An application of conjugate gradient method under strong Wolfe line search for solving unconstrained optimization

Wan Khadijah\(^1\), Mohd Rivaie\(^2\), Mustafa Mamat\(^1\), Nurul Hajar\(^1\), Nurul ‘Aini\(^1\), Zubair’ah Zainal Abidin\(^1\)

\(^1\)Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA), 21300 Terengganu, Malaysia  
\(^2\)Department of Sciences and Mathematics, Universiti Teknologi MARA (UiTM), 21080 Terengganu, Malaysia  
*Corresponding author E-mail: wankhadijahws@gmail.com

Abstract

The conjugate gradient (CG) method is one of the most promising methods for solving linear and nonlinear problems in optimization. In this paper, we propose a CG method with sufficient descent property under strong Wolfe line search. The proposed CG method is then applied to solve systems of linear equations. The numerical results obtained from the tests are evaluated based on number iteration and CPU time and then analyzed through performance profile. In order to examine its efficiency, the performance of our CG formula is compared to that of other CG methods. The results show that the proposed CG formula has better performance than the other tested CG methods.

Keywords: Conjugate Gradient Method; Spectral Conjugate Gradient; Strong Wolfe Line Search.

1. Introduction

Consider the unconstrained optimization problem:

\[
\min_{x \in \mathbb{R}^n} f(x),
\]

where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is continuously differentiable. The CG methods are generated iteratively by:

\[
x_{k+1} = x_k + \alpha_k d_k \quad \text{where} \quad k = 0, 1, 2, \ldots,  
\]

with search direction, \( d_k \) defined by:

\[
d_k = \begin{cases} 
eg g_k, & \text{if} \quad k = 0, \\ -g_k + \beta_k d_{k-1}, & \text{if} \quad k \geq 1. 
\end{cases}
\]

Note that the symbol \( \| \cdot \| \) denotes the Euclidean norm of vectors and \( y_{k+1} = g_k - g_{k-1} \). The methods above are referred as HS [1], FR [2], PRP [3], LS [4], DY [5] and CD [6].

In [7, 8] published two new CG coefficients which are modified based on the HS method namely the RMIL and RMIL* methods. The methods are both easily defined and simple to implement. They are formulated as:

\[
\beta_k^{\text{RMIL}} = \frac{g_k^{T} y_{k+1}}{d_{k+1}^{T} s_{k+1}}, \quad \beta_k^{\text{RMIL}*} = \frac{g_k^{T} y_{k+1}}{\|d_{k+1}\|}.  
\]

These methods have good numerical results and are globally convergent with exact line search.

Other researchers, like [9] are interested to modify the PRP formula as
Their paper discusses the general convergence results for the proposed formula with some line searches such as the exact line search, the Wolfe–Powell line search and the Grippo–Lucidi line search. Recently, there are many researchers interested in the WYL approach. For further research, please refer to [10–14]. For good comparative studies on newer CG methods, please refer to [15–19]. For newers studies on the optimization method, please refer to [20–21].

Inspired by the spectral method proposed by [22–23] combined the advantages of spectral CG method and FR CG method and presented the modified FR (MFR) method. When applied with Armijo-type line search, MFR has been shown to possess global convergence properties. Its search direction is defined as follows:

\[ d_k = \begin{cases} -g_k, & \text{if } k = 0, \\ -\theta_k g_k + \beta_k^{\text{DY}} d_{k-1}, & \text{if } k \geq 1. \end{cases} \quad (1.5) \]

where \( \theta_k \) is written as

\[ \theta_k = \frac{d_k^T s_{k-1}}{\|s_{k-1}\|^2}. \]

The MFR method can be rewritten as:

\[ d_k = -\left(1 + \beta_k^{\text{DY}} \frac{g_k^T d_{k-1}}{\|s_k\|^2}\right)g_k + \beta_k^{\text{DY}} d_{k-1}. \quad (1.6) \]

In [24] also applied the DY conjugate gradient method in the MFR method where the \( \beta_k^{\text{DY}} \) is substituted in the (1.6) formula. In [25] proposed a spectral Kadjiah-Rivaie-Mustafa-Ismail (SKRMI) CG method under strong Wolfe line search to solve unconstrained optimization problems. The method is presented as follows:

\[ d_k = -\left(1 + \beta_k^{\text{RMIL}} \frac{g_k^T d_{k-1}}{\|s_k\|^2}\right)g_k + \beta_k^{\text{RMIL}} d_{k-1}. \]

