

A New Trapezoidal-Simpson 3/8 Method for Solving Systems of Nonlinear Equations

Azure Isaac¹, Twum Boakye Stephen², Baba Seidu²

¹Mathematics Department, University for Development Studies, Tamale, Ghana

²Mathematics Department, C. K. Tedam University of Technology and Applied Sciences, Navrongo, Ghana

Email address:

azureike@yahoo.com (A. Isaac)

To cite this article:

Azure Isaac, Twum Boakye Stephen, Baba Seidu. A New Trapezoidal-Simpson 3/8 Method for Solving Systems of Nonlinear Equations. *American Journal of Mathematical and Computer Modelling*. Vol. 6, No. 1, 2021, pp. 1-8. doi: 10.11648/j.ajmcm.20210601.11

Received: December 18, 2020; **Accepted:** January 4, 2021; **Published:** January 15, 2021

Abstract: Since its introduction, the Broyden method has been used as the foundation to develop several other Broyden-like methods (or hybrid Broyden methods) which in many cases have turned out to be improved forms of the original method. The modified classical Broyden methods developed by many authors to solve system of nonlinear equations have been effective in overcoming the deficiency of the classical Newton Raphson method, however there are new trends of methods proposed by authors, which have proven to be more efficient than some already existing ones. This work introduces two Broyden-like method developed from a weighted combination of quadrature rules, namely the Trapezoidal, Simpson 3/8 and Simpson 1/3 quadrature rules. Hence the new Broyden-like methods named by the authors as TS-3/8 and TS – 1/3 methods have been developed from these rules. After subjecting the proposed methods together with some other existing Broyden-like methods to solve four bench-mark problems, the results of numerical test confirm that the TS-3/8 method is promising (in terms of speed and in most cases accuracy) when compared with other proposed Broyden-like methods. Results gathered after the comparison of TS – 3/8 with the other methods revealed that TS – 3/8 method performed better than all the methods in terms of speed and the number of iterations needed to reach a solution. On the other hand, TS – 1/3 method yielded results for all the benchmark problems but with a relatively higher number of iterations compared with the other methods selected for comparison.

Keywords: Broyden Method, Newton-Raphson Method, Quadrature Rules, Trapezoidal-Simpson- 3/8 Rule, Nonlinear Systems, Convergence, Numerical Examples, Trapezoidal-Simpson- 1/3 Rule

1. Introduction

Extracting roots or finding solutions to equations is an important quest in mathematical computations. The roots of equations provide answers to many practical problems. Finding the most efficient numerical method for the purpose is very critical since accuracy of the result for most practical problems is so essential [3]. A problem becomes even more demanding if it requires solving systems of nonlinear equations after modelling.

Solving systems of nonlinear equations is one of the most important problems in numerical computations, especially for a diverse range of engineering applications, including applications in many scientific fields [13]. Several real – life problems can be reduced to solving systems of nonlinear equations, which is one of the most basic problems in mathematics [1]. Great efforts have been made by a lot of

people and many constructive theories and algorithms are proposed to solve systems of nonlinear equations [12]. However there still exist some setbacks in solving such systems. For most traditional numerical methods such as Newton's method, the convergence and performance characteristics can be highly sensitive to the initial guess of the solution. However, it is very difficult to select a reasonable initial guess of a solution for most system of nonlinear equations [10]. The algorithm may fail or the results may be improper if the initial guess of the solution is unreasonable. Many different combinations of the traditional numerical methods and intelligent algorithms are applied to solve systems of nonlinear equations [14, 15], which can overcome the problem of selecting a reasonable initial guess of the solution. But the algorithms are too complicated or expensive to calculate with when there are a number of nonlinear equations to solve simultaneously.

Among the classes of numerical methods for solving system of nonlinear equations, the Newton–Raphson scheme remains popular. However, the Newton–Raphson method is confronted with some drawbacks, a major one of which is the need to compute the inverse Jacobian matrix iteration by iteration. This makes it inefficient for large sized problems especially [20], which serves as the motivation for the current work.

The Broyden method, which is a quasi-Newton method, has seen significant modifications and improvements and these have motivated other researchers to develop new methods capable of solving efficiently nonlinear systems of equations [1]. Many authors continue to present different techniques which are Newton-like schemes [23, 22, 8, 6, 7], Mixed Free Secant methods, or Quadrature formulas. A leading trend of new methods developed for the computation of solutions of systems of nonlinear equations for the past few years has been to formulate using the quadrature rules. Some references in relation to developed methods using quadrature rules include [16-20]

