World Applied Sciences Journal 21 (Special Issue of Applied Math): 86-94, 2013
ISSN 1818-4952;
© IDOSI Publications, 2013
DOI: 10.5829/idosi.wasj.2013.21.am.2045

Two-Step Derivative-Free Diagonally Newton's Method for Large-Scale Nonlinear Equations

[1]*M.Y. Waziri*, [2]*W.J. Leong*, [3]*M. Mamat and* [4]*A.U. Moyi*

[1]Department of Mathematical Sciences, Faculty of Science, Bayero University Kano (BUK), Kano, Nigeria
[1,2,4]Department of Mathematics, Faculty of Science,
University Putra Malaysia, 43400, Serdang, Selangor Malaysia
[3]Department of Mathematics, Faculty of Science and Technology,
Malaysia Terengganu University, Kuala Lumpur 21030 Terengganu, Malaysia

Abstract: In this study, we extend the technique of Waziri *et al.* (2010a) via incorporating the two-step scheme in the framework of the diagonal Jacobian updating method to solve large-scale systems of nonlinear equations. In this approach we used points from two previous steps unlike one step approach in most Newton's-like methods. The anticipation has been to improve the current Jacobian approximation into a diagonal matrix. Under mild assumptions local convergence of the proposed method is proved. The results of numerical tests are provided to demonstrate the distinctive qualities of this new approach in contrast with other available variants of Newton's method. The method proposed in this paper has out performs some Newton-like methods in terms of computation cost and storage requirements.

Key words: Large systems . multi-step . jacobian approximation . Newton's-like methods

INTRODUCTION

Let us consider the problem of finding the solution of nonlinear Eq. 1:

$$\begin{gathered} f_1(x_1, x_2, x_3, \ldots\ldots, x_n) = 0 \\ f_1(x_1, x_2, x_3, \ldots\ldots, x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, x_3, \ldots\ldots, x_n) = 0 \end{gathered} \tag{1}$$

The above system can be denoted by $F(x) = 0$, where $F = (f_1, f_2, \ldots, f_n): R^n \rightarrow R^n$ is assumed to satisfy the following assumptions:

A1 = F is continuously differentiable in on open neighborhood $E \subset R^n$
A2 = There exists a solution x^* of (1) in E such that $F(x^*) = 0$
A3 = The Jacobian $F'(x_k)$ is Lipchitz continuous at x^*
A4 = $F'(x^*)$ is nonsingular

The famous method for finding the solution of (1) is Newton's method (Dennis, 1983). Newton's method for solving nonlinear equations is a natural extension of Newton's method for single equation and it is the source of numerous variant methods. This method generates an interactive sequence X_k from any initial point x_0 in the neighborhood of solution x^*, via Eq. 2:

$$x_{k+1} = x_k - (F'(x_k))^{-1} F(x_k) \qquad k = 0, 1, 2\ldots\ldots, \tag{2}$$

Corresponding Author: M.Y. Waziri, Department of Mathematical Sciences, Faculty of Science, Bayero University Kano (BUK), Kano, Nigeria where, $F(x_k)$ is the Jacobian matrix evaluated at x_k.

When the Jacobian $F(x^*)$ is nonsingular at a solution of (1) the convergence is guaranteed and the rate is quadratic from any initial point x_0 in the neighborhood of x* (Dennis, 1983; Jose *et al*., 2009):

$$\|x_{k+1}-x^*\| \leq h \left| x_k - x^* \right|^2 \tag{3}$$

For some h.

