A Projected Hybrid Conjugate Gradient Method for Solving Large-scale System of Nonlinear Equations

*Jamilu Sabi’u a and Abubakar M Gadub

a Department of Mathematics Northwest University Kano, Kano, Nigeria
b Department of Mathematics and Computer Science, Umaru Musa Yar’adua University, Katsina, Nigeria

*Corresponding author: sabiujsamilu@gmail.com

Received: 19/09/2018, Accepted: 22/11/2018

Abstract

In this article, a fully derivative-free projected hybrid conjugate gradient method for solving large-scale systems of nonlinear equations is proposed. The proposed method is the convex combination of FR and PRP conjugate gradient methods with the projection method. However, the global convergence of the given method is established under suitable conditions with nonmonotone line search. Numerical results show that the method is efficient for large-scale problems.

Keywords: backtracking line search; secant equation; projection method; global convergence; hybrid cg.

Introduction

We consider the nonlinear systems of equations

\[ F(x) = 0, \]

where \( F: \mathbb{R}^n \rightarrow \mathbb{R}^n \) is a continuously differentiable mapping. The prominent method for finding the solution of (1), is the classical Newton’s method and its variants that included Birgin et al. (2003), Brown and Saad (1994), Zhou and Li (2007), Zhou and Toh (2005) which generates a sequence of iterates

\[ x_{k+1} = x_k - (F'(x_k))^{-1}F(x_k), \]     \hspace{1cm} (2)

where \( k = 0,1,2, \ldots \). The attractive features of these methods are; rapid convergence and easily to implement. However, they are typically unattractive for large-scale problems because they need to solve \( n \) linear system of equations at each iteration using the Jacobian matrix or an approximation of it. However, in the solutions of large-scale unconstrained optimization problems, the conjugate gradient methods are particularly welcome due to their simplicity and lower storage requirement.

More recently, Waziri and Sabi’u (2015) introduced a derivative-free conjugate gradient method and its global convergence for solving symmetric nonlinear equations based on non-monotone line search methods, numerical results showed their method is efficient. Not withstanding, Cheng and Li (2009) extended the non-monotone line search method proposed by Zhang and Hager (2004) to the spectral residual method to solve large-scale nonlinear systems of equations.

Consequently, this paper is organised as follows: Next section is the details of proposed method. Convergence result is presented in Section 3. Some numerical results are reported in Section 4. Finally, conclusion was made in Section 5.

A Projected Hybrid CG Method

Given an initial point \( x_0 \), an iterative scheme for problem (1) generally generates a sequence of iterates

\[
x_k = x_{k-1} + \alpha_k d_{k-1}, \quad k = 1, 2, \ldots
\]

which employs a line search procedure along the direction \( d_k \) to compute the stepsize \( \alpha_k \). Let \( z_k = x_{k-1} + \alpha_k d_{k-1} \), then the hyperplane

\[
H_k = \{x \in \mathbb{R}^n | (x - z_k)^T F(z_k) = 0\}
\]

strictly separates \( x_k \) from the solution set of (1). Therefore, from this facts Solodov and Svaiter (1998) advised to let the next iterate \( x_{k+1} \) be the projection of \( x_k \) onto this hyperplane \( H_k \). Therefore, \( x_k \) is now defined as

\[
x_k = x_{k-1} - \frac{F(x_k)^T(x_{k-1} - z_k)}{||F(x_k)||^2} F(z_k).
\]

The direction \( d_k \) is given by

\[
d_k = \begin{cases} 
-F_k & \text{if } k = 0 \\
-F_k + \beta_k^\text{FR} d_k & \text{if } k \geq 1
\end{cases}
\]

\( F_k \) means \( F(x_k) \) and \( \beta_k^{\text{H+}} \) is given as

\[
\beta_k^{H+} = (1 - \sigma_k)\beta_k^{\text{FR}} + \sigma_k \beta_k^{\text{PRP}}
\]

where

\[
\beta_k^{\text{FR}} = \frac{||F_k||^2}{||F_{k-1}||^2} \quad \text{and} \quad \beta_k^{\text{PRP}} = \frac{F_k^T y_k}{||F_{k-1}||^2}
\]

\( y_k = F_k - F_{k-1} \), Fletcher and Reeves (FR), Polak, Ribiere and Polyak (PRP) and \( \sigma_k \) is a scalar satisfying \( 0 \leq \sigma_k \leq 1 \).

