The Extended CG Method for Non-Quadratic Models

Abbas Y. Al-Bayati **Basim A. Hassan** **Sawsan S. Ismael**
profabbasalbayati@yahoo.com *basimabas39@gmial.com* *sawsan.sami14@yahoo.com*
College of Computer Sciences and Mathematics *College of Education*
University of Mosul, Mosul, Iraq *University of Mosul*

Received on: 25 / 3 / 2007 **Accepted on: 4 / 11 / 2007**

ABSTRACT

This paper investigates an interleaved algorithm which combines between the extended conjugate gradient with the hybrid method of Touati-Storey. This combined algorithm is based on the exact line search to solve a number of non-linear test functions with different dimensions. Experimental results indicate that the modified algorithm is more efficient than the original Sloboda algorithm.

Keywords: A new conjugate gradient method, Numerical Results and Conclusions.

طريقة التدرج المترافق الموسعة للنماذج غير التربيعية

عباس يونس البياتي **باسم عباس حسن** **سوسن سامي أسماعيل**
كلية علوم الحاسبات والرياضيات، جامعة الموصل *كلية التربية، جامعة الموصل*

تاريخ استلام البحث: 2007/3/25 **تاريخ قبول البحث: 2007/11/4**

الملخص

تم في هذا البحث ربط خوارزمية موسعة للتدرج المترافق مع خوارزمية تهجين المتجهات المترافقة لـِ Touati-Storey. الخوارزمية المقترحة تستخدم خط البحث التام لحل دوال غير خطية ذات أبعاد مختلفة لاختبار كفاءة الخوارزمية المقترحة. أثبتت النتائج العملية أن الخوارزمية المقترحة أكثر كفاءة من الخوارزمية الأساسية لـِ Sloboda.

الكلمات المفتاحية : طريقة التدرج المترافق الجديدة , النتائج العددية والاستنتاجات.

1. Introduction

The problem to be solved by the Sloboda algorithm is to calculate the least value of a general differentiable function of several variables. Let n be the number of variables, x be the vector of variables and q(x) be the objective function and g(x) be the gradient of q(x). i.e.

$$g(x) = \nabla q(x) \tag{1}$$

Conjugate gradient algorithm does not require any explicit second derivatives and it is an iterative method. The sequences of points x_1 , x_2 , x_3 , … are calculated by the successive iteration procedure and it should converge to the point in the space of the variable at which q(x) is least.

The conjugate gradient algorithm was first applied to the general unconstrained minimization problem by Fletcher & Reeves [7]. However, now there are several versions of the algorithm for this calculation. Let x_1 be a given starting point in the space of the variable and let i denote the number of the current iteration starting with i=1. the iteration requires the gradient:

$g_i = g(x_i)$,

if i = 1, let d_i be the steepest descent direction,

$$d_i = - g_i \tag{2}$$

otherwise, for i > 1, we apply the formula:

$d_i = - g_i + \beta_i d_{i-1}$ (3)

where β_i and y_i have the values:

$\beta_i = y_i^T g_{i+1} / y_i^T d_i$ (4)

$y_i = g_{i+1} - g_i$

The vector norms being Euclidian and by searching for the least value of f(x) from x_i along the direction d_i we can obtain the vector x_{i+1}:

$x_{i+1} = x_i + \lambda_i d_i$ (5)

Where λ_i is the value of λ that minimizes the function of one variable.

$\phi_i(\lambda) = f(x_i + \lambda d_i)$ (6)

This complete the value of the iteration, and another one is begun if $f(x_{i+1})$ or g_{i+1} is not sufficiently small, [10].

Let A denote a symmetric and positive definite i×i matrix. For $x \in R^i$, we define:

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

Let $\mathbf{F : R \Rightarrow R^i}$ denote a strictly monotonic increasing function and define:

$f(x) = F(q(x))$ (8)

Such a function is called an extended quadratic function, Spedicato [12].

