Available online at http://scik.org
J. Math. Comput. Sci. 6 (2016), No. 5, 855-874
ISSN: 1927-5307

AN ALTERNATIVE CONJUGATE GRADIENT APPROACH FOR LARGE-SCALE SYMMETRIC NONLINEAR EQUATIONS

MOHAMMED YUSUF WAZIRI[1,*], JAMILU SABI'U[2]

[1]Department of Mathematical Sciences, Bayero University, Kano, Nigeria

[2]Department of Mathematics,Northwest University, Kano, Nigeria

Abstract. Nonlinear conjugate gradient method is a popular approach for solving large-scale unconstrained optimization problems due to its simplest iterative form and low storage requirement. In this article, we proposed a derivative free alternative conjugate gradient approach for solving symmetric nonlinear equations. We show that the proposed method has global convergence properties under appropriate conditions. We also report some numerical results to show its efficiency.

Keywords: backtracking line search; secant equation; symmetric nonlinear equations; conjugate gradient method.

2010 AMS Subject Classification: 65H11,65K05,65H12, 65H18.

1. Introduction

Let us consider the systems of nonlinear equations

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

where $F : R^n \to R^n$ is a nonlinear mapping. Often, the mapping, F is assumed to satisfying the following assumptions:

*Corresponding author

Received January 29, 2016

A1. There exists an $x^* \in R^n$ s.t $F(x^*) = 0$

A2. F is a continuously differentiable mapping in a neighborhood of x^*

A3. $F'(x^*)$ is invertible

A4. The Jacobian $F^{'}(x)$ is symmetric.

The prominent method for finding the solution of (1), is the classical Newton's method which generates a sequence of iterates $\{x_k\}$ from a given initial point x_0 via

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

where $k = 0, 1, 2 \ldots$.

The attractive features of this method are; rapid convergence and easy to implement. Nevertheless, Newton's method requires the computation of the Jacobian matrix, which require the first-order derivative of the systems. In practice, computations of some functions derivatives are quite costly and sometime they are not available or could not be done precisely. In this case Newton's method cannot be applied directly [6, 7, 11, 14, 20, 23]..

In this work, we are interested in handling large-scale problems for which the Jacobian is either not available or requires a low amount storage, the best method is CG approach. It is vital to mention that, the conjugate gradient methods are among the popular used methods for unconstrained optimization problems. They are particularly efficient for handling large-scale problems due to their convergence properties, simply to implement and low storage [17]. Not withstanding, the study of conjugate gradient methods for large-scale symmetric nonlinear systems of equations is scanty, this is what motivated us to have this paper.

2. Preliminaries

In general, conjugate gradient methods for solving nonlinear systems of equations generates an iterative points $\{x_k\}$ from initial given point x_0 using

$$x_{k+1} = x_k + \alpha_k d_k, \tag{3}$$

where $\alpha_k > 0$ ia attained via line search, and direction d_k are obtained using

(4) $$d_k = \begin{cases} -F(x_k) & if \quad k=0 \\ -F(x_k)+\beta_k d_k & if \quad k \geq 1 \end{cases}$$

β_k is term as conjugate gradient parameter.

This problems under study, may arise from an unconstrained optimization problem, a saddle point problem, Karush-Kuhn-Tucker (KKT) of equality constrained optimization problem, the discritized two-point boundary value problem, the discritized elliptic boundary value problem, and etc.

Equation (1) is the first-order necessary condition for the unconstrained optimization problem when F is the gradient mapping of some function $f : R^n \longrightarrow R$,

(5) $$min f(x), \quad x \varepsilon R^n.$$

For the equality constrained problem

(6) $$min f(x),$$
$$s.t \quad h(z)=0,$$

where h is a vector-valued function, the KKT conditions can be represented as the system (1) with $x=(z,v)$, and

(7) $$F(z,v) = (\nabla F(z) + \nabla h(z)v, h(z)),$$

where v is the vector of Lagrange multipliers. Notice that the Jacobian $\nabla F(z,v)$ is symmetric for all (z,v) (see, e.g., [22]).

Problem (1) can be converted to the following global optimization problem(5) with our function f defined by

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

A large number of efficient solvers for large-scale symmetric nonlinear equations have been proposed, analyzed, and tested in the last decade. Among them, the most classic one entirely due to Li and fukushima[5], in which a Gauss-Newton-based BFGS method is developed, and the global and superlinear convergence are also established. subsequently, it performance is further improved by Gu et al. [3], where a norm descent BFGS methods are designed. Since then, norm descent type BFGS methods especially coorporating with trust regions strategy are presented in the literature and showed their moderate effectiveness experimentally [16]. Still the matrix storage and solving of n-linear system are required in the BFGS type methods presented in the literature. The recent designed nonmonotone spectral gradient algorithm [2] falls within the frame work of matrix-free.

