Taylor & Francis
Taylor & Francis Group

Optimization

A Journal of Mathematical Programming and Operations Research

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/gopt20

Modified matrix-free methods for solving system of nonlinear equations

Mohammed Yusuf Waziri , Hadiza Usman Muhammad , Abubakar Sani Halilu & Kabiru Ahmed

To cite this article: Mohammed Yusuf Waziri , Hadiza Usman Muhammad , Abubakar Sani Halilu & Kabiru Ahmed (2020): Modified matrix-free methods for solving system of nonlinear equations, Optimization, DOI: 10.1080/02331934.2020.1778689

To link to this article: https://doi.org/10.1080/02331934.2020.1778689

Published online: 30 Jun 2020.

OPTIMIZATION
https://doi.org/10.1080/02331934.2020.1778689

Taylor & Francis
Taylor & Francis Group

Modified matrix-free methods for solving system of nonlinear equations

Mohammed Yusuf Waziri[a,b], Hadiza Usman Muhammad[a,b], Abubakar Sani Halilu[c,b] and Kabiru Ahmed[a,b]

[a]Department of Mathematical Sciences, Bayero University, Kano, Nigeria; [b]Numerical Optimization Research Group, Department of Mathematical Sciences, Bayero University, Kano, Nigeria; [c]Department of Mathematics and Computer Science, Sule Lamido University, Kafin Hausa, Nigeria

ABSTRACT

Some hybrid derivative-free methods for nonlinear system of equations via acceleration and correction parameters are presented. These methods are based on applying the three term hybrid iterative scheme on an enhance matrix free method. The step length was determined using an inexact line search procedure. Moreover, the proposed methods proved to be globally convergent under mild conditions. The comprehensive numerical results presented in this work, show the efficiency of the proposed methods by comparing them with some existing methods in the previous literature.

ARTICLE HISTORY
Received 24 January 2020
Accepted 31 May 2020

KEYWORDS
Derivative-free; matrix-free; decent direction; global convergence; acceleration parameter; correction parameter

MATHEMATICS SUBJECT CLASSIFICATIONS
65H11; 65K05; 65H12; 65H18

1. Introduction

The problem of finding solutions of system of nonlinear equations have gained so much attention from researchers over the decades. This is due to the significant role it plays in science and engineering. Generally, a typical system of nonlinear equations is formulated as

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

where $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is nonlinear map.

In the past decades, researchers have developed numerous methods for solving (1). Notable among them is the Newton's method [1,2], which generates a sequence of points using the recursive formula:

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, 2, \ldots, \tag{2}$$

where, α_k is a step length to be computed by a suitable line search technique, x_{k+1} represents a new iterate and x_k is the previous iterate, while d_k is the search

CONTACT Abubakar Sani Halilu abubakars.halilu@slu.edu.ng

direction to be calculated by solving the following system of linear equations,

$$F'(x_k)d_k = -F(x_k), \tag{3}$$

where $F'(x_k)$ is the Jacobian matrix of $F(x_k)$ at x_k. The method is popular because it converges quadratically and is easy to implement. Furthermore, (1) can come from an unconstrained optimization problem, a saddle point and equality constrained problem [3]. Let f be a merit function defined by

$$f(x) = \tfrac{1}{2}\|F(x)\|^2. \tag{4}$$

The nonlinear system in (1) is equivalent to the following global optimization problem

$$\min f(x), \quad x \in \mathbb{R}^n.$$

Generally, the search direction d_k is required to satisfy the descent condition

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

The step length α_k is determined in different ways either by exact or inexact line search technique. The most commonly used line search in practice is the inexact line search as presented in [2,4–6], which sufficiently decreases the function values i.e to establish

$$\|F(x_k + \alpha_k d_k)\| \leq \|F(x_k)\|. \tag{5}$$

Despite the attractive features of Newton's method, it is not popular for solving large-scale systems of nonlinear equations. This is due to the fact that the scheme requires computation as well as storage of the Jacobian matrix and it's inverse, which requires derivatives at each iterations. It is also a known fact that in some instances, derivatives of some functions may not exist. However, in many practical situations it is preferable to avoid the computation of the Jacobian matrix and avoid the huge cost of matrix storage. To overcome these difficulties, several strategies have been developed that are Jacobian-free such as the Jacobian-free Newton's method for solving systems of nonlinear equations proposed in [1], which approximates the Jacobian inverse with a diagonal matrix D_k given by

$$F'(x_k)^{-1} \approx D_k, \tag{6}$$

where $D_k = \text{diag}(d^i_{k+1})$ and d_{k+1} is defined as

$$d^i_{k+1} = \frac{x^i_{k+1} - x^i_k}{F_i(x_{k+1}) - F_i(x_k)}, \tag{7}$$

where $F_i(x_k)$ is the ith component of the vector $F(x_k)$, x^i_k is the ith component of the vector (x_k) and d^i_k is the ith diagonal element of D_k, respectively. The update

formula of the method is given by

$$x_{k+1} = x_k - D_k F(x_k). \tag{8}$$

Interestingly, the method does not require Jacobian matrix and its inverse, which implies it is derivative free. Moreover, in [3] a matrix free secant method for solving system of nonlinear equation is presented, where the authors utilized data from two preceding steps rather than a single step using weak secant equation. The method generates the sequence of point as

$$x_{k+1} = x_k - Q_k^{-1} F(x_k), \tag{9}$$

where Q_k is a diagonal approximation of the Jacobian matrix updated in each iteration as

$$Q_{k+1} = Q_k + \frac{\rho_k^T \mu_k - \rho_k^T Q_k \rho_k}{Tr(H_k^2)} H_k, \tag{10}$$

where $\rho_k = s_k - \alpha_k s_{k-1}$, $\mu_k = y_k - \alpha y_{k-1}$, $\alpha_k = 1, \forall k$, $H_k = \text{diag}((\rho_k^{(1)})^2, (\rho_k^{(2)})^2, \ldots, (\rho_k^{(n)})^2)$, $Tr(H_k^2) = \sum_{i=1}^{n} (\rho_k^{(i)})^4$ and Tr is the trace of the matrix H_k^2, respectively. Moreover, for Q_{k+1} to be well-defined, the following condition is set

$$Q_{k+1} = \begin{cases} Q_k + \dfrac{\rho_k^T \mu_k - \rho_k^T Q_k \rho_k}{Tr(H_k^2)} H_k; & \text{if } \|\rho_k\| > 10^{-4}. \\ Q_k & \text{otherwise.} \end{cases} \tag{11}$$

