American Journal of Computational and Applied Mathematics 2013, 3(2): 86-90
DOI: 10.5923/j.ajcam.20130302.05

# An Efficient Cubically Convergent Two-Point Newton Method

**Ababu Teklemariam Tiruneh**

Department of Environmental Health Science. University of Swaziland. P.O.Box 369, Mbabane H100, Swaziland

**Abstract** A numerical procedure for solving non-linear equations is presented which is a modification of the Two Point Newton Method developed by the author earlier. It is shown that the method presented here has a third order convergence. The modified procedure incorporates computation of the x value for an intermediate point using the original Two Point Newton Method. The derivative of the intermediate point is also estimated through linear extrapolation of the first derivatives of the previous two points of the iteration while the functional value for the intermediate point is obtained by applying trapezoidal rule on the first derivatives between the nearest point and the intermediate point. The computational efficiency of the proposed method is high since the method requires evaluation of two functions per iterationonly while attaining a third order convergence. Examples have been presented in this paper that show the application of the method and comparison of the rate of convergence of the method with the Newton Method as well as withthe previously developed Two point Newton Method.

**Keywords** Rootsof Equations, Newton Method, Root Approximations, Iterative techniques, Two Point Newton Method

## 1. Introduction

Solutions of equations by numerical iteration are of interest to many science and engineering problems. For this purpose, several methods have evolved over time. The traditional Newton method which requires evaluation of one function and its derivative at each step of the iteration has an order of convergence of two in many cases. Several methods developed later on over the years are largely based on modification of the Newton method and have been shown to give increasingly higher order convergence.

Methods with demonstrated higher order convergence, mostly based on Newton iteration, have been documented in the literature. Starting from a third order convergencemethod, the method developed by S. Weeraksoon and T.G. Fernando[1] that requires evaluation of one function and two first derivatives is a notable example. On the other hand, the method developed by Ostrowsky[2] has a fourth order convergence and requires evaluation of two functional values and one value of a derivative. A number of sixth order methods have also been developed that generally require evaluation of four functions in each step of the iteration, (Grau and D´ıaz-Barrero[3], Sharma and Guha[4], Chun and Ham[5]).

The method developed by Kou, et al[6] has a seventh order convergence and requires evaluation of two functional values and two values of a derivative. The method developed by Bi, et al[7] has eighth order convergence and requires a total of four functional evaluations. Hu et al[8] developed a method that has Ninth order convergence that requires evaluation of four functional values per iteration.

The Two point Newton Method developed earlier by the author[9] using a two-point modification of the traditional Newton method has been shown to give a super-quadratic convergence of order about 2.414. The method is based on application of Newton iteration formula by taking as the independent variable the cotangent of the angle between the line connecting the two successive points of iteration with the vertical and as the dependent variable the given function y = f(x). The resulting iteration formula for a root estimation is shown to be the weighted sum of the estimates of the two previous iterations with a weighing factor that penalizes the iteration point having undesirable characteristics such as a near zero derivative. For example, near the point where the derivative of the function is zero, the weighing factor for that particular point will be close to zero. This condition effectively moves the iteration away from the undesirable point having zero derivative.

The method offers a particular advantage for cases where the traditional Newton method and its variants of higher order convergence may not converge at all. In terms of computational effort, the proposed method requires only evaluation of the value of a function and its derivative at each step of the iteration except for the first iteration step in which two functional evaluations are required for the first

* Corresponding author:
ababute@yahoo.com (Ababu Teklemariam Tiruneh)
Published online at http://journal.sapub.org/ajcam

two starting points.

The Third Order Two Point Newton Method proposed in this paper is a modification of the earlier Two Point Newton method. The modificationstep consists ofestimation of the x-and y values as well as the first derivative for an intermediate point that lies between the two successive points of iteration. The method, as will be shown later, requires evaluation of the function and its derivative for each step of the iteration except the first iteration step which requires evaluation of a function and thederivative for the first two starting points. The proposed procedure as such offers a higher computation efficiency requiring evaluation of two functions per iteration only while attaining a third order convergence.

## 2. Methodology

The two-point Newton method developed earlier by the author[9] takes the form:

$$x_{k+1} = x_{k-1} - \frac{(x_{k-1}-x_k)}{1-\left(\frac{y_k}{y_{k-1}}\right)\left(\frac{\frac{y_k-y_{k-1}}{x_k-x_{k-1}}}{y'_k}\right)} \quad (1)$$

Where

- $x_{k-1}$, $x_k$ and $x_{k+1}$ = the estimate of the root at the k-1$^{th}$, k$^{th}$and k+1$^{th}$ iterations respectively.
- $y_{k-1}$, $y_k$ = the functional values corresponding to the values of $x_{k-1}$ and xk respectively.
- $y'_k$ = the first derivative of the function y at the kth iteration.