Newton Cotes quadrature rules are a group of formulas for numerical integration based on evaluating the integrand at equally spaced points. Named after Isaac Newton and Roger Cotes [7], they fit data to local order k polynomial approximants. The Newton–Cotes quadrature formulas approximate the integral of a function $\int_a^b f(x)dx$ by evaluating the function at k nodes (x_1, x_2, \dots, x_n) and weighting those nodes with n weights w_1, w_2, \dots, w_n . The most common of Newton-Cotes quadrature formulas are the Mid-point, Trapezoidal and Simpson's rules. The general form of the Newton–Cotes formula is;

$$\int_a^b f(x)dx = \sum_{k=1}^n w_k f(x_k) \quad (1)$$

The Newton's method can be derived from the Taylor's series expansion of a function (of a single variable) $f(x)$ about the point x_1 :

$$f(x) = f(x_1) + (x - x_1)f'(x_1) + \frac{1}{2!}(x - x_1)^2 f''(x_1) + \dots \quad (2)$$

where f , and its first and second derivatives f' and f'' are evaluated at x_1 . In the case of a multiple variable function $F: R^n \rightarrow R^n$, (2) can be shown [18], to equivalently give:

$$F(x) = F(x_k) + \int_{x_k}^x F'(t)dt \quad (3)$$

The matrix of partial derivatives $F'(t)$ appearing in (3) is the Jacobian J , where $\int_{x_k}^x F'(t)dt$ is multiple integrals as in (4):

$$\int_{x_k}^x F'(t)dt = \int_{x_{k,1}}^1 \int_{x_{k,2}}^2 \dots \int_{x_{k,n}}^n F'(x_1, x_2, \dots, x_n) dx_n dx_{n-1} \dots dx_1 \quad (4)$$

The alternative approach is to treat the multiple integral as a nested sequence of one-dimensional integrals, and to use one-dimensional quadrature rule with respect to each argument in turn [9]. Hence we can approximate $\int_{x_k}^x F'(t)dt$ with the weighted combination of quadrature formulas. The authors [6-8, 18-20, 23] and the references therein have proposed various methods for approximating the indefinite integral in equation (4) using Newton Cotes formulae of order zero to one. This

study approximates the integral in Equation (4) by using the weighted combination of the Trapezoidal, Simpson - 3/8 and Simpson - 1/3 quadrature rules.

In this study the following objectives are achieved: (i) Broyden-like methods is developed using combined weights of the Trapezoidal, Simpson 3/8 and Simpson 1/3 quadrature rules; (ii) The new methods are analyzed by comparing the number of iterations and the CPU time with the existing Broyden-like methods using selected systems of nonlinear equations as test problems. In the rest of the paper, section 2.0 describes the general formula of Simpson 3/8 quadrature rule while section 3.0 gives details on how the Trapezoidal-Simpson 3/8 method was derived and the numerical schemes of the Trapezoidal - Simpson 1/3 method, with numerical tests and results well illustrated in section 4.0 and section 5.0 gives a summary conclusion on findings from the research.

2. The General Formula of Simpson 3/8 Quadrature Rule

Most (if not all) of the developed formulas for integration are based on a simple concept of approximating a given function $f(x)$ by a simpler function (usually a polynomial function) $f_i(x)$, where i represents the order of the polynomial function. Simpsons 1/3 rule for integration was derived by approximating the integrand $f(x)$ with a 2nd order (quadratic) polynomial function $f_2(x)$ [2], given by:

$$f_2(x) = a_0 + a_1x + a_2x^2 \quad (5)$$

In a similar way, Simpson 3/8 rule for integration can be derived by approximating the given function $f(x)$ with the 3rd order (cubic) polynomial $f_3(x)$ given as

$$f_3(x) = a_0 + a_1x + a_2x^2 + a_3x^3 = [1, x, x^2, x^3] \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad (6)$$

The unknown coefficients a_0, a_1, a_2 , and a_3 in (6) can be obtained by substituting four known coordinate data points $\{x_0, f(x_0)\}$, $\{x_1, f(x_1)\}$, $\{x_2, f(x_2)\}$ and $\{x_3, f(x_3)\}$ into it as follows:

$$\begin{aligned} f(x_0) &= a_0 + a_1x_0 + a_2x_0^2 + a_3x_0^3 \\ f(x_1) &= a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3 \\ f(x_2) &= a_0 + a_1x_2 + a_2x_2^2 + a_3x_2^3 \\ f(x_3) &= a_0 + a_1x_3 + a_2x_3^2 + a_3x_3^3 \end{aligned} \quad (7)$$

The expression (7) can be put in matrix notation as:

$$\begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ f(x_3) \end{bmatrix} \quad (8)$$

Expression (8) can symbolically be represented as

$$[A]_{4 \times 4} a_{4 \times 1} = f_{4 \times 1} \quad (9)$$

Therefore:

$$a_{4 \times 1} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = [A]^{-1} f \quad (10)$$