The conjugate gradient methods for symmetric nonlinear equations has received a good attension and take an appropriate progress. However, Li and Wang [8] proposed a modified Flectcher-Reeves conjugate gradient method which is based on the work of Zhang et al. [4], and the results illustrate that their proposed conjugate gradient method is promising. In line with this development, further studies on conjugate gradient are inspired for solving large-scale symmetric nonlinear equations. Zhou and Shen [12] extended the descent three-term polak-Rebiere-Polyak of Zhang et al [18] for solving (1) by combining with the work of Li and Fukushima [5]. Meanwhile the classic polak-Rebiere-Polyak is successfully used to solve symmetric equation (1) by Zhou and Shen [17].

Subsequentely Yunhai et al. [15] proposed a method based on weel-known conjugate gradient of Hager and Zhang [13], the proposed method converges globally. More recently, we proposed a derivative free conjugate gradient method and its global convergence for solving symmetric nonlinear equations [22]. Extensive numerical experiments showed that each over mentioned method performs quite well.In this paper, we proposed to present a new enhenced CG parameter β_k which is matrix and derivative free respectively. This is made possible by combining classical conjugate gradient (CG) direction with classical Newton direction with the viitue of modified secant equation proposed in [9].

3. Main results

In this setion we present a new CG parameter β_k, as a result of combining classical Conjugate Gradient direction with classical Newton direction. Recall the Classical CG direction is defined by

(9)
$$d_k = \begin{cases} -\nabla f(x_k) & if \quad k=0 \\ -\nabla f(x_k)+\beta_k d_k & if \quad k\geq 1 \end{cases}$$

In [22] we used the term

(10)
$$g_k = \frac{F(x_k+\alpha_k F_k)-F_k}{\alpha_k}.$$

to approximate the gradient $\nabla f(x_k)$, which avoids computing exact gradient. It is clear that, when $||F_k||$ is small, then $g_k \approx \nabla f(x_k)$.

Recall, from Newton's direction

(11)
$$d_{k+1} = -J^{-1}\nabla f(x_{k+1})$$

Combining (9) and (11), we have

(12)
$$-J(x_k)^{-1}\nabla f(x_{k+1}) = -\nabla f(x_{k+1})+\beta_k d_k$$

By multiplying both side of (12) by $J(x_k)$, lead to

(13)
$$-J(x_k)J(x_k)^{-1}\nabla f(x_{k+1}) = -J(x_k)\nabla f(x_{k+1})+J(x_k)\beta_k d_k$$

After little linear algebra, (13) can be transform to

(14)
$$-\nabla f(x_{k+1}) = -J(x_k)\nabla f(x_{k+1})+\beta_k J(x_k)d_k$$

We multiply both side of (14) by s_k^T to obtained:

(15)
$$-s_k^T\nabla f(x_{k+1}) = -s_k^T J(x_k)\nabla f(x_{k+1})+s_k^T\beta_k J(x_k)d_k, \quad s_k = x_{k+1}-x_k$$

Equation (15), can be rewritten as

(16)
$$-s_k^T\nabla f(x_{k+1}) = -s_k^T J(x_k)\nabla f(x_{k+1})+\beta_k s_k^T J(x_k)d_k$$

From Spectral secant condition in [9]; we have

(17)
$$J(x_k)s_k = \theta_k y_k \quad \text{and} \quad s_k = J(x_k)^{-1}\theta_k y_k$$

And

$$s_k^T J(x_k)^T = \theta_k y_k^T \quad \text{and} \quad s_k^T = s_k^T J(x_k)^{-1} \tag{18}$$

It is vital to note that, for this work, we claim that $J(x_k)$ is symmetric matrix $\forall k$. Hence, (18) can also be written as

$$s_k^T J(x_k) = \theta_k y_k^T \tag{19}$$

Substituting (19) into (16), yields

$$-s_k^T \nabla f(x_{k+1}) = -\theta_k y_k^T \nabla f(x_{k+1}) + \beta_k \theta_k y_k^T d_k \tag{20}$$

After, little algebra, we obtained our CG parameter (β_k) as

$$\beta_k = \frac{\theta_k y_k^T \nabla f(x_{k+1}) - s_k^T \nabla f(x_{k+1})}{\theta_k y_k^T d_k} \tag{21}$$

