Sohag Journal of Mathematics
An International Journal

http://dx.doi.org/10.18576/sjm/040203

An Accelerated Three-Term Conjugate Gradient Algorithm for Solving Large-Scale Systems of Nonlinear Equations

Mohammed Yusuf Waziri[1] *and Lawal Muhammad* [2,*]

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

Received: 12 Jul. 2016, Revised: 10 Jan. 2017, Accepted: 14 Jan. 2017
Published online: 1 May 2017

Abstract: Nonlinear conjugate gradient method is very popular in solving large-scale unconstrained optimization problems due to its lower storage requirement and simple iterative procedure. Research activities on extending nonlinear conjugate gradient method to higher dimensional systems of nonlinear equations are just beginning. This paper presents simple three-term conjugate gradient algorithm for solving large-scale systems of nonlinear equations.The strategies of acceleration and restart were incorporated in designing the algorithm to improve its numerical performance. The global convergence of the proposed scheme with the general Wolfe conditions under a suitable assumption was proved. Finally, the computational experiment was presented to show the efficiency of the proposed method.

Keywords: Unconstrained optimization, Systems of nonlinear equations, Conjugate gradient, General wolfe condition.
Mathematics Subject Classification: 65H11, 65K05,65H12, 65H18

1 Introduction

Consider the problem of finding the solution of

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

where $F : R^n \to R^n$ is a nonlinear mapping. Often, the mapping, F is a continuously differentiable mapping. Equation (1) is the first-order necessary condition for the unconstrained optimization problem when F is the gradient mapping of some function $f : R^n \to R$,

$$min\, f(x), x \in R^n. \tag{2}$$

Problem (1) can be converted to the following global optimization problem (2) with our function defined by

$$f = \frac{1}{2}||F(x)||^2 \tag{3}$$

As a system of nonlinear equations there are numerous algorithms for (1), often the following iterative procedure is used:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{4}$$

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

$$d_{k+1} = -F(x_{k+1}) + \beta_k d_k, \qquad d_0 = -F(x_0). \tag{5}$$

In general nonlinear equations, there are numerous algorithms which have been developed to deal with these problems, for example, the Newton and quasi-Newton methods [4], the Gauss-Newton methods [5], the gradient-based and the conjugate gradient methods [6,9, 10,11,12], the trust region method [14], the Levenberg-Marquardt methods [15], the tensor methods [7],the derivative-free methods [16] and the subspace methods [13]. Since conjugate gradient methods do not need the storage of matrices, they are paid attention to as an effective method for solving large-scale unconstrained optimization problem and large-scale nonlinear equations. It is well-known that choices of β_k affect the the numerical performance of the method, and hence many

* Corresponding author e-mail: lawalmuhd.nwu.maths@gmail.com

researchers have studied effective choices of β_k (see [2,6,9], for example). The Recent development of conjugate gradient methods and their global convergence properties is reviewed by Hager and Zhang [9]. There is a weakness of conjugate gradient methods, namely, most of the conjugate gradient methods do not necessarily satisfy the descent condition $\nabla g(x)^T d_k \leq 0$. Recently, some researchers proposed three-term conjugate gradient methods which always generate descent search directions (see [9,10,11,12,13] for example). This is what motivated us, to proposed a simple three conjugate gradient algorithm for solving large-scale systems of nonlinear equations by modifying the classical BFGS inverse approximation of the Jacobian inverse restarted as identity matrix at every step. The method possessed low memory requirement, global convergence properties and simple to be implemented.

The main contribution of this paper is to construct a fast and efficient three-term conjugate gradient method for solving (1) the proposed algorithm is based on the recent three-term conjugate gradient algorithm for Deng and Wan [12] for unconstrained optimization. In other words our algorithm can be thought as an extension to three-term conjugate gradient algorithm to a general systems of nonlinear equations. We present experimental results and performance comparison with a three-terms Polak-Ribiere-Polyak conjugate gradient algorithm for large-scale nonlinear equations [5], to illustrate that the proposed algorithm is efficient and promising.

The rest of the paper is organized as follows: In section 2, we simply recall the latest three-term conjugate gradient algorithm for unconstrained optimization of Deng and Wan and then construct our algorithm . Subsequently Convergence results are presented in Section 3. Some numerical results are reported in Section 4 to show its practical performance. Finally, conclusions are made in Section 5.

Throughout this paper, $||.||$ denote the Euclidean norm of a vector. For simplicity, we abbreviate $g(x_k)$ as g_k in the context.

2 Algorithm

In this section, we briefly review the latest family of three-term conjugate gradient method of Deng and Wan [12] and then construct our method step by step. consider

$$min\, f(x), x \in \mathrm{R}^n, \tag{6}$$

where $f : \mathrm{R}^n \to \mathrm{R}$ is continuously differentiable such that it gradient is available.