Generally, most of the spectral method directly reduces to classical CG method if exact line search is used. In this paper, in [23] approach as written in equation (1.6) is implemented with the RMIL* classical CG method. This modification of RMIL* method is named as MRMIL* CG method. Under strong Wolfe type line search, MRMIL* method is proven to possess sufficient descent and global convergence properties. Other than that, the new proposed method is utilized for solving artificial nonlinear problems and systems of linear equations in order to test their efficiency and applicability in real-life problems. The new method is only tested on numerical experimental simulations as it is difficult to get actual data for real-life application.

In section 2, we present the new spectral CG method and its algorithm. In section 3, the proof of the proposed method which includes the convergence analysis of the modified RMIL* method is presented. Then, section 4 reports the numerical results and discuss the findings. In section 5, the proposed method is tested for solving systems of linear equations and the results obtained are discussed. Finally, the work of this paper is summarized in the last section.

2. New conjugate gradient algorithm

In this section, we present a modification of spectral CG method where the \( \beta_k^{\text{RMIL}} \) is applied. This method is motivated by [23].

The equation of this new method is shown as follows:

\[ d_k = \begin{cases} -g_k, & \text{if } k = 0, \\ -\beta_k^{\text{RMIL}} \frac{g_k^T d_{k-1}}{\|s_k\|^2} g_k + \beta_k^{\text{RMIL}} d_{k-1}, & \text{if } k \geq 1. \end{cases} \quad (1.21) \]

The following is the idea of MRMIL* algorithm.

Algorithm 1: Algorithm of modified RMIL*.

Step 1: Initialization. Given \( x_k \), set \( k = 0 \).

Step 2: Compute \( \beta_k \) based on \( \beta_k^{\text{RMIL}} \).

Step 3: Compute \( d_k \) based on (1.2). If \( g_k = 0 \), then stop.

Step 4: Compute \( \alpha_k \) based on (1.4).

Step 5: Update the new point based on (1.2).

Step 6: Convergence test and stopping criteria.

If \( f(x_{k+1}) < f(x_k) \) and \( \|g_k\| < \varepsilon \) then stop. Otherwise, go to Step 1 with \( k := k + 1 \).

3. The sufficient descent condition

The spectral CG method reduces to standard RMIL* method if exact line search is applied in the algorithm. This is because under exact line search, \( g_k^T d_k = 0 \). Hence, we will have

\[ \theta_k = 1 + \beta_k^{\text{RMIL}} \frac{g_k^T d_{k-1}}{\|s_k\|^2} = 1. \quad (3.1) \]

From there, it is obvious that the search direction of spectral CG method will be reduced to the standard CG method. For the sufficient condition to hold,

\[ g_k^T d_k \leq -C \|s_k\| \quad \text{for } k \geq 0 \text{ where } C > 0. \quad (3.2) \]

It is easy to see that for (2.1) the sufficient descent condition holds for any line search since \( g_k^T d_k = -\|s_k\| \).

Theorem 3.1: Let \( x_k \) and \( d_k \) be generated by the MRMIL*, then condition (3.2) holds for all \( k \geq 0 \).

Proof: If \( k = 0 \), then \( g_0^T d_0 = -\|s_0\| \). Hence, condition (3.2) holds true. We also need to show that for \( k \geq 1 \), condition (3.2) will also hold true. From (2.1), multiply with \( g_k^T \),

\[ g_k^T d_k = -\left(1 + \beta_k^{\text{RMIL}} \frac{g_k^T d_{k-1}}{\|s_k\|^2}\right)\|s_k\| + \beta_k^{\text{RMIL}} g_k^T d_{k-1} \]

\[ = -\|s_k\|^2 - \beta_k^{\text{RMIL}} \frac{g_k^T d_{k-1}}{\|s_k\|^2} \|s_k\| + \beta_k^{\text{RMIL}} g_k^T d_{k-1} \]

By applying strong Wolfe line search,

\[ = -\|s_k\| \left(1 - \beta_k^{\text{RMIL}} \frac{\|s_k\|}{\|s_k\|^2}\right) + \beta_k^{\text{RMIL}} g_k^T d_{k-1} \]
Thus, it is clearly shown that the sufficient descent condition is fulfilled. Hence, the proof is completed.

