Journal of Numerical Mathematics and Stochastics, **10** (1) : 32-44, 2018 © ***JNM@S***
http://www.jnmas.org/jnmas10-3.pdf Euclidean Press, LLC
Online: ISSN 2151-2302

A Modified Conjugate Gradient Method Via a Double Direction Approach for Solving Large-Scale Symmetric Nonlinear Equations

H. ABDULLAHI[1], **A. S. HALILU**[1], **and M. Y. WAZIRI**[2]

[1]Department of Mathematics and Computer Sciences, Sule Lamido University, Kafin Hausa, Jigawa, Nigeria;[2]Department of Mathematical Sciences, Bayero University, Kano, Nigeria,
E-mail: abubakarsaddiqu@gmail.com

Abstract. *We propose a variant of the conjugate gradient method for solving large-scale systems of symmetric nonlinear equations, in which the spectral secant condition and conjugate gradient ideas are combined. Using an approximate norm descent, we show that the proposed method has global convergence properties. Under appropriate conditions, numerical results for benchmark test functions, demonstrate an improved efficiency of this method over other existing alternatives.*

Key words : Double Direction, Conjugate Gradient, Line Search, Nonlinear Equations.

AMS Subject Classifications : 65H11, 65K05, 65H12, 65H18

1. Introduction

Consider the following nonlinear system of equations:

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

where $F : \mathsf{R}^n \to \mathsf{R}^n$ is a nonlinear, continuous mapping and is assumed to satisfy the assumptions that follow.

Assumption 1.
($A1$) There exists $x^* \in \mathsf{R}^n$ such that $F(x^*) = 0$.
($A2$) F is a continuously differentiable mapping.
($A3$) The Jacobian is symmetric.

The renowned method for finding the solution to (1) is the Newton's method. The method is simple to implement, and generates an iterative sequences $\{x_k\}$ from a given initial guess x_0 in a neighborhood of x^* via

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

where $k = 0, 1, 2, \ldots$ and $F'(x_k)$ is the Jacobian matrix.

A quasi-Newton's method is another variant of Newton-type methods that replaces the Jacobian or its inverse, with an approximation which can be updated at each iteration [12]. Its updating scheme is given by

$$x_{k+1} = x_k - B_k^{-1} F(x_k), \tag{3}$$

where B_k is the approximation of the Jacobian at x_k and it is updated at each iteration. There are many types of quasi-Newton methods, but the most efficient and successful ones are Broyden, Fletcher, Goldfarb and Shanno methods, denoted as BFGS [13], where the BFGS update is defined as:

$$B_{k+1} = B_k - \frac{B_k\, s_k\, s_k^T\, B_k}{s_k^T\, B_k\, s_k} + \frac{y_k\, y_k^T}{y_k^T\, s_k}, \tag{4}$$

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

The rationale behind quasi-Newton methods is to do with the evaluation cost of the Jacobian matrix [12, 8]. Steepest descent is also among the earliest methods for solving nonlinear systems of equations, but is rarely used due to its slow convergence rate. Understanding the convergence properties of this method can, nevertheless, lead to a better understanding of many more sophisticated optimization methods [3, 6, 10].

Based on this fact, the well-known conjugate gradient method is a promising development for large-scale unconstrained optimization problems due to its simplicity and low storage requirement [1, 5]. The nonlinear conjugate gradient method came into existence in 1964, by Fletcher and Reeves [1]. Since then the work on conjugate gradients became quite noticed in the literature. Recently, the conjugate gradient metod has been extended to solve large-scale nonlinear systems of equations [12].

Furthermore, also in recent years, much effort has been placed on developing and constructing a new and simple formula for the conjugate gradient parameters to make the method easier to apply to various other fields, with good numerical performance and global convergence properties. Considerations like this, have led to the so-called Hybrid methods, which as the name suggests, are based on two different methods. One of which has good performance at the initial stage, like the gradient methods, and the other has good performance at the final stage, like Newton's methods [13].