Let $g : \mathrm{R}^n \to \mathrm{R}^n$ denote the gradient of g, and let g_k denote the value of g at x_k. The recent designed method of Deng and Wan generate a sequence $\{x_k\}$ with an arbitrary choosing initial starting point $x_0 \in \mathrm{R}^n$ giving by

$$x_{k+1} = x_k + \alpha_k d_k, \tag{7}$$

where $k > 0$, $d_k \in \mathbf{R^n}$ is called a search direction at x_k and $\alpha_k > 0$ is a step size along d_k. Denote $y_k = g_{k+1} - g_k$, $s_k = x_{k+1} - x_k$, the search direction is generated by

$$d_{k+1} = \begin{cases} -g_{k+1} & \text{for} k = 0, \\ -g_{k+1} - \delta_k s_k - \eta_k y_k & \text{for} k \geq 1, \end{cases} \tag{8}$$

where δ_k is defined by

$$\delta_k = (1 - min\left\{1, \frac{||y_k||^2}{y_k^T s_k}\right\})\frac{s_k^T g_{k+1}}{y_k^T s_k} - \frac{y_k^T g_{k+1}}{y_k^T s_k}, \tag{9}$$

$$\eta_k = \frac{s_k^T g_{k+1}}{y_k^T s_k}. \tag{10}$$

According to [12] d_{k+1} determined by [10] and [11] is descent direction and it also noted if $y_k^T s_k \leq ||y_k||^2$ holds then, (8), reduces to the method in [13].

incorporating with an improved line search and a dynamic strategies of accelaration and restart into the algorithm, their proposed three-term conjugate gradient is experimentally illustrated very promising and outerperforms the existent similar state-of-the-art algorithm.

Now we are ready to turn our attention to the general systems of nonlinear equations.

It is well known that if the jacobian matrix of g is posotive definite, the most efficient search direction at x_k is the Newton direction

$$d_{k+1} = -(\nabla g_{k+1})^{-1} g_{k+1}. \tag{11}$$

Thus, for the Newton direction d_{k+1}, it holds true that

$$s_k^T \nabla g_{k+1} d_{k+1} = -s_k^T g_{k+1}. \tag{12}$$

From the secant condition that,

$$\nabla g_{k+1} s_k = y_k, \tag{13}$$

(12) can be approximated by

$$y_k^T d_{k+1} = -s_k^T g_{k+1}. \tag{14}$$

Therefore, if a search direction is d_{k+1} is required to satisfy (14), then it can be regarded as approximate Newton direction. Observe that

$$d_{k+1} = -Q_k g_{k+1}, \tag{15}$$

where Q_k is a matrix, given by

$$Q_k = I - \frac{s_k y_k^T + y_k s_k^T}{y_k^T s_k} + (1 - \frac{||y_k||^2}{y_k^T s_k})\frac{s_k s_k^T}{y_k^T s_k}. \tag{16}$$

Recall that, the classical BFGS Jacobian inverse approximation update method is represented as:

$$B_{k+1} = B_K - \frac{s_k y_k^T B_k + B_k y_k s_k^T}{y_k^T s_k} + (1 - \frac{y_k^T B_k y_k}{y_k^T s_k})\frac{s_k s_k^T}{y_k^T s_k}. \tag{17}$$

It is noticed that there is a close relation between (16) and (17). Actually by setting $B_k = I$ and modifying the sign infront of $y_k^T s_k$ in the third term of (17) we are going to obtain the direction as

$$d_{k+1} = -g_{k+1} - ((1+k\frac{||y_k||^2}{y_k^T s_k})\frac{s_k^T g_{k+1}}{y_k^T s_k} - \frac{y_k^T g_{k+1}}{y_k^T s_k})s_k - \frac{s_k^T g_{k+1}}{y_k^T s_k}. \quad (18)$$

Then it is easy to obtain that $k = -1$ and by direct computation d_{k+1} satisfies (14), from this point of view (18) is not always descent specifically, if $y_k^T s_k < ||y_k||^2$, for this purpose (18) can be modified as

$$d_{k+1} = -g_{k+1} - \delta_k s_k - \eta_k y_k, \quad (19)$$

where

$$\delta_k = (1 - min\{1, \frac{||y_k||^2}{y_k^T s_k}\})\frac{s_k^T g_{k+1}}{y_k^T s_k} - \frac{y_k^T g_{k+1}}{y_k^T s_k}, \quad (20)$$

$$\eta_k = \frac{s_k^T g_{k+1}}{y_k^T s_k}. \quad (21)$$

Finally, we have

$$x_{k+1} = x_k + \xi_k \alpha_k d_k, \quad (22)$$

where

$$\xi_k = -\frac{\overline{a_k}}{\overline{b_k}}, \quad (23)$$

$\overline{a_k} = \alpha_k g_k^T d_k$, $\overline{b_k} = -\alpha_k(g_k - g_z)^T$, $g_z = g(z)$ and $z = x_k + \alpha_k d_k$. Hence If $\overline{b_k} > 0$, then the new estimation of the solution is computed as $x_{k+1} = x_k + \xi_k \alpha_k d_k$, Otherwise, $x_{k+1} = x_k + \alpha_k d_k$.
Therefore with the constructed search direction and the acceleration scheme we find the stepsize by the standard Wolfe line search strategy

$$f(x_k + \alpha_k d_k) - f(x_k) \le \rho \alpha_k g_k^T d_k, \qquad g_{k+1}^T \ge \sigma g_k^T d_k, \quad (24)$$

where d_k is descent direction and $0 < \rho < \sigma < 1$.

