Math. J. Okayama Univ. **47** (2005), 175–188

MODIFICATIONS OF THE LIMITED MEMORY BFGS ALGORITHM FOR LARGE-SCALE NONLINEAR OPTIMIZATION

Leong Wah JUNE and Malik Abu HASSAN

Abstract. In this paper we present two new numerical methods for unconstrained large-scale optimization. These methods apply update formulae, which are derived by considering different techniques of approximating the objective function. Theoretical analysis is given to show the advantages of using these update formulae. It is observed that these update formulae can be employed within the framework of limited memory strategy with only a modest increase in the linear algebra cost. Comparative results with limited memory BFGS (L-BFGS) method are presented.

1. Introduction

Large-scale unconstrained optimization is to minimize a nonlinear function $f(x)$ in a finite dimensional space, that is

$$\min_{x\in\Re^n} f(x) \tag{1.1}$$

where n is large. We assume throughout that both the gradient $g(x) = \nabla f(x)$ and the Hessian matrix $G(x) = \nabla^2 f(x)$ of f exist and are continuous.

In 1980 Nocedal [8] introduced limited memory BFGS (L-BFGS) update for large-scale unconstrained optimization. Subsequent numerical studies on large-scale problems have shown that methods based on this updating scheme can be very effective if the updated inverse Hessian approximation are rescaled at every iteration [6],[13]. Indeed the L-BFGS method is currently the winner on many classes of problems and competes with truncated Newton methods on a variety of large-scale nonlinear problems [13].

The L-BFGS method is a matrix secant method specifically designed for low storage and linear algebra costs in computation of a Newton-like search direction. This is done by employing a clever representation for the L-BFGS update. Recall that the L-BFGS update is obtained by applying BFGS update to an initial positive definite diagonal matrix (a scaling matrix) using data from the few most recent iteration. The search direction is computed by a simple matrix-vector multiplication. Change in the initial scaling matrix and the data from past iterations can be introduced into the update at low

Mathematics Subject Classification. Primary 65; Secondary K10, H10.

Key words and phrases. Large-scale optimization, limited memory methods, BFGS update.

cost. This is especially true if the initial scaling matrix is a positive multiple of identity matrix as it is usually taken to be in practice. Once the search direction is computed, an appropriate step-length is obtained from one of the standard line search procedures.

In this paper we observe that some modified BFGS update such as Biggs'[1], [2] BFGS update and Yuan's[12] BFGS update can be employed within the framework of limited memory strategy. Since these modified updates are known to be quite efficient in minimization of low-dimensional nonlinear objective functions, such an approach may improve efficiency of the BFGS method in limited memory scheme. Moreover, only a modest increase in the linear algebra cost when applying these modified updates.

Therefore, the aim of this paper is to propose an algorithmic framework, which tries to adopt the previous advantages while still ensuring global convergence towards stationary points. It is based on the simple idea of approximating the objective function by different techniques. In Section 2 we will describe these techniques. The limited memory methods using these update formulae are given in Section 3 and 4. In Section 5 the results of our numerical experiments are reported. Convergence results are given in Section 6, and finally the conclusions are made in Section 7.

2. Modified BFGS Methods using Different Function Interpolation

Quasi-Newton methods are a class of numerical methods that are similar to Newton's method except that the inverse of Hessian $(G(x_k))^{-1}$ is replaced by a $n \times n$ symmetric matrix H_k, which satisfies the *quasi-Newton* equation

$$H_k y_{k-1} = s_{k-1}, \tag{2.1}$$

where

$$s_{k-1} = x_k - x_{k-1} = \lambda_{k-1} d_{k-1}, y_{k-1} = g_k - g_{k-1}, \tag{2.2}$$

and $\lambda_{k-1} > 0$ is a step-length which satisfies some line search conditions. Assuming H_k nonsingular, we define $B_k = H_k^{-1}$. It is easy to see that the *quasi-Newton* step

$$d_k = -H_k g_k \tag{2.3}$$

is a stationary point of the following problem:

$$\min_{d\in\Re^n} \phi_k(d) = f(x_k) + d^T g_k + \frac{1}{2} d^T B_k d \tag{2.4}$$

which is an approximation to problem (1.1) near the current itarate x_k, since $\phi_k(d) \approx f(x_k + d)$ for small d. In fact, the definition of $\phi_k(\cdot)$ in (2.4) implies that

$$\phi_k(0) = f(x_k), \nabla\phi_k(0) = g(x_k), \tag{2.5}$$

and the quasi-Newton condiiton (2.1) is equivalent to

$$\nabla\phi_k(x_{k-1} - x_k) = g(x_{k-1}). \tag{2.6}$$

Thus, $\phi_k(x - x_k)$ is a quadratic interpolation of $f(x)$ at x_k and x_{k-1}, satisfying conditions (2.5)-(2.6). The matrix B_k (or H_k) can be updated so that the quasi-Newton equation is satisfied.

One well known update formula is the BFGS formula which updates B_{k+1} from B_k, s_k and y_k in the following way:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k}. \tag{2.7}$$

In Yuan [12], approximate function $\phi_k(d)$ in (2.4) is required to satisfy the interpolation condition

$$\phi_k(x_{k-1} - x_k) = f(x_{k-1}) \tag{2.8}$$

instead of (2.6). This change was inspired from the fact that for one-dimensional problem, using (2.8) gives a slightly faster local convergence if we assume $\lambda_k = 1$ for all k. Equation (2.8) can be rewritten as

$$s_{k-1}^T B_k s_{k-1} = 2\left[f(x_{k-1}) - f(x_k) + s_{k-1}^T g_k\right]. \tag{2.9}$$

In order to satisfy (2.9), the BFGS formula is modified as follows:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + t_k \frac{y_k y_k^T}{s_k^T y_k}, \tag{2.10}$$

where

$$t_k = \frac{2}{s_k^T y_k}\left[f(x_k) - f(x_{k+1}) + s_k^T g_{k+1}\right]. \tag{2.11}$$

If H_{k+1} is the inverse of B_{k+1}, then

$$H_{k+1} = H_k + \frac{1}{s_k^T y_k}\left[\left(\alpha_k + \frac{y_k^T H_k y_k}{s_k^T y_k}\right) s_k s_k^T - s_k y_k^T H_k - H_k y_k s_k^T\right] \tag{2.12}$$

with

$$\alpha_k = \frac{1}{t_k}. \tag{2.13}$$

Assume that B_k is positive definite and that $s_k^T y_k > 0$, B_{k+1} defined by (2.10) is positive definite if and only if $t_k > 0$. The inequality $t_k > 0$ is trivial if f is strictly convex, and it is also true if the step-length λ_k is chosen by an exact line search, which requires $s_k^T g_{k+1} = 0$. For a uniformly convex function, it can be easily shown that there exists a constant $\delta > 0$ such that $t_k \in [\delta, 2]$ for all k, and consequently global convergence proof of the BFGS method for convex functions with inexact line searches, which was given by Powell [10]. However, for a general nonlinear function f, inexact line searches do not imply the positivity of t_k, hence Yuan [12] truncated t_k to the interval $[0.01, 100]$, and showed that the global convergence of the modified BFGS algorithm is preserved for convex functions. If the objective function f is cubic along the line segment between x_{k-1} and x_k then we have the following relation

$$s_{k-1}^T G(x_k) s_{k-1} = 4 s_{k-1}^T g_k + 2 s_{k-1}^T g_{k+1} - 6 \left[f(x_{k-1}) - f(x_k) \right], \tag{2.14}$$

by considering the Hermit interpolation on the line between x_{k-1} and x_k. Hence it is reasonable to require that the new approximate Hessian satisfies condition

$$s_{k-1}^T B_k s_{k-1} = 4 s_{k-1}^T g_k + 2 s_{k-1}^T g_{k+1} - 6 \left[f(x_{k-1}) - f(x_k) \right] \tag{2.15}$$

instead of (2.11). Biggs [1], [2] gives the update of (2.12) with the value t_k chosen so that (2.15) holds. The respected value of t_k is given by

$$t_k = \frac{6}{s_k^T y_k} \left[f(x_k) - f(x_{k+1}) + s_k^T g_{k+1} \right] - 2. \tag{2.16}$$

For one-dimensional problems, Wang and Yuan [11] showed that (2.10) with (2.16) and without line searches (that is $\lambda_k = 1$ for all k) implies $R-$quadratic convergence, and except some special cases (2.10) with (2.16) also give $Q-$convergence. It is well known that the convergence rate of secant method is $(1 + \sqrt{5})/2$ which is approximately 1.618 and less than 2.

