Alexandria Engineering Journal (2022) 61, 11579–11588

HOSTED BY

Alexandria University

Alexandria Engineering Journal

www.elsevier.com/locate/aej
www.sciencedirect.com

Novel hybrid algorithms for root determining using advantages of open methods and bracketing methods

Elsayed Badr [a,*], **Hala Attiya** [b], **Abdallah El Ghamry** [c]

[a] *Scientific Computing Department, Faculty of Computers & Artificial Intelligence, Benha University, Benha, Egypt*
[b] *Basic Science Department, Faculty of Technology and Education, Beni-Suef University, Egypt*
[c] *Computer Science Department, Faculty of Computers & Artificial Intelligence, Benha University, Benha, Egypt*

Received 6 December 2021; revised 11 April 2022; accepted 7 May 2022

KEYWORDS

Computational models;
Analysis of algorithms;
Hybrid methods;
Trisection;
Bisection;
False position;
Newton–Raphson;
Secant;
Trigonometric method

Abstract There are two numerical approaches for root determining methods: open methods and bracketing methods. Open methods usually converge much more quickly than the bracketing methods, but they may diverge in some cases. Bracketing methods always converge but they are slow compared to the open methods. In this paper, two novel blended algorithms are proposed. These algorithms have advantages of the bracketing methods and the open methods. In particular, the first hybrid algorithm consists of false-position method with modified secant method (FP-MSe) and the second blended algorithm consists of false-position method with trigonometric secant method (FP-TMSe). The numerical results show that the proposed algorithms overcome bisection (Bi) and false-position (FP) methods. On the other hand, the presented algorithms overcome the trisection (Tri), secant (Se) and Newton- Raphson (NR) algorithms according to the iteration number and the average of running time. Finally, the implementation results show the superiority of the proposed algorithms on other related algorithms.

© 2022 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Engineering, Alexandria University. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

Finding roots for non-linear equations is very important for many applications in different sciences such as engineering, mathematical chemistry, biomathematics, physics, statistics, etc.).

Numerical approaches for finding the roots of non-linear equations are divided into open methods such as Newton-Raphson, fixed point, secant and its versions and bracketing methods such as bisection, regula falsi and trisection methods. Recently, other trends have emerged, such as the blended methods between two bracketing methods. On the other hand, there is a metheuristic approach that uses the metaheuristic algorithms such as firefly, particle swarm optimization and ant colony, etc.

Open methods usually converge much more quickly than the bracketing methods, but they may diverge in some cases.

* Corresponding author.
E-mail address: badrgraph@gmail.com (E. Badr).

Peer review under responsibility of Faculty of Engineering, Alexandria University.

https://doi.org/10.1016/j.aej.2022.05.007
1110-0168 © 2022 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Engineering, Alexandria University.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Bracketing methods always converge but they are slow compared to the open methods. Unfortunately, there is no a numerical method for finding roots that has the advantages of open methods and bracketing methods although there are two attempts to find blended methods, but these methods used two methods of bracketing methods.

There is no an efficient method at all (it can find the roots of any non-linear equation), but the attempts of researchers are moving towards it. The open methods and bracketing methods have more details in [1–14].

Sabhrwal [15] presented the first blended method that combine between two bracketing method (bisection-false position). He proved that his method overcame other pure methods such as the pure bisection and the pure false position method. On other hand, Badr et al. [16] proposed the second hybrid algorithm that combine between two bracketing method (trisection-false position). They run their method on fifteen benchmark non-linear and linear equations. They deduced that their algorithm outperformed Sabhrwal's method. Also, they stressed the calculation of both the execution time and the number of iterations for each benchmark problem.

The fitness ratio function is used in the genetic algorithm for comparing the pure methods [9–11]. The authors proved that their genetic method outperformed the classical methods for solving the equations $x^2 - x - 2 = 0$ [12] and $x^2 + 2x - 7 = 0$ [11]. A novel iterative algorithm was presented in [12] to find the fixed point of a nonlinear function by Mansouri et al. They proposed a new method that uses the artificial bee colony algorithm [13] and bisection method [14] together.

The authors [17] seek to enlarge the basin of attraction of the classical Newton's by developing a relatively simple multiplicative transform of the original equations, which leads to a reduction in nonlinearity, thereby alleviating the limitation of Newton's method. Gutiérrez [18] proposed study of continuous Newton's method, which is a generic differential equation whose associated flow tends to the zeros of a given polynomial. Ya-Hsuan Hu and Takeshi Emura [19] used Newton–Raphson algorithm for parametric inference when samples are subject to double-truncation.

The reader can find more details about the pure methods, hybrid methods and metaheuristic approaches in [20–31].

The following abbreviations are used in this paper:

Bi: bisection method.
Tri: trisection method.
FP: false position method.
NR: Newton-Raphson method.
Se: secant method.
MSe: modified secant method.
TMSe: trigonometric modified secant method.
FP-MSe: hyprid method between two methods "false position and modified secant".
FP-TMSe: hyprid method between two methods "false position and trigonometric secant".
Bi-FP: hyprid method between two methods "bisection and false position" [15].
Tri-FP: hyprid method between two methods "trisection and false position" [16].

In this paper, two novel blended algorithms are proposed. These algorithms have advantages of the bracketing methods and the open methods. In particular, the first hybrid algorithm consists of false-position method with modified secant method (FP-MSe) and the second blended algorithm consists of false-position method with trigonometric secant method (FP-TMSe). The numerical results show that the proposed algorithms overcome bisection (Bi) and false-position (FP) methods. On the other hand, the presented algorithms overcome the trisection (Tri), secant (Se) and Newton- Raphson (NR) algorithms according to the iteration number and the average of running time. Finally, the implementation results show the superiority of the proposed algorithms on other blended algorithms which were proposed by Sabharwal and Badr et al.

The rest of this work is organized as follows: The classical algorithms for determining the roots of non-linear equations are proposed in Section 2. The hybrid methods for determining the roots of non-linear equations are proposed in Section 3. In Section 4, the computational study that support the superiority of the proposed algorithms are presented. Finally, conclusions and future work are proposed in Section 5.

