method. We incorporate concepts suggested in the Bisection method and the Moth-Flame Optimization algorithm. This algorithm is more impressive for finding fixed point functions. We also implement this method for four functions and finally compare the current method with other methods such as ALO, MVO, SSA, SCA algorithms. the proposed method shows a decent functionality than the other four methods.

Keywords: Fixed-point theory, Moth-Flame Optimization mechanism, Bisection procedure
2020 MSC: 54H25, 47H10, 47N10

1 Introduction

Many real-world problems can be modeled as mathematical problems. Many of these problems come in the form of one or more nonlinear equations that solve a major problem in nature. Therefore, solving equations and gaining roots of functions, is of special importance in engineering, basic sciences and applied sciences. This reason has led many researchers in recent years to research to solve these problems and present the results of their efforts in the form of various articles [17, 1, 37, 8, 41, 59, 46, 3, 54, 26, 30]. The bisection algorithm is a rooting-discovering procedure. It exerts for each function that is continuous for per identifies two valencies by contrary emblems. The procedure includes of frequently halving distance determined with these valencies. Afterwards choosing the sub-distance where in the function shifts emblem. It should include a root. It is a straightforward and sturdy procedure, however it is as well as comparatively lagging. For this reason, it is frequently utilization to gain a precise estimate of one solution that is henceforth utilization as a beginning point to rather quickly converging procedures [7, 50, 52, 53, 62, 63]. Accurate algorithms can discover the optimal answer identically. However, they are not impressive sufficient in strict optimization questions. Their performance period enhances exponentially relevant to the girths of the problems [6, 20, 33, 36, 38, 34, 35, 16, 11, 57, 51]. Metaheuristic algorithms are divided into two general groups, two methods based on one answer and population. One-answer algorithms change an answer during the search process, while population-based algorithms consider a population of answers while searching. One-answer algorithms focus on local search areas; In contrast, population-based algorithms can search simultaneously in different areas of the answer space [6, 13, 60, 22, 49, 19, 28, 33, 2, 40, 25, 58, 30]. Various criteria can be used to classify meta-heuristic algorithms. Some of the one-answer meta-heuristic algorithms that have been introduced in recent years and have

*Corresponding author
Email addresses: S_zekavatmand@mathdep.iust.ac.ir (Sayyed Masood Zekavatmand), Jvahidi@iust.ac.ir (Javad Vahidi)

Received: April 2022 *Accepted:* September 2022

many applications for solving various problems are: simulated annealing algorithm (SA) [61], tabu search algorithm (TSA) [15], algorithm GRASP search (GSA) [30], variable neighborhood search (VNS) algorithm [18], guided local search algorithm (GLS) [32], iterated local search algorithm (ILS) [20]. Also, a number of population-based meta-innovative algorithms that have been introduced in recent years and have many applications for solving various problems are: Imperialist Competitive Algorithm (ICA) [4], Artificial Immune System Algorithm (AIS) [10], Harmony Search Algorithm (HS) [31]. In recent years, researchers have also been able to solve complex problems in the field of engineering and mathematics using a number of these algorithms. Some of these algorithms are: Cuckoo Optimization Algorithm (COA) [43], Gravitational Search Algorithm (GSA) [44], Invasive Weed Optimization Algorithm (IWO) [39], League Championship Algorithm (LCA) [21], Optics Inspired Optimization Algorithm (OIO) [29], Shuffled Frog Leaping Algorithm (SFL) [14], Stochastic Fractal Search Algorithm (SFS) [47] Teaching Learning Based Optimization Algorithm (TLBO) [45], Wind Driven Optimization Algorithm (WD) [62] and etc [13, 42, 31, 56, 24, 12, 48, 9, 55, 55]. In the present article, we bring a novel repetitious method that merges the benefits of both the Moth-Flame Optimization Algorithm and the Bisection method to dissolve the difficult fixed point problem. The residual segments of the present paper are organized as follows: at Sect. 2, a short overview is performed on the Bisection procedure. Also Moth-Flame Optimization mechanism is said. Then the Fixed-point theory is explained. In the following , offered mechanism communicate in the next section. At Sect. 4, evaluates precision of suggested procedure with distinct procedures on various functions. Finally, the result is given in the final section.

2 Fundamental implications

By reading this segment, a concise overview is accomplished on the Moth-flame optimization algorithm, and the Bisection method is clarified. In the continuation of this segment fixed point theory is indicated.

2.1 Moth-Flame Optimization Algorithm

Moth-flame optimization algorithm, which is called MFO for short, is one of the optimization algorithms as if finds a way to dissolve the problem by the behavior of the impellers next to the flame or fire. This algorithm was published in 2015 by Seyed Ali Mir Jalili in an article entitled "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm" in the journal Knowledge-Based Systems [34]. This procedure is a novel exploratory model retrieved from the nature and treatment of butterflies and their interest in flame or fire. Butterflies fly at night for long distances while sustaining a constant hermitage via nexus to the moon. But these fantasy insects are trapped in a useless and deadly spiral path around artificial lights. In the propeller algorithm, this behavior of the propellers is mathematically used to perform the optimization. The MFO algorithm has many similarities with other known nature-inspired algorithms, and statistical results on functions show that this algorithm can provide promising and competitive results. Butterflies are fascinating insects that there are different species of them in our environment. The life of each butterfly is divided into two main parts: larvae and adults. A cocoon is a place where larvae turn into butterflies. One of the most critical features of butterflies is their mechanism of movement at night. Butterflies use the moon to fly at night. They use a structure called orientation. They use the transverse to move at night. Using this method, the constant angle of a butterfly to the moon when it flies remains constant. This provides an effective mechanism for butterflies to travel a relatively long distance in a straight line. Despite the transverse orientation of the propellers, the propellers usually fly in a spiral around the lights, because the propellers are deceived by artificial light and this movement passes through them. This behavior of impellers is modeled and named in the language of mathematics as the Moth-Flame Optimization Algorithm (MFO). In the Moth-Flame Optimization algorithm, we assume that the solutions to the question are the volunteers for the impellers. We also consider the question variables as the situation of the impellers in the air. The impeller set is displayed as a matrix. This is because the Moth-Flame Optimization algorithm is population-based. Also, a matrix can be a good option for displaying the performance of each impeller and its performance (like the M matrix). There is an array for each butterfly to reserve the OM value. Another key feature in the MFO algorithm is the butterflylike matrix. This matrix is a fire matrix, and OF is an array used to reserve the values of the fitness function.