3. Limited Memory BFGS Method

The limited memory BFGS method is described by Nocedal [8], where it is called the *SQN method*. The user specifies the number m of BFGS corrections that are to be kept, and provides a sparse symmetric and positive definite matrix H_0, which approximates the inverse Hessian of f. During the first m iterations the method is identical to the BFGS method. For $k > m$, H_k is obtained by applying m BFGS updates to H_0 using information from the m previous iterations. The method uses the inverse BFGS formula in the form

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T, \tag{3.1}$$

where

$$\rho_k = 1/y_k^T s_k, \; V_k = I - \rho_k y_k s_k^T. \tag{3.2}$$

(see Dennis and Schnabel [4].)

Algorithm 3.1. L-BFGS method

(1) Choose x_0, $0 < \beta' < 1/2$, $\beta' < \beta < 1$, and initial matrix $H_0 = I$. Set $k = 0$.

(2) Compute
$$d_k = -H_k g_k$$
and
$$x_{k+1} = x_k + \lambda_k d_k$$
where λ_k satisfies
$$f(x_k + \lambda_k d_k) \leq f(x_k) + \beta' \lambda_k g_k^T d_k, \tag{3.3}$$
$$g(x_k + \lambda_k d_k)^T d_k \geq \beta g_k^T d_k \tag{3.4}$$
(the step-length $\lambda = 1$ is tried first).

(3) Let $\hat{m} = \min\{k, m-1\}$. Update H_0 for $\hat{m}+1$ times by using the pairs $\{y_j, s_j\}_{j=k-\hat{m}}^{k}$, i.e. let
$$\begin{aligned} H_{k+1} &= (V_k^T \dots V_{k-\hat{m}}^T) H_0 (V_{k-\hat{m}} \dots V_k) \\ &+ \rho_{k-\hat{m}} (V_k^T \dots V_{k-\hat{m}+1}^T) s_{k-\hat{m}} s_{k-\hat{m}}^T (V_{k-\hat{m}+1} \dots V_k) \\ &+ \rho_{k-\hat{m}+1} (V_k^T \dots V_{k-\hat{m}+2}^T) s_{k-\hat{m}+1} s_{k-\hat{m}+1}^T (V_{k-\hat{m}+2} \dots V_k) \\ &\vdots \\ &+ \rho_k s_k s_k^T \end{aligned} \tag{3.5}$$

(4) Set $k := k+1$ and go to Step 2.

4. Modified L-BFGS Algorithms

In practice, it is noted that the performances of Biggs and Yuan's updates are better than the original BFGS update for normal quasi-Newton methods (see, for instance, Phua and Setiono [9] for the performance of Biggs' update and Yuan [12] for Yuan's update).

To improve the performance of the L-BFGS algorithm, one possibility is to use the Biggs and Yuan's updates instead of the BFGS update. We note that the modified BFGS update (2.12) can also be expressed in the form
$$H_{k+1} = V_k^T H_k V_k + \alpha_k \rho_k s_k s_k^T. \tag{4.1}$$
Therefore, the major steps of the new limited memory methods are similar to the L-BFGS algorithm, except that (3.5) in Step 3 of the L-BFGS algorithm will be replaced by the following formula:
$$\begin{aligned} H_{k+1} &= (V_k^T \dots V_{k-\hat{m}}^T) H_0 (V_{k-\hat{m}} \dots V_k) \\ &+ \alpha_{k-\hat{m}} \rho_{k-\hat{m}} (V_k^T \dots V_{k-\hat{m}+1}^T) s_{k-\hat{m}} s_{k-\hat{m}}^T (V_{k-\hat{m}+1} \dots V_k) \\ &+ \alpha_{k-\hat{m}+1} \rho_{k-\hat{m}+1} (V_k^T \dots V_{k-\hat{m}+2}^T) s_{k-\hat{m}+1} s_{k-\hat{m}+1}^T (V_{k-\hat{m}+2} \dots V_k) \\ &\vdots \\ &+ \alpha_k \rho_k s_k s_k^T. \end{aligned} \tag{4.2}$$