The purpose of the two-step multi point scheme to increase the accuracy of the approximation of the Jacobian. Furthermore, in [7], a derivative free method was also developed based on approximating the Jacobian with a diagonal matrix by means of an acceleration parameter as

$$F'(x_k) \approx \gamma_k I,$$

where I is an identity matrix and $F'(x_k)$ is the Jacobian matrix of $F(x_k)$ at x_k. The authors in [7] used the idea of accelerated double direction method for unconstrained optimization problem, originally presented in [8] as follows:

$$x_{k+1} = x_k + \alpha_k b_k + \alpha_k^2 c_k, \tag{12}$$

where b_k and c_k are search directions. To improve good direction towards the solution, they defined the two direction as $b_k = \gamma_k^{-1} F(x_k)$ and $c_k = -F(x_k)$ and

generate the sequence of point as

$$x_{k+1} = x_k + (\alpha_k + \alpha_k^2 \gamma_k) d_k, \tag{13}$$

where $d_k = -\gamma_k^{-1} F(x_k)$ and obtain γ_{k+1} using the Taylor's series expansion of the first order as

$$\gamma_{k+1} = \frac{y_k^T y_k}{y_k^T (\alpha_k + \alpha_k^2 \gamma_k) d_k}, \tag{14}$$

where $y_k = F(x_{k+1}) - F(x_k)$. The step length α_k is obtained using a derivative free line search [9]. In order to improved the convergence properties of double direction scheme, the double step size approach for unconstrained optimization problems is proposed in [10]. The authors in [10] defined the double step size scheme by

$$x_{k+1} = x_k + \alpha_k b_k + \beta_k c_k, \tag{15}$$

where α_k and β_k are step lengths. The authors in [5], incorporated the idea of scheme defined in (15) to solve the system of nonlinear equations by defining the two directions as $b_k = \gamma_k^{-1} F(x_k)$ and $c_k = -F(x_k)$ with additional assumption that $\alpha_k + \beta_k = 1$. The method in [5] produced a sequence of iterates $\{x_k\}$ such that

$$x_{k+1} = x_k + \alpha_k \left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1 \right) F(x_k), \tag{16}$$

and the acceleration parameter as

$$\gamma_{k+1} = \frac{y_k^T y_k}{y_k^T s_k}, \tag{17}$$

where $y_k = F(x_{k+1}) - F(x_k)$ and $s_k = x_{k+1} - x_k$. They also obtain the step length α_k using a derivative free line search proposed in [9]. The double direction and step length methods are further improved in [11–14] by using the hybridization approach of Picard-Maan hybrid iterative process [15]. This is made possible in order to enhance the convergence properties and numerical results of the double direction and step length methods. In addition, some hybrid methods have been proposed over years to improve performance of some existing methods, most of the hybrid methods are obtained by constructing a new update parameter as a linear combination of two or more classical or modified update parameters. Andrei in [16] proposed a hybrid method by combining the two conjugate gradient parameters β^{PRP} and β^{DY} as a convex combination given as

$$\beta = (1 - \theta)\beta^{PRP} + \theta \beta^{DY}, \tag{18}$$

where the parameter in the convex combination is computed in such a way that the conjugacy condition is satisfied. These methods possessed better computational performance and strong convergence properties (see [17,18] for more).

In [19], Petrovic et al. developed a gradient descent algorithm by hybridizing the accelerated gradient descent (SM) [20] method with the Picard-Maan hybrid iterative process [15] defined by the next three relations:

$$\begin{aligned} x_1 &= x \in \mathbb{R}^n \\ y_k &= (1-\beta_k)x_k + \beta_k T x_k, \\ x_{k+1} &= T y_k, \end{aligned} \tag{19}$$

where $T : C \longrightarrow C$ is a mapping defined on nonempty convex subset C of a normed space $\mathbb{E}$, x_k and y_k are sequences determined by the iteration (19) and $\beta_k \in (0, 1)$ is the sequence of positive numbers denoted as correction parameter, the acceleration parameter proposed is given by

$$\gamma_{k+1} = 2\gamma_k \frac{\gamma_k[f(x_{k+1}) - f(x_k)] + (\beta_k + 1)\alpha_k \|g_k\|^2}{(\beta_k + 1)^2 \alpha_k^2 \|g_k\|^2}. \tag{20}$$

The authors determined the accelerated parameter γ_k using the Taylor's series expansion of the second order. Moreover, the step length t_k is determined using the inexact backtracking line search technique. The proposed method was numerically tested and compared with the SM method [20], with respect to number of iterations, CPU time and functions evaluation, respectively. The numerical results indicated that the proposed method performed much better compared to the SM method regarding all the tested characteristics, which is an indication of the positive impact of the correction parameter on the method. So, the choice of the value for the correction parameter affected the efficiency of the method. In [11] a hybrid accelerated double direction gradient method was presented by exploiting nice features of the hybridization method in [19] and the iterative scheme presented in [8], resulted to the iterative scheme

$$x_{k+1} - x_k = \beta {\alpha_k}^2 d_k - \beta \alpha_k \gamma_k^{-1} g_k, \tag{21}$$

for $\beta \in (1, 2)$. By employing the second-order Taylor's expansion, the acceleration parameter for the scheme is presented

$$\gamma_{k+1} = 2\frac{f(x_{k+1}) - f(x_k) - \beta g_k^T(\alpha_k^2 d_k - \alpha_k \gamma_k^{-1} g_k)}{\beta^2 \alpha_k^2 (\alpha_k d_k - \gamma_k^{-1} g_k)^T (\alpha_k d_k - \gamma_k^{-1} g_k)}. \tag{22}$$

According to the conducted numerical experiments, it is observed that the proposed method outperformed the existing accelerated double direction method [8].

It is clear from above literature that hybridization process improves performance of some existing accelerated methods. The nice properties of hybridized methods and the fact that hybridization methods for large-scale systems of nonlinear equations are rare in the literature, are the inspirations for this research.

Furthermore, all the methods discussed above are developed for solving nonlinear unconstrained optimization problems. So, our motive is to extend the ideas discussed above to system of nonlinear equations.

We arrange the remaining sections of the paper as follows: Section 2 is dedicated to derivation of the proposed method and its algorithm. Convergence analysis is presented in Section 3. Numerical results are presented in Section 4. and we make concluding remarks in Section 5.

