Investigation on Extended Conjugate Gradient Non-linear methods for solving Unconstrained Optimization

Khalil K. Abbo **Osama M. T. Waiss**
Department Of Mathematic
College of Computer Science and Mathematics
University of Mosul

Received **Accepted**
07 / 04 / 2011 **10 / 07 / 2011**

الملخص

في هذا البحث تم تعميم طريقة Dai-Yuan لطرائق التدرج المترافق باعتبار المعلمة τ_{k+1} في المقام كتركيب خطي. إذ تم احتساب ثلاث قيم جديدة للمعلمة τ_{k+1} بثلاث طرائق مختلفة بافتراض طريقة الانحدار, الترافق البحثي واستخدام اتجاه نيوتن.

تم إثبات خاصية الانحدار الحاد والتقارب الشامل للخوارزميات المقترحة والتجارب العددية على بعض الدوال القياسية أظهرت لنا تحسن واضح على الطرائق الكلاسيكية في هذا المجال.

Abstract

In this paper we have generalized the extended Dai-Yuan conjugate Gradient method by Considering the parameter τ_{k+1} in the denominator of β_{k+1} as a convex combination. Three values of τ_{k+1} are computed in three different ways namely by assuming descent property, Pure Conjugacy and using Newton direction.

The descent property and global convergence for the proposed algorithms are established. Our numerical experiments on some standard test functions show that there are considerable improvement on other classical methods in this field.

1- Introduction:

Consider the unconstrained optimization problem defined by:

$$Min\, f(x), \quad x \epsilon R^n \tag{1}$$

Where $f: R^n \to R$ is continuously differentiable. The line search algorithm for (1) often generates a sequence of iterates $\{x_k\}$ by letting

$x_{k+1} = x_k + \alpha_k d_k \qquad k = 0,1,2,\ldots$ (2)

where x_k is the current iterate point, d_k is a descent search direction i.e. $g_k^T d_k < 0$ and $\alpha_k > 0$ is a step length.

Different choices of d_k and α_k will determine different line search methods [8-10]. These methods are divided into two stages at each iteration:

a) Choose a descent search direction d_k.
b) Choose a step-size α_k along the search direction d_k.

Throughout this paper, we denote $f(x_k)$ by f_k, $\nabla f(x_k)$ by g_k, and $\nabla f(x_{k+1})$ by g_{k+1} respectively. $\|.\|$ denotes the Euclidian norm of vectors.

One simple line search method is the steepest descent method if we take $d_k = -g_k$ as a search direction at every iteration, which has wide applications in solving large-scale minimization problems [11]. However, the steepest descent method often yields zig-zag phenomena in solving practical problems. Which makes the algorithm converge to an optimal solution very slowly or even fail to converge [6]. then the steepest descent (SD) not recommended for practical use.

If $d_k = -H_k g_k$ is the search direction at each iteration in the algorithm, Where H_k is an $n \times n$ matrix approximation to the $[\nabla^2 f_k]^{-1} = G_k{}^{-1}$, then the corresponding line search method is called Newton like method such as quasi-Newton or variable metric etc, on the other hand if $H_k = G_k{}^{-1}$ the method is called Newton method, Which is one of the more the successful algorithm for unconstrained optimization if $G_k{}^{-1}$ is symmetric and positive definite and satisfies quasi-Newton condition given by

$G_{k+1}{}^{-1} y_k = s_k$ (3)

Where $y_k = g_{k+1} - g_k$ and $s_k = x_{k+1} - x_k$

For the general non-liner objective function the convergence of the Newton Algorithm to a solution cannot be guaranteed from an arbitrary initial point x_1. In general if initial point is not sufficiently close to the solution then the algorithm may not posses the descent property, the other drawback of the Newton or quasi- Newton method is required to store and compute matrix H_k at each iteration and these adds cost of storage and computation. Accordingly these methods is not suitable to solve large scale optimization problems in many cases [7].