Further simplification on (21) gives:

$$\beta_k = \frac{(\theta_k y_k - s_k)^T}{\theta_k y_k^T d_k} \nabla f(x_{k+1}) \tag{22}$$

motivated by the ideas of [8, 17] and replacing the term $\nabla f(x_{k+1})$ by(10), we derive our CG parameter

$$\beta_k = \frac{(\theta_k y_k - s_k)^T}{\theta_k y_k^T d_k} g_{k+1}, \quad y_k = g_{k+1} + g_k \tag{23}$$

Having derived the CG parameter (β_k) in (23) and by using (9), we then present our direction as

$$d_0 = -g(x_0) \qquad d_{k+1} = -g_{k+1} + \frac{(\theta_k y_k - s_k)^T g_{k+1}}{\theta_k y_k^T d_k} d_k \qquad k = 1, 2 \ldots, \tag{24}$$

and

$$\theta_k = \frac{s_k^T s_k}{s_k^T y_k} \tag{25}$$

Finally, we present our scheme as

$$x_{k+1} = x_k + \alpha_k d_k. \tag{26}$$

Moreover, the direction d_k given by (24) may not be a descent direction of (8), then the standard wolfe and Armijo line searches can not be used to compute the stepsize directly. Therefore, we use the nonmonotone line search proposed by Li and Fukushima in [5] to compute our stepsize α_k. Let $\omega_1 > 0$, $\omega_2 > 0$, $r \in (0,1)$ be constants and $\{\eta_k\}$ be a given positive sequence such that

$$\sum_{k=0}^{\infty} \eta_k < \infty. \tag{27}$$

Let $\alpha_k = max\{1, r^k\}$ that satisfy

$$f(x_k + \alpha_k d_k) - f(x_k) \leq -\omega_1 ||\alpha_k F(x_k)||^2 - \omega_2 ||\alpha_k d_k||^2 + \eta_k f(x_k). \tag{28}$$

Now, we can describe the algorithm for our proposed method as follows:

ACGA Algorithm

Step 1 : Given x_0 ,$\alpha > 0$, $\omega \in (0,1)$, $r \in (0,1)$ and a positive sequence η_k satisfying (27), then compute $d_0 = -g_0$ and set $k = 0$.

Step 2 **:** Test a stopping criterion. If yes, then stop; otherwise continue with Step 3.

Step 3 : Compute α_k by the line search (28).

Step 4 : Compute $x_{k+1} = x_k + \alpha_k d_k$.

Step 5 : Compute the search direction as $d_{k+1} = -g_{k+1} + \frac{(\theta_k y_k - s_k)^T g_{k+1}}{\theta_k y_k^T d_k} d_k$.

Step 6 : Consider $k = k+1$ and go to step 2.

Convergence results

To analyze the convergence of our method, we will make the following assumptions on nonlinear systems (1)

(i) F is differentiable in an open convex set Ω in R^n.

(ii) There exists $x^* \in \Omega$ such that $F(x^*) = 0$, $F'(x)$ is continuous for all x .

(iii) In some neighborhood N of Ω, the Jacobian is Lipschitz continous i.e there exists a positive constant L such that

$$\|F'(x) - F'(y)\| \leq L\|x-y\|, \tag{29}$$

for all $x, y \in N$.

Properties (i) and (ii) implies that there exists positive constants M_1, M_2 and L_1 such that

$$||F(x)|| \leq M_1, \quad ||J(x)|| \leq M_2, \quad \forall x \varepsilon N, \tag{30}$$

$$||\nabla f(x) - \nabla f(y)|| \leq L_1||x-y||, \quad ||J(x)|| \leq M_2, \quad \forall x, y \in N. \tag{31}$$

Lemma 1.1.[22] Let the sequence $\{x_k\}$ be generated by the algorithms above. Then the sequence $\{||F_k||\}$ converges and $x_k \in E$ for all $k \geq 0$.

Lemma 1.2. Let the properties of (1) above hold. Then we have

$$\lim_{k\to\infty} ||\alpha_k d_k|| = \lim_{k\to\infty} ||s_k|| = 0, \tag{32}$$

$$\lim_{k\to\infty} ||\alpha_k F_k|| = 0 \tag{33}$$

Proof. by (27) and (28) we have for all $k > 0$,

$$\omega_2||\alpha_k d_k||^2 \leq \omega_1||\alpha_k F(x_k)||^2 + \omega_2||\alpha_k d_k||^2 \leq ||F_k||^2 - ||F_{k+1}||^2 + \eta_k||F_k||^2 \tag{34}$$

by summing the above k inequality, then we obtain:

$$\omega_2 \sum_{i=0}^{k} ||\alpha_k d_k||^2 \leq ||F_k||^2 \left\{ \sum_{i=0}^{k} (1-\eta_i) \right\} - ||F_{k+1}||^2 \tag{35}$$

so, from (30) and the fact that $\{\eta_k\}$ satisfies (27) the series $\sum_{i=0}^{k} ||\alpha_k d_k||^2$ is convergent. This implies (32). By a similar way, we can prove that(33) holds.

