Hindawi Publishing Corporation
International Journal of Mathematics and Mathematical Sciences
Volume 2015, Article ID 961487, 8 pages
http://dx.doi.org/10.1155/2015/961487

Research Article

A Derivative-Free Conjugate Gradient Method and Its Global Convergence for Solving Symmetric Nonlinear Equations

Mohammed Yusuf Waziri[1] and Jamilu Sabi'u[2]

[1]*Department of Mathematical Sciences, Faculty of Science, Bayero University Kano, Kano, Nigeria*
[2]*Department of Mathematics, Faculty of Science, Northwest University Kano, Kano, Nigeria*

Correspondence should be addressed to Mohammed Yusuf Waziri; mywaziri@gmail.com

Received 15 June 2015; Accepted 11 August 2015

Academic Editor: Naseer Shahzad

We suggest a conjugate gradient (CG) method for solving symmetric systems of nonlinear equations without computing Jacobian and gradient via the special structure of the underlying function. This derivative-free feature of the proposed method gives it advantage to solve relatively large-scale problems (500,000 variables) with lower storage requirement compared to some existing methods. Under appropriate conditions, the global convergence of our method is reported. Numerical results on some benchmark test problems show that the proposed method is practically effective.

1. Introduction

Let us consider the systems of nonlinear equations:

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

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

(A1) There exists $x^* \in R^n$ 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

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

where $k = 0, 1, 2, \ldots$. The attractive features of this method are rapid convergence and easy implementation. 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.

In this work, we are interested in handling large-scale problems for which the Jacobian either is not available or requires a low amount of 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, simple implementation, and low storage [1]. Notwithstanding, the study of conjugate gradient methods for large-scale symmetric nonlinear systems of equations is scanty, and this is what motivated us to have this paper.

In general, CG methods for solving nonlinear systems of equations generate iterative points $\{x_k\}$ from initial given point x_0 using

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

where $\alpha_k > 0$ is attained via line search and direction d_k is obtained using

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

where β_k is termed as conjugate gradient parameter.

These problems which are under study may arise from an unconstrained optimization problem, a saddle point problem, Karush-Kuhn-Tucker (KKT) of equality constrained optimization problem, the discretized two-point boundary value problem, the discretized elliptic boundary value problem, and so forth.

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

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

For the equality constrained problem,

$$\begin{aligned} \min \quad & f(x), \\ \text{s.t.} \quad & h(z) = 0, \end{aligned} \quad (6)$$

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

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

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., [2]).

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

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

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 [3], in which a Gauss-Newton-based BFGS method is developed, and the global and superlinear convergence are also established. Subsequently, its performance is further improved by Gu et al. [4], where norm descent BFGS methods are designed. Since then, norm descent type BFGS methods especially cooperating with trust regions strategy are presented in the literature and showed their moderate effectiveness experimentally [5]. Still the matrix storage and solving of n-linear systems of equations are required in the BFGS type methods presented in the literature. The recent designed nonmonotone spectral gradient algorithm [6] falls within the framework of matrix free methods.

The conjugate gradient methods for symmetric nonlinear equations have received a good attention and take an appropriate progress. However, Li and Wang [7] proposed a modified Fletcher-Reeves conjugate gradient method which is based on the work of Zhang et al. [8], 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 [9] extended the descent three-term polak-Ribiere-Polyak of Zhang et al. [10] for solving (1) by combining with the work of Li and Fukushima [3]. Meanwhile the classic polak-Ribiere-Polyak is successfully used to solve symmetric equation (1) by Zhou and Shen [1].

Subsequently Xiao et al. [11] proposed a method based on well-known conjugate gradient of Hager and Zhang [12], and the proposed method converges globally. Extensive numerical experiments showed that each of the above-mentioned methods performs quite well. The combination of conjugate gradient algorithms and the Newton method, for the first time, was presented by Andrei [13, 14]. Hence, in this paper, we intended to present a new enhanced CG parameter β_k which is matrix- and derivative-free, respectively. This is made possible by combining Birgin and Martínez direction with classical Newton direction.