The conjugate gradient method is very useful for solving (1) especially when n is large and has the following form:

$d_1 = -g_1 \qquad k = 1$

$d_{k+1} = -g_{k+1} + \beta_k d_k$ (4)

Where β_k is a parameter, in the case when f is a convex quadratic function and $\alpha_k = arg\,min_{\alpha>0} f(x_k + d_k)$

The conjugate gradient method is such that the conjugacy condition holds [2], namely

$$d_i^T H d_j = 0 \quad ,\forall\, i \neq j \tag{5}$$

For general non-liner function Dai and Liao in [2] replaced the conjugacy condition (5) to the following form

$$d_{k+1}^T y_k = 0 \tag{6}$$

Which is called pure conjugacy conditions additionally if inexact line search is used also see [2], the condition in (6) can be written as:

$$d_{k+1}^T = -t g_{k+1}^T s_k \tag{7}$$

When $t \geq 0$ is scalar.

Several kinds of formulas for β_k has been proposed. For example Fletcher–Reeves (FR). Polak–Ribiere (PR). and Hestenes–Stiegel (HS). formulas are well Known and they are given by:

$$\beta_k^{FR} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k} \tag{8}$$

$$\beta_k^{PR} = \frac{y_k^T g_{k+1}}{g_k^T g_k} \tag{9}$$

$$\beta_k^{HS} = \frac{y_k^T g_{k+1}}{d_k^T y_k} \tag{10}$$

The global convergence properties of the FR, PR and HS methods without regular restarts have been studied by many researchers [1-3].The conjugate gradient methods with regular restart was also found in [4].

To establish convergence properties of these methods it is usually required that the step size α_k should satisfy the strong Wolfe conditions (SWC):

$$f(x_k) - f(x_k + \alpha_k d_k) \geq -\delta \alpha_k g_k^T d_k \tag{11a}$$

$$|g(x_k + \alpha_k d_k)| \leq -\sigma g_k^T d_k \tag{11b}$$

Where $0 < \delta < \sigma < \frac{1}{2}$. On the other hand, many other numerical methods (e.g. the steepest descent methods and quasi- Newton method) for unconstrained optimization are proved to be convergence under the standard Wolfe conditions (SDWC). which are weaker than the (SWC):

$$f(x_k) - f(x_k + x_k d_k) \geq -\delta \alpha_k g_k^T d_k \tag{12a}$$

$$g(x_k + x_k d_k)^T d_k \geq \sigma g_k^T d_k \tag{12b}$$

Line search strategies require the descent condition

$$g_k^T d_k < 0 \qquad \forall\, k \tag{13}$$

However most of conjugate gradient methods don't always generate a descent search direction [5], so condition (13) is usually assumed in the analysis and implementation. Some strategies have been studied which produce a descent search direction within the framework of conjugate gradient methods for example:

Hiroshi and Naoki in [5] generalized the Dai and Yuan (DY) [3], which is defined as follows:

$$d_{k+1} = -g_{k+1} + \beta^{DY} d_k$$

where $\beta^{DY} = \frac{g_{k+1}^T g_{k+1}}{d_k^T y_k}$ (14)

Their generalization of (14) as follows: they are assumed that

$d_{k+1}^T g_{k+1} = -g_{k+1}^T g_{k+1} + \beta_k d_k^T g_{k+1} < 0$ (15)

Where $\beta_k = \frac{g_{k+1}^T g_{k+1}}{\tau_{k+1}}$ (16)

The equation (15) is equivalent to

$\tau_{k+1} > g_{k+1}^T d_k$ (17)

And they are suggested three different values for τ_{k+1}:

1) $\tau_{k+1} = d_k^T y_k + Max\{d_k^T g_{k+1}, 0\}$ therefore

$$\beta_k = \frac{g_{k+1}^T g_k}{d_k^T y_k + Max\{d_k^T g_{k+1}, 0\}} \quad (18)$$