The modified cubically convergent Two-Point Newton method proposed in this paper is based on a two-step extension of the earlier method but with the same number of functional evaluations as the earlier method. The step for the proposed method is outlined below.

First Equation 1 above is applied to locate the intermediate point $x_i$ (refer to Figure 1 below).

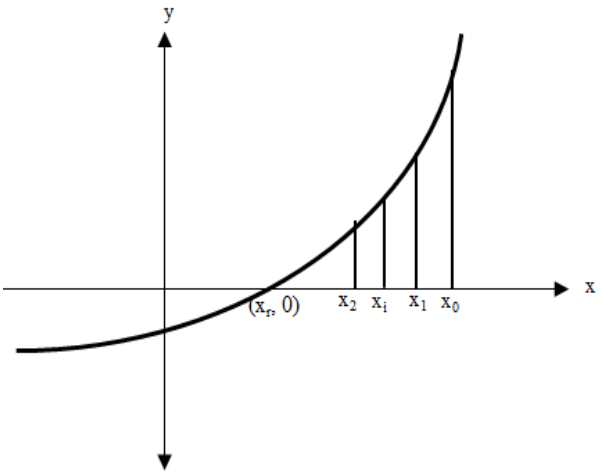


**Figure 1.** The graph of y=f(x) showing iteration points $x_0$, $x_1$, $x_i$ and $x_2$

$$x_i = x_0 - \frac{(x_0-x_1)}{1-\left(\frac{y_1}{y_0}\right)\left(\frac{\frac{y_1-y_0}{x_1-x_0}}{y'_1}\right)} \quad (2)$$

The estimate for the derivative dy/dx at the intermediate point is made using the linear extrapolation of the derivatives at $x_0$ and $x_1$.

$$y_i' = \left.\frac{dy}{dx}\right|_{x=i} = y'_1 - \left(\frac{y'_1-y'_0}{x_1-x_0}\right)(x_1-x_i) \quad (3)$$

The estimate for the functional value at the intermediate point yi is made from the derivatives at the points $x_1$ and point $x_i$ using the Trapezoidal rule. (Refer to Figure 2 below).

Area of the Trapezoid A is calculated as: (Figure 2 above):

$$A = \frac{1}{2}(y'_1 + y'_i)(x_1 - x_i)$$

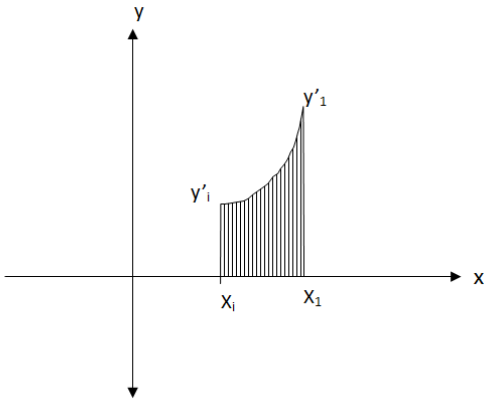


**Figure 2.** Estimate for the intermediate y value using trapezoidal rule

Further yi can be expressed as:

$$y_i = y_1 - \text{Area of Trapezoid, A} \quad (4)$$

Inserting the expression for area A given above;

$$y_i = y_1 - \left[\frac{1}{2}(y'_1 + y'_i)(x_1 - x_i)\right]$$

Substituting the expression for y'$_i$given in Equation 3 above results in:

$$y_i = y_1 - \frac{1}{2}\left[y'_1 + y'_1 - \left(\frac{y'_1-y'_0}{x_1-x_0}\right)(x_1-x_i)\right](x_1-x_i)$$

Further simplification leads to:

$$y_i = y_1 - y'_1(x_1 - x_i) + \left(\frac{1}{2}\right)\left(\frac{y'_1-y'_0}{x_1-x_0}\right)(x_1 - x_i)^2 \quad (5)$$

Finally the estimate for $x_2$ is made by applying the Two Point Newton formula once again:

$$x_2 = x_1 - \frac{x_1-x_i}{1-\left(\frac{y_i}{y_1}\right)\left[\frac{\frac{y_1-y_i}{x_1-x_i}}{y'_i}\right]} \quad (6)$$