We organized the paper as follows. In the next section, we present the details of the proposed method. Convergence results are presented in Section 3. Some numerical results are reported in Section 4. Finally, conclusions are made in Section 5.

2. Derivation of the Method

In this section we present a new CG parameter β_k, as a result of combining Birgin and Martínez direction with classical Newton direction. Recalling the Birgin and Martínez direction in [15] is defined by

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

where $\theta_k = s^T s_k / s_k^T y_k$; see Raydan [16] for detail.

In [2] Ortega and Rheinboldt used the term

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

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

Recall, from Newton's direction,

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

Combining (9) and (11), we have

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

Multiplying both sides of (12) by $J(x_k)$ leads to

$$\begin{aligned} & -J(x_k) J(x_k)^{-1} \nabla f(x_{k+1}) \\ & \quad = -J(x_k) \nabla f(x_{k+1}) \theta_k + J(x_k) \beta_k d_k. \end{aligned} \quad (13)$$

After little linear algebra, (13) transforms to

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

To ensure good approximation, we multiply both sides of (14) by s_k^T to obtain

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

Equation (15) can be rewritten as

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

From secant condition, we have

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

$$s_k^T J(x_k)^T = y_k^T, \quad s_k^T = s_k^T J(x_k)^{-1\,T}. \quad (18)$$

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

$$s_k^T J(x_k) = y_k^T. \quad (19)$$

Substituting (19) into (16) yields

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

After simplification, we obtained our CG parameter (β_k) as

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

Motivated by [3, 7] and using (10) we derive our CG parameter

$$\beta_k = \frac{(\theta_k y_k - s_k)^T}{y_k^T d_k} g_{k+1}. \quad (22)$$

Having derived the CG parameter (β_k) in (22) and by using (9), we then present our direction as

$$d_0 = -g(x_0),$$

$$d_{k+1} = -\theta_k g_{k+1} + \frac{(\theta_k y_k - s_k)^T g_{k+1}}{y_k^T d_k} d_k, \quad k = 1, 2, \ldots, \quad (23)$$

where $\theta_k = s^T s_k / s_k^T y_k$.

Finally, we present our scheme as

$$x_{k+1} = x_k + \alpha_k d_k. \quad (24)$$

The direction d_k given by (23) may not be a descent direction of (8), and then the standard Wolfe and Armijo line searches cannot be used to compute the step size directly. Zhang et al. [8] proved the global convergence of the global PRP method for general nonconvex optimization using some nondescent line search. Motivated by this, we use the nonmonotone line search proposed by Li and Fukushima in [3] to compute our step size α_k. Let $\omega_1 > 0$, $\omega_2 > 0$, and $r \in (0,1)$ be constants and let $\{\eta_k\}$ be a given positive sequence such that

$$\sum_{k=0}^{\infty} \eta_k < \infty. \quad (25)$$

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

$$f(x_k + \alpha_k d_k) - f(x_k) \le -\omega_1 \|\alpha_k F(x_k)\|^2 - \omega_2 \|\alpha_k d_k\|^2 + \eta_k f(x_k). \quad (26)$$

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

Algorithm 1 (derivative-free CG method (DFCG)). Consider the following steps.

Step 1. Given x_0, $\alpha > 0$, $\sigma \in (0,1)$, and compute $d_0 = -g_0$, and set $k = 0$.

Step 2. Compute g_k using (10) and test the stopping criterion. If yes, then stop; otherwise continue with Step 3.

Step 3. Compute α_k by the line search (26).

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

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

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

3. Convergence Result

This section presents global convergence results of the derivative-free conjugate gradient methods. To begin with, let us define the level set

$$\Omega = \{x \mid f(x) \le e^n f(x_0)\}. \quad (27)$$

In order to analyze the convergence of our method, we will make the following assumptions on nonlinear systems F.

Assumption 2. Consider the following:

(i) The level set Ω defined by (27) is bounded.