Substituting (10) into (6), we obtain (11):

$$f_3(x) = [1, x, x^2, x^3][A]^{-1} f \quad (11)$$

Furthermore

$$\begin{aligned} x_0 &= a \\ x_1 &= a + h = a + \frac{b-a}{3} = \frac{2a+b}{3} \\ x_2 &= a + 2h = a + \frac{2b-2a}{3} = \frac{a+2b}{3} \\ x_3 &= a + 3h = a + \frac{3b-3a}{3} = b \end{aligned}$$

Substituting the form of $f_3(x)$ into $I = \int_a^b f(x)dx$, we have

$$I \approx \int_a^b f_3(x)dx = (b-a) \frac{\{f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)\}}{8} \quad (12)$$

Since $h = \frac{b-a}{3} \Rightarrow b-a = 3h$ and Equation (12) becomes

$$I \approx \frac{3h}{8} \times \{f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)\} \quad (13)$$

Substituting $h = \frac{b-a}{3}$ into Equation (13) and rewriting the whole equation in terms of a and b , the general formula for Simpson's 3/8 quadrature rule is given by (14)

$$\int_{x_k}^x F'(t)dt = \frac{x-x_k}{4} [F'(x_k) + F'(x)] + \left(\frac{x-x_k}{16}\right) \left[F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + F'(x)\right] \quad (19)$$

Substituting (19) into (4), we have

$$F(x) = F(x_k) + \frac{x-x_k}{4} [F'(x_k) + F'(x)] + \left(\frac{x-x_k}{16}\right) \left[F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + F'(x)\right] \quad (20)$$

Since $F(x) = 0$, we get

$$0 = F(x_k) + \frac{x-x_k}{4} [F'(x_k) + F'(x)] + \left(\frac{x-x_k}{16}\right) \left[F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + F'(x)\right] \quad (21)$$

Multiplying through (21) by $\frac{16}{x-x_k}$, we get

$$0 = \frac{16}{x-x_k} F(x_k) + 4[F'(x_k) + F'(x)] + \left[F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + F'(x)\right] \quad (22)$$

Expanding (22) gives

$$0 = \frac{16}{x-x_k} F(x_k) + 4F'(x_k) + 4F'(x) + F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + F'(x) \quad (23)$$

$$0 = \frac{16}{x-x_k} F(x_k) + 5F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) \quad (24)$$

$$\Rightarrow \frac{-16}{x-x_k} F(x_k) = 5F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + 5F'(x) \quad (25)$$

3. Derivation of Trapezoidal–Simpson 3/8 Method (TS -3/8)

A Taylor's series expansion of a function (of a single variable) $f(x)$ about a point x given by

$$f(x) = f(x_1) + (x-x_1)f'(x_1) + \frac{1}{2!}(x-x_1)^2 f''(x_1) + \dots \quad (15)$$

where f and its first and second derivatives, f' and f'' are calculated at x_1 , can be used to derive the Newton's method. For multiple vector variable function F , an analogous expression for it [18], as in (4) is

$$F(x) = F(x_k) + \int_{x_k}^x F'(t)dt \quad (16)$$

where $F(x) = (f_1(x), f_2(x), \dots, f_n(x))'$ and $x = (x_1, x_2, \dots, x_n)$. We assume that x^* is a simple root of the nonlinear equation $F(x) = 0$, an F is sufficiently differentiable. We assume further that $F: D \subset R^n \rightarrow R^n$ is a smooth mapping and has continuous second order partial derivatives on a convex open set $D \subset R^n$ and has a locally unique root x in D . Taking into consideration the two quadrature rules that is:

Trapezoidal quadrature rule

$$\int_a^b F(x) \approx \left(\frac{b-a}{2}\right) (F(a) + F(b)) \quad (17)$$

Simpson 3/8 quadrature rule

$$\int_a^b F(x) \approx \left(\frac{b-a}{8}\right) \left(f(a) + 3f\left(\frac{2a+b}{3}\right) + 3f\left(\frac{a+2b}{3}\right) + f(b)\right) \quad (18)$$

Approximating the integral in (4) by the average of the Trapezoidal and the Simpson 3/8 quadrature rules yields:

$$\Rightarrow x - x_k = \frac{-16F(x_k)}{5F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + 5F'(x)} \quad (26)$$

$$\Rightarrow x = x_k - \frac{16F(x_k)}{5F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + 5F'(x)} \quad (27)$$

$$x = x_k - 16 \left[5F'(x_k) + 3F'\left(\frac{2x_k+x}{3}\right) + 3F'\left(\frac{x_k+2x}{3}\right) + 5F'(x) \right]^{-1} F(x_k) \quad (28)$$

Setting $x = x_{k+1}$ and $x_k = x_k$ in (28), we have

$$x_{k+1} = x_k - 16 \left[5F'(x_k) + 3F'\left(\frac{2x_k+x_{k+1}}{3}\right) + 3F'\left(\frac{x_k+2x_{k+1}}{3}\right) + 5F'(x_{k+1}) \right]^{-1} F(x_k) \quad (29)$$

