The spectral form of the Dai-Yuan conjugate gradient algorithm

Dr. Abdul-Ghafoor J. Salem.*    Dr. Khalil K. Abbo**

Abstract

Conjugate Gradient (CG) methods comprise a class of unconstrained optimization algorithms which are characterized by low memory requirements and strong local and global convergence properties. Most of CG methods do not always generate a descent search directions, so the descent or sufficient descent condition is usually assumed in the analysis and implementations. By assuming a descent and pure conjugacy conditions a new version of spectral Dai-Yuan (DY) non-linear conjugate gradient method introduced in this article. Descent property for the suggested method is proved and numerical tests and comparisons with other methods for large-scale unconstrained problems are given.

1-Introduction

The non-linear Conjugate Gradient (CG) method is a very useful technique for solving large scale unconstrained minimization problems and has wide applications in many fields [9]. This method is an iterative process which requires at each iteration the current gradient and previous

*Prof. \ College of Computers Sciences and Math.\ University of Mosul
**Assistant Prof. \ College of Computers Sciences and Math.\ University of Mosul.

Received:1/10 /2011 ____________________Accepted: 21 /12 / 2011
direction, which is characterized by low memory requirements and strong local and global convergence properties [3 and 12].

In this paper, we focus on conjugate gradient methods applied to the non-linear unconstrained minimization problem:

\[ \min f(x), \ x \in \mathbb{R}^n. \] ..........(1)

Where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is continuously differentiable function and bounded below. A conjugate gradient method generates a sequence \( x_k, k \geq 1 \) starting from an initial guess \( x_1 \in \mathbb{R}^n \), using the recurrence

\[ x_{k+1} = x_k + \alpha_k d_k \] ..........(2)

Where the positive step size \( \alpha_k \) is obtained by a line search, and the directions \( d_k \) are generated by the rule:

\[ d_{k+1} = -g_{k+1} + \beta_k d_k, \quad d_k = -g_k \] ..........(3)

Where \( g_k = \nabla f(x_k) \), and let \( y_k = g_{k+1} - g_k \) and \( s_k = x_{k+1} - x_k \), here \( \beta_k \) is the CG update parameter. Different CG methods corresponding to different choice for the parameter \( \beta_k \) see [1, 4 and 10]. The first CG algorithm for non-convex problems was proposed by Fletcher and Revees (FR) in 1964 [11], which is defined as

\[ \beta_k^{FR} = \frac{g_k^T g_k}{g_k^T y_k}. \] ..........(4)

We know that the other equivalents forms for \( \beta_k \) are Polack-Ribeir (PR) and Hestenes- Stiefel (HS) for example

\[ \beta_k^{PR} = \frac{g_k^T y_k}{g_k^T g_k}, \quad \text{and} \quad \beta_k^{HS} = \frac{g_k^T y_k}{d_k^T y_k}. \] ..........(5)

Although all the above formulas are equivalent for convex quadratic functions, but they have different performance for non-quadratic functions, the performance of a non-linear CG algorithm strongly depends on coefficient \( \beta_k \). Dai and Yuan (DY) in [6] proposed a non-linear CG method (2) and (3) with \( \beta_k \) defined as:

\[ \beta_k^{DY} = \frac{g_k^T g_k}{d_k^T y_k}. \] ..........(6)

Which generates a descent search directions

\[ d_k^T g_k < 0. \] ..........(7)

At every iteration \( k \) and convergence globally to the solution if the following Wolfe conditions are used to accept the step-size \( \alpha_k \) [2]:

\[ f(x_k + \alpha_k d_k) \leq f(x_k) + c_1 \alpha_k g_k^T d_k \] ..........(8)

\[ g(x_k + \alpha_k d_k)^T d_k \geq c_2 g_k^T d_k \] ..........(9)
Where $0 < c_1 < c_2 < 1$. Condition (8) stipulates a decrease of $f$ along $d_k$ if (7) satisfied. Condition (9) is called the curvature condition and it's role is to force $\alpha_k$ to be sufficiently far a way from zero [12]. Which could happen if only condition (8) were to be used. Conditions (8) and (9) are called Standard Wolfe Conditions (SDWC). Notice that if equation (8) satisfied then always there exists $\alpha > 0$ such that for any $\alpha_k \in [0, \alpha]$ the conditions (8) and (9) will be satisfied according to the theorem (1) given later. If we wish to find a point $\alpha_k$, which is closer to a solution of the one dimensional problem

$$\min_{\alpha \in \mathbb{R}} f(x_k + \alpha d_k)$$

than a point satisfying (8) and (9) we can impose on $\alpha_k$ the Strong Wolfe Conditions (STWC):

$$f(x_k + \alpha_k d_k) \leq f(x_k) + c_1 \alpha_k g^T_k d_k$$

$$\|g(x_k + \alpha_k d_k)^T d_k\| \leq c_2 \|g^T_k d_k\|$$

Where $0 < c_1 < c_2 < 1$. In contrast to (SDWC) $g^T_k d_k$ cannot be arbitrarily large [12]. The (STWC) with the sufficient descent property

$$d_k^T g_k < -c \|g_k\|, \quad c \in (0,1)$$

Widely used in the convergence analysis for the CG methods.