The following result shows that An Alternative CG method algorithm is globally convergent.

Theorem 1.1. Let the properties of (1) above hold. Then the sequence $\{x_k\}$ be generated ACGA algorithm converges globally, that is,

$$\liminf_{k\to\infty} ||\nabla f(x_k)|| = 0. \tag{36}$$

Proof. We prove this theorem by contradiction. Suppose that (36) is not true, then there exists a positive constant τ such that

$$||\nabla f(x_k)|| \geq \tau, \quad \forall k \geq 0. \tag{37}$$

Since $\nabla f(x_k) = J_k F_k$, (37) implies that there exists a positive constant τ_1 satisfying

$$||F_k|| \geq \tau_1, \quad \forall k \geq 0. \tag{38}$$

Case (i): $\limsup_{k\to\infty} \alpha_k > 0$. then by (33), we have $\liminf_{k\to\infty} ||F_k|| = 0$. This and Lemma 1 show that $\lim_{k\to\infty} ||F_k|| = 0$, which contradicts with (37).

Case (ii): $\limsup_{k\to\infty} \alpha_k = 0$. Since $\alpha_k \geq 0$,this case implies that

$$\lim_{k\to\infty} \alpha_k = 0. \tag{39}$$

by definition of g_k in (10) and the symmetry of the Jacobian, we have

$$\begin{aligned} ||g_k - \nabla f(x_k)|| &= ||\frac{F(x_k + \alpha_{k-1}F_k) - F_k}{\alpha_{k-1}} - J_k^T F_k|| \\ &= ||\int_0^1 J(x_k + t\alpha_{k-1}F_k) - J_k)dtF_k|| \\ &\leq LM_1^2 \alpha_{k-1}, \end{aligned} \tag{40}$$

where we use (30) and (31) in the last inequality. (27), (28) and (37) show that there exists a constant $\tau_2 > 0$ such that

$$||g_k|| \geq \tau_2, \quad \forall k \geq 0. \tag{41}$$

By (10) and (30), we get

$$||g_k|| = \int_0^1 J(x_k + t\alpha_{k-1}F_k)F_k dt|| \leq M_1 M_2, \quad \forall k \geq 0. \tag{42}$$

From (42) and (31), we obtain

$$\begin{aligned} ||y_k|| &= ||g_k - g_{k+1}|| \\ &\leq ||g_k - \nabla f(x_k)|| + ||g_{k-1} - \nabla f(x_{k-1})|| + ||\nabla f(x_k) - \nabla f(x_{k-1})|| \\ &\leq LM_1^2(\alpha_{k-1} + \alpha_{k-2}) + L_1||s_{k-1}||. \end{aligned} \tag{43}$$

This together with (39) and (33) shows that $\lim_{k\to\infty}||y_k|| = 0$.Hence from (22), (43) and (41), we have

$$|\theta_k| \leq \frac{||s_k^T||||s_k||}{||s_k^T||||y_k||} \longrightarrow 0 \tag{44}$$

meaning there exists a constant $\lambda\varepsilon(0,1)$ such that for sufficiently large k

$$|\theta_k| \leq \lambda. \tag{45}$$

Again from the definition of our β_k we obtain

$$|\beta_k| \leq \frac{||\theta_k y_k - s_k||||g_{k+1}||}{||\theta_k y_k^T||||s_k||} \leq M_1 M_2 \frac{||y_k - s_k||}{||y_k^T||||s_k||} \longrightarrow 0 \tag{46}$$

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

$$|\beta_k| \leq \rho. \tag{47}$$

Without lost of generality, we assume that the above inequalities holds for all $k \geq 0$. Then we get

$$||d_{k+1}|| \leq ||\theta_k g_{k+1}|| + |\beta_k|||d_k|| \leq \lambda M_1 M_2 + \rho||d_k|| \tag{48}$$

which shows that the sequence $\{d_{k+1}\}$ is bounded. Since $\lim_{k\to\infty}\alpha_k = 0$, then $\alpha_k^{'} = \frac{\alpha_k}{r}$ does not satisfy (28), namely

$$f(x_k + \alpha_k^{'} d_k) > f(x_k) - \omega_1||\alpha_k^{'} F(x_k)||^2 - \omega_2||\alpha_k^{'} d_k||^2 + \eta_k f(x_k), \tag{49}$$

which implies that

$$\frac{f(x_k + \alpha_k^{'} d_k) - f(x_k)}{\alpha_k^{'}} > -\omega_1||\alpha_k^{'} F(x_k)||^2 - \omega_2||\alpha_k^{'} d_k||^2. \tag{50}$$