$$M = \begin{pmatrix} m_{1,1} & m_{1,2} & \cdots & \cdots & m_{1,d} \\ m_{2,1} & m_{2,2} & \cdots & \cdots & m_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ m_{n,1} & m_{n,2} & \cdots & \cdots & m_{n,d} \end{pmatrix} \Longrightarrow OM = \begin{pmatrix} OM_1 \\ OM_2 \\ \vdots \\ OM_n \end{pmatrix}$$

$$F = \begin{pmatrix} F_{1,1} & F_{1,2} & \cdots & \cdots & F_{1,d} \\ F_{2,1} & F_{2,2} & \cdots & \cdots & F_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ F_{n,1} & F_{n,2} & \cdots & \cdots & F_{n,d} \end{pmatrix} \Longrightarrow OF = \begin{pmatrix} OF_1 \\ OF_2 \\ \vdots \\ OF_n \end{pmatrix}$$

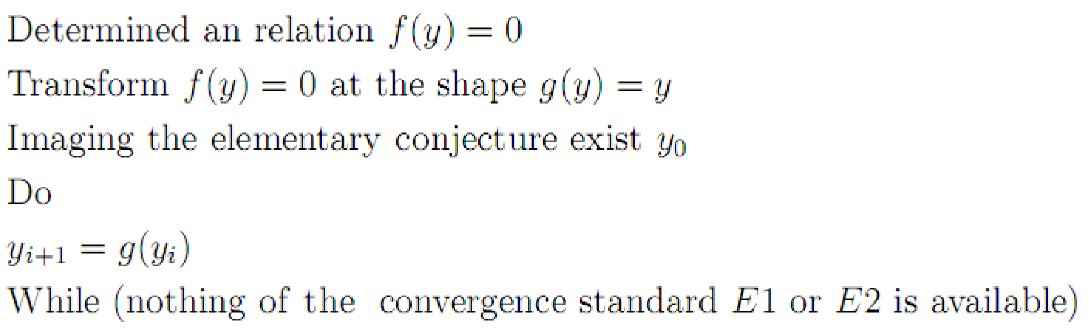
It must be prominent where that flames and butterflies are both solutions. The diversity among them is how they are cliqued and refreshed in per repetition. Butterflies are the probe factors that travel in the probe place, while flames are the best butterfly situation ever arrived. So, per butterfly probes about a flag and refreshes it if it discovers a superior solution. This mechanism does not cause a butterfly to lose its best solution [34].
The pseudo-code of the MFO algorithm is as follows:

Algorithm 1 The pseudo-code of the MFO algorithm

```
Print the best solution
Initialize the parameters for Moth-flames
Initialize Moth position $M_i$ randomly
for $i = 1$ to $n$ do
  Calculate the fitness function $f_i$
end for
while $interation \leq Max_{interations}$ do
  Update the position of $M_i$
  Calculate the number of flames using $flameno = round\big(N - 1 * \frac{N-1}{T}\big)$
  Evaluate the fitness function $f_i$
  if $interation == 1$ then
    $F = sort(M)$ and $OF = sort(OM)$
  else
    $F = sort(M_{t-1}, M_t)$ and $OF = sort(M_{t-1}, M_t)$
  end if
  for $i = 1$ to $n$ do
    for $j = 1$ to $d$ do
      Update the value of r and t
      Calculate the value of D respect to its corresponding moth
      using $D_i = |F_j - M_i|$
      Update $M(i, j)$ respect to its corresponding moth
      using $S(M_i, F_j) = D_i.e^{bt}.\cos(2\pi t) + F_j$
    end for
  end for
end while
```

2.2 Mechanism of fixed point theory

Point (b) is a fixed point for the function g(y), when point (b) holds for $g(b) = b$. This means that the point represented by a function on itself can be considered a fixed point for that function. To find the fixed point of a function, we must convert the relation $y = g(y)$ to the relation $f(y) = 0$. Then using the recessive connection for $y_{i+1} = g(y_i)$, $i = 0, 1, 2, \ldots$ and using the primary value y_0 which is chosen based on conjecture and coincidence, we obtain the fixed point of the function. For further understanding, the mechanism of the fixed point theory is shown in the following figure, and to summarize the iterations of the fixed point theory in (FPI).



```
Determined an relation $f(y) = 0$
Transform $f(y) = 0$ at the shape $g(y) = y$
Imaging the elementary conjecture exist $y_0$
Do
$y_{i+1} = g(y_i)$
While (nothing of the  convergence standard $E1$ or $E2$ is available)
```

Figure 1: (FPI) Mechanism

According to Figure 1, a repetitious procedure that can be used to find the solutions to equation $f(y) = y$ is the recessive connection $f(y_i) = y_{i+1}$, $i = 0, 1, 2, \ldots$ by several premier supposable y_0. The mechanism pauses until per of the tracking pausing scale is available: E1. Confirmation former the whole number of repetitions N. E2. With examining the position $|g(y_i) - x_{i+1}|$ (wherever i is the repetition number) lower than several toleration range, tell epsilon, fixed former. (Figure 1)

For instance, function $f_1 = (\frac{y^3}{150}) - 2\sin(y), y \in [-10, +10]$ has a fixed point, But a series of functions do not have a fixed point. For instance, Function $f(y) = 2y + 3, y \in \mathbb{R}$ has no fixed points, Because no number can be found that establishes the equation $y = 2y + 3$ [30].

2.3 Definition of the Bisection method

Whenever we want to provide a numerical solution for the equation $f(x) = 0$, one of its methods is bisection method. Note that the function f must be a continuous function and the values $f(a)$ and $f(b)$ on the interval $[a, b]$ must have opposite signs. In this case, we can use the theorem of the mean to say that the function f in the interval (a, b) has at least one root [30].