2. Classical methods

In this section, the bracketing method which represented by "False Position" and the open methods that represented by "secant, modified secant and trigonometric secant" are proposed.

2.1. False position (Regula Falsi) method (FP) [14]

The false position method (FP) is a bracketing method for solving nonlinear equation g(s) = 0. This method is convergent. The false position method uses the two ends of the interval [*a, b*] as the initial values (s_0 = a, s_1 = b). The line joining the two points (s_0, g(s_0)), (s_1, g(s_1)) intersects the x-axis at the point s_2 which is the next approximation. The following equation gives successive approximations to the FP:

$$s_n = s_{n-1} - \frac{g(s_{n-1})(s_{n-1} - s_{n-2})}{g(s_{n-1}) - g(s_{n-2})} \quad : \quad \text{for } n > 2. \tag{1}$$

The advantages and disadvantages of FP in addition to its algorithm and some additional details are presented in the reference [16].

2.2. Secant method (Se) [14]

One of the disadvantages of Newton Raphson's method (NR) for solving nonlinear equations g(s) = 0 is finding the derivative g'(s) shown in the Eq. (2):

$$s_{i+1} = s_i - \frac{g(s_i)}{g'(s_i)} \tag{2}$$

The secant method (Sc) provided a solution to calculate the derivative g'(s) by determining an approximation to this derivative as shown in the Eq. (3):

$$g'(s_i) \cong \frac{g(s_{i-1}) - g(s_i)}{s_{i-1} - s_i} \tag{3}$$

Sc method is considered an open method like Newton-Raphson's method because there is no guarantee of reaching a solution (it may fail). The Sc method uses two initial points (s_{i-1} and s_i) to obtain an approximation of the derivative (g'(s)) as shown in the Eq. (3).

The following equation gives successive approximations to the Se:

$$s_{i+1} = s_i - \frac{s_i - s_{i-1}}{g(s_i) - g(s_{i-1})} g(s_i) \tag{4}$$

```
Algorithm 1: Secant(g, a, b, eps)
This function implements Secant method.
Input: The function (g),
    Two initial roots: a and b,
    The absolute error (eps).
Output: The root (x),
    The value of g(x)
    Numbers of iterations (n),
n:= 0
while true do
  n:= n + 1
  x:= b – g(b)*(b - a) / (g(b) – g(a))
  if |g(x)| <= eps
    return x, g(x), n
  a:= b
  b:= x
end (while)
```

Algorithm 1 uses the relation (4) to obtain the successive approximations by the secant method.

When the Sc method is compared with the other classical methods such as bisection, false position and trisection, it is faster in cases that it is convergent. On the other hand, this method may fail to get the root and therefore it belongs to the class of open methods.

The difference between FP and Sc does not appear from Eq. (1) and Eq. (4). The reader may think that the two methods are similar, but in fact they are different. The main difference between FP and Sc is how one of the initial values is replaced by the new estimate. In FP method the latest estimate of the root replaces whichever of the original values yielded a function value with the same sign as $g(s_r)$ so the two estimates always bracket the root. On the other hand, the secant method replaces the values in strict sequence, with the new value s_{i+1} replacing s_i and s_i replacing s_{i-1}. Consequently, the two values can sometimes lie on the same side of the root. For certain cases, this can lead to divergence.

2.3. Modified secant method (MSe) [14]

Rather than using two arbitrary values to estimate the derivative, an alternative approach involves a fractional perturbation of the independent variable to estimate g'(s).

$$g'(x) \cong \frac{g(x_i + \delta x_i) - g(x_i)}{\delta x_i} \tag{5}$$

where δ = a small perturbation fraction. This approximation can be substituted into Eq. (4) to yield the following iterative equation:

$$s_{i+1} = s_i - \frac{\delta s_i . g(s_i)}{g(s_i + \delta s_i) - g(s_i)} \tag{6}$$

2.4. Trigonometric modified secant method (TMSe)

Srivastav et al. [27] presented a new algorithm to find a nonzero real root of the transcendental equations using trigonometric formula (see Fig. 1). In fact, the new proposed algorithm is based on the combination of inverse of sine series and "NR". They deduce that the trigonometric modified secant overcame Newton Raphson method. The new trigonometric iterative formula under consideration is proposed as.

$$s_{i+1} = s_i \left[1 + \sin^{-1}\left(\frac{-g(s_i)}{s_i g'(s_i)} \right) \right] \tag{7}$$

Here, we proposed the new trigonometric modified secant (TMSe) by using Eq. (8) in Eq. (7) to compute the derivative g'(s).

$$g'(s) \cong \frac{g(s_i + \delta s_i) - g(s_i)}{\delta s_i} \tag{8}$$

3. The proposed hybrid methods

How to make hybrid algorithms are proposed in this section. In particular, the hybrid algorithms False position – Modified secant (FP-MSe) and False position – trigonometric secant (FP-TMSe) are proposed in this section.

3.1. Hybrid method (False position – Modified secant) (FP-MSe)

The main objective of this hybrid algorithm (FP-MSc) is to combine the advantages of open methods and bracketing methods. On the other hand, we hope that the FP-MSc will be fast and convergent.

```
Algorithm 2: False Modified Secant(f, a, b, eps)
Input: The function (f),
    The interval [a, b] where the root lies in,
    The absolute error (eps).
Output: The root (x),
    The value of f(x)
    Numbers of iterations (n),
    The interval [a, b] where the root lies in
n:= 0
while true do
    n:= n + 1;
    x = a – (f(a)*(b-a))/(f(b)-f(a))
    if |f(x)| <= eps
      return x, f(x), n, a, b
    else
      xS:= x - (δ * f(x)) / (f(x + δ) - f(x))
      if |f(xS)| < |f(x)| and xS ∈ (a, b)
        if f(a) * f(xS) < 0
          b:= xS
        else
          a:= xS
      else
        if f(a) * f(x) < 0
          b:= x
        else
          a:= x
end (while)
```

