J. KSIAM **Vol.7**, No.1, 7-14, 2003

A NEW LIMITED MEMORY QUASI-NEWTON METHOD FOR UNCONSTRAINED OPTIMIZATION

ISSAM A.R. MOGHRABI

ABSTRACT. The main concern of this paper is to develop a new class of quasi-newton methods. These methods are intended for use whenever memory space is a major concern and, hence, they are usually referred to as limited memory methods. The methods developed in this work are sensitive to the choice of the memory parameter η that defines the amount of past information stored within the Hessian (or its inverse) approximation, at each iteration. The results of the numerical experiments made, compared to different choices of these parameters, indicate that these methods improve the performance of limited memory quasi-Newton methods.

1. Quasi-Newton Method for Unconstrained optimization

Unconstrained Optimization deals with minimizing a certain objective function with no constraints on the solution. This type of problems is of the form :
minimize $f(x)$ (where $f : R^n \to R$) $x \in R^n$
(R^n *is* known as the n-dimensional Euclidean space.)
The solution can be found by using a class of methods known as quasi-Newton method for unconstrained optimization. Quasi-Newton methods require only the function and its first partial derivatives (gradient) to be available. The Hessian (second partial derivatives) is not required to be available or even coded. However, an approximating matrix to the Hessian is used and updated throughout the iterations to incorporate the changes in the function and its gradient.

Given B_i, current approximation to the Hessian, we need to find a new approximating matrix B_{i+1} to the new Hessian, evaluated at the newly computed iterate x_{i+1}. To determine B_{i+1} we may use the Taylor's series approximation of first order to the gradient about the iterate $\underline{x}_{i+1}$to obtain a relation of the form (referred to as the Secant equation):

$$B_{i+1}\underline{s}_i = \underline{y}_i,$$

where

$$\underline{s}_i = \underline{x}_{i+1} - \underline{x}_i\ ,$$

and

$$\underline{y}_i = \underline{g}_{i+1} - \underline{g}_i\ .$$

Therefore, the approximations to the Hessian matrix at each iteration must satisfy the so-called "Secant Equation".

Updating formulae used to find the new Hessian approximating matrix B_{i+1} using information from both the old approximation B_i and the vectors s_i and y_i, have the following form:

$B_{i+1} = B_i + C_i$, where C_i is a correction matrix. Alternatively, it is preferable to use $H_{i+1} = H_i + D_i$, where D_i is a correction matrix and $H_{i+1} = B_{i+1}^{-1}$.

One particular rank-two formula is the well known BFGS formula. This formula is given by:

$$B_{i+1}^{BFGS} = B_i + \frac{\underline{y}_i\ \underline{y}_i^T}{\underline{y}_i^T\ \underline{s}_i} - \frac{B_i\ \underline{s}_i\ \underline{s}_i^T\ B_i}{\underline{s}_i^T\ B_i \underline{s}_i}$$

$$(1) \qquad H_{i+1}^{BFGS} = H_i + \left[1 + \frac{\underline{y}_i^T\ H_i\ \underline{y}_i}{\underline{s}_i^T\ \underline{y}_i}\right] \frac{\underline{s}_i\ \underline{s}_i^T}{\underline{s}_i^T\ \underline{y}_i} - \frac{\underline{s}_i\ \underline{y}_i^T\ H_i\ + H_i\ \underline{y}_i\ \underline{s}_i^T}{\underline{s}_i^T\ \underline{y}_i}.$$

Numerical results indicate that this formula is superior to other updating formulae especially when inaccurate (non-exact) line searches are used. BFGS is considered to be a standard updating formula [5].