2) $\tau_{k+1} = d_k^T y_k + t_k Max\left\{\frac{\theta_k}{s_k^T y_k} d_k^T y_k, 0\right\}$

$$\beta_k = \frac{g_{k+1}^T g_k}{d_k^T y_k + t_k Max\left\{\frac{\theta_k}{s_k^T u_k} d_k^T u_k, 0\right\}} \quad (19)$$

Where $t_k \geq 0$ and $\theta_k = 6(f_k - f_{k+1}) + 3(g_k + g_{k+1})^T s_k$ and u_k any vector with $s_k^T u_k \neq 0$

3) $\tau_{k+1} = d_k^T y_k + \frac{\tau_k}{\alpha_k} Max\{\theta_k, 0\}$

$$\beta_k = \frac{g_{k+1}^T g_k}{d_k^T y_k + \frac{\tau_k}{\alpha_k} Max\{\theta_k, 0\}} \quad (20)$$

The algorithms defined in equation (4) with β_k is defined in (18) or (19) or (20) is called Extension of the Dai-Yuan (DY) method and the search direction generated by the above algorithms generates descent direction whenever the condition (17) satisfied for more detail see [5].

This paper is organized as follows: In section 2 we deal with an extension of the DY method and we give another three different values for τ_{k+1} this values are based to the descent property, pure conjugacy condition and Newton direction. In section 3 the convergence analysis studied and in section 4 the numerical experiments are reported.

2- New proposed algorithms

In this section, we try to find new values for τ_{k+1} that satisfies the condition given in equation (17), using three different methods:

2.1 Descent property

Hiroshi and Naoki in [5] show that if the condition (17) is satisfied then the related conjugate gradient (CG) method will be generates always descent directions for all k. Now consider

$\tau_{k+1} = \lambda_k^{(1)} d_k^T y_k + \left(1 - \lambda_k^{(1)}\right) g_k^T g_k$, $\lambda_k^{(1)} \in [0,1]$

Then the search direction can be defined as:

$$d_{k+1} = -g_{k+1} + \frac{(g_{k+1}^T g_{k+1})}{\lambda_k^{(2)} d_k^T y_k + (1-\lambda_k^{(2)}) g_k^T g_k} \quad d_k \tag{21}$$

If $d_{k+1}^T g_{k+1} = -g_{k+1}^T g_{k+1} + \frac{(g_{k+1}^T g_{k+1})}{\lambda_k^{(2)} d_k^T y_k + (1-\lambda_k^{(2)}) g_k^T g_k} d_k^T g_{k+1} < 0$

With simple algebra

$$\lambda_k^{(1)} = \frac{d_k^T g_{k+1}}{d_k^T y_k} \tag{22}$$

Therefore our first new algorithm say (MH1-CG) can be define as MH1

$$d_{k+1} = -g_{k+1} + \lambda_k^{(1)} \beta^{MH1} d_k \tag{23}$$

Where $\beta^{MH1} = \frac{g_{k+1}^T g_{k+1}}{\lambda_k^{(2)} d_k^T y_k + (1-\lambda_k^{(2)}) g_k^T g_k}$ and $\lambda_k^{(1)}$ is defined in equation (22) with the condition if $\lambda_k^{(1)} \leq 0$ set $\lambda_k^{(1)} = 0$ and if $\lambda_k^{(1)} > 0$ set $\lambda_k^{(1)} = 1$.

In equation (23), we multiply β_k by $\lambda_k^{(1)}$ for the purpose of the global convergence.

