

Some Root Finding With Extensions to Higher Dimensions

Tekle Gemechu
Department of mathematics, Adama Science and Technology University (ASTU), Ethiopia

Abstract
Root finding is an issue in scientific computing. Because most nonlinear problems in science and engineering can be considered as the root finding problems, directly or indirectly. The research in numerical modeling for root finding is still going on. In this study, fixed point iterative methods for solving simple real roots of nonlinear equations, which improve convergence of some existing methods, are thorough. Derivative estimations up to the third order (in root finding, some recent ideas) are applied in Taylor's approximation of a nonlinear equation by a cubic model to achieve efficient iterative methods. We may also discuss possible extensions to two dimensions and consider Newton's method and Halley's method in 1D and 2D problem solving. Several examples for test of efficiency and convergence analyses using C++ are offered. And some engineering applications of root finding are conferred. Graphical demonstrations are supported with matlab basic tools.

Keywords: engineering applications, derivative estimations, iterative methods, simple roots, Taylor's approximation.

1. Introduction

Nonlinear equations appear in most science and engineering models. For example, for solving nonlinear differential equations, in circuit analysis, analysis of state equations for a real gas, mechanical motions /oscillations, weather forecasting, in optimization and many other fields of engineering designing processes. Nonlinear problems are difficult to solve but they occur naturally in fluid motions, heat transfer, wave motions, etc.

Some existing iterative root finding methods such as the Secant, bisection, Regula Falsi and Muller's methods need more than one initial guesses. And others may contain higher derivatives, which are sources of algorithm complexities [1, 2, 3, 10, 14, 15, 16, 18]. In this study, based on the author's work in [14, 15], we surrogate the higher derivative f''' in terms of f, f' and f'' to minimize the number of function evaluations which can also facilitate us develop one-point fixed point iterative algorithms in the third order (cubic model) Taylor's approximation/ interpolation of a nonlinear equation $f(x)$. As already discussed in [14], these derivative estimations are not the actual usual function evaluation but a kind of substitution (to reduce algorithm complexity), which we may use only in root finding. Also we have some theoretical and practical modifications in this present work. One good idea is discussion on possible extensions for nonlinear systems in 2D. This article consists of; (1) introduction (2) materials and procedures or construction method (3) Discussion and results (analysis, experiment and results) (4) Possible extensions (5) applications (6) conclusion and recommendation (7) references.

2. Materials and methods

This work is intended to follow and modify the derivative estimation (replacement, for root finding) recently proposed in the author's previous works in [14, 15]. The study uses mixed design approach. We use C++ and matlab as a tool. The procedures are as below.

2.1. Construction methods

In this section, we apply derivative estimation (substitution) technique based on ideas in [14, 15], in the third order (cubic model) Taylor's (interpolation) approximation to present some fixed-point iterative methods.

We shall apply a technique of replacement of a higher derivative ($f'''=F(f,f',f'')$ by lower derivatives (in terms of f, f' and f''). This will reduce the cost of function evaluations at least by one and may increase efficiency index $(e=p^{1/f})$. where p is the order of convergence. One of the higher derivative estimations proposed in [14, 15] is $f'''=F(f,f',f'')$.

When the replacement of this third derivative f'''' is performed by the lower derivatives, the new method will contain only up to the second derivative of $f(x)$. Which is a success. Further replacement can be done to remove even the second derivative as in [15]. And the new methods may be more efficient than the one with the third derivative.

3. Discussions and Results

Let us first present our previous works in [14]. We start with the Taylor's approximation of $f(x)$ about an approximate root $r=x_o+h$, with x_0 an initial guess for a root r of $f(x)$ and $|h|$ is too small.

$$f(x_o+h)=f(x_o)+hf'(x_o)+1/2h^2f''(x_o)+1/6h^3f'''(x_o)+\ldots \quad (1)$$

Consider the estimation,

$$f(x_o+h)=f(x_o)+hf'(x_o)+1/2h^2f''(x_o)+1/6h^3f'''(x_o)\approx 0. \quad (2)$$

$$(2)\Rightarrow hf'+1/6h^3f'''=h(f'+1/6h^2f''') \\ =-(f+1/2h^2f''). \quad (3)$$

Using $h^2=(-\frac{f}{f'})^2$ in (3), we obtain an iterative algorithm (4)

$$x_{n+1}=\varphi(x_n)=x_n-3\frac{2f(x_n)[f'(x_n)]^2+[f(x_n)]^2f''(x_n)}{6[f'(x_n)]^3+[f(x_n)]^2f'''(x_n)}. \quad (4)$$