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

(iii) In some neighborhood N of Ω, the Jacobian is Lipschitz continuous; that is, there exists a positive constant $L > 0$ such that

$$\|F'(x) - F'(y)\| \le L \|x - y\|, \quad (28)$$

for all $x, y \in N$.

Properties (i) and (ii) imply that there exist positive constants M_1, M_2, and L_1 such that

$$\begin{aligned} \|F(x)\| &\le M_1, \\ \|J(x)\| &\le M_2, \\ &\forall x \in N, \end{aligned} \tag{29}$$

$$\begin{aligned} \|\nabla f(x) - \nabla f(y)\| &\le L_1 \|x - y\|, \\ \|J(x)\| &\le M_2, \\ &\forall x, y \in N. \end{aligned} \tag{30}$$

Lemma 3 (see [3]). *Let the sequence $\{x_k\}$ be generated by the algorithms above. Then the sequence $\{\|F_k\|\}$ converges and $x_k \in E$ for all $k \ge 0$.*

Lemma 4. *Let the properties of (1) above hold. Then one has*

$$\lim_{k\to\infty} \|\alpha_k d_k\| = \lim_{k\to\infty} \|s_k\| = 0, \tag{31}$$

$$\lim_{k\to\infty} \|\alpha_k F_k\| = 0. \tag{32}$$

Proof. By (25) and (26) we have, for all $k > 0$,

$$\begin{aligned} \omega_2 \|\alpha_k d_k\|^2 &\le \omega_1 \|\alpha_k F(x_k)\|^2 + \omega_2 \|\alpha_k d_k\|^2 \\ &\le \|F_k\|^2 - \|F_{k+1}\|^2 + \eta_k \|F_k\|^2 \end{aligned} \tag{33}$$

by summing the above k inequality; then we obtain

$$\omega_2 \sum_{i=0}^{k} \|\alpha_k d_k\|^2 \le \|F_k\|^2 \left\{ \sum_{i=0}^{k} (1 - \eta_i) \right\} - \|F_{k+1}\|^2, \tag{34}$$

so from (29) and the fact that $\{\eta_k\}$ satisfies (25) the series $\sum_{i=0}^{k} \|\alpha_k d_k\|^2$ is convergent. This implies (31). By a similar way, we can prove that (32) holds. □

The following result shows that our derivative-free conjugate gradient methods algorithm is globally convergent.

Theorem 5. *Let the properties of (1) above hold. Then the sequence $\{x_k\}$ which is generated by derivative-free conjugate gradient methods algorithm converges globally; that is,*

$$\liminf_{k\to\infty} \|\nabla f(x_k)\| = 0. \tag{35}$$

Proof. We prove this theorem by contradiction. Suppose that (35) is not true, and then there exists a positive constant τ such that

$$\|\nabla f(x_k)\| \ge \tau, \quad \forall k \ge 0. \tag{36}$$

Since $\nabla f(x_k) = J_k F_k$, (36) implies that there exists a positive constant τ_1 satisfying

$$\|F_k\| \ge \tau_1, \quad \forall k \ge 0. \tag{37}$$

Case (i) is as follows: consider $\limsup_{k\to\infty} \alpha_k > 0$. Then by (32), we have $\liminf_{k\to\infty} \|F_k\| = 0$. This and Lemma 3 show that $\lim_{k\to\infty} \|F_k\| = 0$, which contradicts with (36).

