Journal of Numerical Mathematics and Stochastics, **6** (1) : 73-83, 2014 © ***JNM@S***
http://www.jnmas.org/jnmas6-6.pdf Euclidean Press, LLC
Online: ISSN 2151-2302

Three-Step Derivative-Free Diagonal Updating Method for Solving Large-Scale Systems of Nonlinear Equations

L. Y. UBA, and M. Y. WAZIRI

Department of Mathematical Sciences, Faculty of Sciences, Bayero University, Kano State, Nigeria,
E-mail: mywaziri.mth@buk.edu.ng

Abstract. *In this paper, we construct a three-step derivative-free diagonal updating method for solving large-scale systems of nonlinear equations in order to avoid the computation of the Jacobian matrix which requires first order derivatives. A numerical experiment is reported to show that the proposed method is quite encouraging.*

Key words : Three-Step Derivative-Free Method, Nonlinear Systems of Equations, Diagonal Update.

AMS Subject Classifications : 60H10, 60H05

1. Introduction

The general form of nonlinear systems of equations is

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

where $F = (f_1, f_2, \ldots, f_n)^T$ is a mapping from $\mathbb{R}^n$ to $\mathbb{R}^n$ and $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n$, $n \in \mathbb{N}$, and the mapping F is assumed to have the following *standard assumptions* [4, 5]:
1. The mapping F is continuously differentiable on a convex open set $D \subset \mathbb{R}^n$.
2. There exists a solution x^* of (1) such that $F(\mathbf{x}^*) = 0$.
3. The Jacobian matrix $F'(\mathbf{x})$ is Lipschitz continuous at $\mathbf{x}^*$. That is, there exists a positive constant K such that $\|F(\mathbf{x}) - F(\mathbf{x}^*)\| \leq K\|\mathbf{x} - \mathbf{x}^*\|$, for all $\mathbf{x}, \mathbf{x}^* \in D$.
4. $F'(\mathbf{x}^*)$ is nonsingular.
The Newton's method is the well known method for solving (1). It invokes an iterative scheme which generates an iterative sequence $\{\mathbf{x}_k\}$:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - F'(\mathbf{x}_k)^{-1}F(\mathbf{x}_k), \quad k = 0, 1, 2, \ldots, n, \tag{2}$$

where $F'(\mathbf{x}_k)$ is the Jacobian matrix.

Although the Newton's method has a quadratic convergence and despite it's reliability and simplicity to implement, it has some major shortcomings [4, 9, 10], one of which is the requirement to compute the Jacobian matrix at each iteration. Several efforts have been made

to overcome the shortcomings of Newton's method. Among them is the *Broyden's method* which is a quasi-Newton's method [1]. It approximates the Newton's direction by using an approximation of the Jacobian $F'(\mathbf{x}_k) \approx B_k$, which is updated as the nonlinear iteration progresses.

Algorithm 1.1 (Broyden's method (BM)). Given $\mathrm{x}_0 \in \mathbb{R}^n$, tolerance ε, Set $B_0 = I_n$ and $k = 0$,
Step 1: Compute $F(\mathbf{x}_k)$, if $\|F(\mathbf{x}_k)\| \leq \varepsilon$ stop, else go to step 2.
Step 2: Solve $B_k\, s_k = -F(\mathbf{x}_k)$ for s_k.
Step 3: Update $\mathbf{x}_{k+1} = \mathbf{x}_k + s_k$.
Step 4: Compute $\mathrm{y}_k = F(\mathbf{x}_{k+1}) - F(\mathbf{x}_k)$.
Step 5: Compute $B_{k+1} = B_k + \dfrac{(\mathrm{y}_k - B_k\, s_k)\, s_k^T}{s_k^T s_k}$
Step 6: set $\mathrm{k} = k + 1$, and go to step 1.

The convergence order of Broyden's method is proven to be superlinear [2]. The main idea behind this method is to reduce the computational cost of the Jacobian in every iteration by approximating the Jacobian matrix with a derivative-free matrix which can be updated in each iteration.

Newton's Method with Diagonal Jacobian Approximation is also a modification of Newton's Method [9, 10] with $F'(\mathbf{x}_k) \approx D_{k+1} = diag(d_{k+1}^{(i)})$, $d_{k+1}^{(i)} = \dfrac{F_i(\mathrm{x}_{k+1}) - F_i(\mathrm{x}_k)}{\mathrm{x}_{k+1}^{(i)} - \mathrm{x}_k^{(i)}}$,
$i = 1, 2, \cdots, n$ and $k = 0, 1, 2, \cdots, n$.