Inserting $f'''=\frac{(f'')^2}{f'}$, $f'''=\frac{f'f''}{f}$ and $f'''=-f''$ into (4), the algorithms below could be obtained.

$$x_{n+1} = \xi(x_n) = x_n - 3\frac{2f(x_n)[f'(x_n)]^2 + [f(x_n)]^2 f''(x_n)}{6[f'(x_n)]^3 + \frac{[f(x_n)f''(x_n)]^2}{f'(x_n)}}. \quad (5)$$

$$= x_n - \frac{2f(x_n)f'(x_n)}{(2[f'(x_n)]^2 - f(x_n)f''(x_n))(1+1/6\frac{[f(x_n)f''(x_n)]^2}{[f'(x_n)]^4})}. \quad (6)$$

$$x_{n+1} = \theta(x_n) = x_n - 3\frac{2f(x_n)[f'(x_n)]^2 + [f(x_n)]^2 f''(x_n)}{6[f'(x_n)]^3 + f\ (x_n)f'(x_n)f''(x_n)} \quad (7)$$

$$= x_n - \frac{2f(x_n)f'(x_n)}{(2[f'(x_n)]^2 - f(x_n)f''(x_n))(1+1/6\frac{f(x_n)f''(x_n)}{[f'(x_n)]^2})}. \quad (8)$$

$$\phi(x_n) = x_n - 3\frac{2f(x_n)[f'(x_n)]^2 + [f(x_n)]^2 f''(x_n)}{6[f'(x_n)]^3 - [f(x_n)]^2 f''(x_n)}. \quad (9)$$

$$= x_n - \frac{2f(x_n)f'(x_n)}{(2[f'(x_n)]^2 - f(x_n)f''(x_n))(1-1/6\frac{[f(x_n)]^2 f''(x_n)}{[f'(x_n)]^3})} \quad (10)$$

Now let

$$f(x_o) + hf'(x_o) + 1/2h^2 f''(x_o) + 1/6h^3 f'''(x_o) \approx 0. \quad (11)$$

Eqn (11) gives

$$h = \frac{-f}{f' + h/2 f'' + 1/6h^2 f'''}. \quad (12)$$

If $h = -\frac{f}{f'}$ in the right part of (12), then one will get

$$h = \frac{-6f(f')^2}{6(f')^3 - 3ff'f'' + f^2 f'''}. \quad (13)$$

From which we obtain

$$\phi(x_{k+1}) = x_k - \frac{6f(x_k)[f'(x_k)]^2}{6[f'(x_k)]^3 - 3f(x_k)f'(x_k)f''(x_k) + [f(x_k)]^2 f'''(x_k)}. \quad (14)$$

Note also that (14) is an extension of (23) below.

If we use $f''' = \frac{(f'')^2}{f'}$ and $f''' = f''$ in (14), then we get algorithms (15) and (16) respectively.

$$\phi(x_{k+1}) = x_k - \frac{6f(x_k)[f'(x_k)]^2}{6[f'(x_k)]^3 - 3f(x_k)f'(x_k)f''(x_k) + [f(x_k)]^2 \frac{[f''(x_k)]^2}{f'(x_k)}}, \quad (15)$$

$$\phi(x_{k+1}) = x_k - \frac{6f(x_k)[f'(x_k)]^2}{6[f'(x_k)]^3 - 3f(x_k)f'(x_k)f''(x_k) + [f(x_k)]^2 f''(x_k)} \quad (16)$$

Suppose also

$$f(x_o)+hf'(x_o)+1/2h^2 f''(x_o)+1/6h^3 f'''(x_o)\approx 0.$$

$$\Rightarrow\ hf'(x_o)=-(f(x_o)+1/2h^2 f''(x_o)+1/6h^3 f'''(x_o)) \qquad (17)$$

Using $h=-\frac{f}{f'}$ in the right part of (17) gives

$$h=-\frac{1}{f'}[f+1/2(\frac{f}{f'})^2 f''-1/6(\frac{f}{f'})^3 f''']. \qquad (18)$$

