AN IMPROVED BRACKETING METHOD FOR NUMERICAL SOLUTION OF NONLINEAR EQUATIONS BASED ON RIDDERS METHOD

Shnyar Karim Rahman, Deni Adnan Mohammed, Bahadin Muhammad Hussein, Berivan Azad Salam, Khadija Rzgar Mohammed, Bawar Mohammed Faraj*

Department of Physics, College of Science, University of Halabja, Halabja, 46018, Iraq

Computer Science department, College of Science, University of Halabja, Halabja, 46018, Iraq

*Corresponding Author Email: bawarm.faraj@uoh.edu.iq

ABSTRACT

In the present study numerical solution for non linear equations have been studied. Ridders method have been discussed. An improvement of Ridders method with combination of Bisection and newton Raphson methods have been presented. An algorithm for the proposed method have been stated. Moreover, several examples are included to demonstrate the validity and applicability of the presented technique. Matlab program involved for numerical computations. The proposed method applied for given examples. The error analysis table presents the obtained numerical results. The numerical solutions which found by Matlab program has good results in terms of accuracy.

KEYWORDS

Ridders Method, Root finding, Nonlinear equation, Matlab.

1. INTRODUCTION

The problem of finding the real or complex roots of a nonlinear equation is frequently encountered in scientific work. Nonlinear problems are of interest to engineers, biologists, physicists, mathematicians, and many other scientists because most systems are inherently nonlinear in nature (El Naschie, 2008; Shiinohara, 1972; Anderson and Björck, 1973; Le, 1985; Le, 1988; Park and Hitotumatsu, 1988; Abbashandy and Liao, 2008; Galdino, 2011; Muangchana and Somporn, 2011; Buhit et al., 2013; Hafiz, 2013; Remani, 2013; Stage, 2013).

The problem of finding an approximation to the root of a nonlinear equation can be found in many sciences and engineering fields (Tanakan, 2013; Dowell and Jarratt, 1972; Ridders, 1979; McGuire, 1994; Boutayeb and Chatouani, 2007). A root-finding algorithm is a numerical method or algorithm for finding a value x such that f(x) = 0, for a given function f. Such an x is called a root of the function f. In numerical analysis, a numerical method is a mathematical tool designed to solve numerical problems. Such problems are from the natural sciences, social sciences, engineering, medicine, and business (Franz et al., 2019; Wahls and Poor, 2015). Many numerical methods produce sequences of real numbers. Sometimes, the convergence of these sequences is slow and their utility in solving practical problems is quite limited. Convergence acceleration methods try to transform a slowly converging sequence into a fast convergent one (Cordero, 2010).

In last decades, several numerical methods have been presented for solution of algebraic equations (Grundy, 2008). Atken presented extended the Bernoulli’s method of approximating to the numerically greatest root of an algebraic equation (Atken, 1927). Shinohara presented a geometric method for the numerical solution of algebraic equations. Anderson and Björck presented a novel numerical method by using high order method of regula falsi method. Ridders proposed an iterative method based on false position method for determination of a single real root of a real continuous function. (Kou et al., 2007) use this technique composing a Newton-type three order method with classical Newton method and approximating the derivative in the new estimate by its value in the old one. In the paper of (Parhi and Gupta, 2008) they compose the third order method obtained by [Weerakoon and Fernando, 2000] with Newton’s method and estimate the derivative by linear interpolation. The same idea is used by (Mir and Zaman, 2007), where they compose Ostrosky’s method with Halley’s method, introducing a parameter that allows to obtain families of iterative methods with sixth, seventh and eighth order of convergence. More efficient methods are presented by (Ham et al., 2008) using the idea of composition but introducing an auxiliary function that generates new modifications to known methods. Many numerical methods presented for approximation solution of algebraic equations in (Hafiz, 2013; Wahls and Poor, 2015; Cordero et al., 2010; Parhi and Gupta, 2008; Ham et al., 2008; Proinov and Ivanov, 2015; Proinov, 2015; Binwal, 2021; Hamadil et al., 2021; Kodhnyanko, 2021; Tassaddiq et al., 2021).

The efficiency of numerical methods is typically assessed by the number of iterations necessary to obtain a solution to the equation (Atkinson, 2000; Deuflhard, 2005; Corliss, 1977; Kau, 2009). However, this approach is not the only criterion. Often, the efficiency of a method is determined by the number of calls to the function f(x). For this consideration, the lower the number of calculations the function is required to solve the problem, the higher the method speed. With the high performance of modern computers, the time gain offered by high-speed methods is commonly believed to be negligible compared to reliable methods such as the bisection method (Antia, 2012).