However, for the following reasons we do not use the above mentioned formula for our limited memory scheme:

(1) In order to use (4.2) replacing (3.5), we need to calculate and store $\alpha_{k-\hat{m}}, \alpha_{k-\hat{m}+1}, \ldots, \alpha_k$, which will require an additional of $\hat{m}$=min$\{k, m-1\}$ storage. Resource in storage is the most crucial factor to determine the successfulness of a method when applied to large-scale problems.

(2) So far, the convergence rate of the modified BFGS updates in a limited memory scheme is not established. Convergence analysis of these updates was only given for standard quasi-Newton under certain conditions. On the other hand, Liu and Nocedal [6] established the convergence of the L-BFGS method.

(3) For $n-$dimensional problems, Dennis and Moré [3] showed that the standard BFGS method converged $Q-$superlinearly using an inexact line search if the current iterate x_k is sufficiently close to x^*. Therefore, the BFGS update should be preferred if x_k happened to be so.

For reasons that we have discussed,we shall not use the fully modified BFGS update (4.2) to replace (3.5). Instead a partially modified BFGS update will be applied in Step 3 of the L-BFGS algorithm. The users will specify the number $\tilde{m} < \min\{k, m-1\}$ of modified BFGS corrections that are to be used. During the first $\tilde{m}$ iterations the method is identical to the modified BFGS method and H_k is obtained by applying $\tilde{m}$ modified BFGS update to H_0. After $\tilde{m}$ iteration, BFGS update will be used instead of the modified BFGS update, the method is then identical to the L-BFGS method proposed by Nocedal [8]. We shall described the above steps in details:

Step 3. Given an integer $\tilde{m} < \min\{k, m-1\}$ and Let $\hat{m} = \min\{k, m-1\}$. Update H_0 for $\hat{m}+1$ times by using the pairs $\{y_j, s_j\}_{j=0}^{\tilde{m}}$, i.e. let

$$\begin{aligned}
H_{k+1} \quad = \quad & (V_k^T \ldots V_{k-\hat{m}}^T) H_0 (V_{k-\hat{m}} \ldots V_k) \\
+ \quad & \alpha_{k-\hat{m}} \rho_{k-\hat{m}} (V_k^T \ldots V_{k-\hat{m}+1}^T) s_{k-\hat{m}} s_{k-\hat{m}}^T (V_{k-\hat{m}+1} \ldots V_k) \\
+ \quad & \alpha_{k-\hat{m}} \rho_{k-\hat{m}+1} (V_k^T \ldots V_{k-\hat{m}+2}^T) s_{k-\hat{m}+1} s_{k-\hat{m}+1}^T (V_{k-\hat{m}+2} \ldots V_k) \\
& \vdots \\
+ \quad & \alpha_{k-\hat{m}+\tilde{m}} \rho_{k-\hat{m}+\tilde{m}} (V_k^T \ldots V_{k-\hat{m}+\tilde{m}+1}^T) s_{k-\hat{m}+\tilde{m}} \\
& \qquad\qquad\qquad \times\ s_{k-\hat{m}+\tilde{m}}^T (V_{k-\hat{m}+\tilde{m}+1} \ldots V_k) \\
+ \quad & \rho_{k-\hat{m}+\tilde{m}+1} (V_k^T \ldots V_{k-\hat{m}+\tilde{m}+2}^T) s_{k-\hat{m}+\tilde{m}+1} \\
& \qquad\qquad\qquad \times\ s_{k-\hat{m}+\tilde{m}+1}^T (V_{k-\hat{m}+\tilde{m}+2} \ldots V_k) \\
& \vdots \\
+ \quad & \rho_k s_k s_k^T
\end{aligned} \tag{4.3}$$

with $\alpha_k = 0.01$ if $\alpha_k \leq 0.01$, and $\alpha_k = 100$ if $\alpha_k \geq 100$.