From which

$$\phi(x_{k+1})=x_k-\frac{f(x_k)}{f'(x_k)}[1+1/2\frac{f(x_k)f''(x_k)}{[f'(x_k)]^2}-1/6\frac{[f(x_k)]^2 f'''(x_k)}{[f'(x_k)]^3}.] \qquad (19)$$

One can note that (19) is the obvious extension of (24). In (19), if we use

$f'''=\frac{f'f''}{f}$, $f'''=(f'')^2/f'$ then we obtain the algorithms below respectively.

$$\phi(x_{k+1})=x_k-\frac{f(x_k)}{f'(x_k)}[1+1/3\frac{f(x_k)f''(x_k)}{[f'(x_k)]^2}], \qquad (20)$$

$$\omega(x_{k+1})=x_k-\frac{f(x_k)}{f'(x_k)}[1+1/2\frac{f(x_k)f''(x_k)}{[f'(x_k)]^2}-1/6\frac{[f(x_k)]^2\frac{[f''(x_k)]^2}{f'(x_k)}}{[f'(x_k)]^3}]. \qquad (21)$$

The linear approximation of f(x),

$f(x_o)+hf'(x_o)\approx 0$ gives the Newton's method (22)[2, 3, 4, 18]

$$x_{k+1}=x_k-\frac{f(x_k)}{f'(x_k)}:k=0,1,...,n \qquad (22)$$

The linear interpolation also yields an other fixed point method (22b) with only

two function evaluations ($f(x)\,and\,f'(x)$).

$$x_{k+1}=x_k-\frac{f(x_k)f'(x_k)-f(x_k)}{f'(x_k)+f'^2(x_k)}:k=0,1,...,n \qquad (22b)$$

If we consider quadratic interpolation model of $f(x)$ and let

$$f(x_o+h)=f(x_o)+hf'(x_o)+1/2h^2 f''(x_o)\approx 0,$$

then we obtain Halley's method (23) and Chebyshev's method (24) [2, 3, 5, 6, 14, 15].

$$x_{k+1} = x_k - \frac{f(x_k)f'(x_k)}{[f'(x_k)]^2 - 1/2 f(x_k)f''(x_k)} \quad (23)$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} - 1/2\frac{[f(x_k)]^2 f''(x_k)}{[f'(x_k)]^3} \quad (24)$$

3.1. Other additional methods

Consider again

$$f(x_o) + hf'(x_o) + 1/2h^2 f''(x_o) + 1/6h^3 f'''(x_o) \approx 0. \quad (25)$$

$$\Rightarrow hf'(x_o) + 1/2h^2 f''(x_o)$$
$$= -[f(x_o) + 1/6h^3 f'''(x_o)].$$

$$\Rightarrow h = -[f(x_o) + 1/6h^3 f'''(x_o)]/[f'(x_o) + 1/2hf''(x_o)].$$

From which we get

$$\phi(x_k) = x_k - [f(x_k) - 1/6[\frac{f(x_k)}{f'(x_k)}]^3 f'''(x_k)]/[f'(x_k) - 1/2\frac{f(x_k)}{f'(x_k)} f''(x_k)] \quad (26)$$

And from which (27) is obtained.

$$\phi(x_k) = x_k - 2f'(x_k)(f(x_k) - 1/6[\frac{f(x_k)}{f'(x_k)}]^3 f'''(x_k))/(2[f'(x_k)]^2 - f(x_k)f''(x_k)). \quad (27)$$

If we use $f''' = \frac{f'f''}{f}, \frac{[f'']^2}{f}$ in (27),

then we can get algorithms below respectively.

$$\phi(x_k) = x_k - [f(x_k) - 1/6[\frac{f(x_k)}{f'(x_k)}]^3 \frac{f'(x_k)f''(x_k)}{f(x_k)}]/[f'(x_k) - 1/2\frac{f(x_k)}{f'(x_k)} f''(x_k)] \quad (28)$$

$$\cong x_k - 2f'(x_k)(f(x_k) - 1/6[f(x_k)]^2 \frac{f''(x_k)}{[f'(x_k)]^2})/(2[f'(x_k)]^2 - f(x_k)f''(x_k)). \quad (29)$$

$$\varphi(x_k) = x_k - [f(x_k) - 1/6[\frac{f(x_k)}{f'(x_k)}]^3 \frac{[f''(x_k)]^2}{f'(x_k)}]/[f'(x_k) - 1/2\frac{f(x_k)}{f'(x_k)} f''(x_k)] \quad (30)$$

$$= x_k - 2f'(x_k)(f(x_k) - 1/6[f(x_k)]^3 \frac{[f''(x_k)]^2}{[f'(x_k)]^4})/(2[f'(x_k)]^2 - f(x_k)f''(x_k)). \quad (31)$$

3.2. Convergence Analysis

We shall use the following important definition, theorem and statements..