Finding an equation’s root is necessary to solve this class of problems. In this work, we focus on Ridders method (Press and Teukolsky, 2007; Kyrurkchier, 1998) which can be used to find approximation solutions to equations (Schadolin, 2012).
2. The Method

\[ f(x) = 0 \]  \hspace{1cm} (1)

Firstly, it will be assumed that the function \( f(x) \) is continuous on a closed interval. In this interval \([a, b]\), the function has a root and the following inequality holds

\[ f(x_a) \cdot f(x_b) < 0 \]  \hspace{1cm} (2)

We aim to determine the root \( x_c \) of the nonlinear equation with least numerical evaluations as possible. Let the function be represented by \( f(x) \) by the mean value theorem if \( f(x_a) \cdot f(x_b) < 0 \) then \( f(x) \) has a zero in \([x_a, x_b]\). We create a new function \( G(x) = f(x)e^{mx} \) in such a way that \( x_1 = \frac{x_a + x_b}{2} \) for three equidistant \( x \) values \( x_0, x_1, x_2 \) the following requirement is met [16]:

\[ G_2 - 2G_1 + G_0 = 0, \text{ where } G_n = G(x_n), n = 0, 1, 2, 3, \ldots \]  \hspace{1cm} (3)

Let us define a distance constant between data as \( d = x_2 - x_1 \), from equation (3) dividing both sides of the equation with \( e^{mb1} \), we get

\[ f(x_2)e^{m(x_2-x_1)} - 2f(x_1)e^{m(x_1-x_0)} + f(x_0) = 0 \]  \hspace{1cm} (4)

By simplifying (4), we obtain

\[ f(x_2)e^{2dm} - 2f(x_1)e^{dm} + f(x_0) = 0 \]  \hspace{1cm} (5)

Equation (5) has two exact solutions, by using the quadratic rule

\[ e^{dm} = \frac{2f_1 \pm \sqrt{4f_1^2 - 4f_2f_0}}{2f_2} \quad \text{and} \quad e^{dm} = \frac{f_1 \mp \sqrt{f_1^2 - f_2f_0}}{f_2} \]  \hspace{1cm} (6)

From the fact that \( e^{dm} > 0 \), and \( \frac{f_1}{f_2} - \frac{f_2}{f_0} > 0 \), equation (6) can be rewritten by dividing numerator and denominator \( f_0 \), as follow

\[ e^{dm} = \frac{f_1f_0 - \sqrt{f_1^2f_2^2 - f_2f_0}}{f_2f_0} \]  \hspace{1cm} (7)

Now by applying Newton Raphsons method for finding the root of \( G(x) \) we obtain

\[ x_{n+1} = x_n - \frac{G(x_n)}{G'(x_n)}, \text{ } n \in N \]  \hspace{1cm} (8)

Now, we shall construct the first order of accuracy difference scheme for \( G'(x_c) \), to find difference scheme for (9), it yields

\[ x_2 = x_2 - \frac{G_2}{G_2 - G_1} \]  \hspace{1cm} (9)

Dividing numerator and denominator by \( G(x_2) \) \( \forall f(x_2) \neq 0 \), we obtain

\[ x_2 = x_2 - \frac{d}{1 - \frac{G_2}{G_1}} \]  \hspace{1cm} (10)

By using assumption \( G(x) = f(x)e^{mx} \), can be simplified to

\[ x_3 = x_2 - \frac{d}{1 - \frac{G_2}{G_1}} \]  \hspace{1cm} (11)

Putting value of \( e^{-md} \) in (11), yields

\[ x_3 = x_2 - \frac{d}{1 - \frac{G_2}{G_1}} \]  \hspace{1cm} (12)

After few simplification steps, (13) can be written as follow:

\[ x_3 = x_2 + \frac{d}{s} \]  \hspace{1cm} (13)

We can define \( s = \sqrt{a^2 - b} \), such that \( a = \frac{f_0}{f_1}, b = \frac{f_2}{f_0} \) Restating (14), yields

\[ x_3 = x_2 + \frac{(a - s)}{s} \]  \hspace{1cm} (14)

2.1 Proposed Method Algorithm

Step 1: Find two points \( x_3 \) and \( x_2 \) such that \( f(x_3)f(x_2) < 0 \), and tolerance \( \varepsilon \), and \( k = 1 \);

Step 2: Compute \( x_k = \frac{x_2 + x_{k+1}}{2} \), set \( d = x_{k+1} - x_k \), \( a = \frac{f_0}{f_1}, b = \frac{f_2}{f_0} \), and \( s = \sqrt{a^2 - b} \), for \( k \in N \)

Step 3: find \( x_{k+1} = x_k + \frac{d(\varepsilon - s)}{s} \), for \( k \in N \)

Step 4: If \( |x_{k+2} - x_{k+1}| \leq \varepsilon \) or \( f(x_{k+1}) \leq \varepsilon \), stop, accept \( x_{k+2} \) as a root of the function.

