Generalized CG-algorithms Based on Rational Sigmoid Model for Unconstrained Optimization

Abbas Y. AL-Bayati*        Abdulghafour J. S. **  Khalil K. Abbo***

Abstract
In this paper, we present extension forms of Dai, Yuan (DY) and Fletcher, Revere (FR) CG algorithms. Our modifications and based on introducing a non-quadratic model (sigmoid function model). These modified algorithms are implemented with Wolfe conditions, where initial step size $\alpha_k$ in each iteration is taken as $\alpha_k = \rho_{k-1} \cdot \sqrt{\|d_{k-1}\|/\|d_k\|}$ and the global convergence of the modified DY algorithm is investigated. These modified algorithms are tested on some standard test functions and compared with the original DY and FR algorithms showing considerable improvements over all these comparisons.
**Introduction:**

Conjugate gradient (CG) methods represent an important class of unconstrained optimization algorithms with strong local and global convergence properties and low memory requirements. A survey of development of different versions of non linear CG method with special attentions to global convergence properties is presented by Hager and Zhang [12]. This family of algorithms includes a lot of variants (see[2]) well known in the literature, with important convergence properties and numerical efficiency.

For solving the non-linear unconstrained optimization problem:

\[\text{Min } f(x), \quad x \in \mathbb{R}^n \quad \cdots (1)\]

where \( f: \mathbb{R}^n \to \mathbb{R} \) is continuously differentiable function bounded from below. Starting from an initial guess: \( x_0 \in \mathbb{R}^n \), a non linear CG method generates a sequence \( \{x_k\} \) as:

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

and

\[d_{k+1} = \begin{cases} -g_k & k = 0 \\ -g_{k+1} + \beta_k d_k & k \geq 1 \end{cases} \quad \cdots (3)\]

where \( \alpha_k \) is obtained by line search .In eq. (3) \( \beta_k \) is known as the conjugate gradient parameter, defined \( g = \nabla f(x_k) \). Let \( S_k = x_{k+1} - x_k \), \( y_k = g_{k+1} - g_k \), the line search in the CG algorithms often is based on the standard Wolfe conditions [4].

\[f(x_k + \alpha_k d_k) \leq f(x_k) + c_1 \alpha_k g_k^T d_k \quad \cdots (4a)\]

\[g_{k+1}^T d_k \geq c_2 g_k^T d_k \quad \cdots (4b)\]

where \( d_k \) is descent direction, i.e.:

\[g_k^T d_k < 0 \quad \cdots (5)\]

and \( 0 < c_1 \leq c_2 < 1 \), for some CG algorithms stronger version of the Wolf conditions (4a) and:

\[g_{k+1}^T d_k \leq -c_2 g_k^T d_k \quad \cdots (6)\]

are needed to ensure convergence and to enhance stability [2].

Different CG algorithms corresponding to different choice for the parameter \( \beta_k \) therefore a crucial element in any CG algorithm is the formula definition of \( \beta_k \) because \( \alpha_k \) is not exact in practice and objective function \( f \) is not quadratic many formulas have been proposed to compute \( \beta_k \) four well-known formulas \( \beta_k \) are:
The Second Scientific Conference of Mathematics-Statistics & Information

\[ \beta_k^{HS} = \frac{g_{k+1}^T y_k}{y_k^T d_k} \]

\[ \beta_k^{PR} = \frac{g_{k+1}^T y_k}{g_k^T g_k} \]

\[ \beta_k^{DY} = \frac{g_{k+1}^T g_k}{y_k^T d_k} \]

where \( HS \) stands Hestenes and Stiefel [13], \( FR \) stands Fletcher and Reeves [11], \( PR \) stands Polak-Riebier [14] and \( DY \) stands Dai and Yuan [9]. All these methods are equivalent if the step size is exact and objective function is quadratic.

It is shown that CG methods with \( g_{k+1}^T g_k \) in the numerator of \( \beta_k \) have strong global convergence theorems with exact and inexact line searches (especially with Wolfe conditions) but has poor performance in practice although DY-CG method is better than FR-CG method in application. On the other hand, the CG methods with \( g_{k+1}^T y_k \) in the numerator of \( \beta_k \) has uncertain global convergence for general non-linear functions, but has good performance in practice (see [12] for the details). Therefore CG methods have been frequently modified and improved by many authors Beale [6] and Powell [15] have described CG methods with improved restart. Andrei [5] introduced scaled CG methods. Fried [10], Al-Bayati [1], Tassopoulos el al [16] have proposed further modifications of the conjugate gradient methods which are based on some non-quadratic models.