Algorithm 1.2 (Newton's Method with Diagonal Jacobian Approximation (NDJA)).
Considering F as a function from $\mathbb{R}^n$ to $\mathbb{R}^n$ with the same properties as (1):
Step 1: Given x_0 , ε and $D_0 = I_0$, set $k = 0$.
Step 2: Compute $F(\mathbf{x}_k)$.
Step 3: Compute $\mathbf{x}_{k+1} = \mathbf{x}_k - D_{k+1}^{-1} F(\mathbf{x}_k)$, where $D_{k+1} = diag(d_{k+1}^{(i)})$, $d_{k+1}^{(i)} = \dfrac{F_i(\mathrm{x}_{k+1}) - F_i(\mathrm{x}_k)}{\mathrm{x}_{k+1}^{(i)} - \mathrm{x}_k^{(i)}}$,
$i, k = 1, 2, \cdots, n$, provided $|\mathbf{x}_{k+1}^{(i)} - \mathbf{x}_k^{(i)}| > \varepsilon$, else set $d_{k+1}^{(i)} = d_k^{(i)}$ for $k = 0, 1, 2, \cdots, n$.
Step 4: If $\|\mathbf{x}_{k+1} - \mathbf{x}_k\| + \|F(\mathbf{x}_k)\| \leq \varepsilon$ stop, else set $k = k + 1$ and go to step 2.

The advantage of the diagonal updating method is that, it requires only the diagonal entries and storage of n-entries in every iteration. This paper presents an iterative derivative-free three-step diagonal updating method for solving systems of nonlinear equations . The rest of this paper is arranged as follows. In sections 2 and 3 we describe our new method and analyze its convergence. The numerical results are presented in section 4 and a conclusion is given in section 5, which is followed by references.

3. Three-Step Derivative-Free Diagonal Updating Method

Consider the iterative scheme

$$\begin{cases} \mathbf{y}_k &= \mathbf{x}_k - F'(\mathbf{x}_k)^{-1}F(\mathbf{x}_k), \quad k \in \mathbb{N} \\ \mathbf{z}_k &= \mathbf{y}_k - \eta F'(\mathbf{y}_k)^{-1}F(\mathbf{y}_k), \quad k \in \mathbb{N} \\ \mathbf{x}_{k+1} &= \mathbf{z}_k - \zeta F'(\mathbf{z}_k)^{-1}F(\mathbf{z}_k), \quad k \in \mathbb{N} \end{cases}, \tag{3}$$

where

$$\begin{cases} \mathbf{w}_k = \mathbf{x}_k + \beta F(\mathbf{x}_k), \quad k \in \mathbb{N}, \quad \beta \in \mathbb{R}^+, \\ \eta = 1 - \dfrac{\|F(\mathrm{y}_k)\|}{\|F(\mathrm{w}_k)\|}, \quad k \in \mathbb{N}, \\ \zeta = \left(1 - \dfrac{\|F(\mathrm{y}_k)\|}{\|F(\mathrm{w}_k)\|}\right)\left(\dfrac{2\|F(\mathrm{y}_k)\|^3}{\|F(\mathrm{w}_k)\|^2\|F(\mathrm{x}_k)\|}\right), \quad k \in \mathbb{N}. \end{cases} \tag{4}$$

To complete construction of the proposed method, we introduce a suitable approximation of the Jacobian matrix using diagonal updating to reduce the number of function evaluations. So we approximate the Jacobian matrices $F'(\mathbf{x}_k)$, $F'(\mathbf{y}_k)$ and $F'(\mathbf{z}_k)$ in equation (3), by certain diagonal matrices D_{x_k}, D_{y_k} and D_{z_k} respectively, which are updated at each iteration. Then

$$\mathbf{w}_k = \mathbf{x}_k + \beta F(\mathbf{x}_k), \quad k \in \mathbb{N}, \quad \beta \in \mathbb{R}^+, \tag{5}$$

$$F'(\mathbf{x}_k) \approx D_{\mathrm{x}_k} = diag(d^{(i)}_{\mathrm{x}_k}), \ d^{(i)}_{\mathrm{x}_k} = \frac{F_i(\mathrm{w}_k) - F_i(\mathrm{x}_k)}{\mathrm{w}^{(i)}_k - \mathrm{x}^{(i)}_k} \quad , i = 1, 2, \ldots, \tag{6}$$

$$F'(\mathbf{y}_k) \approx D_{\mathrm{y}_k} = diag(d^{(i)}_{\mathrm{y}_k}), \quad d^{(i)}_{\mathrm{y}_k} = \frac{F_i(\mathrm{x}_k) - F_i(\mathrm{y}_k)}{\mathrm{x}^{(i)}_k - \mathrm{y}^{(i)}_k} \quad , i = 1, 2, \ldots, \tag{7}$$

$$F'(\mathbf{z}_k) \approx D_{\mathrm{z}_k} = D_{\hat{\mathrm{z}}_k} - D_{\mathrm{y}_k} + D_{\check{\mathrm{z}}_k} = diag(d_{\hat{\mathrm{z}}_k}) - diag(d_{\mathrm{y}_k}) + diag(d_{\check{\mathrm{z}}_k}), \tag{8}$$