A standard Quasi-Newton Algorithm has the form: [4]

```
Start with any estimation point $x_0$ to the minimum
Start with a symmetric positive-definite matrix $H_0$ (usually $H_0 = I$)
$i \leftarrow 0$
Find $\underline{g}_0 = \underline{g}(x_0)$
Repeat
Step 1. Let $\underline{p}_i = -H_i\ \underline{g}_i$
Step 2. Normalize the vector $\underline{p}_i$ [1]
If $i \leq n$ and $\| \underline{p}_i \|_2 > 1$, then $\underline{p}_i = \dfrac{\underline{p}_i}{\| \underline{p}_i \|_2}$.
Step 3. Minimize $f\left(\underline{x}_i + \alpha \underline{p}_i\right)$, where $\alpha \in R$ $(\alpha \geqslant 0)$ to find a step length $\alpha_i$ along $\underline{p}_i$.
Step 4. $x_{i+1} = x_i + \alpha_i\ \underline{p}_i$
        $\underline{s}_i = \underline{x}_{i+1} - \underline{x}_i$ and $\underline{y}_i = \underline{g}_{i+1} - \underline{g}_i$.
Step 5. If $\underline{s}_i^T\ \underline{y}_i > 0$, and $i = 0$, and $n > 10$, then
        $H_0 = \left(\dfrac{\underline{s}_0^T\ \underline{y}_0}{\underline{y}_0^T\ H_0\ \underline{y}_0}\right) H_0$      {*Shanno and Phua Scaling* [1 − 3]}
Step 6. If $\underline{s}_i^T\ \underline{y}_i > 0$, then
        $H_{i+1}^{BFGS} = H_i + \left[1 + \dfrac{\underline{y}_i^T\ H_i\ \underline{y}_i}{\underline{s}_i^T\ \underline{y}_i}\right] \dfrac{\underline{s}_i\ \underline{s}_i^T}{\underline{s}_i^T\ \underline{y}_i} - \dfrac{\underline{s}_i\ \underline{y}_i^T\ H_i\ + H_i\ \underline{y}_i\ \underline{s}_i^T}{\underline{s}_i^T\ \underline{y}_i}$
```

(or any similar updating formula)
Step 7. $i \leftarrow i+1$ until $\| g_i \|_2 < \varepsilon$, {where $\varepsilon \in R$ is a convergence tolerance}.

2. APPLICATION TO LIMITED-MEMORY METHODS

In this section we discuss methods which address the application of Q-N algorithms when the storage resources are incapable of storing the Hessian approximation matrix (or its inverse). We briefly discuss an existing method and present a new alternative for implementing limited-memory methods.

Concentrating on the BFGS updating method 1, we focus attention on the formula for updating the inverse Hessian approximation at iteration i, given as:

$$(2) \qquad H_{i+1} = H_i + \left(\underline{s}_i^T \, \underline{y}_i\right)^{-1} \{ \left[1 + \frac{\underline{y}_i^T \, r_i}{\underline{s}_i^T \, \underline{y}_i} \right] \underline{s}_i \underline{s}_i^T - s_i \, r_i^T - r_i \, s_i^T \}$$

where $r_i = H_i \;\, y_i$.

Limited memory methods are derived using the identity

$$p_{i+1} = -H_{i+1} \, g_{i+1}$$

which is equivalent to

$$(3) \quad p_{i+1} = p_i - r_i - \left(s_i^T \, y_i\right)^{-1} \{ \left(\left[1 + \frac{r_i^T \, y_i}{s_i^T \, y_i} \right] s_i^T g_{i+1} - r_i^T g_{i+1} \right) s_i + \left(s_i^T g_{i+1}\right) r_i \, \}.$$

However, the problem in using 3 lies in computing r_i without explicitly storing H_i.

The simplest case would be to set $H_i = I$ at each iteration and, thus, $r_i = y_i$. However, this is computationally a poor choice since it discards previous information. Another alternative, we propose here, is to store a form of the vector r_i and update it at each iteration. This can be carried out using the following recurrence :-

we start with $r_0 = H_0 \, y_0 = y_0$ (since $H_0 = I$)

$$\begin{aligned} r_1 &= H_1 y_1 \\ &= y_1 + \left(s_0^T \, y_0\right)^{-1} \{ \left(\left[1 + \frac{y_0 \, y_0^T}{s_0^T \, y_0} \right] s_0^T \, y_1 - \left(y_0^T \, y_1\right) \right) s_0 - \left(s_0^T \, y_1\right) \} \end{aligned}$$

and, iteratively, for any i, we use

$$\begin{aligned} r_i &= H_{i-1} y_i + \left(s_{i-1}^T \, y_{i-1}\right)^{-1} \{ \left(\left[1 + \frac{y_{i-1}^T \, r_{i-1}}{s_{i-1}^T \, y_{i-1}} \right] s_{i-1}^T \, y_i - \left(r_{i-1}^T \, y_i\right) \right) s_{i-1} \\ (4) \qquad & \quad - \left(s_{i-1}^T \, y_i\right) \} \end{aligned}$$

Relation 4 can then be used in 2 to compute the new direction vector. Still, the term $H_{i-1} \, y_i$ in 2 is not readily available. The obvious alternative is to set, in 2 r_{i-1} to y_{i-1}, in which case $H_{i-1} \, y_i$ reduces to y_i.

