New low storage VM-algorithm for constrained optimization

Abbas Y. Al-Bayati **Hamsa Th. Chilmerane**
profabbasalbayati@yahoo.com *hamsathrot@uomosul.edu.iq*
College of Computer Sciences and Mathematics
University of Mosul

Received on: 13/4/2005 **Accepted on: 4/9/2005**

ABSTRACT

In this paper a new low-storage VM-algorithm for constrained optimization is investigated both theoretically and experimentally. The new algorithm is based on both the well-known Fletcher's low storage algorithm which generates columns Z spanned on the gradient vectors g_1, g_2, ... g_n and the idea of both Buckley and LeNir of combined variable storage-conjugate gradient method. The well-known SUMT algorithm is adapted to implement the new idea. The new algorithm is very robust compared with the standard low-storage Fletcher algorithm and the standard SUMT algorithm which was designed for solving constrained problems, of the numerical results of application very promising.

Keywords: constrained, low storage, SUMT.

خوارزمية جديدة لتقليل الخزن للمتري المتغير في الأمثلية المقيدة

عباس يونس البياتي **همسة ثروت جلميران**
كلية علوم الحاسوب والرياضيات
جامعة الموصل

تاريخ استلام البحث: 2005/4/13 **تاريخ قبول البحث: 2005/9/4**

الملخص

في هذا البحث تم استحداث خوارزمية جديدة لتقليل الخزن في المتري المتغير للأمثلية المقيدة وتم دراستها نظرياً وعمليا. الخوارزمية الجديدة تعتمد على خوارزمية Fletcher القياسية لتقليل الخزن التي تولد أعمدة Z_1 المكونة من المتجهات المترافقة.$g_1,g_2,\ldots g_n$ الخوارزمية الهجينية لفكرة Buckley & LeNirالتي تربط خوارزميتي المتري المتغير بخوارزمية المتجهات المترافقة . الخوارزمية المعروفة SUMT تم تطويرها لكي تكون مناسبه للخوارزمية الجديدة . وهذه الخوارزمية نشطه نظرياً وعملياً وتم مقارنتيها مع خوارزميات Fletcher القياسية لتقليل الخزن وخوارزمية SUMT القياسية مع الحصول على نتائج مشجعة جداً.

الكلمات المفتاحية: مقيدة، خزن قليل، SUMT.

1. Interior point method (Barrier function):

In the interior penalty function method, the Barrier term is defined to keep the solution from leaving the feasible region. It is impossible to directly handle equality constraints in this method. Thus the original problem for this method is defined as follows:

$$\left.\begin{array}{l} Mininum\ \ f(x) \\ c_i(x) \succ 0 \end{array}\right\} \quad \text{for } i = 1, \ldots\ldots, m \qquad \ldots(1)$$

Two commonly used interior penalty functions are defined as follows:
-Logarithmic barrier function

$$\theta(x_k, \mu_k) = f(x_k) - \mu_k \left[\sum_{i=1}^{m} \log(c_i(x_k)) \right] \qquad \ldots(2)$$

-Inverse barrier function

$$\theta(x_k, \mu_k) = f(x_k) + \mu \left[\sum_{i=1}^{m} \frac{1}{c_i(x_k)} \right] \qquad \ldots(3)$$

Where μ is a positive constant , known as the barrier parameter. It is chosen that the inequality constraints are satisfied(>0) always ,and a positive term is added to the objective function. As we move closer to a constraint boundary $c_i \to 0$, causing the need for a large term to be added to the objective function. Thus the method keeps the solution away from the constraint boundaries and hence is also known as the barrier function method (INT [1]).