Algorithm 2.1 (STTCG)

Step 1 : Given x_0 ,$\alpha > 0$, $\sigma \in (0,1)$, and compute $d_0 = -g_0$, set $k = 0$.
Step 2 : If $||g_k|| < \varepsilon$ then stop; otherwise continue with step 3.
Step 3 : Determine the stepsize α_k by using the standard Wolfe line search strategy in (24).
Step 4 :Compute $z = x_k + \alpha_k d_k$, $g_k = g(z_k)$ and $y_k = g_k - gz$.
Step 5 :Compute $\overline{a_k} = \alpha_k g_k^T d_k$ and $\overline{b_k} = -\alpha_k y_k^T d_k$.
Step 6 : Acceleration scheme: If $\overline{b_k} > 0$, then computed as $\xi_k = -\frac{\overline{a_k}}{\overline{b_k}}$ and update the iterate points $x_{k+1} = x_k + \xi_k \alpha_k d_k$; Otherwise, $x_{k+1} = z$ and compute g_{x+1} and $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$
Step 7 : Determine δ_k and η_k by (20) and (21) respectively.
Step 8 :compute the search direction by (19).
Step 9 : powel restart criterion. If $||g_{k+1}||^2 > 0.2||g_{k+1}||^2$, then set $d_{k+1} = -g_{k+1}$. Step 10 :Consider $k = k+1$ and go to step 2.

3 Convergence Result

In this Section, we will establish the global convergence for Algorithm 2.1.
Definition
Let Ω be the level set defined by

$$\Omega = \{x| ||g(x)|| \le \sqrt{||g(x)||^2 + \tau}\}$$

where τ is a positive constant.
The following Assumptions are needed to establish the global convergence of the Algorithm 2.1
Assumption A. (i) The level set $\Omega = \{x| ||g(x)|| \le \sqrt{||g(x)||^2 + \tau}\}$ is bounded;
(ii) In some neighborhood N of Ω, $g(x)$ is lipschitz continuous, i.e there exist a constant $L > 0$ s.t for all $x, y \in N$,

$$||g(x) - g(y)|| \le L||x - y||. \quad (25)$$

Asssumption A(ii) implies that there exists a constant $\kappa > 0$ such that

$$||g(x)|| \le \kappa, \quad (26)$$

for all $x \in \Omega$. Thus the sequence $\{g_k\}$ is bounded.
Since $g(x_k)$ is decreasing as $k \to +\infty$, it follows that from assumption A(i) that the sequence $\{x_k\}$ generated by Algorithm 2.1 is clearly contained in a bounded region. Thus, there exist a convergent subsequence of $\{x_{k_j}\}$. without lost of generality, it is supposed that $\{x_k\}$ is convergent.

Lemma 1. *Let $g : R^n \to R^n$ be a continuously differentiable mapping, suppose that the line search satisfies the general Wolfe line search in (24), then d_{k+1} defined by (19),(20) and (21) is descent direction.*

Proof. From general Wolfe conditions in (24), we have $y_k^T s_k > 0$. if $y_k^T s_k < ||y_k||^2$, then,

$$g_{k+1}^T d_{k+1} = -||g_{k+1}||^2 < 0. \quad (27)$$

Otherwise,

$$g_{k+1}^T d_{k+1} = -||g_{k+1}||^2 - \left(1 - min\{1, \frac{y_k^T \theta_k y_k}{y_k^T s_k}\}\right)\frac{(g_{k+1}^T s_k)^2}{y_k^T s_k} \le 0. \quad (28)$$

Lemma 2. *Suppose that the line search satisfies general Wolfe (24). Then d_{k+1} defined by (19),(20) and (21) satisfies the conjugacy condition that*

$$d_{k+1}^T y_k = -t_k g_{k+1}^T s_k. \quad (29)$$

Proof. By direct computation we get

$$y_k^T d_{k+1} = -\left(1 - min\{1, \frac{y_k^T \theta_k y_k}{y_k^T s_k}\}\right) + \frac{y_k^T \theta_k y_k}{y_k^T s_k})s_k^T g_{k+1}, \quad (30)$$

where

$$t_k = 1 - min\{1, \frac{y_k^T \theta_k y_k}{y_k^T s_k}\}) + \frac{||y_k||^2}{y_k^T s_k} > 0. \quad (31)$$

In particular, if $||y_k|| < y_k^T s_k$, then $t_k = 1$. Therefore, the direction d_{k+1} defined by (19),(20) and (21) satisfies conjugacy condition (29).