In the bisection method, in each step, using the relation $c = \frac{(a+b)}{2}$, we divide the interval into two parts. Then we calculate the value of $f(c)$. In this case, two situations may occur:
Case 1: If the values $f(a)$ and $f(c)$ have different symbols, then using the mean value theorem, the function f in the interval (a, c) has at least one root.
Case 2: If the values $f(c)$ and $f(b)$ have different symbols, then using the mean value theorem, the function f in the interval (c, b) has at least one root [60].
These steps continue until the interval is small enough and finally we reach a small interval where the two ends of the interval are the same size.
Note that the values at each end of the interval must have different symbols so that we can select it as a new, smaller interval [30].
In Algorithm 2, you can see the pseudo-code of the bisection method.

3 The Bisection-Moth-Flame Optimization Algorithm

The main purpose of studying this section is to fully explain the mechanism of our new method, which is a combination of the MFO algorithm and the bisection method. We show this algorithm briefly as BMFO. The introduced algorithm provides the ability to obtain fixed points of different functions with very close approximations to the exact answer. For this purpose, consider the function $g(x) = 0$. We know that to get the fixed points of this function, it must be $g(x) = x$. Now consider the function $f(x)$ as $f(x) = g(x) - x$. We can obtain the answers of the function $f(x)$ to obtain the fixed points of the function $g(x)$. Now consider the function $h(x)$ as $h(x) = |f(x)|$. In simpler terms, we can say that the minimum solutions of the function $h(x)$ are the same as the fixed point of the function $g(x)$. So instead of solving the function $g(x) = x$, we get the minimum answer of the function $h(x)$. In this step, using the MFO algorithm, consider an answer such as x_k that is randomly selected in the range $I_k = [a_k, b_k]$. In fact, the selection of the initial answer x_k is completely random and is done using the MFO algorithm. Then, at this stage, the bisection method is used. We calculate the values of $f(x_k)$ and $f(a_k)$ and see if the product of them is a negative number or not. We also do the same for $f(x_k)$ and $f(b_k)$ and see if the product of these two numerical values is negative. Note

Algorithm 2 Algorithm of the bisection method

```
Require: A continuous function $f(x)$, interval $[a, b]$, Tolerance: TOL; maximum number of iterations: $n_0$
  if $\frac{(b-a)}{2} < TOL$ then
    exit
    $Set i = 1$
    while $i \leq N_0$ and $\frac{(b-a)}{2} \geq TOL$ do
      $Set c = a + \frac{(b-a)}{2}$ Compute $f(x)$
      if $f(c) = 0$ then
        output c and exit
      end if
      if $f(a)f(c) > 0$ then
        set $a = c$ and $f(a) = f(c)$
      else
        set $b = c$ and $f(b) = f(c)$
      end if
      $i = i + 1$
    end while
    if $\frac{(b-a)}{2} < TOL$ then
      output c, 'tolerance limit exceeded!'
    end if
    if $i > N_0$ then
      output c, 'number of iterations reached $N_0$'
    end if
  end if
```

that the termination condition of the algorithm is that $f(x_k) = 0$.
In Algorithm 3 for per repetition we gain accidentally modern estimation amount x_k of the answer equation with MVO algorithm on I_k, for per $k \in N$. I_{k+1} is either $[a_k, \frac{(x_k + a_k)}{2}]$ (just b_k shifts) or we have $[\frac{(a_k + x_k)}{2}, x_k]$ or $[x_k, \frac{(x_k + b_k)}{2}]$ (both a_k and b_k shifts) or $[\frac{(b_k + x_k)}{2}, b_k]$ (just a_k shifts). So, we achieve $I_{k+1} \subseteq I_k$.
Afterwards, c belongs to the subscription of all I_k.
Finally, the limit $c - x_k$ will be equal to c when k tends to infinity.

4 Using the introduced algorithm for several functions

At present part, clarify proposed method by several examples and compare the outcomes with other evolutionary optimization methods similar ALO, MVO, SSA and as well as SCA. Consider the following functions.

$$
\begin{aligned}
g_1(x) :& \frac{x^3}{150} - 2\sin(x) = x, \; x \in [-10, +10] \\
g_2(x) :& -\cos(\sqrt{\frac{x}{50}}) + 250 = x, \; x \in [200, 300] \\
g_3(x) :& 10 + \frac{e}{2} - 10e^{-0.2\sqrt{x^2}} - \frac{1}{2}e^{\cos(2\pi x)} \\
g_4(x) :& 5 + \frac{x^2}{2} - 5\cos(2\pi x) = x, \; x \in [-20, 1)
\end{aligned}
$$

In Table 1, these functions are categorized and introduced.

Algorithm 3 the pseudo-code of the Bisection-Moth-Flame Optimization Algorithm

```
Given fixed point problem $g(x) = x,\ x \in [a, b]$
End points: $a_0 = a,\ b_0 = b$ ; tolerance: $TOL(1e^{-t}, t \gg 0)$
Let $f(x) = g(x) - x,\ h(x) = |f(x)|,\ k = 0$
while $h(x_k) \leq TOL$ do
    $x_k \in [a_k, b_k]$ is generated by MFO algorithm that minimize $h(x)$
    if $f(x_k)f(a_k) < 0$ then
        $b_{k+1} = \frac{(x_k + b_k)}{2}$ $temp = \frac{(x_k + a_k)}{2}$
        if $f(temp)f(x_k) < 0$ then
            $a_{k+1} = temp$
        else
            $a_{k+1} = a_k$
        end if
    else
        $a_{k+1} = \frac{(x_k + a_k)}{2}$ $temp = \frac{(x_k + b_k)}{2}$
        if $f(temp)f(x_k) < 0$ then
            $b_{k+1} = temp$
        else
            $b_{k+1} = b_k$
        end if
    end if
    $k = k + 1$
end while
$Output : c = x_k$
```