5. Computational Results

In this section, we present and discuss some numerical experiments that were conducted in order to test the performance of limited memory quasi-Newton methods for unconstrained optimization using the modified BFGS formulae againt those using BFGS update.

The algorithms used for limited memory methods are from L-BFGS, which provides the line search strategy for calculating global step. The line search is based on backtracking, using quadratic and cubic modeling of $f(x)$ in the direction of search.

Five test functions, with variable dimensions, have been chosen from literature of optimization. The description of these test problems can be found, for instance, in Moré et al. [7]. Each function is tested with three different dimensions, namely $n = 8, 200$ and 1000. All test functions are tested with a single standard starting point.

All algorithms are implemented in Fortran77. The runs were performed with a double precision arithmetic, for which the unit roundoff is approximately 10^{-16}. In all cases, convergence is assumed if

$$\| g_k \| < 10^{-5} max\{1, \| x_k \|\}. \tag{5.1}$$

In all tables, n_I is the number of iterations and n_f is the number of function/gradient evaluations required by that algorithm in solving each test problem. The numbers of modification made, $\tilde{m}$ is listed by the interger numbers in bracket ( ) for Biggs' and Yuan's limited memory BFGS methods. For our case, $\tilde{m}$ varies from 2 to 5.

Numerical results obtained by applying the above algorithms are given in Tables 1-3.

Comparing the performance of all these algorithms, Tables 1-3 show that in terms of the total number of iterations, limited memory algorithms using Yuan's BFGS update scored the best while Bigg's BFGS is the second best, with L-BFGS the last. In terms of the total number of function/gradient evaluations, Biggs' BFGS requires the lowest number of function/gradient calls, Yuan's BFGS is the second, with again L-BFGS the last. The improvements of limited memory Biggs' BFGS and Yuan's BFGS methods are 11% in terms of the numbers of iterations, and saving of 10.7% and 6.4% respectively, in terms of the number of functon/gradient evaluations over L-BFGS method for $m = 5$. For $m = 10$, a savings of 9.4% and 9.3% respectively, in terms of the number of function/gradient evaluations over L-BFGS method. Finally, the improvements of limited memory Biggs' BFGS

TABLE 1. Comparative results of limited memory Biggs' BFGS, Yuan's BFGS and L-BFGS methods with $m = 5$

Test Problems	Biggs' n_I	BFGS n_f	Yuan's n_I	BFGS n_f	L-BFGS n_I	n_f
Trigonometric						
$n = 8$	24(3)	28	22(3)	29	24	31
$n = 200$	39(3)	46	46(4)	52	40	45
$n = 1000$	46(5)	52	46(5)	54	48	45
Rosenbrook						
$n = 8$	35(3)	47	32(3)	41	38	49
$n = 200$	34(3)	46	34(2)	46	36	45
$n = 1000$	34(3)	44	35(3)	51	37	48
Powell						
$n = 8$	31(3)	35	35(3)	38	46	54
$n = 200$	42(3)	48	34(4)	43	37	46
$n = 1000$	37(3)	40	33(3)	56	67	78
Beale						
$n = 8$	14(3)	16	14(3)	16	15	17
$n = 200$	14(3)	16	14(3)	17	15	16
$n = 1000$	14(2)	17	14(3)	19	15	16
Wood						
$n = 8$	83(3)	103	90(4)	118	91	118
$n = 200$	81(3)	104	84(2)	108	91	121
$n = 1000$	94(3)	123	88(3)	114	99	128
Total	622	765	621	802	699	857

and Yuan's BFGS methods over L-BFGS method are 5.4% and 7.4% respectively, in terms of the number of function/gradient evaluations. On the other hand, the increments of memory requirement is only of maximum 5 units ($\tilde{m} = 5$), which is less than 0.5% for $n = 200$ or 1000.