where $d^{(i)}_{\hat{\mathrm{z}}_k} = \dfrac{F_i(\mathrm{y}_k) - F_i(\mathrm{z}_k)}{\mathrm{y}^{(i)}_k - \mathrm{z}^{(i)}_k}$, $d^{(i)}_{\check{\mathrm{z}}_k} = \dfrac{F_i(\mathrm{x}_k) - F_i(\mathrm{z}_k)}{\mathrm{x}^{(i)}_k - \mathrm{z}^{(i)}_k}$, $F_i(\mathbf{w}_k)$ is the i^{th} component of the vector $F(\mathbf{w}_k)$, $F_i(\mathbf{x}_k)$ is the i^{th} component of the vector $F(\mathbf{x}_k)$, $F_i(\mathbf{y}_k)$ is the i^{th} component of the vector $F(\mathbf{y}_k)$, $F_i(\mathbf{z}_k)$ is the i^{th} component of the vector $F(\mathbf{z}_k)$, $\mathrm{w}^{(i)}_k$ is the i^{th} component of the vector $\mathbf{w}_k$, $\mathrm{x}^{(i)}_k$ is the i^{th} component of the vector $\mathbf{x}_k$, $\mathrm{y}^{(i)}_k$ is the i^{th} component of the vector $\mathbf{y}_k$, $\mathrm{z}^{(i)}_k$ is the i^{th} component of the vector $\mathbf{z}_k$ and provided that the denominators in (6)-(8) are not equal to zero [11]. Substituting (6)-(8) into (3), we get our new scheme

$$\begin{cases} \mathbf{y}_k &= \mathbf{x}_k - D_{\mathrm{x}_k}{}^{-1}F(\mathbf{x}_k), \quad k \in \mathbb{N} \\ \mathbf{z}_k &= \mathbf{y}_k - \eta D_{\mathrm{y}_k}{}^{-1}F(\mathbf{y}_k), \quad k \in \mathbb{N} \\ \mathbf{x}_{k+1} &= \mathbf{z}_k - \zeta D_{\mathrm{z}_k}{}^{-1}F(\mathbf{z}_k), \quad k \in \mathbb{N} \end{cases}. \tag{9}$$

Algorithm 2.1 (Three-step derivative-free diagonal updating method (TSDU)). Given an initial guess $x_0 \in \mathbb{R}^n$, tolerance ε and $\beta \in \mathbb{R}^+$. set $k = 0$.
Step 1: compute $F(\mathbf{x}_k)$, If $\|F(\mathbf{x}_k)\| \leq \varepsilon$ stop. Else go to step 2.
Step 2: Compute $\mathbf{w}_k = \mathbf{x}_k + \beta F(\mathbf{x}_k)$.
Step 3: compute $\mathbf{y}_k = \mathbf{x}_k - D_{x_k}^{-1} F(\mathbf{x}_k)$, where D_{x_k} is define by (6), provided that $|w_k^{(i)} - x_k^{(i)}| > \varepsilon$, else set $d_{x_k}^{(i)} = d_{x_{k-1}}^{(i)}$, for $k = 1, 2, \ldots, n$.
Step 4: compute $\mathbf{z}_k = \mathbf{y}_k - \eta D_{y_k}^{-1} F(\mathbf{y}_k)$, where D_{y_k} is define by (7), provided that $|x_k^{(i)} - y_k^{(i)}| > \varepsilon$, else set $d_{y_k}^{(i)} = d_{y_{k-1}}^{(i)}$, for $k = 1, 2, \ldots, n$.
Step 5: compute $\mathbf{x}_{k+1} = \mathbf{z}_k - \zeta D_{z_k}^{-1} F(\mathbf{z}_k)$, where D_{z_k} is define by (8), provided that $|y_k^{(i)} - z_k^{(i)}|$, $|x_k^{(i)} - y_k^{(i)}|$ and $|x_k^{(i)} - z_k^{(i)}|$ are greater than ε, else set $d_{\hat{z}_k}^{(i)} = d_{\hat{z}_{k-1}}^{(i)}$, $d_{y_k}^{(i)} = d_{y_{k-1}}^{(i)}$ and $d_{\check{z}_k}^{(i)} = d_{\check{z}_{k-1}}^{(i)}$ for $k = 1, 2, \ldots, n$.
Step 6: set k=k+1, and go to step 1.

3. Convergence Analysis

In this section, we consider reporting on a result for F which presents the condition under which the TSDU algorithm converges linearly to $\mathbf{x}^*$.

