Science and Technology Publishing (SCI & TECH)
ISSN: 2632-1017
Vol. 4 Issue 5, May - 2020

Perturbation-Based Seeded Regular Falsi Iteration Method

Akpasam Joseph Ekanem

Department of Electrical/Electronic Engineering,
Akwa Ibom State University Mkpat Enin, Akwa Ibom State, Nigeria

***Abstract*— In this paper, perturbation-based seeded Regular Falsi iteration method is presented. The approach combined the perturbations-based initial root generation approach with Regular Falsi iteration method. The procedure requires that a single guess root value be used with a given perturbation value to generate two initial root values that are suitable for application in Regular Falsi iteration method. Two sample functions were used to demonstrate the applicability of the scheme. The simulation was conducted in Matlab software for the two selected functions. The seed (or the single initial root value), xo was selected and the perturbation (δ) value of $x_0/10^9$ was used to generate the two initial root values (X_L and X_U) such that $f(X_L) * f(X_U) < 0$. For the first function; $f(x) = x^6 - x - 1 = 0$, the seed value of 1.02125173 was selected and the generated two initial root values for iteration were, X_L =1.02125173 and X_U =1.17778050 and the cycle was 14. Similarly, for the second function; $f(x) = x^{3.5} - 80 = 0$, the results showed that the seed value of 1.02125173 was selected and the generated two initial root values for iteration are, X_L =3.14762152 and X_U =3.54866727 and the cycle was 10.**

Keywords*— *Regular Falsi, Initial Root Selection, Seeded Regular Falsi, Numerical Iteration , Convergence Cycle

1. INTRODUCTION

There are several numerical iteration methods for finding the roots of functions [1,2,3,4,5]. Among the iteration methods, are the bracketing methods which require two initial guess root values such that one of the two roots is above the expected root value and the other one is below the expected root value [6,7,8,9,10]. Regular Falsi is one of such bracketing iteration methods [11,12,13,14,15]. While it has been easy to develop modified Secant iteration method that can use only one initial guess root value and a constant perturbation value, of say 0.01 to implement the Secant iteration, the Regular Falsi proves difficult because of the additional requirement that the two guess roots must bracket (that is, one root value above and the other below) the actual root of the function.

Consequently, in this paper, a method of using the modified Secant approach with a perturbation value and single initial guess root value is adapted and applied to the Regular Falsi method. The procedure first use the initial single guess root (the seed value) and a perturbation value which is any fraction of the guess value to generate the two initial root values that brackets the actual root of the function. After the two initial roots are generated, the normal Regular Falsi iteration method is implemented based on those two generated initial root values. Relevant procedure for implementing the proposed perturbation based seeded Regular Falsi is presented along with sample numerical examples . The simulation was conducted in Matlab program.

2. DEVELOPMENT OF THE PERTURBATION-BASED SEEDED REGULAR FALSI ITERATION METHOD

The word seeded here means that only one seed (that is, only one initial guess root value, which denoted here as x_o) is required to conduct the Regular Falsi iteration. This is contrary to the classical Regular Falsi method that requires two initial guess root values, say x_o and x_1, such that the product of the function, f(x) for $x = x_o$ and $x = x_1$ is less than zero, (that is, $f(x_0) * f(x_1) < 0$). The perturbation-based seeded Regular Falsi iteration algorithm uses one seed value , x_0 to first generate the second seed value, x_1 such that the condition $f(x_0) * f(x_1) < 0$) is satisfied. The algorithm is given as follows:

Step 1:
Step 1.1: Input: Initial value for x_o
Step 1.2: Set value for $\delta = x_o/0.01$
Step 1.3: Input: Desired Accuracy, ϵ
Step 1.4: Input: Maximum Number of Iterations, n
Step 4
Step 4.1: $j = 1$