2. Derivation of the methods and their algorithms (MDF)

Here, we present the hybrid of Picard-Mann hybrid iterative process (19) with En enhanced matrix-free method via double step length approach for solving systems of nonlinear equations [5]. We present the following iterative scheme:

$$x_1 = x \in \mathbb{R}^n, \tag{23}$$

$$y_k = (1 - \beta_k)x_k + \beta_k T x_k, \tag{24}$$

$$x_{k+1} = T y_k, \tag{25}$$

where the function T is defined as

$$T(x_k) = x_k - \alpha_k \left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1 \right) F(x_k), \tag{26}$$

$$y_k = x_k - \beta_k \alpha_k \left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1 \right) F(x_k), \tag{27}$$

and substituting (27) in (25) we obtain

$$x_{k+1} = x_k - \alpha_k(\beta_k + 1) \left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1 \right) F(x_k), \tag{28}$$

where γ_k and β_k are acceleration and correction parameters, respectively.

From (28) we can define our proposed direction as:

$$d_k = -(\beta_k + 1) \left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1 \right) F(x_k), \tag{29}$$

then using (28) and (29) we have the general scheme as:

$$x_{k+1} = x_k + \alpha_k d_k. \tag{30}$$

Now, to find γ_{k+1} we consider the Taylor's series expansion of order 1 at x_{k+1} as

$$F(x_{k+1}) = F(x_k) + F'(\xi)(x_{k+1} - x_k), \tag{31}$$

where $\xi_k \in (x_k, x_{k+1})$. The distance between x_k and x_{k+1} is small enough and $\xi_k = x_k + \rho(x_{k+1} - x_k)$, $\rho \in [0, 1]$, we take $\rho = 1$ such that $\xi_k = x_{k+1}$. Therefore, we assume that

$$F'(\xi) \approx \gamma_{k+1} I. \tag{32}$$

By substituting (32) in (31), we obtain

$$F(x_{k+1}) - F(x_k) = \gamma_{k+1}(x_{k+1} - x_k), \tag{33}$$

where $y_k = F(x_{k+1}) - F(x_k)$ and $s_k = (x_{k+1} - x_k) = -\alpha_k(\beta_k + 1)(\gamma_k^{-1} + (1/\alpha_k) - 1)F(x_k)$ such that

$$y_k = -\gamma_{k+1}\alpha_k(\beta_k + 1)\left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1\right)F(x_k), \tag{34}$$

by multiplying both side of (34) by y_k^T, we obtain the proposed acceleration parameter as

$$\gamma_{k+1} = -\frac{y_k^T y_k}{y_k^T \alpha_k(\beta_k + 1)\left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1\right)F(x_k)}. \tag{35}$$

To compute the step-length α_k, we use the derivative-free line search used in [9,21,22,28]. Let $\omega_1 > 0, \omega_2 > 0$ and $r \in (0, 1)$ be constants and let η_k be a given positive sequence such that

$$\sum_{k=0}^{\infty} \eta_k < \eta < \infty \tag{36}$$

and

$$f(x_k + \alpha_k d_k) - f(x_k) \leq -\omega_1 \|\alpha_k F(x_k)\|^2 - \omega_2 \|\alpha_k d_k\|^2 + \eta_k f(x_k). \tag{37}$$

Let i_k be the smallest nonnegative integer i such that (37) holds for $\alpha = r^i$. Let $\alpha_k = r^{i_k}$. Next we present the algorithm of the proposed method. Remark: For the correction parameter, we chose a fixed value for all k as $\beta_k + 1 \equiv \lambda$ i.e $(\beta_k + 1 \equiv \lambda \in (1, 2)\ \forall k)$, we proposed a direction

$$d_k^* = -(\beta_k + 1)\left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1\right)F(x_k). \tag{38}$$

Algorithm 2.1 (MDF1):
STEP 1: Given $x_0, \gamma_0 = 1, \epsilon > 0, \alpha_0 = 1$ $((\beta_k + 1) \equiv \lambda \in (1, 2)$ set $k = 0$.
STEP 2: Compute $F(x_k)$, if $\|F(x_k)\| \leq \epsilon$, stop, else goto step 3.
STEP 3: Compute d_k^* (using (38)).
STEP 4: Compute step length α_k (using (37)).
STEP 5: Set $x_{k+1} = x_k + \alpha_k d_k^*$.
STEP 6: Determine γ_{k+1} (using (35)).
STEP 7: Set $k = k + 1$, and go to STEP 2.

The Proposed second choice of the correction parameter Here, we try to obtain a sequence of β_k which will be updated in each iteration.

By (Barzilai and Borwein, 23) [23], the parameter γ_k is defined as

$$\gamma_{k+1}^{BB} = \frac{s_k^T y_k}{s_k^T s_k}. \tag{39}$$

So going by this, we adapt (39) to be our correction parameter β_k i.e

$$\beta_{k+1} = \frac{s_k^T y_k}{s_k^T s_k}, \tag{40}$$

where $y_k = F(x_{k+1}) - F(x_k)$ and $s_k = x_{k+1} - x_k$. By substituting (40) in (28) we have the scheme

$$x_{k+1} = x_k - \alpha_k \left(\frac{s_k^T y_k}{s_k^T s_k} + 1 \right) \left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1 \right) F(x_k). \tag{41}$$

Our proposed second direction d_k is given by

$$d_k^{**} = \begin{cases} -(\beta_k + 1)\gamma_k^{-1} F(x_k) & if \quad k = 0, \\ -\left(\frac{s_k^T y_k}{s_k^T s_k} + 1 \right) \left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1 \right) F(x_k), & if \quad k \geq 1, \end{cases} \tag{42}$$

where, $\gamma_k = y_k^T y_k / y_k^T s_k$.

We present the second algorithm as follows

Algorithm 2.2 (MDF2):
STEP 1: Given x_0, $\gamma_0 = 1$, $\epsilon > 0$, $\alpha_0 > 0$, ($\beta_k = 0.2$), set $k = 0$.
STEP 2: Compute $F(x_k)$, if $\|F(x_k)\| \leq \epsilon$, stop, else goto step 3.
STEP 3: Compute d_k^{**} (using (42)).
STEP 4: Compute step length α_k (using (37)).
STEP 5: Set $x_{k+1} = x_k + \alpha_k d_k^{**}$.
STEP 6: Determine γ_{k+1} (using (35)).
STEP 7: Determine β_{k+1} (using (40)).
STEP 8: Set $k = k + 1$, and go to STEP 2.