There are numerous variants of Newton's method for solving (1) for example, Quasi-Newton method, Fixed Newton's method and Inexact Newton's methods (Dennis, 1983). Quasi-Newton (QN) methods are among the most famous iterative methods for solving (1). The general QN procedure to solve the problem (1) is by calculating the Newton's direction $d_k = -\tilde{B}_k^{-1} F_k$ and set $x_{k+1} = x_k + d_k$ at the K^{th} iteration, where F_k is the function evaluation at x_k and B_k denote Jacobian approximation updated in each iteration. It is noteworthy to mention that, the most critical part of most Newton's-like methods is on forming and storing a full Jacobian matrix(directly or indirectly) in each iteration and the floating points operations is $O(n^3)$. To cope with these well-known weaknesses, some diagonal variants of Newton method via single step approach have been suggested in (Waziri *et al*. 2010a; Leong *et al*., 2011; Waziri *et al*., 2010b; Waziri *et al*., 2010c; Waziri *et al*., 2011). Incorporating this strategy Waziri *et al*. (2010a) showed that their algorithm is significantly cheaper than Newton's method and some of its variants. Apart from these achievements, they utilize a standard one-step two-point approach in Jacobian approximation, which is commonly used by Newton's-like methods. In disparity, this study develops a new diagonal-type Newton's method for solving large-scale systems of nonlinear equations by broadening the scheme of (Waziri *et al*. 2010a) whereas employs a two-step multi-point approach to increase the accuracy of Jacobian approximation into diagonal matrix. We organized the rest of this study as follows: In section 2, we present the proposed method, Section 3 presents Convergence analysis and Numerical results are reported in Section 4 and lastly Conclusion in Section 5.

MATERIALS AND METHOD

Two-step dagonal jacobian (2-DNM): In this section, we shall present our new variant of Newton's method via two-step multi-point scheme. This new scheme generates a sequence of points $\{X_k\}$via Eq. 4:

$$x_{k+1} = x_k - M_k^{-1} F(x_k) \tag{4}$$

where, M_k is a diagonal approximation of the Jacobian matrix. We need to constructa matrix M_k using diagonal updating approach which is a good approximation of the Jacobian. waziri *et al*. (2010) uses data from one preceding step to improve the current approximate Jacobian into diagonal matrix i.e., $y_k = F(x_{k+1})-F(x_k)$ and $s_k = x_{k+1}-x_k$, this scheme is known as one-step approach. Therefore, in order to make M_k to be a more accurate approximation of the Jacobian matrix, we make use of an interpolating curve in the variable-space to develop an incomplete Taylor series expansion of F at x_k. This is accomplished by considering some of the most successful of two-step methods (Ford and Moghrabi, 1997; Ford and Moghrabi, 1994; Ford and Thrmlikit, 2003) for more details). By using this two-step approach we can present the improved incomplete Taylor series expansion of F(x) as follows Eq. 5:

$$M_k(s_k - \lambda k s_{k-1}) \approx (y_k - \lambda_k y_{k-1}) \tag{5}$$

By letting

$$\varpi_k = s_k - \lambda k s_{k-1} \text{ and } \vartheta_k = y_k - \lambda_k y_{k-1},$$

then it follows from (5) that Eq. 6:

$$M_k \varpi_k \approx \vartheta_k \tag{6}$$

As we used information from the last two steps instead of one previous step in (5) and (6) we anticipate that the novel incomplete Taylor series expansion of F(x) derived from this scheme, will also improve the precision of the Jacobian approximation. To achieve this we require to build an interpolating quadratic curves $x(\in)$ and $y(\in)$, whereby $x(\in)$ interpolates the last two earlier iterates x_{k-1}, x_k and x_{k+1} and $y(\in)$ interpolates the last two earlier function evaluation F_{k-1}, F_k and F_{k+1} (which are assumed to be available). By using the approach introduced by (Ford and Moghrabi, 1997), we can obtain the λ_k in (5) using the value of $\in_0, \in_1$ and $\in_2$. We assumed that $\in_1 = 0$ and $\{\in_j\}_j^2 = 0$ then λ_k is given as follows Eq. 7-9:

$$\begin{aligned} -\in_1 &= \in_2 - \in_1 \\ &= \left\| x(\in_2) - x(\in_1) \right\| M_k \\ &= \left\| x_{k+1} - x_k \right\| M_k \\ &= \left\| s_k \right\| M_k \\ &== \left(s_k^T M_k s_k \right)^{\frac{1}{2}} \end{aligned} \tag{7}$$

And:

$$\begin{aligned} -\in_0 &= \in_2 - \in_0 \\ &= \left\| x(\in_2) - x(\in_0) \right\| M_k \\ &= \left\| x_{k+1} - x_{k-1} \right\| M_k \\ &= \left\| s_k + s_{k-1} \right\| M_k \\ &= \left(\left(s_k + s_{k-1} \right)^T M_k \left(s_k + s_{k-1} \right) \right)^{\frac{1}{2}} \end{aligned} \tag{8}$$

