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

Abbas Y. AL-Bayati Abdulghafour J. S. Khalil K. Abbo
College of Computers Sciences & Mathematics
Department of Mathematics

Abstract

In this paper, we present extension forms of Dai, Yuan (DY) and Fletcher, Reveres (FR) CG algorithms. Our modifications based on introducing a non-quadratic model (sigmoid function model). These modified algorithms are implemented with Wolfe conditions, where initial step size a_k in each iteration is taken as $a_k = a_{k-1} * \sqrt{\|d_{k-1}\| / \|d_k\|}$ 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 overall these comparisons.

خوارزميات التدرج المترافق الموسع المعتمدة على نموذج سكمود في الأمثلية غير المقيدة

عباس البياتي عبد الغفور جاسم خليل خضر
كلية علوم الحاسبات والرياضيات
قسم الرياضيات

الملخص

في هذا البحث تم تقديم صيغ معممة إلى خوارزميات التدرج المترافق من نوع Dai, Yuan (DY) و Fletcher, Reveres (FR). التطوير اعتمد على استخدام نموذج غير تربيعي وهو دالة سكمود. استخدمت الخوارزميات المستحدثة شروط Wolfe في التطبيق مع أخذ $a_k = a_{k-1} * \sqrt{\|d_{k-1}\| / \|d_k\|}$ وتم التقصي عن التقارب المطلق لخوارزمية DY-CG. اختبرت الخوارزميات المستحدثة باستخدام عدد من دوال الاختبار وتم مقارنة النتائج مع خوارزميات DY و FR الأصلية مع الحصول على نتائج كفوءة في هذا الاختبار.

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 surrey 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:

$$Min\ f(x), \qquad x \in R^n \qquad \ldots(1)$$

where $f : R^n \to R$ is continuously differentiable function bounded from below. Starting from an initial guess: $x_1 \in R^n$, a non linear CG method generates a sequence $\{x_k\}$ as:

$$x_{k+1} = x_k + a_k d_k \qquad \ldots (2)$$

and

$$d_{k+1} = \begin{cases} -g_1 & k = 0 \\ -g_{k+1} + b_k d_k & k \geq 1 \end{cases} \qquad \ldots (3)$$

where a_k is obtained by line search .In eq. (3) b_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].

$$f(x_k + \alpha_k d_k) \leq f(x_k) + c_1 \alpha_k g_k^T d_k \qquad \ldots (4a)$$

$$g_{k+1}^T d_k \qquad \geq c_2 g_k^T d_k \qquad \ldots (4b)$$

where d_k is descent direction, i.e.:

$$g_k^T d_k < 0 \qquad \ldots(5)$$

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

$$\left| g_{k+1}^T d_k \right| \leq -c_2 g_k^T d_k \qquad \ldots(6)$$

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

Different CG algorithms corresponding to different choice for the parameter b_k therefore a crucial element in any CG algorithm is the formula definition of b_k because a_k is not exact in practice and objective

function f is not quadratic many formulas have been proposed to compute b_k four well known formulas b_k are :

$$b_k^{Hs} = \frac{g_{k+1}^T y_k}{y_k^T d_k} \qquad b_k^{HR} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$$

$$b_k^{PR} = \frac{g_{k+1}^T y_k}{g_k^T g_k} \qquad b_k^{Dy} = \frac{g_{k+1}^T g_k}{y_k^T d_k}$$

where HS stands Hestenes and Stiefle [13], FR stands Fletcher and Reevers [11], PR stands Polak Rieber [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 $g_{k+1}^T g_{k+1}$ in the numerator of b_k has 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 b_k has uncertain global convergence for general non-linear functions, but has good performance in practice (see[12]for the details). Therefore CG methods has 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:

$$q(x) = \frac{1}{2} x^T G x + b^T + c \qquad \ldots(7)$$

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]:

$$f(x) = F(q(x)) \quad , \quad q > 0 \qquad ...(8)$$

and

$$\frac{dF}{dq} > 0 \qquad ...(9)$$

The following proportions are immediately derived from the above conditions:

1. Every contour line of $q(x)$ is a contour line of f .
2. If x^* is minimize of $q(x)$ then it's also a minimize of f .