In this paper we generalize the FR-CG and DY-CG methods by considering more general function than quadratic which we call it as quasi- sigmoid function, our goal is to preserve the convergence properties of these methods (FR, DY) and to force their performance in practice.

2- Generalized CG methods based on non-quadratic model.

2-1 Introduction to non-quadratic models

most of the currently used optimization methods use a local quadratic representation of the objective function. But the use of quadratic model may be inadequate to incorporate all the information [10] so that more general models than quadratic are proposed as a basis for CG algorithms. If \( q(x) \) is a quadratic function defined by:

\[ q(x) = \frac{1}{2} x^T G x + b^T x + c \]...

where \( G \) is \( n \times n \) symmetric and positive definite matrix and \( b \) is Constant vector in \( R^n \) and \( c \) constant. Then we say that \( f \) is defined as a nonlinear scaling of \( q(x) \) if the following conditions hold [7]:

\[ f(x) = F(q(x)), \quad q > 0 \]...

(8)
and
\[ \frac{dF}{dq} > 0 \quad \text{... (9)} \]

In this area there are various published works.

a) A CG methods which minimize the function:
\[ f(x) = (q(x))^p \quad p > 0 \quad x \in \mathbb{R}^n \quad \text{... (10)} \]
most n-steps have been described by [10].

b) The special polynomial case
\[ F(q(x)) = e_1 q(x) + \frac{1}{2} e_2 q^2(x) \quad \text{... (11)} \]
where \( e_1 \), \( e_2 \) scalars have been investigated by [7].

c) A rational model has been developed by [16] where
\[ F(q(x)) = \frac{e_1 q(x) + 1}{e_2 q(x)} \quad e_1 q(x) < 0 \quad \text{... (12)} \]
d) Another rational model was considered by [1] where:
\[ F(q(x)) = \frac{e_1 q(x)}{1 - e_2 q(x)} \quad \text{... (13)} \]
\( e_1 > 0 \quad \text{,} \quad e_2 < 0 \)

2-2 The New Extended CG method

In this paper we consider the new model function defined by:
\[ h(x) = \log\left( f(x) \right) \quad \text{... (14.a)} \]
where
\[ f(x) F(q(x)) = \frac{q(x)}{1 + e^{-q(x)}} \quad \text{... (14.b)} \]

This is the logarithmic quasi-sigmoid function where \( q > 0 \) with further assumption that:
\[ \frac{dF}{dq} > 0 \quad \text{... (15)} \]

From equation (14.b) we have:
\[ \frac{dF}{dq} = F' = \left( 1 + e^{-q} \right) + q e^{-q} \quad f \ast (1 + \frac{1}{q - f}) \quad \text{... (16.a)} \]
\[ \therefore F' = f \ast (1 + q - f) \quad \text{... (16.b)} \]

Return to equation (14) and solve it for \( q \) assuming that \( e^{-q} = 1 - q + \frac{1}{2} q^2 + r \)
where \( r = \sum_{n=3}^{\infty} \frac{(-1)^n q^n}{n!} \quad q^n (1 - \frac{1}{n+1} q) \) and set \( \sigma = r + 1 \) since \( r \) is too small.

Then
\[ f = \frac{q}{\frac{1}{2} q^2 - q + \sigma} \quad \text{... (17.a)} \]
where 
\[ q = \eta + \sqrt{\eta^2 - 2 \sigma} \] ... (17.b)
where 
\[ \eta = 1 + \frac{1}{f} \] ...(17.c)

From (16) and (17) to compute \( F' \), assuming \( \sigma = \frac{1}{2} \)

\[ F' = f^* \left( \frac{2 - f + \frac{1}{f} + \sqrt{\eta^2 - 1}}{\eta + \sqrt{\eta^2 - 1}} \right) \] ... (18)

Now from (14)
\[ h'(x) = \frac{F'}{f} \] ...(19.a)

And hence
\[ \rho_k = \frac{h'_k}{h_{k+1}} \] ...(19.b)

Then we can compute the value of \( \rho_k \) using function values at two points \( x_{k+1} \) and \( x_k \) as follows:

\[ \rho_k = \frac{(2 - f_k + \frac{1}{f_k} + \sqrt{\eta_k^2 - 1})}{(\eta_k + \sqrt{\eta_k^2 - 1})} \times \frac{(\eta_{k+1} + \sqrt{\eta_{k+1}^2 - 1})}{(2 - f_{k+1} + \frac{1}{f_{k+1}} + \sqrt{\eta_{k+1}^2 - 1})} \] ... (20)