By the mean-value theorem, there exists $\delta_k \in (0,1)$ such that

$$\frac{f(x_k + \alpha_k^{'} d_k) - f(x_k)}{\alpha_k^{'}} = \nabla f(x_k + \delta_k \alpha_k^{'} d_k)^T d_k. \tag{51}$$

Since $\{x_k\} \subset \Omega$ is bounded, without loss of generality, we assume $x_k \longrightarrow x^*$. By (10) and (24), we have

$$\lim_{k\to\infty} d_{k+1} = -\lim_{k\to\infty}\theta_{k+1} g_{k+1} + \lim_{k\to\infty}\beta_k^* d_k \leq -\lim_{k\to\infty} g_{k+1} + \lim_{k\to\infty}\beta_k^* d_k = -\nabla f(x^*), \tag{52}$$

where we use (46), (28) and the fact that the sequence $\{d_{k+1}\}$ is bounded.

On the other hand, we have

$$\lim_{k\to\infty} \nabla f(x_k + \delta_k \alpha_k' d_k) = \nabla f(x^*). \tag{53}$$

Hence, from (50)-(53), we obtain $-\nabla f(x^*)^T \nabla f(x^*) \geq 0$, which means $||\nabla f(x^*)|| = 0$. This contradicts with (37). The proof is completed.

Numerical Experiment

We compared the performance of our method for solving nonlinear equation (1) with an inexact prp conjugate gradient method for symmetric nonlinear equations [17].

- An alternative CG method (ACGA): We set $\omega_1 = \omega_2 = 10^{-4}$, $\alpha_0 = 0.01$, $r = 0.1$ and $\eta_k = \frac{1}{(k+1)^2}$.
- For an inexact prp (IPRP) conjugate gradient method for symmetric nonlinear equations. We set $\omega_1 = \omega_2 = 10^{-4}$, $\alpha_0 = 0.01$, $r = 0.1$ and $\eta_k = \frac{1}{(k+1)^2}$..

The code for both ACGA and IPRP methods were written in Matlab 7.4 R2010a and run on a personal computer 1.8 GHz CPU processor and 4 GB RAM memory. We stopped the iteration if the toatal number of iterations exceeds 1000 or $||F_k|| \leq 10^{-3}$. We use "-" to represent failure due one of the following:

(i) Memory requirement

(ii) Number of iteration exceed 1000

(ii) If $||F_k||$ is not a number

We tested the methods on eight test problems with different initial points and values. Problem 1-3 are from[12, 17] while problem 6-8 are from [21]

Problem 1

$F_1(x) = x_1(x_1^2 + x_2^2) - 1$

$F_i(x) = x_i(x_{i-1}^2 + 2x_i^2 + x_{i+1}^2) - 1 \quad ;1,2,\dots,n-1$

$F_n(x) = x_n(x_{n-1} + x_n^2).$

Problem 2: For $i = 1,2,,n/3$,

$F_{3i-2}(x) = x_{3i-2}x_{3i-1} - x_{3i}^2 - 1$,

$F_{3i-1}(x) = x_{3i-2}x_{3i-1}x_{3i} - x_{3i-2}^2 + x_{3i-1}^2 - 2$,

$F_{3i}(x) = e^{-x_{3i-2}} - e^{-x_{3i-1}}$.

Problem 3: For $i = 1,2,,n$

$F_i(x) = 2(n + i(1 - cosx_i) - sinx_i - \sum_{j=1}^{n} cosx_j)(2sinx_i - cosx_i)$

Problem 4:

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

Problem 5:

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

Problem 6:

$F_i(x) = 0.05(x_i - 1) + 2sin(\sum_{j=1}^{n}(x_j - 1) + \sum_{j=1}^{n}(x_j - 1)^2(1 + 2(x_i - 1)) + 2sin(\sum_{j=1}^{n}(x_j - 1))$

Problem 7:

$F_1(x) = 4(x_1 - x_2^2)$

$F_i(x) = 8x_i(x_i^2 - x_{i-1}) - 2(1 - x_i) + 4(x_i - x_{i+1}^2) \quad i = 2,3,\ldots,n-1$

$F_n(x) = 8x_n(x_n - x_{n-1}) - 2(1 - x_n)$

Problem 8:

$F_1(x) = (3 - 0.5x_1)x_1 - 2x_2 + 1$

$F_i(x)(3 - 0.5x_i)x_i - x_{n-1} - 2x_{i+1} + 1$

$F_n(x) = (3 - 0.5x_n)x_n - x_{n-1} + 1$

Test results for the two methods, where e =ones(n,1)

Table 1			ACGA			IPRP		
Problem (P)	x_0	n	Iter	Time(s)	$\|\|F_k\|\|$	Iter	Time(s)	$\|\|F_k\|\|$
p1	e	10	42	0.752608	9.85E-04	196	0.240149	9.74E-04
		50	44	0.050456	9.21E-04	170	0.182533	9.89E-04
		100	36	0.047633	8.58E-04	204	0.20819	9.92E-04
		500	49	0.095838	8.72E-04	210	0.360981	9.74E-04
		1000	44	0.136491	7.85E-04	187	0.503445	9.50E-04
		5000	52	0.521437	8.89E-04	188	1.970965	9.76E-04
		10000	34	0.63899	9.73E-04	240	4.58276	9.60E-04
		20000	44	1.585452	8.63E-04	241	9.41398	9.97E-04
		50000	48	4.282012	8.68E-04	246	22.844312	9.40E-04
	0.1e	100	35	0.052414	9.66E-04	215	0.212664	9.71E-04
		500	42	0.082621	9.59E-04	218	0.352023	9.91E-04
		1000	39	0.100708	9.90E-04	223	0.561376	9.76E-04
		2000	30	0.130572	9.10E-04	222	0.927342	9.98E-04
		10000	45	0.802444	9.48E-04	231	4.37962	9.84E-04
		20000	53	1.988779	9.58E-04	225	9.06472	9.97E-04
		50000	40	3.602575	9.82E-04	228	20.774155	9.91E-04
	0.01e	2000	38	0.16429	9.83E-04	217	0.897524	9.93E-04
		10000	35	0.684191	8.85E-04	219	3.872882	9.76E-04
		100000	44	10.153122	8.16E-04	223	47.827076	9.63E-04

Table 1 continue			ACGA			IPRP		
Problem (P)	x_0	n	Iter	Time(s)	$\|\|F_k\|\|$	Iter	Time(s)	$\|\|F_k\|\|$
p2	e	50	7	0.033037	6.71E-04	13	0.023542	5.70E-04
		100	7	0.015066	9.64E-04	13	0.024732	8.19E-04
		1000	8	0.0433	6.13E-04	15	0.058468	5.77E-04
		10000	9	0.267192	3.88E-04	17	0.445501	4.03E-04
		50000	9	1.084196	8.67E-04	17	1.743164	9.02E-04
	0.1e	50	98	0.127368	5.06E-04	-	-	-
		100	98	0.144261	7.27E-04	-	-	-
		500	109	0.244072	9.85E-04	-	-	-
		1000	115	0.355587	4.33E-04	-	-	-
		5000	115	1.294865	9.68E-04	-	-	-
		10000	125	2.742368	9.39E-04	-	-	-
		20000	131	5.800331	4.09E-04	-	-	-
p2	-0.1e	10	125	0.164449	9.45E-04	-	-	-
		50	140	0.196023	9.62E-04	-	-	-
		200	155	0.25351	9.29E-04	-	-	-
		500	170	0.377719	6.74E-04	-	-	-
		1000	170	0.537462	9.55E-04	-	-	-
		10000	200	4.600113	6.49E-04	-	-	-
p3	e	10	12	0.020635	9.02E-04	15	0.028958	3.64E-04
		50	27	0.073073	8.50E-04	12	0.029466	5.10E-04
		100	9	0.025248	6.67E-04	17	0.042195	8.00E-04
		500	8	0.179431	2.27E-06	-	-	-
		1000	14	0.232427	9.67E-04	-	-	-
		2000	18	0.542088	3.52E-04	-	-	-