The most frequently used line search in practice is the inexact line search [5, 6, 8], which is chosen in such a way that the function values along the ray $x_k + \alpha_k\, d_k,\ \alpha_k > 0$ decreases, i.e.,

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

Most of the approaches to solving systems of nonlinear equations are single direction methods. One of the advantages, however, of the double direction method over single direction is the presence of two direction vectors which work jointly as a correction factor towards boosting the convergence of the system. The presence of the two direction vectors help the system immensely in such a way that if one of the directions fails, the other direction will correct the system automatically [3]. Whereas in a single direction scheme, if the direction fails the whole process fails. To improve the effectiveness of the algorithm for solving such systems of nonlinear equations, Halilu & Waziri, [7], presented a double direction iterative scheme given by

$$x_{k+1} = x_k + \alpha_k c_k + \alpha_k^2 d_k , \tag{6}$$

where x_{k+1} represents a new iterative point, x_k is the previous iterative point, α_k is the step length, while c_k and d_k are the search directions respectively.

Furthermore (1) can emerge either from an unconstrained optimization problem, a saddle point problem, or from an equality constrained problem [8]. Let f be a norm function defined by

$$f(x) = \frac{1}{2}\|F(x)\|^2. \tag{7}$$

where $f : \mathbb{R}^n \to \mathbb{R}$. The nonlinear equations (1) is equivalent to the following global optimization problem

$$\min f(x),\ x \in \mathbb{R}^n. \tag{8}$$

Our paper is organized as follows. In the next section, we present the proposed method. The convergence results are presented in section 3, while some numerical results are presented in section 4. Finally, a conclusion is provided in section 5.

2. Derivation of the Method

In this section, we introduce the two direction vectors in (6). In order to incorporate more information of the iterates at each iteration and to guarantee rapid convergence towards the solution, we suggest the first direction c_k, to be defined as

$$c_k = -F(x_k), \tag{9}$$

while the second direction d_k is to be derived by combining the direction presented in [14] with the classical Newton's direction. It should be noted that the direction presented in [14] is defined as

$$d_k = \begin{cases} -F(x_k), \text{ if } k = 0, \\ -F(x_k) + \beta_k^{PRP} d_{k-1} - v_k y_k , \text{ if } k \geq 1, \end{cases} \tag{10}$$

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

$$\beta_k^{PRP} = \frac{F^T(x_{k+1}) y_k}{\|F(x_k)\|^2} , \tag{11}$$

and

$$v_k = \frac{F^T(x_{k+1})\, d_{k+1}}{\|F(x_k)\|^2} . \tag{12}$$

However, the Newton's direction, d_k, is given by

$$d_k = -J^{-1}(x_k) F(x_k), \tag{13}$$

where $J(x_k)$ is the Jacobian matrix.

Combining (10) and (13), we have

$$-J^{-1}(x_k) F(x_k) = -F(x_k) + \beta_k\, d_{k-1} - v_k\, y_k . \tag{14}$$

Then multiplying both sides of (14) by $J(x_k)$ leads to

$$-F(x_k) = -J(x_k) F x_k) + \beta_k\, J(x_k)\, d_{k-1} - v_k\, J(x_k)\, y_k . \tag{15}$$

From the spectral secant condition used in [9], we have

$$J(x_k)\, S_k = \theta_k\, y_k , \tag{16}$$

where, $s_k = x_{k+1} - x_k = J^{-1}(x_k)\theta_k\ y_k$, and $y_k = F(x_{k+1}) - F(x_k)$, which after transposition becomes

$$s_k^T J^T(x_k) = \theta_k y_k^T . \tag{17}$$

It is vital to note that, for this work, we claim that the Jacobian matrix is symmetric, $\forall k$. So we accordingly have

$$s_k^T J(x_k) = \theta_k y_k^T. \tag{18}$$

Pre-multiplication of both sides of (15) by s_k^T and substitution of (18) in the result leads to:

$$-s_k^T F(x_k) = -\theta_k y_k^T F(x_k) + \theta_k y_k^T \beta_k d_{k-1} - \theta_k v_k y_k^T y_k. \tag{19}$$