The extended DY-CG method can be done by modifying search directions given in (3) as follows:

\[ \tilde{d}_1 = -g_1 \] ... (21)

And for \( k \geq 0 \)

\[ \tilde{d}_{k+1} = -g_{k+1} + \rho_k \tilde{\beta}_k \tilde{d}_k \] ... (22)

\[ \tilde{\beta}_k = \frac{g_k^{T} \tilde{g}_{k+1}}{\tilde{d}_k^{T} \tilde{g}_{k+1}} = \frac{g_{k+1}^{T} \tilde{g}_{k+1}}{\tilde{d}_k^{T} (\rho_k \tilde{g}_{k+1} - g_k)} \] ... (23)

Where: \( \tilde{g} = \nabla F(q(x)) = \frac{\partial f}{\partial x} = \frac{dF}{dq} \frac{\partial q}{\partial x} = F' g \)

\( \tilde{d} \) is the search direction applied to \( F(q(x)) \) and \( \tilde{y} = \tilde{g}_{k+1} - \tilde{g}_k \)

The original DY-CG method and EDY-CG algorithm defined in (21-23) generates the same set of directions and same sequence of points \( \{x_k\} \) by using the following theorem:
Theorem (1):
Given an identical starting point \( x_0 \in \mathbb{R}^n \).
The method of DY-CG defined by (2) and (3) with \( \beta^{dy} \) and applied to \( f(x) = q(x) \) and extended EDY-CG method defined by (21-23) and applied to \( f(x) = F(q(x)) \) with \( \rho_k \) defined by (20) generate identical set of directions and identical sequence of points \( \{x_k\} \).

Proof:
The prove is by induction for \( k = 0 \)

We have:
\[
\tilde{d}_1 = -g_1 = -\frac{\partial q}{\partial x} \hat{c}F = -F'g_1 = F'd_1
\]

Suppose:
\[
\tilde{d}_k = F'_k d_k \text{ , to prove for } k + 1
\]

\[
\begin{align*}
\tilde{d}_{k+1} &= -g_{k+1} + \rho_k \tilde{d}_k \\
&= -F'_k g_{k+1} + \frac{F'_k}{F'_{k+1}} F'_k d_k \left( \frac{F^2_{k+1} g_{k+1} g_{k+1}}{F'_{k+1}} - F'_k g_{k+1} \right) \\
&= F'_k d_k \left[ \frac{F^2_{k+1} g_{k+1} g_{k+1}}{F'_{k+1}} - F'_k g_{k+1} \right] \\
&= F'_k d_k \\
\end{align*}
\]

Hence the two methods generate the same set of directions.

The FR-CG method can be extended in a similar way, i.e.: the search direction in EFR-CG method is given by:

\[
\tilde{d}_{k+1} = -g_{k+1} + \rho_k \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k} \tilde{d}_k 
\]

Also we can show that:

\[
\tilde{d}_{k+1} = F'_k d_{k+1}
\]

3-Convergence Analysis:
In this section we are going to discuss the convergence theorem of EDY-CG which is similar to the theorem given by Dai and Yuan in [9]. We assume that the objective function satisfies the following conditions:

1. \( f \) is bounded below and belong to \( C^2 \).
2. Level set \( L = \{x : f(x) \leq f(x_i)\} \) is bounded.
3. \( g \) is Lipschitz continuous in \( N \), where \( N \) is neighborhood of \( L \) and \( \exists \ L > 0 \) S.t:
   \[
   \|g(x) - g(y)\| \leq L\|x - y\| \tag{25}
   \]

Most of CG methods use the Zoutendijk theorem to establish global convergence hence we state this theorem.
**Zoutendijk theorem:**

Suppose $x$ is a starting point for which assumptions (1, 2, 3) are satisfied. Consider any algorithm of the form $x_{k+1} = x_k + \alpha_k d_k$ where $d_k$ is descent direction and $\alpha_k$ satisfies the standard Wolfe conditions (4a, 4b) then:

$$\sum_{k=1}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty$$

Proof: see [9]

**Theorem (2):** The search directions generated by EDY-CG algorithm are descent directions.

Proof:

The proof is by indication for $k = 0$ we have:

$$\tilde{d}_1 = -g_1$$

$$\tilde{d}_1 g_1 = -g_1 g_1$$

$$= -F_k^{\gamma} \|g_k\| < 0$$

Suppose: $d_k^T g_k < 0$ i.e. $F_k^{\gamma} d_k^T g_k < 0$

From theorem (1) we have:

$$\tilde{d}_{k+1} = F_k^{\gamma} d_{k+1} = F_k^{\gamma} \left[ -g_{k+1} + \frac{g_{k+1}^T g_{k+1} d_k}{y_k^T d_k} \right]$$

$$= F_k^{\gamma} \left[ -g_{k+1} y_k^T d_k + \frac{g_{k+1}^T g_{k+1} d_k}{y_k^T d_k} \right]$$

$$= F_k^{\gamma} \|g_{k+1}\| \left[ -g_{k+1} d_k + \frac{g_{k+1}^T g_{k+1} d_k}{y_k^T d_k} \right]$$

$$= F_k^{\gamma} \|g_{k+1}\| \frac{g_{k+1}}{y_k^T d_k}$$

By (4b)

$$y_k^T d_k = g_{k+1}^T d_k - g_k^T d_k \geq C_2 g_{k+1}^T d_k - g_k^T d_k$$

$$= (C_2 - 1)g_k^T d_k$$

$$\tilde{g}_{k+1} \tilde{d}_{k+1} = F_k^{\gamma} \|g_{k+1}\| \frac{g_{k+1}}{y_k^T d_k} \leq F_k^{\gamma} \frac{\|g_{k+1}\|^2}{C_2 - 1} < 0$$

$0 < C_2 < 1$

Hence the search direction are descent and independent to $\rho_k$. 
Theorem (3): Consider any iteration of the form \( x_{k+1} = x_k + \alpha_k d_k \) where \( d_k \) defined in (22) and \( \alpha_k \) satisfies the standard Wolfe conditions (4a, 4b) further assume that assumption (1, 2, 3) are valid then the algorithm either stops at stationary point.
i.e.: \( \|g_k\| = 0 \) or \( \lim \inf \|g_k\| = 0 \).

Proof:

\[
\tilde{d}_{k+1} = -g_{k+1} + \rho_k \frac{g_{k+1}^{T} g_{k+1}}{y_k^T d_k} \tilde{d}_k
\]

\[
F'_{k+1} (d_{k+1} + g_{k+1}) = \frac{F'_{k+1}(g_{k+1})}{y_k^T d_k}
\]

Or

\[
d_{k+1} + g_{k+1} = \beta_{k+1} d_k \quad \text{...(26)}
\]

(the rest of the proof is similar to one given in [9]). Taking the square of each side and noting that

\[
\beta_{k+1} = \frac{d_{k+1}^T g_{k+1}}{d_k^T g_k} \quad \text{...(27)}
\]

we get:

\[
\|d_{k+1}\|^2 = (\beta_{k+1})^2 \|d_k\|^2 - 2d_{k+1}^T g_{k+1} - \|g_{k+1}\|^2
\]

Divide each term in above equation by: \((d_{k+1}^T g_{k+1})^2\)

\[
\therefore \frac{\|d_{k+1}\|^2}{(d_{k+1}^T g_{k+1})^2} = \frac{\|d_k\|^2}{d_k^T g_k} - 2 \frac{d_{k+1}^T g_{k+1}}{(d_{k+1}^T g_{k+1})^2} - \frac{\|g_{k+1}\|^2}{(d_{k+1}^T g_{k+1})^2}
\]

Completing the squares for last two terms:

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

\[
\frac{d_{k+1}^T g_{k+1}}{(d_{k+1}^T g_{k+1})^2} \leq \frac{d_k^T g_k}{(d_k^T g_k)^2} + \frac{1}{\|g_{k+1}\|^2}
\]

Noting that \( d_1 = -g_1 \rightarrow \frac{\|d_1\|^2}{\|g_1\|^2} = \frac{1}{\|g_1\|^2} \)

i.e.: if \( k \) starts from zero we get:

\[
\therefore \frac{\|d_k\|^2}{(d_k^T g_k)^2} \leq \sum_{i=0}^{k} \frac{1}{\|g_i\|^2} \quad \forall \ k \geq 0
\]

now if \( \|g_k\| \neq 0 \) then \( \exists \) positive scalar \( \gamma > 0 \)

s.t. \( \|g_k\| \geq \gamma \) \( \forall \ k \geq 0 \rightarrow \frac{\|d_k\|^2}{(d_k^T g_k)^2} \leq \frac{1}{\gamma} \)
\[
\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} \geq \sum_{k=0}^{\infty} \gamma > \infty
\]

which is contradiction with Zoutendijk theorem hence \( \|g_0\| = 0 \) therefore EDY-CG algorithm is globally convergent. The global convergence of EFR-CG method can be established by further assumptions such as sufficient descent and \( C_2 < \frac{1}{2} \) in standard Wolfe conditions, but we can not be sure that EFRCG method generates sufficient descent directions! And hence convergence analysis of EFR-CG method is omitted.