Lemma [10] **3.1**. *Let $F(\mathbf{x}) : R^n \to R^n$ be continuously differentiable in an open convex set $D \subset R^n$. If D_k defined by $D_{k+1} = diag(d_{k+1}^{(i)})$, where* $d_{k+1}^{(i)} = \dfrac{F_i(x_{k+1}) - F_i(x_k)}{x_{k+1}^{(i)} - x_k^{(i)}}$ *and* $D_0 = I_n$, $i = 1, 2, \ldots, n$ *then D_k is bounded for each $k > 0$.*

Theorem 3.1. *Let $\mathbf{x}^* \in D$ be a solution of sufficiently differentiable function $F(\mathbf{x}) : D \subset R^n \to R^n$ in a convex open interval D. Assume $\{\mathbf{x}_k\}$ converges to $\mathbf{x}^*$ and initial guess vector $\mathbf{x}_0$ is sufficiently close to $\mathbf{x}^*$, then the convergence rate of the proposed method defined by (6) is linear.*

Proof. Let $\mathbf{x}^*$ be a root of $F(\mathbf{x})$, i.e. $F(\mathbf{x}^*) = 0$ and $F'(\mathbf{x}^*) \neq 0$. The Taylor series expansion of $F(\mathbf{x}_k)$ about $\mathbf{x}^*$ is

$$F(\mathbf{x}_k) = F(\mathbf{x}^*) + F'(\mathbf{x}^*)(\mathbf{x}_k - \mathbf{x}^*) + O(\|\mathbf{x}_k - \mathbf{x}^*\|^2). \tag{10}$$

Since $F(\mathbf{x}^*) = 0$, then we have

$$F(\mathbf{x}_k) \approx F'(\mathbf{x}^*)(\mathbf{x}_k - \mathbf{x}^*). \tag{11}$$

Subtraction of $\mathbf{x}^*$ and substitution of (11) in the first step of (9) gives

$$\mathbf{y}_k - \mathbf{x}^* = \mathbf{x}_k - \mathbf{x}^* - D_{x_k}^{-1} F'(\mathbf{x}^*)(\mathbf{x}_k - \mathbf{x}^*), \tag{12}$$

which is the same as

$$\mathbf{y}_k - \mathbf{x}^* = \left[E - D_{\mathrm{x}_k}{}^{-1}F'(\mathbf{x}^*)\right](\mathbf{x}_k - \mathbf{x}^*), \tag{13}$$

where E is the identity matrix. The Taylor series expansion of $F(\mathrm{w}_k)$ about $\mathbf{x}^*$ is

$$F(\mathbf{w}_k) = F(\mathbf{x}^*) + F'(\mathbf{x}^*)(\mathbf{w}_k - \mathbf{x}^*) + O(\|\mathbf{w}_k - \mathbf{x}^*\|^2). \tag{14}$$

Substitution of (5) and (11) in (14) leads to

$$F(\mathbf{w}_k) \approx F(\mathbf{x}^*) + F'(\mathbf{x}^*)[\mathbf{x}_k - \mathbf{x}^* + \beta F'(\mathbf{x}^*)(\mathbf{x}_k - \mathbf{x}^*)]. \tag{15}$$

Since $F(\mathbf{x}^*) = 0$, we have

$$F(\mathbf{w}_k) \approx F'(\mathbf{x}^*)[E + \beta F'(\mathbf{x}^*)](\mathbf{x}_k - \mathbf{x}^*). \tag{16}$$

The Taylor series expansion of $F(\mathbf{y}_k)$ about $\mathbf{x}^*$ is

$$F(\mathbf{y}_k) = F(\mathbf{x}^*) + F'(\mathbf{x}^*)(\mathbf{y}_k - \mathbf{x}^*) + O(\|\mathbf{y}_k - \mathbf{x}^*\|^2). \tag{17}$$

Since $F(\mathbf{x}^*) = 0$, then we have

$$F(\mathbf{y}_k) = F'(\mathbf{x}^*)(\mathbf{y}_k - \mathbf{x}^*) + O(\|\mathbf{y}_k - \mathbf{x}^*\|^2). \tag{18}$$

It follows from (18) that

$$F(\mathbf{y}_k) = F'(\mathbf{x}^*)(\mathbf{y}_k - \mathbf{x}^*). \tag{19}$$

Then substitute (13) in (19) to get

$$F(\mathbf{y}_k) \approx F'(\mathbf{x}^*)\left[E - D_{\mathrm{x}_k}{}^{-1}F'(\mathbf{x}^*)\right](\mathbf{x}_k - \mathbf{x}^*). \tag{20}$$

From equation (4), we have

$$\eta = 1 - \frac{\|F(\mathrm{y}_k)\|}{\|F(\mathrm{w}_k)\|}, \quad \textit{where} \quad F(\mathbf{w}_k) \neq 0 \quad \textit{and} \quad k \in \mathbb{N}. \tag{21}$$

Substitute (16) and (20) in (21) to obtain

$$\eta \leq 1 - \frac{\|E - D_{\mathrm{x}_k}^{-1}F'(\mathbf{x}^*)\|}{\|E + \beta F'(\mathbf{x}^*)\|}. \tag{22}$$

Subtract $\mathbf{x}^*$ from both sides of the second step of (9) to get

$$\mathbf{z}_k - \mathbf{x}^* = \mathbf{y}_k - \mathbf{x}^* - \eta\, D_{\mathrm{y}_k}{}^{-1}F(\mathbf{y}_k). \tag{23}$$