Setting $F'\left(\frac{2x_k+x_{k+1}}{3}\right) \approx F'\left(\frac{x_k+2x_{k+1}}{3}\right) \approx F'\left(\frac{x_k+x_{k+1}}{2}\right)$, equation (29) becomes

$$x_{k+1} = x_k - 16 \left[5F'(x_k) + 3F'\left(\frac{x_k+x_{k+1}}{2}\right) + 3F'\left(\frac{x_k+x_{k+1}}{2}\right) + 5F'(x_{k+1}) \right]^{-1} F(x_k) \quad (30)$$

$$x_{k+1} = x_k - 16 \left[5F'(x_k) + 6F'\left(\frac{x_k+x_{k+1}}{2}\right) + 5F'(x_{k+1}) \right]^{-1} F(x_k) \quad (31)$$

In (31) we have an implicit equation because of the presence of x_{k+1} on both sides of it. To avoid its implicit nature we use the $(k+1)^{th}$ iteration of the Broyden's method on the right hand side of (31). Thus we have:

$$x_{k+1} = x_k - 16[5F'(x_k) + 6F'(z_k) + 5F'(m_k)]^{-1} F(x_k) \quad (32)$$

with $m_k = x_k - B_k^{-1}F(x_k)$ and $z_k = \frac{x_k+m_k}{2}$

Now replacing $F'(x_k)$, $F'(m_k)$ and $F'(z_k)$ by $B(x_k)$, $B(m_k)$ and $B(z_k)$ respectively and using the same procedure as prescribed in [5, 4, 9], we get

$$x_{k+1} = x_k - 16[5B(x_k) + 6B(z_k) + 5B(m_k)]^{-1} F(x_k) \quad (33)$$

$$\text{Let } B_k = 5B(x_k) + 6B(z_k) + 5B(m_k)$$

$$\Rightarrow x_{k+1} = x_k - 16B_k^{-1}F(x_k) \quad (34)$$

Hence we have the following method using initial matrix $B_0 = I$ and an initial guess x_0 . For a given x_0 using initial matrix $B_0 = I$, an approximated solution for x_{k+1} can be computed by the iterative schemes as in [18];

$$m_k = x_k - B_k^{-1}F(x_k) \quad (35)$$

$$x_{k+1} = x_k - 16[5B(x_k) + 6B(z_k) + 5B(m_k)]^{-1} F(x_k) \quad (36)$$

where: $z_k = \frac{x_k+m_k}{2}$, $k = 0, 1, \dots$

In a similar way as in the above derivations, a weighted combination of the Trapezoidal – Simpson – 1/3 quadrature rules the the numerical scheme as follows;

$$x_{k+1} = x_k - 12[5B(x_k) + 8B(z_k) + 5B(m_k)]^{-1} F(x_k) \quad (37)$$

Algorithm for the TS – 3/8:

1. Given initial guess x_0 , let $k = 0$ and $B_0 = I$
2. Compute $F(x_k)$, if $F(x_k) \leq 10^{-12}$ is satisfied stop, Else go to step 3
3. Compute (m_k) , from (35)
4. Compute $B(m_k)$, using

$$B(m_k) = B(x_k) + \frac{(y_k - B(x_k)s_k)s_k^T}{s_k^T s_k}$$

where

$$y_k = F(m_k) - F(x_k) \text{ and } s_k = m_k - x_k$$

5. Compute $B(z_k)$ using

$$B(z_k) = B(x_k) + \frac{(g_k - B(x_k)b_k)b_k^T}{b_k^T b_k}$$

where

$$z_k = \frac{x_k + m_k}{2}, g_k = F(z_k) - F(x_k), \text{ and } b_k = z_k - x_k$$

6. Compute x_{k+1} from (36)

7. Set $k = k + 1$ and go to step 2, and let $B_{k+1} = B_k + \frac{(y_k - B(x_k)s_k)s_k^T}{s_k^T s_k}$

where

$$y_k = F(x_{k+1}) - F(x_k) \text{ and } s_k = x_{k+1} - x_k$$

Convergence of the TS–3/8 Method

The properties of the local convergence for the proposed method are presented here with the following standard assumptions on the nonlinear function F :

1. F is differentiable in an open convex set $D \in R^n$.
2. There exist $x^* \in D \subset R^n$ such that $F(x^*) = 0$ and $F(x^*)$ is continuous for every $x \in D$.
3. $F'(x)$ is Lipschitz continuous of order 1 so that there exists a positive constant λ such that

$$\|F(x) - F(y)\| \leq \lambda \|x - y\| \quad \forall x, y \in R^n$$

Definition 1.0 (q-super-linear convergence) [11]