Observe that, the selection of \( \sigma_k \in [0,1] \) is random selection. However, to obtain the best selection of \( \sigma_k \forall k \), we employ the following procedure;

\[
-J^{-1} F_k = -F_k + (1 - \sigma_k) \frac{||F_k||^2}{||F_{k-1}||^2} s_k + \sigma_k \frac{F_k^T y_k}{||F_{k-1}||^2} s_k,
\]

after some algebraic manipulations, we get

\[
\sigma_k = \frac{\frac{s_k^T F_k - s_k^T k F_k^*}{||F_{k-1}||^2} ||F_{k-1}||^2 s_k}{\frac{F_k^T F_{k-1}}{||F_{k-1}||^2} s_k}
\]

To get rid of the Jacobian matrix, we used the modified secant equation in Kafaki and Ghanbari (2014) given by,
\[ J_k^{-1} y_k = 2 \frac{||y_k||^2}{s_k^2} s_k \]  

or equivalently

\[ J_k s_k = \frac{1}{2} \frac{s_k^T y_k}{||y_k||^2} y_k = v_k \]  

substituting (12) in (10), the following hybridization parameter is obtained

\[ \sigma_k = \frac{(s_k-v_k)^T F_k ||F_{k-1}||^2 + v_k^T s_k ||F_k||^2}{v_k^T s_k^T F_{k-1}} \]  

It is vital to note that, the hybridization parameter \( \sigma_k \) given by (13) may be outside the interval \([0,1]\). However, in order to have convex combination in (7), we adopt the consideration of Andrei (2008) in the sense that if \( \sigma_k < 0 \), then we let \( \sigma_k = 0 \), and if \( \sigma_k > 1 \), then we let \( \sigma_k = 0 \).

However, in order to compute the stepsize \( \alpha_k \), the nonmonotone line search used in Sabi’u and Waziri (2017) is an interesting idea that avoids the necessity of \( d_k \) to be a descent one. Let \( \omega_1 > 0, \omega_2 > 0, r \in (0,1) \) be constants and \( \{\eta_k\} \) be a given positive sequence such that

\[ \sum_{k=0}^{\infty} \eta_k < \infty. \]  

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

\[ ||F(x_k + \alpha_k d_k)||^2 - ||F(x_k)||^2 \leq -\omega_1 ||\alpha_k F(x_k)||^2 - \omega_2 ||\alpha_k d_k||^2 + \eta_k ||F(x_k)||^2. \]  

A Projected Hybrid Algorithm (PHGC)

Step 1: Given \( x_0, \alpha > 0, \omega \in (0,1), r \in (0,1) \) and a positive sequence \( \eta_k \) satisfying (14), and set \( k = 0 \).

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

Step 3: Compute \( d_k \) by (6).

Step 4: Compute \( \alpha_k \) by the line search (15).

Step 5: Compute \( x_{k+1} = x_{k-1} - \frac{F(z_k)^T(x_{k-1} - z_k)}{||F(z_k)||^2} F(z_k). \)

Step 6: Consider \( k = k + 1 \) and go to step 2.

Global Convergence

In order to get the global convergence of projected hybrid CG algorithm, we need the following assumption.

**Assumption**

(i) The level set \( \Omega = \{x | F(x) \leq e^x F(x_0)\} \) is bounded

(ii) In some neighborhood \( N \) of \( \Omega \), \( F(x) \) is Lipschitz continuous i.e there exists a positive constant \( L > 0 \) such that

\[ ||F(x) - F(y)|| \leq L ||x - y||, \]  

for all \( x, y \in N \).

**Lemma 3.1** Li and Fukushima (1999). Let \( \{x_k\} \) be generated by the projected hybrid algorithm. Then \( \{x_k\} \in \Omega \).

**Lemma 3.1** Suppose that \( \{x_k\} \) is generated by projected hybrid algorithm. Let \( s_k = x_k - x_{k-1}. \) Then, we have

\[ \lim_{k \to \infty} ||\alpha_k d_k|| = \lim_{k \to \infty} ||s_k|| = 0, \]  

**Proof.**

\[ ||x_k - x_{k-1}|| = \frac{||F(x_k)^T(x_{k-1} - z_k)||}{||F(z_k)||} \leq \frac{||F(z_k)|| ||x_{k-1} - z_k||}{||F(z_k)||} \]

\[ = ||x_k - z_k|| = \alpha_k ||d_k||. \]
Also is not difficult enough to see that, if \( \| F_k \| \leq M \) then \( \| F(z_k) \| \leq M' \) for some \( M, M' > 0 \).