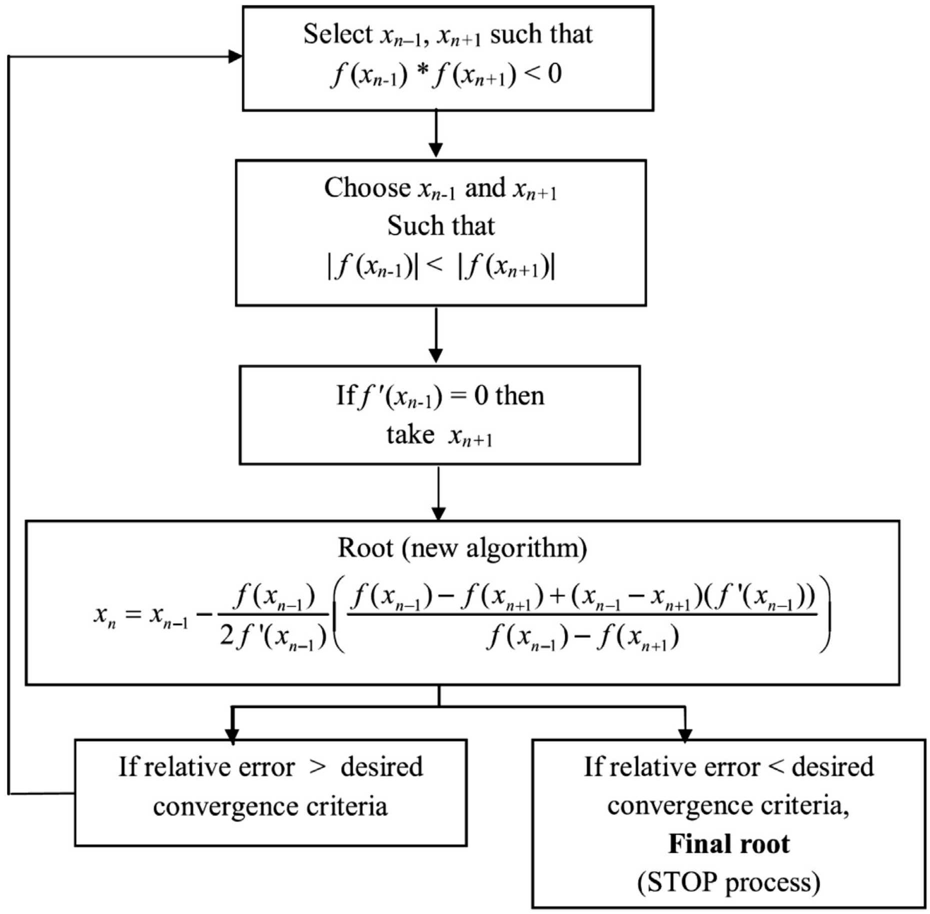

Fig. 1 Flow chart for trigonometric modified secant method.

FP-MSc starts with calculating the root using the false position method, taking into account the achievement of the *eps* error rate, and then FP-MSc calculates the root using the secant method, taking into account the achievement of the *eps* error rate. FP-MSc compares the value of the two roots, chooses the closest root within the interval. FP-MSc repeats the previous steps in each iteration. It is noticed that if the secant method fails at a certain point (because it follows the open methods), we take the value of the root that was calculated in the false position and then the secant method resumes its work in the next iterations. Thus, FP-MSc benefited from the advantages of the secant method in every iteration except for the iterations in which the secant method failed.

Algorithm 2 uses hybrid method (False position – Modified secant) FP-MSe to obtain the successive approximations by the secant method.

3.2. Hybrid method (False position – Trigonometric modified secant) (FP-TMSe)

The main objective of this hybrid algorithm false position – trigonometric modified secant (FP-TMSc) is to combine the advantages of open methods and bracketing methods. On the other hand, we hope that the FP-MSc will be fast and convergent.

Algorithm 3: False Trig Modified Secant (*f*, *a*, *b*, *eps*)

Input: The function (f), The interval $[a, b]$ where the root lies in, The absolute error (eps).
Output: The root (x), The value of $f(x)$, Numbers of iterations (n), The interval $[a, b]$ where the root lies in

```
n:= 0
while true do
    n:= n + 1;
    x = a – (f(a)*(b-a))/(f(b)-f(a))
    if |f(x)| < = eps
      return x, f(x), n, a, b
    else
      g(x):= (f(x + δ) – f(x)) / δ
      xS = x * (1 + sin^-1(-f(x) / (x * g(x))))
      if |f(xS)| < |f(x)| and xS ∈ (a, b)
        if f(a) * f(xS) < 0
          b:= xS
        else
          a:= xS
      else
        if f(a) * f(x) < 0
          b:= x
        else
          a:= x
end (while)
```

Table 1 Fifteen standard nonlinear equations.

No.	Problem	Intervals	References
P1	$x^2 - 3$	[1,2]	Harder [32]
P2	$x^2 - 5$	[2,7]	Srivastava [9]
P3	$x^2 - 10$	[3,4]	Harder [32]
P4	$x^2 - x - 2$	[1,4]	Moazzam [10]
P5	$x^2 + 2x - 7$	[1,3]	Nayak [11]
P6	$x^3 - 2$	[0, 2]	Harder [32]
P7	$xe^x - 7$	[0, 2]	Callhoun [33]
P8	$x - \cos(x)$	[0, 1]	Ehiwario [6]
P9	$x\sin(x) - 1$	[0, 2]	Mathews [34]
P10	$x\cos(x) + 1$	[−2, 4]	Esfandiari [35]
P11	$x^{10} - 1$	[0, 1.3]	Chapra [24]
P12	$x^2 + e^{x/2} - 5$	[1,2]	Esfandiari [35]
P13	$\sin(x)\sinh(x) + 1$	[3,4]	Esfandiari [35]
P14	$e^x - 3x - 2$	[2,3]	Hoffman [36]
P15	$\sin(x) - x^2$	[0.5, 1]	Chapra [24]

FP-TMSc starts with calculating the root using the false position method, taking into account the achievement of the *eps* error rate, and then FP-MSc calculates the root using the secant method, taking into account the achievement of the *eps* error rate. FP-TMSc compares the value of the two roots, chooses the closest root within the interval. FP-TMSc repeats the previous steps in each iteration. It is noticed that if the secant method fails at a certain point (because it follows the open methods), we take the value of the root that was calculated in the false position and then the secant method resumes its work in the next iterations. Thus, FP-TMSc benefited from the advantages of the secant method in every iteration except for the iterations in which the secant method failed.