When a minimization algorithm is applied to f, the ith iterate is denoted by x_i, the corresponding function value by f_i and its gradient by G_i , the function value and gradient value of q are denoted by q_i & g_i , respectively and the derivative of $\mathbf{F_i}$ at q_i is denoted by F_i'. we note that $G_i = F_i' g_i$ and define $\rho_i = c_i / c_{i+1}$ for i, where $\mathbf{c_i = F_i'}$, Al-Assady and Al-Bayati [4]

2. Non-Quadratic Sloboda Method

Sloboda [11] proposed a generalized conjugate gradient algorithm for minimizing a strictly convex function of the general form:

$f(x) = F(q(x)) \quad ; \quad \frac{dF}{dq} > 0$ (9)

This algorithm is as follows:

- **<u>Algorithm 1</u>:**

Step 1: Set i = 1 ; $d_1 = - G_1$; $G_1 = g_1^*$.

Step 2: Compute λ by ELS & set $x_{i+1} = x_i + \lambda_i d_i$.

Step 3: Compute $g_{i+\frac{1}{2}} = g(x - \lambda d/2)$

Step 4: Test for convergence, if achieved stop. if not continue.

Step 5: If i=0 nod(n) go to Step 1, else continue.

Step 6: $g^*_{i+1} = \mathrm{w}\, \mathrm{g}_{\mathrm{i}+\frac{1}{2}} - g^*_i$ where $\mathrm{w} = \dfrac{\mathrm{d}_\mathrm{i}^\mathrm{T}\, \mathrm{g}_\mathrm{i}^*}{\mathrm{d}_\mathrm{i}^\mathrm{T}\, \mathrm{g}_{\mathrm{i}+\frac{1}{2}}}$.

Step 7: Compute the new search direction $\mathrm{d}_{\mathrm{i}+1} = -\mathrm{g}^*_{\mathrm{i}+1} + \beta_\mathrm{i}\, \mathrm{d}_\mathrm{i}$ where:

$$\beta_i = \frac{\mathrm{y}_\mathrm{i}^\mathrm{T}\, \mathrm{g}^*_{\mathrm{i}+1}}{\mathrm{d}_\mathrm{i}^\mathrm{T}\, \mathrm{y}_\mathrm{i}} \quad \text{and} \quad \mathrm{y}_\mathrm{i} = \mathrm{g}^*_{\mathrm{i}+1} - \mathrm{g}_\mathrm{i} \ .$$

Step 8: Set $\mathrm{i} = i + 1$; and go to step 2.

Algorithm 1 terminates after i iterations in the case of a nonlinear scaled quadratic function using ELS.

A more general scaling has been considered by Spedicato [12], such a scaling transforms F into a new function where (9), is satisfied. He shows that the sequence of points generated is invariant with respect to nonlinear scaling if:

$$\mathrm{y} = \frac{\mathrm{g}_{\mathrm{i}+\frac{1}{2}}}{\mathrm{dF}/\mathrm{dq}} - \frac{\mathrm{g}_\mathrm{i}}{\mathrm{dF}/\mathrm{dq}} \tag{10}$$

$$\otimes = \mathrm{v}_\mathrm{i}^\mathrm{T}\, \mathrm{H}_\mathrm{i}^{-1}\, \mathrm{v} = \lambda_\mathrm{i}^2\, \mathrm{g}_\mathrm{i}^\mathrm{T}\, \mathrm{H}_\mathrm{i}\, \mathrm{g}$$

However, this type of scaling also uses functions for which the analytic form is known apriority.

Al-Bayati [1] introduced another family of self-scaling VM-methods given by:

$$\mathrm{H}_{\mathrm{i}+1} = \mathrm{H}_\mathrm{i} + (\mu + \frac{\mathrm{a}}{\mathrm{b}}) \frac{\mathrm{v}_\mathrm{i}\, \mathrm{v}_\mathrm{i}^\mathrm{T}}{\mathrm{b}} + \frac{\theta - 1}{\mathrm{a}} \mathrm{H}_\mathrm{i}\, \mathrm{y}_\mathrm{i}\, \mathrm{y}_\mathrm{i}^\mathrm{T}\, \mathrm{H}_\mathrm{i} - \frac{\theta}{\mathrm{b}} (\mathrm{H}_\mathrm{i}\, \mathrm{y}_\mathrm{i}\, \mathrm{v}_\mathrm{i}^\mathrm{T} + \mathrm{v}_\mathrm{i}\, \mathrm{y}_\mathrm{i}^\mathrm{T}\, \mathrm{H}_\mathrm{i}) \tag{11}$$