Then substitution of (13) and (20) in (23) leads to

$$\begin{aligned}\mathbf{z}_k - \mathbf{x}^* = {} & \left[E - D_{\mathrm{x}_k}{}^{-1}F'(\mathbf{x}^*)\right](\mathbf{x}_k - \mathbf{x}^*) \\ & - \eta\, D_{\mathrm{y}_k}{}^{-1}F'(\mathbf{x}^*)\left[E - D_{\mathrm{x}_k}{}^{-1}F'(\mathbf{x}^*)\right](\mathbf{x}_k - \mathbf{x}^*),\end{aligned} \tag{24}$$

which is the same as

$$\mathbf{z}_k - \mathbf{x}^* = \left[E - \eta\, D_{\mathrm{y}_k}{}^{-1}F'(\mathbf{x}^*)\right]\left[E - D_{\mathrm{x}_k}{}^{-1}F'(\mathbf{x}^*)\right](\mathbf{x}_k - \mathbf{x}^*). \tag{25}$$

Consider now the Taylor series expansion of $F(\mathbf{z}_k)$ about $\mathbf{x}^*$

$$F(\mathbf{z}_k) = F(\mathbf{x}^*) + F'(\mathbf{x}^*)(\mathbf{z}_k - \mathbf{x}^*) + O(\|\mathbf{z}_k - \mathbf{x}^*\|^2). \tag{26}$$

Since $F(\mathbf{x}^*) = 0$, then we have

$$F(\mathbf{z}_k) = F'(\mathbf{x}^*)(\mathbf{z}_k - \mathbf{x}^*) + O(\|\mathbf{z}_k - \mathbf{x}^*\|^2), \tag{27}$$

and it follows from (27) that

$$F(\mathbf{z}_k) \approx F'(\mathbf{x}^*)(\mathbf{z}_k - \mathbf{x}^*). \tag{28}$$

Substitute further (25) in (28)to arrive at

$$F(\mathbf{z}_k) \approx F'(\mathbf{x}^*)\Big[E - \eta\, D_{\mathrm{y}_k}{}^{-1}F'(\mathbf{x}^*)\Big]\Big[E - D_{\mathrm{x}_k}{}^{-1}F'(\mathbf{x}^*)\Big](\mathbf{x}_k - \mathbf{x}^*). \tag{29}$$

Now,

$$1 - \frac{\|F(\mathrm{z}_k)\|}{\|F(\mathrm{w}_k)\|} \leq 1 - \frac{\|[E - \eta\, D_{\mathrm{y}_k}^{-1}F'(\mathbf{x}^*)][E - D_{\mathrm{x}_k}^{-1}F'(\mathbf{x}^*)]\|}{\|E + \beta F'(\mathbf{x}^*)\|}, \tag{30}$$

$$\|F(\mathbf{w}_k)\|^2\ \|F(\mathbf{x}_k)\| \leq (\|F'(\mathbf{x}^*)\|)^3\|E + \beta F'(\mathbf{x}^*)\|^2\ (\|\mathbf{x}_k - \mathbf{x}^*\|)^3, \tag{31}$$

$$2\|F(\mathbf{y}_k)\|^3 \leq 2(\|F'(\mathbf{x}^*)\|)^3\|E - D_{\mathrm{x}_k}{}^{-1}F'(\mathbf{x}^*)\|^3(\|\mathbf{x}_k - \mathbf{x}^*\|)^3, \tag{32}$$

$$\frac{2\|F(\mathrm{y}_k)\|^3}{\|F(\mathrm{w}_k)\|^2\|F(\mathrm{x}_k)\|} \leq 1 - \frac{2\|E - D_{\mathrm{x}_k}^{-1}F'(\mathbf{x}^*)\|^3}{\|E + \beta F'(\mathbf{x}^*)\|^2}, \tag{33}$$

where $F(\mathbf{x}_k) \neq 0$, $F(\mathbf{w}_k) \neq 0$ and $k \in \mathbb{N}$. But,

$$\begin{aligned}\zeta &= \left(1 - \frac{\|F(\mathrm{z}_k)\|}{\|F(\mathrm{w}_k)\|}\right)\left(\frac{2\|F(\mathrm{y}_k)\|^3}{\|F(\mathrm{w}_k)\|^2\|F(\mathrm{x}_k)\|}\right) \\ &\leq \left(1 - \frac{\|[E - \eta\, D_{\mathrm{y}_k}^{-1}F'(\mathbf{x}^*)][E - D_{\mathrm{x}_k}^{-1}F'(\mathbf{x}^*)]\|}{\|E + \beta F'(\mathbf{x}^*)\|}\right)\left(\frac{2\|E - D_{\mathrm{x}_k}^{-1}F'(\mathbf{x}^*)\|^3}{\|E + \beta F'(\mathbf{x}^*)\|^2}\right).\end{aligned} \tag{34}$$

Subtract next $\mathbf{x}^*$ from both sides of third step of (9) to obtain

$$\mathbf{x}_{k+1} - \mathbf{x}^* = \mathbf{z}_k - \mathbf{x}^* - \zeta D_{\mathrm{z}_k}{}^{-1}F(\mathbf{z}_k). \tag{35}$$