Definition 3.1 A sequence (x_n) generated by an iterative method is said to converge to a root r with order p ≥ 1

if there exists c > 0 such that $e_{n+1} \le c e_n^{\,p}, \forall n \ge n_o$, for some integer $n_0 \ge 0$ and $e_n = |r - x_n|$ [1, 2,3, 14,15].

Theorem 3.1 (Order of Convergence) Assume that $\phi(x)$ has sufficiently many derivatives at a root r of $f(x)$. The order of any one-point iteration function $\phi(x)$ is a positive integer p, more especially $\phi(x)$ has order p if and only if $\phi(r) = r$ and $\phi^{(j)}(r) = 0$ for 0 < j < p, $\phi^{(p)}(r) \ne 0$ [2, 5, 14].

Statement-1: If $\phi = \phi(x, f, f', f'', f''')$ converges to the root r of $f(x)$ for a given initial guess, then $\varphi = \varphi(x, f, f', f'')$ in which, the third derivative f''' is replaced by lower derivatives will converge (probably faster) to the root r with the same initial guess x_0.

Statement-2: In this study, suppose that $\phi(x)$ is expressible in terms of an iteration function $\varphi(x)$ of order p, $f(x)$ and its derivatives as in the proofs below. Then the order q of $\phi(x)$ [14] can be determined and $q \ge p$.

All the algorithms we presented need an appropriate choice of only one suitable initial guess x_o in an interval Io = [a, b][14]. Random choice of x_o leads to unnecessary works, we do not do it [14].

1) **Proof of order of convergence of algorithm in (27)**

We can write (27) as $\phi(x) = x - (x - \varphi(x))H$. With $H(x) = 1 - 1/6\dfrac{[f(x)]^2 f'''(x)}{[f'(x)]^3}$.

And $x = \varphi(x)$ is Halley's iteration function of order 3.

Let r be a simple root of $f(x) = 0$. We have $\varphi(r) = r$, $\varphi(x) = x$ and $\phi(x) = x$.

And $\varphi'(r) = \varphi''(r) = 0$ but $\varphi'''(r) \ne 0$.

Differentiating $\phi(x) = x - (x - \varphi(x))H$, we find that $\phi'(r) = \phi''(r) = 0$ but $\phi'''(r) \ne 0$.

So, $p \ge 3$. Conversely, if p = 3, then we can show that $\phi'(r) = \phi''(r) = 0$ but $\phi'''(r) \ne 0$.

Hence, (27) is third order convergent method by the theorem 3.1 above.

2) To prove order of convergence of algorithm in (31).

We can write (31) as $\varphi(x) = x - (x - \theta(x))T$. Where $T = T(f, f', f'')$.

And $x = \theta(x)$ is Halley's iteration function. Doing as in 1) above we can show that (31) is also a cubic order.

Some other iterative methods can be referred to [14, 15]. And methods of higher orders in this report shall be analyzed in detail in our future works.

3.3. Experimental examples and results

We selected the following equations for tests of efficiency, each with 3 initial guesses.

$f_1(x) = 6x - 3\cos x - 2 = 0$, x_o = 0, 1, 2, r≈0.607102,

$f_2(x) = 3e^{x-1} - \dfrac{3}{x} = 0$, x_o = 0.5, 2, 3, r = 1.000000.

$f_3(x) = 2x^6 - 2x - 2 = 0$, x_o = 1, 2, 3, r ≈ 1.134724 ,

$f_4(x) = 3x^3 - 3x - 3 = 0$, x_o = 1, 2, 3, r ≈ 1.324718.

$f_5(x) = x^4 - x - e^x = 0$, x_o = -1.5, -1, 0 r≈-0.52065.

$f_6(x) = \cos x + x^3 - e^x = 0$, x_o = -2.5, -1, -0.5, r ≈ -0.649565.

$f_7(x) = \log_{10}(x) - 2x + 2$, x_o = 0.5, 1.5, 2.2, r =1.000000.

