Open Journal of Science and Technology (OJST)
ISSN:
www.readersinsight.net/ojst

A Derivative-Free Decent Method via Acceleration Parameter for Solving Systems of Nonlinear Equations

A.S. Halilu[1*], M.K. Dauda[2], M.Y. Waziri[3], M. Mamat[4]

[1]*Department of Mathematics and Computer Science, Sule Lamido University, Jigawa, Nigeria.*
[2]*Department of Mathematical Sciences, Bayero University Kano, Nigeria.*
[3]*Department of Mathematical Sciences, Kaduna State University, Nigeria.*
[4]*Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Gong Badak, Terengganu, Malaysia*

* *Corresponding author: abubakarsani.halilu@jsu.edu.ng*

Abstract

An algorithm for solving large-scale systems of nonlinear equations based on the transformation of the Newton method with the line search into a derivative-free descent method is introduced. Main idea used in the algorithm construction is to approximate the Jacobian by an appropriate diagonal matrix. Furthermore, the step length is calculated using inexact line search procedure. Under appropriate conditions, the proposed method is proved to be globally convergent under mild conditions. The numerical results presented show the efficiency of the proposed method.

Keywords: Acceleration parameter, decent direction, derivative free, Global Convergent

ARTICLE INFORMATION

Received: 15-Jul-2019
Revised: 02-Aug-2019
Accepted: 29-Aug-2019

DOI:
Web Address:

INTRODUCTION

Consider the systems of nonlinear equations:

$$F(x) = 0, \qquad (1)$$

where $F: R^n \to R^n$ is nonlinear map.

Among various methods for solving nonlinear equations (1), Newton's method is quite welcome due to its nice properties such as the rapid convergence rate, the decreasing of the function value sequence [10]. However, at each iteration, Newton's method needs the computation of the derivative F^o as well as the solution of some system of linear equations.

The iterative formula of a Newton method is given by

$$x_{k+1} = x_k + s_k, \quad s_k = \alpha_k d_k, k = 0,1,\cdots,$$

where, α_k is a step length to be computed by a line search technique [2, 3], x_{k+1} represents a new iterative point, x_k is the previous iteration, while d_k is the search direction to be calculated by solving the following linear system of equations,

$$F^o(x_k)d_k = -F(x_k), \qquad (2)$$

Where $F^o(x_k)$ is the Jacobian matrix of $F(x_k)$ at x_k. A basic requirement of the line search is to sufficiently decrease the function values i.e

$$\|F(x_k + \alpha_k d_k)\| \le \|F(x_k)\|. \qquad (3)$$

In fact, this problem can come from an unconstrained optimization problem, a saddle point, and equality constrained problem [6]. Let f be a norm function defined by

$$f(x) = \frac{1}{2}\|F(x)\|^2 \qquad (4)$$

where, $\|\cdot\|$ to stand for the Euclidian norm. Then the nonlinear equations problem (1) is equivalent to the following global optimization problem

$$min f(x), x \in R^n.$$

and condition (3) is equivalent to

$$f(x_k + \alpha_k d_k) \le f(x_k) \qquad (5)$$

Furthermore, the search direction d_k is generally required to satisfy the descent condition

$$\nabla f(x_k)^T d_k < 0.$$

The derivative-free direction can be obtained in several ways [4,5,7,9,12]. An iterative method that generates a sequence $\{x_k\}$ satisfying (3) or (5) is called a norm descent method. If d_k is a descent direction of f at x_k, then inequality (5) holds for all $\alpha_k > 0$ sufficiently small. In particular, Newton method with line search is norm descent. For a quasi-Newton method, however, d_k may not be a descent direction of f at x_k even if B_k is symmetric and positive definite. To globalize a quasi-Newton method, Li and Fukushima [6] proposed an approximately norm descent line search technique and established global and super-linear convergence of a Gauss-Newton based BFGS method for solving symmetric nonlinear equations. The method in [6] is not norm descent. In addition, the global convergence theorem is established under the assumption that $F^o(x_k)$ is uniformly nonsingular.

The drawback of the technique (2) is the need to compute the Jacobian matrix $F^o(x_k)$ at every iteration, which will increase the computing difficulty, due to the first-order derivative of the system because sometimes they are not even available or could not be obtained exactly [10] especially for the large-scale problems. Therefore, motivated by [1] the purpose of this article is to develop a derivative-free method with decent direction for solving system of nonlinear equations via $F^o(x_k) \approx \gamma_k I$,

Where I is an identity matrix. The presented method has a norm descent property without computing the Jacobian matrix with less number of iterations and CPU time that is globally convergent.