After a slight simplification in (19), our CG parameter(β_k) becomes

$$\beta_k = \frac{(\theta_k y_k - s_k)^T F(x_k) + \theta_k v_k \| y_k\|^2}{\theta_k y_k^T d_k} , \tag{20}$$

where

$$\theta_k = \frac{s_k^T s_k}{s_k^T y_k},$$

see [12]. Having derived, in (20), the new CG parameter β_k we present our direction using the frame of the conjugate gradient method in the form

$$d_k = \begin{cases} -F(x_k), \text{ if } k = 0, \\ -F(x_k) + \beta_k d_{k-1} - v_k y_k , \text{ if } k \geq 1, \end{cases} \tag{21}$$

where β_k is defined in (20). Further substitution of (9) and (21) into the iterative procedure given by (6), builds our scheme viz

$$x_{k+1} = x_k + \alpha_k c_k + \alpha_k^2 d_k. \tag{22}$$

By this (22), the first direction is with a descent nature, but the second direction may not be of descent nature, as such. An approximate norm descent line search described in [8], is used to compute the step length α_k via : let $\omega_1, \omega_2, r \in (0,1)$ be constants, and let η_k be a given positive sequence such that

$$\sum_{k=0}^{\infty} \eta_k < \eta < \infty, \tag{23}$$

and

$$f(x_k - \alpha_k F(x_k) + \alpha_k^2 d_k) - f(x_k) \leq -\omega_1 \| \alpha_k F(x_k) \|^2 - \omega_2 \| \alpha k dk \|^2 + \eta_k f(x_k). \tag{24}$$

Furthermore, let i_k be the smallest nonnegative integer i such that (24) holds for $\alpha = r^i$, and $\alpha_k = r^{i_k}$.

Now we present the algorithm of the proposed method as follows.

Algorithm 1. (MCGD)
STEP 1: Given $x_0, \epsilon = 10^{-4}, d_0 = -F(x_0)$, set $k = 0$.
STEP 2: Compute $F(x_k)$.
STEP 3: If $\| F(x_k) \| \leq \epsilon$, then stop.
STEP 4: Compute step length α_k (by (24)).
STEP 5: Set $x_{k+1} = x_k - \alpha_k F(x_k) + \alpha_k^2 d_k$.
STEP 6: Compute $F(x_{k+1})$.
STEP 7: Compute β_k (using (20)).
STEP 8: Update d_{k+1}(using (21)).
STEP 9: Set $k = k + 1$, and go to STEP 3.

3. Convergence Analysis

Our interest here is in the global convergence of our proposed method (MCGD). To begin with, let Ω be the level set defined by:

$$\Omega = \left\{ x \in \mathsf{R}^n \mid \| F(x) \| \leq \sqrt{e^{\eta}} \| F(x_0) \| \right\}, \tag{25}$$

where η is a positive constant such that (23) is satisfied. Here, we can easily see that the level set Ω is bounded. In order to analyze the global convergence of the MCGD algorithm, we need the following assumption.

Assumption 2.

In some neighborhood N of Ω the nonlinear function F is Lipschitz continuous i.e., there exists a positive constant $L > 0$, such that

$$\| F(x) - F(y) \| \leq L \| x - y \|, \ \forall x, y \in N. \tag{26}$$

From the level set, there exists a positive constant $M_1 > 0$, such that

$$\| F(x) \| \leq M_1, \forall x \in \Omega. \tag{27}$$

Lemma 3.1. *Let $\{x_n\}$ be a sequence generated by the MCGD algorithm. Then $\{x_n\} \subset \Omega$.*

Proof. From (24) we have $\forall_k$,

$$\begin{aligned} f(x_{k+1}) &\leq (1 + \eta_k) f(x_k) \\ &\leq f(x_0) \prod_{i=0}^{k} (1 + \eta_i) \\ &\leq f(x_0) \left[\frac{1}{k+1} \sum_{i=0}^{k} (1 + \eta_i) \right]^{k+1} \\ &\leq f(x_0) \left[1 + \frac{1}{k+1} \sum_{i=0}^{k} \eta_i \right]^{k+1} \\ &\leq f(x_0) \left(1 + \frac{\eta}{k+1} \right)^{k+1} \leq e^{\eta} f(x_0). \end{aligned}$$