Lemma 3. *Suppose that d_k is a descent direction and assumption A (ii) holds then for all x on the line segment connecting x_k and x_{k+1}. Under the general Wolfe line search (24) it holds true that*

$$\alpha_k \geq \frac{(1-\sigma)|g_k^T d_k|}{L||d_k||}. \quad (32)$$

Proof. subtracting $g_k^T d_k$ from both side of inequality (24) we are going to obtain

$$(\sigma - 1)g_k^T d_k \leq (g_{k+1} - g_k)^T d_k = y_k^T d_k \leq ||y_k||||d_k|| \leq \alpha_k L||d_k||^2. \quad (33)$$

since d_k is decsent and $0 < \sigma < 1$, then the desired result i.e (32) follows directly .

Lemma 4. *Let $\{d_k\}$ and $\{\alpha_k\}$ be two sequences generated by Algorithm 2.1. suppose that Assumptions A(i) and A(ii) hold. Then,*

$$\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{||d_k||} < +\infty. \quad (34)$$

Proof. From (24) and lemma 3 it follows that

$$f_k - f_{k+1} \geq -\rho \alpha_k g_k^T d_k \geq \rho \frac{(1-\sigma)(g_k^T d_k)^2}{L||d_k||^2} \quad (35)$$

Then by assumption A(i) (34) is proved.

Lemma 5. *Let $\{d_k\}$ and $\{\alpha_k\}$ be two sequences generated by Algorithm 2.1. where α_k is obtained by the (24). suppose that Assumption A(i) and A(ii) hold. If*

$$\sum_{k \geq 1} \frac{1}{||d_k||} = +\infty, \quad (36)$$

then,

$$\liminf_{k \to \infty} ||g_k|| = 0 \quad (37)$$

Proof.
Denote $\rho_k = \frac{d_k^T g_k}{||g_k||||d_k||}$ In view of lemma 3 $d_k^T g_k \leq -||g_k||^2$. Therefore,

$$\rho_k \leq -\frac{||g_k||}{||d_k||}. \quad (38)$$

which yields

$$\rho_k^2 \geq \frac{||g_k||^2}{||d_k||^2}. \quad (39)$$

Let by contradiction that

$$\liminf{}_k \to \infty ||g_k|| \neq 0. \quad (40)$$

Then there exist $\lambda > 0$ such that

$$||g_k|| \geq \lambda > 0, \forall k \geq 1. \quad (41)$$

Thus,

$$\frac{\lambda^2}{||d_k||^2} < \frac{||g_k||^2}{||d_k||^2} \leq \rho_k^2 = \frac{(d_k^T g_k)^2}{||d_k||^2||g_k||^2} < \frac{(d_k^T g_k)^2}{\lambda^2||d_k||^2}. \quad (42)$$

From lemma 3

$$\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{||d_k||^2} < +\infty, \quad (43)$$

which implies that $\sum_{k=0}^{\infty} \frac{1}{||d_k||^2} < +\infty$ is convergent. This contradict the condition (36). Hence the proof is complete.

Theorem 1. *Let $\{d_k\}$ and $\{\alpha_k\}$ be two sequences generated by Algorithm 2.1, where α_k is obtained via standard Wolfe condition. Suppose that Assumptions A(i) and A(ii) hold. If g is a uniformly convex function on S, i.e there exists a constant $\nu > 0$ such that*

$$(g(x) - g(y))^T (x - y) \geq \nu ||x - y||^2, \quad , \; for\ all\ x, y \in N, \quad (44)$$

then,

$$\lim_{k \to \infty} ||g_k|| = 0. \quad (45)$$

Proof. By Lipchitz continuity, we know that $||y_k|| \leq L||s_k||$. On the other hand by uniform convexity, it yields

$$y_k^T s_k \geq \nu ||s_k||^2. \quad (46)$$

Thus,

$$|\delta_k| \leq \frac{|s_k^T g_{k+1}|}{|y_k^T s_k|} + \frac{||y_k||^2 |s_k^T g_{k+1}|}{|y_k^T s_k|^2} + \frac{|y_k^T g_{k+1}|}{|y_k^T s_k|}, \quad (47)$$

$$\leq \frac{\lambda}{\nu||s_k||} + \frac{L^2 \lambda}{\nu^2||s_k||} + \frac{L\lambda}{\nu||s_k||}, \quad (48)$$

$$|\delta| = \frac{\lambda}{\nu}(1 + L + \frac{L^2}{\nu})\frac{1}{||s_k||}. \quad (49)$$

Since

$$|\eta_k| = \frac{|s_k^T g_{k+1}|}{|y_k^T s_k|} \leq \frac{||s_k||||g_{k+1}||}{\nu||s_k||^2} \leq \frac{\lambda}{\nu||s_k||}, \quad (50)$$

Finally

$$||d_{k+1}|| \leq ||g_{k+1}|| + |\delta_k|||s_k|| + |\eta_k|||y_k|| \leq \lambda + \frac{\lambda}{\nu}(2 + L + \frac{L^2}{\nu}). \quad (51)$$

Hence, d_{k+1} is bounded. In view of Lemma 3, (45) holds true. And the proof is completed.