Algorithm 3 uses hybrid method (false position – trigonometric modified secant) FP-TMSe to obtain the successive approximations by the secant method.

3.3. Complexity analysis

The proposed algorithms are FP-MSe and FP-TMSe. FP-MSe is a hybrid of the false position and modified secant methods while FP-TMSe is a hybrid of the false position and trigonometric modified secant methods. The absolute error, the used function and the used algorithm are three criteria effect on the number of iterations. For functions on interval [a, b] with the bisection method, the upper bound of $n_b(\in)$ the number of iterations can be predetermined from$\log_2(b-a)/\in$. Although the Newton-Raphson and secant approaches show quadratic convergence, they are not guaranteed to converge unless the functions are subjected to additional constraints. The genetic algorithm has the complexity that closes to the false position [10]. The number of iterations $n_f(\in)$ for false position method cannot be predetermined because of the convexity and the root near the endpoint of the bracketing interval.

The number of iterations of the proposed algorithms FP-MSe and FP-TMSe is less than min $(n_f(\in),n_m(\in))$ and min $(n_f(\in),n_t(\in))$ where $n_m(\in)$ and $n_t(\in)$ are the number of iterations for modified secant and trigonometric modified secant method respectively. This is supported by the algorithm and corroborated by Section 4′s empirical study..

4. Empirical study

The empirical results of the classical numerical algorithms Bi, Tri, FP, NR, Se, MSe and TMSe are presented. On the other hand, empirical results for the hybrid methods FP-MSe, FP-TMSe, Bi-FP and Tri-FP are presented. The comparison between classical methods and blended methods are proposed (according to the averages of both running time and iterations number. Fifteen standard nonlinear equations are used to apply this comparison. Each code runs ten times on the equation and the averages of both running time and iterations number are listed.

Table 2 Results of 15 equations using FP.

Problem	False Position Method (FP)					
	Iter	Average CPU Time	Approximate Root	Function Value	Lower Bound	Upper Bound
P1	12	0.134719	1.7320508075688599	0.0000000000000000	1.7320508075686347	2.0000000000000000
P2	46	0.510051	2.2360679774997747	0.0000000000000000	2.2360679774997609	7.0000000000000000
P3	14	0.155169	3.1622776601683644	0.0000000000000000	3.1622776601682516	4.0000000000000000
P4	34	0.382718	1.9999999999999558	0.0000000000000000	1.9999999999998894	4.0000000000000000
P5	20	0.556411	1.8284271247461896	−0.0000000000000027	1.8284271247461874	3.0000000000000000
P6	40	0.455044	1.2599210498948719	−0.0000000000000062	1.2599210498948701	2.0000000000000000
P7	29	0.330403	1.5243452049841437	−0.0000000000000075	1.5243452049841419	2.0000000000000000
P8	11	0.124456	0.7390851332151551	−0.0000000000000092	0.7390851332150500	1.0000000000000000
P9	6	0.078088	1.1141571408719306	0.0000000000000008	1.0997501702946164	1.1141571408730828
P10	12	0.138026	2.0739328090912146	0.0000000000000007	2.0739328090912039	2.5157197710146586
P11	127	1.447224	0.9999999999999812	0.0000000000000000	0.9999999999999755	1.3000000000000000
P12	15	0.176429	1.6490132683031899	−0.0000000000000008	1.6490132683031871	2.0000000000000000
P13	44	0.507856	3.2215883990939416	0.0000000000000063	3.2215883990939407	4.0000000000000000
P14	44	0.504918	2.1253911988111285	−0.0000000000000079	2.1253911988111267	3.0000000000000000
P15	16	0.185620	0.8767262153950552	0.0000000000000080	0.8767262153950091	1.0000000000000000

Table 3 Results of 15 equations using Se.

NO.	Secant Method (Se)			
	Iter	Average CPU Time	Approximate Root	Function Value
P1	6	0.068071	1.7320508075688772	0.0000000000000000
P2	7	0.077763	2.2360679774997898	0.0000000000000000
P3	5	0.055822	3.1622776601683764	0.0000000000000000
P4	8	0.089142	2.0000000000000000	0.0000000000000000
P5	6	0.066767	1.8284271247461907	0.0000000000000036
P6	10	0.114367	1.2599210498948716	−0.0000000000000073
P7	9	0.101345	1.5243452049841444	0.0000000000000002
P8	6	0.067514	0.7390851332151607	0.0000000000000001
P9	5	0.073486	1.1141571408719304	0.0000000000000004
P10	8	0.091206	2.0739328090912150	−0.0000000000000003
P11	**Fail**			
P12	6	0.069605	1.6490132683031902	0.0000000000000002
P13	8	0.093442	3.2215883990939420	0.0000000000000004
P14	7	0.081160	2.1253911988111298	−0.0000000000000007
P15	7	0.080784	0.8767262153950625	−0.0000000000000000

The characteristics of the used environment are MATLAB v7.01 Software, 64-bit Window 8.1 Operating System, Core (TM)i5 CPU M 460 @2.53 GHz, 4.00 GB of memory.

4.1. Dataset and evaluation metrics

The absolute error (eps) and the number of iterations are two methods for terminating numerical computations. The absolute error (eps = 10^{-14}) is used to terminate all algorithms in this study. There may be a technique with a minimal number of iterations but a long execution time, or vice versa. As a result, the number of iterations and the running time are key parameters to consider while evaluating algorithms. Unfortunately, most researchers ignored the finer points of determining the running duration. Furthermore, they did not discuss or respond to the following question: why does the running time of the used software package differ from one run to the next? As a result, we execute each algorithm 10 times and average the running times to obtain accurate running times and eliminate operating system issues.

We utilize fifteen benchmark functions (Table 1) to assess the proposed technique because it is inaccurate to draw conclusions from just one function.