Reconsideration of (25) and (29) in (35) yields

$$\mathbf{x}_{k+1} - \mathbf{x}^* = k_1(\mathbf{x}_k - \mathbf{x}^*), \tag{36}$$

where $k_1 = \Big[E - \zeta D_{\mathrm{z}_k}{}^{-1}F'(\mathrm{x}^*)\Big]\Big[E - \eta D_{\mathrm{y}_k}{}^{-1}F'(\mathrm{x}^*)\Big]\Big[E - D_{\mathrm{x}_k}{}^{-1}F'(\mathrm{x}^*)\Big]$. The norms of both sides of (36) are

$$\|\mathbf{x}_{k+1} - \mathbf{x}^*\| \leq \|k_1\|\,\|\mathbf{x}_k - \mathbf{x}^*\|, \tag{37}$$

$$\|\mathbf{x}_{k+1} - \mathbf{x}^*\| \leq k\|\mathbf{x}_k - \mathbf{x}^*\|, \tag{38}$$

where $k = \|k_1\|$. From lemma 3.1 and boundedness of the Jacobian,there exist constants $\tau_1 \leq \tau_2$ such that $\tau_1\|c\|^2 \leq c^T F'(\mathbf{x})c \leq \tau_2\|c\|^2$ for all $\mathrm{x}, c \in \mathbb{R}^n$. We then let $\|D_k\| \leq max[\sqrt{n}, \beta_0, \beta_1, \ldots, \beta_n] = \alpha$ and $\delta = max[\tau_1, \tau_2]$, to write

$$\eta \leq 1 - \left(\frac{\sqrt{n} - \alpha^{-1}\delta}{\sqrt{n} + \beta\delta}\right), \tag{39}$$

$$\zeta \leq \left(1 - \frac{[\sqrt{n} - \eta\,\alpha^{-1}\delta][\sqrt{n} - \alpha^{-1}\delta]}{\sqrt{n} + \beta\delta}\right)\left(\frac{2[\sqrt{n} - \alpha^{-1}\delta]^3}{[\sqrt{n} + \beta\delta]^2}\right), \tag{40}$$

and

$$k = \|k_1\| \leq [\sqrt{n} - \zeta\alpha^{-1}\delta][\sqrt{n} - \eta\alpha^{-1}\delta][\sqrt{n} - \alpha^{-1}\delta], \tag{41}$$

which means that the sequence $\{\mathbf{x}_{k+1}\}_{k\geq 0}$ generated by (9) converges linearly to $\mathbf{x}^*$. Here the

proof ends. ■

4. Numerical Results

Definition [3] **4.1**. Let P and S be the set of problems and the set of solvers respectively. If n_s is the number of solvers and n_p is the number of problems, then for each problem $p \in P$ and for each solver $s \in S$, we define the performance profile $P : \mathbb{R} \rightarrow [0,1]$ by

$$P(\tau) = \frac{1}{n_p} size\{r_{p,s} \leq \tau : p \in P\}, \tag{42}$$

where

$$r_{p,s} = \frac{t_{p,s}}{min\{t_{p,s}: s \in S\}}, \tag{43}$$

and $t_{p,s}$=(computing time (or number of iterations, e.t.c) required to solve problem p by solver s). Then, $P(\tau)$ is the probability for solver $s \in S$ that a performance ratio $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio.

We shall apply here our proposed method for solving systems of nonlinear equations and explore its performance on some benchmark problems, with distinct test functions . We shall also compare our method with the following three well known methods, which are namely,
1. Newton's method (NM).
2. Classical Broyden's method (CB).
3. Newton's Method with Diagonal Jacobian Approximation (NDJA).

Problem 1: Spare function of Byeong (Beyong et. al, 2010)
$F_i(x) = (1 - x_i)^2 + x_i(1 + x_i x_{n-2} x_{n-1} x_n) - 2;$
$i = 1,2,\ldots,n$ and $x_0 = (0.5, 0.5, \ldots, 0.5)^T$

Problem 2: System of n nonlinear equations
$F_i(x) = (x_i^2 - 1)^2 - 2;$
$i = 1,2,\ldots,n$ and $x_0 = (1.2, 1.2, \ldots, 1.2)^T$

Problem 3: Extended system of Byeong, 2010
$F_i(x) = cos(x_i^2 - 1) - 1;$
$i = 1,2,\ldots,n$ and $x_0 = (0.5, 0.5, \ldots, 0.5)^T$

Problem 4: System of n nonlinear equations
$F_i(x) = \sum_{j=1}^{n} (x_j^2 sin(x_j) - x_j^4 + sin(x_j^2));$
$i = 1,2,\ldots,n$ and $x_0 = (0.2, 0.2, \ldots, 0.2)^T$

Problem 5: Extended system of Byeong, 2010
$F_i(x) = x_i^3;$
$i = 1,2,\ldots,n$ and $x_0 = (2, 2, \ldots, 2)^T$

Table 1. Dimension of problem 1 (n), Number of iterations (NI), Out of memory (-) and CPU Time (in seconds) and initial guess $x_0 = (0.5, 0.5, 0.5, \ldots\ldots, 0.5)^T$ and $\beta = 0.05$.