where θ is again a free parameter; $\mu = \dfrac{1}{M}$, and a = $\mathrm{y}_\mathrm{i}^\mathrm{T}\, \mathrm{H}_\mathrm{i} \mathrm{y}_\mathrm{i}$, b= $\mathrm{v}_\mathrm{i}^\mathrm{T} \mathrm{y}_\mathrm{i}$

$$\mathrm{M} = (\varepsilon/\mathrm{b})\, \beta + (b/\mathrm{a})\, (1 - \beta)\, ; 0 \le \beta \le 1 \tag{11a}$$

where $\varepsilon = \otimes$ (11b)

$$\mathrm{H}_{\mathrm{i}+1} = \mathrm{H}_\mathrm{i} + (\mu + \frac{\mathrm{a}}{\mathrm{b}}) \frac{\mathrm{v}_\mathrm{i}\, \mathrm{v}_\mathrm{i}^\mathrm{T}}{\mathrm{b}} - \frac{1}{\mathrm{b}} (\mathrm{H}_\mathrm{i}\, \mathrm{y}_\mathrm{i}\, \mathrm{v}_\mathrm{i}^\mathrm{T} + \mathrm{v}_\mathrm{i}\, \mathrm{y}_\mathrm{i}^\mathrm{T}\, \mathrm{H}_\mathrm{i}) \tag{12}$$

If an estimate of the inverse Hessian in maintained (rather than an estimate of the Hessian itself which is sometime preferred) then there is a strong motivation for choosing $\beta = 0$ in (11 a), namely, that H^{-1} is not required, this gives $\mu = a/b$.

However, it is possible to generalize Al-Bayati's family of self-scaling VM-updates (12) to be invariant to a nonlinear scaling by the following algorithm, Al-Bayati [2].

- **<u>Algorithm 2</u>:**

Step 1: Set i = 1; H_1 = I ; d_1 = - H_1 G_1 ; $\mathrm{G}_1 = \mathrm{g}_1^*$.

Step 2: Compute $\mathrm{x}_{\mathrm{i}+1} = \mathrm{x}_\mathrm{i} + \lambda_\mathrm{i}\, \mathrm{d}_\mathrm{i}$; λ determined by ELS.

Step 3: Set $\mathrm{g}_1^* = \mathrm{w}\, \mathrm{g}_{\mathrm{i}+\frac{1}{2}} - \mathrm{g}_\mathrm{i}^*$; $\mathrm{w} = \dfrac{\mathrm{d}_\mathrm{i}^\mathrm{T}\, \mathrm{g}_\mathrm{i}^*}{\mathrm{d}_\mathrm{i}^\mathrm{T}\, \mathrm{g}_{\mathrm{i}+\frac{1}{2}}}$ and $\mathrm{g}_{\mathrm{i}+\frac{1}{2}} = \mathrm{g}(\mathrm{x}_\mathrm{i} - \dfrac{1}{2} \lambda\, \mathrm{d})$.

Step 4: Test for convergence; if not continue.

Step 5: Update the matrix H using (12).
Step 6: Compute $d_{i+1} = - H_i\, g^*_{i+1}$.
Step 7: Set $i = i + 1$ and go to Step 2.

In this section we shall describe also another algorithm which effectively interleaves CG & VM-steps. It is also related to one given originally by Buckley [6], but our implementation differs in that we use the scaled quadratic model instead of the quadratic itself.

CG-algorithm 1. and the generalized VM-algorithm 2. The objective here is to show that, using Al-Bayati's self-scaling VM-update (12), the sequence of the generating points is the same in the generalized CG-algorithm 1.