Step 4.2: Compute

$x_1 = x_0 - (j)\left(\frac{(\delta)}{f(x_0+\delta)-f(x_0)}\right)f(x_0)$

Step 4.3: If $f(x_0) * f(x_1) < 0$ Then

Step 4.4: Goto Step 5

Step 4.5: Else

Step4.6: **j = j +1**

Step4.7: $x_o = x_1$

Step4.8: Goto Step 4.2

Step 4.9: Endif

Step 5: If $(|f(x_1) - f(x_0)| < \epsilon)$ then

Step 5.1: Output $x_1, f(x_1)$

Step 5.2: Got Step 12

Step 5.3: Else

Step 5.4: If $(x_1 < x_0)$ then

Step 5.4.1: $xU = x_0$

Step 5.4.2: $xL = x_1$

Step 5.4.3: $x_1 = xU$

Step 5.4.4: Else

Step 5.4.5: $xU = x_1$

Step 5.4.6: $xL = x_0$

Step 5.7: Endif

Step 6: For K = 2 To n Step 1 do:

Step 7:

Step 7.1: Compute $f(xL)$

Step 7.2: Compute $f(xU)$

Step 7.3: Compute $x_k = \frac{xL*f(xU)-xU*f(xL)}{f(xU)-f(xL)}$

Step 7.4: Compute $f(x_k)$

Step 8:

Step 8.1: If $|x_k - x_{k-1}| < \epsilon$ Then

Step 8.2: Output $x_k, f(x_k)$

Step 8.3: Goto Step 12;

Step 8.4: EndIf

Step 9:

Step 9.1: If $(f(xL)) * f(x_k) < 0)$ then

Step 9.2: $xU = x_k$

Step 9.3: Else

Step 9.4: $xL = x_k$

Step 9.5: Endif

Step 10:

Step 10.1: K = K +1

Step 10.2: Goto Step 7

Step 11: Output "Maximum number of iterations exceeded; Method failed to find the root"

Step 12: End

3. RESULTS AND DISCUSSIONS

3.1 SELECTED FUNCTIONS

The two selected functions for the demonstration of the applicability of the ideas presented in these papers are:

1) $\mathbf{f(x) = x^6 - x - 1 = 0}$

2) $\mathbf{f(x) = x^{3.5} - 80 = 0}$

The simulation was run in Matlab software for the two selected functions. The seed (or the single initial root value), xo was selected and the perturbation ($\boldsymbol{\delta}$) value of $x_0/10^9$ was used to generate the two initial root values (X_L and X_U) such that $f(X_L) * f(X_U) < \mathbf{0}$.

The results in Table 1 shows the initial single root, x_0 and the derive two initial roots (X_L and X_U) for the first function; $\mathbf{f(x) = x^6 - x - 1 = 0}$ while Table 2 shows the convergence cycle with error tolerance of of 10^{-13} for the first function; $\mathbf{f(x) = x^6 - x - 1 = 0}$. According to Table 1, the seed value of 1.02125173 was selected and the generated two initial root values for iteration are, X_L =1.02125173 and X_U =1.17778050. The error tolerance for the simulation was 10^{-13}. The results in Table 2 show that the error tolerance value was attained in cycle 14.

Similarly, the results in Table 3 shows the initial single root, x_0 and the derive two initial roots (X_L and X_U) for the second function; $f(x) = x^{3.5} - 80 = 0$ while Table 4 shows the convergence cycle with error tolerance of of 10^{-13} for the second function. According to Table 3, the seed value of 1.02125173 was selected and the generated two initial root values for iteration are, X_L =3.14762152 and X_U = 3.54866727. The error tolerance for the simulation was 10^{-13}. The results in Table 4 show that the error tolerance value was attained in cycle 10.

The overall results show that the perturbation-based approach can be used to run Regular Falsi iterative method but the procedure is broken into two parts, the initial seed generation and the iteration using the normal Regular Falsi method and the generated initial root values.