In terms of the iteration steps, k-1, k and k+1, Equation 6 above is generalized as:

$$x_{k+1} = x_k - \frac{x_k-x_{ki}}{1-\left(\frac{y_{ki}}{y_k}\right)\left[\frac{\frac{y_k-y_{ki}}{x_k-x_{ki}}}{y'_{ki}}\right]} \quad (7)$$

**Computation steps**

The computational procedure for each step of the iteration

consists of evaluation of equations 8,9,10 and 11 that are shown below. It is to be noted that, for $x_{k-1}$ the most recent value obtained from the previous intermediate iteration , i.e., $x_{(k-1)i}$ will be used instead of $x_{k-1}$ which represents a point further away from the root than the point $x_{(k-1)i}$ .

$$x_{ki} = x_{(k-1)i} - \frac{(x_{(k-1)i}-x_k)}{1-\left(\frac{y_k}{y_{(k-1)i}}\right)\left(\frac{\frac{y_k-y_{(k-1)i}}{x_k-x_{(k-1)i}}}{y'_k}\right)} \quad (8)$$

$$y_{ki}' = y'_k - \left(\frac{y'_k-y'_{(k-1)i}}{x_k-x_{(k-1)i}}\right)(x_k - x_{ki}) \quad (9)$$

$$y_{ki} = y_k - y'_k(x_k - x_{ki}) + \left(\frac{1}{2}\right)\left(\frac{y'_k-y'_{(k-1)i}}{x_k-x_{(k-1)i}}\right)(x_k - x_{ki})^2 \quad (10)$$

$$x_{k+1} = x_k - \frac{x_k - x_{ki}}{1-\left(\frac{y_{ki}}{y_k}\right)\left[\frac{\frac{y_k-y_{ki}}{x_k-x_{ki}}}{y'_{ki}}\right]} \quad (11)$$

# 3. Proof of Third Order Convergence

The iteration process follows the following sequence:

| | | | |
|---|---|---|---|
| $x_0$ | $x_1$ | $x_{i1}$ | $x_2$ |
| $x_{i1}$ | $x_2$ | $x_{i2}$ | $x_3$ |
| $x_{i2}$ | $x_3$ | $x_{i3}$ | $x_4$ |
| $x_{i3}$ | $x_4$ | $x_{i4}$ | $x_5$ |
| $x_{i4}$ | $x_5$ | $x_{i5}$ | $x_6$ |
| … | … | … | … |
| $x_{ik-1}$ | $x_k$ | $x_{ik}$ | $x_{k+1}$ |

Examination of the columns of x values shown above reveals that the intermediate x values of the iteration $x_{i1}$, $x_{i2}$, $x_{i3}$, etc., are bounded both to the left and to the right with the same sequence of x values of the iteration, i.e., $x_1$, $x_2$, $x_3$, etc. Therefore, the intermediate points also converge to the same root towards which the x values converge. They also converge to the root with the same rate of convergence as the x values.

Using Taylor series expansion of the functions y and its derivative about the root r gives the following expression:

$$y_{k-1} = (y_r = 0) + y'_r[(x_{k-1} - r) + C_1(x_{k-1} - r)^2 + C_2(x_{k-1} - r)^3 + C_3(x_{k-1} - r)^4 + \cdots..]$$

Where the $C_m$ values are given by:

$$C_m = \frac{y^{m+1}(r)}{(y'_r)m!}\, m = 1, 2, 3, \ldots ....$$

In the above expression, r is the desired root of the equation and $y^{m+1}(r)$ is the $m+1^{th}$ derivative of the function y evaluated at the root x = r.

Denoting the error terms $e_k = x_k - r$; $e_{k-1} = x_{k-1} - r$, etc.;

$$y_{k-1} = (y_r = 0) + y'_r[(e_{k-1}) + C_1(e_{k-1})^2 + C_2(e_{k-1})^3 + C_3(e_{k-1})^4 + \cdots..]$$

$$y_k = (y_r = 0) + y'_r[(e_k) + C_1(e_k)^2 + C_2(e_k)^3 + C_3(e_k)^4 + \cdots..]$$

$$y'_{k-1} = y'_r[1 + 2C_1(e_{k-1}) + 3C_2(e_{k-1}) + 4C_3(e_{k-1}) + \cdots..]$$

$$y'_k = y'_r[1 + 2C_1(e_k) + 3C_2(e_k) + 4C_3(e_k) + \cdots..]$$

It is now possible to write the expression for the intermediate point of iteration, $x_i$, i.e.