In this area there are various published works.

a) A CG methods which minimize the function :

$$f(x)=(q(x))^p \quad ,p>0 \ , \ x\in R^n \qquad ...(10)$$

in at most n-steps has been described by [10].

b) The special polynomial case

$$F(q(x)) = \varepsilon_1 q(x) + \frac{1}{2}\varepsilon_2 q^2(x) \qquad ...(11)$$

where ε_1 , ε_2 scalars are has been investigated by [7].

c) A rational model has been developed by [16] where

$$F(q(x)) = \frac{e_1 q(x)+1}{e_2 q(x)} \qquad ,e_1 q(x) < 0 \qquad ...(12)$$

d) Another rational model was considered by [1] where :

$$F(q(x)) = \frac{\varepsilon_1 q(x)}{1-\varepsilon_2 q(x)} \qquad ...(13)$$

$$\varepsilon_1 > 0 \quad , \quad \varepsilon_2 < 0$$

2-2 The Extended CG methods

In this paper we consider the new model function defined by:

$$f(x) = F(q(x)) = \frac{q(x)}{1+e^{-q(x)}} \ . \qquad ...(14)$$

We call it as quasi sigmoid function where $q>0$ with further assumption that:

$$\frac{dF}{dq} > 0 \ . \qquad ...(15)$$

From equation (14) we have:

$$\frac{dF}{dq} = F' = \frac{\left(1+e^{-q}\right)+qe^{-q}}{\left(1+e^{-q}\right)^2} = f*(1+\frac{1}{q}-\frac{f}{q})$$

$$\therefore F' = f*(\frac{1+q-f}{q}). \qquad ...(16)$$

Return to equation (14) and solve it for q assuming that $e^{-q} = 1 - q + \frac{1}{2}q^2 + r$

where $r = \sum_{n=3}^{\infty} \frac{(-1)^n}{n!} q^n (1 - \frac{1}{n+1} q)$ it is clear that r < 0 and set $s = r + 1$

Then $f = \frac{q}{\frac{1}{2}q^2 - q + s}$ and

$$q = (1 + \frac{1}{f}) + \sqrt{(1 + \frac{1}{f})^2 - 2s} \qquad ...(17)$$

Use (16) and (17) to compute F' using only function values

$$F' = f * (\frac{2 - f + \frac{1}{f} + a}{1 + \frac{1}{f} + a}) \qquad ...(18)$$

where $a = \sqrt{(1 + \frac{1}{f})^2 - 1}$ assuming $s = \frac{1}{2}$

Now define r_k as follows

$$r_k = \frac{F_k'}{F_{k+1}'} \qquad ...(19)$$

Then we can compute the value of r_k using function value at two points x_{k+1} and x_k from eq. (18).

$$r_k = \frac{f_k(2 - f_k + \frac{1}{f_k} + a_1)}{(1 + \frac{1}{f_k} + a_1)} * \frac{1 + \frac{1}{f_{k+1}} + a_2}{f_{k+1}(2 - f_{k+1} + \frac{1}{f_{k+1}} + a_2)} \qquad ...(20)$$

where: $a_1 = \sqrt{(1 + \frac{1}{f_k})^2 - 1}$, $a_2 = \sqrt{(1 + \frac{1}{f_{k+1}})^2 - 1}$

Our objective is to investigate an extended CG method defined in (2) and (3) (for this paper we consider b^{FR} and b^{Dy}) applied to function $F(q(x))$ obtained by non-linear scaling of $q(x)$.

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

$$\tilde{d}_1 = -\tilde{g}_1 \qquad ...(21)$$

And for $k \geq 0$

$$\tilde{d}_{k+1} = -\tilde{g}_{k+1} + r_k \tilde{b}_k \tilde{d}_k \qquad ...(22)$$

$$\tilde{b}_k = \frac{\tilde{g}_{k+1}^T \tilde{g}_{k+1}}{\tilde{d}_k^T \bar{y}} = \frac{g_{k+1}^T g_{k+1}}{d_k^T (r_k g_{k+1} - g_k)} \qquad ...(23)$$

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

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

We can show that the original DY-CG method and extended DY-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 $x_0 \in R^n$.