Before making a few more observations we shall outline briefly the proposed strategy for the interleaved generalized CG-VM method Al-Bayati [2].

- **Algorithm 3:**

Let f be a non linear scaling of the quadratic function f; given x_1 and a matrix $H_1 = I$; set $G_1 = g^*_1$; i = 1 and t = 1 initially, k is the iteration index.

Step 1: Set $d_t = - H_i\, g^*_t$ (13)

Step 2: For k = t , t+1 , t+2 , … iterate with

$$x_{k+1} = x_k + \lambda_k\, d_k ,$$

$$\beta_k = (g^{*T}_{k+1}\, H_i\, \bar{y}_k) \Big/ d^T_k\, \bar{y}_k ,$$

$$d_{k+1} = - H_i\, g^*_{k+1} + \beta_k\, d_k ,$$

$$\beta_k = \frac{y^T_k\, g^*_{k+1}}{d^T_k\, y_k} , \quad w = \frac{d^T_i\, g^*_i}{d^T_i\, g_{i+1/2}}$$

$$g^*_{k+1} = w_k\, g_{k+1/2} - g^*_k$$

$$g_{k+1/2} = g(x_k - \frac{1}{2} \lambda_k\, d_k)$$

$$\bar{y} = g^*_{k+1} - g_k$$

Here *i* is the index of the matrix updated only at restart steps and k is the index of iteration and the algorithm is not converged, until a restart is indicated.

Step 3: If a restart is indicated, namely that the Powell [9] restarting criterion is satisfied, i.e.

$$|g^{*T}_{k+1}\, g_k| \geq 0.2\, |g^{*T}_{k+1}\, g_{k+1}|, \tag{14}$$

Then reset t to the current k , update H_i by:

$$H_{i+1} = H_i - (H_i\, y_t\, v^T_t) \Big/ b_t + v_i\, [(2\, a_t \big/ b^2_t)\, v_t - (H_i\, y_t \big/ b_t)^T] .$$

where $a_t = y^T_t H_i y_t$ and $b_t = v^T_t\, y_t$

Step 4: Replace i by i + 1 and repeat from (13).

3. Hybrid Conjugate Gradient Method

Despite the numerical superiority of PR-method over FR-method the later has better theoretical properties than the formal see Al-Baali [3]. Under certain conditions FR-method can be shown to have global convergence with exact line search Powell [10] and also with inexact line search satisfying the strong Wolf-Powell condition. This anomaly leads to speculation on the best way to choose β_i .

Touati – Ahmed and Storey in (12) proposed the following hybrid method:

Step 1: if $\lambda \; \|\mathbf{g_{i+1}}\|^2 \le \mathbf{(2m)^{i-1}}$, with $\frac{1}{2} > \mathbf{m} > \eta$ and $\lambda > 0$ go to Step 2. Otherwise, Set $\beta_i = 0$.

Step 2: If $\beta_i^{PR} < 0$, Set $\beta_i = \beta_i^{PR}$, otherwise go to Step 3.

Step 3: If $\beta_i^{PR} \le (1/2\text{m})\left(\| g_{i-1} \|^2 / \| g_i \|^2 \right)$, with $\mathbf{m} = \eta$, set $\beta_i = \beta_i^{PR}$. Otherwise Set $\beta_i = \beta_i^{PR}$.

Here m, η and λ user supplied parameters. This hybrid was shown to be globally convergence under both exact and inexact line searches and to be quite competitive with PR- and FR-methods.

4. New hybrid algorithm (Algorithm 4):

Step1 : Set $d_t = - H_i \, g_t^*$, i = 1, t = 1, k is the index of iterations.

Step 2: For k = t , t+1 , … iterate with

$$x_{k+1} = x_k + \lambda_k \, d_k \ ,$$

and

$$\beta_k = (g_{k+1}^{*T} \, H_i \, \bar{y}_k) / d_k^T \, \bar{y}_k \ ,$$

Where i is the index of the matrix updated only at restart steps.