Tables 3–5 represent the classical methods while Tables 6–7 represent the presented methods (hyprid methods FP-MSe and FP-TMSe). Table 8 summaries the results of 15 equations using Bi, Tri, FP, NR, Sc, MSc, TSc, Tri-FP [16], Bi-FP [15], FP-MSc and FP-TSc according iterations number. On the other hand, Table 9 summaries the results of 15 equations using Bi, Tri, FP, NR, Sc, MSc, TSc, Tri-FP [16], Bi-FP [15], FP-MSc and FP-TSc according CPU time.

From Tables 2–5, we deduce that Se, MSe and TMSe may fail in some cases while FP method always finds the root. Tables 6–7 show that the proposed methods FP-MSe and FP-TMSe always determine the root. Table 8 and Fig. 2 show that the proposed methods FP-MSe and FP-TMSe overcome

Table 4 Results of 15 equations using MSe.

NO.	Modified Secant Method (MSe)			
	Iter	Average CPU Time	Approximate Root	Function Value
P1	7	0.055600	1.7320508075688772	0.0000000000000000
P2	6	0.046685	2.2360679774997898	0.0000000000000000
P3	5	0.038694	3.1622776601683800	0.0000000000000000
P4	7	0.054179	2.0000000000000751	0.0000000000000000
P5	6	0.048135	1.8284271247461901	0.0000000000000074
P6	**Fail**			
P7	15	0.120189	1.5243452049841444	0.0000000000000002
P8	7	0.055538	0.7390851332151607	0.0000000000000001
P9	**Fail**			
P10	14	0.110470	−4.9171859252871322	−0.0000000000000034
P11	**Fail**			
P12	7	0.056755	1.6490132683031902	0.0000000000000002
P13	7	0.056114	3.2215883990939420	0.0000000000000004
P14	6	0.048102	2.1253911988111298	0.000000000000017
P15	10	0.078752	0.8767262153950625	−0.0000000000000000

Table 5 Results of 15 equations using TMSe.

NO.	Trig Modified Secant method (TMSe)			
	Iter	Average CPU Time	Approximate Root	Function Value
P1	7	0.054225	1.7320508075688816	0.0000000000000000
P2	6	0.046937	2.2360679774997898	0.0000000000000000
P3	5	0.038868	3.1622776601683800	0.0000000000000000
P4	8	0.064959	2.0000000000000000	−0.0000000000000000
P5	7	0.056603	1.8284271247461901	−0.0000000000000002
P6	**Fail**			
P7	**Fail**			
P8	**Fail**			
P9	**Fail**			
P10	10	0.083605	−0.8981130139865969	−0.0000000000000005
P11	**Fail**			
P12	7	0.057141	1.6490132683031902	0.0000000000000002
P13	7	0.055901	3.2215883990939420	0.0000000000000004
P14	6	0.047446	2.1253911988111298	0.0000000000000017
P15	10	0.078693	0.8767262153950625	−0.0000000000000000

Table 6 Results of 15 equations using FP-MSe.

NO.	False Position-Modified Secant (FP-MSc)					
	Iter	Average CPU Time	Approximate Root	Function Value	Lower Bound	Upper Bound
P1	4	0.069575	1.7320508075688772	0.0000000000000000	1.7320508075686911	1.7320508316261378
P2	4	0.069612	2.2360679774997836	0.0000000000000000	2.2360679774997836	2.2360679823646143
P3	3	0.050055	3.1622776601683791	0.0000000000000000	3.1622776601439160	3.1623373524252090
P4	5	0.088835	2.0000000000000000	0.0000000000000000	1.9999999999983424	2.0000000995063365
P5	4	0.068722	1.8284271247461901	−0.0000000000000002	1.8284271247458566	1.8284272347682684
P6	7	0.126987	1.2599210498948732	0.0000000000000001	1.2599210498843008	1.2599211933114407
P7	6	0.111020	1.5243452049841444	0.0000000000000002	1.5243452032440634	1.5243452049841444
P8	4	0.070539	0.7390851332151607	0.0000000000000001	0.7390851332151492	0.7390851353889391
P9	4	0.076844	1.1141571408719302	0.0000000000000001	1.1141571408719300	2.0000000000000000
P10	5	0.088867	2.0739328090912150	−0.0000000000000003	2.0739328090885363	2.0740081831508101
P11	22	0.419434	0.9999999999999998	0.0000000000000000	0.9999999999035674	1.0000004609524096
P12	4	0.073829	1.6490132683031902	0.0000000000000002	1.6490132683030334	1.6490132923605685
P13	5	0.091479	3.2215883990939420	0.0000000000000004	3.2215883990939420	3.2215975484631514
P14	4	0.071120	2.1253911988111298	−0.0000000000000007	2.1253911988106498	2.1253912583967129
P15	5	0.092020	0.8767262153950625	−0.0000000000000000	0.8767262127372651	0.8767262153950625

Table 7 Results of 15 equations using FP-TMSe.

NO.	False Position-Trigonometric Modified Secant (FP-TMSc)					
	Iter	Average CPU Time	Approximate Root	Function Value	Lower Bound	Upper Bound
P1	4	0.069034	1.7320508075688772	0.0000000000000000	1.7320508075688510	1.7320508415725790
P2	4	0.069209	2.2360679774997885	0.0000000000000000	2.2360679774997885	2.2360679869226434
P3	3	0.050180	3.1622776601683795	0.0000000000000000	3.1622776601662728	3.1623377560861936
P4	5	0.089779	2.0000000000000000	0.0000000000000000	1.9999999999996521	2.0000002110988420
P5	4	0.069483	1.8284271247461901	−0.0000000000000002	1.8284271247461481	1.8284272628503575
P6	7	0.128832	1.2599210498948732	0.0000000000000001	1.2599210498948628	1.2599210531443681
P7	6	0.109441	1.5243452049841444	0.0000000000000002	1.5243452049841384	1.5243452068482486
P8	4	0.072784	0.7390851332151607	0.0000000000000001	0.7390851332151573	0.7390851396067568
P9	4	0.070942	1.1141571408719302	0.0000000000000001	1.1141571408719300	2.0000000000000000
P10	5	0.090834	2.0739328090912150	−0.0000000000000003	2.0739328090907851	2.0740364649000012
P11	22	0.412652	1.0000000000000000	0.0000000000000000	0.9999999999869418	1.0000006419525811
P12	4	0.070093	1.6490132683031902	0.0000000000000002	1.6490132683031677	1.6490133025240414
P13	4	0.072612	3.2215883990939420	0.0000000000000004	3.2215883990777541	3.2215978802910361
P14	4	0.071659	2.1253911988111298	−0.0000000000000007	2.1253911988110628	2.1253912824462167
P15	5	0.092131	0.8767262153950625	−0.0000000000000000	0.8767262153950038	0.8767262210882244

Table 8 Results of 15 equations using Bi, Tri, FP, NR, Sc, MSc, TSc, Tri-FP [16], Bi-FP [15], FP-MSc and FP-TSc according iterations number.