2.2 Pure Conjugacy property

The second method to evaluate the value of τ_{k+1} is the pure conjugacy condition defined in equation (6), we assume that the following search direction generates conjugate directions

$$d_{k+1}^T y_k = -g_{k+1}^T y_k + \frac{g_{k+1}^T g_{k+1}}{\lambda_k^{(2)} d_k^T y_k + (1-\lambda_k^{(2)}) g_k^T g_k} d_k^T y_k = 0$$

Solve the above equation for $\lambda_k^{(2)}$ where $\lambda_k^{(1)} \in [0,1]$ to get

$$\lambda_k^{(2)} = \frac{g_{k+1}^T g_{k+1} d_k^T y_k - y_k^T g_{k+1} g_k^T g_k}{y_k^T g_{k+1} (d_k^T y_k - g_k^T g_k)} \tag{24}$$

Then the second new algorithm (MH2-CG) say can be defined as

$$d_{k+1} = g_{k+1} + \lambda_k^{(2)} \beta^{MH2} d_k \tag{25}$$

When $\beta^{MH2} = \frac{g_{k+1}^T g_{k+1}}{\lambda_k^{(2)} d_k^T y_k + (1-\lambda_k^{(2)}) g_k^T g_k}$ and $\lambda_k^{(2)}$ in equation (24) and if $\lambda_k^{(2)} < 0$ or $\lambda_k^{(2)} > 1$ set $\lambda_k^{(2)} = 1$

In equation (25), we multiply β_k by $\lambda_k^{(2)}$ for the purpose of the global convergence.

2.3 Assuming parallel to the Newton direction

As we know when initial x_1 is close enough to a local minimum point x^* then the best direction to be followed in the current point x_{k+1} is the Newton direction - $G_{k+1}^{-1} g_{k+1}$. Therefore our motivation is to choose the parameter β_{k+1} in (4) so that for every $k \geq 1$ the direction d_{k+1} given by

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

Can be best direction. Hence using the direction from the equality

$$-G_k^{-1} g_{k+1} = -g_{k+1} + \frac{g_{k+1}^T g_{k+1}}{\lambda_k^{(3)} d_k^T y_k + (1-\lambda_k^{(3)}) g_k^T g_k} d_k \quad (26)$$

When G^{-1} is inverse Hessian which is symmetric and positive definite. Multiply equation (26) by y_k noting that $G_k^{-1} y_k = s_k$ and using equation (3) with simple computations we obtain

$$\lambda_k^{(3)} = \frac{(g_{k+1}^T g_{k+1})(d_k^T y_k) - g_k^T g_k (y_k^T g_{k+1} - s_k^T g_{k+1})}{(y_k^T g_{k+1} - s_k^T g_{k+1})(d_k^T y_k - g_k^T g_k)} \quad (27)$$

Then the third algorithm (MH3-CG) say is given by

$$d_{k+1} = -g_{k+1} + \lambda_k^{(3)} \beta^{MH3} d_k \quad (28)$$

Where $\beta^{MH3} = \frac{g_{k+1}^T g_{k+1}}{\lambda_k^{(3)} d_k^T y_k + (1-\lambda_k^{(3)}) g_k^T g_k}$ (29)

Where $\lambda_k^{(3)}$ computed from (27) with the condition if $\lambda_k^{(3)} < 0$ or $\lambda_k^{(3)} > 1$ set $\lambda_k^{(3)} = 1$

In equation (28), we multiply β_k by $\lambda_k^{(3)}$ for the purpose of the global convergence.

3- Convergence analysis

In this section we have proved the global convergence property of the algorithm MH1. Our proof are based to the theorem given in the paper proposed by Gilbert and Nocedal (Gilbert and Nocedal, 1992), They show that any non-liner conjugate gradient algorithm that satisfies the assumpssion (3.1) below will be globally convergent according to the theorem (1) and theorem (2) (given later on).