Table 1 continue			ACGA			IPRP		
	0.5e	10	10	0.016647	9.59E-04	14	0.013692	9.33E-04
		50	12	0.020676	8.86E-04	4	0.009348	2.46E-04
		100	11	0.036904	5.13E-04	25	0.078017	6.12E-04
		500	10	0.071195	4.25E-05	13	0.108413	9.06E-04
		1000	9	0.13924	1.25E-04	16	0.263713	2.27E-04
		4000	16	0.391362	3.35E-04	-	-	-
		5000	21	5.494875	3.79E-04	-	-	-
		10000	20	3.436877	4.80E-04	-	-	-
p4	e	10	31	0.360259	8.86E-04	56	0.625474	9.60E-04
		50	26	0.326918	9.55E-04	101	1.362075	9.77E-04
		100	25	0.356038	8.10E-04	107	1.745258	9.77E-04
		500	114	3.639123	8.07E-04	480	17.578523	9.00E-04
		1000	126	10.108339	9.16E-04	642	63.880333	9.94E-04
		2000	114	22.171047	7.91E-04	375	74.620778	9.98E-04
	0.1e	10	28	0.31605	8.44E-04	36	0.402393	9.63E-04
		50	27	0.333383	8.53E-04	39	0.461376	9.29E-04
		100	33	0.582877	9.04E-04	42	0.69488	8.92E-04
		500	32	1.068035	8.55E-04	94	3.555216	9.52E-04
		1000	31	3.332772	6.09E-04	92	8.187002	9.83E-04
		2000	25	5.815651	8.00E-04	115	21.717311	9.58E-04
		5000	33	39.496329	9.63E-04	111	149.64502	8.96E-04

Table 1 continue			ACGA			IPRP		
Problem (P)	x_0	n	Iter	Time(s)	$\|\|F_k\|\|$	Iter	Time(s)	$\|\|F_k\|\|$
p5	e	1000	25	2.11235	8.20E-04	31	2.575008	8.70E-04
		2000	26	6.220177	7.92E-04	32	7.644628	9.22E-04
		5000	27	28.502911	8.35E-04	34	39.972992	7.68E-04
p6	0.01e	10	4	0.009875	5.18E-04	7	0.016171	5.12E-04
		100	8	0.034561	7.12E-04	14	0.052137	6.82E-04
		250	5	0.029561	2.57E-04	12	0.054734	1.40E-04
		300	19	0.100332	7.02E-04	-	-	-
		500	8	0.072018	2.80E-04	-	-	-
		1000	33	0.234581	9.03E-04	-	-	-
p7	0.4e	10	519	0.505866	9.36E-04	-	-	-
		20	880	0.850211	9.97E-04	-	-	-
		57	950	1.007061	8.52E-04	-	-	-
p8	-1e	10	85	0.066362	9.57E-04	102	0.099925	9.85E-04
		50	111	0.125444	8.66E-04	152	0.151121	8.97E-04
		100	109	0.128249	9.83E-04	159	0.16323	8.80E-04
		500	119	0.204833	9.54E-04	158	0.268675	9.67E-04
		1000	119	0.305788	9.68E-04	162	0.402651	9.51E-04
		2000	135	0.623371	9.75E-04	-	-	-
		3000	133	0.894332	8.91E-04	-	-	-

In the table 1, we listed numerical results, where ”Iter” and ”Time” stand for the total number of all iterations and the CPU time in seconds, respectively;$||F_k||$ is the norm of the residual at the stopping point.The numerical results indicate that the proposed method ACGA compared

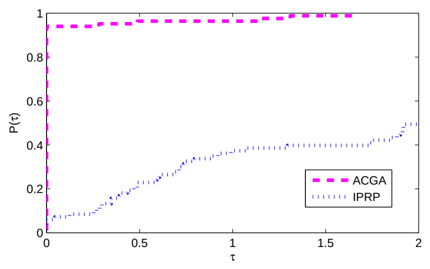

FIGURE 1. Comparison of the performance of ACGA and IPRP methods (in term of CPU time)

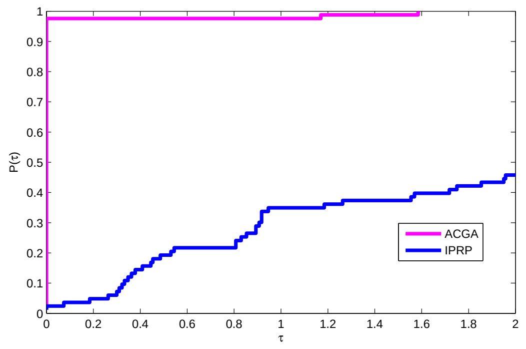

FIGURE 2. Comparison of the performance of ACGA and IPRP methods (in term of number of iterations)

to IPRP has minimum number of iteration and CPU time respectively. Figure (1) and (2) are performance profile derived by Dolan and more[19] which show that our claim is justified i.e. Less CPU time and number of iteration for each test problem.

Moreover, in average, our $||F(x_k)||$ is too small which signifies that, the solution obtained is true approximation of the exact solution compared to the IPRP.

Conflict of Interests

The authors declare that there is no conflict of interests.

REFERENCES

[1] E. Birgin, J.M. Martińez, A spectral conjugate gradient method for unconstrained optimization, Appl. Math. optim. 43(2001), 117-128.