The following theorem establishes the global convergence of the projected hybrid CG method.

**Theorem 3.2** Let \( \{x_k\} \) be generated by projected hybrid algorithm. Then, we have
\[
\liminf_{k \to \infty} \| F(x_k) \| = 0.
\]  

**Proof.** We prove this theorem by contradiction. Suppose that (20) does not hold, then there exists a positive constant \( \tau \) such that
\[
\| F(x_k) \| \geq \tau, \quad \forall k \geq 0.
\]  

Clearly, \( \| F_k \| \leq \| d_k \| \), which implies
\[
\| d_k \| \geq \tau, \quad \forall k \geq 0.
\]  

Observe that,
\[
\| y_k \| = \| F(x_k) - F(x_{k-1}) \| \leq L \| s_k \|
\]  

Therefore, from (12), and the fact that \( \| F(z_k) \| = \| M \| \) we have
\[
\| v_k \| = \| \frac{s_k^T y_k}{2 \| y_k \|^2} y_k \| \leq \frac{\| s_k \|}{2}.
\]  

Furthermore,
\[
| \sigma_k | \leq | \frac{(s_k-v_k)^T F_k \| F_k \| - s_k} {v_k^T s_k F_k \| F_k \| - s_k} | \to 0
\]  

meaning there exists a constant \( \lambda \in (0,1) \) such that for sufficiently large \( k \),
\[
| \sigma_k | \leq \lambda.
\]  

Again from the definition of our \( \beta_k^* \) we obtain
\[
| \beta_k^* | \leq | (1 - \sigma_k) \frac{\| F_k \|}{\| F_{k-1} \|} + | \sigma_k | \frac{\| F_k \|}{\| F_{k-1} \|} \leq 2M \| y_k \| \to 0
\]  

which implies there exists a constant \( \rho \in (0,1) \) such that for sufficiently large \( k \)
\[
| \beta_k^* | \leq \rho.
\]  

Without lost of generality, we assume that the above inequalities holds for all \( k \geq 0 \). Then we get
\[
\| d_k \| \leq \| F_k \| + | \beta_k^* | \leq M + \rho \| d_k \|
\]  

which shows that the sequence \( \{d_k\} \) is bounded.
This together with Lemma 3.1 and inequalities \( \| F_k \| \geq \tau, \| F_k \| \geq \tau \) implies for all \( k \) sufficiently large,
\[
\| d_k \| \geq M \| d_k \| > 0.
\]  

The last inequality yields a contradiction with (21). Consequently, (20) holds. The proof is complete.

**Numerical Results**

In this section, the performance of projected hybrid conjugate gradient (PHCG) was compared with three-terms Polak-Ribiere-Polyak conjugate gradient method Gonglin and Maojun (2015) and Inexact PRP conjugate gradient method Xhou and Shen (2014).

- For projected hybrid CG method (DFCG): \( \omega_1 = \omega_2 = 10^{-4}, r = 0.2 \) and \( \eta_k = \frac{1}{(k+1)^2} \).
- For a three terms PRP (TPRP): \( \omega = 10^{-4}, r = 0.2 \) and \( s = 1 \).
- For an inexact PRP (IPRP): \( \omega_1 = \omega_2 = 10^{-4}, \alpha = 0.02, r = 0.2 \) and \( \eta_k = \frac{1}{(k+1)^2} \).

The code for all the three methods were written in Matlab 7.4 R2010a and run on a personal computer 1.8 GHz CPU processor and 4 GB RAM memory. Iterations were stopped if the total number of iterations exceeds 2000 or \( \| F_k \| \leq 10^{-4} \). Nine benchmark test problems with different initial points and \( n \) values were tested. Problems 1-3 are from Waziri and Sabi’u (2015) while the remaining are from Lacruz et al. (2004).
Problem 1. The strictly convex function:
\[ F_i(x) = e^{x_i} - 1 \quad ; 1, 2, \ldots, n \]

Problem 2: (n is multiple of 3) for \( i = 1, 2, \ldots, n/3, \)
\[ F_{3i-2}(x) = x_{3i-2}x_{3i-1} - x^2_{3i} - 1, \]
\[ F_{3i-1}(x) = x_{3i-2}x_{3i-1}x_{3i} - x^2_{3i-2} + x^2_{3i-1} - 2, \]
\[ F_{3i}(x) = e^{-x_{3i-2}} - e^{-x_{3i-1}}. \]