NO.	Bi	Tri	FP	NR	Sc	MSc	TSc	Tri-FP [16]	Bi-FP [15]	FP-MSc	FP-TMSc
P1	44	26	12	6	6	7	7	7	8	4	4
P2	44	28	46	5	7	6	6	8	10	4	4
P3	44	28	14	5	5	5	5	6	7	3	3
P4	45	1	34	7	8	7	8	1	2	5	5
P5	48	29	20	6	6	6	7	7	5	4	4
P6	49	30	40	**fail**	10	**fail**	**fail**	8	9	7	7
P7	46	31	29	14	9	15	**fail**	7	11	6	6
P8	44	29	11	6	6	7	**fail**	7	8	4	4
P9	46	28	6	**fail**	5	**fail**	**fail**	5	6	4	4
P10	45	28	12	14	8	14	10	8	10	5	5
P11	44	26	127	**fail**	**fail**	**fail**	**fail**	9	12	22	22
P12	48	26	15	6	6	7	7	6	8	4	4
P13	48	31	44	6	8	7	7	7	9	5	4
P14	46	28	44	5	7	6	6	7	9	4	4
P15	45	29	16	9	7	10	10	5	7	5	5

Table 9 Results of 15 equations using Bi, Tri, FP, NR, Sc, MSc, TSc, Tri-FP [16], Bi-FP [15], FP-MSc and FP-TSc according CPU time.

NO.	Bi	Tri	FP	NR	Sc	MSc	TSc	Tri-FP [16]	Bi-FP [15]	FP-MSc	FP-TSc
P1	0.514839	0.292349	0.134719	0.240500	0.068071	0.055600	0.054225	0.131418	0.121232	0.069575	0.069034
P2	0.339006	0.311319	0.510051	0.186819	0.077763	0.046685	0.046937	0.149270	0.148720	0.069612	0.069209
P3	0.330300	0.312939	0.155169	0.185136	0.055822	0.038694	0.038868	0.111231	0.103442	0.050055	0.050180
P4	0.339274	0.011161	0.382718	0.244393	0.089142	0.054179	0.064959	0.018535	0.030053	0.088835	0.089779
P5	0.413062	0.330426	0.556411	0.214349	0.066767	0.048135	0.056603	0.131906	0.074941	0.068722	0.069483
P6	0.373710	0.341553	0.455044	**fail**	0.114367	**fail**	**Fail**	0.152130	0.137275	0.126987	0.128832
P7	0.381111	0.349806	0.330403	0.436972	0.101345	0.120189	**Fail**	0.131670	0.165907	0.111020	0.109441
P8	0.345850	0.326833	0.124456	0.214512	0.067514	0.055538	**Fail**	0.131345	0.120322	0.070539	0.072784
P9	0.556300	0.773690	0.078088	**fail**	0.073486	**fail**	**fail**	0.213409	0.101488	0.076844	0.070942
P10	0.454494	0.316154	0.138026	0.439575	0.091206	0.110470	0.083605	0.150378	0.150611	0.088867	0.090834
P11	0.338134	0.297432	1.447224	**fail**	**fail**	**fail**	**fail**	0.170638	0.184145	0.419434	0.412652
P12	0.379392	0.299995	0.176429	0.221387	0.069605	0.056755	0.057141	0.115872	0.123939	0.073829	0.070093
P13	0.390438	0.360716	0.507856	0.221157	0.093442	0.056114	0.055901	0.135481	0.139654	0.091479	0.072612
P14	0.354950	0.323873	0.504918	0.191465	0.081160	0.048102	0.047446	0.134990	0.139775	0.071120	0.071659
P15	0.359546	0.334640	0.185620	0.306000	0.080784	0.078752	0.078693	0.096275	0.107493	0.092020	0.092131

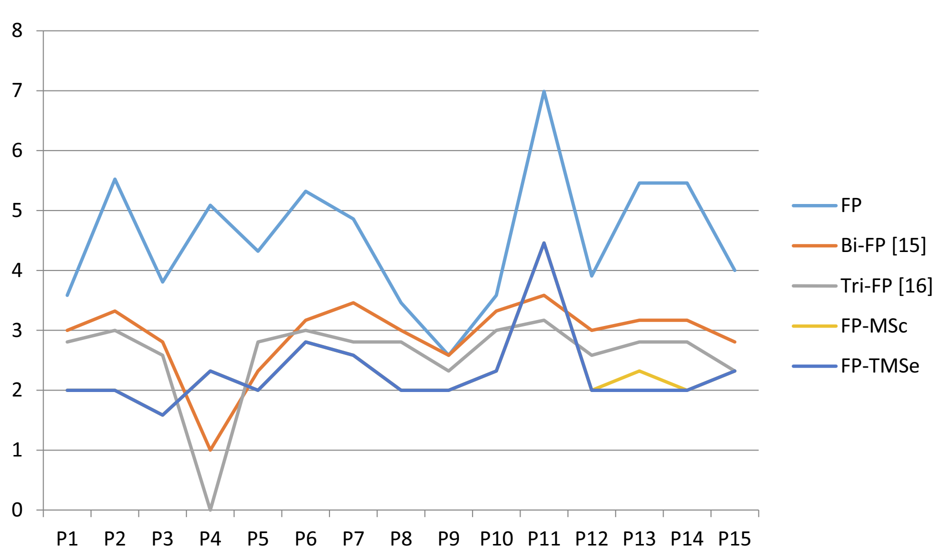

Fig. 2 A comparison among 5 methods on fifteen benchmark problems according to $\log_2$ (iterations number).