The next section of this paper present the proposed method. In section 3, the convergence analysis of the proposed method is given. In Sections 4 and 5, report on some numerical results and a conclusion were given respectively.

Derivation of the Method

The main idea used in the proposed algorithm construction is approximation of the Jacobian in (2) by a diagonal matrix via acceleration parameter. Now, from Taylor's expansion of the first order the approximation of $F(x_{k+1})$ can be brought as follows:

$F(x_{k+1}) \approx F(x_k) + F^o(\delta)(x_{k+1} - x_k)$ (6)

where the parameter δ fulfills the conditions $\delta \in [x_k, x_{k+1}]$,

$\delta = x_k + \lambda(x_{k+1} - x_k) \qquad 0 \leq \lambda \leq 1.$ (7)

Having in mind that the distance between x_k, and x_{k+1} is small enough.

By taking $\lambda = 1$ in (7) and get $\delta = x_{k+1}$. Therefore we have

$F^o(\delta) \approx \gamma_{k+1} I.$ (8)

Knowing this, the expression (6) becomes

$F(x_{k+1}) - F(x_k) = \gamma_{k+1}(x_{k+1} - x_k).$ (9)

From (9) we have the standard secant condition

$\gamma_{k+1} s_k = y_k,$ (10)

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

In [6], Li and Fukushima used the term

$g_k = \frac{F(x_k + \alpha_k F(x_k)) - F(x_k)}{\alpha_k}$ (11)

to approximate the gradient $\nabla f(x_k)$, which avoids computing exact gradient and α_k updated via line search method. It is clear that when $\|F(x_k)\|$ is small, then $g_k \approx \nabla f(x_k)$.

pre-multiplying both side of (10) by g_k^T, the relation allows us to compute the parameter γ_{k+1} in the following way:

$\gamma_{k+1} = \frac{y_k^T[F(x_k + \alpha_k F(x_k)) - F(x_k)]}{s_k^T[F(x_k + \alpha_k F(x_k)) - F(x_k)]}$ (12)

we can easily show that, our direction is

$d_k = -\gamma_{k-1} F(x_k),$ (13)

we finally have the general scheme as:

$x_{k+1} = x_k + \alpha_k d_k.$ (14)

we can easily show that, our direction is

Therefore, the derivative-free line search used in [1,3,6] is the best choice to compute the step length α_k.

Let $\omega_1 > 0, \omega_2 > 0$ and $q, r \in (0,1)$ be constants and let $\{\eta_k\}$ be a given positive sequence such that

$\sum_{k=0}^{\infty} \eta_k < \eta < \infty$ (15)

$f(x_k + \alpha_k d_k) - f(x_k) \leq -\omega_1 \|\alpha_k F(x_k)\|^2 - \omega_2 \|\alpha_k d_k\|^2 + \eta_k f(x_k).$ (16)

Let i_k be the smallest non negative integer i such that (16) holds for $\alpha = r^i$. Let $\alpha_k = r^i k$.

Algorithm 1(ADDA).

Step 1: Given $x_0, \gamma_0 = 1, \epsilon = 10^{-4}$, set $k = 0$.

Step 2: Compute $F(x_k)$ and test a stopping criterion. If yes, then stop; otherwise continue with Step 3.

Step 3: Compute search direction $d_k = \gamma_k^{-1} F(x_k)$.

Step 4: Compute step the length α_k (using (16).

Step 5: Set $x_{k+1} = x_k + \alpha_k d_k$.

Step 6: Compute $F(x_{k+1})$.

Step 7: determine, $\gamma_{k+1} = \frac{y_k^T[F(x_k + \alpha_k F(x_k)) - F(x_k)]}{s_k^T[F(x_k + \alpha_k F(x_k)) - F(x_k)]}$

Step 8: Consider $k = k + 1$, and go to Step 3.

Remark 1:

It is clear that the line search (16) is well defined. Otherwise, for any integer $i > 0$,

$f(x_k + r^i d_k) - f(x_k) > -\omega_1 k r^i F(x_k) k^2 - \omega_2 k r^i d_k k^2 + \eta_k f(x_k)$

Let $i \to \infty$, then $0 \geq \eta_k f(x_k)$. This leads to a contradiction since $\eta_k f(x_k)$ is positive.