Table 1 : The initial single root, x_0 and the derive two initial roots (X_L and X_U) for the first function; $\mathbf{f(x) = x^6 - x - 1 = 0}$

j		xj	f(xj)	f(xj-1)*f(xj)	XL	XU
0	f(xo)=f(x)	1.02125172600	-0.88677179		1.02125172702	1.17778059954
1	f(x1) = f(x+δ)	1.02125172702	-0.88677178	0.78636419		
2	f(x2) = f(1-(Δ*x))	1.17778049954	0.49145048	-0.43580442		
3	f(x3) = f(1-(2*Δ)*x)	-1.30256429843	5.18678211	2.54904656		

Table 2 The convergence cycle with error tolerance of 10^{-13} for the first function; $\mathbf{f(x) = x^6 - x - 1 = 0}$

n	XL	XU	f(XL)	f(XU)	X	f(X)	f(XL)*f(X)
1	1.0212517	1.177780	-8.8677177979E-01	0.49145048	1.1219650	-0.127273	1.1286192045E-01
2	1.1219650	1.177780	-1.2727279219E-01	0.49145048	1.1334464	-0.013104	1.6678419669E-03
3	1.1334464	1.177780	-1.3104465913E-02	0.49145048	1.1345978	-0.001299	1.7020304701E-05
4	1.1345978	1.177780	-1.2988171219E-03	0.49145048	1.1347117	-0.000128	1.6655728632E-07
5	1.1347117	1.177780	-1.2823767374E-04	0.49145048	1.1347229	-0.000013	1.6230604087E-09
6	1.1347229	1.177780	-1.2656658230E-05	0.49145048	1.1347240	-0.000001	1.5809761537E-11
7	1.1347240	1.177780	-1.2491260528E-06	0.49145048	1.1347241	0.000000	1.5399199600E-13
8	1.1347241	1.177780	-1.2327978882E-07	0.49145048	1.1347241	0.000000	1.4999237388E-15
9	1.1347241	1.177780	-1.2166825991E-08	0.49145048	1.1347241	0.000000	1.4609647921E-17
10	1.1347241	1.177780	-1.2007772554E-09	0.49145048	1.1347241	0.000000	1.4230342149E-19
11	1.1347241	1.177780	-1.1850942450E-10	0.49145048	1.1347241	0.000000	1.3858467354E-21
12	1.1347241	1.177780	-1.1693979118E-11	0.49145048	1.1347241	0.000000	1.3499645276E-23
13	1.1347241	1.177780	-1.1544099010E-12	0.49145048	1.1347241	0.000000	1.2995955863E-25
14	1.1347241	1.177780	-1.1257661470E-13	0.49145048	1.1347241	0.000000	1.1498633770E-27

Table 3 : The initial single root, x_0 and the derive two initial roots (X_L and X_U) for the second function; $\mathbf{f(x) = x^{3.5} - 80 = 0}$

j		x_0	$f(x_j)$	$f(x_{j-1})* f(x_j)$	The Two Initial Roots	
					X_L	X_U
0	f(xo)=f(x)	3.14762151600	-24.67279023		3.14762152	3.54866727
1	f(x1) = f(x+δ)	3.14762151915	-24.67279004	608.74657314		
2	f(x2) = f(1-(Δ*x))	3.54866727226	4.18379199	-103.22582143		
3	f(x3) = f(1-(2*Δ)*x)	3.94971302852	42.45581101	177.62628217		

Table 4 The convergence cycle with error tolerance of 10^{-13} for the second function; $\mathbf{f(x) = x^{3.5} - 80 = 0}$