Let x_k and $x^* \in R^n$. Then $x_k \rightarrow x^*$ is q – superlinear if

$$\lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0$$

Lemma 1.1 [21]

Let $F: R^n \rightarrow R^n$ be continuous and differentiable on an

open convex set $D \subset R^n, x \in D$. If $F'(x)$ is Lipschitz continuous with Lipschitz constant λ , then for any $u, v \in D$ $\|F(v) - F(u) - F'(x)(v - u)\| \leq \lambda \max\{\|u - x\|, \|v - x\|\}$. Moreover, if $F'(x)$ is invertible, then there exists ϵ and $\rho > 0$ such that $\frac{1}{\rho}\|v - u\| \leq \|F(v) - F(u)\| \leq \rho\|v - u\|$ for all $u, v \in D$ for which

$$\lambda \max\{\|u - x\|, \|v - x\|\} \leq \epsilon.$$

Lemma 1.2 [21]

Let $x_k \in R^n, k \geq 0$. If x_k converges q-super-linearly to $x^* \in R^n$, then

$$\lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0$$

Here, we present the main result which is a modified result in [16] to prove that the local order of convergence analysis

is super-linear.

Theorem 1.0

Let $F: R^n \rightarrow R^n$ satisfy the hypothesis of Lemma 1.1 on the set D . Let B_k be a sequence of non-singular matrices in the linear space $L(R^n)$ of real matrices of order n . Suppose for some x_0 the sequence x_k generated by (24) remains in D and $\lim_{k \rightarrow \infty} x_k = x^*$ for each $x_k \neq x^*$. Then the sequence $\{x_k\}$ converges q-super-linearly to x^* and $F(x^*) = 0$ if and only if

$$\lim_{k \rightarrow \infty} \frac{\left\| \frac{1}{16} B_k F'(x^*) s_k \right\|}{\|s_k\|} = 0 \quad (38)$$

Where $s_k = x_{k+1} - x_k$ and $B_k = 5B(x_k) + 3B(z_k) + 5B(m_k)$.

Proof

Given that (38) holds; it implies that (35) becomes

$$0 = \frac{1}{16} B_k s_k + F(x_k) \quad (39)$$

$$0 = \frac{1}{16} B_k s_k + F(x_k) - F'(x^*) s_k + F'(x^*) s_k \quad (40)$$

$$0 = \frac{1}{16} B_k s_k - F'(x^*) s_k + F(x_k) + F'(x^*) s_k \quad (41)$$

$$-F(x_{k+1}) + F(x_{k+1}) = \left(\frac{1}{16} B_k - F'(x^*) \right) s_k + F(x_k) + F'(x^*) s_k \quad (42)$$

$$-F(x_{k+1}) = \left(\frac{1}{16} B_k - F'(x^*) \right) s_k + (-F(x_{k+1}) + F(x_k) + F'(x^*) s_k) \quad (43)$$

Taking norm of both sides, we have:

$$\| -F(x_{k+1}) \| = \left\| \left(\frac{1}{16} B_k - F'(x^*) \right) s_k + (-F(x_{k+1}) + F(x_k) + F'(x^*) s_k) \right\| \quad (44)$$

Using vector norm properties, it implies that;

$$\| -F(x_{k+1}) \| \leq \left\| \left(\frac{1}{16} B_k - F'(x^*) \right) s_k \right\| + \| (-F(x_{k+1}) + F(x_k) + F'(x^*) s_k) \| \quad (45)$$

Dividing through by $\|s_k\|$, we have;

$$\frac{\| -F(x_{k+1}) \|}{\|s_k\|} \leq \frac{\left\| \left(\frac{1}{16} B_k - F'(x^*) \right) s_k \right\|}{\|s_k\|} + \frac{\| (-F(x_{k+1}) + F(x_k) + F'(x^*) s_k) \|}{\|s_k\|} \quad (46)$$

Using Lemma 1.1

$$\| -F(x_{k+1}) \| \leq \frac{\left\| \left(\frac{1}{16} B_k - F'(x^*) \right) s_k \right\|}{\|s_k\|} + \lambda \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\} \quad (47)$$

Since $x_{k+1} \rightarrow x^* \forall k$, then from (21), we have

$$\lim_{k \rightarrow \infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} \leq \frac{\left\| \left(\frac{1}{16} B_k - F'(x^*) \right) s_k \right\|}{\|s_k\|} + \lambda \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\} \quad (48)$$

$$F(x^*) = F(\lim_{k \rightarrow \infty} x_k) = \lim_{k \rightarrow \infty} F(x_k) = 0$$

But $F'(x^*)$ is non-singular, thus by Lemma 1.1 $\exists \rho > 0, k_0 \geq 0$ such that we have;

$$\|F(x_{k+1})\| = \|F(x_{k+1}) - F(x^*)\| \geq \frac{1}{\rho} \|x_{k+1} - x^*\| \quad (49)$$