Step 3: If $\lambda \; \|\mathbf{g_{i-1}}\|^2 \le \mathbf{(2m)^{k+1}}$ with $\frac{1}{2} > \mathbf{m} > \eta$ go to Step 1. Otherwise set $\beta_i = 0$.

Step 4: If $\beta_i^{PR} < 0$ set $\beta_i = \beta_i^{PR}$, otherwise go to Step 5.

Step 5:If $\mathbf{B_i^{PR}} \le (\frac{1}{2}\mathbf{m}) \; \|\mathbf{g_{i-1}}\|^2/\|\mathbf{g_i}\|^2$ with $m > \eta$, Set $B_i = B_i^{PR}$, otherwise set $B_i = B_i^{FR}$.

Step 6: Compute $d_{k+1} = - H_i \, g_{k+1}^* + \beta_k \, d_k$, where $g_{k+1}^* = w_k \, g_{k+1/2} - g_k^*$

Step 7: If a restart is indicated, namely that the Powell, restarting criterion is satisfied, i.e.: $|\mathbf{g_{k-1}^{*T}} \; \mathbf{g_k} \; | \ge \mathbf{0.2} \; |\mathbf{g_k^{*T}} \; \mathbf{g_k} \; |$, then reset t to the current k, update H_i by:

$$H_{i+1} = H_i - (H_i \, y_t \, v_t^T) / b_t + v_t \, [(2 \, a_t / b_t^2) \, v_t - (H_i \, y_t / b_t)]^T$$

Step 8: Replace i by i+1 and repeat from (13).

5. Conclusions and Numerical results:

Several standard test functions were minimized ($2 \le n \le 400$) to compare the proposed algorithm with standard Sloboda algorithm which are coded in double precision Fortran 90. The proposed hybrid algorithm needs matrix calculation for 400×400, this is the approximately the latest range for this computation of the matrix. The numerical results are obtained on personal Pentium IV Computer. The compete set of results are given in tables 5.1 and 5.2.

The linear search routine used was a cubic interpolation which use function and gradient values and it is adaptation of the routine published by Bunday [5].

We tabulate for all the algorithms; the number of functions evaluations (NOF) and the number of iterations (NOI). Overall totals are also given for NOF and NOI with each algorithm.

Table 5.1 gives the comparison between the standard Sloboda algorithm and the proposed algorithm. Table 5.2 indicates that the suggested algorithm is more efficient than the standard Sloboda algorithm. Namely, there are an improvement of about (53 %) in both NOI and NOF according to our selected group of test functions.

Table 5.1
Comparison between Sloboda method and the proposed hybrid method for $2 \leq n \leq 400$

Test function	**Dimension**	**Sloboda method**		**Hybrid method**	
		NOI	**NOF**	**NOI**	**NOF**
ROSEN	2	33	85	9	30
BEAL	2	10	26	8	20
DIXON	2	6	17	5	14
NON.DI	10	16	49	9	30
ROSEN	10	16	49	9	30
POWELL	4	51	130	26	77
WOOD	4	33	80	27	70
SHALLO	4	8	21	8	19
BEAL	40	8	20	9	23
POWELL-3	40	17	37	11	25
POWELL	40	60	165	35	88
ROSEN	60	17	48	9	30
WOOD	60	102	217	28	72
WOLFE	80	52	105	39	79
POWELL	80	90	260	37	99
WOOD	80	98	209	28	72
BEAL	100	8	20	9	23
POWELL	100	112	384	36	90
WOLFE	100	53	107	39	79
WOOD	400	103	213	28	72
WOLF	400	53	107	39	79
TOTAL		946	2349	448	1121

Table 5.2
Performance of the new algorithm in relation to Sloboda's algorithm

Measurement	**Sloboda**	**New Algorithm**
NOI	100	47.35

NOF	100	47.72

All the algorithms terminated when $|f - f_{min}| < 5\times10^{-10}$.

Appendix

These test function are from general literature [8].