The method of DY-CG defined by (2) and (3) with b^{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 r_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: $\tilde{d}_1 = -\tilde{g}_1 = -\frac{dF}{dq}\frac{\partial q}{\partial x} = -F_1'g_1 = F_1'd_1$

Suppose: $\tilde{d}_k = F_k'd_k$, to prove for $k+1$

$$\tilde{d}_{k+1} = -\tilde{g}_{k+1} + r_k \tilde{b}\,\tilde{d}_{k-1}$$

$$= -F_{k+1}'g_{k+1} + \frac{F_k'}{F_{k+1}'}\frac{F_{k+1}^2 g_{k+1}^T g_{k+1}}{F_k'd_k^T(\frac{F_k'}{F_{k+1}'}F_{k+1}'g_{k+1} - F_k'g_k)}F_k'd_k$$

$$= F_{k+1}'[-g_{k+1} + \frac{g_{k+1}^T g_{k+1}}{d_k^T y_k}d_k]$$

$$= F_{k+1}d_{k+1}$$

Hence the DY-CG and EDY-CG methods generates the same set of directions.

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

$$\tilde{d}_{k+1} = -\tilde{g}_{k+1} + r_k \frac{g_{k+1}'^T \tilde{g}_{k+1}}{g_k'^T \tilde{g}_k}\tilde{d}_k \qquad \ldots(24)$$

Also we can show that:

$$\tilde{d}_{k+1} = F_{k+1}'d_{k+1}$$

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 $L=\{x: f(x)\le f(x_i)\}$ is bounded.
3. g is Lipschitz continuous in N, where N is neighborhood of L and $\exists \quad L>0$ S.t:

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

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

Zoutendijk theorem:

Suppose x_1 is starting point for which assumptions (1,2,3) are satisfies. Consider any algorithm of the for $x_{k+1}=x_k+a_k d_k$ where d_k is descent direction and a_k satisfies the standard Wolfe conditions (4a,4b) then:

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

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:

$$\tilde{d}_1 = -\tilde{g}_1$$
$$\tilde{d}_1^T \tilde{g}_1 = -\tilde{g}_1^T \tilde{g}_1$$
$$= -F_1'^2\|g_1\| < 0$$

Suppose: $\tilde{d}_k^T \tilde{g}_k < 0$ i.e. $F_k'^2 d_k^T g_k < 0$

From theorem (1) we have:

$$\tilde{d}_{k+1} = F'_{k+1} d_{k+1} = F'_{k+1}\left[-g_{k+1}+\frac{g_{k+1}^T g_{k+1}}{y_k^T d_k}d_k\right]$$
$$= F'_{k+1}\left[\frac{-g_{k+1}y_k^T d_k + g_{k+1}^T g_{k+1}}{y_k^T d_k}d_k\right]$$
$$\tilde{g}_{k+1}^T \tilde{d}_{k+1} = F'^2_{k+1}\left[\frac{-\|g_{k+1}\|^2 y_k^T d_k + \|g_{k+1}\|^2 g_{k+1}^T d_k}{y_k^T d_k}\right]$$
$$= F'^2_{k+1}\|g_{k+1}\|^2\left[\frac{-g_{k+1}^T d_k + g_k^T d_k + g_{k+1}^T d_k}{y_k^T d_k}\right]$$
$$= F'^2_{k+1}\|g_{k+1}\|^2\left[\frac{g_k^T d_k}{y_k^T d_k}\right]$$

By (4b)

$$y_k^T d_k = g_{k+1}^T d_k - g_k^T d_k \geq C_2 g_k^T d_k - g_k^T d_k$$
$$= (C_2 - 1) g_k^T d_k$$
$$\tilde{g}_{k+1}^T \tilde{d}_{k+1} = F'^2_{k+1} \|g_{k+1}\|^2 \frac{g_k^T d_k}{y_k^T d_k} \leq F'^2_{k+1} \frac{\|g_{k+1}\|^2}{C_2 - 1} < 0$$

$0 < C_2 < 1$

Hence the search direction are descent and independent to r_k.