For all $k \geq k_0$, we have

$$\begin{aligned}
0 &= \lim_{k \rightarrow \infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} \\
&\geq \lim_{k \rightarrow \infty} \frac{1}{\rho} \frac{\|x_{k+1} - x^*\|}{\|s_k\|} \\
&\geq \lim_{k \rightarrow \infty} \frac{1}{\rho} \frac{\|x_{k+1} - x^*\|}{\|x_{k+1} - x^*\| + \|x_{k+1} - x^*\|} \\
&= \lim_{k \rightarrow \infty} \frac{\frac{1}{\rho} t_k}{1 + t_k}
\end{aligned} \tag{50}$$

where

$$t_k = \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|}$$

It implies that

$$\lim_{k \rightarrow \infty} t_k = 0$$

Therefore x_k converges q-super-linearly to x^* . Conversely, supposed that x_k converges q-super-linearly to x^* and $F(x^*) = 0$. Then by Lemma 1.1, there exist a $\rho > 0$ such that we have

$$\|F(x_{k+1})\| \leq \rho \|x_{k+1} - x^*\| \tag{51}$$

Then

$$\begin{aligned}
0 &= \lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \\
&\geq \lim_{k \rightarrow \infty} \frac{1}{\rho} \frac{\|F(x_{k+1})\|}{\|x_k - x^*\|}
\end{aligned} \tag{52}$$

$$= \lim_{k \rightarrow \infty} \frac{\|F(x_{k+1})\|}{\rho \|s_k\|} \frac{\|s_k\|}{\|x_k - x^*\|} \tag{53}$$

Using Lemma 1.2, we have

$$\lim_{k \rightarrow \infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} = 0 \tag{54}$$

It implies that

$$\frac{\left\| \left(\frac{1}{16} B_k - F'(x^*) \right) s_k \right\|}{\|s_k\|} \leq \lim_{k \rightarrow \infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} + \lim_{k \rightarrow \infty} \frac{\|(-F(x_{k+1}) + F(x_k) + F'(x^*)s_k)\|}{\|s_k\|} \leq 0 + \lambda \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\} \tag{55}$$

Since x_k converges to x^* , then

$$\lim_{k \rightarrow \infty} \|x_k - x^*\|$$

Which proves that

$$\lim_{k \rightarrow \infty} \frac{\left\| \left(\frac{1}{16} B_k - F'(x^*) \right) s_k \right\|}{\|s_k\|} = 0$$

4. Numerical Tests

In order to evaluate the performance of the new method (i.e. the Trapezoidal Simpson-3/8, denoted as TS-3/8), it was tested, together with three other well-known methods (i.e. Classical Broyden Method (CB), Trapezoidal-Simpson Method (TS), and Trapezoidal Simpson Midpoint Method (TSM), on four benchmark problems [18], using a set of seven dimensions

ranging from 5 to 1065 variables. The results were then compared on the bases of two main characteristic features namely, the number of iterations (NI) to convergence and the Central Processing Unit (CPU) time in seconds. The computation was done in MATLAB 2020R with a double precision arithmetic on a computer with specification as follows; processor: AMD EI-2100APU with Radeon™ Graphics 1.00GHz, Installed memory (RAM): 4.00GB and the system type is 64-bit Operating System, x 64-based processor. The programme was designed to terminate whenever the number of iterations reached 500 and when x_k satisfied $\|F(x_k)\| \leq 10^{-12}$; any failure of a method to satisfy these convergence criteria is denoted by a dash (i.e. '-') as in Table 1.

The problems are:

Problem One

$$F_i(x) = x_i x_{i+1} - 1, F_n(x) = x_n x_1 - 1, i = 1, 2, \dots, n -$$

$$1 \text{ and } x_0 = (0.8, 0.8, \dots, 0.8)^T$$

Problem Two

$$F_i(x) = x_i x_{i+1} - 1, F_n(x) = x_n x_1 - 1, i = 1, 2, \dots, n - 1 \text{ and } x_0 = (0.5, 0.5, \dots, 0.5)^T$$

Problem Three

$$F_i(x) = x_i^2 - \cos(x_1 - 1), i = 1, 2, \dots, n \text{ and } x_0 = (0.5, 0.5, \dots, 0.5)^T$$

Problem Four

$$F_i(x) = \exp(x_i^2 - 1) - \cos(1 - x_i^2), i = 1, 2, \dots, n \text{ and } x_0 = (0.5, 0.5, \dots, 0.5)^T$$

5. Results and Discussion

Methods developed in this research (Trapezoidal – Simpson 3/8 and Trapezoidal – Simpson – 1/3 methods) were compared with existing methods (Trapezoidal – Simpson and Trapezoidal – Simpson – Midpoint methods). The first and second results in Table 1, corresponds to the existing methods Trapezoidal – Simpson – Midpoint (TSM) and Trapezoidal – Simpson (TS) methods respectively, whilst the third and fourth methods are the results of the developed new methods namely the Trapezoidal – Simpson 1/3 (TS – 1/3) and Trapezoidal – Simpson 3/8 (TS – 3/8) methods.