n	XL	XU	f(XL)	f(XU)	X	f(X)	f(XL)*f(X)
1	3.1476215	3.5486670	-2.4672790040E+01	4.183791993	3.4905214	-5.4594729638E-01	1.3470043017E+01
2	3.4905214	3.5486670	-5.4594729638E-01	4.183791993	3.4972331	-9.9408323985E-03	5.4271705717E-03
3	3.4972331	3.5486670	-9.9408323985E-03	4.183791993	3.4973550	-1.8032482147E-04	1.7925788275E-06
4	3.4973550	3.5486670	-1.8032482147E-04	4.183791993	3.4973572	-3.2708339575E-06	5.8981254944E-10
5	3.4973572	3.5486670	-3.2708339575E-06	4.183791993	3.4973572	-5.9328215229E-08	1.9405274101E-13
6	3.4973572	3.5486670	-5.9328215229E-08	4.183791993	3.4973572	-1.0761453950E-09	6.3845785614E-17
7	3.4973572	3.5486670	-1.0761453950E-09	4.183791993	3.4973572	-1.9497292669E-11	2.0981921721E-20
8	3.4973572	3.5486670	-1.9497292669E-11	4.183791993	3.4973572	-3.8369307731E-13	7.4809762235E-24
9	3.4973572	3.5486670	-3.8369307731E-13	4.183791993	3.4973572	0.0000000000E+00	0.0000000000E+00
10	3.4973572	3.5486670	0.0000000000E+00	4.183791993	3.4973572	0.0000000000E+00	0.0000000000E+00

4. CONCLUSION

The approach to combine the perturbations-based initial root generation approach in Regular Falsi iteration method is presented. The procedure requires that a single guess root value be used with a given perturbation value to generate two initial root values that are suitable for application in Regular Falsi iteration method. Two sample functions were used to demonstrate the applicability of the scheme.

REFERENCES

1) Aggarwal, Ankush, and Sanjay Pant. "Beyond Newton: a new root-finding fixed-point iteration for nonlinear equations." *Algorithms* 13.4 (2020): 78.
2) Kopecky, Karen A. "Root-finding methods." *Economics* 613 (2007): 614.
3) Kansal, Munish, V. Kanwar, and Saurabh Bhatia. "On Some Optimal Multiple Root-

Finding Methods and their Dynamics." *APPLICATIONS AND APPLIED MATHEMATICS-AN INTERNATIONAL JOURNAL* 10.1 (2015): 349-367.

4) Yun, Beong In. "Iterative methods for solving nonlinear equations with finitely many roots in an interval." *Journal of Computational and Applied Mathematics* 236.13 (2012): 3308-3318.
5) Mekwi, Wankere R. "Iterative methods for roots of polynomials." (2001).
6) Tiruneh, Ababu Teklemariam, William N. Ndlela, and Stanley J. Nkambule. "A Two-Point Newton Method Suitable for Nonconvergent Cases and with Super-Quadratic Convergence." *Advances in Numerical Analysis* (2013).
7) Kanwar, Vinay, Saurabh Bhatia, and Munish Kansal. "New optimal class of higher-order methods for multiple roots, permitting f′(xn)= 0." *Applied Mathematics and Computation* 222 (2013): 564-574.
8) Sangah, Ali Asghar, Asif Ali Shaikh, and Syed Feroz Shah. "Comparative Study of Existing Bracketing Methods with Modified Bracketing Algorithm for solving Nonlinear Equations in single variable." *Sindh University Research Journal-SURJ (Science Series)* 48.1 (2016).
9) Intep, Somkid. "A review of bracketing methods for finding zeros of nonlinear functions." *Applied Mathematical Sciences* 12.3 (2018): 137-146.
10) Hafiz, M. A. "A new combined bracketing method for solving nonlinear equations." *J. Math. Comput. Sci.* 3.1 (2013): 87-93.
11) Sangah, Ali Asghar, Asif Ali Shaikh, and Syed Feroz Shah. "Comparative Study of Existing Bracketing Methods with Modified Bracketing Algorithm for solving Nonlinear Equations in single variable." *Sindh University Research Journal-SURJ (Science Series)* 48.1 (2016).
12) Suhadolnik, Alojz. "Superlinear bracketing method for solving nonlinear equations." *Applied Mathematics and Computation* 219.14 (2013): 7369-7376.\
13) Intep, Somkid. "A review of bracketing methods for finding zeros of nonlinear functions." *Applied Mathematical Sciences* 12.3 (2018): 137-146.
14) Jia, Yan-Bin. "Solution of Nonlinear Equations." (2019).
15) Gomes, Abel, and José Morgado. "A generalized regula falsi method for finding zeros and extrema of real functions." *Mathematical Problems in Engineering* 2013 (2013).