Problem	n	NM		NDJA		CB		TSDU	
		NI	CPU Time	NI	CPU Time	NI	CPU Time	NI	CPU Time
1	50	11	0.040085	8	0.003209	10	0.007449	4	0.002590
	200	11	0.229625	8	0.003903	10	0.128108	4	0.003184
	500	11	4.031931	8	0.006046	10	0.268096	4	0.004857
	1000	11	32.071952	8	0.008633	10	1.305538	4	0.004767
	250000	-	-	9	0.803323	-	-	4	0.646243

Table 2. Dimension of problem 2 (n), Number of iterations (NI), Out of memory (-) and CPU Time (in seconds) and initial guess $x_0 = (1.2, 1.2, 1.2, \ldots\ldots, 1.2)^T$ and $\beta = 0.0005$.

Problem	n	NM		NDJA		CB		TSDU	
		NI	CPU Time	NI	CPU Time	NI	CPU Time	NI	CPU Time
2	50	6	0.012765	9	0.001811	11	0.007295	2	0.001769
	200	6	0.131159	9	0.003145	11	0.065743	2	0.001964
	500	6	2.212296	10	0.005431	11	0.256695	2	0.001999
	1000	6	16.411831	10	0.006914	11	1.606041	2	0.002130
	250000	-	-	10	0.490253	-	-	2	0.204103

Table 3. Dimension of problem 3 (n), Number of iterations (NI), Out of memory (-) and CPU Time (in seconds) and initial guess $x_0 = (0.5, 0.5, 0.5, \ldots\ldots, 0.5)^T$ and $\beta = 0.5$.

Problem	n	NM		NDJA		CB		TSDU	
		NI	CPU Time	NI	CPU Time	NI	CPU Time	NI	CPU Time
3	50	37	0.082947	20	0.004176	20	0.012065	5	0.001284
	200	59	1.079917	20	0.006523	20	0.096731	6	0.002992
	500	90	35.062470	21	0.009232	21	0.552159	6	0.005726
	1000	-	-	21	0.018446	21	3.094933	6	0.009375
	250000	-	-	24	1.558995	-	-	7	0.903323

Table 4. Dimension of problem 4 (n), Number of iterations (NI), Out of memory (-) and CPU Time (in seconds) and initial guess $x_0 = (0.2, 0.2, 0.2, \ldots\ldots, 0.2)^T$ and $\beta = 0.0005$.

Problem	n	NM		NDJA		CB		TSDU	
		NI	CPU Time	NI	CPU Time	NI	CPU Time	NI	CPU Time
4	50	30	0.137274	16	0.007096	17	0.012739	6	0.004310
	200	33	1.144202	-	-	27	0.108190	7	0.006489
	500	34	16.148742	-	-	30	0.874449	7	0.024518
	1000	35	119.612709	-	-	31	4.744052	8	0.038428
	250000	-	-	-	-	-	-	11	6.029565

Table 5. Dimension of problem 5 (n), Number of iterations (NI), Out of memory (-) and CPU Time (in seconds) and initial guess $x_0 = (2, 2, \ldots, 2)^T$ and $\beta = 0.000001$.

Problem	n	NM		NDJA		CB		TSDU	
		NI	CPU Time	NI	CPU Time	NI	CPU Time	NI	CPU Time
5	50	51	0.149090	35	0.007965	29	0.020773	7	0.001829
	200	52	1.433389	36	0.018387	30	0.092941	8	0.005140
	500	54	24.569028	36	0.019504	30	0.795550	8	0.009990
	1000	55	149.558292	36	0.039307	31	4.757919	8	0.016604
	250000	-	-	40	9.707047	-	-	9	3.760534

Problem 6:(Generalized function of Rosenbrock)
$F_i(x) = x_i - \sum_{j=1}^{n} \frac{(x_j^2)}{n^2} + \sum_{j=1}^{n}(x_j) - n;$
$i = 1, 2, \ldots, n$ and $x_0 = (0.2, 0.2, \ldots, 0.2)^T$

Table 6. Dimension of problem 6 (n), Number of iterations (NI), Out of memory (-) and CPU Time (in seconds) and initial guess $x_0 = (0.2, 0.2, \ldots, 0.2)^T$ and $\beta = 0.00005$.