1. Generalized Powell Function

$$F(X) = \sum_{i=2}^{n/4}[(x_{4i-3} - 10\,x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2\,x_{4i-1})^4 + 10\,(X_{4i-3} - x_{4i})^4]$$

$x_o = (3, -1, 0, 1, \ldots)^T$

2. Generalized Wood Function

$$F(x) = \sum_{i=2}^{n/4}[100(x_{4i-2} - x_{4i-3}^2)^2 + (1 - x_{4i-3})^2 + 90(x_{4i} - x_{4i-1}^2)^4 + (1 - x_{4i-1}^2)^2 + 1.0]$$

$x_o = (-3, -1, -3, -1, \ldots)^T$

3. Non-diagonal Functions:

$$F(x) = \sum_{i=2}^{n}[100(x_i - x_o^2)^2 + (1 - x_i)^2]$$

$x_o = (-1, \ldots)^T$

4. Generalized Dixon Function

$$F(x) = (1 - x_1)^2 + (1 - x_o)^2 + \sum_{i=2}^{9}(x_i - x_{i+1})$$

$x_o = (-1, \ldots)^T$

5. Wolfe Functions:

$$F(X) = (-x(3 - x_1/2) + 2x_2 - 1)^2 + \sum_{i=1}^{n-1}(x_{i-1} - x_i\,(3 - x_i/2) + 2x_{i+1} - 1)^2 + (x_{n-1} - x_n\,(3x_n/2) - 1)^2$$

$x_o = (-1, \ldots)^T$

6. Shallo Function

$$F(x) = \sum_{i=1}^{n/2}(x_{2i-1}^2 - x_{2i})^2 + (1 - x_{2i-1})^2$$

$x_o = (-2, -2, \ldots)^T$

7. Generalized Rosenbrock Function

$$F(x) = \sum_{i=2}^{n/2}100(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1})^2$$

$x_o = (-1, 2, 1, \ldots)^T$

8. Beale Function

$$F(x) = (1.5 - x_1(1 - x_2^2)) + (2.25 - x_1(1 - x_2^2))^2 + (2.625 - x_1(1 - x_2^3))^2$$

$x_o = (0, 0)^T$

REFERENCES

[1] Al-Bayati, A.Y. (1991). “A new family of self-scaling variable metric algorithms for unconstrained optimization”, Journal of Education & Science, Mosul University, IRAQ, 12, pp. (25-54).

[2] Al-Bayati, A.Y. (2001). “New generalized conjugate gradient methods for the non-quadratic model in unconstrained optimization”, Journal of Al-Yarmouk, Jordan, 10, pp. (9-25).

[3] Al-Baali, M. (1985). “Descent property and global convergence of the Fletcher Reeves method with inexact line search”, IMA Journal of Numerical Analysis, 5, pp.(121-164).

[4] Al-Assady, N. H. and Al-Bayati, A. Y. (1994). “Minimization of extended quadratic function with inexact line searches”, JOTA 82 pp.(139-147).

[5] Bunday, B.D. (1984). “Basic optimization method”, Edward Arnold, Bedford Square, London.

[6] Buckley, A.G. (1978). “A combined conjugate gradient quasi Newton minimization algorithm”, Mathematical Programming, 15, pp.(200-210).

[7] Fletcher, R. and Reeves, C.M. (1964). “Function minimization in conjugate gradients”, Computer Journal, 7, pp.(54-149).

[8] Nocedal J. (2005), " Unconstrained Optimization Test Function Research Institute for Informatics", Center for advanced Modeling and Optimization, Bucharest1, Romania.

[9] Powell, M.J.D. (1977). “Restart procedure for the conjugate gradient method”, Mathematical Programming, 12, pp.(241-254).

[10] Powell, M.J. (2000). “UOBYQA unconstrained optimization by quadratic approximation”, DAMTP 2000, NA14, pp.(2-32).

[11] Sloboda, F. (1980). “A generalized conjugate gradient algorithm”, Numerical Mathematics, 35, pp.(223-230).

[12] Spedicato, E. (1978). “A note on the determination of the scaling parameters in a class of quasi-Newton methods for function minimization”, JOTA, 20, pp.(1-20).