Remark 2.1: Since the correction parameter $\beta_k \in (0, 1)$, so if in some iterations the value of $\beta_k \geq 1$, then we take $\beta_k = 0.5$.

3. Convergence analysis

In this section we present the global convergence of our methods. First, we define the level set

$$\Omega = \{x : \|F(x)\| \leq \|F(x_0)\|\}. \tag{43}$$

In order to analyse the convergence of Algorithms 2.2, we need the following assumptions:

Assumption 3.1: (1) There exists $x^* \in \mathbb{R}^n$ such that $F(x^*) = 0$.
(2) F is continuously differentiable in some neighbourhood 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

$$\|F'(x)\| \leq M \quad \forall x \in N, \tag{44}$$

and

$$m\|d\|^2 \leq d^T F'(x)d \quad \forall x \in N, d \in \mathbb{R}^n. \tag{45}$$

Remark 3.1: Assumption 3.1 implies that there exists a constants $M > m > 0$ such that

$$m\|d\| \leq \|F'(x)d\| \leq M\|d\| \quad \forall x \in N, d \in \mathbb{R}^n. \tag{46}$$

$$m\|x - y\| \leq \|F(x) - F(y)\| \leq M\|x - y\| \quad \forall x, y \in N. \tag{47}$$

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

$$m\|x - x^*\| \leq \|F(x)\| = \|F(x) - F(x^*)\| \leq M\|x - x^*\|, \tag{48}$$

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

Lemma 3.1: *Suppose that Assumption 3.1 holds $\{x_k\}$ is generated by Algorithm 2.2. Then there exists a constant $m > 0$ such that for all k.*

$$s_k^T[F(x_k - \alpha_k(\beta_k + 1)(\gamma_k^{-1} + \alpha_k^{-1} - 1)F(x_k)) - F(x_k)] \geq m\|s_k\|^2. \tag{49}$$

Proof: By mean-value theorem and (45) we have,
$s_k^T[F(x_k - \alpha_k(\beta_k + 1)(\gamma_k^{-1} + (1/\alpha_k) - 1)F(x_k)) - F(x_k)] = s_k^T F'(\xi)s_k \geq m\|s_k\|^2$.
Where $\xi_k = x_k + \zeta(x_{k+1} - x_k)$, $\zeta \in (0, 1)$. The proof is complete.
Using $y_k^T s_k \geq m\|s_k\|^2 > 0$, γ_{k+1} is always generated by the update formula (35), and we can deduce that $\gamma_{k+1}I$ inherits the positive definiteness of $\gamma_k I$. ■

Proposition 3.1: *By lemma 3.1 and (47), we obtained*

$$\frac{y_k^T s_k}{\|s_k\|^2} \geq m, \quad \frac{\|y_k\|^2}{y_k^T s_k} \leq \frac{M^2}{m}. \tag{50}$$

Lemma 3.2: *Suppose that assumption 1 holds and $\{x_k\}$ is generated by Algorithm 2.2. Then we have*

$$\lim_{k\to\infty} \|\alpha_k d_k\| = 0, \tag{51}$$

and

$$\lim_{k\to\infty} \|\alpha_k F(x_k)\| = 0. \tag{52}$$

Proof: By (37) we have for all $k > 0$

$$\begin{aligned}\omega_2\|\alpha_k d_k\|^2 &\leq \omega_1\|\alpha_k F(x_k)\|^2 + \omega_2\|\alpha_k d_k\|^2 \\ &\leq \|F(x_k)\|^2 - \|F(x_{k+1})\|^2 + \eta_k\|F(x_k)\|^2, \end{aligned} \tag{53}$$

by summing the above inequality, we have

$$\begin{aligned}\omega_2\sum_{i=0}^{k}\|\alpha_i d_i\|^2 &\leq \sum_{i=0}^{k}\left(\|F(x_i)\|^2 - \|F(x_{i+1})\|^2\right) + \sum_{i=0}^{k}\eta_i\|F(x_i)\|^2 \\ &= \|F(x_0)\|^2 - \|F(x_{k+1})\|^2 + \sum_{i=0}^{k}\eta_i\|F(x_i)\|^2 \\ &\leq \|F(x_0)\|^2 + m_1^2\sum_{i=0}^{k}\eta_i \\ &\leq \|F(x_0)\|^2 + m_1^2\sum_{i=0}^{\infty}\eta_i, \end{aligned} \tag{54}$$

so from the level set and fact that $\{\eta_k\}$ satisfies (36) then the series $\sum_{i=0}^{\infty}\|\alpha_i d_i\|^2$ is convergent. This implies (51). By similar argument we can prove that (52) holds. ■

Lemma 3.3: *Suppose that Assumption 3.1 holds and $\{x_k\}$ is generated by algorithm 2.2. Then there exists a positive constant m_1 such that for all $k > 0$,*

$$\|d_k\| \leq m_1. \tag{55}$$

Proof: From (40), we have

$$|\beta_k| = \left|\frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}}\right| \leq \frac{\|s_{k-1}\|\|y_{k-1}\|}{\|s_{k-1}\|\|s_{k-1}\|} \leq \frac{M\|s_{k-1}\|}{\|s_{k-1}\|} \leq M. \tag{56}$$

Therefore, from (29), (47), (52) and (56) we have

$$\begin{aligned}\|d_k\| &= \left\|-(\beta_k + 1)(\gamma_k^{-1} + \alpha_k^{-1} - 1)F(x_k)\right\|, \\ &\leq |\beta_k + 1|\left\|\frac{y_{k-1}^T s_{k-1}}{\|y_{k-1}\|^2}F(x_k) + (\alpha_k^{-1} - 1)F(x_k)\right\|, \\ &\leq (M+1)\left\|\frac{y_{k-1}^T s_{k-1}F(x_k)}{\|y_{k-1}\|^2}\right\| + \|(\alpha_k^{-1} - 1)F(x_k)\|, \\ &\leq (M+1)\frac{\|y_{k-1}\|\|s_{k-1}\|\|F(x_k)\|}{m^2\|s_{k-1}\|^2} + \|(\alpha_k^{-1} - 1)F(x_k)\|, \end{aligned}$$

$$\leq (M+1)\frac{M\|s_{k-1}\|\|F(x_0)\|}{m^2\|s_{k-1}\|} + \|(\alpha_k^{-1}-1)F(x_k)\|,$$
$$\leq (M+1)\frac{M\|F(x_0)\|}{m^2} + P, \tag{57}$$

where P is some positive constant. Taking $m_2 = \frac{(M+1)M\|F(x_0)\|}{m^2} + P$, we have (55). The proof is completed.