4 Numerical performance of the STTCG and comparison

In this section we report some numerical results obtained with an implementation of the STTCG algorithm. The code is written in MATLAB R2013a on a PC with Intel COREi5 processor with 4GB of RAM and CPU 1.80GHz. We used 10 test problems with dimension between 100 to 10000 to test the performance of the proposed methods in terms of the number of iterations (iter) and the CPU time (in seconds). We define a termination criterion for the methods as

$$\|F(\mathrm{x}_k)\| < 10^{-4}. \quad (52)$$

The tables list the 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. We also used ”-” to represent failure during iteration process due one of the following:

1. The number of iteration and/or the CPU time in second reaches 1000;
2. Failure on code execution due to insufficient memory;
3. If $||F_k||$ is not a number (NaN).

We present here the benchmark problems used to test the proposed methods in this research.

Problem 1. *The strictly convex function* [45]:
$F_i(x) = e^{x_i} - 1$
$i = 1,2,...,n.$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 2. System of n nonlinear equations[45]:
$F_i(x) = x_i - 3x_i(\frac{sinx_i}{3} - 0.66) + 2$
$i = 2,3,...,n.$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 3. System of n nonlinear equations [28]:
$F_i(x) = \cos x_1 - 9 + 3x_1 + 8e^{x_2},$
$F_i(x) = \cos x_i - 9 + 3x_i + 8e^{x_{i-1}},$
$i = 1,2,...,n$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 4. System of n nonlinear equations:
$F_i(x) = (0.5 - x_i)^2 + (n+1-i)^2 - 0.25x_i - 1,$
$F_n(x) = \frac{n}{10}1 - e^{-x_n^2},$
$i = 1,2,...,n-1.$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 5. System of n nonlinear equations [2]:
$F_i(x) = 4x_i + x_{i+1} - 2x_i - x_{\frac{i+1}{3}},$
$F_n(x) = 4x_n + x_{n-1} - 2x_n - x_{\frac{n+1}{3}},$
$i = 1,2,...,n-1.$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 6. System of n nonlinear equations [48]:
$F_i(x) = x_i^2 - 4,$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 7. System of n nonlinear equations [5]:
$F_1(x) = \sin(x_1 - x_2) - 4e^{2-x_2} + 2x_1,$
$F_i(x) = \sin(2 - x_i) - 4e^{x_i - 2} + 2x_i + \cos(2 - x_i) - e^{2-x_i},$
$i = 2,3,...,n.$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 8. System of n nonlinear equations [45]:
$F_i(x) = \sum_{i=1}^n x_i x_{i-1} + e^{x_{i-1}} - 1,$
$i = 2,3,...,n$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

Problem 9. System of n nonlinear equations [48]:
$F_i(x) = x_i - \sum_{i=1}^n \frac{x_i^2}{n^2} + \sum_{i=1}^n x_i - n,$
$x_0 = (0.5,0.5,0.5,...,0.5)^T$
$i = 1,2,...,n.$

Problem 10. System of n nonlinear equations [48]:
$F_i(x) = 5x_i^2 - 2x_i - 3,$
$i = 1,2,...,n$
$x_0 = (0.5,0.5,0.5,...,0.5)^T.$

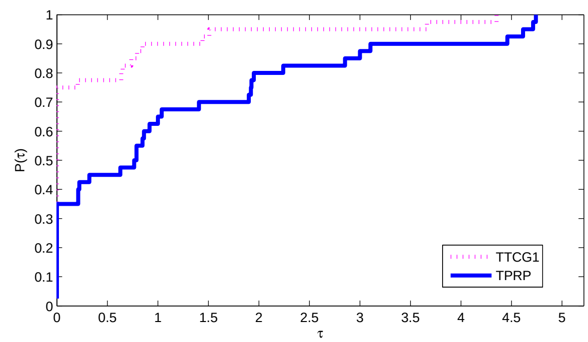


Fig. 1: Performance profile of STTCG and TPRP methods with respect to number of iterations of problems 1-10

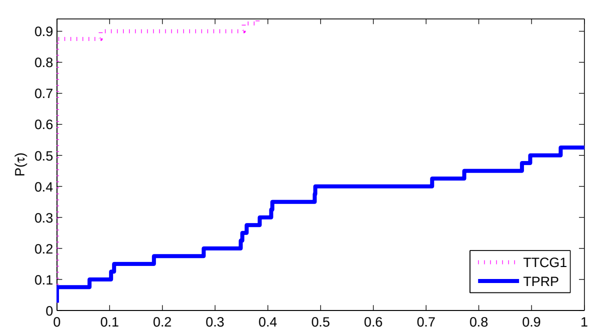


Fig. 2: Performance profile of STTCG and TPRP methods with respect to CPU time in seconds of problems 1-10