Newton's method (NM), Chebyshev's method (CM), and the algorithms in equations (4), (6), (27) and (31) were considered. C++ implementation was done for each algorithms and the number of iterations taken to converge to a root r to six decimal places was recorded and written in the body cells of the next **table-1.1** under each method.

The stopping criteria were using the residual error $E_i = f(x_i)$ such that $f(x_i) \leq \varepsilon$, for chosen $\varepsilon = 10^{-7}$.

We also checked this by other stopping criteria in the literature.

The triplets of numbers in each cells of table-1.1 correspond to the number of iterations needed for convergence for each of the three initial guesses of a root r. In the first column, ´**functions** (f) ´´refers to the number of function evaluations, ´**Efficiency** (e) ´´represents the computational efficiency index calculated by e = $\mathbf{p^{1/f}}$. The average number of iteration **Nar** is estimated [14]. In [14] the initial guesses were not given, but we would be free to choose closer to an indicated root or any root. There in [14], the focus was to derive new methods based on derivative replacements. Note that if a method with third order derivative converges to an exact root r, then the new method with third order derivative removed (by the replacement) also converges to r and may be even more efficient.

Table 1.1 Summary of numerical results

f_i	NM	CH	(4)	(6)	(27)	(31)
f_1	3, 3,3	2, 2, 3	2, 2, 3	2, 2, 3	3, 2,3	2, 2,3
f_2	4,4, 4	3, 3, 4	3, 3, 4	3, 3, 4	3,3, 4	3,3, 3
f_3	5, 7,8	4, 5, 6	4, 5, 7	3, 4, 5	4, 4,6	3, 4, 5

f_4	5,5, 6	4, 3, 4	4, 3, 4	4, 3, 4	3,3, 4	3,3, 4
f_5	6,4, 3	4, 3, 3	4, 3, 3	3, 3, 3	3,3, 3	3,3, 3
f_6	7, 5,4	5, 3, 4	5, 3, 4	4, 3, 3	5, 3,3	4, 3,3
f_7	3, 3,3	3, 2, 3	3, 2, 2	3, 2, 3	3, 2,3	3, 2,3
Nar	≈4	≈ 3	≈ 3	≈ 3	3	≈3
p	2	3	3	3	3	3
f	2	3	4	3	4	3
e	1.414	1.442	1.316	1.442	1.316	1.442

From table 1.1 one can study the order p, efficiency index e, number of function evaluations f, and average number of iterations (Nar) of the algorithms. All the methods in the table are relatively efficient.

4. Extensions in 2D

Even though most real world problems consist several independent variables (are multidimensional), studying 1D problem solving is the basis for many 2D problems which is more difficult task. Let us begin with Newton's method to solve systems of nonlinear equations in 2D. Consider a 2D nonlinear systems of equation [1, 3, 18, 19]

$$f = f(x,y) = \begin{Bmatrix} f1 = g(x,y) = 0 \\ f2 = h(x,y) = 0 \end{Bmatrix} \qquad (32)$$

We desire to get X =(x, y) that satisfies f. If $X_{0\,=\,(x0,y0)}$ is an initial guess and $X_{1\,=(x1,y1)}$ is an improved approximation , then one can apply Taylor's linear estimation as

$$f(X1) = f(X0) + f'(X0)(X1 - X0) = 0 \qquad (32b)$$

Where the Jacobean matrix of f is

$$f' = f'(X = (x, y)) = \begin{bmatrix} g_x & g_y \\ h_x & h_y \end{bmatrix} \qquad (33)$$

The linear system (32b) can be solved by elimination, or by Newton's method

$$X1 = X0 - f'(X0)^{-1}(X1 - X0). \qquad (33b)$$

Provided that the inverse $f'(X0)^{-1}$ exists. And the iteration process repeats until convergence. The convex acceleration of Newton's method (33b) in 2D to solve f is [19]

$$X_{k+1} = X_k - \{I + 1/2L(X_k)[I - L(X_k)]^{-1}\} f'(X_k)^{-1} f(X_k) \qquad (34)$$

With I is identity matrix of order 2 and $L = f'(X_k)^{-1} f''(X_k) f'(X_k)^{-1} f(x_k)$ is called the logarithmic degree of convexity of f. Equation (34) is super-Halley's method to solve (32). And Halley's method is given by

$$X_{k+1} = X_k - \{I + 1/2L(X_k)[I - 0.5L(X_k)]^{-1}\} f'(X_k)^{-1} f(X_k) \qquad (35)$$