Since $\gamma_k I$ approximates $F'(x_k)$ along direction s_k, we can give the following assumption. ■

Assumption 3.2: $\gamma_k I$ is a good approximation to $F'(x_k)$, i.e.

$$\|(F'(x_k) - \gamma_k I)d_k\| \leq \epsilon\|F(x_k)\|, \tag{58}$$

where $\epsilon \in (0,1)$ is a small quantity [6].

Lemma 3.4: *Let Assumption 3.2 holds and $\{x_k\}$ be generated by Algorithm 2.2. Then d_k is a descent direction for $f(x_k)$ at x_k i.e*

$$\nabla f(x_k)^T d_k < 0. \tag{59}$$

Proof: from (29), we have

$$\begin{aligned}\nabla f(x_k)^T d_k &= F(x_k)^T F'(x_k)d_k \\ &= F(x_k)^T[(F'(x_k) - \gamma_k I)d_k - \gamma_k\sigma_k F(x_k)] \\ &= F(x_k)^T((F'(x_k) - \gamma_k I)d_k - \gamma_k\sigma_k\|F(x_k)\|^2,\end{aligned} \tag{60}$$

by Cauchy-Schwartz inequality and (58), we have,

$$\begin{aligned}\nabla f(x_k)^T d_k &\leq \|F(x_k)\|\|((F'(x_k) - \gamma_k I)d_k\| - \gamma_k\sigma_k\|F(x_k)\|^2, \\ &\leq -(|\sigma_k| - \epsilon)\|F(x_k)\|^2,\end{aligned} \tag{61}$$

where $\sigma_k = (\beta_k + 1)(\gamma_k^{-1} + (1/\alpha_k) - 1)$ and $\gamma_k^{-1} \geq \frac{m}{M^2} > 0$. Hence for $\epsilon \in (0,1)$ this lemma is true. ■

Lemma 3.5: *Let Assumption 3.2 hold and $\{x_k\}$ be generated by Algorithm 2.2. Then $\{x_k\} \subset \Omega$.*

Proof: By lemma 3.2, we have $\|F(x_{k+1})\| \leq \|F(x_k)\|$. Moreover, we have for all k,

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

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

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 3.1: *Suppose that Assumption 3.1 holds,*$\{x_k\}$ *is generated by Algorithm 2.2. Assume further for all* $k > 0$,

$$\alpha_k \geq c\frac{|F(x_k)^T d_k|}{\|d_k\|^2}, \tag{62}$$

where c is some positive constant. Then

$$\lim_{k\to\infty} \|F(x_k)\| = 0. \tag{63}$$

Proof: From lemma 3.5 we have (55). Therefore by (51) and the boundedness of $\{\|d_k\|\}$, we have

$$\lim_{k\to\infty} \alpha_k \|d_k\|^2 = 0, \tag{64}$$

on the other hand from (29) we have,

$$F(x_k)^T d_k = -\sigma_k \|F(x_k)\|^2, \tag{65}$$

where $\sigma_k = (\beta_k + 1)(\gamma_k^{-1} + (1/\alpha_k) - 1)$.

Now,

$$\begin{aligned} \|F(x_k)\|^2 &= \left\| -F(x_k)^T d_k \sigma_k^{-1} \right\| \\ &\leq | - F(x_k)^T d_k \| \sigma_k^{-1}|, \end{aligned} \tag{66}$$

but from (49) we have,

$$\gamma_k \leq \left(\frac{M^2}{m}\right),$$

then

$$|\gamma_k| \leq \left(\frac{M^2}{m}\right).$$

Also,

$$\sigma_k = (\beta_k + 1)\left(\gamma_k^{-1} + \frac{1}{\alpha_k} - 1\right) \geq (m+1)\left(\frac{m}{M^2} + \frac{1}{\alpha_k} - 1\right),$$

$$\sigma_k^{-1} \leq \frac{1}{(m+1)(\frac{m}{M^2} + \frac{1}{\alpha_k} - 1)},$$

then

$$|\sigma_k^{-1}| \leq \frac{1}{(m+1)(\frac{m}{M^2}+\frac{1}{\alpha_k}-1)}.$$

So from (65)we have

$$\|F(x_k)\|^2 \leq |F(x_k)^T d_k| \frac{1}{(m+1)(\frac{m}{M^2}+\frac{1}{\alpha_k}-1)}. \tag{67}$$

Thus

$$0 \leq \|F(x_k)\|^2 \leq |F(x_k)^T d_k| \frac{1}{(m+1)(\frac{m}{M^2}+\frac{1}{\alpha_k}-1)} \longrightarrow 0. \tag{68}$$

Therefore

$$\lim_{k\to\infty} \|F(x_k)\| = 0. \tag{69}$$

This completes the proof. ■

4. Numerical results

In this section, we compared the performance of our methods (MDF1) and (MDF2) with an enhanced matrix-free method via double step length approach for solving systems of nonlinear equations (EMFD) [5] and A transformed double step-length method for solving large-scale systems of nonlinear equations (TDS) [24]. For all the algorithms, the following parameters are set $\omega_1 = \omega_2 = 10^{-4}$, $r = 0.2$ and $\eta_k = 1/(k+1)^2$.

The codes ware written in Matlab (8.3.0 532) R2014a and run on a personal computer 1.60 GHz CPU processor and 4 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; (1) Memory requirement (2) Number of iterations exceed 1000. (3) If $\|F(x_k)\|$ is not a number.

