A matrix-free approach for solving systems of nonlinear equations

Article · January 2016

DOI: 10.20454/jmmnm.2016.1025

2 authors:

Auwal Bala Abubakar
King Mongkut's University of Technology Tho…
3 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Marianus Waziri
Bayero University, Kano
21 PUBLICATIONS **62** CITATIONS
SEE PROFILE

Journal of Modern Methods in Numerical Mathematics 7:1 (2016), 1–9

ModernScience Publishers

JOURNAL OF MODERN METHODS IN NUMERICAL MATHEMATICS

journal homepage: http://www.m-sciences.com/index.php?journal=jmmnm

A matrix-free approach for solving systems of nonlinear equations

Auwal Bala Abubakar[a,*], Mohammed Yusuf Waziri[a]

[a]*Department of Mathematical Sciences, Faculty of Science, Bayero University, Kano, Nigeria.*

Abstract

The computation of the Jacobian, its inverse and $n \times n$ matrix storage requirement in each iteration limits the application of Newton-like method for solving systems of nonlinear equations, because computing the Jacobian entails derivative and to compute its inverse, it must be non-singular. We present an enhancement of a three-step Newton-like method by J. R. Sharma and P. Gupta (2014) where the Jacobian inverse is approximated into a diagonal matrix. Numerical results proved that the proposed method in terms of CPU time and number of iterations compared to the Sharma and Gupta's method is very encouraging.

Keywords: Nonlinear systems, Newton-like method, linear convergence, matrix inversion.
2010 MSC: 65K05, 58C15, 41A25, 15A09

1. Introduction

Consider the system of nonlinear equations

$$F(x) = 0, \tag{1}$$

where $F : D \subset R^n \to R^n$ is a mapping satisfying the following assumptions

1. there exist $x^* \in D$ with $F(x^*) = 0$;
2. F is continuously differentiable in a neighborhood of x^*;
3. $F'(x^*)$ is nonsingular.

One of the basic procedure for solving (1) is the classical Newton's method which converges quadratically provided a good initial guess is given [3, 6]. It is defined by

$$x_{k+1} = x_k - (F'(x_k))^{-1}F(x_k), \tag{2}$$

*Corresponding author
Email addresses: ababubakar.mth@buk.edu.ng (Auwal Bala Abubakar), mywaziri@gmail.com (Mohammed Yusuf Waziri)

Received: 12 December 2015 Accepted: 12 January 2016
http://dx.doi.org/10.20454/jmmnm.2016.1025
2090-8296

where $k = 0, 1, 2 \ldots$ and $F'(x_k)$ is the Jacobian matrix which is assumed to be nonsingular [1, 4]. Despite its good qualities, it has some shortcomings as it requires computing, storing the Jacobian matrix and solving n linear system in each iteration.

Several modifications have been proposed in literature, in order to overcome the shortcomings of Newton's method. For example, in [13] Darvishi and Barati presented a fourth order method from quadrature formulae to solve systems of nonlinear equations . Also in [9], Homeier modified Newton's method for root finding with cubic order of convergence, which he later in [10] extend to multi-variable case. However, in (2010) Waziri *et al.*[14], proposed a new Newton's method with diagonal Jacobian approximation for systems of nonlinear equations and in [15], a Jacobian free diagonal Newton's method for solving nonlinear systems of equations was given. Recently, some improvements of Newton-like methods were considered in [8], and references therein.

2. Method of Sharma and Gupta (AEM)

In their paper titled "An efficient fifth order method for solving systems of nonlinear equations", Sharma and Gupta presented a three-step iterative method of convergence order five. This methodology was based on the two-step Homeier's method with cubic convergence [10]. They considered a well-known fact in numerical analysis that the construction of higher order iterative methods is a noneffective exercise unless they have low computational cost.