This results in a method which is computationally much better than setting [as in 1] H_i to be the identity matrix, since this means that previously accumulated information is not totally abandoned. This suggestion still discards previous information accumulated in r_{i-1}(since H_{i-1} in 2 is set to I). It may therefore be effective to avoid setting H_{i-1} to I and we prefer, rather, to approximate the product $H_{i-1}y_i$ as will be shown next.

Given a diagonal matrix H_0, one constructs

$H_1 = H_0 + U(H_0, y_0, s_0)$ (in our case, U corresponds to the BFGs double rank correction)

$H_2 = H_0 + U(H_0, y_0, s_0) + U(H_1, y_1, s_1)$ .... etc

Let η be the maximum number of correction matrices U that can be stored. Since H_0 is diagonal, this means that the maximum number of n-vectors that can be used to define the Quasi-Newton matrix is $2\eta + 1$. Once H_η is generated we have reached the storage limit:

$$H_\eta = H_0 + U(H_0, y_0, s_0) + \ldots + U(H_{\eta-1}, y_{\eta-1}, s_{\eta-1})$$

This suggestion is inspired by Nocedal's method ([6]).Nocedal's approach is based on the argument that one should incorporate the most recent information in the update and one appealing choice is to replace the oldest information. However, there is no guarantee now that the generated matrices will be positive-definite and/or that the quadratic termination property 4 will still hold. Losing positive-definiteness results in search directions that are not descent which threatens the convergence of the method. This can be avoided by writing the updating formula [concentrating on the BFGS method] in the form :-

$$H_{i+1} = V_i^T H_i V_i + \rho_i s_i s_i^T ,$$

where $\rho = \left(s_i^T y_i\right)^{-1}$ and $V_i = I + \rho_i \, y_i \, s_i^T$.

Thus, applying the strategy suggested above, Nocedal's method expresses the update as :-

For $i + 1 \leq \eta$,

$$\begin{aligned} H_{i+1} \; = \; & V_i^T \; V_{i-1}^T \ldots V_0 \; H_0 \ldots V_{i-1} \; V_i \\ & + V_i^T .. V_1^T \; \rho_0 \; s_0 \; s_0^T \; V_1 \ldots V_i + \ldots . \\ & + V_i^T .. V_{i-1}^T \; \rho_{i-2} \; s_{i-2} \; s_{i-2}^T \; V_{i-1} \ldots V_i + \\ & + V_i^T \; \rho_{i-1} \; s_{i-1} \; s_{i-1}^T \; V_i + \rho_i \; s_i \; s_i^T \end{aligned}$$

For $i + 1 > \eta$,

$$\begin{aligned} H_{i+1} \; = \; & V_i^T \; V_{i-1}^T \ldots V_{i-\eta+1} \; H_0 \ldots V_{i-\eta+1} \; V_{i-1} V_i \\ & + V_i^T .. V_{i-\eta+2}^T \; \rho_{i-\eta+1} \; s_{i-\eta+1} \; s_{i-\eta+1}^T \ldots V_i + \ldots . \\ & + V_i^T .. V_{i-1}^T \; \rho_{i-2} \; s_{i-2} \; s_{i-2}^T \; V_{i-1} \ldots V_i + \\ & + V_i^T \; \rho_{i-1} \; s_{i-1} \; s_{i-1}^T \; V_i + + \rho_i \; s_i \; s_i^T \end{aligned}$$

The matrices generated by these last two relations are positive-definite, if H_0 is positive definite and $s_i^T\, y_i$ is positive. Also, if the function is quadratic and the vectors $\{s_k\}$ are conjugate with respect to the Hessian of f, then the Hereditary Property,

$$H_i\, y_j = s_j \qquad j = i-1, ..., i-\eta \qquad (i > \eta),$$

holds, since

$$V_j y_j = 0, \text{ for } j = i,\ i-1, ..., i-\eta+1$$

and

$$V_j\, y_k = y_k \quad \text{for every} \quad j > k.$$

Returning to our problem, we use Nocedal's method with a small value of η to approximate the product $H_{i-1}\, y_i$ in 2. For instance, for $\eta = 1$, we have

$$\begin{aligned} H_{i-1}\, y_i &= V_{i-2}^T\, H_0\, V_{i-2}\, y_i + \left(\, \rho_{i-1}\, s_{i-2}^T\, y_i\right) s_{i-2} \\ &= H_0\, y_i - \left[\rho_{i-2}\, s_{i-2}^T\, y_i\right] H_0\, y_{i-2} \\ &\quad + \rho_{i-2}^2 \left[\left(y_{i-2}^T H_0\, y_{i-2}\right)\left(s_{i-2}^T\, y_i\right) - \left(y_{i-2}^T H_0\, y_i\right)\left(s_{i-2}^T\, y_{i-2}\right) s_{i-2}\right] \\ &\quad + \left(\rho_{i-2}\, s_{i-2}^T\, y_i\right) s_{i-2}\,. \end{aligned}$$