Theorem (3): Consider any iteration of the form $x_{k+1} = x_k + a_k d_k$ where d_k defined in (22) and a_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:

$$\tilde{d}_{k+1} = -g_{k+1} + \rho_k \frac{\tilde{g}_{k+1}^T \tilde{g}_{k+1}}{\tilde{y}_k^T \tilde{d}_k} \tilde{d}_k$$

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

Or $$d_{k+1} + g_{k+1} = b^{Dy} d_k \qquad ...(26)$$

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

$$b_{k+1}^{Dy} = \frac{d_{k+1}^T g_{k+1}}{d_k^T g_k} \qquad ...(27)$$

we get:

$$\|d_{k+1}\|^2 = (b_{k+1}^{Dy})^2 \|d_k\|^2 - 2 d_{k+1}^T g_{k+1} - \|g_{k+1}\|^2$$

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

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

Completing the squares for last two terms:

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

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

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

i.e.: if k starts from zero we get:

$$\therefore \frac{\|d_k\|^2}{(g_k^T d_k)^2} \le \sum_{i=0}^{k} \frac{1}{\|g_k\|^2} \qquad \forall \quad k \ge 0$$

now if $\|g_k\| \neq 0$ then $\exists$ positive scalar $g > 0$

s.t. $\|g_k\| \ge g \qquad \forall \quad k \ge 0 \rightarrow \frac{\|d_k\|^2}{(d_k^T g_k)^2} \le \frac{}{g}$

$$\therefore \quad \sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} \ge \sum_{k=0}^{\infty} g \quad > \infty$$

Which is contradiction with Zoutendijk theorem hence $\|g_k\| = 0$ 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 sure that EFRCG method generate 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

$C_1 = 0.0001$, $C_2 = 0.9$ and $a_1 = 1/\|g_1\|$ and $a_k = a_{k-1} * \sqrt{\|d_{k-1}\| / \|d_k\|}$

In the all cases the stopping criteria is $\|g_k\| \le 10^{-6}$,

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

All cods are written in double precision FORTRAN (2000) and compiled with f77 default compiler settings, these cods 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 the less than the maximum number of iterations which is considered 2000 in our tests. It is shown in Table (2) 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 the each column by the sum of that column divided by 15(the number of the test problems). In Tables (3) and (4)we presented the performance of the all algorithms in terms of percentage where FR-CG method considered as 100%, from Table(3) we see that all algorithms improves FR-CG method about (4%-37%) in number of iteration (nit), and about (6%-29%) in number of function gradient evolutions for n=100 and for n=500, also from Table (4) observed 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

		DY-CG		FR-CG		EDY-CG		EFR-CG	
PN_0	n	nit	fge	nit	fung	nit	fge	nit	fge
1	10^2	18	34	19	35	18	34	19	35
	500	75	122	76	126	75	119	76	126
2	10^2	10	81	47	93	40	80	55	110
	500	40	89	44	89	40	89	16	94
3	10^2	83	125	95	150	94	147	106	168
	500	226	366	228	371	213	325	215	343
4	10^2	73	113	102	161	76	119	101	160
	500	243	395	380	675	179	285	285	424
5	10^2	10	21	32	64	10	21	32	64
	500	11	22	26	52	11	22	26	52
6	10^2	40	61	37	67	39	60	37	67
	500	42	76	44	77	40	74	42	71
7	10^2	79	151	180	313	87	165	163	308
	500	80	147	248	372	80	152	241	369
8	10^2	46	84	124	231	47	83	124	230
	500	133	232	274	472	120	207	273	471
9	10^2	31	59	71	110	31	59	70	110
	500	24	48	70	126	35	67	42	81
10	10^2	39	59	40	65	37	57	40	70
	500	33	52	39	63	36	56	38	62
11	10^2	15	30	13	25	12	23	13	25
	500	12	24	11	22	12	24	11	22

12	10^2	85	133	121	218	84	131	285	424
	500	179	275	193	355	167	263	205	373
13	10^2	106	166	106	166	101	156	116	176
	500	254	401	220	359	240	381	227	373
14	10^2	55	874	34	57	40	567	30	52
	500	88	929	30	55	28	55	30	55
15	10^2	78	177	89	174	77	173	96	190
	500	67	146	380	675	65	142	196	354
Total		2305	5492	3373	5818	2134	4136	3240	5459