Case (ii) is as follows: consider $\limsup_{k\to\infty} \alpha_k = 0$. Since $\alpha_k \ge 0$, this case implies that

$$\lim_{k\to\infty} \alpha_k = 0, \tag{38}$$

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

$$\begin{aligned} \|g_k - \nabla f(x_k)\| &= \left\| \frac{F(x_k + \alpha_{k-1} F_k) - F_k}{\alpha_{k-1}} - J_k^T F_k \right\| \\ &= \left\| \int_0^1 \left(J(x_k + t\alpha_{k-1} F_k) - J_k\right) dt F_k \right\| \\ &\le L M_1^2 \alpha_{k-1}, \end{aligned} \tag{39}$$

where we use (29) and (30) in the last inequality. Inequalities (25), (26), and (36) show that there exists a constant $\tau_2 > 0$ such that

$$\|g_k\| \ge \tau_2, \quad \forall k \ge 0. \tag{40}$$

By (10) and (29), we get

$$\begin{aligned} \|g_k\| = \left\| \int_0^1 J(x_k + t\alpha_{k-1} F_k) F_k dt \right\| &\le M_1 M_2, \\ &\forall k \ge 0. \end{aligned} \tag{41}$$

From (41) and (30), we obtain

$$\begin{aligned} \|y_k\| &= \|g_k - g_{k+1}\| \\ &\le \|g_k - \nabla f(x_k)\| + \|g_{k-1} - \nabla f(x_{k-1})\| \\ &\quad + \|\nabla f(x_k) - \nabla f(x_{k-1})\| \\ &\le L M_1^2 (\alpha_{k-1} + \alpha_{k-2}) + L_1 \|s_{k-1}\|. \end{aligned} \tag{42}$$

This together with (38) and (32) shows that $\lim_{k\to\infty} \|y_k\| = 0$. Hence from (41), (42), and (40), we have

$$|\theta_k| \le \frac{\|s_k^T\| \|s_k\|}{\|s_k^T\| \|y_k\|} \longrightarrow 0 \tag{43}$$

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

$$|\theta_k| \le \lambda. \tag{44}$$

Again from the definition of our β_k^* we obtain

$$\begin{aligned} |\beta_k^*| \le \frac{\|\theta_k y_k - s_k\| \|g_{k+1}\|}{\|y_k^T\| \|s_k\|} &\le M_1 M_2 \frac{\|\theta_k y_k - s_k\|}{\|y_k^T\| \|s_k\|} \\ &\longrightarrow 0 \end{aligned} \tag{45}$$

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

$$|\beta_k^*| \le \rho. \tag{46}$$

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

$$\|d_{k+1}\| \leq \|\theta_k g_{k+1}\| + |\beta_k| \|d_k\| \leq \lambda M_1 M_2 + \rho \|d_k\| \quad (47)$$

which shows that the sequence $\{d_{k+1}\}$ is bounded. Since $\lim_{k\to\infty}\alpha_k = 0$, then $\alpha_k' = \alpha_k/r$ does not satisfy (26); namely,

$$f\left(x_k + \alpha_k' d_k\right) > f\left(x_k\right) - \omega_1 \left\|\alpha_k' F\left(x_k\right)\right\|^2 - \omega_2 \left\|\alpha_k' d_k\right\|^2 + \eta_k f\left(x_k\right), \quad (48)$$

which implies that

$$\frac{f\left(x_k + \alpha_k' d_k\right) - f\left(x_k\right)}{\alpha_k'} > -\omega_1 \left\|\alpha_k' F\left(x_k\right)\right\|^2 - \omega_2 \left\|\alpha_k' d_k\right\|^2. \quad (49)$$

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

$$\frac{f\left(x_k + \alpha_k' d_k\right) - f\left(x_k\right)}{\alpha_k'} = \nabla f\left(x_k + \delta_k \alpha_k' d_k\right)^T d_k. \quad (50)$$

Since $\{x_k\} \subset \Omega$ is bounded, without loss of generality, we assume $x_k \to x^*$. By (10) and (23), we have

$$\begin{aligned}\lim_{k\to\infty} d_{k+1} &= -\lim_{k\to\infty}\theta_{k+1} g_{k+1} + \lim_{k\to\infty}\beta_k^* d_k \\ &\leq -\lim_{k\to\infty} g_{k+1} + \lim_{k\to\infty}\beta_k^* d_k = -\nabla f\left(x^*\right),\end{aligned} \quad (51)$$

where we use (45) and (26) and the fact that the sequence $\{d_{k+1}\}$ is bounded.