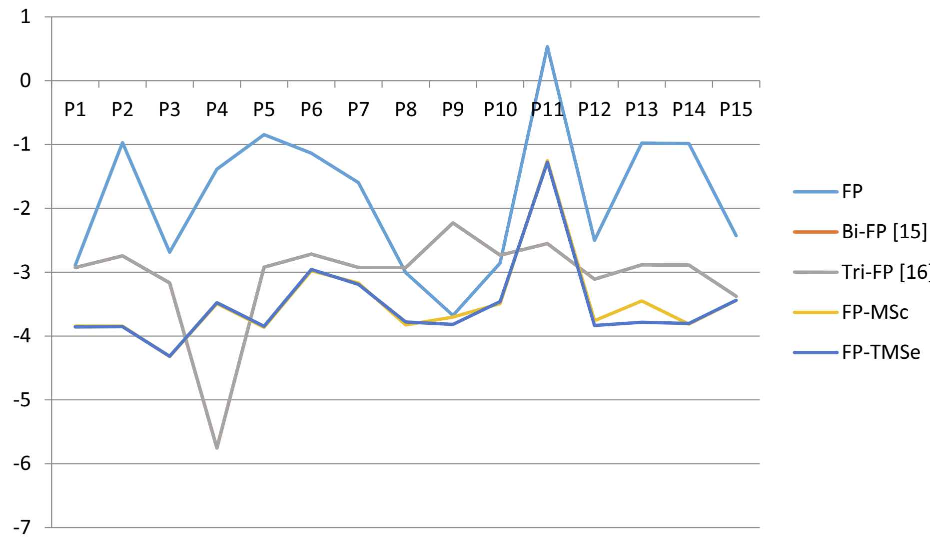

Fig. 3 A comparison among 5 methods on fifteen benchmark problems according to $\log_2$ (CPU time).

FP, Bi and Tri methods (They achieved 100% and 100% accuracy rates according to the number of iterations). On the other hand, according to the number of iterations, the proposed methods FP-MSe and FP-TMSe achieved 87% accuracy rates but FP-Bi method [15] and FP-Tri method [16] achieved 13% of fifteen problems. Finally, the proposed method FP-TMSe overcomes the proposed method FP-MSe by one problem only.

Table 9 and Fig. 3 show that the proposed methods FP-MSe and FP-TMSe overcome FP, Bi and Tri methods (They achieved 100% and 100% accuracy rates according to CPU time). On the other hand, according to CPU time, the proposed methods FP-MSe and FP-TMSe achieved 87% accuracy rates but FP-Bi method [15] and FP-Tri method [16] achieved 13% of fifteen problems.

5. Conclusion and future works

In this paper, two novel blended algorithms were proposed. These algorithms have advantages of the bracketing methods and the open methods. In particular, the first hybrid algorithm consists of false-position method with modified secant method (FP-MSe) and the second blended algorithm consists of false-position method with trigonometric secant method (FP-TMSe). The numerical results showed that the proposed algorithms overcame bisection (Bi) and false-position (FP) methods. On the other hand, the presented algorithms overcame the trisection (Tri), secant (Se) and Newton- Raphson (NR) algorithms according to the iteration number and the average of running time. Finally, the implementation results showed the superiority of the proposed algorithms on other blended algorithms which were proposed by Sabharwal and Badr *et al*. We are studying the possibility of using the proposed algorithms in software packages such as Matlab, Python, Mathematica, etc.

In future work, the metaheuristic approach [37–43] will apply with different methods for root determining of transcendental equations. The proposed algorithms will evaluate by using different nonlinear equations.

6. Data availability

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

Declaration of Competing 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.

References

[1] A. Hasan, Numerical study of some iterative methods for solving nonlinear equations, Int. J. Eng. Sci. Invent. 5 (2016) 1–10.
[2] A. Hasan, N. Ahmad, Compartive study of a new iterative method with that Newtons Method for solving algebraic and transcesental equations, Int. J. Comput. Math. Sci. 4 (2015) 32–37.
[3] M.Q. Khirallah, M.A. Hafiz, Solving system of nonlinear equations using family of jarratt methods. Int. J. Differ. Equ. Appl. 2013, 12, 69–83. [CrossRef].
[4] C. Remani, Numerical Methods for Solving Systems of Nonlinear Equations, Lakehead University, Thunder Bay, ON, Canada, 2012, p. 13.
[5] C.H. Lally, A faster, high precision algorithm for calculating symmetric and asymmetric. arXiv 2015, arXiv:1509.01831.
[6] J.C. Ehiwario, S.O. Aghamie, Comparative Study of Bisection, Newton-Raphson and Secant Methods of Root-Finding Problems, IOSR J. Eng. 4 (2014) 1–7.
[7] S. Ait-Aoudia, I. Mana, Numerical solving of geometric constraints by bisection: A distributed approach, Int. J. Comput. Inf. Sci. 2 (2004) 66.
[8] S. Baskar, S.S. Ganesh, Introduction to Numerical Analysis; Department of Mathematics, Indian Institute of Technology Bombay Powai, Mumbai, India, 2016.
[9] R.B. Srivastava, S. Srivastava, Comparison of numerical rate of convergence of bisection, Newton and secant methods, J. Chem. Biol. Phys. Sci. 2 (2011) 472–479.
[10] G. Moazzam, A. Chakraborty, A. Bhuiyan, A robust method for solving transcendental equations, Int. J. Comput. Sci. Issues 9 (2012) 413–419.