6. Analysis of Convergence

Consider methods with an update of the form

$$H_{k+1} = \delta_k P_k^T H_0 Q_k + \sum_{i=1}^{k} w_{ik} z_{ik}^T. \tag{6.1}$$

Here,

(1) H_0 is an $n \times n$ symmetric positive definite matrix that remains constant for all k, and δ_k is a nonzero scalar that can be thought of as an iterative rescaling of H_0;

TABLE 2. Comparative results of limited memory Biggs' BFGS, Yuan's BFGS and L-BFGS methods with $m = 10$

Test Problems	Biggs' BFGS n_I	n_f	Yuan's BFGS n_I	n_f	L-BFGS n_I	n_f
Trigonometric						
$n = 8$	21(3)	25	22(3)	27	24	29
$n = 200$	37(3)	42	45(5)	52	48	55
$n = 1000$	45(3)	52	44(3)	52	50	61
Rosenbrook						
$n = 8$	33(3)	42	32(3)	40	37	44
$n = 200$	34(3)	48	35(3)	49	37	49
$n = 1000$	34(3)	41	35(3)	47	35	44
Powell						
$n = 8$	32(3)	37	35(3)	39	38	43
$n = 200$	43(2)	48	34(4)	46	31	33
$n = 1000$	39(3)	45	33(3)	39	51	58
Beale						
$n = 8$	14(3)	16	14(3)	16	14	16
$n = 200$	13(3)	15	14(3)	17	16	17
$n = 1000$	15(5)	17	14(3)	19	15	16
Wood						
$n = 8$	80(3)	99	79(3)	95	89	118
$n = 200$	86(3)	108	81(3)	101	89	118
$n = 1000$	73(3)	91	82(3)	98	86	117
Total	598	726	599	737	690	818

(2) P_k is an $n \times n$ matrix that is the product of projection matrices of the form

$$I - \frac{uv^T}{u^T v}, \tag{6.2}$$

where $u \in span\{y_0, \ldots, y_k\}$ and $v \in span\{s_0, \ldots, s_{k+1}\}$[1], and Q_k is an $n \times n$ matrix that is the product of projection matrices of the same form where u is any $n-$vector and $v \in span\{s_0, \ldots, s_k\}$;

(3) m_k is a nonnegative integer, w_{ik} $(i = 1, \ldots, k)$ is any $n-$vector, and z_{ik} $(i = 1, \ldots, k)$ is any vector in $span\{s_0, \ldots, s_k\}$.

This form of updates is discussed by Kolda et al. [5], and is referred as the *general form*. The general form fits many known methods, including the L-BFGS method and the proposed limited memory modified BFGS. The limited memory modified update with limited memory constant m can be

[1] The vector s_{k+1} has not yet been explicitly calculated but is needed here only for the theoretical framework, not for the computational algorithms.

TABLE 3. Comparative results of limited memory Biggs' BFGS, Yuan's BFGS and L-BFGS methods with $m = 30$

Test Problems	Biggs' BFGS n_I	n_f	Yuan's BFGS n_I	n_f	L-BFGS n_I	n_f
Trigonometric						
$n = 8$	20(3)	25	21(3)	26	23	28
$n = 200$	43(3)	52	43(5)	49	46	55
$n = 1000$	44(5)	52	43(5)	51	45	53
Rosenbrook						
$n = 8$	32(3)	42	32(3)	40	37	44
$n = 200$	34(3)	48	34(2)	45	37	49
$n = 1000$	34(3)	41	34(2)	45	35	44
Powell						
$n = 8$	31(3)	33	32(3)	33	36	39
$n = 200$	43(3)	46	30(3)	31	45	46
$n = 1000$	37(5)	46	36(2)	41	36	41
Beale						
$n = 8$	14(3)	16	14(3)	16	14	16
$n = 200$	13(3)	15	14(3)	17	16	17
$n = 1000$	14(5)	17	14(3)	19	15	16
Wood						
$n = 8$	86(5)	99	82(5)	98	87	113
$n = 200$	85(5)	98	85(5)	106	87	115
$n = 1000$	79(3)	95	82(3)	102	85	114
Total	609	724	596	719	644	780