The Chebyshev's method is

$$X_{k+1} = X_k - \{I + 1/2L(X_k)\} f'(X_k)^{-1} f(X_k) \qquad (36)$$

To see extendibility of some of the methods in this article in 2D via equation (35) or (34) or (33b) above as an example, observe that equation (27) can be expressed as

$\phi(x) = x - (x - \varphi(x))F.$ With $F = F(f, f', f''')$. and $x = \varphi(x)$ is Halley's iterative function in 1D.

Similarly equation (31) can be written as $\varphi(x) = x - (x - \theta(x))T$. Where $T = T(f, f', f'')$. $x = \theta(x)$ is Halley's iteration function in 1D. The same can be done for equations (4), (6) and some others in this article. Anyway we shall make detail studies in our future work.

5. Discussion on some applications

Here, we shall proffer some applications of iterative root finding and investigate convergence graphically. For more examples, refer to [1, 9, 10, 12, 14, 15, 17, 18]. One has to note that to find roots is the only application of root finding in our world.

1) **Manning's equation** to compute the velocity u of water in a rectangular open channel can be given by

$$u = \frac{\sqrt{s}}{n}\left(\frac{BH}{B+2H}\right)^{2/3} [12, 17] \qquad (37)$$

Compute the depth H = d if the roughness coefficient n = 0.035; the slope S = 0.0001;the width B =10 , and u= 0.3624.

Using (37),we obtain the equation to be solved for H= d is to be

$$F(d) = 0.8354(d/(5+d))^{(2/3)} - 0.3624 \qquad (38)$$

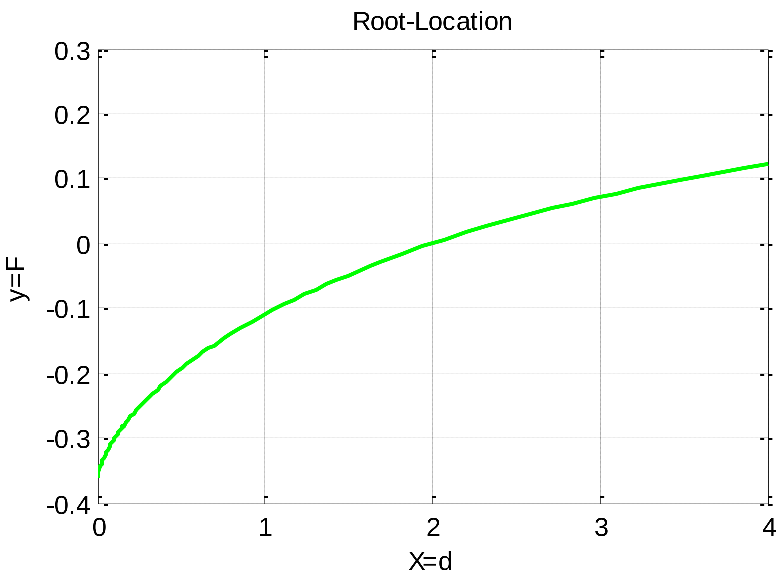

FIGURE 1: A partial graph of F showing a root location

To find the value of H = d for which F(d) = 0 using Newton's method, we need F'. .

$$F' = 4177/7500/(d/(5+d))^{(1/3)}(1/(5+d)-d/(5+d)^2) \qquad (39)$$

And the Newton method for F is

$$d_{i+1} = d_i - F(d_i)/F'(d_i) \qquad (40)$$

With an initial guess of $d_0 = 0.5$, the method converges to the root value d = 2.