Convergence Analysis

In this section, we present the global convergence of our method (ADDA). To begin with, let us defined the level set $\Omega = \{x | kF(x)k \leq kF(x_0)k\}$. (17)

In order to analyze the convergence of algorithm 1 we need the following assumption:

Assumption 1.

(1) There exists $x^* \in R^n$ such that $F(x^*) = 0$. (2) F is continuously differentiable in some neighborhood say N of x^* containing Ω. (3) The Jacobian of F is bounded and positive definite on N. i.e there exists a positive constants $M > m > 0$ such that $\|F'(x)\| \leq M \forall x \in N$, (18)

and

$m\|d\|^2 \leq d^T F'(x) d \quad \forall x \in N, d \in R^n.$ (19)

From the level set we have:

$\|F(x)\| \leq m_1 \quad \forall x \in \Omega.$ (20)

Remark 2:

Assumption 1 implies that there exists a constants $M > m > 0$ such that

$m\|d\| \leq \|F'(x)d\| \leq M\|d\| \qquad \forall x \in N, d \in R^n.$ (21)

$m\|x - y\| \leq \|F(x) - F(y)\| \leq M\|x - y\| \qquad \forall x, y \in N.$ (22)

In particular $\forall x \in N$ we have

$m\|x - x^*\| \leq \|F(x)\| = \|F(x) - F(x^*)\| M\|x - x^*\|,$ (23)

where x^* stands for the unique solution of (1) in N.

Since $\gamma_k I$ approximates $F^0(x_k)$ along direction s_k, we can contemplate another assumption

Assumption 2.

$\gamma_k I$ is a good approximation to $F^0(x_k)$, i.e

$\|(F'(x_k) - \gamma_k I)d_k\| \leq \epsilon \|F(x_k)\|$ (24)

Where $\epsilon \in (0,1)$ is a small quantity [8].

Lemma 1. Let assumption 2 holds and $\{x_k\}$ be generated by algo rithm 1. Then d_k is a descent direction for $f(x_k)$ at x_k i.e

$\nabla f(x_k)^T d_k < 0.$ (25)

Proof. from (13), we have

$\nabla f(x_k)^T d_k < F(x_k)^T F'(x_k) d_k$

$= F(x_k)^T [(F'(x_k) - \gamma_k I) d_k - \mathrm{F}(x_k)]$ (26)

$= F(x_k)^T \big((F'(x_k) - \gamma_k I) d_k\big) - \|F(x_k)\|^2,$

by chauchy schwatz we have,

$\nabla f(x_k)^T d_k \leq \|F(x_k)\| \|(F'(x_k) - \gamma_k I) d_k\| - \|F(x_k)\|^2$ (27)

$\leq -(1 - \epsilon)\|F(x_k)\|^2.$

Hence for $\epsilon \in (0,1)$ this lemma is true.

By lemma 1, we can deduce that the norm function $f(x_k)$ is a descent along d_k, which means that $\|F(x_{k+1})\| \leq \|F(x_k)\|$ is true.

Lemma 2. Let assumption 2 hold and $\{x_k\}$ be generated by algorithm 1. Then $\{x_k\} \subset \Omega$.

Proof. By lemma 1 we have $\|F(x_{k+1})\| \leq \|F(x_k)\|$. Moreover, we have for all k.

$\|F(x_{k+1})\| \leq \|F(x_k)\| \leq \|F(x_{k-1})\| \leq \cdots \leq \|F(x_0)\|$

This implies that $\{x_k\} \subset \Omega$.

Lemma 3. Suppose that assumption 1 holds $\{x_k\}$ is generated by algorithm 1. Then there exists a constant $m > 0$ such that for all k.

$s_k^T\big[F\big(x_k + x_k F(x_k)\big) - F(x_k)\big] \geq m\|s_k\|^2.$ (28)

Proof. By mean-value theorem and (19) we have,

$s_k^T\big[F\big(x_k + x_k F(x_k)\big) - F(x_k)\big] = s_k^T F'(\xi) s_k \geq m\|s_k\|^2.$

where $\xi = x_k + \zeta(x_{k+1} - x_k)$, $\xi \in (0,1)$. The proof is complete.