On the other hand, we have

$$\lim_{k\to\infty}\nabla f\left(x_k + \delta_k \alpha_k' d_k\right) = \nabla f\left(x^*\right). \quad (52)$$

Hence, from (49)–(52), we obtain $-\nabla f(x^*)^T \nabla f(x^*) \geq 0$, which means $\|\nabla f(x^*)\| = 0$. This contradicts with (36). The proof is then completed. □

4. Numerical Results

In this section, we compare the performance of our following methods for solving nonlinear equation (1) with an inexact PRP conjugate gradient method for solving symmetric nonlinear equations [1]:

(i) A derivative-free CG method (DFCG): we set $\omega_1 = \omega_2 = 10^{-4}$, $\alpha_1 = 0.01$, $r = 0.2$, and $\eta_k = 1/(k+1)^2$.

(ii) An inexact PRP (IPRP): we set $\omega_1 = \omega_2 = 10^{-4}$, $\alpha_1 = 0.01$, $r = 0.2$, and $\eta_k = 1/(k+1)^2$.

The code for the DFCG method was 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 total number of iterations exceeds 1000 or $\|F_k\| \leq 10^{-3}$. We tested the two methods on nine test problems with different initial points and n values. Problems 6–9 are from [9].

Problem 1:

$$F_i(x) = \left(1 - x_i^2\right) + x_i\left(1 + x_i x_{n-2} x_{n-1} x_n\right) - 2, \quad i = 1, 2, \ldots, n. \quad (53)$$

Problem 2:

$$\begin{aligned}F_i(x) &= x_i - 0.1 x_{i+1}^2, \quad i = 1, 2, \ldots, n-1, \\ F_n(x) &= x_n - 0.1 x_1^2.\end{aligned} \quad (54)$$

Problem 3:

$$\begin{aligned}F_1(x) &= x_1^2 - 3x_1 + 1 + \cos\left(x_1 - x_2\right), \\ F_i(x) &= x_i^2 - 3x_i + 1 + \cos\left(x_i - x_{i-1}\right), \\ &\qquad i = 1, 2, \ldots, n.\end{aligned} \quad (55)$$

Problem 4:

$$\begin{aligned}F_i(x) &= x_i\left(\cos x_i - \frac{1}{n}\right) \\ &\quad - x_n\left[\sin x_i - 1 - \left(x_i - 1\right)^2 - \frac{1}{n}\sum_{i=1}^{n} x_i\right], \\ &\qquad i = 1, 2, \ldots, n.\end{aligned} \quad (56)$$

Problem 5:

$$F_i(x) = x_i - 3x_i\left(\frac{\sin x_i}{3} - 0.66\right) + 2, \quad \text{for } i = 1, 2, \ldots, n. \quad (57)$$

Problem 6:

$$\begin{aligned}F_1(x) &= x_1\left(x_1^2 + x_2^2\right) - 1, \\ F_i(x) &= x_i\left(x_{i-1}^2 + 2x_i^2 + x_{i+1}^2\right), \quad i = 2, 3, \ldots, n-1, \\ F_n(x) &= x_n\left(x_{n-1}^2 + x_n^2\right).\end{aligned} \quad (58)$$

Problem 7:

$$\begin{aligned}F_{3i-2}(x) &= x_{3i} - 2x_{3i-1} - x_{3i}^2 - 1, \\ F_{3i-1}(x) &= x_{3i-2} x_{3i-1} x_{3i} - x_{3i-2}^2 + x_{3i-1}^2 - 2, \\ F_{3i}(x) &= e^{-x_{3i-2}} - e^{-x_{3i-1}}.\end{aligned} \quad (59)$$

Problem 8:

$$F(x) = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix} x + \left(e_1^x - 1, \ldots, e_n^x - 1\right)^T. \quad (60)$$

Table 1: Test results for the two methods, where $e = \text{ones}(n, 1)$.