Problem 1 ([25]):

$$F_i(x) = x_i - \left(1 - \frac{c}{2n}\sum_{j=1}^{n}\frac{\mu_i x_j}{\mu_i + \mu_j}\right)^{-1}$$
$$i = 1, 2, \ldots, n, \quad j = 1, 2, \ldots, n.$$
$$x_0 = (1, 1/4, 1/9, \ldots, 1/n^2)^T.$$

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

Problem 2 ([26]):

$$\begin{aligned} F_i(x) &= x_i^2 + x_i - 2, \\ i &= 1, 2, 3, \ldots, n, \\ x_0 &= (0, 1/2, 2/3, \ldots, 1 - 1/n)^T. \end{aligned}$$

Problem 3 ([27]):

$$\begin{aligned} F_i(x) &= x_i^2 - cos(x_i - 1), \\ i &= 1, 2, \ldots, n, \\ x_0 &= (0.1, 0.1, \ldots, 0.1)^T. \end{aligned}$$

Problem 4 ([24]):

$$\begin{aligned} F_i(x) &= (1 - x_i^2) + x_i(1 + x_i x_{n-2} x_{n-1} x_n) - 2. \\ i &= 1, 2, \ldots, n, \\ x_0 &= (0.3, 0.3, \ldots, 0.3)^T. \end{aligned}$$

Problem 5 ([26]):

$$\begin{aligned} F_i(x) &= x_i - 3x_i \left(\frac{\sin x_i}{3} - 0.66 \right) + 2. \\ i &= 1, 2, \ldots, n, \\ x_0 &= (1, 1/2, \ldots, 1/n)^T. \end{aligned}$$

Problem 6 ([24]):

$$\begin{aligned} F_1(x) &= x_1(x_1^2 + x_2^2) - 1, \\ F_i(x) &= x_i(x_{i-1}^2 + 2x_i^2 + x_{i+1}^2) - 1, \\ F_n(x) &= x_n(x_{n-1}^2 + x_n^2), \\ i &= 2, 3, \ldots n - 1. \\ x_0 &= (0, 1/2, \ldots, 1 - 1/n)^T. \end{aligned}$$

Problem 7 ([24]):

$$\begin{aligned} F_1(x) &= 3x_1^3 + 2x_2 - 5 + \sin(x_1 - x_2) \sin(x_1 + x_2), \\ F_i(x) &= -x_{i-1} e^{x_{i-1} - x_i} + x_i(4 + 3x_i^2) + 2x_{i+1} \\ &\quad + \sin(x_i - x_{i+1}) \sin(x_i + x_{i+1}) - 8, \\ F_n(x) &= -x_{n-1} e^{x_{n-1} - x_n} + 4x_n - 3, \end{aligned}$$

$$i = 2, 3, \ldots, n-1.$$
$$x_0 = (1, 0 \cdots, 2/n-1)^T.$$

Problem 8 ([26]):

$$F_i(x) = e^{x_i^{2-i}} - cos(1 - x_i),$$
$$i = 1, 2, \ldots, n.$$
$$x_0 = (0.9, 0.9, \ldots, 0.9)^T.$$

Problem 9 ([24]):

$$F_i(x) = x_i^2 - 1,$$
$$i = 1, 2, 3, \ldots, n.$$
$$x_0 = (0.8, 0.8, \ldots, 0.8)^T.$$

Problem 10:

$$F_i(x) = (0.5 - x_i)^2 + x_{n+1-i}^2 - 0.25x_i - 1,$$
$$i = 1, 2, \ldots, n.$$
$$x_0 = (0.1, 0.1, \ldots, 0.1)^T.$$

The numerical results of the two(2) methods are reported in Table 1, 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.

Considering the numerical experiment recorded in Table 1, the hybrid methods MDF1 and MDF2 gives better results compared to the original method EMFD and the other TDS method considering the two performance criteria, the, number of iterations and CPU time. We have seen that our proposed methods outperform the TDS and EMFD methods as they solves all the problems while EMFD method fails to solve Problem 1 with 1000 dimension, Problems 3,8 and 9 with all the dimensions tested, the TDS method also fails to solve problems 2 and 10 completely. An observation shows the better performance of MDF2 method as it outperforms the MDF1 method in most of the problems which is as a result of updating the parameter β_k in every iteration. MDF1 method fixed a value for the parameter β_k in every iteration. Summary of the numerical results from Table 2 shows clearly that the MDF2 method is superior in number of iterations as it solves 53.3% of the problems, MDF1 has 30%, EMFD has 3.3% and TDS method has 0%. Figures 1 and 2 gives the better performance of our method related to the number of iterations and CPU time, which were evaluated using the profiles of Dolan and More [30]. That is, for each method, we plot the fraction

Table 1. The numerical results of MDF1, MDF2, EMFD and TDS for Problems 1–10.

		MDF1			MDF2			EMFD			TDS	
Problems	Dim	NI	CPU	$\|F(x_k)\|$	NI	CPU	$\|F(x_k)\|$	NI	CPU	$\|F(x_k)\|$	NI	CPU	$\|F(x_k)\|$
1	1000	9	0.047634	8.72E−05	7	0.032763	8.52E−05	_	_	_	13	0.114019	5.97E−05
	10000	9	0.223298	5.84E−05	8	0.190146	8.61E−05	240	5.050164	9.99E−05	13	0.477615	7.50E−05
	100000	9	2.421766	5.84E−05	5	1.219029	8.54E−05	2	0.498384	7.23E−05	12	3.747139	9.94E−05
2	1000	8	0.02876	2.63E−06	6	0.021293	7.06E−07	18	0.071541	7.50E−05	_	_	_
	10000	8	0.172964	2.79E−05	6	0.120879	7.97E−06	19	0.567301	9.61E−05	_	_	_
	100000	9	2.163286	7.57E−06	6	1.280159	8.11E−05	21	7.863988	4.87E−05	_	_	_
3	1000	13	0.066967	4.74E−05	10	0.024316	2.15E−05	_	_	_	22	0.039729	7.52E−05
	10000	14	0.448677	9.18E−05	11	0.214572	4.04E−05	_	_	_	24	0.183432	8.56E−05
	100000	16	5.989696	3.44E−05	12	2.489935	7.60E−05	_	_	_	26	2.123003	9.74E−05
4	1000	6	0.025462	7.50E−06	7	0.027143	1.42E−05	12	0.046199	4.79E−05	12	0.018007	8.75E−05
	10000	6	0.134201	7.50E−05	8	0.183082	1.11E−05	13	0.352261	3.03E−05	14	0.120614	4.43E−05
	100000	7	1.906209	1.08E−05	9	2.54147	8.71E−06	13	4.080707	9.58E−05	15	1.232585	5.60E−05
5	1000	6	0.02744	2.91E−06	4	0.025983	6.66E−06	9	0.033801	4.00E−05	14	0.02704	5.68E−05
	10000	6	0.167957	2.88E−05	4	0.107692	6.44E−05	10	0.298398	2.53E−05	15	0.154354	7.19E−05
	100000	7	1.941717	4.08E−06	5	1.24497	1.06E−06	10	3.193021	8.01E−05	16	1.551129	9.11E−05
6	1000	10	0.049556	4.53E−05	13	0.058171	4.20E−05	29	0.157358	9.05E−05	30	0.045294	8.53E−05
	10000	11	0.331908	2.50E−05	13	0.398135	7.46E−05	29	1.172564	9.30E−05	28	0.238698	6.99E−05
	100000	12	4.747427	2.01E−05	13	5.233994	8.06E−05	29	15.51903	9.30E−05	30	3.207395	6.74E−05
7	1000	16	0.270655	6.50E−05	15	0.254221	9.33E−05	27	0.815773	4.95E−05	46	0.105632	8.12E−05
	10000	16	1.855052	9.36E−05	16	1.887946	8.06E−05	27	6.219932	9.18E−05	46	0.778372	8.46E−05
	100000	17	25.35916	4.40E−05	17	26.69653	8.00E−05	28	68.28999	7.98E−05	46	10.43807	9.34E−05
8	1000	9	0.245363	2.70E−05	7	0.037193	4.80E−05	_	_	_	21	0.038252	9.95E−05
	10000	9	0.245363	2.70E−05	9	0.245132	3.51E−05	_	_	_	24	0.223715	6.79E−05
	100000	10	2.944476	5.22E−05	10	3.149003	9.50E−05	_	_	_	26	2.330902	7.73E−05
9	1000	8	0.033651	2.22E−05	6	0.018856	3.80E−05	_	_	_	23	0.020676	6.77E−05
	10000	9	0.153381	4.31E−05	8	0.138424	2.77E−05	_	_	_	25	0.145547	7.71E−05
	100000	10	1.886577	8.34E−05	9	1.626121	7.50E−05	_	_	_	27	1.483078	8.78E−05
10	1000	8	0.030028	4.62E−05	10	0.048343	4.00E−05	19	0.093766	9.70E−05	_	_	_
	10000	9	0.218861	2.45E−05	11	0.284328	4.62E−05	21	0.722959	6.21E−05	_	_	_
	100000	10	2.886788	1.29E−05	12	3.746113	5.35E−05	22	9.368962	8.84E−05	_	_	_