Problem 3. The variable band function:
\[ F_i(x) = -2x^2_i + 3x_1 - 2x_2 + 0.5x_3 + 1 \]
\[ F_i(x) = -2x^2_i + 3x_i - x_{i-1} - 1.5x_{i+1} + 1 \quad \text{for} \ i = 2, 3, \ldots, n - 1 \]
\[ F_n(x) = -2x^2_n + 3x_n - 0.5x_{n-1} + 1 \]

Problem 4. The Exponential function:
\[ F_i(x) = \frac{i}{10} (1 - x^2_i - e^{-x^2_i}) \quad ; 1, 2, \ldots, n - 1 \]
\[ F_n(x) = \frac{n}{10} (1 - e^{-x^2_n}). \]

Problem 5. Trigonometric Function:
\[ F_i(x) = 2(n + i(1 - \cos x_i) - \sin x_i - \sum^n_{j=1} \cos x_j)(2\sin x_i - \cos x_i) \quad \text{for} \ i = 1, 2, \ldots, n \]

Problem 6. The Hanbook function:
\[ F_i(x) = 0.05(x_i - 1) + 2\sin(\sum^n_{j=1} (x_j - 1) + \sum^n_{j=1} (x_j - 1)^2(1 + 2(x_i - 1)) + 2\sin(\sum^n_{j=1} (x_j - 1)), \]
\[ \text{for} \ i = 1, 2, \ldots, n \]

Problem 7. The Tridiagonal system:
\[ F_1(x) = 4(x_1 - x_2^2) \]
\[ F_i(x) = 8x_i(x^2_i - x_{i-1}) - 2(1 - x_i) + 4(x_i - x^2_{i+1}) \quad \text{for} \ i = 2, 3, \ldots, n - 1 \]
\[ F_n(x) = 8x_n(x^2_n - x_{n-1}) - 2(1 - x_n) \]

Problem 8: The Broyden Tridiagonal function
\[ F_1(x) = (3 - 0.5x_1)x_1 - 2x_2 + 1 \]
\[ F_i(x) = (3 - 0.5x_i)x_i - x_{i-1} - 2x_{i+1} + 1 \]
\[ F_n(x) = (3 - 0.5x_n)x_n - x_{n-1} + 1 \]

Problem 9: The Generalised function of Rosenbrock
\[ F_1(x) = x_1 - e^{\cos(x_1 + x_2)_{n+1}} \]
\[ F_i(x) = x_i - e^{\cos(x_{i-1} + x_{i+1})_{n+1}} \]
\[ F_n(x) = x_n - e^{\cos(x_{n-1} + x_n)_{n+1}} \]
Table 1. Numerical comparison of PHCG versus TPRP and IPRP conjugate gradient methods, where \( e = \text{ones}(n, 1) \).