In general, we use the following recurrence to compute the above product for a general η :

```
0.LET $k = i - 1$;
1. IF $k \le \eta$   THEN ($incr = 0; bound = k$)
                     ELSE ($incr = k - \eta; bound = \eta$)
2. $q_{bound} = y_i$
3. FOR $l = bound - 1...0$
       { $j = l + incr$ ;
         $\alpha_l = \rho_j\, s_j^T q_{l+1}$ ;
         $q_l = q_{l+1} - \alpha_l\, y_j$ ;}
4. $z_0 = H_0\, q_0$
5. FOR $l = 0, 1, ..., bound - 1$;
       {$j = l + incr$;
        $\beta_j = \rho_j\, y_j^T z_l$;
       IF ($k \succ \eta$) THEN $z_{l+1} = z_l + s_j\left(\alpha_l - \beta_j\right)$
                           ELSE $z_{l+1} = z_l + s_j\,(\alpha_l - \beta_l)$}
```

Lemma 1. *Suppose that the search direction is computed using 3, with r_i given by 4. If the recurrence 4 is applied to a quadratic function with exact line search, then it possesses the quadratic termination property provided that $s_i^T A s_j = 0$, for $i \neq j$, holds.*

Proof. We wish to establish (by induction) the Hereditary Property

$$H_k\ y_j = s_j, \quad \text{for} \quad 0 \le j < k.$$

The property trivially holds for the case $k = 1$. Therefore assume that it holds when $k = i$,

i.e., $H_i \ y_j = s_j$, for $j = 0, ...i-1$
We wish to prove that
$H_{i+1} \ y_j = s_j$, for $j = 0, ...i-1$
To carry out the proof, we need first to establish that

$$r_k^T \ y_j = 0 \ for \ \ 0 \le j < k \ .$$

For the case $k = 1$, we have, using 2, $r_1^T \ y_0 = 0$
Assume that $r_k^T \ y_j = 0 \ \ for \ \ 0 \le j < k$ holds for the case $k = i-1$. We now consider the case $k = i$:-
From 2, we have [for $j = 0..i-2$].

$$\begin{aligned} y_j^T r_i &= y_j^T H_{i-1} \ y_i + \alpha s_{i-1}^T \ y_j - \beta r_{i-1}^T y_j \\ &= s_j^T y_i + \alpha s_{i-1}^T \ y_j - \beta r_{i-1}^T \ y_j \end{aligned}$$

for

$$\alpha = \left(s_{i-1}^T \ y_{i-1}\right)^{-1} \{ \left[\frac{1 + y_{i-1}^T \ r_{i-1}}{s_{i-1}^T \ y_{i-1}} \right] s_{i-1}^T y_i - \left(r_{i--1}^T \ y_i\right) \}$$

$$\beta = \left(s_{i-1}^T \ y_{i-1}\right)^{-1} \left(s_{i-1}^T \ y_i\right)$$

and using $H_i \ y_j = s_j$ for $j = i-1, ..., i-\eta$ and our inductive hypothesis. It is now straightforward to prove that $r_k^T \ y_j = 0 \ for \ \ 0 \le j < k$. ∎

Our experiments with the above method seem to favour method 3 for $\eta = 1$. For larger values of η, Nocedal's method seems to be preferred. We thus use 3 with $\eta = 1$ in experimenting with the limited-memory version of the BFGS method.

3. Numerical Results and Conclusions

The multi-step quasi-Newton methods are tested on 30 different functions having different dimensions varying from dimension 2 to 100. The functions tested are grouped into three categories and they are found in [4, 7, 8].