Table 1: Category of functions introduced

Function	Domain
$f_1(x) \rightarrow \dfrac{x^3}{150} - 2\sin x = 0$	$[-10, +10]$
$f_2(x) \rightarrow -\cos(\sqrt{\frac{x}{50}}) + 250 = 0$	$[200, 300]$
$f_3(x) \rightarrow 10 + \frac{e}{2} - 10e^{-0.2\sqrt{x^2}} - \frac{1}{2}e^{\cos(2\pi x)} = 0$	$[1, 21]$
$f_4(x) \rightarrow 5 + \dfrac{x^2}{2} - 5\cos(2\pi x) = 0$	$[-20, 1)$

According to the definitions mentioned in section 2.3, for the function g_1, the fixed point is the number 0. Also, fixed point of g_2 function is a number very close to 250 and the fixed point of the g_3 function is a number very close to 10, and finally the fixed point of the function g_4 is 0.
Here we want to provide a complete description of solving the function $f_1(x) = 0$ using the mentioned algorithm. First we produce an accidental initial value for the function f_1 at $x_0 \in I_0 = [-10, +10]$ using the MFO algorithm. In the next step, using the method mentioned in Section 2.3, we gain modern approximate value $x_1 \in I_1 \subseteq I_0$ as a solution. Then we continue this work until the desired solution is obtained with the favorable accuracy. The outcomes acquired toward any function via ALO, MVO, SSA, SCA and BMFO mechanisms are presented in Table 2. Shapes 2 to 5 demonstrate scheme of improvement procedure of g_1 to g_4 functions via BMFO mechanism in (a) and shape of solving fixed point of g_1 to g_4 functions via BMFO mechanism in (b). Shapes 6 to 9 demonstrate scheme of the outcome of ALO, MVO, SSA, SCA algorithms and suggested algorithm to dissolve the fixed point problem of each function in one graph. The results and figures show that the BMFO algorithm, compared to the other four algorithms, has a better performance and a more accurate solution for finding fixed points of functions.

Table 2: The outcomes acquired toward any function via ALO, MVO, SSA, SCA and BMFO

algorithm	Components	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$
ALO	error	$9.76E-12$	$1.73E-09$	$4.24E-11$	$1.04E-09$
	X-best	$-3.3E-12$	250.6194	$9.64E+00$	0.010084403
	mean(e)	$3.58E-05$	0.000379	$3.13E-05$	$1.95E-05$
	std(e)	0.000547	0.002278	$1.77E-04$	$1.11E-04$
MVO	error	$2.48E-08$	0.014449	$1.23E-05$	$5.31E-07$
	X-best	$-8.3E-09$	250.6339	$9.64E+00$	0.010083872
	mean(e)	0.000101	0.014449	0.000164044	$4.31E-05$
	std(e)	0.001331	0	0.000737533	$2.12E-04$
SCA	error	$2.6E-129$	0.000238	$9.07E-06$	$8.11E-128$
	X-best	$8.8E-130$	250.6192	8.831861647	$8.11E-128$
	mean(e)	0.000211	0.001452	$5.40E-05$	$1.68E-05$
	std(e)	0.006502	0.005189	0.000128074	$4.60E-04$
SSA	error	$4.46E-10$	$2.88E-08$	$2.54E-09$	$1.04E-10$
	X-best	$-1.5E-10$	250.6194	9.640214463	0.010084404
	mean(e)	$3.32E-05$	0.000101	$4.41E-05$	$1.31E-05$
	std(e)	0.000134	0.000657	0.000531736	$1.71E-05$
BMFO	error	$3.92E-95$	0	0	0
	X-best	$1.31E-95$	250.6194	9.640214461	0.010084404
	mean(e)	$6.47E-22$	0	0	$4.62E-15$
	std(e)	$7.03E-21$	0	0	$1.14E-14$

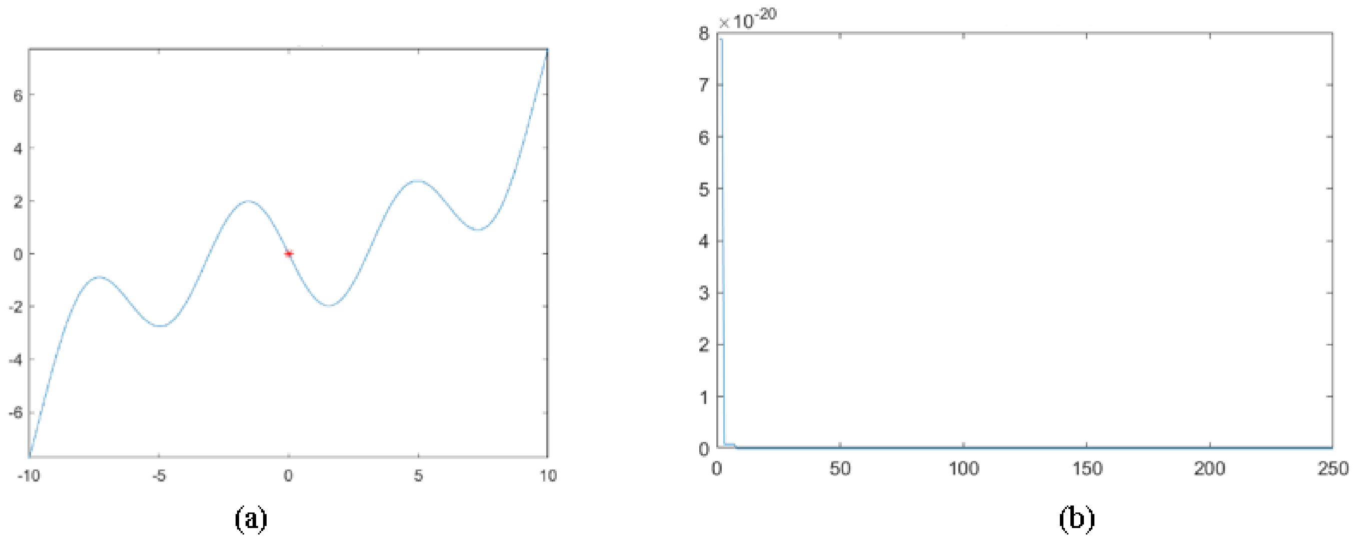


Figure 2: Shape of improvement procedure of $g_1(x)$ function via BMFO algorithm at form (a) and shape of solving fixed point of the $g_1(x)$ function via BMFO algorithm at form (b)