Else, find \( \min V = \min(x_{k+1}, x_{k+2}, x_{k+3}) \)

\[ x_{k+1} = \frac{x_{k+1} + x_{k+2}}{2}, \text{ if } f(x_{k+2}) < \min V \]

\[ (\min V(x_{k+1})+x_{k+2}), \text{ if } f(x_{k+2}) > \min V \]

Step 5: Set \( k = k+1 \), go to step 2.

3. Numerical Computations

In this section we provide several different examples for testing the accuracy of the presented method. For each example the comparison was made between most wellknown methods.

In Table 1. and table 2. We have presented two different examples, we have applied the bisecction, false position, Ridders and Improved method. The calculation done by matlab. For each example a results presented in a Table. Absolute error has been calculated by putting the approximate solution in the function \( f(x) \).

| Table 1: Presents a Comparison Between Wellknown Methods with Improved Method for \( f(x) = xe^x - 1 \) in \([-1,1] \). With \( \varepsilon = 10^{-4} \). |
|---|---|---|---|
| Method | Approximate Root | Absolute Error | Iteration No. |
| Bisection | 0.5673828125 | 6.6198×10^{-4} | 11 |
| Regula Falsi | 0.566930232156059 | 5.8838×10^{-4} | 8 |
| Newton Raphson | 0.567143296530296 | 1.6912×10^{-8} | 3 |
| Ridders Method | 0.567143276652295 | 3.8264×10^{-8} | 3 |
| Improved Method | 0.5672180695890405 | 2.0664×10^{-4} | 2 |

Table 3 presented a comparison between Improved method with other wellknown methods in terms of iteration number. In most cases the Improved algorithm outperforms other bracketing method (see 1,2,5). However, in some cases the Improved method slowly converges to the root of the function (case 3,4) convergends in same iteration number with ridders method. In these cases all algorithms have a large number of iterations except the most ridders method and improved method. In all cases the improved ridders method keeps a low number of iterations. Generally, the improved method converges slowly in polynomial function, but it converges quickly in non polynomial functions.
Table 2: Presents a Comparison Between The Improved Method and Wellknown Method for \( f(x) = \frac{1}{e^x} + \cos(x) \) in The Interval in \([-2,2]\) With A Tolerance 10^{-7}

<table>
<thead>
<tr>
<th>Method</th>
<th>Approximate Root</th>
<th>Absolute Error</th>
<th>Iteration No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bisection</td>
<td>1.746139585971832</td>
<td>6.4405 \times 10^{-8}</td>
<td>26</td>
</tr>
<tr>
<td>Regula falsi</td>
<td>1.746139575307979</td>
<td>5.2044 \times 10^{-8}</td>
<td>16</td>
</tr>
<tr>
<td>Ridders method</td>
<td>1.74613953047187</td>
<td>7.4013 \times 10^{-11}</td>
<td>4</td>
</tr>
<tr>
<td>Proposed method</td>
<td>1.74613953048012</td>
<td>4.0672 \times 10^{-16}</td>
<td>4</td>
</tr>
</tbody>
</table>

Remark 1. The proposed method converges quadratically.

Remark 2. Improved method has more effect in non polynomial functions.

4. CONCLUSION

In this paper, a numerical bracketing algorithm for the iterative root finding of the nonlinear equations developed. The algorithm is based on the improvement of ridders method, and newton raphson’s method with the combination of bisection method. However, an algorithm for the proposed method given. On the other hand, different examples are included to demonstrate the validity and applicability of the presented technique. MATLAB-based program is built for the proposed method. The error analysis table presents the obtained numerical results. The capabilities of the method were studied with results supporting that it is faster than compared methods. The improved method has a quadratic convergence rate. Moreover, the proposed algorithm can be used as a good substitute to the well-known bracketing methods. The weakness and strength of the algorithm are presented in some typical examples and a comparison with other methods is given. The theoretical terms for the solution of these methods are supported by the results of numerical experiments.

REFERENCES