<table>
<thead>
<tr>
<th>Problem (P)</th>
<th>( x_0 )</th>
<th>( n )</th>
<th>PHCG ( \text{Iter} )</th>
<th>PHCG ( \text{Time} )</th>
<th>TPRP ( \text{Iter} )</th>
<th>TPRP ( \text{Time} )</th>
<th>IPRP ( \text{Iter} )</th>
<th>IPRP ( \text{Time} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1 ( e )</td>
<td>10</td>
<td>10</td>
<td>0.013312</td>
<td>34</td>
<td>0.156228</td>
<td>81</td>
<td>0.157954</td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>11</td>
<td>0.012343</td>
<td>39</td>
<td>0.087552</td>
<td>82</td>
<td>0.069933</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>12</td>
<td>0.026558</td>
<td>44</td>
<td>0.139673</td>
<td>83</td>
<td>0.170848</td>
<td></td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>13</td>
<td>5.726047</td>
<td>58</td>
<td>27.2775</td>
<td>87</td>
<td>27.44766</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.1e</td>
<td>10</td>
<td>0.004334</td>
<td>26</td>
<td>0.03682</td>
<td>5</td>
<td>0.005436</td>
<td></td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>5</td>
<td>0.009236</td>
<td>35</td>
<td>0.084312</td>
<td>5</td>
<td>0.00865</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>4</td>
<td>0.071098</td>
<td>40</td>
<td>0.454774</td>
<td>5</td>
<td>0.044675</td>
<td></td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>4</td>
<td>0.197998</td>
<td>45</td>
<td>2.737452</td>
<td>7</td>
<td>0.349126</td>
<td></td>
</tr>
<tr>
<td>P2 ( e )</td>
<td>10</td>
<td>8</td>
<td>0.015</td>
<td>10</td>
<td>0.025074</td>
<td>27</td>
<td>0.05829</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>12</td>
<td>0.045308</td>
<td>13</td>
<td>0.069558</td>
<td>35</td>
<td>0.162844</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>13</td>
<td>0.266608</td>
<td>15</td>
<td>0.446318</td>
<td>39</td>
<td>1.202523</td>
<td></td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>13</td>
<td>0.97505</td>
<td>16</td>
<td>1.637988</td>
<td>43</td>
<td>5.281537</td>
<td></td>
</tr>
<tr>
<td></td>
<td>500000</td>
<td>14</td>
<td>11.96578</td>
<td>17</td>
<td>16.38332</td>
<td>47</td>
<td>67.32499</td>
<td></td>
</tr>
<tr>
<td>P3 ( 0.001e )</td>
<td>50</td>
<td>26</td>
<td>0.039045</td>
<td>78</td>
<td>0.184094</td>
<td>46</td>
<td>0.07393</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>42</td>
<td>0.560787</td>
<td>92</td>
<td>1.410526</td>
<td>46</td>
<td>0.735011</td>
<td></td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>38</td>
<td>3.340754</td>
<td>100</td>
<td>8.990049</td>
<td>2000</td>
<td>190.7377</td>
<td></td>
</tr>
<tr>
<td>P4 ( e )</td>
<td>10</td>
<td>24</td>
<td>0.024274</td>
<td>28</td>
<td>0.033153</td>
<td>2000</td>
<td>1.308739</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>18</td>
<td>0.019433</td>
<td>14</td>
<td>0.176098</td>
<td>1604</td>
<td>1.139648</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.1e</td>
<td>10</td>
<td>0.021551</td>
<td>21</td>
<td>0.026414</td>
<td>2000</td>
<td>1.261194</td>
<td></td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>14</td>
<td>0.012014</td>
<td>10</td>
<td>0.014599</td>
<td>492</td>
<td>0.410706</td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1</td>
<td>0.001563</td>
<td>1</td>
<td>0.002624</td>
<td>3</td>
<td>0.004971</td>
<td></td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>1</td>
<td>0.003329</td>
<td>-</td>
<td>-</td>
<td>59</td>
<td>0.103045</td>
<td></td>
</tr>
<tr>
<td>P5 ( e )</td>
<td>20</td>
<td>10</td>
<td>0.021043</td>
<td>41</td>
<td>0.084674</td>
<td>13</td>
<td>0.023319</td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>22</td>
<td>0.085487</td>
<td>75</td>
<td>0.403924</td>
<td>14</td>
<td>0.033986</td>
<td></td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>17</td>
<td>0.157702</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Problem (P)</td>
<td>$x_0$</td>
<td>n</td>
<td>Iter</td>
<td>Time</td>
<td>Iter</td>
<td>Time</td>
<td>Iter</td>
<td>Time</td>
</tr>
<tr>
<td>------------</td>
<td>-------</td>
<td>----</td>
<td>------</td>
<td>--------</td>
<td>------</td>
<td>--------</td>
<td>------</td>
<td>--------</td>
</tr>
<tr>
<td>P5 e</td>
<td>20</td>
<td>10</td>
<td>0.021043</td>
<td>41</td>
<td>0.084674</td>
<td>13</td>
<td>0.023319</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>100</td>
<td>22</td>
<td>0.085487</td>
<td>75</td>
<td>0.403924</td>
<td>14</td>
<td>0.033986</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>500</td>
<td>17</td>
<td>0.157702</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>0.1e</td>
<td>10</td>
<td>5</td>
<td>0.009133</td>
<td>26</td>
<td>0.078424</td>
<td>9</td>
<td>0.013801</td>
<td></td>
</tr>
<tr>
<td>P6 0.1e</td>
<td>20</td>
<td>25</td>
<td>0.080709</td>
<td>7</td>
<td>0.022731</td>
<td>10</td>
<td>0.015425</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>10</td>
<td>108</td>