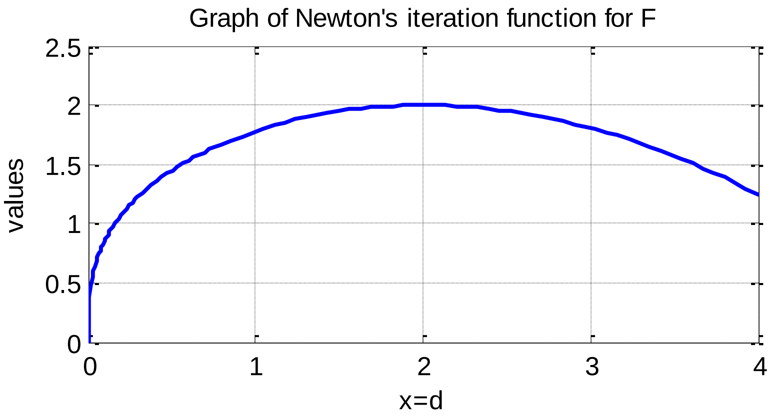

FIGURE 2: Newton's iteration equation showing convergence to a root

2) The model below can be used to estimate Oxygen level c (mg/l) in a river downstream from a sewage discharge [12, 14]:

$$c = 10\text{-}20(e\text{-}0.15x\text{-}e\text{-}0.5x) \quad (41)$$

Where x is the distance downstream in kilometers. Then we may need the value of x for which Oxygen is minimum. We mean the zeros of the first derivative of (41), which is

$$p = 3e^{(-3/20*x)}-10*e^{(-1/2*x)} \quad (42)$$

One may notice that c and x are nonnegative.

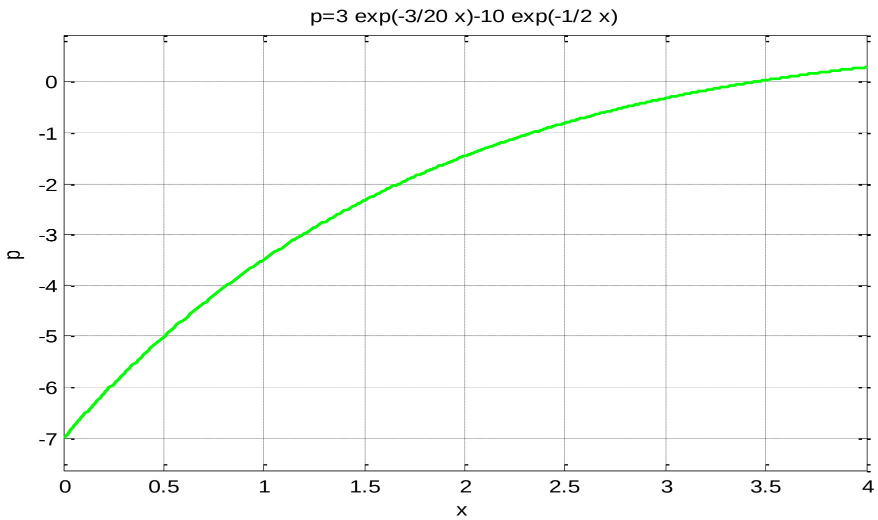

FIGURE 2: A root location for p

Taking an initial guess x0 = 3 and applying Newton's method we get x = 3.4399 with minimum Oxygen level c = 1.6433.

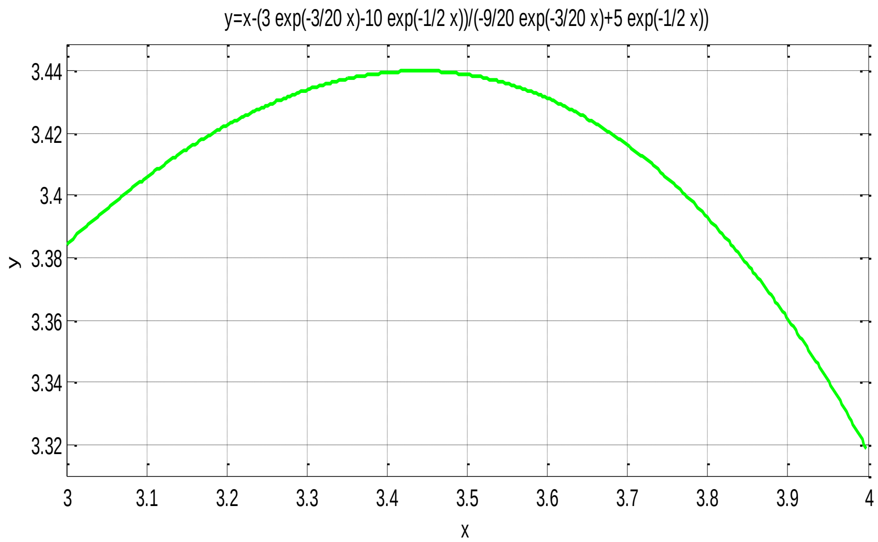

FIGURE 3: Newton fixed point for p on [3, 4]