$$x_{ki} = x_{k-1} - \frac{(x_{(k-1)i}-x_k)}{1-\left(\frac{y_k}{y_{(k-1)i}}\right)\left(\frac{\frac{y_k-y_{(k-1)i}}{x_k-x_{(k-1)i}}}{y'_k}\right)} \quad (12)$$

In terms of the error terms $e_k$, $e_{k-1}$, etc., equation 12 above can be rewritten as;

$$e_{ki} = e_{k-1} - \frac{(e_{(k-1)i}-e_k)}{1-\left(\frac{y_k}{y_{(k-1)i}}\right)\left(\frac{\frac{y_k-y_{(k-1)i}}{e_k-e_{(k-1)i}}}{y'_k}\right)} \quad (13)$$

Substituting the Taylor series forms for the variables y, $y_{(k-1)i}$ and $y'_k$ in Equation 13 above and using the commands available in MATLAB symbolic manipulation platform MUPAD resulted in the following simplified expression for $e_{ki}$. Note that terms containing fourth order errors involving $e_k$ and $e_{(k-1)i}$ have been discarded.

$$e_{ki} = (C_1^2 - C_2)e_k^2 e_{(k-1)i} + O(e_{(k-1)i,k})^4 \quad (14)$$

Similar manipulation of the error terms using MATLAB for $e_{k+1}$ also gives the following expression:

$$e_{k+1} = \left(\frac{-3}{2}\right)C_2 e_k^2 e_{(k-1)i} + O(e_{(k-1)i,k})^4 \quad (15)$$

Defining positive real terms $S_k$ and $S_{k-1}$ so that;

$$S_k = \frac{|e_{k+1}|}{|e_k|^\alpha} \text{ and } S_{(k-1)i} = \frac{|e_{ki}|}{|e_{(k-1)i}|^\alpha}$$

$$|e_{(k-1)i}| = |S_{(k-1)i}|^{\frac{-1}{\alpha}}|e_{ki}|^{\frac{1}{\alpha}}$$

From Equation 14 above;

$$\frac{|e_{ki}|}{|e_k|^2|e_{(k-1)i}|} = \frac{|e_{ki}|^{\frac{\alpha-1}{\alpha}}|S_{(k-1)i}|^{\frac{1}{\alpha}}}{|e_k|^2} = |C_1^2 - C_2|$$

$$|e_k| = \frac{|e_{ki}|^{\frac{\alpha-1}{2\alpha}}|S_{(k-1)i}|^{\frac{1}{2\alpha}}}{|C_1^2 - C_2|^{\frac{1}{2}}} \quad (16)$$

Also from Equation 15 above;

$$\frac{|e_{k+1}|}{|e_k^2||e_{(k-1)i}|} = \frac{|S_k||e_k|^{\alpha-2}}{|e_{ki}|^{\frac{1}{\alpha}}|S_{(k-1)i}|^{\frac{-1}{\alpha}}} = \left|\frac{-3}{2}C_2\right|$$

Solving for $e_k$ once again;

$$|e_k| = \frac{\left|\left(\frac{-3}{2}C_2\right)\right|^{\frac{1}{\alpha-2}}|e_{ki}|^{\frac{1}{\alpha(\alpha-2)}}}{|S_k|^{\frac{1}{\alpha-2}}|S_{(k-1)i}|^{\frac{1}{\alpha(\alpha-2)}}} \quad (17)$$

Equating the constants of the expressions for $e_k$ in Equations 16 and 17 gives;

$$\frac{|S_{(k-1)i}|^{\frac{1}{2\alpha}}}{|C_1^2 - C_2|^{\frac{1}{2}}} = \frac{\left|\frac{-3}{2}C_2\right|^{\frac{1}{\alpha-2}}}{|S_k|^{\frac{1}{\alpha-2}}|S_{(k-1)i}|^{\frac{1}{\alpha(\alpha-2)}}}$$

Rearranging further;

$$|S_{(k-1)i}|^{\frac{1}{2(\alpha-2)}}|S_k|^{\frac{1}{\alpha-2}} = \left|\left(\frac{-3}{2}C_2\right)\right|^{\frac{1}{\alpha-2}}|C_1^2 - C_2|^{\frac{1}{2}}$$