Problem (P)	x_0	n	DFCG			IPRP		
			Iter	Time (s)	$\Vert F_k \Vert$	Iter	Time (s)	$\Vert F_k \Vert$
1	0.5e	100	7	0.011199	$2.03E-04$	15	0.018191	$7.56E-04$
		500	7	0.019229	$4.54E-04$	16	0.039993	$9.77E-04$
		1000	7	0.025726	$6.42E-04$	17	0.058292	$7.43E-04$
		2000	7	0.042199	$9.08E-04$	18	0.099609	$6.03E-04$
		5000	8	0.138149	$1.69E-05$	18	0.263836	$9.53E-04$
2	e	10	5	0.005881	$7.53E-06$	7	0.007772	$9.53E-05$
		500	5	0.009256	$5.32E-05$	7	0.016192	$6.74E-04$
		5000	5	0.073385	$1.68E-04$	9	0.097816	$2.92E-06$
		10000	5	0.16378	$2.38E-04$	9	0.172704	$4.14E-06$
		100000	5	2.308894	$7.53E-04$	9	1.923969	$1.31E-05$
		500000	6	8.43021	$3.28E-07$	9	10.123595	$2.92E-05$
3	0.5e	100	3	0.006565	$2.42E-05$	5	0.005039	$1.23E-06$
		1000	3	0.014514	$7.64E-05$	5	0.023714	$3.90E-06$
		5000	3	0.068582	$1.71E-04$	5	0.08417	$8.71E-06$
		10000	3	0.132546	$2.42E-04$	5	0.16412	$1.23E-05$
		100000	3	1.49153	$7.64E-04$	5	2.297944	$3.90E-05$
		500000	4	9.227381	$7.21E-07$	5	9.234416	$8.71E-05$
4	0.01e	100	3	0.006919	$9.15E-05$	51	0.090634	$8.20E-04$
		500	3	0.010871	$2.06E-04$	57	0.17519	$9.40E-04$
		1000	3	0.021833	$2.92E-04$	61	0.276824	$8.50E-04$
		5000	3	0.088476	$6.53E-04$	67	1.277072	$9.73E-04$
		20000	4	0.312725	$8.75E-07$	73	4.686063	$9.97E-04$
5	e	50	4	0.006221	$1.67E-04$	11	0.015403	$5.79E-04$
		500	4	0.014341	$5.28E-04$	14	0.040054	$5.73E-04$
		5000	5	0.090997	$2.69E-06$	15	0.236538	$2.79E-04$
		50000	5	0.725966	$8.52E-06$	15	1.627153	$8.82E-04$
	0.001e	500	6	0.01594	$6.07E-04$	17	0.044403	$5.63E-04$
		5000	7	0.105773	$1.22E-06$	19	0.253817	$5.54E-04$
		50000	7	0.811278	$3.86E-06$	20	2.042188	$9.98E-04$
6	0.1	50	35	0.038655	$8.04E-04$	54	0.06337	$7.59E-04$
		500	33	0.060248	$5.98E-04$	43	0.086961	$9.77E-04$
		1000	46	0.109835	$8.61E-04$	68	0.198439	$9.88E-04$
		2000	37	0.13327	$8.20E-04$	58	0.245378	$9.26E-04$
		3000	35	0.217145	$8.02E-04$	59	0.395438	$9.60E-04$
		8000	39	0.53251	$9.06E-04$	60	1.015941	$7.87E-04$
		15000	48	1.237265	$8.80E-04$	62	1.938341	$9.80E-04$
7	e	10	4	0.00539	$4.32E-05$	17	0.029569	$8.17E-04$
		50	4	0.006503	$9.98E-05$	21	0.042007	$6.78E-04$
		500	4	0.011576	$3.21E-04$	25	0.08001	$7.86E-04$
		5000	5	0.099718	$5.91E-07$	29	0.442995	$8.96E-04$
		50000	5	0.731552	$1.87E-06$	35	4.100204	$6.12E-04$
8	e	10	34	0.366137	$8.54E-04$	31	0.403368	$8.94E-04$
		50	26	0.29999	$4.94E-04$	36	0.495837	$3.31E-04$
		100	28	0.410634	$8.53E-04$	29	0.490114	$9.47E-04$
		500	224	6.894744	$9.19E-04$	317	11.140713	$7.85E-04$
		1000	140	10.284715	$7.87E-04$	226	19.377201	$7.54E-04$
		2000	134	18.935605	$4.21E-04$	149	30.958082	$9.70E-04$
	0.1	10	18	0.202136	$9.48E-04$	28	0.373317	$9.96E-04$
		100	32	0.460852	$4.80E-04$	29	0.484755	$5.21E-04$
		500	32	0.96522	$8.97E-04$	37	1.365365	$8.93E-04$
		1000	23	1.717939	$7.32E-04$	32	3.189859	$7.99E-04$
		2000	27	5.596633	$9.00E-04$	34	9.085955	$7.93E-04$
		5000	27	25.832759	$5.92E-04$	28	33.107487	$7.68E-04$