Lemma 4. Suppose that assumption 1 holds and $\{x_k\}$ is generated by algorithm 1. Then we have

$$\lim_{k\to\infty} \|\alpha_k d_k\| = 0, \tag{29}$$

and

$$\lim_{k\to\infty} \|\alpha_k F(x_k)\| = 0, \tag{30}$$

Proof. By (16) we have for all $k > 0$.

$$\omega_2 \|\alpha_k d_k\|^2 \leq \omega_1 \|\alpha_k F(x_k)\|^2 + \omega_2 \|\alpha_k d_k\|^2$$
$$\leq \|F(x_k)\|^2 - \|F(x_{k+1})\|^2 + \eta_k \|F(x_k)\|^2 \tag{31}$$

by summing the above inequality, we have

$$\omega_2 \sum_{i=0}^{k} \|\alpha_i d_i\|^2 \leq \sum_{i=0}^{k} (\|F(x_i)\|^2 - \|F(x_{i+1})\|^2) + \sum_{i=0}^{k} \eta_i \|F(x_i)\|^2$$
$$= \|F(x_0)\|^2 - \|F(x_{k+1})\|^2 + \sum_{i=0}^{k} \eta_i \|F(x_i)\|^2$$
$$\leq \|F(x_0)\|^2 + m_1^2 \sum_{i=0}^{k} \eta_i$$
$$\leq \|F(x_0)\|^2 + m_1^2 \sum_{i=0}^{\infty} \eta_i \tag{32}$$

so from the level set and fact that $\{\eta_k\}$ satisfies (15) then the series $\sum_{i=0}^{\infty} \|\alpha_i d_i\|^2$ is convergent. This implies (29). By similar way we can prove that (30) holds.

Lemma 5. Suppose that assumption 1 holds and $\{x_k\}$ is generated by algorithm 1. Then there exists some positive constants m_2 such that for all $k > 0$,

$$\|x_k\| \leq m_2 \tag{33}$$

proof. from (22), we have

$$\|d_k\| = \left\|-\frac{F(x_k) y_{k-1}^T s_{k-1}}{\|y_{k-1}\|^2}\right\|$$
$$\leq \frac{\|F(x_k)\| \|s_{k-1}\| \|y_{k-1}\|}{m^2 \|s_{k-1}\|^2}$$
$$\leq \frac{\|F(x_k)\| M \|s_{k-1}\|}{m^2 \|s_{k-1}\|}$$
$$\leq \frac{\|F(x_k)\| M}{m^2}$$
$$\leq \frac{\|F(x_0)\| M}{m^2} \tag{34}$$

Taking $m_2 \leq \frac{\|F(x_0)\| M}{m^2}$, we have (33).

The proof is completed.

We can deduce that for all k (33) hold.

Now we are going to establish the following global convergence theorem to show that under some suitable conditions, there exist an accumulation point of $\{x_k\}$ which is a solution of problem (1).

Theorem 1. Suppose that assumption 1 holds,$\{x_k\}$ is generated by algorithm 1. Assume further for all $k > 0$,

$$\alpha_k \geq c \frac{|F(x_k)^T d_k|}{\|d_k\|^2}, \tag{35}$$

where c is some positive constant. Then

$$\lim_{k\to\infty} \|F(x_k)\| \tag{36}$$

Proof. From lemma 5 we have (33). Therefore by (29) and the boundedness of $\{\|d_k\|\}$, we have

$$\lim_{k\to\infty} \alpha_k \|d_k\| = 0, \tag{37}$$

from (35) and (37) we have

$$\lim_{k\to\infty} |F(x_k) d_k| = 0 \tag{38}$$

on the other hand from (13) we have,

$$F(x_k)^T d_k = -\gamma_k^{-1} \|F(x_k)\|^2 \tag{39}$$
$$\|F(x_k)\|^2 = \|-F(x_k)^T d_k \gamma_k\|$$
$$\leq |F(x_k)^T d_k| |\gamma_k| \tag{40}$$

but

$$|\gamma_k^{-1}| < \frac{\|y_{k-1}\| \|g_{k-1}\|}{\|s_{k-1}\| \|g_{k-1}\|} = \frac{M \|s_{k-1}\|}{\|s_{k-1}\|} = M$$

so from (40) we have,

$$\|F(x_k)\|^2 \leq |F(x_k)^T d_k| M \tag{41}$$

Thus

$$0 < \|F(x_k)\|^2 < |F(x_k)^T d_k| M \to 0 \tag{42}$$

Therefore

$$\lim_{k\to\infty} \|F(x_k)\| = 0 \tag{43}$$

The proof is completed.

Numerical Results

In this section, we compared the performance of our method with an improved derivative free method via double direction approach for solving systems of nonlinear equations (IDFDD) [5]. For the both algorithms the following parameters are set $\omega_1 = \omega_2 = 10^{-4}$, $r = 0.2$ and $\eta_k = \frac{1}{(k+1)^2}$.