Table 1. Numerical Results of all Four Methods.

Problem	n	1		2		3		4	
		TSM		TS		TS – 1/3		TS -3/8	
		NI	CPU	NI	CPU	NI	CPU	NI	CPU
1	5	9	0.266	4	0.198	23	0.217	4	0.186
	35	35	0.190	4	0.178	23	0.206	4	0.201
	65	53	0.226	4	0.188	23	0.322	4	0.188
	165	123	0.415	4	0.202	23	0.337	4	0.209
	365	218	1.191	4	0.241	23	0.361	4	0.213
	665	477	7.353	4	1.255	23	0.711	4	0.314
	1065	769	31.838	4	0.399	23	0.218	4	0.398
2	5	11	1.057	4	0.185	24	0.208	4	0.220
	35	36	0.204	4	0.189	24	0.226	4	0.210
	65	60	0.221	4	0.192	24	0.204	4	0.187
	165	146	0.409	4	0.197	24	0.253	4	0.239
	365	301	2.025	4	0.221	24	0.330	4	0.244
	665	498	6.125	4	0.286	24	0.911	4	0.278
	1065	804	22.414	4	0.451	24	1.564	4	0.859
3	5	9	0.246	4	0.224	25	0.223	7	0.206
	35	9	0.218	7	0.183	24	0.213	7	0.208
	65	9	0.210	7	0.197	24	0.238	7	0.202
	165	9	0.239	7	0.236	24	0.253	7	0.220
	365	9	0.236	7	0.260	24	0.359	7	0.280
	665	9	0.327	7	0.289	24	0.780	7	1.238
	1065	9	0.607	7	0.418	24	1.036	7	0.659
4	5	4	0.271	4	0.259	24	1.283	4	0.202
	35	-	-	4	2.522	24	0.233	4	2.635
	65	-	-	4	8.343	24	0.220	4	8.343
	165	-	-	4	47.078	24	0.273	4	51.077
	365	-	-	4	232.026	24	0.485	4	227.191
	665	-	-	4	773.373	24	0.576	4	759.877
	1065	-	-	4	1926.21	24	1.094	4	1979.338

Table 1 presents the results for solving the four benchmark problems with each of the four methods. The results indicate clearly that the proposed methods did not fail to meet the convergence criteria specified for all the selected benchmark problems. An observation from the table revealed that TSM method was unable to obtain solutions for problem four with n values equal to 35, 65, 165, 365 665 and 1065, in addition, it recorded the highest number of iterations for most of the problems solved. TS and TS – 3/8 methods recorded the lowest number of iterations for all four benchmark problems with each one of them recording the same number of iterations for each problem. Another important observation made from the data above shows clearly that the new

developed TS – 3/8 method required lesser CPU time to execute all four problems under consideration with n values equal to 5, 35, 65, 165, 365 and 665 however in most cases for n equal to 1065, TS method recorded a lesser CPU time compared to TS – 3/8 method. Lastly, The TS – 1/3 method also proposed in this research recorded a relatively high CPU time for all the problems compared with TS and TS – 3/8 methods.

6. Conclusions

This paper has proposed and developed two new Broyden – like methods for solving system of nonlinear equations.

The methods namely TS - 3/8 and TS - 1/3 methods, preserved the local order convergence of the classical Broyden method with TS - 3/8 method performing extremely well in terms of a lesser number of iterations and CPU time required for all computations when compared with other existing methods. In addition, the TS - 3/8 method well just as the already existing TS method in terms of the number of iterations for each of the benchmark problems, however, It did better than the TS - method in terms of the CPU time required for values of n equal to 5, 35, 65, 165, 365 and 665 whilst for values of n equal to 1065, TS method had a lesser CPU time. This can be concluded that though the TS - 3/8 method will yield a lesser CPU time for smaller values of n , TS method may perform equally well or better for some higher values of n , hence from the data gathered in this research the TS - 3/8 method out performs the TS method and the other three methods for all benchmark problems for smaller values of n .

Results gathered from this research using the newly developed TS-3/8 method, suggests that weighted combination of quadrature rules such as Simpson 1/3 and 3/8 against known ones like Midpoint, Simpson and Trapezoidal rules might equally yield much more better iterative schemes capable of solving system of nonlinear equations.