<td>0.407842</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>50</td>
<td>32</td>
<td>0.130477</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>100</td>
<td>20</td>
<td>0.075832</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>P7 0.1e</td>
<td>10</td>
<td>673</td>
<td>1.144507</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>15</td>
<td>618</td>
<td>0.964643</td>
<td>1563</td>
<td>3.395375</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>30</td>
<td>683</td>
<td>1.034815</td>
<td>1532</td>
<td>3.236133</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>60</td>
<td>559</td>
<td>1.01497</td>
<td>1524</td>
<td>3.168755</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>120</td>
<td>616</td>
<td>1.375322</td>
<td>1517</td>
<td>3.560148</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>500</td>
<td>761</td>
<td>2.472581</td>
<td>1538</td>
<td>5.625074</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>P8 e</td>
<td>5</td>
<td>57</td>
<td>0.078681</td>
<td>115</td>
<td>0.188794</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>10</td>
<td>72</td>
<td>0.115632</td>
<td>135</td>
<td>0.200729</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>15</td>
<td>62</td>
<td>0.089213</td>
<td>153</td>
<td>0.226583</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>0.1e</td>
<td>10</td>
<td>61</td>
<td>0.08101</td>
<td>136</td>
<td>0.259408</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>50</td>
<td>51</td>
<td>0.079408</td>
<td>146</td>
<td>0.225568</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Numerical comparison of PHCG versus TPRP and IPRP conjugate gradient methods cont.
<table>
<thead>
<tr>
<th>P9</th>
<th>Pe</th>
<th>100</th>
<th>110</th>
<th>0.169278</th>
<th>147</th>
<th>0.236581</th>
<th>-</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>71</td>
<td>0.152256</td>
<td>152</td>
<td>0.327622</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1900</td>
<td>80</td>
<td>0.2873</td>
<td>158</td>
<td>0.720556</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td>114</td>
<td>0.652422</td>
<td>160</td>
<td>1.140294</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P9</td>
<td>Pe</td>
<td>10</td>
<td>22</td>
<td>0.041775</td>
<td>30</td>
<td>0.041681</td>
<td>97</td>
<td>0.120178</td>
</tr>
<tr>
<td>50</td>
<td>68</td>
<td>0.063843</td>
<td>42</td>
<td>0.079043</td>
<td>7</td>
<td>0.008948</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>22</td>
<td>0.029566</td>
<td>44</td>
<td>0.072499</td>
<td>5</td>
<td>0.00754</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>17</td>
<td>0.045969</td>
<td>49</td>
<td>0.207139</td>
<td>3</td>
<td>0.013159</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5000</td>
<td>16</td>
<td>0.169529</td>
<td>53</td>
<td>0.76243</td>
<td>3</td>
<td>0.045772</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50000</td>
<td>16</td>
<td>0.994025</td>
<td>58</td>
<td>5.727897</td>
<td>1</td>
<td>0.12565</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500000</td>
<td>26</td>
<td>22.85605</td>
<td>63</td>
<td>88.6272</td>
<td>1</td>
<td>1.563131</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1. Performance profile of PHCG, TPRP and IPRP conjugate gradient methods as the dimension increases (in term of CPU time)
The tables above represented the numerical comparison of PHCG, TPRP and IPRP method in terms of number of iterations and CPU time in second. "Iter" and "Time" stand for the total number of all iterations and the CPU time in second respectively. However from the table PHCG has less number of iterations and CPU time in most of the problems compared to TPRP and IPRP methods. This is due to the introduction of the projection technique in our scheme and the good selection of the hybridization parameter $\sigma_k$. For more details on performance profile, see Dolan and Moore, (2002).

Moreover, Figures 1 and 2 are comparisons using the performance profile of Dolan and Moore’ that shows PHCG out performs TPRP and IPRP both in number of iteration and the CPU time as the dimension increases. This also shows the paramount advantages of the convex combination and also the logical selection of the convex parameter (hybridization parameter).

Conclusion

In this research article a projected hybrid conjugate gradient method for solving large-scale system of nonlinear equations was presented. The method is completely a derivative-free method with less number of iterations and CPU time compared to the existing methods. Using classical assumptions the global convergence was also proved. Numerical comparisons using a set of large-scale test problems show that the proposed method is promising. However to extend the method to general nonsmooth equations will be our further research.

Conflict of Interests

There is no conflict of interest regarding the publication of this paper.

References