4-Numerical results and comparisons:

In this section we present computation performance of a Fortran implementation of the of EDY-CG and the EFR-CG algorithms on a set of unconstrained optimization test problem, these problems are taken from [8] and [3]. We selected Large scale unconstrained optimization problems in extended or generalized form see appendix for each function we have considered numerical experiments with the number of variables \( n=100, 500, 1000, 10000 \).

All algorithms implement with the standard Wolfe line search conditions with

\[
C_1 = 0.0001 \quad , \quad C_2 = 0.9 \quad \text{and} \quad \alpha_i = 1/\|g_i\| \quad \text{and} \quad \alpha_k = \alpha_{k-1} \cdot \sqrt{\|d_{k-1}\|/\|d_k\|}
\]

In all cases the stopping criteria is \( \|g_k\| \leq 10^{-6} \),

The comparison is based on number of iteration (nit), number of function gradient evolutions (fge) , and ability of the algorithms to solve particular problems.

All codes are written in double precision FORTRAN (2000) and compiled with f77 default compiler settings, these codes are originally authored by Necula Andrei and modified by the authors.

In table (1) and (2) we compare the EDY-CG and EFR-CG with DY-CG and FR-CG for \( n=100, 500 \) and \( 1000,10000 \), respectively, Where * in table (1) and (2) means that the algorithm is unable to solve the problem in less than the maximum number of iterations which is considered to be 2000 in our tests. It is shown in Table (2) that some algorithms fail to solve problems (4,7,12), for \( n=1000 \) or \( n=10000 \). To find the total number of iterations or total number of function gradient evolutions, we replaced the * in each column by the sum of that column divided by 15(the number of test problems ). In Tables (3) and (4) we presented the
performance of the all algorithms in terms of percentage where FR-CG method is considered as 100%, from Table (3) we see that all algorithms improve FR-CG method about (4%-37%) in the number of iteration (nit), and about (6%-29%) in the number of function gradient evolutions for n=100 and for n=500, also from Table (4) we observe that the improvements over FR-CG method are about (5%-43%) in (nit), and (7%-31%).

Table (1) Comparison of algorithms for n=100, 500

<table>
<thead>
<tr>
<th>PN_0</th>
<th>n</th>
<th>DY-CG</th>
<th>FR-CG</th>
<th>EDY-CG</th>
<th>EFR-CG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>nit</td>
<td>fge</td>
<td>nit</td>
<td>fge</td>
</tr>
<tr>
<td>1</td>
<td>10^2</td>
<td>18</td>
<td>34</td>
<td>19</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>75</td>
<td>122</td>
<td>76</td>
<td>126</td>
</tr>
<tr>
<td>2</td>
<td>10^2</td>
<td>10</td>
<td>81</td>
<td>47</td>
<td>93</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>40</td>
<td>89</td>
<td>44</td>
<td>89</td>
</tr>
<tr>
<td>3</td>
<td>10^2</td>
<td>83</td>
<td>125</td>
<td>95</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>226</td>
<td>366</td>
<td>228</td>
<td>371</td>
</tr>
<tr>
<td>4</td>
<td>10^2</td>
<td>73</td>
<td>113</td>
<td>102</td>
<td>161</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>243</td>
<td>395</td>
<td>380</td>
<td>675</td>
</tr>
<tr>
<td>5</td>
<td>10^2</td>
<td>10</td>
<td>21</td>
<td>32</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>42</td>
<td>76</td>
<td>44</td>
<td>77</td>
</tr>
<tr>
<td>6</td>
<td>10^2</td>
<td>40</td>
<td>61</td>
<td>37</td>
<td>67</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>80</td>
<td>147</td>
<td>248</td>
<td>372</td>
</tr>
<tr>
<td>7</td>
<td>10^2</td>
<td>79</td>
<td>151</td>
<td>180</td>
<td>313</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>133</td>
<td>232</td>
<td>274</td>
<td>472</td>
</tr>
<tr>
<td>8</td>
<td>10^2</td>
<td>46</td>
<td>84</td>
<td>124</td>
<td>231</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>133</td>
<td>232</td>
<td>274</td>
<td>472</td>
</tr>
<tr>
<td>9</td>
<td>10^2</td>
<td>31</td>
<td>59</td>
<td>71</td>
<td>110</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>24</td>
<td>48</td>
<td>70</td>
<td>126</td>
</tr>
<tr>
<td>10</td>
<td>10^2</td>
<td>39</td>
<td>59</td>
<td>40</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>33</td>
<td>52</td>
<td>39</td>
<td>63</td>
</tr>
<tr>
<td>11</td>
<td>10^2</td>
<td>15</td>
<td>30</td>
<td>13</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>12</td>
<td>24</td>
<td>11</td>
<td>22</td>
</tr>
<tr>
<td>12</td>
<td>10^2</td>
<td>85</td>
<td>133</td>
<td>121</td>
<td>218</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>179</td>
<td>275</td>
<td>193</td>
<td>355</td>
</tr>
<tr>
<td>13</td>
<td>10^2</td>
<td>106</td>
<td>166</td>
<td>106</td>
<td>166</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>254</td>
<td>401</td>
<td>220</td>
<td>359</td>
</tr>
<tr>
<td>14</td>
<td>10^2</td>
<td>55</td>
<td>874</td>
<td>34</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>88</td>
<td>929</td>
<td>30</td>
<td>55</td>
</tr>
<tr>
<td>15</td>
<td>10^2</td>
<td>78</td>
<td>177</td>
<td>89</td>
<td>174</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>67</td>
<td>146</td>
<td>380</td>
<td>675</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2305</td>
<td>5492</td>
<td>3373</td>
<td>5818</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td>2134</td>
</tr>
</tbody>
</table>
Table (2) Comparison of algorithm for n=1000. 1000