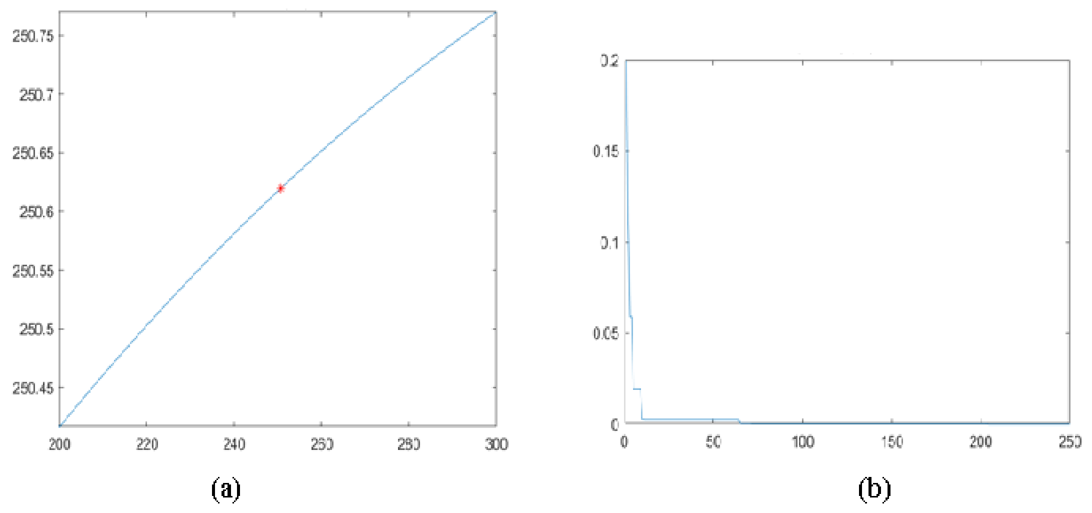


Figure 3: Shape of improvement procedure of $g_2(x)$ function via BMFO algorithm at form (a) and shape of solving fixed point of the $g_2(x)$ function via BMFO algorithm at form (b)

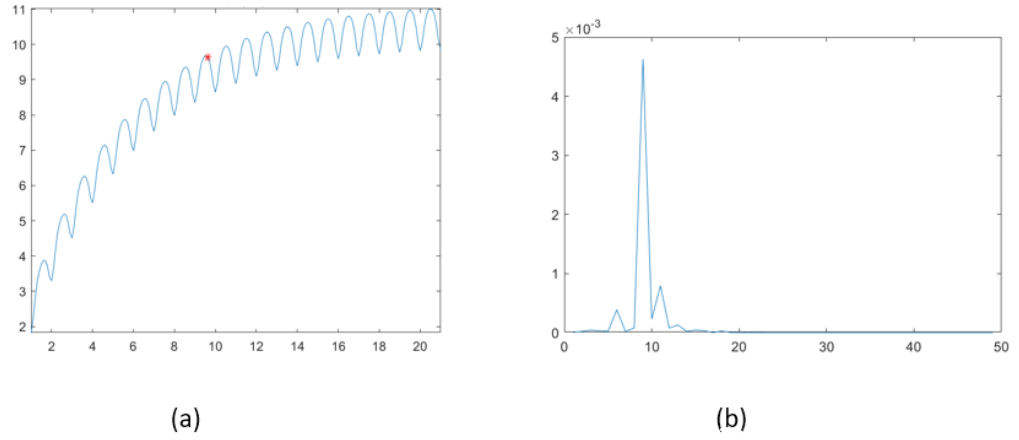


Figure 4: Shape of improvement procedure of $g_3(x)$ function via BMFO algorithm at form (a) and shape of solving fixed point of the $g_3(x)$ function via BMFO algorithm at form (b)

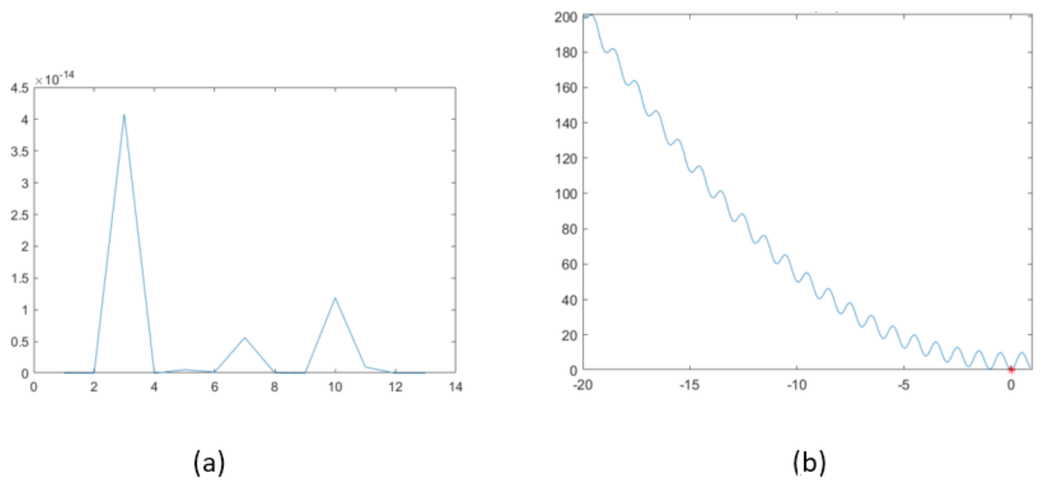


Figure 5: Shape of improvement procedure of $g_4(x)$ function via BMFO algorithm at form (a) and shape of solving fixed point of the $g_4(x)$ function via BMFO algorithm at form (b)