The main goal and motivation in this development of iterative methods is to achieve as high as possible convergence order which requires as small as possible functions evaluations, derivative and matrix inversions. With this in mind, they proposed a three-step method of fifth order convergence based on [10]. By condering the Homeier's method [10], which is defined as

$$\begin{aligned} p_k &= x_k - \frac{1}{2}F'(x_k)^{-1}F(x_k), \\ z_k &= x_k - F'(p_k)^{-1}F(x_k), \\ k &= 0, 1, 2, ..., n. \end{aligned} \tag{3}$$

Sharma and Gupta now constructed a method to obtain the approximation x_{k+1} to a solution $F(x) = 0$ by considering the scheme defined as

$$\begin{cases} p_k &= x_k - \frac{1}{2}F'(x_k)^{-1}F(x_k), \\ z_k &= x_k - F'(p_k)^{-1}F(x_k), \\ x_{k+1} &= z_k - [2F'(p_k)^{-1} - F'(x_k)^{-1}]F(z_k). \end{cases} \tag{4}$$

To obtain an assesment of the efficiency of the proposed method , they made use of efficiency index, according to which the efficiency of an iterative method is given by $E = P^{\frac{1}{c}}$, where P is the convergence order and c is the computational cost per iteration. In addition, they compare the efficiency of their proposed method computationally with the third order Homeier's method [10], the fourth order method by Darvishi and Barati [13], which requires the evaluation of two functions, three different Jacobians and two matrices inversions, the fourth order Generalised Jarrats method [11] requiring the evaluation of one function, two different Jacobians and two matrix inversions, the fourth order Generalised Ostrowski's method [7] that requires the evaluation of two functions, one Jacobian, one divided difference and two matrix inversions, and the fifth order method by Cordero et al. [2] requiring the evaluations of two functions, two different Jacobian and three matrix inversions. From the numerical results, they observed that like the existing methods their method showed consistent convergence behaviour.

Despite its advantage over Newton's method, i.e., it possesses higher convergence order than Newton's method. But still requires the computation of two Jacobian matrices and their inverses [12]. Below is the algorithm of the method (AEM).

2.1. AEM Algorithm

Algorithm 2.1. Given an initial guess x_0, ε, and set $k = 0$.

Step 1: Compute $F(x_k)$, if $\|F(x_k)\| \leq \varepsilon$ stop. Else go to step 2.

Step 2: Compute $F'(x_k)$ and $F'(x_k)^{-1}$.

Step 3: Compute $p_k = x_k - \frac{1}{2}F'(x_k)^{-1}F(x_k)$.

Step 4: Compute $F'(p_k)^{-1}$.

Step 5: Compute $z_k = x_k - F'(p_k)^{-1}F(x_k)$.

Step 6: Compute $F(z_k)$.

Step 7: Compute $x_{k+1} = z_k - (2F'(p_k)^{-1} - F'(x_k)^{-1})F(z_k)$.

Step 8: Set $k = k + 1$, and go to step 1.

In this paper, we present an extension of the (AEM) method for solving nonlinear systems by approximating the Jacobian inverse involved into a diagonal matrix. The rationale behind the approach is to reduce the cost of computing and storing of the Jacobian, as well as solving n linear equations per iteration.

The outline of the paper is as follows. In section 2, we present our propose method. In section 3, some numerical results were presented. Section 4 contains the conclusion.

3. A matrix-free approach for solving systems of nonlinear equations (AMFA)

The three-step iterative scheme (4) of fifth order requires two Jacobian computations, one at x_k and the other at p_k which is sometimes costly due to derivative requirement and storage. In this section we propose to approximate the two Jacobian inverse matrices with derivative free matrices (Diagonal matrices). The idea behind this approach is to get an approximation of the two Jacobian inverse matrices without even computing the Jacobian, which will enable us to avoid the point in which the Jacobian is singular and to reduce cost of computation and storage of the Jacobian and its inverse, as well as solving n linear equations in each iteration.

Let the Jacobian matrices in (4), $F'(\mathrm{x}_k)^{-1}$, $F'(\mathrm{p}_k)^{-1}$ be approximated by certain diagonal matrices D_{x_k}, D_{p_k} and $2D_{p_k} - D_{x_k} = D_{z_k}$. So that

$$F'(\mathrm{x}_k)^{-1} \approx D_{\mathrm{x}_k} = diag(d^{(i)}_{\mathrm{x}_k}), \text{ where } d^{(i)}_{\mathrm{x}_k} = \frac{\mathrm{x}^{(i)}_k - \mathrm{z}^{(i)}_k}{F_i(\mathrm{x}_k) - F_i(\mathrm{z}_k)},\ i,k = 1,2,...,n. \tag{5}$$

and

$$F'(\mathrm{p}_k) \approx D_{\mathrm{p}_k} = diag(d^{(i)}_{\mathrm{p}_k}),\ d^{(i)}_{\mathrm{p}_k} = \frac{\mathrm{p}^{(i)}_k - \mathrm{x}_k{}^{(i)}}{F_i(\mathrm{p}_k) - F_i(\mathrm{x}_k)},\ i,k = 1,2,...,n. \tag{6}$$