Table (2) Comparison of algorithm for n=1000. 1000

		DY-CG		FRCG		EDYCG		EFRCG	
PN_0	N	nit	fge	nit	fge	nit	fge	nit	fge
1	10^3	38	65	38	65	38	65	38	73
	10^4	35	61	32	60	35	61	33	60
2	10^3	39	85	78	131	39	84	78	131
	10^4	38	86	54	106	38	86	54	106
3	10^3	393	629	349	568	415	649	377	616
	10^4	1051	1654	1417	2160	1241	1949	1289	2077
4	10^3	326	545	*	*	329	554	517	811
	10^4	*	*	*	*	*	*	*	*
5	10^3	15	29	77	129	15	29	77	129
	10^4	13	27	1392	1443	16	33	1384	1430
6	10^3	64	101	73	115	65	102	73	115
	10^4	65	102	180	300	85	153	1293	1366
7	10^3	85	156	*	*	61	114	*	*
	10^4	89	170	*	*	100	191	*	*
8	10^3	160	280	445	711	182	317	445	710
	10^4	609	1072	1279	2203	550	965	1279	2204
9	10^3	26	51	47	84	53	97	56	108
	10^4	28	54	47	84	30	59	104	162
10	10^3	69	1081	43	68	39	59	43	65
	10^4	478	4662	160	3964	500	5363	168	3958
11	10^3	15	31	15	29	14	29	14	28
	10^4	9	20	11	22	8	19	11	22
12	10^3	243	381	345	634	230	361	235	434
	10^4	662	1027	*	*	560	871	*	*
13	10^3	362	564	335	541	321	496	348	562
	10^4	1208	1891	1137	1846	1029	1765	1668	2151
14	10^3	141	3956	142	3616	157	4477	140	3614
	10^4	176	4862	203	5655	129	3327	203	5655
15	10^3	100	235	107	211	102	252	186	389
	10^4	161	452	1132	1496	148	332	646	827
		7144	25950	12184	34988	6964	24382	11601	32845

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

Measure	FR-CG	DY-CG	EFR-CG	EDY-CG
nit	100%	68%	96%	63%
fge	100%	94%	93%	71%

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

Measure	FR-CG	DY-CG	EFR-CG	EDY-CG
nit	100%	58%	95%	57%
fge	100%	74%	93%	69%

Appendix

1. Extended Trigonometric Function

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

2. Extended Rosenbrok Function

$$f(x)=\sum_{i=1}^{\frac{n}{2}} c(x_{2i}-x_{2i-1}^3)^2+(1-x_{2i-1})^2, \qquad x_0=[-1.2,1,...,-1.2,1], \quad c=100$$

3. Perturbed Quadratic Function

$$f(x)=\sum_{i=1}^{n} ix_i^2+\frac{1}{10}\left(\sum_{i=1}^{n}x_i\right)^2, \qquad x_0=[0.5,0.5,...,0.5]$$

4. Raydan (1)

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

5. Extended Tridiagonal1

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

6. Generalized Tridiagonal 2 Function

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

7. Extended Powell Function

$$f(x)=\sum_{i=1}^{n/4}(x_{4i-3}+10x_{4i-2})^2+5(x_{4i-1}-x_{4i})^2+(x_{4i-2}-2x_{4i-1})^4+10(x_{4i-3}-x_{4i})^4,$$

$$x_0=[3,-1,0,1,\ldots,3,-1,0,1]$$

8. Quadratic Diagonal Perturbed

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

9. Extended Wood Function

$$f(x)=\sum_{i=1}^{n/4}100(x_{4i-3}^2-x_{4i-2})^2+(x_{4i-3}-1)^2+90(x_{4i-1}^2-x_{4i})^2+(1-x_{4i-1})^2+10.1\{(x_{4i-2}-1)^2+(x_{4i}-1)^2\}+19.8(x_{4i-2}-1)(x_{4i}-1),$$

$$x_0=[-3,-1,-3,-1,\ldots-3,-1,-3,-1]$$

10. Extended Tridiagonal 2 Function

$$f(x)=\sum_{i=1}^{n-1}(x_ix_{i+1}-1)^2+c(x_i+1)(x_{i+1}+1), \qquad x_0=[1,1,\ldots,1],\ c=0.1$$