References

- [1] Al-Towaiq, M. H., & Abu Hour, Y. S. (2017). Two improved classes of Broyden's methods for solving nonlinear systems of equations. *JOURNAL OF MATHEMATICS AND COMPUTER SCIENCE-JMCS*, 17 (1), 22-31.
- [2] Autar, K., Egbu, E. K., Duc, N. (2017). *Numerical Methods with Applications*. <http://numericalmethodng.usf.edu>.
- [3] Azure, I., Aloliga, G., & Doabil, L. (2020). Comparative Study of Numerical Methods for Solving Non-linear Equations Using Manual Computation. *Mathematics Letters*, 5 (4), 41.
- [4] Cordero, A., Torregrosa, J. R., & Vassileva, M. P. (2012). Pseudocomposition: a technique to design predictor-corrector methods for systems of nonlinear equations. *Applied Mathematics and Computation*, 218 (23), 11496-11504.
- [5] Cordero, A., & Torregrosa, J. R. (2006). Variants of Newton's method for functions of several variables. *Applied Mathematics and Computation*, 183 (1), 199-208.
- [6] Darvishi, M. T., & Shin, B. C. (2011). High-order Newton-Krylov methods to solve systems of nonlinear equations. *Journal of the Korean Society for Industrial and Applied Mathematics*, 15 (1), 19-30.
- [7] Dhamacharoen, A. (2014). An efficient hybrid method for solving systems of nonlinear equations. *Journal of Computational and Applied Mathematics*, 263, 59-68.
- [8] Frontini, M. A. R. C. O., & Sormani, E. (2003). Some variant of Newton's method with third-order convergence. *Applied Mathematics and Computation*, 140 (2-3), 419-426.
- [9] Hafiz, M. A., & Bahgat, M. S. (2012). An efficient two-step iterative method for solving system of nonlinear equations. *Journal of Mathematics Research*, 4 (4), 28.
- [10] Jain, M. K., Iyengar, S. R. K & Jain, R. K. (2012). *Numerical Methods for Scientific and Engineering Computation (Sixth Edition)*. New Age International (P) Limited, Publishers, New Delhi.
- [11] Kelley, C. T. (1995). *Iterative methods for linear and nonlinear equations*. Society for Industrial and Applied Mathematics.
- [12] Kou, J., Li, Y., & Wang, X. (2007). A composite fourth-order iterative method for solving non-linear equations. *Applied Mathematics and Computation*, 184 (2), 471-475 (11).
- [13] Li, Y., Wei, Y., & Chu, Y. (2015). Research on solving systems of nonlinear equations based on improved PSO. *Mathematical Problems in Engineering*, 2015 (1).
- [14] Luo, Y. Z., Tang, G. J., & Zhou, L. N. (2008). Hybrid approach for solving systems of nonlinear equations using chaos optimization and quasi-Newton method. *Applied Soft Computing*, 8 (2), 1068-1073 (12).
- [15] Mo, Y., Liu, H., & Wang, Q. (2009). Conjugate direction particle swarm optimization solving systems of nonlinear equations. *Computers & Mathematics with Applications*, 57 (11-12), 1877-1882 (13).
- [16] Mohammad, H., & Waziri, M. Y. (2015). On Broyden-like update via some quadratures for solving nonlinear systems of equations. *Turkish Journal of Mathematics*, 39 (3), 335-345.
- [17] Muhammad, K., Mamat, M., & Waziri, M. Y. (2013). A Broyden's-like Method for solving systems of Nonlinear Equations. *World Appl Sc J*, 21, 168-173.
- [18] Osinuga, I. A., & Yusuff, S. O. (2018). Quadrature based Broyden-like method for systems of nonlinear equations. *Statistics, Optimization & Information Computing*, 6 (1), 130-138.
- [19] Osinuga I. A. & Yusuff, S. O. (2018). A robust Broyden-like method for systems of nonlinear equations. *Transactions of the Nigerian Association of Mathematical Physics*, 6 (Jan., 2018), 81-89.
- [20] Osinuga, I. A., & Yusuff, S. O. (2017). Construction of a Broyden-like method for Nonlinear systems of equations. *Annals. Computer Science Series*, 15 (2), 128-135.
- [21] Van de Rotten, B., & Lunel, S. V. (2005). A limited memory Broyden method to solve high-dimensional systems of nonlinear equations. In *EQUADIFF 2003* (pp. 196-201).
- [22] Weerakoon, S., & Fernando, T. G. I. (2000). A variant of Newton's method with accelerated third-order convergence. *Applied Mathematics Letters*, 13 (8), 87-93.
- [23] Yusuff, S. O., Osinuga, I. A., Adeniran, O. J., & Onashoga, S. A. (2019). ROBUST THREE-STEP BROYDEN-LIKE ALGORITHMS FOR FUNCTIONS OF SEVERAL VARIABLES. *Annals. Computer Science Series*, 17 (2).