Assumption (3.1):

(a)

1- The level set $L = \{x: f(x) \leq f(x_1)\}$ is bounded below, where x_1 is initial estimate for the minimizer.
2- In some neighborhood N of L the objective function f is continuously differentiable and its gradient is Lipchitz continuous
3- The step size α_k satisfies the Wolfe conditions

(b) The parameter β_k satisfies the following inequality

$$0 < \beta_k \leq \beta_k^{FR} \quad , \quad \forall k \geq 0$$

Theorem (1):

suppose that assumption (3.1) hold. consider any method of the form (2) and (4). with $0 < \sigma < \frac{1}{2}$ in SWC, then the method generator descent directions d_k satisfying

$$-\frac{1}{1-\sigma} \leq \frac{g_k^T d_k}{\|g_k\|^2} \leq \frac{2\sigma-1}{1-\sigma} \qquad k = 1,2,3 \ldots$$

Proof (see Gilbert and Nocedal, 1992)

Theorem (2):
Suppose that assumption (3.1) hold. consider any method of the form (2) and (4) with $0 \ll \sigma \ll \frac{1}{2}$, then $\lim_{k\to\infty} inf\|g_{k+1}\| = 0$

Proof (see Gilbert and Nocedal).
Theorem (1) and Theorem (2) shows that for conjugate gradient methods, for which $0 < \beta_k \leq \beta_k^{FR}$ and α_k satisfies strong Wolfe conditions then the methods generates descent direction and they are globally convergent.
Therefore to prove descent property and global convergent to the MH1 or (MH2,MH3)-conjugate gradient methods we need only to show

$$0 < \beta^{MH1} \leq \beta^{FR} \tag{30}$$

To prove the inequality (30). Since $\lambda_k^{(1)}$, α_k are positive scalars ($\lambda_k, \alpha_k \in [0,1]$) and $d_k^T y_k > 0$ by second Wolfe condition then

$$\lambda_k d_k^T y_k + (1-\lambda_k) g_k^T g_k > 0 \tag{31}$$

$$\therefore \frac{\lambda_k^{(1)} (g_{k+1}^T g_{k+1})}{\lambda_k^{(1)} d_k^T y_k + \left(1-\lambda_k^{(1)}\right) g_k^T g_k} > 0$$

To establish the second part of the inequality (30) from (31) we have

$$\lambda_k^{(1)} d_k^T y_k + g_k^T g_k - \lambda_k^{(1)} g_k^T g_k \geq g_k^T g_k - \lambda_k^{(1)} g_k^T g_k \geq \lambda_k^{(1)} g_k^T g_k$$

$$\frac{1}{\lambda_k^{(1)} d_k^T y_k + \left(1-\lambda_k^{(1)}\right) g_k^T g_k} \leq \frac{1}{\lambda_k^{(1)} g_k^T g_k} \tag{32}$$

Multiply equation (32) by $\lambda_k^{(1)} g_{k+1}^T g_{k+1}$ to set

$$\frac{\lambda_k^{(1)} g_{k+1}^T g_{k+1}}{\lambda_k^{(1)} d_k^T y_k + \left(1-\lambda_k^{(1)}\right) g_k^T g_k} \leq \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k} = \beta^{FR}$$

4- Numerical Experiments

This section presents the performance of FORTRAN implementation of our new conjugate gradient algorithms (MH1,MH2 and MH3) on a set of unconstrained optimization test problems taken from (Andrei, 2008). We select (15) large scale test problems in extended or generalized form (see Appendix), for each function we have considered numerical experiments with number of variables $n = 100$ and $n = 1000$.

We have compared the performance of these algorithms versus β_k given in equation (18) [which is better than from the β_k given in equation (19) or (20) see (Heroshi and Naoki,2005)].

All these algorithms are implemented with the standard Wolfe line search conditions with $\delta = 0.001$ and $\sigma = 0.9$, where the initial step-size $\alpha_1 = \frac{1}{\|g_1\|}$ and guess for other iterations i.e. $k > 1$; $\alpha_k = \alpha_{k-1} * \frac{\|d_{k-1}\|}{\|d_k\|}$.

In the all cases the stopping criteria is the $\|g_{k+1}\| < 10^{-6} * \max[1, |f_{k+1}|]$ and the maximum number of iterations is 2000. Our comparisons includes the following:

1- NOI: Number of iterations
2- FGN: Number of function and gradient evalutions which are same in these algorithms
3- Lins: number of calling subroutine $t =$ compute step-size α_k

Table (1) and (2) illustrates the details of the results for $n = 100$ and $n = 1000$.