Then, we have

$$\begin{cases} p_k & = x_k - \frac{1}{2}D_{x_k}F(x_k) \\ z_k & = x_k - D_{p_k}F(x_k) \\ x_{k+1} & = z_k - D_{z_k}F(z_k). \end{cases} \tag{7}$$

Scheme (7) above requires two evaluation of functions, 0 derivative and 0 matrix inversion. In the following Algorithm, we present the stages of implementation of our method:

3.1. AMFA Algorithm

Algorithm 3.1. Given an initial guess x_0, ε, and set $k = 0$.

Step 1: compute $F(x_k)$, If $\|F(x_k)\| \leq \varepsilon$ stop. Else goto step 2.

Step 2: compute $p_k = x_k - \frac{1}{2}D_{x_k}F(x_k)$ using (4), provided that $|F_i(p_k) - F_i(x_k)| > \varepsilon$, else set $d^{(i)}_{x_k} = d^{(i)}_{x_{k-1}}$, for $i, k = 1, 2, ..., n$.

Step 3: compute $z_k = x_k - D_{p_k}F(x_k)$ using (5), provided that $|F_i(z_k) - F_i(x_k)| > \varepsilon$, else set $d^{(i)}_{p_k} = d^{(i)}_{p_{k-1}}$, for $i, k = 1, 2, ..., n$.

Step 4: compute $x_{k+1} = z_k - D_{z_k}F(z_k)$ using (4) and (5), provided that $|F_i(p_k) - F_i(x_k)|$ and $|F_i(z_k) - F_i(x_k)|$ are greater than ε, else set $d^{(i)}_{x_k} = d^{(i)}_{x_{k-1}}$ and $d^{(i)}_{p_k} = d^{(i)}_{p_{k-1}}$, for $i, k = 1, 2, ..., n$ respectively.

Step 5: set k=k+1, and go to step 1.

4. Convergence Result

Definition 4.1 (Linear Convergence)**.** [3] Let $\{x_k\} \subset D$ and $x^* \in D$. Then $x_k \to x^*$ *linearly* if $x_k \to x^*$ and there exist λ in $(0, 1)$ such that

$$\|x_{k+1} - x^*\| \leq \lambda \|x_k - x^*\|.$$

Lemma 4.2. [14] Let $F : \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable in an open convex set $\Omega \in \mathbb{R}^n$. If $\{D_{x_k}\}$ defined by Algorithm 3.1 and $D_0 = I_n$, then $\{D_k\}$ is bounded for each $k > 0$.

Theorem 4.3. Let C be an open convex subset of $\mathbb{R}^n$ and $F : C \to \mathbb{R}^n$ be a continuously differentiable mapping. Assume that there exist $x^* \in C$, with $F(x^*) = 0$ and $F'(x^*) \neq 0$. Then the sequence $\{x_k\}$ generated in (7) defined by

$$p_k = x_k - \frac{1}{2}D_{x_k}F(x_k) \tag{8}$$

$$z_k = x_k - D_{p_k}F(x_k) \tag{9}$$

$$x_{k+1} = z_k - D_{z_k}F(z_k) \tag{10}$$

converges linearly to the solution x^*.

Proof. The Taylor series expansion of $F(x_k)$ about x^* is

$$F(x_k) = F(x^*) + F'(x^*)(x_k - x^*) + O(\|x_k - x^*\|^2). \tag{11}$$

Since $F(x^*) = 0$,

$$\Rightarrow \quad F(x_k) \approx F'(x^*)(x_k - x^*).$$

Subtracting x^* from both side of (8),

$$\begin{aligned} p_k - x^* &= x_k - x^* - \frac{1}{2}D_{x_k}F(x_k) \\ &= x_k - x^* - \frac{1}{2}D_{x_k}F'(x^*)(x_k - x^*) \\ &= \left[I - \frac{1}{2}D_{x_k}F(x^*)\right](x_k - x^*) \end{aligned}$$

where I is an identity matrix. Also subtracting x^* from both side of (9),

$$\begin{aligned} z_k - x^* &= x_k - x^* - D_{p_k}F(x_k) \\ &= x_k - x^* - D_{p_k}F'(x^*)(x_k - x^*) \\ &= \left[I - D_{p_k}F(x^*)\right](x_k - x^*). \end{aligned}$$