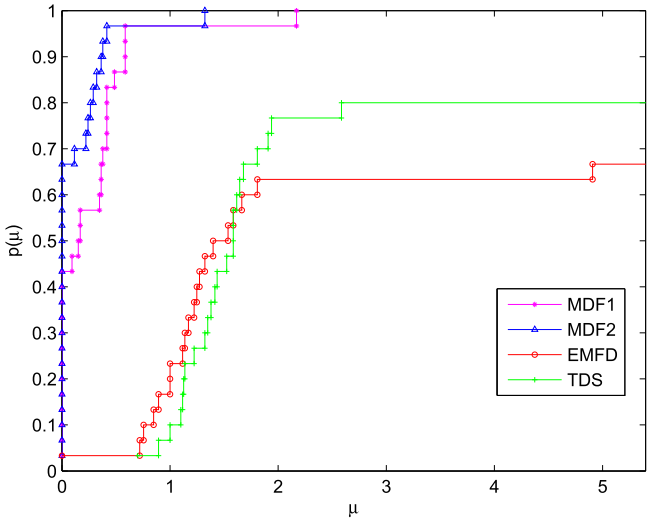

Figure 1. Performance profile of MDF1, MDF2, TDS and EMFD methods with respect to the number of iteration for the Problems 1–10.

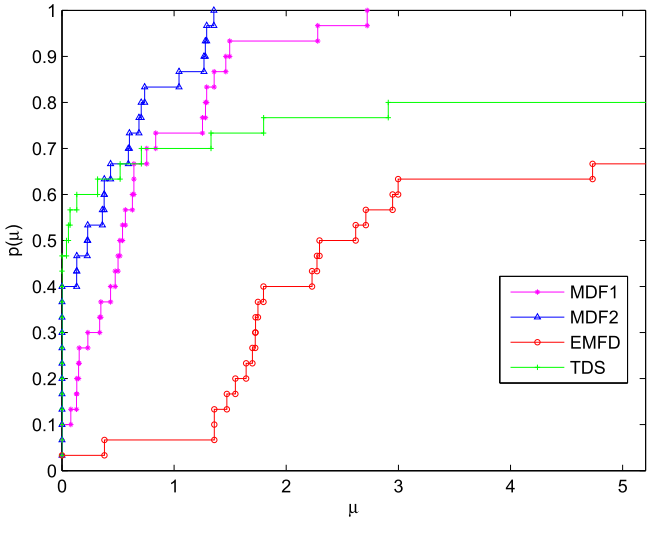

Figure 2. Performance profile of MDF1, MDF2, TDS and EMFD methods with respect to the CPU time for the Problems 1–10.

$p(\mu)$ of the problems for which the method is within a factor μ of the best time. The top curve is the method that solved most problems in a time that was within a factor μ of the best time. We observed from Figure 1 that the curve above all other curves is that of MDF2 method, so its recorded as the method with the best performance in terms of number of iterations. Next is the curve representing the MDF2 scheme, which is above the other curves for the other methods and also Figure 1 shows that the MDF1 method is more efficient than the TDS and EMFD methods. From Figure 2, it is observed that TDS method performs better than all

Table 2. Summary of results from Table 1 for MDF1, MDF2, EMFD and TDS methods.

	Method	NI	Percentage	CPU time	Percentage
Number of problems and	MDF1	9	30%	3	10%
percentage for each method with	MDF2	16	53.3%	12	40%
respect to iterations and CPU time.	EMFD	1	3.3%	1	0.3%
	TDS	0	0%	14	46.7%
	Undecided	4	13.3%	0	0%

the method in terms of CPU time, which may be as a result of less parameters in the direction of the TDS method (Table 2).

We further explain the accuracy of the reported results by taking the average of the norm of the residuals recorded at the stopping point for each of the four methods, where MDF1 has 4.77×10^{-6}, MDF2 has 4.60×10^{-6}, TDS has 6.20×10^{-6}, and EMFD 4.85×10^{-5}. This clearly shows that our proposed methods converge faster to the solution than the other methods.