Thus we have,

$$\| F(x_{k+1}) \| \leq \sqrt{e^{\eta}} \| F(x_0) \|. \tag{28}$$

Then we can see that from (28) that $\{x_n\} \subset \Omega$. ■

Lemma 3.2. *Suppose that the assumption 2 holds and x_k is generated by the MCGD algorithm, then we have*

$$\lim_{k \to \infty} \| \alpha_k d_k \|^2 = 0, \tag{29}$$

and

$$\lim_{k \to \infty} \| \alpha_k F(x_k) \|^2 = 0. \tag{30}$$

Proof. From (24) and for all $k > 0$,

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

Then by summing the above inequality, we have,

$$\omega_2 \sum_{i=0}^{k} \| \alpha_i d_i \|^2 \leq \sum_{i=0}^{k} \left(\| F(x_i) \|^2 - \| F(x_{i+1}) \|^2 \right) + \sum_{i=0}^{k} \eta_i \| F(x_i) \|^2$$

$$= \left(\| F(x_0) \|^2 - \| F(x_{k+1}) \|^2 \right) + \sum_{i=0}^{k} \eta_i \| F(x_i) \|^2$$

$$\leq \| F(x_0) \|^2 + M_1^2 \sum_{i=0}^{k} \eta_i$$

$$\leq \| F(x_0) \|^2 + M_1^2 \sum_{i=0}^{\infty} \eta_i . \quad (32)$$

Hence, from (27) and by the fact that $\{\eta_k\}$ satisfies (23), the series $\sum_{i=0}^{\infty} \| \alpha_i d_i \|^2$ is convergent. This implies (29). By a similar argument, with $\omega_1 \| \alpha_k F(x_k) \|^2$ at the left hand side,we can show that (30) holds. ■

Lemma 3.3. *Let the level set Ω be bounded and $\{x_k\}$ be generated by the algorithm 1, then we have*

$$\sum_{k=1}^{\infty} \| s_k \| < \infty . \quad (33)$$

Proof. The inequality (24) can easily be rewritten as, we have,

$$\| s_k \| + \| F(x_{k+1}) \|^2$$
$$\leq \| F(x_k) \|^2 - \omega_1 \| \alpha_k d_k \|^2 - \omega_2 \| \alpha_k F(x_k) \| + \eta_k \| F(x_k) \|^2 + \| s_k \|$$
$$\leq \| F(x_k) \|^2 - \| F(x_{k+1}) \|^2 + \eta_k \| F(x_k) \|^2 + \| s_k \|. \quad (34)$$

Then by summing the above inequality, we have,

$$\sum_{i=0}^{k} \| s_i \| \leq \sum_{i=0}^{k} \left(\| F(x_i) \|^2 - \| F(x_{i+1}) \|^2 \right) + \sum_{i=0}^{k} \| s_i \| + \sum_{i=0}^{k} \eta_i \| F(x_i) \|^2$$

$$= \left(\| F(x_0) \|^2 - \| F(x_{k+1}) \|^2 \right) + \sum_{i=0}^{k} \| s_i \| + \sum_{i=0}^{k} \eta_i \| F(x_i) \|^2$$

$$\leq \| F(x_0) \|^2 + \sum_{i=0}^{k} \| \alpha_i^2 d_i \| + \sum_{i=0}^{k} \| \alpha_i F(x_i) \| + \sum_{i=0}^{k} \eta_i \| F(x_i) \|^2$$

$$\leq \| F(x_0) \|^2 + \sum_{i=0}^{\infty} \| \alpha_i^2 d_i \| + \sum_{i=0}^{\infty} \| \alpha_i F(x_i) \| + M_1^2 \sum_{i=0}^{\infty} \eta_i. \tag{35}$$

Hence by (27), (29), (30) and (23), $\sum_{i=0}^{k} \| s_i \|$ is convergent and hence the proof completes.■

Lemma 3.4. *Suppose that assumption 2 holds, and let* $\{x_k\}$ *be generated by the MCGD algorithm. If there is a constant* $\epsilon > 0$ *such that for all k,*

$$\| F(x_k) \| \geq \epsilon, \tag{36}$$

then there exists a constant $M > 0$ *such that for all k,*

$$\| d_k \| \leq M. \tag{37}$$

Proof. By (26), (29) and (30), we can write

$$\| y_k \| = \| F(x_{k+1}) - F(x_k) \| \leq L \| x_{k+1} - x_k \| = L \| \alpha_k^2 d_k - \alpha_k F(x_k) \|$$

$$\leq L\left(\| \alpha_k^2 d_k \| + \| \alpha_k F(x_k) \| \right) \to 0. \tag{38}$$