Now, the Taylor series expansion of $F(z_k)$ about x^* is given by

$$F(z_k) = F'(x^*)(z_k - x^*) + O(\|z_k - x^*\|^2), \tag{12}$$

$$\begin{aligned} &\Rightarrow F(z_k) \approx F'(x^*)(z_k - x^*) \\ &= F'(x^*)[I - D_{p_k}F(x^*)](x_k - x^*), \end{aligned}$$

since $F(x^*) = 0$, subtracting x^* from both sides of (10), it becomes

$$x_{k+1} - x^* = z_k - x^* - D_{z_k}F(z_k), \tag{13}$$

which is the same as

$$\begin{aligned} x_{k+1} - x^* &= [I - D_{p_k}F'(x^*)](x_k - x^*) - D_{z_k}F'(x^*)[I - D_{p_k}F'(x^*)](x_k - x^*), \\ \Rightarrow x_{k+1} - x^* &= [I - D_{z_k}F'(x^*)][I - D_{p_k}F'(x^*)](x_k - x^*). \end{aligned} \tag{14}$$

Applying norm on both sides of (15), we get

$$\|x_{k+1} - x^*\| \leq \|[I - D_{z_k}F'(x^*)][I - D_{p_k}F'(x^*)]\|\|x_k - x^*\|.$$

Now letting

$$\theta_k = \|[I - D_{z_k}F'(x^*)][I - D_{p_k}F'(x^*)]\|,$$

where $\theta_k \in (0, 1)$, we get

$$\|x_{k+1} - x^*\| \leq \theta_k\|x_k - x^*\|.$$

This shows that the sequence $\{x_k\}$ converges linearly to x^*.

5. Numerical results

In this section, we present numerical results to show the performance of our (AMFA) method, by comparing with the method of J. R. Sharma and P. Gupta (AEM) based on number of iterations and CPU time in seconds. All numerical experiments are done using MATLAB R2010a on Intel(R) Core (TM) i5-3317U CPU with 1.70Ghz speed. The stopping criterion used is:

$$\|F(x_k)\| \leq 10^{-8}.$$

The benchmark test problems are:

Problem 1(Exponential function)

$$f_1(x) = \exp^{x_1} - 1,$$

$$f_i(x) = \frac{i}{10}(\exp^{x_i} + x_i - 1),$$

$$i = 2, ..., n \quad \text{and} \quad x_0 = (\frac{1}{4n^2}, \frac{2}{4n^2}, ..., \frac{n}{4n^2})^T.$$

Problem 2 (Trigonometric function)

$$f_i(x) = 2(n + i(1 - \cos x_i) - \sin x_i - \sum_{j=1}^{n} \cos x_j)(2 \sin x_i - \cos x_i),$$

$$i = 1, 2, ..., n \quad \text{and} \quad x_0 = (\frac{101}{100n}, \frac{101}{100n}, ..., \frac{101}{100n})^T.$$

Problem 3 (Logarithmic function)

$$f_i(x) = \ln(x_i + 1) - \frac{x_i}{n},$$
$$i = 1, 2, ..., n \quad \text{and} \quad x_0 = (1, 1, ..., 1)^T.$$

Problem 4 (n is a multiple of 3)

$$f_{3i-2}(x) = x_{3i-2}x_{3i-1} - x_{3i}^2 - 1,$$
$$f_{3i-1}(x) = x_{3i-2}x_{3i-1}x_{3i} - x_{3i-2}^2 + x_{3i-1}^2 - 2,$$
$$f_{3i}(x) = \exp^{-x_{3i}-2} - \exp^{-x_{3i}-1},$$
$$i = 1, 2, ..., \frac{n}{3} \quad \text{and} \quad x_0 = (0, 0, ..., 0)^T.$$

Problem 5(Linear function-full rank)

$$f_i(x) = x_i - (\frac{2}{n})\sum_{j=1}^{n} x_j + 1,$$
$$i = 1, 2, ..., n \quad \text{and} \quad x_0 = (100, 100, ..., 100)^T.$$