5. Conclusion

In this research, we present some hybrid methods for solving large-scale system of nonlinear equations by hybridizing the Picard–Mann hybrid process with En enhanced matrix-free method. We compared the performance of the hybrid methods with that of a transformed double step length method (TDS) for solving large-scale systems of nonlinear equations and En enhanced matrix-free method via double step length approach for solving systems of nonlinear equations (EMFD), by doing some intensive numerical experiments through solving different benchmark test problems, better performances of the proposed methods were observed regarding all three tested characteristics, which showed how the correction parameter improved the methods, we therefore proved that the proposed methods, i.e MDF1 and MDF2 converge faster and more efficient compared to TDS and EMFD methods, respectively, The correction parameter is the essential element in the proposed methods, so therefore, the choice of β_k affected the numerical performances of the proposed methods. In the future research, the proposed approaches will be used to solve monotone nonlinear equations with application in compressive sensing.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

[1] Dennis JE, Schnabel RB. Numerical methods for unconstrained optimization and nonlinear equations. Prentice hall, Englewood Cliffs, NJ. 1983.

[2] Waziri MY, Leong WJ, Hassan MA, et al. Jacobian computation-free Newton method for systems of non-linear equations. J Numer Math Stochast. 2010;2:54–63.

[3] Waziri MY, Majid ZA. An enhanced matrix-free secant mehod via predictor-corrector modified line search strategies for solving systems of nonlinear equations. J Math Math Sci. 2013. Article ID 814587: 6 pp. doi.org/10.1155/2013/814587.
[4] Zang L, Zhou W, Li DH. Global convergence of modified Fletcher-Reeves conjugate gradient method with Armijo-type line search. Numerische Mathematik. 2005;164(1):277–289.
[5] Halilu AS, Waziri MY. En enhanced matrix-free method via double steplength approach for solving systems of nonlinear equations. Inter J Appl Math Res. 2017;6(4):147–156. doi.org/10.14419/ijamr.v6i4.8072.
[6] Gonglin Y, Xiwen L. A new backtracking inexact BFGS method for symmetric nonlinear equations. J Comput Math Appl. 2008;55:116–129.
[7] Halilu AS, Waziri MY. An improved derivative-free method via double direction approach for solving systems of nonlinear equations. J Ramanujan Math Soc. 2018;33(1):75–89.
[8] Petrović MJ, Stanimirović PS. Accelerated double direction method for solving unconstrained optimization problems. Math Probl Eng. 2014. Article ID 965104: 8 pp. doi.org/10.1155/2014/965104.
[9] Li D, Fukushima M. A global and superlinear convergent Gauss-Newton based BFGS method for symmetric nonlinear equation. SIAM J Numer Anal. 1999;37: 152–172.
[10] Petrović MJ. An accelerated double step size model in unconstrained optimization. Appl Math Comput. 2015;250:309–319. doi.org/10.1016/j.amc.2014.10.104.
[11] Petrović MJ, Stanimirović PS, Kontrec N, et al. Hybrid modification of accelerated double direction method. Math Probl Eng. 2018. Article D 1523267: 8 pp. doi.org/10.1155/2018/1523267.
[12] Petrović MJ. Hybridization rule applied on accelerated double step size optimization scheme. Filomat. 2019;33(3):655–665. doi.org/10.2298/FIL1903655P.
[13] Petrović MJ, Rakoević V, Valjarević D, et al. A note on hybridization process applied on transformed double step size model. Numer Algor. 2019. doi.org/10.1007/s11075-019-00821-8.
[14] Panić S, Petrović MJ, Carević MM. Initial improvement of the hybrid accelerated gradient descent process. Bull Aust Math Soc. 2018;98(2):331–338. doi.org/10.1017/S0004972718000552.
[15] Safeer HK. A Picard-Mann hybrid iterative process. Khan Fixed Point Theory Appl. 2013;69:10.
[16] Andrei N. A hybrid conjugate gradient algorithm for unconstrained optimization as a convex combination of Hestenes-Stiefel and Dai-Yuan. Stud Inform Control. 2008;17(4):55–70.
[17] Babaie-Kafaki S, Ghanbari SR. Two hybrid nonlinear conjugate gradient methods based on a modified secant equation. Optimization. 2014;63(7):1027–1042.
[18] Li JK, Liu SJ. New hybrid conjugate gradient method for unconstrained optimization. Appl Math Comput. 2014;245:36–43.
[19] Petrović M, Rakočević V, Kontrec N, et al. Hybridization of accelerated gradient descent method. Numer Algor. 2017;79(3):769–786. doi.org/10.1007/s11075-017-0460-4.
[20] Stanimirović PS, Miladinović MB. Accelerated gradient descent methods with line search. Numer Algor. 2010;54:503–520. doi.org/10.1007/s11075-009-9350-8.
[21] Halilu AS, Waziri MY, Musa YB. Inexact double step length method for solving systems of nonlinear equations. Stat Optim Inf Comput. 2020;8:165–174. doi:10.19139/soic-2310-5070-532.

[22] Halilu AS, Waziri MY, Yusuf I. Efficient matrix-free direction method with line search for solving large-scale system of nonlinear equations. Yugoslav J Oper Res. doi.org/10.2298/YJOR160515005H.
[23] Barzilai J, Borwein JM. Two point step size gradient method. IMA J Numer Anal. 1988;8(1):141–148. doi.org/10.1093/imanum/8.1.141.
[24] Halilu AS, Waziri MY. A transformed double steplength method for for solving large-scale systems of nonlinear equations. J Numer Math Stochastics. 2017;9(1):20–23.
[25] Waziri MY, Ahmed K, Sabi'u J. A family of Hager–Zhang conjugate gradient methods for system of monotone nonlinear equations. Appl Math Comput. 2019;361:645–660.
[26] La Cruz W, Martinez JM, Raydan M. spetral residual method without gradient information for solving large-scale nonlinear systems of equations: theory and experiments. P Optim. 2006;75(225):1429–1448.
[27] Abubakar AB. On improved Broyden-type method for systems of nonlinear equations [M.Sc. Thesis BUk]. 2014. 1–54.
[28] Halilu AS, Dauda MK, Waziri My, et al. A derivative-free decent method via acceleration parameter for solving systems of nonlinear equations . Open J sci. tech. 2019;2(3):1–4.
[29] Stanimirović PS, Milovanović GV, Petrović MJ, et al. A transformation of accelerated double step size method for unconstrained optimization. Math Probl Eng. 2015; Article ID 283679: 8 pp. doi.org/10.1155/2015/283679.
[30] Dolan E, More J. Benchmarking optimization software with performance profiles. J Math Program. 2002;91(2):201–213.
[31] Waziri MY, Sabiu J. A derivative-free conjugate gradient method and its global convergence for symmetric nonlinear equations. J Math Mathematical Sc. 2015; Article ID 961487: 8 pp.