Table 1: Numerical results based on dimension of problem (n), Number of iterations and CPU Time (in seconds) of problems 1-6

	STTCG Algorithm			TPRP Algorithm			
P	Dim	Iter	$\|\|F_k\|\|$	CPU time	Iter	$\|\|F_k\|\|$	CPU time
	100	4	1.5264e-08	0.003181	88	9.6201e-05	0.031619
1	1000	4	4.8268e-08	0.015766	98	9.20E-05	0.103898
	5000	4	1.0793e-07	0.084453	105	8.91E-05	0.414489
	10000	4	1.5264e-07	0.163825	107	9.92E-05	0.80703
	100	9	6.4647e-05	0.005891	17	8.1857e-05	0.009647
2	1000	10	4.0993e-05	0.022897	18	5.9288e-05	0.042653
	5000	10	9.1664e-05	0.111459	20	5.6441e-05	0.216124
	10000	11	2.5994e-05	0.219785	20	7.9820e-05	0.375422
	100	26	6.9265e-05	0.014515	97	9.4677e-05	0.061587
3	1000	28	6.8690e-05	0.059455	106	8.9246e-05	0.224129
	5000	29	8.6013e-05	0.240648	112	8.9051e-05	1.051879
	10000	30	6.8119e-05	0.430044	114	9.6236e-05	1.857360
	100	16	9.9520e-05	0.007855	20	6.3527e-05	0.01041
4	1000	18	6.3729e-05	0.035541	21	7.1739e-05	0.045256
	5000	19	6.4126e-05	0.135590	22	3.6882e-05	0.190270
	10000	19	9.0688e-05	0.261424	22	5.2159e-05	0.36712
	100	14	4.6340e-05	0.008503	14	7.5054e-05	0.009104
5	1000	15	5.8615e-05	0.037034	15	9.6216e-05	0.039770
	5000	16	5.2427e-05	0.143230	16	5.5932e-05	0.180076
	10000	16	7.4143e-05	0.274368	16	7.9100e-05	0.350064
	100	10	3.6828e-05	0.005975	17	2.1584e-05	0.011010
6	1000	11	2.3292e-05	0.025098	17	6.8254e-05	0.030440
	5000	11	5.2083e-05	0.108580	19	5.4749e-05	0.144925
	10000	11	7.3657e-05	0.199006	19	7.7426e-05	0.255382

Table 2: Numerical results based on dimension of problem (n), Number of iterations and CPU Time (in seconds) of problems 7-10

	STTCG Algorithm			TPRP Algorithm			
P	Dim	Iter	$\|\|F_k\|\|$	CPU time	Iter	$\|\|F_k\|\|$	CPU time
	100	143	9.7928e-05	0.060383	7	-	0.007176
7	1000	152	9.4715e-05	0.216088	12	-	0.068942
	5000	144	9.6823e-05	0.786077	53	9.9910e-05	0.614886
	10000	148	9.6419e-05	1.479564	52	9.5775e-05	1.136529
	100	45	8.5992e-05	0.034983	39	5.0179e-05	0.039739
8	1000	20	9.1045e-05	0.080477	53	5.8972e-05	0.228211
	5000	23	9.0727e-05	0.396575	184	9.8877e-05	3.811658
	10000	9	7.7739e-06	0.409578	65	6.9965e-05	2.735592
	100	10	7.3224e-05	0.007216	86	9.5750e-05	0.061622
9	1000	37	8.8631e-05	0.068108	76	9.3339e-05	0.23127
	5000	42	8.3241e-05	0.331232	27	9.6663e-05	0.345739
	10000	22	8.7128e-05	0.424679	104	9.9467e-05	2.175488
	100	22	9.1535e-05	0.009544	14	4.7435e-05	0.010289
10	1000	25	6.2523e-05	0.031216	15	8.2750e-06	0.030116
	5000	26	8.3884e-05	0.134385	15	1.8503e-05	0.126861
	10000	27	7.1178e-05	0.26030	15	2.6168e-05	0.339827

We have tested our proposed method STTCG by solving twenty (10) benchmark nonlinear systems of equations. The results in Tables 1 and 2 demonstrated very clearly that the use of STTCG has reduced the number of iterations and CPU time for solving the tested problems compared to TPRP.This happened due to the low computational cost of the method, which was achieved by incoporating acceleration scheme and a Powell restart in to the algorithm.

In Table 1 and 2, we listed numerical results.The numerical results indicated that the proposed method, STTCG, compared to TPRP has minimum number of iterations and CPU time. Except for problems 7 and 10 where the TPRP has less number of iterations than STTCG. Figures 1 and 2 are performance profiles derived by Dolan and More [44] which show that our claim is justified, that is, less CPU time and number of iterations for each test problem with the exception of problems 7, and 10. Furthermore, on average, our $||F_k||$ is small which signifies that the solution obtained is true approximation of the exact solution compared to the TPRP.