11. NONDILA Function

$$f(x)=(x_1-1)^2+\sum_{i=2}^{n}100(x_1-x_{i-1}^2)^2 \qquad x_0=[-1,\ldots,-1]$$

12. DIXMANE Function

$$f(x)=1+\sum_{i=1}^{n}ax_i^2\left(\frac{i}{n}\right)^{k1}+\sum_{i=1}^{n-1}bx_i^2(x_{i+1}+x_{i+1}^2)^2\left(\frac{i}{n}\right)^{k2}+\sum_{i=1}^{2m}gx_i^2x_{i+m}^4\left(\frac{i}{n}\right)^{k3}+\sum_{i=1}^{2m}dx_i\,x_{i+2m}\left(\frac{i}{n}\right)^{k4},$$

$a=1,\ b=0,\ g=0.125,\ k1=1, k2=0,\ k3=0,\ k4=1, \quad x_0=[2,2,\ldots,2]$

13. Tridiagonal Perturbed Quodratic

$$f(x)=x_i^2+\sum_{2}^{n-1}ix_i^2+(x_{i-1}+x_i+x_{i+1})^2, \qquad x_0=[0.5,0.5,\ldots,0.5]$$

14. ENGAL1 Function (CUTE)

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

15. Extended Maratos Function, $(c=100)$

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

References:

[1] AL. Bayate. A., "A new non-quadratic model for unconstrained nonlinear optimization "
Mu tah.J. For research and studies Vol.8, No.1 , 1993.
[2] Andrei. N,"40 conjugate gradient algorithm for unconstrained optimization" ICI Technical Report No.13108 , 2008
[3] Andrei. N., "An unconstrained optimization test functions Collection", Advanced Modeling and Optimization, 10 , 2008.
[4] Andrei. N., "Numerical comparison of conjugate gradient algorithms for unconstrained optimization"
Studies in formtics and control, 16 , 2007.
[5] Andrei. N., "A scaled conjugate gradient algorithms for unconstrained optimization",
Computational opti. And Applications Vol.38, No.3, 2007.

[6] Beal. E. M., "A derivation of conjugate gradients",
In Nonlinear Optimization. (Lootsma F. A., ed) Newyourk Academic Press , 1972.
[7] Boland. W.R., Kamgnia. E. and Kowallik. J.,"A conjugate gradient optimization method invariant to non-linear scaling"
J. of optimization theory and Application,27 , 1979.
[8] Bongartz. I.,. Conn. A. R, Gould. N. I. M. and Toint. P.L., CUTE," Constrained and unconstrained"
testing environment, ACM Trans. Math. Software, 21 , 1995.
[9] Dai. Y. H. and Yuan. Y., "A nonlinear conjugate gradient method with a strong global convergence property"
SIAM J. optimization, 10 , 1999.
[10] Fried. I.,"N-step conjugate gradient minimization scheme for Non-quadratic functions", AIAAJ.9 , 1997.

[11] Fletcher. R. and Reeves. C., "Function immunization by conjugate gradients", Comput .J.7 , 1964.
[12] Hager W. W. and Zhang. H, "A surrevy of nonlinear conjugate gradient methods"
Paaific Journal of optimization.2,2006.
[13] Hestenes. M.R and Stiefel. E. L. ,"Methods of conjugate gradients for solving linear systems"
J. Research Nat.Bur.Standards.49 , 1952.
[14] Polok. E. and Rbiere. G. R. "Note sure La conjugate de directions conjugies"

ev. Francaise Informants Recherché operatioelle,3e Annei ,16. (1969)

[15] Powell. M. J., "Restart Procedure for conjugate gradient method", Math. Programming, 12 , 1977.

[16] Tassopoulus. A. and Story. C., "A variable metric method using non-Quadratic model"
J. of optimization theory and Applications Vol.43, No.3 , 1984.