<table>
<thead>
<tr>
<th>PN0</th>
<th>N</th>
<th>DY-CG nit</th>
<th>FRCG nit</th>
<th>EDY-CG nit</th>
<th>EFRCG nit</th>
<th>DY-CG fge</th>
<th>FRCG fge</th>
<th>EDY-CG fge</th>
<th>EFRCG fge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10^3</td>
<td>38</td>
<td>65</td>
<td>38</td>
<td>65</td>
<td>38</td>
<td>73</td>
<td>38</td>
<td>73</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>35</td>
<td>61</td>
<td>32</td>
<td>60</td>
<td>35</td>
<td>61</td>
<td>33</td>
<td>60</td>
</tr>
<tr>
<td>2</td>
<td>10^3</td>
<td>39</td>
<td>85</td>
<td>78</td>
<td>131</td>
<td>39</td>
<td>84</td>
<td>78</td>
<td>131</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>38</td>
<td>86</td>
<td>54</td>
<td>106</td>
<td>38</td>
<td>86</td>
<td>54</td>
<td>106</td>
</tr>
<tr>
<td>3</td>
<td>10^3</td>
<td>393</td>
<td>629</td>
<td>349</td>
<td>568</td>
<td>415</td>
<td>649</td>
<td>377</td>
<td>616</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>1051</td>
<td>1654</td>
<td>1417</td>
<td>2160</td>
<td>1241</td>
<td>1949</td>
<td>1289</td>
<td>2077</td>
</tr>
<tr>
<td>4</td>
<td>10^3</td>
<td>326</td>
<td>545</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>329</td>
<td>554</td>
<td>517</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>5</td>
<td>10^3</td>
<td>15</td>
<td>29</td>
<td>77</td>
<td>129</td>
<td>15</td>
<td>29</td>
<td>77</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>13</td>
<td>27</td>
<td>1392</td>
<td>1443</td>
<td>16</td>
<td>33</td>
<td>1384</td>
<td>1430</td>
</tr>
<tr>
<td>6</td>
<td>10^3</td>
<td>64</td>
<td>101</td>
<td>73</td>
<td>115</td>
<td>65</td>
<td>102</td>
<td>73</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>65</td>
<td>102</td>
<td>180</td>
<td>300</td>
<td>85</td>
<td>153</td>
<td>1293</td>
<td>1366</td>
</tr>
<tr>
<td>7</td>
<td>10^3</td>
<td>85</td>
<td>156</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>61</td>
<td>114</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>89</td>
<td>170</td>
<td>*</td>
<td>*</td>
<td>100</td>
<td>191</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>8</td>
<td>10^3</td>
<td>160</td>
<td>280</td>
<td>445</td>
<td>711</td>
<td>182</td>
<td>317</td>
<td>445</td>
<td>710</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>609</td>
<td>1072</td>
<td>1279</td>
<td>2203</td>
<td>550</td>
<td>965</td>
<td>1279</td>
<td>2204</td>
</tr>
<tr>
<td>9</td>
<td>10^3</td>
<td>26</td>
<td>51</td>
<td>47</td>
<td>84</td>
<td>53</td>
<td>97</td>
<td>56</td>
<td>108</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>28</td>
<td>54</td>
<td>47</td>
<td>84</td>
<td>30</td>
<td>59</td>
<td>104</td>
<td>162</td>
</tr>
<tr>
<td>10</td>
<td>10^3</td>
<td>69</td>
<td>1081</td>
<td>43</td>
<td>68</td>
<td>39</td>
<td>59</td>
<td>43</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>478</td>
<td>4662</td>
<td>160</td>
<td>3964</td>
<td>500</td>
<td>5363</td>
<td>168</td>
<td>3958</td>
</tr>
<tr>
<td>11</td>
<td>10^3</td>
<td>15</td>
<td>31</td>
<td>15</td>
<td>29</td>
<td>14</td>
<td>29</td>
<td>14</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>9</td>
<td>20</td>
<td>11</td>
<td>22</td>
<td>8</td>
<td>19</td>
<td>11</td>
<td>22</td>
</tr>
<tr>
<td>12</td>
<td>10^3</td>
<td>243</td>
<td>381</td>
<td>345</td>
<td>634</td>
<td>230</td>
<td>361</td>
<td>235</td>
<td>434</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>662</td>
<td>1027</td>
<td>*</td>
<td>*</td>
<td>560</td>
<td>871</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>13</td>
<td>10^3</td>
<td>362</td>
<td>564</td>
<td>335</td>
<td>541</td>
<td>321</td>
<td>496</td>
<td>348</td>
<td>562</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>1208</td>
<td>1891</td>
<td>1137</td>
<td>1846</td>
<td>1029</td>
<td>1765</td>
<td>1668</td>
<td>2151</td>
</tr>
<tr>
<td>14</td>
<td>10^3</td>
<td>141</td>
<td>3956</td>
<td>142</td>
<td>3616</td>
<td>157</td>
<td>4477</td>
<td>140</td>
<td>3614</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>176</td>
<td>4862</td>
<td>203</td>
<td>5655</td>
<td>129</td>
<td>3327</td>
<td>203</td>
<td>5655</td>
</tr>
<tr>
<td>15</td>
<td>10^3</td>
<td>100</td>
<td>235</td>
<td>107</td>
<td>211</td>
<td>102</td>
<td>252</td>
<td>186</td>
<td>389</td>
</tr>
<tr>
<td></td>
<td>10^4</td>
<td>161</td>
<td>452</td>
<td>1132</td>
<td>1496</td>
<td>148</td>
<td>332</td>
<td>646</td>
<td>827</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7144</td>
<td>25950</td>
<td>12184</td>
<td>34988</td>
<td>6964</td>
<td>24382</td>
</tr>
</tbody>
</table>