5 Conclusion

In this paper an alternative three-term conjugate gradient algorithm for solving nonlinear system of equations as modification of BFGS quasi-Newton update with descent direction has been presented. Intensive numerical experiments on some benchmark nonlinear system of equations of different characteristics indicated that the suggested method i.e STTCG are faster and more efficient compared to TPRP [5]. The global convergence of the proposed method is also provided using general Wolfe line search.

References

[1] Q. Li, D. Hui L, *A class fo derivative free- methods for large-scale nonlinear monotone equations* journal of Numerical Analysis (2011) 31 1625-1635.

[2] W. Leong, M.A. Hassan, M. Y. Waziri, *A matrix-free quasi-Newton method for solving nonlinear systems*. Computers and Mathematics with Applications (2011) 62 2354-2363.

[3] W. Cheng, *A PRP type method for systems of monotone equations*,Journal of Computational and Applied Mathematics (2009) 50 15-20.

[4] C.G. Broyden, *A class of methods for solving nonlinear simultaneous equations*. Math. Comput. (1965) 19 577-593.

[5] G. Yuan, M. Zhang, *A three-terms Polak-Ribiere-Polyak conjugate gradient algorithm for large-scale nonlinear equations* Journal of Computational and Applied Mathematics (2015) 286 186-195.

[6] M. R. Hestenes, and E. L. Stiefel, *Methods of conjugate gradients for solving linear systems*. J. Research Nat. Bur. Standards, (1952) 49 409-436.

[7] R. Fletcher and C. Reeves, *Function minimization by conjugate gradients* Comput. J. (1964) 7 149-154.

[8] Y.H. Dai, Y. Yuan, *A nonlinear conjugate gradient method with a strong global convergence property* SIAM Journal on Optimization, (1999) 10 177182.

[9] W.W. Hager, H. Zhang, *A survey of nonlinear conjugate gradient methods* Pacific Journal of Optimization (2006) 2 3558.

[10] N. Andrei, *A simple three-term conjugate gradient algorithm for unconstrained optimization*. Journal of Computational and Applied Mathematics, (2013) 241 19-29.

[11] N. Andrei, On three-term conjugate gradient algorithms for unconstrained optimization. Applied Mathematics and Computation, (2013) 219 6316-6327.

[12] S. Deng, Z. Wan, *A three-term conjugate gradient algorithms for large-scale unconstrained optimization problems*. Applied Numerical Mathematics, (2015), http:dx.doi.org/10.1016/j.apnum.2015.01.008

[13] Y.F. Hu, C. Storey, *Global convergence result for conjugate gradient methods* J. Optim. Theory Appl. (1991) 71 399-405.

[14] J.K. Reid, *On the method of conjugate gradients for the solution of large sparse systems of linear equations*. In J.K. Reid (Ed.) Large Sparse Sets of Linear Equations, pages 231-254, Academic Press, 1971.

[15] J. W. Daniel, *The conjugate gradient method for linear and nonlinear operator equations* SIAM J. Numer. Anal., (1967) 4 10-26.

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

[17] E. Polak and G. Ribiere, Note sur la convergence de directions conjugates Rev. Francaise Informat Recherche Opertionelle, Annee (1969) 16 35-43.

[18] R. Fletcher, *Practical Methods of Optimization* vol. 1 Unconstrained Optimization, John Wiley & Sons, New York, 1987.

[19] H. P. Crowder and P. Wolfe, *Linear convergence of the conjugate gradient method* IBM J. Res. Dev., (1969) 16 431-433.

[20] M. J. D. Powell, *Some convergence properties of the conjugate gradient method* Math. Prog., (1976) 11 42-49.

[21] G. Zoutendijk, *Nonlinear Programming, Computational Methods, in Integer and Nonlinear Programming* J. Abadie, ed., North-Holland, Amsterdam, (1970) 37-86.

[22] M. Al-Baali, *Descent property and global convergence of the Fletcher-Reeves method with inexact line search* IMA J. Numer. Anal., (1985), 5 121-124.

[23] M. J. D. Powell, *Nonconvex minimization calculations and the conjugate gradient method*, Numerical Analysis (Dundee, 1983), Lecture Notes in Mathematics, Springer-Verlag, Berlin, (1984) 1066 122-141.,

[24] Y. Yuan, *Analysis on the conjugate gradient method*, Optimization Methods Software, 2 (1993) 19-29.

[25] Y. H. Dai, *Analysis of conjugate gradient methods*, Ph.D. thesis, Institute of Computational Mathematics and Scientific/Engineering Computing, Chinese Academy of Sciences, 1997.

[26] D. Touati-Ahmed and C. Storey, *Efficient hybrid conjugate gradient techniques*, Journal of Optim. Theory Appl., (1990) 64 379 - 397.

[27] J. C. Gilbert and J. Nocedal, *Global convergence properties of conjugate gradient methods for optimization*, SIAM J. Optim., (1992) 2 21- 42.

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

[29] E.M.L. Beale, *A derivation of conjugate gradients*, in: F.A.,Lootsma(Ed.), Numerial Methods for Nonlinear Optimization, Academic Press, London, (1972) 39-43.