1- Functions of Fixed Low Dimension $(2 \le n \le 15)$:
We have tested 14 functions having fixed low dimensions, and these are listed in the following table

Function Name (dimension)
Rosenbrock $(n = 2)$
Quadratic function $(n = 2)$
Freudenstein & Roth$(n = 2)$
Powell Badly Scaled $(n = 2)$
Brown Badly Scaled $(n = 2)$
Beale $(n = 2)$
Beale $(n = 2)$
Bard $(n = 3)$

Gaussian ($n = 3$)
Box three-dimensional ($n = 3$)
Powell Singular ($n = 4$)
Wood ($n = 4$)
Biggs EXP6 ($n = 6$)
Quadratic Function ($n = 6$)

2 - Function of Fixed Medium Dimension ($16 \leq n \leq 45$):
We have only one fixed medium dimension function, and it is given by

Function Name (dimension)
Sum of Quartics function ($n = 25$)

3 - Function of Variable Dimension ($2 \leq n \leq 100$) :
We have 15 functions of variable dimension. We have chosen the dimension n to vary according to following ranges

Low dimension ($2 \leq n \leq 15$)
Medium dimension ($16 \leq n \leq 45$)
Moderate High dimension ($46 \leq n \leq 80$)
High dimension ($81 \leq n \leq 100$)
The names of these functions are listed in the following table

Function Name (dimension)
Watson function ($3 \leq n \leq 31$)
Extended Rosenbrock ($2 \leq n \leq 100, n$ even)
Extended Powell ($2 \leq n \leq 100, n$ divisible by 4)
Penalty function I ($2 \leq n \leq 100$)
Variably dimensioned function ($2 \leq n \leq 100$)
Trigonometric function ($2 \leq n \leq 100$)
Modified Trigonometric function ($2 \leq n \leq 100$)
Broyden Tridiagonal function ($2 \leq n \leq 100$)
Discrete Boundary value function ($2 \leq n \leq 100$)
Oren and Spedicato Power function ($2 \leq n \leq 100$)
Full Set of Distinct Eigen Values Problem ($2 \leq n \leq 100$)
Tridiagonal function ($2 \leq n \leq 100$)
Wolfe function ($2 \leq n \leq 100$)
Diagonal Rosenbrock's function ($2 \leq n \leq 100$, n even)
Generalized Shallow function ($2 \leq n \leq 100$, n even)

The overall numerical results are given in table 1. The tables reveal the total function and gradient evaluations for each method, the total number of iterations, the total time taken by each method, and the number of scores for each method (a score is given to a function if it has the minimum number of evaluations with ties resolved on the number of iterations).

The comparisons between the limited memory BFGS ($LMBFGS$) method and the new methods (LM, for different values of η) are given in table 1. These show clearly that the LM_1 (corresponding to $\eta = 1$) method is really competitive to both $LMBFGS$ and the other limited memory methods. LM_1, gives the best performance especially for medium dimension problems.

Table 1 : Overall Results (876 problems)

Method	Evaluations	Iterations	Time (sec.)	Scores
$LMBFGS$	86401(100.00%)	73090 (100.00%)	39171.185 (100.00%)	128
LM_1	76164 (88.15%)	61335 (83.92%)	31474.713 (80.35%)	179
LM_2	77364(89.54%)	61404 (84.01%)	31547.687 (80.54%)	132
LM_3	92105(106.60%)	71520 (97.85%)	37164.552 (94.88%)	98

The new methods have shown their superiority in the numerical experiments compared to the original Nocedal's limited memory $BFGS$ method. In specific, the new method LM_1 proves overall that it is numerically superior especially in medium dimension problems.

References

[1] P. Gill, W. Murray and M. Wright, Numerical Linear Algebra on Optimization, volume I, Addison-Wesley, U.S.A., (1991).
[2] R.F. Fletcher, Pratical Methods Of Optimization, John Wiley, Great Britain, (1991).
[3] J.A. Ford and A.F. Saadallah, Efficient utilization of function-values in unconstrained minimization, Colloq. Math. Soc. Jan. Bolyai, 50, 539-563 (1986).
[4] J.E. Dennis and R.B. Schnabel, Minimum change variable metric update formulae, SIAM Review, 21, 443-459 (1979).
[5] C.G. Broyden, The convergence of a class of double-rank minimization algorithms, J. Inst. Math. Applic.,6, 76-90 and 222-231 (1970).
[6] Nocedal, A Memoryless variable metric algorithms, Comput. J., 13, 317-322 (1980).
[7] D. Goldfarb, A family of variable metric methods derived by variational means, Maths. Comp., 24, 23-26 (1970).
[8] D.F. Shanno, Conditioning of quasi-Newton methods for function minimization, Maths. Comp., 24, 647-656 (1970).

Math Dept., Faculty of Science
Beirut Arab University
Beirut, Lebanon
email: i_moghrabi@yahoo.com