The employed computational codes was written in Matlab 7.9.0 (R2009b) and run on a personal computer 2.00 GHz CPU processor and 3 GB RAM memory. We stopped the iteration if the total number of iterations exceeds 1000 or $\|F(x_k)\| \leq 10^{-4}$. We claim that the method fails, and use the symbol " − " to represents failure due to; (i) Memory requirement (ii) Number of iterations exceed 1000. (iii) If $\|F(x_k)\|$ is not a number. The methods were tested on some Benchmark test problems with different initial points. problem 1 and 3 below are from [5] while problem 2 is from [11].

Problem 1.

$$F_i(x) = x_i - 0.1 x_{i+1}^2,$$
$$F_n(x) = x_n - x_1^2$$
$$i = 1,2,\cdots,n-1.$$

Problem 2.

The discretized Chandrasehars H-equation:

$$F_i(x) = x_i - \left(1 - \frac{c}{2n} \sum_{j=1}^{n} \frac{\mu_i x_i}{\mu_i + \mu_j}\right)^{-1}$$
$$i = 1,2,\cdots,n. \quad j = 1,2,\cdots,n$$

with $c \in [0,1)$ and $\mu = \frac{i-0.5}{n}$. (In our experiment we take $c = 0.1$).

Problem 3.

$$\begin{pmatrix} 2 & -1 & & & \\ 0 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix} x + (sinx_1 - 1, sinx_2 - 1, \cdots, sinx_n - 1,)$$

Table 1: Problem 1

		ADDA			IDFDD		
Dimension	Guess	NI	Time	$\|F(x_k)\|$	NI	Time	$\|F(x_k)\|$
100	x_1	6	0.001547	3.17E-06	60	0.021327	9.49E-05
	x_2	10	0.002261	7.42E-04	258	0.058595	9.56E-05
	x_3	20	0.003787	2.93E-04	-	-	-
	x_4	20	0.003768	2.63E-04	-	-	-
1000	x_1	6	0.001551	3.20E-05	68	7.63E-05	1.34E-02
	x_2	15	0.002948	5.51E-04	-	-	-
	x_3	30	0.005519	2.16E-05	-	-	-
	x_4	30	0.005392	2.13E-05	-	-	-
10000	x_1	3	0.001087	1.47E-04	-	-	-
	x_2	20	0.003915	2.61E-04	-	-	-
	x_3	39	0.006656	4.62E-04	-	-	-
	x_4	39	0.006846	4.61E-04	-	-	-

Table 2: problem 2

		ADDA			IDFDD		
Dimension	Guess	NI	Time	$\|F(x_k)\|$	NI	Time	$\|F(x_k)\|$
1100	x_1	2	0.001117	3.89E-04	343	0.089121	9.87E-05
	x_2	3	0.001281	8.42E-05	446	0.087016	9.87E-05
	x_3	3	0.001233	1.80E-05	422	0.085348	9.9520e-05
	x_4	3	0.001244	1.80E-05	436	0.080774	9.76E-05
1000	x_1	3	0.001248	1.53E-07	360	0.094551	9.83E-05
	x_2	3	0.001285	2.00E-05	481	0.128377	9.79E-05

	x_3	3	0.001231	1.78E-05	452	0.094691	9.83E-05
	x_4	3	0.001281	1.78E-05	445	0.107394	9.82E-05
10000	x_1	3	0.001548	4.81E-07	372	0.088074	9.78E-05
	x_2	3	0.001373	1.80E-05	441	0.097173	9.79E-05
	x_3	3	0.001293	1.78E-05	521	0.106895	9.96E-05
	x_4	3	0.001291	1.78E-05	521	0.113734	9.97E-05

Table 3: problem 3

		ADDA			IDFDD		
Dimension	Guess	NI	Time	$\|F(x_k)\|$	NI	Time	$\|F(x_k)\|$
100	x_1	5	0.026795	4.57E-05	30	0.161017	7.25E-05
	x_2	4	0.021828	2.42E-04	38	0.205967	9.13E-05
	x_3	5	0.025738	5.91E-05	53	0.352845	7.32E-05
	x_4	5	0.025583	1.53E-04	53	0.293711	6.93E-05
1000	x_1	5	0.026157	2.56E-06	31	0.202076	8.90E-05
	x_2	4	0.021917	5.68E-05	45	0.236591	9.58E-05
	x_3	4	0.022024	3.71E-04	67	0.380298	7.21E-05
	x_4	4	0.020806	2.08E-02	67	0.371045	7.17E-05
10000	x_1	5	0.027645	9.76E-05	31	0.165042	9.48E-05
	x_2	4	0.022119	1.59E-04	53	0.300573	6.89E-05
	x_3	5	0.024003	2.70E-05	81	0.565639	7.41E-05
	x_4	5	0.026698	9.52E-06	81	0.648324	7.41E-05