Table (3) Comparison of the algorithms, for n=100 and n=500

<table>
<thead>
<tr>
<th>Measure</th>
<th>FR-CG</th>
<th>DY-CG</th>
<th>EFR-CG</th>
<th>EDY-CG</th>
</tr>
</thead>
<tbody>
<tr>
<td>nit</td>
<td>100%</td>
<td>68%</td>
<td>96%</td>
<td>63%</td>
</tr>
<tr>
<td>fge</td>
<td>100%</td>
<td>94%</td>
<td>93%</td>
<td>71%</td>
</tr>
</tbody>
</table>

Table (4) Comparison of the algorithms, for n=1000 and n=10000

<table>
<thead>
<tr>
<th>Measure</th>
<th>FR-CG</th>
<th>DY-CG</th>
<th>EFR-CG</th>
<th>EDY-CG</th>
</tr>
</thead>
<tbody>
<tr>
<td>nit</td>
<td>100%</td>
<td>58%</td>
<td>95%</td>
<td>57%</td>
</tr>
<tr>
<td>fge</td>
<td>100%</td>
<td>74%</td>
<td>93%</td>
<td>69%</td>
</tr>
</tbody>
</table>
Appendix

1. Extended Trigonometric Function

\[ f(x) = \sum_{i=1}^{n} \left( \left( n - \sum_{j=1}^{n} \cos x_j \right) + (1 - \cos x_i) - \sin x_i \right)^2, \quad x_0 = [0.2, 0.2, 0.2, ..., 0.2] \]

2. Extended Rosenbrok Function

\[ f(x) = \sum_{i=1}^{n} c(x_{2i} - x_{2i-1})^2 + (1 - x_{2i-1})^2, \quad x_0 = [-1.2, 1, ..., -1.2], \quad c = 100 \]

3. Perturbed Quadratic Function

\[ f(x) = \sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} \left( \sum_{i=1}^{n} \frac{x_i}{100} \right)^2, \quad x_0 = [0.5, 0.5, ..., 0.5] \]