6. Conclusions and Recommendations

In this study, we presented iterative methods for estimating simple roots of nonlinear equations. We applied derivative estimation (a very new concept in root finding), presented in [14, 15] combined with Taylor's third order approximation and investigated algorithms which are more efficient than some existing methods. We have already stated that if a method containing the third derivative works well, then when the derivative f''' is replaced it can perform better. We observed that estimation of f''' in terms of lower derivatives could result in an efficient method, important in science and engineering. We also discussed possible extensions in 2D and presented some examples of engineering applications. In the future, we will make further analyses of these algorithms. We hope that this result will be more valuable and trigger one to perform further research.

Acknowledgements

I would like to thank any source of contribution and all the mathematics staff of ASTU.

References

[1] Alfio Quarteroni, Riccardo Sacco, Fausto Saleri.(2000), Numerical mathematics (Texts in applied mathematics; 37), Springer-Verlag New York, Inc., USA.

[2] Anthony Ralston, Philip Rabinowitz (1978), A first course in numerical analysis (2nd ed.), McGraw-Hill Book Company, New York.

[3] Germund Dahlquist, and Ake Bjorck (2008), Numerical methods in scientific computing Volume I, Siam - Society for industrial and applied mathematics. Philadelphia, USA.

[4] Jorgen Gerlach (1994), Accelerated Convergence in Newton's Method, Society for industrial and applied mathematics, Siam Review 36, 272-276.

[5] J.Stoer, R. Bulirsch (1993), Texts in Applied mathematics 12, Introduction to Numerical Analysis (2nd ed.), springer-Verlag New York, Inc., USA.

[6] Kou Jisheng, Li Yitian, Wang Xiuhua (2006), A uniparametric Chebyshev-type method free from second derivative, Applied Mathematics and Computational, 179, 296-300.

[7] Ljiljana D.Petkovic, and Miodrag S.Petkovic (2006), on the fourth order root finding methods of Euler type, NOVI SAD J.MATH., 36, 157-165.

[8] M.M. Hussein (2009), A note on one-step iteration methods for solving nonlinear equations, World Applied Sciences Journal 7 (special issue for applied math), IDOSI, Publications, 90-95.

[9] Malvin R.Spencer (PhD Dissertation, 1994), Polynomial Real Root Finding in Bernstein Form, Department of engineering, Brigham Young University.

[10] Pakdemirli, H. Boyacl and H. A. Yurtsever (2008),. A root finding algorithm with fifth order derivatives, Mathematical and Computational Applications, Vol.13, 123-128.

[11] Philipp Rostalski (PHD.Thesis, 2009), Algebraic Moments – Real Root Finding and Related Topics, ETH Zurich (TU Hamburg-Harburg, German), Dissertation ETH Zurich No. 18371

[12] Steven C Chapra, Raymond P Canale (2007), Numerical methods for engineers (5th ed.), Tata McGraw Hill, India.

[13] Shengfeng Li and Rujing Wang (2011), Two fourth order *iterative methods based on Continued Fraction for root* finding problems, World Academy of Science, Engineering and Technology 60

[14] Tekle Gemechu (2016), Root Finding With Some Engineering Applications, International Journal of Innovative Science, Engineering and Technology, Vol. 3, Issue-6, ISSN 2348-7968.

[15] Tekle Gemechu (Dinka)-Dissertation (PhD Thesis, 2014), Perturbation effect of scalar polynomial coefficients and real root finding algorithms with applications, CCNU.

[16] YI JIN AND BAHMAN KALANTARI (communicated by Peter A. Clarkson) (2010), A combinatorial construction of high order algorithms for finding polynomial roots of known multiplicity, American Mathematical Society, 138, 1897-1906.

[17] Steven C Chapra (2012), Applied Numerical methods with Matlab for Engineers and Scientists (3rd ed.), McGraw Hill Education (India), New Delhi

[18] Jain, M.K., Iyenger, S.R.K., and Jain, R.K. (2007): Numerical Methods for Scientific and Engineering Computation. Fifth Edition. New Age International (P) Limited Publishers

[19] Grigore Albeanu (2008), ON THE GENERALIZED HALLEY METHOD FOR SOLVING NONLINEAR EQUATIONS, ROMAI J., 4, 2(2008), 1-6