Table (3) presents the performance of all algorithms in terms of percentage where DY method considered as 100%. From table we see that all algorithms improves DY-CG method about $(2\% - 20\%)$ in terms of NOI.

Table (1): comparison of algorithms w.r. to percentage of NOI, FGN & Lins N=100

	FR	DY	EXDY	MH1	MH2	MH3
	NOI / FGN / Lins	NOI / FGN / Lins	NOI / FGN / Lins	NOI / FGN / Lins	NOI / FGN / Lins	NOI / FGN / Lins
1	659/16198/594	57 / 1062 / 48	13 / 29 / 13	22 / 40 / 14	13 / 28 / 13	47 / 395 / 34
2	19 / 35 / 13	18 / 34 / 13	18 / 33 / 12	18 / 34 / 13	19 / 32 / 10	18 / 34 / 13
3	47 / 93 / 42	40 / 81 / 34	34 / 67 / 26	40 / 81 / 34	34 / 72 / 28	45 / 89 / 39
4	43 / 88 /33	34 / 68 / 26	30 / 65 / 25	34 / 68 / 26	38 / 81 / 32	34 / 77 / 32
5	2001/2025/20	61 / 105 / 42	57 / 91 / 31	60 / 103 / 41	61 / 102 / 38	54 / 99 / 42
6	25 / 43 /15	22 / 44 / 19	21 / 42 / 18	21 / 42 / 18	21 / 42 / 18	26 / 43 / 14
7	15 / 25 / 9	16 / 23 / 6	14 / 20 / 5	14 / 20 / 5	7 / 13 / 5	16 / 26 / 9
8	37 / 67 / 29	40 / 61 / 20	34 / 53 / 18	39 / 59 / 19	37 / 55 / 17	36 / 61 / 24
9	180/313/132	79 / 151 / 71	59 / 111 / 51	71 / 135 / 63	58 / 110 / 51	78 / 143 / 64
10	63 / 98 / 33	63 / 98 / 33	55 / 86 / 29	55 / 86 / 29	55 / 86 / 29	55 / 86 / 29
11	2001/2007/4	14 / 34 / 8	10 / 28 / 9	6 / 7 / 0	6 / 7 / 0	6 / 7 / 0
12	32 / 64 / 31	10 / 21 / 10	6 / 13 / 6	7 / 15 / 7	7 / 15 / 7	15 / 28 / 12
13	74 / 123 / 48	87 / 136 / 48	85 / 130 / 44	84 / 132 / 47	76 / 118 / 41	83 / 139 / 55
14	98 / 157 / 58	104 / 161 / 56	82 / 131 / 48	98 / 153 / 54	89 / 137 / 47	101 / 164 / 62
15	69 / 1202 / 56	28 / 176 / 22	23 / 44 / 18	23 / 43 / 17	27 / 48 / 18	25 / 48 / 20
total		673 / 2255 / 456	541 / 943 / 353	592 / 1018 / 387	548 / 946 / 354	639 / 1439 / 449