2. Low storage methods:

Quasi–Newton (variable metric) methods, which are based on generating an approximation to the inverse of the Hessian matrix, require only the gradient of the objective function .The advantageous due to their fast convergence and absence of second –order derivative computation.
Limited memory Quasi–Newton methods are known to be effective technique for solving certain classes of large–scale unconstrained problems (Buckley and LeNir (1983), Liu and Nocedal (1989), Gilbert and Lemarechal (1989). They make simple approximation of Hessian matrix, which are often good enough to provide a fast rate of linear convergence, and require minimal storage. For these reasons it is desirable to use limited memory approximation for solving problems.(see Richard ,etal.,1990)

3. Fletcher low storage algorithm:

The columns of $g_1, g_2, \ldots\ldots\ldots\ldots, g_n$ z span while z$^?$ contain only unit matrix – like information. Initially $Z_1 = g_{(1)} / \|g_{(1)}\|$ and for $k = 1,2,.........$ the following steps are repeated

(a) $d_k = -(ZZ^T)g_k$

(b) Line search $\rightarrow g_{k+1}, x_{k+1}$

(c) $g^{\perp} = Z^{\perp}Z^{\perp^T}$ g_{k+1}

(d) if $g^{\perp} \neq 0$ extend $[Z \mid g^{\perp} / \| g^{\perp} \|]$else $\tilde{Z} = Z$

(e) Update $\tilde{Z}$ to give Z_{k+1} as for L in the BFGS method.

It is now possible to describe a BFGS like low storage method based of this information structure. Let storage for *e* vectors in Z be available. The method can be followed for *e*– 1 iterations. After which it is possible to carry out PCG steps as the preconditioner for $k \succ e$. These steps are continued as in the Buckley – LeNir method until some test

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

or

$k = n$

Then Z is reset to $g_k / \|g_k\|$ and the whole process is restarted (Fletcher ,1990). The rate of improvement as *e* increase rather slow, at which point the low storage method dose not performe as well as BFGS.

4. Self –Scaling variable metric method:

This is a variable metric CG method depended by Buckely in (1978) for the first time and it combines the CG and QN methods in an attempt to provide their main advantages, i.e. The low storage requirements of CG methods and the rapped convergence of the QN method.The CG – QN algorithm implemented to use a variable amount of storage depending on the variability of space, with minimum requirement of locations. In order to eliminate the truncation and rounding error, the new scalar parameter ? is added to make the sequence and efficiency(as problem dimension)increase. The poor scaling is an imbalance between the values of the function and change in *x* ,the function values may be change very little even though *x* is changing significantly. This difficulty can sometime be remove by good scaling factor for the updating *H* and the performance of self-scaling method is undoubtedly favorable in some cases especially when the number of variables are large (scales,1985).

5. The derivation of new self-scaling parameter:

In QN methods the approximation H_k to the inverse of the Hessian can be selected to satisfy the QN condition which can be written in the form

$H_{k+1} y_k = \tilde{\rho} v_k$ ...(4)

where $\tilde{\rho} = 1$ is scalar

We introduce suitable special alternative equivalent scalars to the special case due to the QN condition. The weakened form is duo to the following secant condition.

$H_{k+1} = \eta H_k$ ...(5)

For such case we are suggested for the first time the following several values of scalar factors

1- $\eta_k = 1$ (to obtained standard $EBFGS$) ...(6)

2- $\eta_2 = \dfrac{-v_k^T g_k}{(g_k^T g_k)^2}$ ...(7)

3- $\eta_3 = \dfrac{-\lambda_k d_k^T g_k}{(g_k^T g_k)^2}$ ...(8)

4- $\eta_4 = \dfrac{\lambda_k g_k^T Z_k \ Z_k^T g_k}{(g_k^T g_k)^2}$ ...(9)

To derive those scalar from the special case of the QN condition which causes weakened form of the secant condition, using the secant condition in QN condition (5) we get,

$\eta H_k \ y_k^T = v_k^T$ ...(10)

We know that $y_k^T = g_{k+1}^T - g_k^T$ that eq. (10) can be write as

$\eta \, H_k (g_{k+1}^T - g_k^T) = v_k^T$ ...(11)

The matrix is defined by

$H_k = Z_k \, Z_k^T$ ...(12)

Submit in (4) we have

$\eta Z_k Z_k^T (g_{k+1}^T - g_k^T) = v_k^T$ ...(13)

Multiplying it by vector g_k, we have

$\eta Z_k Z_k^T (g_{k+1}^T - g_k^T) g_k = v_k^T g_k$ ...(14)