Problem 6(Tridiagonal exponential problem)

$$f_1(x) = x_1 - \exp^{(\cos(h(x_1+x_2)))},$$
$$f_i(x) = x_i - \exp^{(\cos(h(x_{i-1}+x_i+x_{i+1})))} \quad \text{for } i = 2, ..., n-1$$
$$f_n(x) = x_n - \exp^{(\cos(h(x_{n-1}+x_n)))}$$
$$h = \frac{1}{n+1},$$
$$i = 1, 2, ..., n \quad \text{and} \quad x_0 = (1.5, 1.5, ..., 1.5)^T.$$

Problem 7(Trigonometric system)

$$f_i(x) = 5 - (l+1)(1 - \cos x_i) - \sin x_i - \sum_{j=5l+1}^{5l+5} \cos x_j,$$
$$l = div(i-1, 5),$$
$$i = 1, 2, ..., n-1 \quad \text{and} \quad x_0 = (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})^T.$$

We compare the performance among the tested methods based on the performance profile presented by Dolan and *Moré* [5]. For n_s solvers and n_p problems, the performance profile $P : \mathbb{R} \to [0, 1]$ is defined as follows: let $\mathcal{P}$ and $\mathcal{S}$ be the set of problems and set of solvers respectively. For each problem $p \in \mathcal{P}$ and for each solver $s \in \mathcal{S}$ we define $t_{p,s}$:= (number of iterations required to solve problem p by solver s). The performance ratio is given by

$$r_{p,s} := \frac{t_{p,s}}{\min\{t_{p,s} | s \in \mathcal{S}\}}.$$

Then the performance profile is defined by

$$P(\tau) := \frac{1}{n_p}\text{size}\{p \in \mathcal{P}|r_{p,s} \leq \tau\}, \forall \tau \in \mathbb{R},$$

where $P(\tau)$ is the probability for solver $s \in \mathcal{S}$ that a performance ratio $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio.

Table 1: **Numerical results based on dimension of problem (n), Number of iterations (NI) and CPU Time (in seconds) of problem 1-7**

		AMFA		AEM	
Problems	n	NI	CPU time	NI	CPU time
1	100	5	0.003726	-	-
	1000	4	0.007782	-	-
	10000	2	0.042754	-	-
	100000	1	0.101976	-	-
	1000000	1	1.122615	-	-
2	100	10	0.349775	-	-
	1000	13	0.011364	-	-
	10000	16	0.033580	-	-
	100000	32	1.145670	-	-
	1000000	-	-	-	-
3	100	4	0.004006	2	0.103318
	1000	4	0.006951	2	1.838901
	10000	4	0.056571	-	-
	100000	4	0.321032	-	-
	1000000	4	3.659388	-	-
4	100	11	0.004155	-	-
	1000	12	0.016500	-	-
	10000	27	0.040731	-	-
	100000	33	1.400094	-	-
	1000000	-	-	-	-
5	100	1	0.001203	2	0.638548
	1000	1	0.001832	2	21.887876
	10000	1	0.008273	-	-
	100000	1	0.081792	-	-
	1000000	1	0.995562	-	-
6	100	2	0.002225	2	0.131218
	1000	1	0.003855	1	11.088463
	10000	1	0.017051	-	-
	100000	1	0.135032	-	-
	1000000	1	1.756308	-	-
7	100	4	0.004815	3	0.347076
	1000	5	0.01364	3	33.19648
	10000	4	0.079395	-	-
	100000	4	0.563716	-	-
	1000000	4	5.164712	-	-

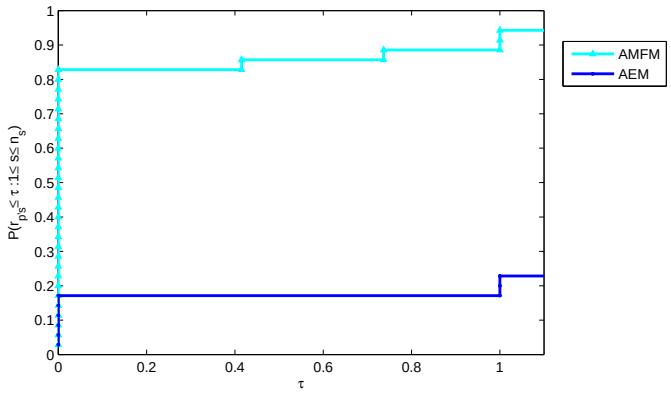

Figure 1: Performance profile by the number of iterations metric for problems 1-7