Furthermore,

$$\| (y_k - s_k)^T F(x_k) \| = \| y_k - s_k \| \| F(x_k) \| \leq M_1(\| y_k \| + \| s_k \|), \tag{39}$$

which is bounded by (33) and (27). Hence

$$|\beta_k| \leq \frac{\|y_k - s_k\| \|F(x_k)\| + |v_k| \|y_k\|^2}{|y^T d_{k-1}|} \to 0. \tag{40}$$

This implies that there exists a constant $\rho \in (0,1)$ such that for sufficiently large k,

$$|\beta_k| \leq \rho. \tag{41}$$

Now using

$$\| d_k \| \leq \| F(x_k) \| + |\beta_k| \| d_{k-1} \| - |v_k| \| y_k \|, \tag{42}$$

and setting

$$M_3 = max\left\{ \| d_1 \|, \| d_2 \|, \ldots, \| d_{(k_0)} \|, \frac{M_1}{1 - \epsilon_0} \right\}, \tag{43}$$

we can deduce that for all k, (37) holds, i.e.,$\| d_k \|$ is uniformly bounded. ■

Now we are going to establish the following global convergence theorem to show that under some suitable conditions, there exists an accumulation point of $\{x_k\}$ which is the solution of (2).

Theorem 3.1. *Suppose that assumption 2 holds, and* $\{x_k\}$ *is generated by the MCGD algorithm. Assume further that for all* $k > 0$*,*

$$\alpha_k \geq c \frac{|F^T(x_k) d_k|}{\|d_k\|^2}, \tag{44}$$

where c is some positive constant, then

$$\lim_{k\to\infty} \| F(x_k) \| = 0. \tag{45}$$

Proof. By lemma 3.4, relation (37) should hold. Therefore by (29) and the boundedness of $\{\|d_k\|\}$, we have

$$\lim_{k\to\infty} \alpha_k \| d_k \|^2 = 0. \tag{46}$$

Then by (44) and (46), we obtain

$$\lim_{k\to\infty} |F_k^T d_k| = 0. \tag{47}$$

On the other hand, from (21), it follows that

$$F^T(x_k)\, d_k = - \| F(x_k) \|^2 + \beta_k F^T(x_k)\, d_{k-1} - F^T(x_k) v_k\, y_{k-1},$$

which can be written as

$$\| F(x_k) \| \leq |F^T(x_k) d_k| + |\beta_k| \| F(x_k) \| \| d_{k-1} \| + \| F(x_k) \| \| v_k \| \| y_k \|. \tag{48}$$

So that by Equation (27), (37), (41), and by taking the limit of the above inequality, we have

$$\lim_{k\to\infty} \| F(x_k) \| = 0, \tag{49}$$

which is the required result. And hence the proof completes. ■

4. Numerical Results

The performance of the proposed MCGD method for solving nonlinear equations (1) is compared in this section, with a double direction conjugate gradient method for solving large-scale system of nonlinear equations (DDLS) of [2].

For both methods we set the following parameters:

$\omega_1 = \omega_2 = 10^{-4}$, $r = 0.3$ and $\eta_k = 1/(k+1)^3$.

The employed computational codes were written by Matlab (R2013a) and run on a personal computer 2.10 GHZ CPU processor and 2.00 GB RAM memory. We stop the iterations if the total number of iterations exceeds 1000 or $\| F(x_k) \| \leq 10^{-4}$. Ten test problems from different sources, with dimension between 100 to 10,000, have been considered , with different initial points, which are not restricted to a point that is too close to the solution.

Problem [6] 1:

$$F(x) = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix} x + (e_1^x - 1, \ldots, e_n^x - 1)^T. \quad x_0 = (0.5, 0.5, 0.5, \ldots, 0.5)^T.$$

Problem [6] 2:

$$F(x) = \begin{pmatrix} 2 & -1 & & & \\ 0 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix} x + (sinx_1 - 1, \ldots, sinx_n - 1)^T. \quad x_0 = (0.5, 0.5, 0.5, \ldots, 0.5)$$

Problem [11] 3:
$F_i(x) = x_i^2 - 1,$
$i = 1, 2, 3, \ldots, n,$
$x_0 = (-0.001, -0.001, -0.001, \ldots, -0.001)^T.$

Problem [11] 4:
$F_i(x) = x_i^2 + x_i - 2,$
$i = 1, 2, 3, \ldots, n,$
$x_0 = (-0.5, -0.5, -0.5, \ldots, -0.5)^T$