Let us define ρ as:

$$\rho = \frac{\in_2 - \in_0}{\in_1 - \in_0} \tag{9}$$

Then ϖk and ϑk are given as:

$$\varpi_k = s_k - \frac{\rho^2}{1+2\rho} s_{k-1} \tag{10}$$

$$\vartheta_k = y_k - \frac{\rho^2}{1+2\rho} y_{k-1} \tag{11}$$

Using (10) and (11) in (6) and the fact that $M_k \approx F'(x_k)$ then we consider the following:

$$m_{k+1}^{(i)} = \frac{\vartheta_k^{(i)}}{\varpi_k^{(i)}} \tag{12}$$

Therefore:

$$M_{k+1} = \mathrm{diag}\,(m_{k+1}^{(i)}) \tag{13}$$

For i = 1, 2,…., n and k = 0, 1, 2,…, n.

where, $\vartheta_k^{(i)}$ is the ith component of the vector ϑ_k, $\varpi_k^{(i)}$ is the ith component of the vector ϖk and $m_{k+1}^{(i)}$ is the ith diagonal element of a diagonal matrix M_{k+1} respectively.

To safeguard against the possibilities of small denominator $\varpi_k^{(i)}$.We utilize (13) only when $\left|\varpi_k^{(i)}\right| > 10^{-4}$ for i = 1, 2,…, if not set $m_k^{(i)} = m_{k-1}^{(i)}$.

The updating formula for this novel diagonal variant of Newton's method is given as Eq. 14:

$$x_{k+1} = x_k - M_k^{-1} F(x_k) \tag{14}$$

Finally we present the first result of this study as follows:

Algorithm 2-DNM:

```
Step 1: Choose an initial guess $x_0$ and $M_0 = I$, let k:= 0
Step 2: Compute $F(x_k)$. If $||F(x_k)|| \le 10^{-4}$ stop
Step 3: If k:= 0 define $x_1 = x_0 - M_0^{-1}F(x_0)$. Else if k:= 1 set $\varpi_k = s_k$ and $\vartheta_k = y_k$ and goto 5
Step 4: If $k \ge 2$ compute $\in_1, \in_0$ and $\rho$ via (7)-(9), respectively and find $\varpi_k$ and $\vartheta_k$ using (10) and (11), respectively.
        If $\varpi_k^T \vartheta_k \le 10^{-4} \|\varpi_k\|_2 \|\vartheta_k\|_2$ set $\varpi_k = s_k$ and $\vartheta_k = y_k$
Step 5: Let $x_{k+1} = x_k - M_k^{-1}F(x_k)$ and update $M_{k+1}$ as dene by (14)
Step 6: Check if $||\varpi_k||_2 \ge \in_1$ where $\in_1 = 10^{-4}$, if yes retain $M_{k+1}$ that is computed by step 5. Else set, $M_{k+1} = M_k$
Step 7: Set k:=k+1 and goto 2
```

CONVERGENCE RESULTS

In this section, we discuss the condition under which the diagonal updating formula derived in this study is linearly convergence to x*. To analyze the convergence of the method, we will make the following assumptions on nonlinear systems F.

Assumption 3.1:

- F is differentiable in an open convex set E in $\Re^n$
- There exists $x^* \in E$ such that $F(x^*) = 0$, $F'(x)$ is continuous for all x.
- F'(x) satisfies Lipschitz condition of order one i.e., there exists a positive constant μ such that Eq. 15:

$$\|F'(x)-F'(y)\| \le \mu \|x - y\| \tag{15}$$

For all $x, y \in \Re^n$.
There exists constants $c_1 \le c_2$ such that $c_1||\omega||^2 \le \omega^T F'(x)\omega \le c_2||\omega||^2$ for all $x \in E$ and $\omega \in \Re^n$.
Then we have the following result:

Theorem: Let F satisfies Assumption 3.1 and $\|\varpi_k\| \neq 0$ for some finite k and there exist positive constants θ and δ such that M_k is bounded. Then the sequence$\{x_k\}$ $k \ge 0$ generated by Algorithm 2-DNM converges to x* linearly.