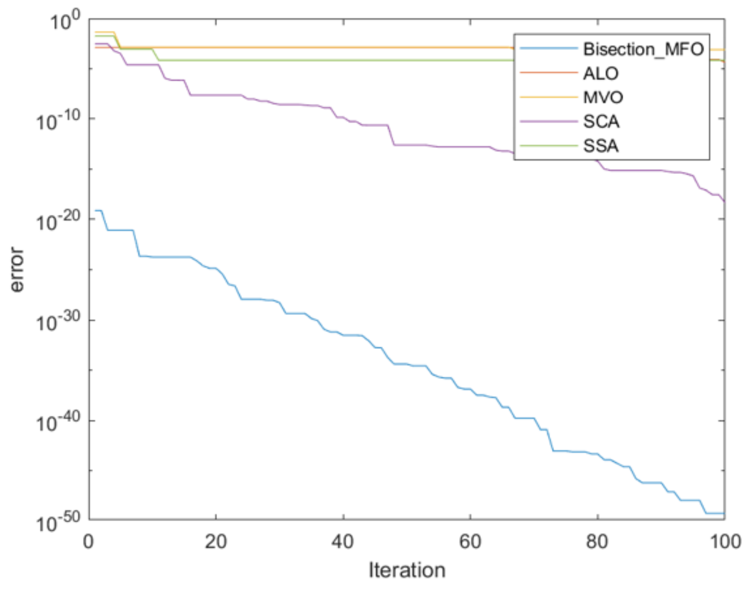


Figure 6: Shape of WOA, SSA, ALO, MVO algorithms and suggested algorithm to dissolve the fixed point of function g_1

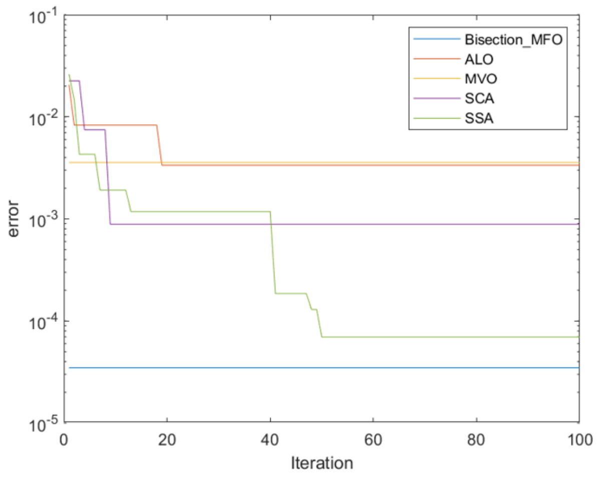


Figure 7: Shape of WOA, SSA, ALO, MVO algorithms and suggested algorithm to dissolve the fixed point of function g_2

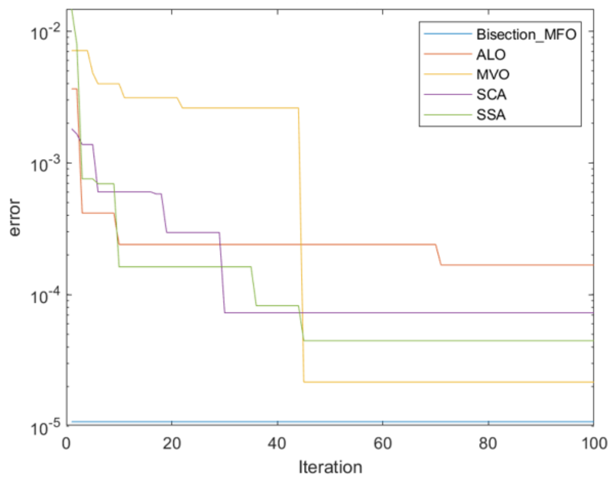


Figure 8: Shape of WOA, SSA, ALO, MVO algorithms and suggested algorithm to dissolve the fixed point of function g_3

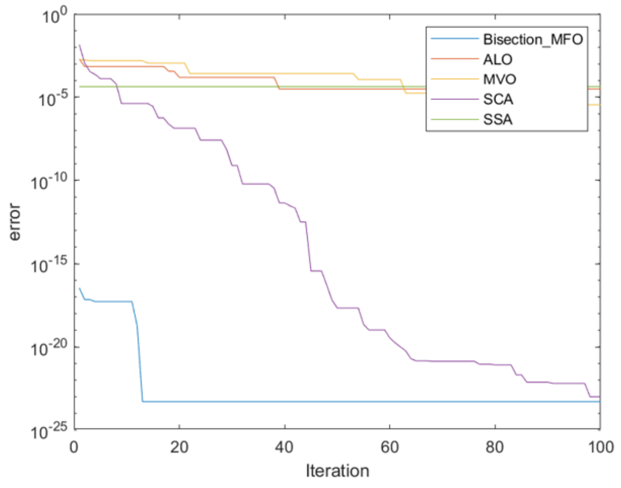


Figure 9: Shape of WOA, SSA, ALO, MVO algorithms and suggested algorithm to dissolve the fixed point of function g_4

The results and figures show that the BMFO algorithm, compared to the other four algorithms, has a better performance and a more accurate solution for finding fixed points of functions.

5 Conclusion

In present article, the main endeavor is to discover a modern repetitive procedure in order that discovery a fixed point of a function by utilizing Moth-Flame Optimization Algorithm and Bisection method. Finding the good initial value in a proportional interval where the fixed point of the function is located can sometimes be difficult for functions that are hard. In that case, they do not have a derivative or are difficult to calculate. Then obtaining solutions of their derivative is time-consuming. So finding derivative of the function $f(y) = g(y) - y$ and then finding their solutions for all functions Not recommended. MFO algorithm aids in detecting an acceptable primary solution. In the following, suggested procedure does nowhere near via the want to calculate the derivative. This suggested method is ordinary to utilize and trustworthy. While a collation via alternative methods represents, precision of this suggested algorithm is acceptable.