Problem [11] 5:

$$F_i(x) = x_i - \left[1 \Big/ n^2 \left(\sum_{i=1}^{n} x_i \right)^2 \right] + \left(\sum_{i=1}^{n} x_i \right) - n,$$

$i = 1, 2, 3, \ldots, n,$
$x_0 = (5,\ 5,\ 5, \ldots, 5)^T.$

Problem [11] 6:

$$F_i(x) = \left(\sum_{i=1}^{n} x_i + i \right)(x_i - 1) + e^{\,x_i} - 1,$$

$i = 1, 2, 3, \ldots, n,$
$x_0 = (0.3, 0.3, 0.3, \ldots, 0.3)^T.$

Problem [11] 7:
$F_1(x) = 3x_1^3 + 2x_2 - 5 + \sin(x_1 - x_2)\sin(x_1 + x_2),$
$F_i(x) = -x_{i-1}\, e^{(x_{i-1} - x_i)} + x_i(4 + 3x_i^2) + 2x_{i+1} + \sin(x_i - x_{i+1})\sin(x_i + x_{i+1}) - 8,$
$F_n(x) = -x_{n-1}\, e^{(x_{n-1} - x_n)} + 4x_n - 3,$
$i = 2, 3, 4, \ldots, n - 1,$
$x_0 = (0, 0, 0, \ldots, 0)^T.$

Problem [11] 8:
$F_i(x) = x_i^2 - \cos(x_i - 1),$
$i = 1, 2, 3 \ldots, n,$
$x_0 = (2,2,2...,2)^T.$

Problem [11] 9:
$F_1(x) = 3x_1^3 + 2x_2 - 5 + \sin(x_1 - x_2)\sin(x_1 + x_2)$,
$F_i(x) = -x_{i-1}\, e^{(x_{i-1} - x_i)} + x_i(4 + 3x_i^2) + 2x_{i+1} + \sin(x_i - x_{i+1})\sin(x_i + x_{i+1}) - 8$,
$F_n(x) = -x_{n-1}\, e^{(x_{n-1} - x_n)} + 4x_n - 3$,
$i = 2, 3, 4, \ldots, n-1$,
$x_0 = (1, 1, 1, \ldots, 1)^T$.

Problem [11] 10:
$F_i(x) = 5x_i^2 - 2x_i - 3$,
$i = 1, 2, 3, \ldots, n$,
$x_0 = (-0.01, -0.01, -0.01, \ldots, -0.01)^T$.

The numerical results for the two methods, mentioned earliear, are reported in Table 1, where "NI" and "Time" stand for the total number of all iterations and the CPU time in seconds, respectively. $\| F(x_k) \|$ is the norm of the residual at the stopping point. We claim that the method fails, and use the symbol "-", when the number of iterations is greater than or equal to 1×103, but no x_k satisfying $\| F(x_k) \| \leq 10^{-4}$ is reached.

Table 1: Numerical results for the MCGD and DDLS methods for problems 1 to 10.

		MCGD			DDLS		
Problems	Dim	NI	Time	$\|F(x_k)\|$	NI	CPU	$\|F(x_k)\|$
1	10	22	0.175664	6.56E-05	24	0.186838	9.51E-05
	1000	24	0.943085	7.43E-05	25	0.983593	6.20E-05
	5000	26	11.92271	6.14E-05	26	12.35829	7.89E-05
2	10	13	0.106446	6.71E-05	23	0.191258	9.17E-05
	1000	13	0.677609	5.33E-05	38	1.503056	8.03E-05
	5000	12	7.389241	6.78E-05	39	22.77284	8.96E-05
3	10	10	0.070472	5.78E-05	14	0.004965	9.52E-05
	1000	11	0.007959	4.02E-05	16	0.010032	5.56E-05
	10000	12	0.063938	2.80E-05	17	0.079654	7.55E-05
4	10	7	0.00375	8.17E-05	10	0.001772	1.31E-05
	1000	8	0.007194	4.39E-05	10	0.009093	4.15E-05
	10000	9	0.065825	2.36E-05	12	0.10169	1.58E-06
5	10	6	0.034072	9.50E-06	7	0.005677	5.35E-07
	1000	6	0.010845	3.00E-05	7	0.014299	1.69E-06
	10000	6	0.115347	9.50E-05	7	0.121814	5.35E-06