Proof: Note that, it is diffculty to guarantee that (14) is always bounded. Due to the fact that,the scheme was derived from component-wise approximation approach. Nevertheless,we consider the situation where the updating matrix is bounded hence, the Taylor series of F(x) about (x_k) is given as Eq. 16:

$$F(x) = F(x_k) + F'(x_k)(x-x_k) + O(\|x-x_k\|^2) \tag{16}$$

When x = x*, (16) becomes:

$$F(x^*) = F(x_k) + F'(x_k)(x^*-x_k) + O(\|x^*-x_k\|^2) \tag{17}$$

But F(x*) = 0, then we have:

$$-F(x_k) = F(x_k) = F'(x_k)(x^*-x_k) + O(\|x^*-x_k\|^2) \tag{18}$$

It follows that:

$$x_{k+1} - x^* = x_k - x^* - M_k^{-1} F(x_k) \tag{19}$$

Substituting (18) into (19), yields to:

$$x_{k+1} - x^* = x_k - x^* + M_k^{-1}\left[F(x_k) \left(x^* - x_k\right) + O\left(\left\|x^* - x_k\right)\right\|^2 \right] \tag{20}$$

It follows from (20) that:

$$x_{k+1} - x^* = x_k - x^* - M_k^{-1} F'(x_k) \left(x_k - x^*\right) + O\left(\left\|x_k - x^*\right)\right\|^2) \tag{21}$$

Neglecting the term with highest order, (21) turns into:

$$x_{k+1} - x^* = x_k - x^* \left[A - M_k^{-1} F'\left(x_k\right)\right] \tag{22}$$

where, A is identity matrix.
Taking the norm of both sides of (22), arrive at:

$$\|x_{k+1} - x^*\|x^*\| \leq \|A - M_k^{-1} F'(x_k)\|\|x_k - x^*\| \tag{23}$$

Suppose that M_k is bounded and the fact that Jacobian is bounded where $\delta = \max[\gamma_1, \gamma_2]$,
we have:

$$\|x_{k+1} - x^*\| \leq \left(\sqrt{n} - \alpha\delta\right)\|x_k - x^*\| \tag{24}$$

Then:

$$\|x_{k+1} - x^*\| \leq \theta\|xk - x^*\| \tag{25}$$

For some θ where $\theta = \sqrt{n} - \alpha\delta$. Therefore the sequence $\{x_k\}k \geq 0$ generated by Algorithm 2-DNM converges to x* linearly.

NUMERICAL RESULTS

This section illustrates some numerical outcomes on the test functions problems on our algorithm. Each problem is tested with dimensions ranging from 25-250,000. Weimplemented the methods (CN,FN,BM,2-DNM) using MATLAB 7:0. All the computational experiments were carried out in double precision computer, the termination state is considered to be Eq. 26:

$$\|s_k\| + \|F(x_k)\| \leq 10^{-4} \tag{26}$$

Based on comparison indices presented in (Bogle and Perkins, 1990), we reported on robustness, efficiency and combined robustness and efficiency of CN,FN,BM and 2-DNM methods respectively. If r_{ij} is the CPU time required to solve the problem i by the method j, $r_{ib} = \min_k r_{ij}$, i.e., the best result for problem i by any of tested methods, t_j the number of successes by method j and n_j the number of problems attempted by method j; then the robustness index is:

$$Rj = \frac{t_j}{n_j}$$

The efficiency index is:

$$Ej = \sum_{i=1;rij\neq 0}^{m} \left(\frac{r_{ib}}{r_{ij}}\right) / t_j$$

And the combined robustness and efficiency index is:

$$E_j \times R_j = \sum_{i=1;rij\neq 0}^{m} \left(\frac{r_{ib}}{r_{ij}}\right) / n_j$$

where, R is the percentage of cases in which each method found a solution. For E and E × R the best possible result is 1 and closer to one value of indices indicate a better result (Natasa and Zorna, 2001).
We also force the routine to terminate whenever:

- The number of iteration is at least 500 but no point of x_k satisfies (26) is obtained
- CPU time in second reaches 500
- Insufficient memory to initial the run
  Represents a failure due to any of (i)-(iii)

We then illustrate some details of the used test problems as follows:

Problem 1: Trigonometric System of (Byeong *et al*., 2010):

$$f_i(x) = \cos(x_i)-1$$

$i = 1, 2,\ldots, n$and $x_0 = (0.5, 0.5,\ldots, 0.5)\times\frac{10\pi}{18} = (0.87, 0.87,\ldots, 0.87)$.