Table(2): comparison of algorithms w.r. to percentage of NOI, FGN & Lins N=1000

	FR	DY	EXDY	MH1	MH2	MH3
	NOI / FGN / Lins	NOI / F3GN / Lins	NOI / FGN / Lins	NOI / FGN / Lins	NOI / FGN / Lins	NOI / FGN / Lins
1	1585/44127/1566	13 / 27 / 13	12 / 26 / 12	13 / 27 / 13	17 / 33 / 15	66 / 1515 / 66
2	38 / 65 / 22	38 / 65 / 22	29 / 54 / 20	27 / 50 /18	30 / 58 / 23	28 / 53 / 20
3	78 / 131 / 44	39 / 85 / 35	38 / 81 / 32	38 / 83 / 34	34 / 75 / 29	43 / 91 / 40
4	46 / 92 / 38	34 / 74 / 29	32 / 64 / 24	32 / 69 / 27	37 / 85 / 32	45 / 95 / 36
5	2001 / 2005 / 3	201 / 329 / 120	191 / 308 / 109	189 / 311 / 114	204 / 332 / 120	174 / 287 / 105
6	46 / 741 / 46	26 / 56 / 25	23 /51 / 23	23 / 51 / 23	27 / 50 / 18	21 / 47 / 21
7	127 / 3531 / 124	11 / 19 / 7	9 / 16 /9	10 / 17 / 6	7 / 13 / 5	10 / 17 / 6
8	73 / 115 / 40	64 / 101 / 35	46 / 72 / 24	63 / 99 / 34	55 / 89 / 32	67 / 110 /

						41
9	2001 / 2110 / 108	85 / 156 / 70	79 / 148 / 68	76 / 139 / 62	66 / 125 / 58	74 / 130 / 55
10	61 / 96 / 32	64 / 102 / 35	55 / 87 / 29	52 / 84 / 29	52 / 83 / 28	52 / 84 / 29
11	2001 / 2025 / 11	31 / 65 / 12	9 / 20 / 5	18 / 19 /0	18 / 19 /0	18 / 19 / 0
12	77 / 129 / 51	15 / 29 / 13	12 / 24 / 11	12 / 23 /10	11 / 22 / 10	10 / 21 / 10
13	370 / 616 / 245	250 / 421 / 170	189 / 314 / 124	242 / 406 / 163	255 / 428 / 172	232 / 383 / 150
14	314 / 519 / 204	296 / 469 / 172	341 / 529 / 187	271 / 427 / 155	304 / 466 / 161	326 / 500 /173
15	98 / 1967 / 86	37 / 349 / 27	34 / 59 / 20	26 / 46 / 16	28 / 51 / 19	24 / 47 / 19
total		1204 /2347 /785	1099 /1853 /697	1092 /1851 /704	1145 /1929 /722	1190 /3399 /771

Table(3): comparison of algorithms w.r. to percentage of NOI

N	Measure	DY	EXDY	MH1	MH2	MH3
100	NOI	100%	80.7%	88.3%	81.8%	91.7%
1000	NOI	100%	90.8%	90.5%	95%	98.7%

Appendix

1- Extended Freudenstein & Roth Function

$$f(x) = \sum_{i=1}^{\frac{n}{2}} \left(-13 + x_{2i-1} + \left((5 - x_{2i})x_{2i} - 2\right)x_{2i}\right)^2 + \left(-29 + x_{2i-1} + \left((x_{2i} + 1)x_{2i} - 14\right)x_{2i}\right)^2 ,$$

$$x_0 = [0.5, -2, 0.5, -2, \dots, 0.5, -2]$$

2- Extended Trigonometric Function

$$f(x) = \sum_{i=1}^{n} \left(\left(n - \sum_{j=1}^{n} \cos x_j\right) i(1 - \cos x_j) - \sin x_i\right)^2 ,$$

$$x_0 = [0.2, 0.2, \dots, 0.2].$$

3- Extended Rosenbrock Function

$$f(x) = \sum_{i=1}^{\frac{n}{2}} c\left(x_{2i} - x_{2i-1}^2\right)^2 + (1 - x_{2i-1})^2 ,$$

$$x_0 = [-1.2, 1, \dots, -1.2, 1]. \quad c = 100$$

4- Extended White & Holst Function

$$f(x) = \sum_{i=1}^{\frac{n}{2}} c\left(x_{2i} - x_{2i-1}^3\right)^2 + (1 - x_{2i-1})^2 ,$$