4. Results and Discussion

In this section, the numerical results for MRMIL* based on the number of iteration and CPU time are compared with other methods in order to analyze the efficiency of the proposed spectral methods.

We consider $\varepsilon = 10^{-6}$ and norm of gradient value as the stopping criteria. All the problems are solved by MATLAB R2011b and the calculations are performed by using Intel Core i5 processor with RAM 2GB and Windows 8 operation system.

Table 1 presents the list of problem functions selected from [26]. For each of the test function, four initial points are considered starting from a point that is close to the solution point to the point that is far from it, as suggested by [27].

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
<th>Dimension</th>
<th>Initial Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Extended White and Holst</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(-3, -3), (3, -3), (6, -6), (9, -9)</td>
</tr>
<tr>
<td>2</td>
<td>Extended Rosenbrock</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(13, -13), (16, -16), (20, -20), (30, -30)</td>
</tr>
<tr>
<td>3</td>
<td>Extended DENSCHNB</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(10, -10), (50, -50), (100, -100), (200, -200)</td>
</tr>
<tr>
<td>4</td>
<td>Extended Himmelblau</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(10, -10), (50, -50), (100, -100), (200, -200)</td>
</tr>
<tr>
<td>5</td>
<td>Quadratic QF1</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(10, -10), (50, -50), (100, -100), (200, -200)</td>
</tr>
<tr>
<td>6</td>
<td>Extended Tridiagonal 1</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(10, -10), (50, -50), (100, -100), (200, -200)</td>
</tr>
<tr>
<td>7</td>
<td>Extended Maratos</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(10, -10), (50, -50), (100, -100), (200, -200)</td>
</tr>
<tr>
<td>8</td>
<td>Extended DENSCHF</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(10, -10), (50, -50), (100, -100), (200, -200)</td>
</tr>
<tr>
<td>9</td>
<td>Diagonal 4</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(10, -10), (50, -50), (100, -100), (200, -200)</td>
</tr>
<tr>
<td>10</td>
<td>Raydan 2 Function</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(5, -5), (10, -10), (25, -25), (50, -50)</td>
</tr>
<tr>
<td>11</td>
<td>Extended Freudenstein and Roth</td>
<td>2,4, 10, 100, 500, 1000</td>
<td>(2, -2), (20, -20), (75, -75), (202, -202)</td>
</tr>
</tbody>
</table>

Then, the numerical results are analyzed via the performance profile which is first introduced by [28]. The performance results are shown in Figure 1 and Figure 2. This method of performance profile provide the mean results over the rest of the solvers.

The performance of solver $s$ on any given problem might be of significance, but we would like to acquire a general evaluation of the performance of the solver, then the following is defined:

$$\rho_s(\tau) = \frac{1}{n_p} \min \{p \in P : r_{ps} \leq \tau\}.$$  

Thus, $\rho_s(\tau)$ is the probability that the solver $s \in S$ that a performance ratio $r_{ps}$ is within a factor $\tau \in R$ of the best possible ratio.

The function $\rho_s(\tau)$ is the cumulative distribution function for the performance ratio. The performance profile $p_s : R \mapsto [0,1]$ for a solver is non-decreasing, piecewise and continuous from the right. The value $p_s(1)$ is the probability that the solver is successful over the rest of the solvers.

For better understanding about the performance profiles, note that the left side of the figure gives the percentage of the test problems for which method is the fastest; while the right side gives the percentage of the test problems that are successfully solved by each of the methods. The methods that solve the most of the problems in a time that is within a factor $\tau$ of the best time can be determined by the top curve of the graph in the figures.

In both figures, MFR, MDY, MCD, MRMIL and MRMIL* represent the spectral CG methods used to solve the test problems under strong Wolfe line search. Fig. 1 represents performance profile based on the number of iterations and Fig. 2 represents performance profile based on CPU time. From both figures, MRMIL solves 94.58%, MRMIL* solves 97.5%, MFR solves 80%, MDY solves 83.75% and MCD solves 80.42%. It is clear in both Fig. 1 and Fig. 2 that the performance of MRMIL* CG method is the best with respect to the number of iterations and CPU time. This is followed by MRMIL, MDY, MCD and lastly MFR method.