**Theorem (1):** Assume that $f$ is continuously differentiable and that is bounded below along the line $x = x_k + \alpha d_k, \quad \alpha \in (0, \infty)$. Suppose also that $d_k$ is a direction of descent (7) is satisfied if $0 < c_1 < c_2 < 1$ then there exists nonempty intervals of step lengths satisfying the (SDWC) and (STWC) conditions. For proof see [12].

The Fletcher-Revees (FR) and Dai-Yuan (DY) methods have common numerator $g_{k+1}^T g_k$. One theoretical difference between these methods and other choices for the update parameter $\beta_k$ is that the global convergence theorems only require the Lipschitz assumption not the bounded ness assumption [9].

The global convergence for the methods with $g_{k+1}^T g_{k+1}$ in the numerator of $\beta_k$ established with exact and inexact line searches for general functions [2, 7, and 13]. Despite the strong convergence theory that has been developed for methods with $g_{k+1}^T g_{k+1}$ in the numerator of $\beta_k$, these methods are all susceptible to jamming, that is they begin to take small steps without making significant progress to the minimum [9]. On the other hand the convergence of the methods with $g_{k+1}^T y_k$ in the numerator (PR) and (HS) for general non-linear function are uncertain, in general
the performance of these methods is better than the performance of the methods with $g_{k+1}^T g_{k+1}$ in the numerator of $\beta_k$ see [9], but they have weaker convergence theorems.

This paper is organized as follows: in section 2 new spectral form for DY non-linear conjugate gradient algorithm is suggested. In section 3 we will show that our algorithm satisfies sufficient descent condition for every iteration. Section 4 presents numerical experiments and comparisons.

2. New spectral form for Dai and Yuan CG method

An attractive feature of the CG method is that the following (pure conjugacy condition)

$$y^T_k d_{k+1} = 0 \quad \text{.........(14)}$$

is always holds if the objective function $f(x)$ is convex quadratic and line search is exact [8]. In this section we use the relation (7) and (14) to derive new spectral DY conjugate gradient method. Consider the search direction of the form

$$d_{k+1} = -\gamma_{k+1} g_{k+1} + \frac{g_{k+1}^T g_{k+1}}{y^T_k s_k} s_{k+1}, \quad d_i = -g_i \quad \text{.........(15)}$$

where $\gamma_k$ is parameter. Assume that the search direction in (15) satisfies the relation (7) i.e

$$0 < -\gamma_{k+1} + \frac{s^T_{k+1} g_{k+1}}{y^T_k s_k} < 0$$

or

$$\gamma_{k+1} = \frac{s^T_{k+1} g_{k+1}}{y^T_k s_k} + c, \quad c > 0$$

Then

$$\gamma_{k+1} = \frac{s^T_{k+1} g_{k+1} + c y^T_k s_k}{y^T_k s_k} \quad \text{.........(17)}$$

To find the value of $c$, we use the pure conjugacy condition (14) i.e

$$y^T_k d_{k+1} = 0 \quad \text{.........(18)}$$

With simple algebra we get

$$c = \frac{g^T_{k+1} g_{k+1} y^T_k s_k - s^T_k g_{k+1} g_{k+1} y^T_k s_k}{y^T_k g_{k+1} y^T_k s_k} \quad \text{.........(19)}$$
Equations (17) and (19) give
\[ \gamma_{k+1} = \frac{g_{k+1}^T g_{k+1}}{y_k^T g_{k+1}} \] (20)

Therefore the new spectral DY search direction is
\[ d_{k+1} = -\gamma_{k+1} g_{k+1} + \frac{g_{k+1}^T g_{k+1}}{y_k^T s_k} s_k \] ...........(21)

With \( \gamma_{k+1} \) defined in the equation (17)

**Algorithm (spectral form of DY. SPDY say)**

Step (1): Choose an initial starting point \( x_1 \in R^n \) and \( \varepsilon > 0 \), consider
\[ d_1 = -g_1, \quad \alpha_1 = \frac{1}{\|g_1\|}, \quad \text{and} \quad k = 1 \]