Problem 2: System of n nonlinear equations: (Waziri *et al*., 2012)

$$f_i(x) = \ln(x_i)\cos((1-(1+(x^T x)^2)^{-1}))\exp((1-(1+(x^T x)^2)^{-1}))$$
$$i = 1, 2,\ldots, n \text{ and } x_0 = (2.5, 2.5,\ldots, 2.5)$$

Problem 3: Trigonometric system:

$$f_1(x) = \cos x_1 - 9 + 3x_1 + 8\exp x_2$$
$$f_i(x) = \cos x_i - 9 + 3x_i + 8\exp x_{i-1}$$
$$i = 2,\ldots, n-1 \text{ and } (5, 5,\ldots, 5)$$

Problem 4: System of n nonlinear equations:

$$f_i(x) = (1-x_i^2) + x_i(1+x_i x_{n-2} x_{n-1} x_n)-2$$
$$i = 1, 2,\ldots, n \text{ and } x_0 = (2, 2,\ldots, 2)$$

Problem 5: System of n nonlinear equations: (Waziri *et al*., 2012)

$$f_i(x) = n(x_i-3)^2 + \frac{\cos(x_i-3)}{2} - \frac{x_i-2}{\exp(x_i-3)+\log(x_i^2+1)}$$
$$i = 1, 2,\ldots, n \text{ and } x_0 = (-3, -3-3,\ldots, -3)$$

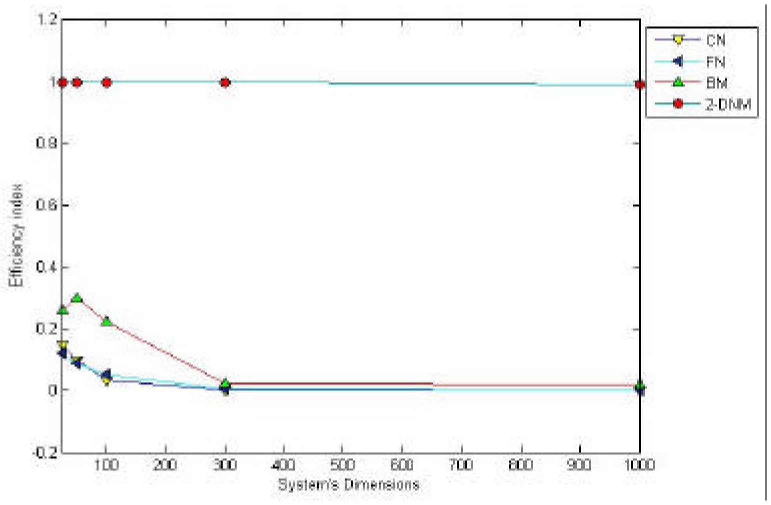

Fig. 1: Efficiency profile of CN, FN, BM and 2-DNM methods as the dimensions increases (in term of CPU time)

Table 1: Performance profile of CN, FN BM and 2-DNM methods for n = 25 (in terms of CPU time in seconds)