[2] W. Cheng, Z. Chen, Nonmonotone Spectral method for large-Scale symmetric nonlinear equations. Numer. Algorithms, 62(2013), 149-162.

[3] G.Z. Gu, D.H. Li, D, L. Qi, S.Z. Zhou, Descent direction of quasi-Newton methods for symmetric nonlinear equations, SIAM J. Numer. Anal. 40(2002), 1763-1774.

[4] L. Zhang, W. Zhou, D.-H. Li, A descent modified polak-Rebiére-Polyak conjugate gradient method and its global convergence, IMA J. Numer. Anal. 26(2006), 629-640.

[5] D.-H. Li, M. Fukushima, A globally and superlinearly convergent Gauss-Newton-based BFGS methods for symmetric nonlinear equations, SIAM J. Numer. Anal. 37(1999), 152-172.

[6] M.Y.Waziri and Z. A. Majid, An Enhanced Matrix-Free Secant Method via Predictor-Corrector Modified Line Search Strategies for Solving Systems of Nonlinear Equations, Journal of Mathematics and mathematical sciences, Volume (2013), Article number 814587.

[7] H. Mohammad and M. Y. Waziri, On Broyden-like update via some quadratures for solving nonlinear systems of equations , Turkish Journal of Mathematics, (2015) 39: 335 - 345, doi:10.3906/mat-1404-41.

[8] D.-H. Li, X. Wang, A modified Fletcher-Reeves-type derivative-free method for symmetric nonlinear equations, Numer. Algebra Control Optim. 1(2011), 71-82.

[9] H. Liu, Y. Yao, X. Qian, H. Wang, Some nonlinear conjugate gradient methods based on spectral scaling secant equations, Comp. Appl. Math.DOI 10.1007/s40314-014-0212-1.

[10] J.M. Ortega, W.C. Rheinboldt, Iterative solution of nonlinear equations in several variables: Academic press, New York 1970.

[11] M.Y.Waziri, W.J. Leong, M.A. Hassan and M. Monsi , A New Newton's Method with Diagonal Jacobian Approximation for Systems of Nonlinear Equations, Journal of mathematics and statistic,6(3) (2010):246-252.

[12] W. Zhou, D. Shen, Convergence properties of an iterative method for solving symmetric nonlinear equations, J. Optim. Theory Appl.(2014) doi: 10. 1007/s10957-014-0547-1.

[13] W.W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, SIAM J. Optim. 16(2005), 170-192.

[14] M.Y. Waziri, W.J. Leong, M.A. Hassan, M. Monsi, Jacobian computation-free Newton method for systems of Non-Linear equations. Journal of numerical Mathematics and stochastic. 2 (2010): 54-63.

[15] X.Yunhai, W. Chunjie, Y.W. Soon, Norm descent conjugate gradient method for solving symmetric nonlinear equations, J. Glo. Optim.(2015) DOI 10.1007/s10898-014-0218-7.

[16] G. Yuan, X. Lu, Z. Wei, BFGS trust-region method for symmetric nonlinear equations, J. Comput. Appl. Math. 230(2009), 44-58.

[17] W. Zhou, D. Shen, An inexact PRP conjugate gradient method for symmetric nonlinear equations, Numer. Functional Anal. Opt. 35(2014), 370-388.

[18] L. Zhang, W. Zhou, D.-H. Li, A descent modified polak-Rebi$\acute{e}$re-Polyak conjugate gradient method and its global convergence, IMA J. Numer. Anal. 26(2006), 629-640.

[19] E.D. Dolan, J.J. Mor$\acute{e}$, Benchmarking Optimization software with Performance profiles, Maths. program. 91(2002), 201-213.

[20] M.Y. Waziri, W.J. Leong, M.A. Hassan, M. Monsi, Newton method for systems of Non-Linear equations with singular Jacobian using diagonal updating. International Journal of Mathematics and computations.8(2008), 1-8.

[21] W. La Cruz, J.M. Mart$\acute{i}$nez, M. Raydan, spetral residual method without gradient information for solving large-scale nonlinear systems of equations: Theory and experiments, P. optimization 76(2004),79.1008-00.

[22] M.Y. Waziri, J. Sabiu, A derivative-free conjugate gradient method and its global convergence for solving symmetric nonlinear equations, International J. of mathematics and mathematical science 2015(2015), 8 doi:10.1155/2015/961487.

[23] M.Y.Waziri, W.J. Leong, M.A. Hassan, M. Monsi, An efficient solver for systems of Non-Linear equations with singular Jacobian Via diagonal updating. Applied mathematical sciences 69(2010), 3403 - 3412.