References

[1] A.Y. Abdelaziz, E.S. Ali and S.M. Abd Elazim, *Flower pollination algorithm and loss sensitivity factors for optimal sizing and placement of capacitors in radial distribution systems*, J. Electr. Power Energy Syst. **78** (2016), 207–214.

[2] A. Alizadegan, B. Asady and M. Ahmadpour, *Two modified versions of artificial bee colony algorithm*, Appl. Math. Comput. **225** (2013), 601–609.

[3] A.A. Alderremy, R.A. Attia, J.F. Alzaidi, D. Lu and M. Khater, *Analytical and semi-analytical wave solutions for longitudinal wave equation via modified auxiliary equation method and Adomian decomposition method*, Therm. Sci. **23** (2019), 1943–1957.

[4] E. Atashpaz-Gargari and C. Lucas, *Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition*, IEEE Cong. Evol. Comput., IEEE, 2007, pp. 4661–4667.

[5] Z. Bayraktar, M. Komurcu and D.H. Werner, *Wind Driven Optimization (WDO): A novel nature-inspired optimization algorithm and its application to electromagnetics*, IEEE Antennas Propag. Soc. Int. Symp., IEEE, 2010, pp. 1–4.

[6] R. L. Burden, and J. Douglas, Faires. Numerical analysis, BROOKS/COLE, 1985.

[7] A. Colorni, M, Dorigo and V. Maniezzo, *Distributed optimization by ant colonies*, Proc. First Eur. Conf. Artific. Life, **142** (1991), 134–142.

[8] E. Cuevas, M. Cienfuegos, D. Zaldívar and M.A. Prez-Cisneros, *A swarm optimization algorithm inspired in the behavior of the social-spider*, Expert Syst. Appl. **40** (2013), no. 18, 6374–6384.

[9] G. David Edward, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesle, 2002.

[10] L.N. De Castro, L.N. Castro and J. Timmis, *Artificial immune systems: A new computational intelligence approach*, Springer Science and Business Media, 2002.

[11] K. Deb, *Optimization for engineering design: Algorithms and examples*, PHI Learning Pvt. Ltd, 2012.

[12] A. Djerioui, A, Houari, M. Machmoum and M. Ghanes, *Grey wolf optimizerbBased Predictive torque control for electric buses applications*, Energies **13** (2020), no. 19, 5013.

[13] M. Dorigo, M. Birattari and T. Stutzle, *Ant colony optimization*, IEEE Comput. Intell. Mag. **1** (2006), no. 4, 28–39.

[14] M. Eusuff, K. Lansey and F. Pasha, *Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization*, Eng. Optim. **38** (2006), no. 2, 129–154.

[15] F. Glover, *Future paths for integer programming and links to artificial intelligence*, Comput Oper Res. **13** (1986), no. 5, 533–549.

[16] S. Harifi, M. Khalilian, J. Mohammadzadeh and S. Ebrahimnejad, *Emperor Penguins Colony: a new metaheuristic algorithm for optimization*, Evol. Intell. **12** (2019), no. 2, 211–226.

[17] J.H. Holland, *Adaptation in natural and artifficial systems*, Ann Arbor, 1975.

[18] W. Jomaa, M. Eddaly and B. Jarboui, *Variable neighborhood search algorithms for the permutation flowshop scheduling problem with the preventive maintenance*, Oper. Res. **21** (2021), no. 4, 2525–2542.

[19] D. Karaboga, *An idea based on honey bee swarm for numerical optimization*, Technical Report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[20] D. Karaboga and B. Basturk, *A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm*, J. Glob. Optim. **39** (2007), no. 3, 459–471.

[21] A.H. Kashan, *League Championship Algorithm (LCA): An algorithm for global optimization inspired by sport championships*, Appl. Soft Comput. **16** (2014), 171–200.

[22] J. Kennedy and R. Eberhart, *Particle swarm optimization*, Proc. ICNN'95-Int. Conf. Neural Networks, IEEE, Vol. 4, 1995, pp. 1942–1948.

[23] J. Kennedy, R.C. Eberhart and Y. Shi, *Swarm Intelligence*, Morgan Kaufmanns Academic Press, San Francisco, 2001.

[24] S.R. Kenneth and R.M. Storn, *Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces*, J. Glob. Optim. **11** (1997), no. 4, 341–359.

[25] M. Khater, R.A. Attia and D. Lu, *Modified auxiliary equation method versus three nonlinear fractional biological models in present explicit wave solutions*, Math. Comput. Appl. **24** (2018), no. 1, 1.

[26] M.M. Khater, C. Park, D. Lu and R.A. Attia, *Analytical, semi-analytical, and numerical solutions for the Cahn-Allen equation*, Adv. Differ. Equ. **2020** (2020), no. 1, 1–12.

[27] S. Kirkpatrick, C.D. Gelatt Jr and M.P. Vecchi, *Optimization by simulated annealing*, Science **220** (1983), no. 4598, 671–680.

[28] H. Ma and D. Simon, *Blended biogeography-based optimization for constrained optimization*, Eng. Appl. Artif. Intell. **24** (2011), no. 3, 517–525.

[29] A. Maheri, S. Jalili, Y. Hosseinzadeh, R. Khani and M. Miryahyavi, *A comprehensive survey on cultural algorithms*, Swarm Evol. Comput. **62** (2021), 100846.

[30] P. Mansouri, B. Asady and N. Gupta, *The Bisection-artificial bee colony algorithm to solve fixed point problems*, Appl. Soft Comput. **26** (2015), 143–148.

[31] F. Merrikh-Bayat, *The runner-root algorithm: a metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature*, Appl. Soft Comput. **33** (2015), 292–303.

[32] P. Mills and E. Tsang, *Guided local search for solving SAT and weighted MAX-SAT problems*, J. Autom. Reason. **24** (2000), no. 1, 205–223.

[33] S. Mirjalili, *Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems*, Neural. Comput. Appl. **27** (2016), no. 4, 1053–1073.

[34] S. Mirjalili, *Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm*, Knowledge-Based Syst. **89** (2015), 228–249.

[35] S. Mirjalili, *The ant lion optimizer*, Adv. Eng. Softw. **83** (2015), 80–98.