	CN	FN	BM	2-DNM
R	0.7778	0.3333	0.7777	1.0000
E	0.1219	0.0819	0.3554	1.0000
E×R	0.0948	0.0273	0.2764	1.0000

Table 2: Numerical results of CN, FN, BM and 2-DNM methods

		CN		FN		BM		2-DNM	
Prob	Dim	NI	CPU	NI	CPU	NI	CPU	NI	CPU
1	25	7	0.062	236	0.047	12	0.005	25	0.001
2	25	5	0.062	-	-	7	0.016	6	0.015
3	25	-	-	-	-	-	-	14	0.031
4	25	6	0.031	57	0.031	10	0.031	14	0.004
5	25	13	0.190	-	-	-	-	23	0.014
1	100	7	0.872	356	0.168	12	0.018	27	0.019
2	100	6	0.593	-	-	7	0.312	6	0.031
3	100	-	-	-	-	-	-	14	0.036
4	100	6	0.593	60	0.172	10	0.125	14	0.015
5	100	14	1.232	-	-	-	-	27	0.030
1	300	7	2.082	444	7.098	13	0.033	27	0.018
2	300	6	4.243	-	-	7	0.593	6	0.031
3	300	-	-	-	-	-	-	14	0.037
4	300	6	4.212	-	-	10	0.905	14	0.031
5	300	15	11.263	-	-	-	-	30	0.062
1	1000	7	101.471	-	-	14	7.167	31	0.040
2	1000	6	107.874	-	-	7	8.736	6	0.031
3	1000	-	-	-	-	-	-	14	0.094
4	1000	6	108.340	-	-	10	14.398	14	0.031
5	1000	17	280.830	-	-	-	-	33	0.140
1	250000	-	-	-	-	-	-	38	4.892
2	250000	-	-	-	-	-	-	7	1.435
3	250000	-	-	-	-	-	-	14	2.293
4	250000	-	-	-	-	-	-	15	2.761
5	250000	-	-	-	-	-	-	45	11.623

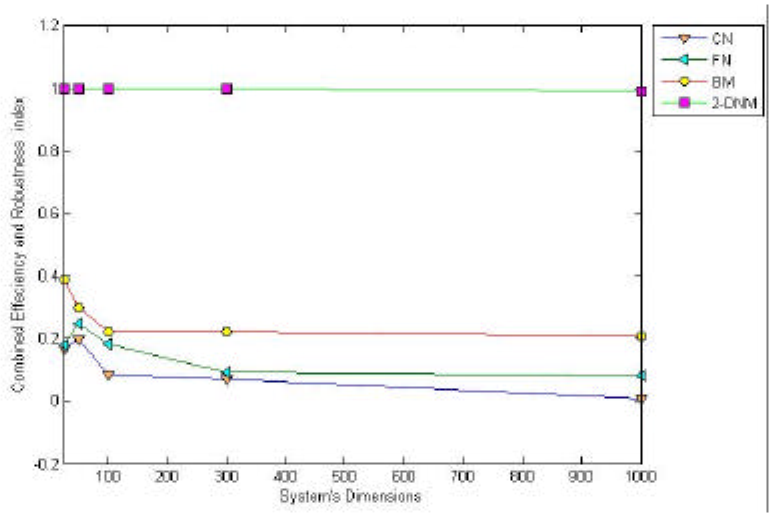

Fig. 2: Combined efficiency and robustness profile of CN, FN, BM and 2DNM methods as the dimensions increases (in term of CPU time)

DISCUSSION

We summarize our numerical experiments in Table 1 and 2, by using number of iteration and CPU time required to solve the tested problems. As exp ected, the results imply that diagonal updating proposed in the study improves notably over the performance of CN, FN and BM methods, as having greatest efficiency, robustness and combined efficiency and robustness indices. This is not surprising that any method that required storage of full elements of Jacobean matrix, particularly when solving large-scale systems,CN, FN and BM methods are necessary inferior.

In addition, the method 2-DNM solves for n = 250,000 using a negligible number of storage locations as appose by CN, FN and BM methods, the fact that for n>3000 not all algorithms converge due to the "out-of-memory" circumstances. Based on this, we claim that our methods have outperforms the Newton’s method, especially for handling large-scale systems. Moreover, we used Fig. 1 and 2 to detail the growth of the CN, FN, BM and 2-DNM method’s CPU time as the dimension increases, which shows that 2-DNM increases linearly while CN, FN, BM growths exponentially.

Finally according to these numerical results, we also remark that, the use of two-stepapproach to approximating the Jacobian into diagonal matrix is very effective and encouraging as it has the least CPU time, floating point operations and storage requiremints respectively.

CONCLUSION

A novel approach on approximating the Jacobain matrix into diagonal form has been presented. The method uses two-previous points to build up the updating scheme, unlike the conventional single point method. We applied the new scheme to solve large scale system of nonlinear equations.