$$x_0 = [-1.2, 1, \dots, -1.2, 1]. \quad c = 100$$

5- Diagonal2 Function

$$f(x) = \sum_{i=1}^{n} \left(exp(x_i) - \frac{x_i}{i}\right) ,$$

$$x_0 = [1/1 , 1/2 , \dots , 1/n].$$

6- Generalized Tridiagonal-1 Function

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

$$x_0 = [2,2,\dots,2].$$

7- Extended Three Exponential Terms Function

$$f(x) = \sum_{i=1}^{\frac{n}{2}}\left(exp(x_{2i-1} + 3x_{2i} - 0.1) + exp(x_{2i-1} - 3x_{2i} - 0.1) + exp(-x_{2i} - 0.1)\right) \ ,$$

$$x_0 = [0.1,0.1,\dots,0.1].$$

8- Generalized Tridiagonal-2 Function

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

$$x_0 = [-1,-1,\dots,-1].$$

9- Extended Powell Function

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

$$x_0 = [3,-1,0,\dots,3,-1,0].$$

10- Extended Block Diagonal BD1 Function

$$f(x) = \sum_{i=1}^{\frac{n}{2}}\left(x_{2i-1}^2 + x_{2i}^2 - 2\right)^2 + (exp(x_{2i-1} - 1) - x_{2i})^2 \ ,$$

$$x_{0=[0.1,0.1,\dots,0.1]}.$$

11- Extended Cliff Function

$$f(x) = \sum_{i=1}^{\frac{n}{2}}\left(\frac{x_{2i-1}-3}{100}\right)^2 - (x_{2i-1} - x_{2i}) + exp\left(20(x_{2i-1} - x_{2i})\right)$$

$$x_0 = [0,-1,\dots,0,-1].$$

12- Extended Tridiagonal-1 Function

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

$$x_0 = [2,2,\dots,2].$$

13- Partial Perturbed Quadratic

$$f(x) = x_1^2 + \sum_{i=1}^{n}\left(ix_i^2 + \frac{1}{100}(x_1 + x_2 + \cdots + x_i)^2\right) \ ,$$

$$x_0 = [0.5,0.5,\dots,0.5].$$

14- Almost Perturbed Quadratic

$$f(x) = \sum_{i=1}^{n} i x_i^2 + \frac{1}{100}(x_1 + x_n)^2,$$
$$x_0 = [0.5, 0.5, \ldots, 0.5].$$

15- VARDIM Function (Cut)

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

References

1) AL-Baali, M. "Descent property and global convergence of the FR method with inexact line search" IMA Journal of Numerical Analysis, 5, 1985.

2) Dai Y. and Liao Z. "New Conjugacy conditions and Related Non-Liner conjugate Gradient Methods" Applied Mathematics & optimization 48, 2001.

3) Dai Y. and Yuan Y. "A Non-Linear conjugate Gradient method with strong global convergence property" SIAM. J. optimization,10 (1999).

4) Gilbert J. and Nocedel J. "Global convergence properties of conjugate gradient methods for optimization" SIAM Journal of optimization, 2, 1992.

5) Hiroshi Y. and Naoki S. "A New Non-Liner conjugate Gradient for unconstrained optimization" Journal of the operation Research Society of Japan, Vol.48, No.4, 2005.

6) Lenberger D. "Liner and Non-Liner programming" end Edition, Addition Wesley, Reading, MA, 1989.

7) Nocedel J. and Wright S. "Numerical optimization" Springer, Berlin, Heidelberg, 1999.

8) Yuan G. and La X. "A New line search method with Trust Region four Unconstrained optimization" communication on Applied Non-liner Analyses Vol. 15.No1, 2008.

9) Yuan G. and La X. and Wei Z. "New Tow point step–size Gradient Methods for solving Unconstrained optimization problems" Natural science Journal of Xiangtan University Vol.29, No.1, 2007.

10) Yuan G. and Wei Z. "New line search Methods for Unconstrained optimization" Journal of Korean Stoical Society, Vol.38, No.1, 2009.

11) Yuan Y. and Sun W, "Theory and Methods of optimization" Science press of China, Beijing, 1999.