4. Raydan (1)

\[ f(x) = \sum_{i=1}^{n} \frac{\exp(x_i) - x_i}{10}, \quad x_0 = [1, 1, ..., 1] \]

5. Extended Tridiagonal 1

\[ f(x) = \sum_{i=1}^{n} (x_{2i-1} + 2i - 3)^2 + (x_{2i-1} - x_{2i} + 1)^4, \quad x_0 = [2, 2, ..., 2] \]

6. Generalized Tridiagonal 2 Function

\[ f(x) = ((5 - 3x_1 - x_1^2)x_1 - 3x_2 + 1)^2 + \sum_{i=1}^{n-1} ((5 - 3x_i - x_i^2)x_i - 3x_{i+1} + 1)^2 + ((5 - 3x_n - x_n^2)x_n - x_{n+1} + 1)^2, \quad x_0 = [-1, -1, ..., -1] \]

7. Extended Powell Function

\[ f(x) = \sum_{i=1}^{n} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4, \quad x_0 = [3, -1, 0, ..., 3, -1, 0, 1] \]

8. Quadratic Diagonal Perturbed

\[ f(x) = \left( \sum_{i=1}^{n} x_i \right)^2 + \sum_{i=1}^{n} \frac{x_i}{100}, \quad x_0 = [0.5, ..., 0.5] \]

9. Extended Wood Function

\[ f(x) = \sum_{i=1}^{n} 100(x_{4i-3}^2 - x_{4i-2})^2 + (x_{4i-3} - 1)^2 + 90(x_{4i-1}^2 - x_{4i})^2 + (1 - x_{4i-1})^2 + 10.1(x_{4i-2} - 1)^2 + (x_{4i} - 1)^2 + 19.8(x_{4i-2} - 1)(x_{4i} - 1), \quad x_0 = [-3, -1, -3, ..., -3, -1, -3, -1, -3, -1] \]
10. Extended Tridiagonal 2 Function

\[ f(x) = \sum_{i=1}^{n-1} (x_1 x_{i+1} - 1)^2 + c(x_1 + 1)(x_{i+1} + 1), \quad x_0 = [1,1,...,1], c = 0.1 \]

11. NONDILA Function

\[ f(x) = (x_1 - 1)^2 + \sum_{i=2}^{n-1} 100(x_i - x_{i-1}^2)^2 \quad x_0 = [-1,...,-1] \]

12. DIXMANE Function

\[
\begin{align*}
f(x) &= 1 + \sum_{i=1}^{n-1} \alpha \int_{\pi}^{\frac{1}{n}} x_1^2 (L_\pi)^{k_1} + \sum_{i=1}^{n-1} \beta x_1^2 (x_{i+1} + x_{i-1})^2 (L_\pi)^{k_2} + \sum_{i=1}^{2m} \gamma x_1^4 x_{i+2m}^2 (L_\pi)^{k_3} \\
&\quad + \sum_{i=1}^{2m} \delta x_i x_{i+2m} (L_\pi)^{k_4}, \\
\end{align*}
\]

\[ \alpha = 1, \beta = 0, \gamma = 0.125, k1=1, k2=0, k3=0, k4=1, \quad x_0 = [2,2,...,2] \]

13. Tridiagonal Perturbed Quadtratic

\[ f(x) = x_1^2 + \sum_{i=1}^{n-1} \int_{\pi}^{\frac{1}{n}} x_1^2 + (x_{i+1} + x_i + x_{i+1})^2, \quad x_0 = [0.5,0.5,...,0.5] \]

14. ENGAL1 Function (CUTE)

\[ f(x) = \sum_{i=1}^{n-1} (x_1^2 + x_{i+1}^2)^2 + \sum_{i=1}^{n-1} (-4x_i + 3) \quad x_0 = [2,2,...,2] \]

15. Extended Maratos Function, (c=100)

\[ f(x) = \sum_{i=1}^{n/2} x_{2i-1} + c(x_{2i-1} + x_{2i}^2 - 1)^2, \quad x_0 = [1.1,0.1,1.1,0.1,...,0.1] \]

References:


[10] Fried. I., "N-step conjugate gradient minimization scheme for Non- 
quadratic functions", AIAAJ.9, 1997.


conjugies"ev. Francaise Informants Recherché operatioelle,3e 
Annee,16. (1969)


[16] Tassopoulus. A. and Story. C., "A variable metric method using non-
Quadratic model" J. of optimization theory and Applications Vol.43, 