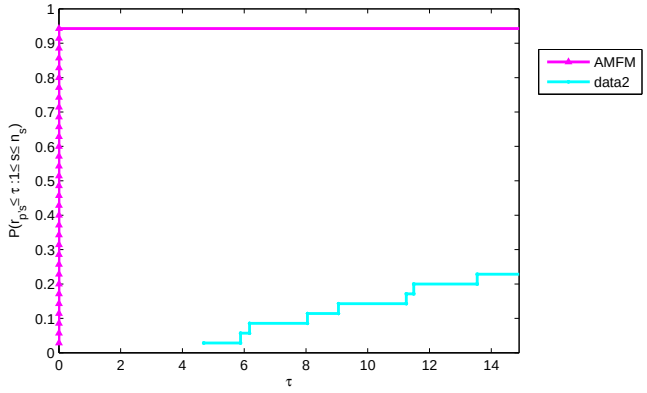

Figure 2: Performance profile by the CPU time metric for problems 1-7

Table 1 clearly demonstrate how the proposed method (AMFA) outperform the (AEM) for all problems tested, as it solves all problems whereas the (AEM) method fails in most problems, especially as the dimension increases. This is achieved due to low computational cost associated with approximating the Jacobian inverse. The method avoids matrix storage as well as solving n linear systems of equation per iteration. However the proposed method is more promising as the dimension increases.

The results obtained in Figure 1-2 shows that the method presented in this research (AMFA) is a good alternative as it has more than 80 percent performance profile against the (AEM) with less than 30 percent.

6. Conclusion

In this paper, we present an enhancement of the method proposed by J.R Sharma and P. Gupta [12] described in section 2 to solve systems of nonlinear equations. We introduce a new algorithm by approximating the Jacobian inverse into diagonal matrix. Among the advantages of the proposed method is that it requires no matrix storage, it also possesses low CPU time to converge due to low computational cost of the scheme. The performance profile also indicate a good significance of the proposed method (AMFA). In view of the above we pronounce the (AMFA) method faster and cheaper than (AEM) method.

7. Acknowledgment

I am very thankful to my colleague Hassan Mohammad for his suggestions and criticisms during the preparation of this paper.

References

[1] Broyden CG, Dennis Jr, and More JJ, *On the local and superlinear convergence of quasi-newton methods*, JIMA **12** (1973), 223–246.
[2] Cordero, Hueso, Martinez, and Terregrosa, *Increasing the convergence order of an iterative method for nonlinear systems*, Appl Math Lett **25** (2007), 2369–2374.
[3] Kelly CT, *Iterative methods for linear and nonlinear equations*, Philadelphia, PA, USA:SIAM, 1995.
[4] _____, *Solving nonlinear systems with newton method*, Philadelphia, PA, USA:SIAM, 2003.
[5] Dolan ED and More JJ, *Benchmarking optimization software with perfomance profiles*, Math Program ser A **91** (2002), 201–213.
[6] Broyden C. G, *A class of methods for solving nonlinear simultaneous equations*, Math Comput **19** (1965), 577–593.
[7] Grau-Sanchez, Grau, and Noguera, *Ostrowski type method for solving systems of nonlinear equations*, Appl Math Comput **218** (2011), 2377–2385.
[8] Mohammad H and Waziri MY, *On broyden-like update via some quadratures for solving nonlinear systems of equations*, Turk J Math **39** (2015), 335–345.
[9] Homeier HHH, *A modified newton method for root finding with cubic convergence*, Comput Appl Math **157** (2003), 227–230.
[10] _____, *A modified newton method with cubic convergence: The multivariable case*, Comput Appl Math **169** (2004), 161–169.
[11] P Jarratt, *Some fourth order multipoint iterative methods for solving equations*, Math Comp **20** (1966), 434–437.
[12] Sharma JR and Gupta P, *An efficient fifth order method for solving systems of nonlinear equations*, Comput Appl Math **67** (2014), 591–601.
[13] Darvishi MT and Barati A, *A fourth-order method from quadrature formulae to solve systems of nonlinear equations*, Appl Math Comput **188** (2007), 257–261.
[14] Waziri MY, Leong WJ, and Monsi M, *A new newton's method with diagonal jacobian approximation for systems of nonlinear equations*, Math Stat **6** (2010), 246–252.
[15] Waziri MY, Leong WJ, and Hassan MA, *Jacobian-free diagonal newton's method for solving nonlinear systems with singular jacobian*, Malays J Math **5** (2011), no. 2, 241–255.