From the definition of orthogonal condition we have

$-\eta g_k^T Z_k Z_k^T g_k = v_k^T g_k$ ...(15)

Z span by $g_1, g_2 \ldots g_n$

$$-\eta g_k^T \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_k \end{bmatrix} \left[g_1^T, g_2^T \ldots g_k^T \right] g_k = v_k^T g_k$$

...(16)

for *k=1,2,……....*and for orthogonal property holds

$$-\eta g_k^T g_k \left[g_1^T, g_2^T \ldots g_k^T\right] g_k = v_k^T g_k \qquad \text{...(17)}$$

$$-\eta g_k^T g_k g_k^T g_k = v_k^T g_k \qquad \text{...(18)}$$

$$\eta = \frac{-v_k^T g_k}{(g_k^T g_k)^2} \qquad \text{...(19)}$$

Which is defined in (7)

we kwon that $v_k = x_{k+1} - x_k = \lambda_k d_k$

$$\eta = \frac{-\lambda_k d_k^T g_k}{(g_k^T g_k)^2} \qquad \text{...(20)}$$

Which is defined in (8)

But we have $d_k = -Z_k Z_k^T\ g_k$ the eq.(20), can be written as

$$\eta = \frac{\lambda_k g_k^T Z_k Z_k^T g_k}{(g_k^T g_k)^2} \qquad \text{...(21)}$$

Which is defined in (9)

5.1 New proposed low storage method:

Step 1: Find an initial approximation x_0 in the interior of the feasible region for the inequality constrains $c(x) \succ 0$. *e* is scalar

Step 2: Set $k=1$ and $\mu_0 = 1$ is the initial value of μ_0, $Z^{\perp}$ is a unit matrix.

Step 3: set $Z = g_k / \|g_k\|$

Step 4: $d_k = -\left(Z_k\ Z_k^T\right) g_k$, where the columns of Z generated by $g_1, g_2, .. g_k$.

Step 5: $x_{k+1}=x_k+?_k\ d_k$

Step 6:if $\left|\frac{(f_{k+1} - f_k)}{f_{k+1}}\right| < \varepsilon$ go to step (6) else go to step (7)

Step 7: check $\mu_k \sum_{i=1}^{m} \frac{1}{c_i\,(x)} < \varepsilon$, then stop else go to (7)

Step 8: if $k = e$ set $H_e=H_n$ and go to step (11)

Step 9: Find $g^{\perp} = Z^{\perp} Z^{\perp^T} g_{k+1}$

Step 10: check g?? 0 extend $\left[Z \mid g^{\perp} / \| g^{\perp} \|\right] = \tilde{Z}$,else $Z = \tilde{Z}$

Step 11: update Z to give Z_{k+1} with new scaling factor, which is define in eq. (21)
Step 12: compute $d_{k+1} = -Z_{k+1} g_{k+1} + ?_k d_k$
Step 13: check for restarting criterion, i.e if
$d^T_{k+1} g_{k+1} \geq -0.8 g^T_{k+1} g_{k+1}$ Satisfied go to step (2) else set $k=k+1$, $\mu_{k+1} = \frac{\mu_k}{10}$ and go to step (4)

6. Numerical results:

Several standard nonlinear constrained test functions are minimized to compare the new algorithm with standard algorithm see (Appendix) with $1 \leq c(x) \leq 7$
FORTRAN programs were written to implement the suggested and previous algorithms. All numerical results quoted here are obtained using (Pentium 4 computer). All cases the stopping criterion taken to be
$\|g_{k+1}\| \langle 1 \times 10^{-5}$
and
$\mu \sum_{i=1}^{m} \frac{1}{c_i(x)} \leq 1 \times 10^{-5}$
All the algorithms in his paper use the same ELS, which is the cubic fitting technique fully, described from (Bundy, 1989).
The comparative performance for all these algorithms are evaluated by considering NOF , NOI and NOC , are considered as the comparative performance of the following algorithms:
1 F/R low storage algorithm (1990).
2 New self – scaling F/R low storage algorithm.
In Table (1) we have compared between F/R low storage algorithm and the new self – scaling F/R low storage algorithm.