[36] S. Mirjalili, *SCA: A Sine Cosine Algorithm for solving optimization problems*, A School of Information and Communication Technology, Griffith University, Nathan Campus, Brisbane, QLD, 4111, 2016.

[37] S. Mirjalili and A. Lewis, *The whale optimization algorithm*, Adv. Eng. Softw. **95** (2016), 51–67.

[38] S. Mirjalili, S.M. Mirjalili and A. Hatamlou, *Multi-verse optimizer: A nature-inspired algorithm for global optimization*, Neural. Comput. Appl. **27** (2016), no. 2, 495–513.

[39] M. Misaghi and M. Yaghoobi, *Improved invasive weed optimization algorithm (IWO) based on chaos theory for optimal design of PID controller*, J. Comput. Des. Eng. **6** (2019), no. 3, 284–295.

[40] A. Ochoa, L. Margain, A. Hernandez, J. Ponce, A. De Luna, A. Hernandez and O. Castillo, *Bat Algorithm to*

improve a financial trust forest, World Cong. Nature Bio, Inspired Comput., IEEE, 2013, pp. 58–62.

[41] S. Olariu and A.Y. Zomaya, *Biology-derived algorithms in engineering optimization*, Handbook of Bioinspired Algorithms and Applications, Chapman and Hall/CRC, 2005.

[42] T. Rahkar Farshi, *Battle royale optimization algorithm*, Neural. Comput. Appl. **33** (2021), no. 4, 1139–1157.

[43] R. Rajabioun, *Cuckoo optimization algorithm*, Appl. Soft Comput. **11** (2011), no. 8, 5508–5518.

[44] E. Rashedi, H. Nezamabadi-Pour and S. Saryazdi, *GSA: a gravitational search algorithm*, Inf. Sci. Lett. **179** (2009), no. 13, 2232–2248.

[45] R.V. Rao, V.J. Savsani and D.P. Vakharia, *Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems*, Computer-aided Design **43** (2011), no. 3, 303–315.

[46] H. Rezazadeh, M. Younis, M. Eslami, M. Bilal and U. Younas, *New exact traveling wave solutions to the (2+ 1)-dimensional Chiral nonlinear Schrodinger equation*, Math. Model. Nat. Phenom. **16** (2021), 38.

[47] H. Salimi, *Stochastic fractal search: a powerful metaheuristic algorithm*, Knowledge-Based Syst. **75** (2015), 1–18.

[48] S. Saremi, S. Mirjalili and A. Lewis, *Grasshopper optimisation algorithm: theory and application*, Adv. Eng. Software **105** (2017), 30–47.

[49] H. Shah-Hosseini, *The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm*, Int. J. Bio-Inspit. Com. **1** (2009), no. 1-2, 71–79.

[50] J. Vahidi, S.M. Zekavatmand and H. Rezazadeh, *An efficient method for solving hyperbolic partial differential equations*, Fifth Nat. Conf. New Approach. Educ. Res., Mahmudabad, 2020.

[51] J. Vahidi, S.M. Zekavatmand, A. Rezazadeh, M. Inc, M.A. Akinlar and Y.M. Chu, *New solitary wave solutions to the coupled Maccaris system*, Results Phys. **21** (2021), 103801.

[52] J. Vahidi, S.M. Zekavatmand and S.M.S. Hejazi, *A modern procedure to solve fixed point functions using Bisection-Social Spider Algorithm*, Sixth Nat. Conf. New Approach. Educ. Res., Mahmudabad, 2021.

[53] J. Vahidi, S.M. Zekavatmand and S.M.S. Hejazi, *A novel way to obtain fixed point functions using sine cosine algorithm*, Sixth Nat. Conf. New Approach. Educ. Res., Mahmudabad, 2021.

[54] A.M. Wazwaz, *A sine-cosine method for handlingnonlinear wave equations*, Math. Comput. Model. Dyn. Syst. **40** (2004), no. 5-6, 499–508.

[55] A.M. Wazwaz, *The tanh method and the sine-cosine method for solving the KP-MEW equation*, Int. J. Comput. Math. **82** (2005), no. 2, 235–246.

[56] X.S. Yang, *Firefly algorithms for multimodal optimization*, Int. Symp. Stoch. Algorithms, Springer, Berlin, Heidelberg, 2009, pp. 169–178.

[57] X.S. Yang, *Nature-inspired metaheuristic algorithms*, Luniver press, 2010.

[58] S.W. Yao, S.M. Zekavatmand, H. Rezazadeh, J. Vahidi, M.B. Ghaemi and M. Inc, *The solitary wave solutions to the Klein-Gordon-Zakharov equations by extended rational methods*, AIP Adv. **11** (2021), no. 6, 065218.

[59] A. Yokus, H. Durur and H. Ahmad, *Hyperbolic type solutions for the couple Boiti-Leon-Pempinelli system*, FU Math Inf. **35** (2020), no. 2, 523–531.

[60] Y. Yonezawa and T. Kikuchi, *Ecological algorithm for optimal ordering used by collective honey bee behavior*, MHS'96 Proc. Seventh Int. Symp. Micro Machine Human Sci., IEEE, 1996, pp. 249–256.

[61] S.M. Zekavatmand, H. Rezazadeh, M. Inc, J. Vahidi and M.B. Ghaemi, *The new soliton solutions for long and short-wave interaction system*, J. Ocean Engin. Sci. In Press (2021), https://doi.org/10.1016/j.joes.2021.09.020. JOES, 2021.

[62] S.M. Zekavatmand, H. Rezazadeh and M.B. Ghaemi, *Exact travelling wave solutions of nonlinear Cahn-Allen equation*, 5th Nat. Conf. Modern Approach. Educ. Res., Mahmudabad, 2020.

[63] S.M. Zekavatmand, J. Vahidi and M.B. Ghaemi, *Obtain the fixed point of nonlinear equations through the Whale optimization algorithm*, Sixth Nat. Conf. New Approach. Educ. Res., Mahmudabad, 2021.