![Fig. 1: Performance Profile based on Iteration Number](image1)

![Fig. 2: Performance Profile based on CPU Time](image2)
5. Application of conjugate gradient method

In this section, the CG method is applied for solving the systems of linear equations. Consider a system of linear equations

\[ Ax = b, \]

where \( A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^n, m \geq n \) and rank \( A = n \). The matrix \( Ax = b \) can be solved directly by \( x = A^{-1}b \) if the matrix is a symmetric or square matrix. However, in cases involving non-square matrix problem, the following theorem can be used which is further discussed in this section.

**Theorem 5.1** [29]: The unique vector \( x^* \) that minimizes \( \|Ax - b\| \) is given by the solution of the equation \( A^TAx = A^Tb \); that is

\[ x = \left(A^TA\right)^{-1}A^Tb. \]

The proof of this theorem can be seen in [29]. The following problem is cited from [29], where Theorem 5.1 is applied to solve the problem effectively. The problem can be formulated as

\[ A = \begin{bmatrix} 0.3 & 0.1 \\ 0.4 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}, \quad b = \begin{bmatrix} 5 \\ 3 \\ 4 \end{bmatrix}, \]

where the decision variable is \( x = [x_1, x_2]^T \). This paper only deals with small scale application of CG method to test the capability of CG method for solving systems of linear equation.

<table>
<thead>
<tr>
<th>Initial points</th>
<th>MRMIL</th>
<th>MRMIL*</th>
<th>MFR</th>
<th>MDY</th>
<th>MCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10,10)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>(50,50)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>(100,100)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>(200,200)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

**Table 2: Result of Application in No. of Iteration**

<table>
<thead>
<tr>
<th>Initial Points</th>
<th>MRMIL</th>
<th>MRMIL*</th>
<th>MFR</th>
<th>MDY</th>
<th>MCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10,10)</td>
<td>0.0042</td>
<td>0.0038</td>
<td>0.0033</td>
<td>0.0043</td>
<td>0.00334</td>
</tr>
<tr>
<td>(50,50)</td>
<td>0.0026</td>
<td>0.0027</td>
<td>0.0029</td>
<td>0.0028</td>
<td>0.0034</td>
</tr>
<tr>
<td>(100,100)</td>
<td>0.0026</td>
<td>0.0027</td>
<td>0.0027</td>
<td>0.0026</td>
<td>0.0033</td>
</tr>
<tr>
<td>(200,200)</td>
<td>0.0026</td>
<td>0.0027</td>
<td>0.0033</td>
<td>0.0026</td>
<td>0.0035</td>
</tr>
</tbody>
</table>

**Table 3: Result of Application in CPU Time**

<table>
<thead>
<tr>
<th>Method</th>
<th>CPU Time per Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRMIL</td>
<td>0.001525</td>
</tr>
<tr>
<td>MRMIL*</td>
<td>0.001488</td>
</tr>
<tr>
<td>MFR</td>
<td>0.001525</td>
</tr>
<tr>
<td>MDY</td>
<td>0.001538</td>
</tr>
<tr>
<td>MCD</td>
<td>0.001700</td>
</tr>
</tbody>
</table>

From Table 2 and Table 3, all of the tested methods can solve the presented problem. However, MRMIL* method shows the best overall result compared to the other methods as it gives the least number of CPU time per iteration. Thus, based on the results of the numerical test, we may conclude that the MRMIL* method is applicable for real-life application.

6. Conclusion

In this paper, we propose a new version of Rmil* CG method under strong Wolfe line search. The classical Rmil* coefficient is combined with spectral CG method, namely the MRMIL* method. The results show that the modified MRMIL* method performs the best overall compared to the other modified CG methods since MRMIL* converges faster and is able to successfully solve most of the test problems. Based on the promising results, the MRMIL* method should also be applicable in real-life problem.

**Acknowledgement**

These authors would like to thank the Ministry of Higher Education of Malaysia (MOHE) for funding this research under the MyPhD KPM Sponsorship.

**References**

[23] Zhang L, Zhou W and Li D (2006), Global Convergence of a Modified Fletcher-Reeves Conjugate Gradient Method with Armijo-