Table 1: Continued.

Problem (P)	x_0	n	DFCG			IPRP		
			Iter	Time (s)	$\|F_k\|$	Iter	Time (s)	$\|F_k\|$
		2000	11	0.027851	$7.39E-06$	13	0.030663	$6.19E-04$
9	e	3000	66	25.412837	$8.44E-04$	69	34.946683	$9.34E-04$
		5000	63	72.39168	$8.03E-04$	69	83.7963	$9.95E-04$

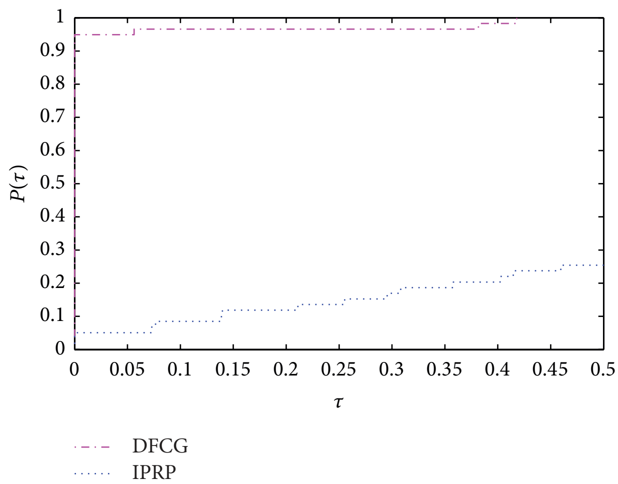

Figure 1: Comparison of the performance of DFCG and IPRP methods as the dimension increases (in terms of CPU time).

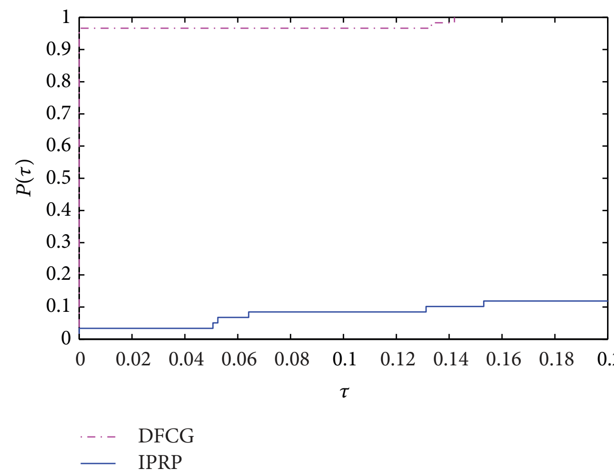

Figure 2: Comparison of the performance of DFCG and IPRP methods as the dimension increases (in terms of number of iterations).