Table (1)
Comparison of our new algorithm with standard F/R low storage algorithms

Test fun.	**F/R low storage algorithm NOF (NOI) NOC**			**New algorithm NOF (NOI) NOC**		
1	156	(54)	403	170	(63)	477
2	165	(49)	681	138	(37)	611
3	248	(52)	423	249	(55)	389

4	104	(30)	695	98	(28)	678
5	90	(29)	416	90	(29)	416
6	86	(27)	302	87	(28)	296
7	134	(47)	338	139	(48)	253
8	99	(32)	260	91	(31)	235
9	45	(19)	26	43	(19)	92
10	111	(40)	217	104	(38)	191
Total	1238	(379)	3761	1209	(376)	3638

REFERENCES

[1] Buckly B.and A. LeNir, (1983) " **QN–like variable storage conjugate gradient"**, Mathematical programming 27, 103-340.

[2] Bundy, B. (1984) "***Basic Optimization Method***" Edward Arnold, Bedford Square, London, U.K.

[3] Fletcher, R. (1990) "**Low Storage Methods for Unconstrained Optimization",** Vol.26.

[4] Gillbert and C. Lemarechal, (1989) "**Some numerical experience with variable storage quasi – Newton algorithms**", mathematical programming, 407-436.

[5] Liu and J. Nocedal, (1989) "**On the limited memory BFGS meth**od **for large scale optimization", mathematical programming,** 45, 503-528.

[6] Richard H. Byrd,JorgeNccedal and Robert B. SCHNABEL, (1990)"**Representation of quasi-Newton matrices and their use in limited memory methods".**

[7] Scales L. E. (1985) "**Introduction to Non – Linear Optimization"** Academic Press, London.

[8] **"Interior Penalty `function", 2004** http://library.wolfram.com /conferences/ devconf99/bhatti/Links/Bhatti_Nonlinear_lnk_7.html/

Appendix

1- min $f(x) = x_1 - 2x_2$

s.t

$1 + x_1 - x_2^2 \geq 0$

$x_2 \geq 0$

2- min $f(x) = -x_1 x_2 x_3$

s.t

$20 - x_1 \geq 0$

$11 - x_2 \geq 0$

$42 - x_3 \geq 0$

$72 - x_1 - 2x_2 - 2x_3 \geq 0$

$x_i \geq 0$

3- min $f(x) = -x_1 x_2 x_3$

s.t

$2x_1^2 + x_2^2 + 3x_3^2 \leq 51$

4- min $f(x) = -x_1 x_2 x_3$

s.t

$42 - x_i \geq 0$

$x_1 + 2x_2 + 2x_3 \leq 72$

$x_i \geq 0$

5- min $f(x) = -x_1^2 - x_2^2$

s.t

$x_1^2 + x_2^2 - 17x_1 - 5x_2 + 66 \geq 0$

$x_1^2 + x_2^2 - 10x_1 - 10x_2 + 41 \geq 0$

$$x_1^2 + x_2^2 - 4x_1 - 14x_2 + 45 \geq 0$$
$$-x_i + 7 \geq 0$$

6- min $f(x) = (x_1 - 3)^2 + (x_2 - 4)^2$
s.t
$$2x_1^2 + x_2^2 \leq 34$$
$$2x_1 + 3x_2 \leq 18$$

7- min $f(x) = x_1^2 + x_2^2 - 14x_1 - 6x_2 - 7$
s.t
$$x_1 + x_2 \leq 2$$
$$-x_1 - 2x_2 \leq 3$$

8- min $f(x) = x_1^2 + x_2^2$
s.t
$$x_1 - 1 \geq 0$$
$$x_2 + 1 \geq 0$$

9- min $f(x) = x_1^2 + x_2^2$
s.t
$$x_1^2 + x_2^2 \leq 9$$
$$x_1 + x_2 \leq 1$$

10- min $f(x) = x_1^2 + 3x_2^2 + 1.5x_3$
s.t
$$2x_1 + x_2 + x_3 - 20 \geq 0$$
$$x_1 + x3 - 10 \geq 0$$
$$x_i \geq 0$$