		MCGD			DDLS		
Problems	Dim	NI	Time	$\|F(x_k)\|$	NI	CPU	$\|F(x_k)\|$
6	100	12	0.0329	3.18E-05	_	_	_
	1000	13	0.014714	2.21E-05	_	_	_
	10000	13	0.140491	7.00E-05	_	_	_
7	100	17	0.074017	8.43E-05	27	0.045427	8.68E-05
	1000	20	0.109742	5.95E-05	34	0.136662	8.06E-05
	10000	19	0.607895	9.57E-05	31	0.958193	6.53E-05
8	100	10	0.005207	2.63E-05	11	0.005033	8.48E-05
	1000	10	0.011801	8.33E-05	13	0.012242	4.97E-05
	10000	11	0.092168	5.79E-05	14	0.107806	6.76E-05
9	10	22	0.009851	7.81E-05	23	0.008073	7.65E-05
	1000	24	0.020505	9.19E-05	25	0.022873	9.00E-05
	10000	27	0.169706	6.59E-05	28	0.159173	6.46E-05
10	10	9	0.004911	3.43E-05	10	0.005519	4.76E-05
	1000	10	0.010817	2.34E-05	11	0.012221	2.01E-05
	10000	10	0.093376	7.39E-05	11	0.087952	6.37E-05

From Table 1, we can easily observe that both of these methods attempt to solve the considered systems of nonlinear equations and a better efficiency and effectiveness of our proposed algorithm is clear since it works where the DDLS fails. This is quite evident, for instance, in problem 6. In general, the MCGD method considerably outperforms the DDLS for almost all the tested problems, as it has the least number of iterations and CPU time.

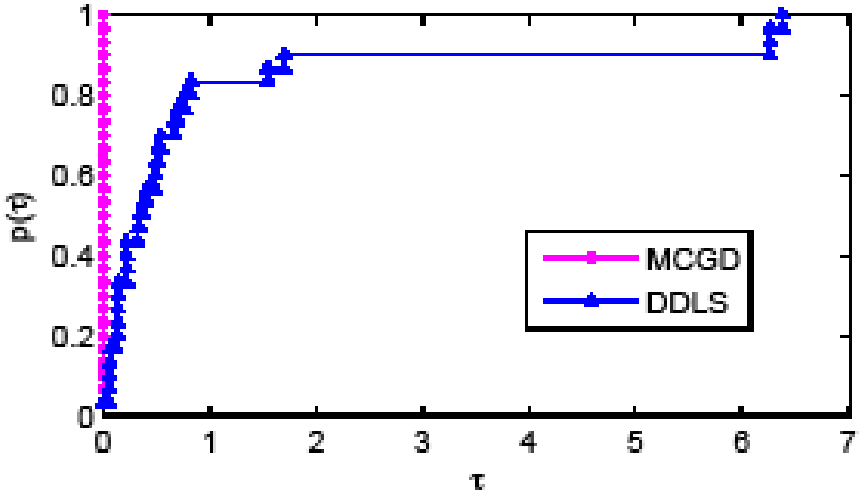

Figure 1: Performance profile of the MCGD and DDLS methods, in relation to the number of iterations NI, for the problems 1-10.

Figures (1-2) exhibit the comparative performance of our method in relation to the number of iterations and CPU time, which were evaluated using the profiles of Dolan and Mor´e, [4]. That is, for each method, we plot the fraction P(τ) of the problems for which the method is within a factor τ of the best time. The top curve is the method that solved most problems in a time that was within a factor τ of the best time.