Problem 9:

$$F(x) = \begin{pmatrix} 2 & -1 & & & \\ 0 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & 0 & 2 \end{pmatrix} x + \left(\sin x_1 - 1, \ldots, \sin x_n - 1\right)^T. \quad (61)$$

In 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 DFCG compared to IPRP has minimum number of iterations and CPU time, respectively. Figures 1 and 2 are performance profile derived by Dolan and Moré [17] which show that our claim is justified, that is, less CPU time and number of iterations for each test problem.

Moreover, on 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.

5. Conclusions

Many practical problems possessed symmetrical property. In this paper we present a derivative-free conjugate gradient (DFCG) method for symmetric nonlinear equations and compare its performance with that of an inexact PRP conjugate gradient method [1] by doing some numerical experiments. We also proved the global convergence of our proposed method by using a backtracking type line search, and the numerical results show that our method is efficient.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] W. Zhou and D. Shen, "An inexact PRP conjugate gradient method for symmetric nonlinear equations," *Numerical Functional Analysis and Optimization*, vol. 35, no. 3, pp. 370–388, 2014.

[2] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, NY, USA, 1970.

[3] D. H. Li and M. Fukushima, "A globally and superlinearly convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations," *SIAM Journal on Numerical Analysis*, vol. 37, no. 1, pp. 152–172, 2000.

[4] G.-Z. Gu, D.-H. Li, L. Qi, and S.-Z. Zhou, "Descent directions of quasi-Newton methods for symmetric nonlinear equations," *SIAM Journal on Numerical Analysis*, vol. 40, no. 5, pp. 1763–1774, 2002.

[5] G. Yuan, X. Lu, and Z. Wei, "BFGS trust-region method for symmetric nonlinear equations," *Journal of Computational and Applied Mathematics*, vol. 230, no. 1, pp. 44–58, 2009.

[6] W. Cheng and Z. Chen, "Nonmonotone spectral method for large-scale symmetric nonlinear equations," *Numerical Algorithms*, vol. 62, no. 1, pp. 149–162, 2013.

[7] D.-H. Li and X.-L. Wang, "A modified fletcher-reeves-type derivative-free method for symmetric nonlinear equations," *Numerical Algebra, Control and Optimization*, vol. 1, no. 1, pp. 71–82, 2011.

[8] L. Zhang, W. Zhou, and D.-H. Li, "Global convergence of a modified Fletcher–Reeves conjugate gradient method with Armijo-type line search," *Numerische Mathematik*, vol. 104, no. 4, pp. 561–572, 2006.

[9] W. Zhou and D. Shen, "Convergence properties of an iterative method for solving symmetric non-linear equations," *Journal of Optimization Theory and Applications*, vol. 164, no. 1, pp. 277–289, 2015.

[10] L. Zhang, W. Zhou, and D.-H. Li, "A descent modified Polak-Ribière-Polyak conjugate gradient method and its global convergence," *IMA Journal of Numerical Analysis*, vol. 26, no. 4, pp. 629–640, 2006.

[11] Y. Xiao, C. Wu, and S.-Y. Wu, "Norm descent conjugate gradient methods for solving symmetric nonlinear equations," *Journal of Global Optimization*, vol. 62, no. 4, pp. 751–762, 2015.

[12] W. W. Hager and H. Zhang, "A new conjugate gradient method with guaranteed descent and an efficient line search," *SIAM Journal on Optimization*, vol. 16, no. 1, pp. 170–192, 2005.

[13] N. Andrei, "A scaled BFGS preconditioned conjugate gradient algorithm for unconstrained optimization," *Applied Mathematics Letters*, vol. 20, no. 6, pp. 645–650, 2007.

[14] N. Andrei, "Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization," *Optimization Methods and Software*, vol. 22, no. 4, pp. 561–571, 2007.

[15] E. G. Birgin and J. M. Martínez, "A spectral conjugate gradient method for unconstrained optimization," *Applied Mathematics and Optimization*, vol. 43, no. 2, pp. 117–128, 2001.

[16] M. Raydan, "The Barzilaiai and Borwein gradient method for the large scale unconstrained minimization problem," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 26–33, 1997.

[17] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, no. 2, pp. 201–213, 2002.