The numerical results of the two(2) methods are reported in Tables 1,2 and 3, where ”NI” and ”Time” stand for the total number of all iterations and the CPU time in seconds respectively, while $F(x_k)$ is the norm of the residual at the stopping point. We also set

$$x_1 = (1,\frac{1}{2},\frac{1}{3},\cdots,\frac{1}{n})^T,$$
$$x_2 = (1-1,1-\frac{1}{2},1-\frac{1}{3},\cdots,1-\frac{1}{n})^T,$$
$$x_3 = (2,4,6,\cdots,2n)^T,$$
$$x_4 = (1,3,5,\cdots,2n-1)^T$$

From Tables 1, 2 and 3 we can easily observe that both of these methods attempt to solve the systems of nonlinear equations (1), but the better efficiency and effectiveness of our proposed algorithm was clear for it solves where IDFDD fails. This is quite evident for instance with problem 1. In particular, the ADDA method considerably outperforms the IDFDD for almost all the tested problems, as it has the least number of iterations and CPU time, which are even much less than the CPU for the IDFDD method.

Conclusion

In this paper, a derivative-free decent method via acceleration parameter for solving systems of nonlinear equations is given. It is a fully derivative-free iterative method which possesses global convergence under some appropriate conditions. Numerical comparisons using a set of large-scale test problems show that the proposed method is practically quite effective.

References

1. Predrag S. Stanimirovic and Marko B. Miladinovic, Accelerated gradient descent methods with line search, Numerical Algorithms, 54 (2010), 503520.
2. L. Zang, W. Zhou and D.H. Li, Global convergence of modified Fletcher-Reeves conjugate gradient method with Armijo-type line search, Numerische mathematik, 164(1), (2005), 277-289.
3. A.S. Halilu, and M.Y. Waziri, A transformed double steplength method for for solving large-scale systems of nonlinear equations, Journal of Numerical Mathematics and Stochastics, 9(1) (2017), 20-23.
4. H. Abdullahi, M.Y. Waziri, and M.K. Dauda, ”A Double Direction Conjugate Gradient Method for Solving Large-Scale System of Nonlinear Equation”, Journal of Mathematical and Computational Science, 7(3) (2017), 606-624.
5. A.S. Halilu, and M. Waziri, An improved derivative-free method via double direction approach for solving systems of nonlinear equations, Journal of Ramanujan Mathematical Society, 33(1) (2018), 7589.
6. D. Li and M. Fukushima, A global and superlinear convergent Gauss-Newton based BFGS method for symmetric nonlinear equation, SIAM Journal on numerical Analysis, 37 (2000), 152-172.
7. K. Natasa and L. Zorna, Newton-Like method with modification of rigtht-hand vector, Mathematics of Computation, 71 (2001), 237-250.
8. Gongli Yuan∗, Xiwen Lu, A new backtracking inexact BFGS method for symmetric nonlinear equations”, Journal of computers and mathematics with applications, 55, (2008), 116-129.
9. M.K. Dauda, M. Mamat, M.Y. Waziri, F. Ahmad and F.S. Mohamad, Inexact CG-Method via SR1 update for solving systems of nonlinear equations, Far East Journal of Mathematical Sciences, 100(11) (2016), 787-1804.
10. J.E. Dennis, Jr and R.B. Schnabel, Numerical method for unconstrained optimization and non-linear equations, practice Hall, Englewood cliffs, NJ, USA, 1983.
11. Jamilu Sabiu, Efficient hybrid conjugate gradient method for solving symmetric nonlinear equations, SIAM Palestine Journal of Mathematics, 6 (2017), 157-164.
12. M. Mamat, M. K. Dauda, M. A. bin Mohamed, M. Y. Waziri and F. S. Mohamad, H. Abdullah. Derivative free Davidon-Fletcher-Powell (DFP) for solving symmetric systems of nonlinear equations, IOP Conf. Series: Materials Science and Engineering 332 (2018), doi:10.1088/1757-899X/332/1/012030