Numerical testing provides strong indication that the 2-DNM methods exhibits enhanced performance in all the tested problems (as measured by the CPU time, foating points operations and matrix storage requirement) by comparison with the other varaints of Newton’s methods, an attribute which becomes more obvious as the dimension of the problems increases.

Finally we claim that, the use of multi-points approach to approximate Newton’s step is capable of improving the efficiency and robustness of Newton’s method to an acceptable level, especially when the function derivatives are relatively expensive or the Jacobain is reasonably nearly singular or large-scale systems.

REFERENCES

1. Jose, L.H., M. Eulalia and R.M. Juan, 2009. Modified Newton’s method for systemsof nonlinear equations with singular Jacobian. Compt. Applied Math., 224: 77-83.
2. Dennis, J.E., 1983. Numerical methods for unconstrained optimization and nonlinearequations, Prince-Hall, Inc., Englewood Clis, New Jersey.
3. Binh, L., 1978. On the convergence of a quasi-newtons method for sparse nonlinear systems Math. Comp., 32: 447-451.
4. Dembo, R.S., S.C. Eisenstat and T. Steihaug, 1982. InexactNewton method. SIAM J. Numer. Ana., 19: 400-408.
5. Natasa, K. and L. Zorna, 2001. Newton-like method with modification of the right-hand vector. J. Math. Compt., 71: 237-250.
6. Spedicator, E., 1975. Computational experience with quas-Newton algorithms for minimization problems of moderately large size Rep. CISE-N-175, Segrate (Milano).
7. Waziri, M.Y., W.J. Leong, M.A. Hassan and M. Monsi, 2010a. A New Newton method withdiagonal Jacobian approximation for systems of Non-Linear equations. J. Math. Stati. Sci. Publication, 6: 246-252.
8. Ford, J.A. and L.A. Moghrabi, 1997. Alternating multi-step quasi-Newton methods for unconstrained optimization. J. Comput. Applied Math., 82: 105-116.
9. Ford, J.A. and L.A. Moghrabi, 1994. Multi-step quasi-Newton methods for optimization. J. Comput. Applied Math., 50: 305-323.
10. Ford, J.A. and S. Thrnlikit, 2003. New implicite updates in multi-step quasi-Newton methodsfor unconstrained optimization. J. Comput. Applied Math., 152: 133-146.

11. Luksan, E.S.C. and T. Steihaug, 1982. Inexact trustregion method for large sparse systems of nonlinear equations. Jota, 81: 569-590.
12. Waziri, M.Y., W.J. Leong, M.A. Hassan and M. Monsi, 2010b. Jacobian computation-free New-ton method for systems of Non-Linear equations. J. Numerical Math. Stochastic, 2: 154-163.
13. Hao, L. and N. Qin, 2008. Incomplete Jacobian Newton method for nonlinear equation. Comp. Math. Appli., 56: 218-227.
14. Byeong, C.S., M.T. Darvishi and H.K. Chang, 2010. A comparison of the New-tonKrylov method with high order Newton-like methods to solve nonlinearsystems. Applied Math. Comput., 217: 3190-3198.
15. Leong, W.J., M.A. Hassan and M.Y. Waziri, 2011. A matrix-free quasi-Newtonmethod for solving large-scale nonlinear systems. Comput. Math. Applied, 625: 2354-2363.
16. Bogle, I. and J.D. Perkins, 1990. A new sparsity preservingquasi-Newton updatefor solving nonlinear equations. SIAM J. Sci. Statist., 11: 621-630.
17. Waziri, M.Y., W.J. Leong, M.A. Hassan and M. Monsi, 2010c. An efficient solver for systems of Non-Linear equations with singular Jacobian Via diagonal updating. Applied Mathematical Sciences, 4 (69): 3403-3412.
18. Waziri, M.Y., W.J. Leong and M.A. Hassan, 2011. Jacobian-free diagonal Newton's method for solving nonlinear systems with singular Jacobian. Malaysian Journal of Mathematical Sciences, 5 (2): 241-255.
19. Waziri, M.Y., W.J. Leong and M.A. Hassan, 2012, Diagonal Broyden-like Method for Large-scale Systems of Nonlinear Equations. Malaysian Journal of Mathematical Sciences, 6 (1): 59-73.