Equating expressions containing the $e_k$ term in Equations 16 and 17 gives;

$$|e_{ki}|^{\frac{\alpha-1}{2\alpha}} = |e_{ki}|^{\frac{1}{\alpha(\alpha-2)}}$$

$$|e_{ki}|^{\left(\frac{\alpha-1}{2\alpha}\right)-\left(\frac{1}{\alpha(\alpha-2)}\right)} = 1$$

For the above expression to be true, the exponent term should be equal to zero, i.e.

$$\frac{\alpha-1}{2\alpha} - \frac{1}{\alpha(\alpha-2)} = 0$$

$$\alpha(\alpha-2)(\alpha-1) = 2\alpha$$

$$(\alpha-2)(\alpha-1) = 2$$

$$\alpha^2 - 3\alpha + 2 = 2$$

$$\alpha^2 - 3\alpha = 0$$

$$\alpha(\alpha-3) = 0$$

$$\alpha = 0 \quad or \alpha = 3$$

Since $\alpha = 0$ is not valid solution for a convergent iteration, $\alpha = 3$ is taken as the feasible solution.

Therefore, the iteration process given in Equations 8, 9,10 and 11 has a third order (cubic) convergence.

# 4. Computational Efficiency

The computational efficiency is usually estimated through the formula[10]:

$$\text{Efficiency} = \alpha^{(1/C)} \quad (18)$$

Where $\alpha$= the order convergence of the method and C = the number of functional evaluations required at each step of the iteration. Table 1 below gives a comparison of the computational efficiency of the proposed method with some of the existing methods for computing roots of various orders.

It is clear that, compared with all the methods listed below in Table 1 that have convergence order less than or equal to eight, the proposed method has a higher computational efficiency. Its efficiency is equal to that of a Ninth order method requiring four functional evaluations. This comparison serves to highlight the computational efficiency of the proposed Third Order - Two Point Newton method which requires only two functional evaluations for all the steps of the iteration except the first one.

**Table 1.** Comparison of computational efficiency of the proposed method with some of existing methods

| Method | convergence rate($\alpha$) | Number of functional evaluations(C) | Computational Efficiency (E) |
|---|---|---|---|
| Third Order Two Point Newton Method (Proposed) | 3 | 2 | 1.73 |
| Secant | 1.618 | 1 | 1.62 |
| Newton | 2 | 2 | 1.41 |
| S. Weeraksoon and T.G. Fernando | 3 | 3 | 1.44 |
| Ostrowsky | 4 | 3 | 1.59 |
| Grau and D´ıaz-Barrero | 6 | 4 | 1.57 |
| Kou | 7 | 4 | 1.63 |
| Sharma | 8 | 4 | 1.68 |
| Hu | 9 | 4 | 1.73 |

# 5. Application Examples

**Table 2.** Comparison of result of iterations of the Third Order Two-Point Newton method with Newton Method and the unmodified Two point Newton method

| Equation | Root | Starting points ($x_0$, $x_1$) | Newton Method | Two-point Newton Method (unmodified) | Two-point Newton Method (Third Order) |
|---|---|---|---|---|---|
| $y = x^3+4x^2-10$ | 1.365230013414100 | (0.6, 0.5) | 8 | 6 | 4 |
| | | (0.8, 1) | 6 | 5 | 4 |
| $y = \left[sin(x)\right]^2 - x^2+1$ | -1.404991648215340 | (-0.8, -1) | 7 | 5 | 4 |
| | | (-2.5, -3) | 7 | 6 | 5 |
| $y = x^5+x^4+4x^2-20$ | 1.466279073864720 | (1.2, 1.1) | 6 | 5 | 4 |
| | | (2.0, 1.6) | 6 | 5 | 4 |
| $y = (x-1)^6-1$ | 2.00000000000000 | (1.4, 1.5) | 17 | 8 | 5 |
| | | (2.4, 2.5) | 8 | 6 | 6 |
| | | (3.4, 3.5) | 11 | 8 | 8 |
| $y = sin(x)\cdot e^x + ln(x^2+1)$ | -0.603231971557215 | (-0.9, -0.8) | 7 | 5 | 4 |
| | | (-0.55, -0.65) | 5 | 4 | 4 |
| $y = e^{x^2+7x-30} - 1$ | 3.00000000000000 | (3.8, 4) | 20 | 14 | 11 |
| | | (4.4, 4.5) | 28 | 18 | 15 |
| $y = x-3ln(x)$ | 1.8571838602077840 | (2.1, 2) | 5 | 4 | 4 |
| | | 0.5 | 8 | 5 | 5 |