Problem	n	NM		NDJA		CB		TSDU	
		NI	CPU Time	NI	CPU Time	NI	CPU Time	NI	CPU Time
6	50	4	0.018255	16	0.003565	9	0.003218	2	0.000725
	200	4	0.103352	12	0.002480	8	0.046891	1	0.000824
	500	4	1.649348	10	0.002435	8	0.189331	1	0.001838
	1000	4	12.297475	10	0.005326	7	0.949198	1	0.001936
	250000	-	-	772	62.543454	-	-	-	-

The numerical results of the comparison between the methods is shown in Tables 1 to 6. Each problem is tested with five different dimensions, namely, n =50, 200, 500, 1000 and 250000. All functions are tested using a specified starting point. In all the tables NI stand for number of

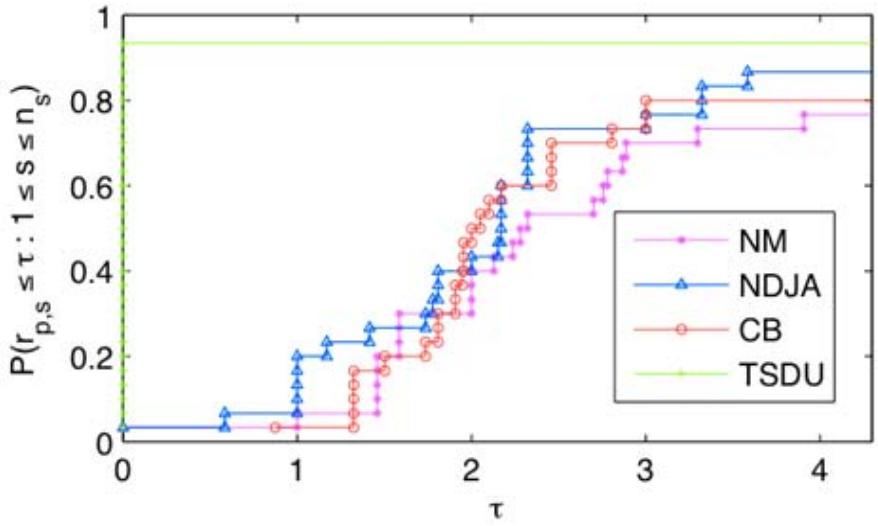

Figure 1. Comparison of the performance profile of NM, NDJA, CB and TSDU methods as the dimension increases (in terms of number of iterations)

iterations and CPU time is in seconds. In this paper TSDU stands for the method proposed. Comparing the performance of the methods in all the Tables shows that TSDU is better than all the methods compared, in terms of number of iterations and CPU time, and for all the tested 6 problems.The computational experiments were done using MATLAB 7.5 with double precision arithmetic, and the comparison was done based on performance profile of the methods in terms of the number of iterations. The stopping criterion used is:

$\|F(\mathbf{x}_k)\| \leq 10^{-8}$.

4. Conclusion

In this work, we proposed a linear convergence iterative derivative-free diagonal updating method for solving large-scale systems of nonlinear equations. The proposed method uses Newton's method in all the three steps with approximate Jacobian matrix as a diagonal matrix. The proposed method is attractive as it converges with less iterations than the Classical Newton's, Broyden's and Newton's method with diagonal Jacobian approximation. Finally, we may conclude that the proposed method can be a good alternative to Classical Newton's and

Broyden's methods, especially for large scale nonlinear systems of equations.

Acknowledgements
The authors would like to thank the referee for some comments which have led to an improvement of the paper.

References

[1] C. G. Broyden, A class of methods for solving nonlinear simultaneous equations, *Mathematics of Computation* **19** (92), (1965), 577-593.

[2] J. E. Dennis, and J. J. Jr. More, A characterization of superlinear convergence and its application to quasi-Newton methods, *Mathematics of Computation* **28**, (1974), 199-208.

[3] E. D. Dolan, and J. J. More, Benchmarking optimization software with performance profiles, *Mathematical Programing* **91**, (1964), 201-213.

[4] C. T. Kelly, *Iterative Methods for Linear and Nonlinear Equations,* SIAM, Philadelphia, PA, 1995.

[5] C. T. Kelly, *Solving Nonlinear Systems With Newton Method*, SIAM, Philadelphia, PA, 2003.

[6] K. Natasa, and L. Zorana, Newton-like methods with modification of the right hand side vector, *Mathematics of Computation* **71**, (2002), 237-250.

[7] J. F. Steffenses, Remarks on iteration, *Skand Aktuar Tidsr* **16**, (1933), 64-72.

[8] R. Thukral, A family of three-point derivative-free methods of eighth-order for solving nonlinear equations, *Journal of Modern Methods in Numerical Mathematics* **3**(2), (2012), p.11.

[9] M. Y. Waziri, W. J. Leong, M. A. Hassan, and M. Monsi, Jacobian computation-free Newton's method for systems of nonlinear equations, *Journal of Numerical Mathematics and Stochastics* **2**(1), (2010), 54-63.

[10] M. Y. Waziri, W. J. Leong, M. A. Hassan , and A. U. Moyi, Two-step derivative-free diagonally Newton's method for large-scale nonlinear equations, *World Applied Sciences Journal* **21** *(Special issue of Applied Mathematics),* (2013), 86-94.

[11] M. Y. Waziri, W. J. Leong, M. A. Hassan, and M. Monsi, A Newton's method with diagonal Jacobian approximation for systems of nonlinear equations, *Journal of Mathematics and Statistics* **6**(3), (2010), 246-252.

Article history: Submitted November, 17, 2014; Accepted March, 21, 2015