Step(2): Test for convergence. If \( \|g_k\| < \varepsilon \) stop \( x_k \) is optimal
Else go to step(3)

Step(3): Compute \( \alpha_k \) satisfying the (SDWC) or (STWC) and update the Variable \( x_{k+1} = x_k + \alpha_k d_k \) and compute \( f_{k+1}, \ g_{k+1}, \ y_k \) and \( s_k \)

Step(4): Direction computation: compute \( \gamma_{k+1} \) from (20) and set
\[ d = -\gamma_{k+1} g_{k+1} + \frac{g_{k+1}^T g_{k+1}}{y_k^T s_k} s_k \]. If Powell restart is satisfied then
\[ d_{k+1} = -\gamma_{k+1} g_{k+1} \]

Else \( d_{k+1} = d \), compute initial guess for \( \alpha_{k+1} = \alpha_k \left( \frac{d_{k+1}}{d_{k+1}} \right) \) and

Set \( k = k + 1 \) go to step(2)

3. **Descent property of the SPDY algorithm**

An important feature for any minimization algorithm is the descent (7) or the sufficient descent (13) property. In this section we proof that our suggested new algorithm (SPDY) generates a sufficient descent directions for each iteration \( k \).

**Theorem (1):**
Suppose that the step-size \( \alpha_k \) satisfies the standard Wolfe conditions (SDWC), consider the search directions \( d_k \) generated from (21) where \( \gamma_{k+1} \) computed from (20) then the search directions \( d_{k+1} \) satisfies the sufficient descent condition (13) for all \( k \).

**Proof**
The proof is by induction. If \( k=1 \) then \( d_i^T g_i = -g_i^T g_i = -\|g_i\| < 0 \) then the sufficient descent holds with \( c=1 \), know let \( s_k^T g_k < -\epsilon \|g_k\| \) to proof for \( k+1 \), multiply (21) by \( g_{k+1}^T \) to get

\[
g_{k+1}^T d_{k+1} = -y_{k+1}^T g_{k+1}^T g_{k+1} + \frac{g_{k+1}^T g_{k+1}^T}{y_k^T g_k} g_{k+1}^T s_k
\]

(22)

Note that from second standard Wolfe condition (9) we have

(i) \( y_k^T s_k \geq (c_2 - 1) s_k^T g_k > 0 \)

\[
s_k^T g_{k+1} = s_k^T g_{k+1} - s_k^T g_k + s_k^T g_k \leq s_k^T g_{k+1} - s_k^T g_k = s_k^T y_k
\]

(23)

From (20), (2.9) and (23) we get

\[
g_{k+1}^T d_{k+1} \leq \frac{-g_{k+1}^T g_{k+1}}{y_k^T g_k} g_{k+1}^T g_{k+1} + g_{k+1}^T g_{k+1}
\]

\[
= -(g_{k+1}^T g_{k+1} - 1) g_{k+1}^T g_{k+1} = -(g_{k+1}^T g_{k+1} g_{k+1}) g_{k+1}^T g_{k+1}
\]

Applying the inequality \( u^T v \leq \frac{1}{2} (\|u\|^2 + \|v\|^2) \) then

\[
g_{k+1}^T d_{k+1} \leq - \left( \frac{\|g_{k+1}^2 + \|g_k^2\|^2}{y_k^2 + \|g_k^2\|^2} \right) g_{k+1}^T g_{k+1} = -\|g_{k+1}\|^2
\]

where \( c = \frac{\|g_{k+1}\|^2 + \|g_k\|^2}{y_k^2 + \|g_k^2\|^2} \)

4. Numerical results and comparisons

In this section we present the computation performance of a FORTRAN implementation of the SPDY, DY and FR algorithms on a set of unconstrained optimization test problems. We selected (15) large-scale unconstrained optimization test problems in extended or generalized form from [5]. For each function we have considered \( n=100, 1000 \) (where \( n \) is the number of variables). All algorithms implement the standard Wolfe line search conditions with \( c_1 = 0.0001 \) and \( c_2 = 0.9 \) and same stopping criterion \( \|g_k\| < 10^{-5} \), where \( \| \cdot \|_\infty \) is the maximum absolute component of a vector.

The comparison of algorithms are given in the following context. We say that, in the particular problem \( i \) the performance of Algorithm(Alg1)
was better than the performance of Alg2 if the number of iterations (iter) or the number of function-gradient evolutions (fg) or the number of restart (irs) of Alg1 was less than the number of (iter) or (fg) or the (irs) corresponding to Alg2, respectively. Table(1) and table(2) shows the details of numerical results for the Fletcher-Revees (FR), Dai-Yuan (DY) and our algorithm (SPDY).