Equations used to test the efficiency of root finding methods have been used in this paper to evaluate the number of iterations required to reach to a specified level of convergence (Table 2). The stopping criterion used for the iteration process is given by:

$$|x_k - x_{k-1}| + |y_k| < 10^{-15} \quad (19)$$

The rate of convergence$\alpha$, towards the root x = r for each step of the iteration is evaluated using the formula:

$$\alpha = \frac{\text{Log}\,(|e_{k+1}|)}{\text{Log}\,(|e_k|)} = \frac{\text{Log}\,(|x_{k+1}-r|)}{\text{Log}\,(|x_k-r|)} \quad (20)$$

Table 2 above shows a comparison of the proposed Third Order Two-Point Newton method with the traditional Newton method as well as the Two Point Newton method developed earlier[9] and which has a super-quadratic convergence of order 2.414. The equations tested include the ones used to test efficiency of root finding methods elsewhere. A third order (cubic) convergence with which the proposed method converges to the root is mostly evident with the $\alpha$ values being close to 3.0 or above for most of the test equations. It can also be seen from Table 2 that less number of iterations are required to reach to the desired root for the proposed method than the number of iterations required for Newtonmethod and for the original Two Point Newton method.

## 6. Conclusions

A numerical iterative procedure of root finding that uses a third order modification of the Two-Point Newton method has been presented in this paper. It is proved that the method has a third order (cubic) convergence. In terms of computational effort involved, the proposed method displays a higher computational efficiency requiring only two functional evaluations per iteration except for the first iteration step. Most of the higher order methods documented in the literature (third order and above) require evaluation of at least three functions to achieve the desired rate of convergence. This proposed method, on the other hand, achieves a third order convergence with two functional evaluations per iteration. The computational efficiency for the proposed method is comparable with a $9^{th}$ order method that requires four functional evaluations. The computational efficiency is also shown to be greater than $3^{rd}$, $4^{th}$ and higher order methods that requirethree or more functional evaluations per iteration.

The stability of the proposed method against equations for which the traditional Newton method may fail to converge is still retained as in the unmodified Two Point Newton method proposed earlier[9]. This characteristic of stability makes the proposed method applicable to a wide range of equations.

## REFERENCES

[1] S.Weerakoon, T.G.I. Fernando, A variant of Newton's method with accelerated third order convergence, Applied Mathematics Letters 13 (2000) 87-93

[2] A. M. Ostrowski, Solution of Equations in Euclidean and Banach Spaces, Academic Press, New York, NY, USA, 1960.

[3] M. Grau and J. L. D´ıaz-Barrero, "An improvement to Ostrowski root-finding method," Applied Mathematics and Computation, vol. 173, no. 1, pp. 450–456, 2006.

[4] J. R. Sharma and R. K. Guha, A family of modified Ostrowski methods with accelerated sixth order convergence, Applied Mathematics and Computation, vol. 190, no. 1, pp. 111–115, 2007.

[5] C. Chun and Y. Ham, Some sixth-order variants of Ostrowski root-finding methods, Applied Mathematics and Computation, vol. 193, no. 2, pp. 389–394, 2007.

[6] J. Kou, Y. Li, and X. Wang, Some variants of Ostrowski's method with seventh-order convergence, Journal of Computational and Applied Mathematics, vol. 209, no. 2, pp. 153–159, 2007.

[7] W. Bi, H. Ren, and Q. Wu, "New family of seventh-order methods for nonlinear equations," Applied Mathematics and Computation, vol. 203, no. 1, pp. 408–412, 2008.

[8] Zhongyong Hu, Liu Guocai and Li Tian, An Iterative Method with Ninth-Order Convergence for Solving Nonlinear Equations. International Journal of Contemporary Mathematical Sciences, Vol. 6, 2011, no. 1, 17 - 23

[9] A. T. Tiruneh, W.N. Ndlela and S.J. Nkambule. A Two-Point Newton Method suitable for non-convergent Cases and with Super-Quadratic Convergence Advances in Numerical Analysis, Hindawi Publishing Corporation, Volume 2013, Article ID 687382.

[10] R. Sharma and J. R. Sharma, An Efficient family of root-finding methods with optimal eighth-order convergence, Advances in Numerical Analysis, Hindawi Publishing Corporation, Volume 2012, Article ID 346420