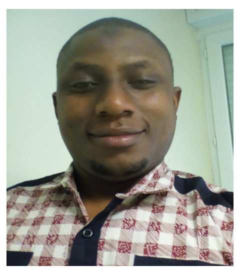
[30] M.F. McGuire, P. Wolfe, *Evaluating a restart procedure for conjugate gradients*. Report RC-4382 IBM Research Center, Yorktown Heights, 1973.

[31] L. Nazareth, *A conjugate direction algorithm without line search*, J. Optim. Theor. Appl. (1977) 23 373 387.

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

[33] L. Zhang, W. Zhou, D.H. Li, *Some descent three-term conjugate gradient methods and their global convergence,* Optim. Methods Software (2007) 22 697 711.
[34] L. Zhang, Y. Zhou, *A note on the convergence properties of the original three-term HestenesStiefel method,* AMO Adv. Model. Optim. (2012) 14 159 - 163.
[35] J. Zhang, Y. Xiao, Z. Wei, *Nonlinear conjugate gradient methods with sufficient descent condition for large-scale unconstrained optimization,* Mathematical Problems in Engineering, vol. 2009 (2009) Article ID 243290. http://dx.doi.org/10.1155/2009/243290.
[36] A.Y. Al-Bayati, W.H. Sharif, *A new three-term conjugate gradient method for unconstrained optimization,* Can. J. Sci. Eng. Math. (2010) 1 (5) 108 124.
[37] W. Cheng, A two-term PRP-based descent method, Numer. Funct. Anal. Optim. (2007) 28 1217 1230.
[38] Y. Narushima, H. Yabe, J.A. Ford,*A three-term conjugate gradient method with sufficient descent property for unconstrained optimization,* SIAM J. Optim. (2011) 21 (1) 212 230.
[39] N. Andrei, *A modified Polak Ribire Polyak conjugate gradient algorithm for unconstrained optimization,* Optimization (2011) 60 14571471.
[40] N. Andrei, *Acceleration of conjugate gradient algorithms for unconstrained optimization,* Appl. Math. Comput. (2009) 213 361369.
[41] M. J. D. Powell, Restart procedures of the conjugate gradient method. Math. Program., (1977) 2 241 - 254.
[42] 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. 152172, 2000.
[43] Y.Xiao, C.Wu, S.Yiwu, *Norm descent conjugate gradient methods for solving symmetric nonlinear equations,* J. of Glob Optim (2014) 14
[44] E. D. Dolan and J. J. More, *Benchmarking optimization software with performance profiles,* Mathematical Programming, vol. 91, no. 2 (2002) 201213.
[45] M.Y. Waziri, J. Sabiu. *An alternative conjugate gradient approach for large-scale symmetric nonlinear equations,* 6(5): 855, 2016.
[46] Y. H. Dai and Y. Yuan, *An efficient hybrid conjugate gradient method for unconstrained optimization,* Ann. Oper. Res., (2001) 103 33-47.
[47] Y. H. Dai, *A nonmonotone conjugate gradient algorithm for unconstrained optimization,* J. Syst. Sci. Complex., (2002), 15 139-145.
[48] A. B. Abubakar,*ON Improved Broyden-Type Method For Systems of Nonlinear Equations*, M.Sc.Thesis BUk. (2014) 5154.
[49] M.V. Solodov, B.F. Svaiter, *A globally convergent inexact Newton method for systems of monotone equations*, in: M. Fukushima, L. Qi (Eds.), Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, Kluwer Academic Publishers, 1998, pp. 355369.
[50] M. Raydan, *The Barzilaiai and Borwein gradient method forthe large scale unconstrained minimization problem*, SIAM Journal on Optimization. 7 (1997) 26-33.
[51] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, USA 1970.
[52] W. Cheng et.al, *A family of derivative-free conjugate gradient methods for large-scale nonlinear systems of equations*, Journal of Computational and Applied Mathematics 222 (2009) 11-19.

Mohammed Yusuf Waziri received his B.Sc., M.Sc. in Applied Mathematics from Bayero University Kano (BUK), Kano, Nigeria and his PhD in Applied Mathematics from Universiti Putra Malaysia in 2011. His research interest and current publications are on unconstrained optimization problems, nonlinear equations and fuzzy optimization. He is a Senior lecturer in Mathematics Department at Bayero University Kano (BUK), Kano, Nigeria and he has done Post-Doctoral Research at university Putra Malaysia 2012.



Lawal Muhammad is an assistant lecturer at Northwest University, kano. He obtained Bachelor Degree in Mathematics from Kano University of Science and Technology, Kano (Nigeria) in 2011, M.Sc. Mathematics from Bayero University, Kano Nigeria (2016). And also currently undergoing Master in Mathematical Physics at University of Burgundy, Dijon, France. His researches are in the fields of algebra of rhotrix, integral transforms, and unconstrained optimization problems.