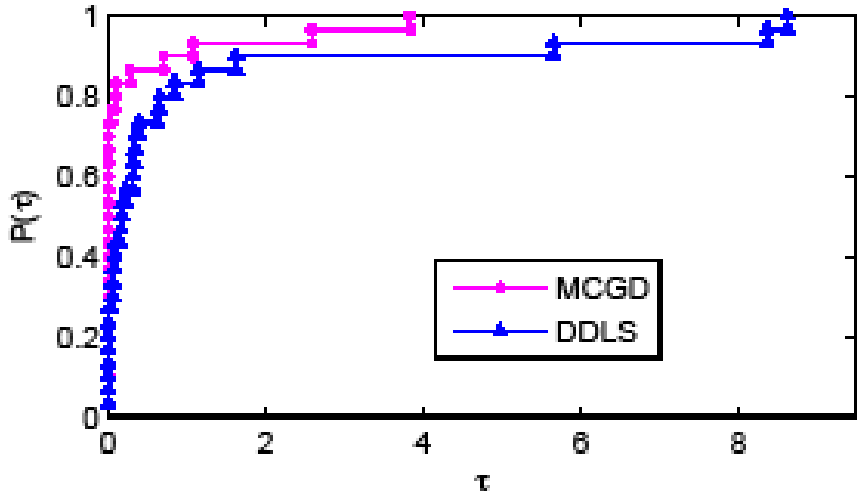

Figure 2: Performance profile of the MCGD and DDLS methods, in relation to the CPU time (in sec), for the problems 1-10.

5. Conclusion

In this paper we present a novel double direction iterative scheme MCGD for solving large-scale systems of symmetric nonlinear equations and compare its numerical performance with that of a double direction conjugate gradient method DDLS, [2], for solving these quations. It is quite obvious, from the numerical results the MCGD algorithm is robust in solving large-scale systems of nonlinear equations. In addition, we have proved the global convergence of the MCGD method by using the approximate norm descent line search proposed in [8]. The reported numerical results of the test experiments demonstrate the efficiency and good performance of this new method.

Acknowledgments
The authors would like to thank an anonymous referee for his careful reading of the original manuscript and for a number of useful comments.

References

[1] M. Al-baali, Descent property and global convergence of Fletcher-Reeves methods with inexact line search, *IMA Journal on Numerical Analysis* **5**, (1985), 121-124.

[2] H. Abdullahi, M. Y. Waziri, and M. K. Dauda, A double direction conjugate gradient method for solving large-scale system of nonlinear equations, *Journal of Mathematical and*

Comptational Science **7**(3), (2017), 606-6024.

[3] N. I. Dbaruranovic-milicic, A multistep curve search algorithm in nonlinear optimization, *Yugoslav Journal of Operation Research* **18**, (2008), 47-52.

[4] E. D. Dolan, and J. J. Mor´e, Benchmarking optimization software with performance profile, *Journal of Mathematical Programming* **91**(2), (2002), 201-213.

[5] W.W. Hager, and H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, *SIAM Journal on Optimization* **16**(1), (2005), 170-192.

[6] A. S. Halilu, and M.Y. Waziri, A transformed double step length method for solving large-scale systems of nonlinear equations, *Journal of Numerical Mathematics and Stochastics* **9**(1), (2017), 20-23.

[7] A. S. Halilu, and M.Y. Waziri, An improved derivative-free method via double direction approach for solving systems of nonlinear equations, *Journal of the Ramanujan Mathematical Society* **33**(1) (2018), 75-89.

[8] D. Li, and M. Fukushima, A globally and superlinearly convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations, *SIAM Journal on Numerical Analysis* **37**(1), (2000), 152-172.

[9] H. Liu, Y. Yao, X. Qian, and H. Wang, Some nonlinear conjugate gradient methods based on spectral scaling secant equations, *Computational and Applied Mathematics* **35**(2), (2016), 639-651.

[10] J. C. Meza, Steepest descent, *WIREs Computational Statistics* **2**(6), (2010),719-722.

[11] Y. B. Musa, *A Family of BFGS Methods For Solving Symmetric Nonlinear Systems of Equations* (Unpublished Master's Dissertation), Bayero University, Kano, Nigeria, 2015.

[12] M. Y. Waziri, and J. Sabiu, A derivative-free conjugate gradient method and its global convergence for symmetric nonlinear equations, *International Journal of Mathematics and Mathematical Sciences* **2015**, (2015), ID 961487, 8p.

[13] G. Yuan, and X. Lu, A new backtracking inexact BFGS method for symmetric nonlinear equations, *Computers and Mathematics With Applications* **55** (1), (2008), 116-129.

[14] L. Zhang, W. Zhou, and D. Li, A descent modified Polak-Ribirie-Polyak conjugate gradient method and its global convergence, *IMA Journal of Numerical Analysis* **26**, (2006), 629-640.

Article history: Submitted January, 01, 2018; Accepted September, 29, 2018.