[11] T. Nayak, T. Dash, Solution to quadratic equation using genetic algorithm, in: Proceedings of the National Conference on AIRES-2012, Vishakhapatnam, India, 29–30 June 2012.
[12] P. Mansouri, B. Asady, N. Gupta, The Bisection–Artificial Bee Colony algorithm to solve Fixed point problems, Appl. Soft Comput. 26 (2015) 143–148.
[13] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J Glob Optim 39 (3) (2007) 459–471.
[14] L.R. Burden, F.J. Douglas, Numerical Analysis, Prindle, Weber & Schmidt, 3rd ed., Amazon, Seattle, WA, USA, 1 January 1985.
[15] C.L. Sabharwal, Blended root finding algorithm outperforms bisection and Regula Falsi Algorithms, Mathematics 7 (11) (2019) 1118, https://doi.org/10.3390/math7111118.
[16] E. Badr, S. Almotairi, A.E. Ghamry, A comparative study among new hybrid root finding algorithms and traditional methods, Mathematics 9 (2021) 1306, https://doi.org/10.3390/math9111306.
[17] A. Aggarwal, S. Pant, Beyond Newton: A new root-finding fixed-point iteration for nonlinear equations, Algorithms 13 (2020) 78, https://doi.org/10.3390/a13040078.
[18] J.M. Gutiérrez, Numerical properties of different root-finding algorithms obtained for approximating continuous Newton's method, Algorithms 8 (2015) 1210–1218, https://doi.org/10.3390/a8041210.
[19] Y.H. Hu, T. Emura, Maximum likelihood estimation for a special exponential family under random double-truncation, Comput. Stat. 30 (2015) 1199–1229, https://doi.org/10.1007/s00180-015-0564-z.
[20] Iain L. MacDonald, Does Newton-Raphson really fail?, Statist Methods Med. Res. 23 (3) (2014) 308–311.
[21] E. Badr, H. Elgendy, A hybrid water cycle-particle swarm optimization for solving the fuzzy underground water confined steady flow, Indones. J. Electr. Eng. Comput. Sci. 19 (2020) 492–504. [CrossRef].
[22] E. Badr, S. Almotairi, M.A. Salam, H. Ahmed, New sequential and parallel support vector machine with grey wolf optimizer for breast cancer diagnosis, Alexandria Eng. J. 61 (3) (2022) 2520–2534.
[23] E. Badr, M. Abdul Salam, S. Almotairi, H. Ahmed, R. Natella, From linear programming approach to metaheuristic approach: scaling techniques, Complexity 2021 (2021) 1–10.
[24] S.C. Chapra, R.P. Canale, Numerical Methods for Engineers, 7th ed., McGraw-Hill, Boston, MA, USA, 2015.
[25] M. Heydari, S.M. Hosseini, G.B. Loghmani, On two new families of iterative methods for solving nonlinear equations with optimal order, Applicable Anal. Discrete Math. 5 (1) (2011) 93–109.
[26] M. Heydari, S.M. Hosseini, G.B. Loghmani, Convergence of a family of third-order methods free from second derivatives for finding multiple roots of nonlinear equations, World Appl. Sci. J. 11 (5) (2010) 507–512.
[27] M.M. Sehati, S.M. Karbassi, M. Heydari, G.B. Loghmani, Several new iterative methods for solving nonlinear algebraic equations incorporating homotopy perturbation method (HPM), Int. J. Phys. Sci. 7 (27) (2012) 5017–5025.
[28] M. Heydari, G.B. Loghmani, Third-Order and Fourth-Order Iterative Methods Free from Second Derivative for Finding Multiple Roots of Nonlinear Equations, Caspian J. Math. Sci. (CJMS) peer 3 (1) (2014) 67–85.
[29] H. Garg, A hybrid PSO-GA algorithm for constrained optimization problems, Appl. Math. Comput. 274 (1) (February 2016) 292–305.
[30] H. Garg, A hybrid GSA-GA algorithm for constrained optimization problems, Inform. Sci. 478 (April 2019) 499–523.
[31] T. Kundu, H. Garg, A hybrid ITLHHO algorithm for numericaland engineering optimization problems, Int. J. Intell. Syst. 2021; 1–81. doi:10.1002/int.22707.
[32] D.W. Harder, Numerical Analysis for Engineering. Available online: https://ece.uwaterloo.ca/~\{\}dwharder/NumericalAnaly-sis/ 10RootFinding/falseposition/ (accessed on 11 June 2019).
[33] D. Calhoun, Available online: https://math.boisestate.edu/~\{\}calhoun/teaching/matlab- tutorials/lab_16/html/lab_16.html (accessed on 13 June 2019).
[34] J.H. Mathews, K.D. Fink, Numerical Methods Using Matlab, 4th ed., Prentice-Hall Inc., Upper Saddle River, NJ, USA, 2004.
[35] R.S. Esfandiari, Numerical Methods for Engineers and Scientists Using MATLAB, CRC Press, Boca Raton, FL, USA, 2013.
[36] D.H. Joe, Numerical Methods for Engineers and Scientists, 2nd ed., CRC Press, Boca Raton, FL, USA, 2001.
[37] A. Demir, Trisection method by k-Lucas numbers, Appl. Math. Comput. 198 (1) (2008) 339–345.
[38] V.K. Srivastav, S. Thota, M. Kumar, A new trigonometrical algorithm for computing real root of non-linear transcendental equations, Int. J. Appl. Comput. Math 5 (2) (2019), https://doi.org/10.1007/s40819-019-0600-8.
[39] E.M. Badr, H. elgendy, A Hybrid water cycle - particle swarm optimization for solving the fuzzy underground water confined steady flow, Indonesian J. Electr. Eng. Comput. Sci. 19(1) 2020.
[40] D. Salama AbdELminaam, A.M. Almansori, M. Taha, E. Badr, S. Mirjalili, A deep facial recognition system using computational intelligent algorithms, PLoS ONE 15 (12) (2020), https://doi.org/10.1371/journal.pone.0242269.
[41] Elsayed Badr, Mustafa Abdul Salam, Sultan Almotairi, Hagar Ahmed, From linear programming approach to metaheuristic approach: scaling techniques, Complexity vol. 2021, Article ID 9384318, 10 pages, 2021.
[42] K. Jain, A. Saxena, Simulation on supplier side bidding strategy at day-ahead electricity market using ant lion optimizer, J. Computat. Cognit. Eng. 00 (00) (2022) 1–12.
[43] Jie Zan, Research on robot path perception and optimization technology based on whale optimization algorithm, J. Computat. Cognit. Eng. Vol. 00(00) (2022) 1–8.