### Table (1) Comparison of the algorithms for n=100

<table>
<thead>
<tr>
<th>Test Problems</th>
<th>FR</th>
<th>DY</th>
<th>SPDY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Iter</td>
<td>fg</td>
<td>irs</td>
</tr>
<tr>
<td>Trigonometric</td>
<td>18</td>
<td>34</td>
<td>10</td>
</tr>
<tr>
<td>Ex. Rosenbrock (CUTE)</td>
<td>41</td>
<td>84</td>
<td>22</td>
</tr>
<tr>
<td>Ex. White &amp; Holst</td>
<td>36</td>
<td>76</td>
<td>20</td>
</tr>
<tr>
<td>Perturbed Quadratic</td>
<td>101</td>
<td>154</td>
<td>31</td>
</tr>
<tr>
<td>Diagonal 2</td>
<td>67</td>
<td>107</td>
<td>23</td>
</tr>
<tr>
<td>Hager</td>
<td>28</td>
<td>46</td>
<td>11</td>
</tr>
<tr>
<td>Generalized Tridiagonal 2</td>
<td>36</td>
<td>57</td>
<td>11</td>
</tr>
<tr>
<td>Extended Powell</td>
<td>59</td>
<td>113</td>
<td>20</td>
</tr>
<tr>
<td>Extended BD1</td>
<td>42</td>
<td>70</td>
<td>39</td>
</tr>
<tr>
<td>Extended Maratos</td>
<td>70</td>
<td>160</td>
<td>36</td>
</tr>
<tr>
<td>Ex. Quad. Penalty QP2</td>
<td>28</td>
<td>60</td>
<td>15</td>
</tr>
<tr>
<td>Partial Perturbed Quad.</td>
<td>74</td>
<td>114</td>
<td>26</td>
</tr>
<tr>
<td>Almost Perturbed Quad.</td>
<td>84</td>
<td>133</td>
<td>21</td>
</tr>
<tr>
<td>Tridiago. Perturbed Quad.</td>
<td>105</td>
<td>168</td>
<td>35</td>
</tr>
<tr>
<td>ENGVAL1 (CUTE)</td>
<td>27</td>
<td>47</td>
<td>9</td>
</tr>
<tr>
<td>Total</td>
<td>816</td>
<td>1423</td>
<td>329</td>
</tr>
</tbody>
</table>

### Table (2) Comparison of the algorithms for n=1000

<table>
<thead>
<tr>
<th>Test Problems</th>
<th>FR</th>
<th>DY</th>
<th>SPDY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Iter</td>
<td>fg</td>
<td>irs</td>
</tr>
<tr>
<td>Trigonometric</td>
<td>29</td>
<td>53</td>
<td>19</td>
</tr>
<tr>
<td>Ex. Rosenbrock (CUTE)</td>
<td>40</td>
<td>92</td>
<td>20</td>
</tr>
<tr>
<td>Ex. White &amp; Holst</td>
<td>36</td>
<td>76</td>
<td>18</td>
</tr>
<tr>
<td>Perturbed Quadratic</td>
<td>284</td>
<td>437</td>
<td>83</td>
</tr>
<tr>
<td>Diagonal 2</td>
<td>219</td>
<td>360</td>
<td>71</td>
</tr>
<tr>
<td>Hager</td>
<td>278</td>
<td>496</td>
<td>248</td>
</tr>
<tr>
<td>Generalized Tridiagonal 2</td>
<td>64</td>
<td>98</td>
<td>25</td>
</tr>
<tr>
<td>Extended Powell</td>
<td>67</td>
<td>128</td>
<td>22</td>
</tr>
<tr>
<td>Extended BD1</td>
<td>53</td>
<td>87</td>
<td>53</td>
</tr>
<tr>
<td>Extended Maratos</td>
<td>70</td>
<td>155</td>
<td>36</td>
</tr>
<tr>
<td>Ex. Quad. Penalty QP2</td>
<td>36</td>
<td>87</td>
<td>20</td>
</tr>
<tr>
<td>Partial Perturbed Quad.</td>
<td>225</td>
<td>373</td>
<td>56</td>
</tr>
<tr>
<td>Almost Perturbed Quad.</td>
<td>323</td>
<td>503</td>
<td>88</td>
</tr>
<tr>
<td>Tridiago. Perturbed Quad.</td>
<td>406</td>
<td>628</td>
<td>114</td>
</tr>
<tr>
<td>ENGVAL1 (CUTE)</td>
<td>104</td>
<td>180</td>
<td>90</td>
</tr>
<tr>
<td>Total</td>
<td>2234</td>
<td>3753</td>
<td>963</td>
</tr>
</tbody>
</table>
References