witten as

$$H_{k+1} = V^T_{k-m_k+1,k} H_0 V_{k-m_k+1,k} + \sum_{i=k-m_k+1}^{k} V^T_{i+1,k} \frac{s_i^T s_i}{t_i s_i^T y_i} V_{i+1,k} \tag{6.3}$$

where $m_k = \min\{k+1, m\}$.

The above method fits the general form (6.1) if at iteration k we choose

$$\delta_k = 1(\text{without scaling}), m_k = \min\{k+1, m\},$$

$$P_k = Q_k = V^T_{k-m_k+1,k}, \text{ and}$$

$$w_{ik} = z_{ik} = \frac{V^T_{k-m_k+i+1,k} s_{k-m_k+i}}{\sqrt{t_{k-m_k+i} s^T_{k-m_k+i} y_{k-m_k+i}}}.$$

We show that methods fitting the general form (6.1) produce conjugate search directions (see Theorem 6.1) and terminate in n iterations (see Corollary 6.2) *if and only if* P_k maps the vector y_0 through y_k into span $\{y_0, \ldots, y_{k-1}\}$ for each $k = 1, 2, \ldots, n$.

Theorem 6.1. *Suppose that we apply a quasi-Newton (QN) method with an update of the form (2.11) to minimize an n-dimensional strictly convex quadratic function. Then for each k before ternmination (i.e., $g_{k+1} \neq 0$),*

$$g_{k+1}^T s_j = 0, \text{ for all } j = 0, 1, \ldots, k,$$

$$s_{k+1}^T A s_j = 0, \text{ for all } j = 0, 1, \ldots, k, \text{ and}$$

$$span\{s_0, \ldots, s_{k+1}\} = span\{H_0 g_0, \ldots, H_0 g_{k+1}\},$$

if and only if

$$P_j y_i \in span\{y_0, \ldots, y_{k+1}\} \text{ for all } i = 0, 1, \ldots, k, j = 0, 1, \ldots, k. \tag{6.4}$$

Proof. See Theorem 2.2 in Kolda et al. [5]. □

When a method produces conjugate search directions, we can say something about termination.

Corollary 6.2. *Suppose we have a method of the type described in Theorem 6.1 satisfying (2.11). Suppose further that $H_k g_k \neq 0$ whenever $g_k \neq 0$. Then the scheme reproduces the iterates from the conjugate gradient method with preconditioner H_0 and terminates in no more than n iteration.*

Proof. See Corollary 2.3 in Kolda et al. [5]. □

Note that we require that $H_k g_k$ be nonzero whenever g_k is nonzero; this requirement is equivalent to positive definite updates and is necessary since not all methods produce positive definite updates. It is possible to construct an update that maps g_k to zero even it is not positive definite. If this were to happen, we would have a breakdown in the method. In our case, positive definiteness is trivial if $t_k > 0$.

The next corollary gives some ideas for methods that relate to L-BFGS and also will terminate in at most n iterations on strictly convex quadratics.

Corollary 6.3. *The L-BFGS method with exact line search will terminate in n iteration on an n−dimensional strictly convex quadratic function even if any of the following modification is made to the update:*

(1) *Every BFGS updates in (3.5) are replaced by (4.1).*
(2) *Any m BFGS updates before the method is restarted are replaced by (4.1).*
(3) *Replacing any $\hat{m} \leq min\{m, k+1\}$ BFGS updates by (4.1) before the method is restarted.*

Proof. For each variant, we show that the method fits the general form in (6.1), satisfies condition (6.4) of Theorem 6.1, and hence terminates by Corollary 6.2.

(1) Let $m > 0$ be any integer, which is preset by users or may change from iteration, and define

$$V_{ik} = \prod_{j=i}^{k}(I - \frac{y_j s_j^T}{s_j^T y_j}).$$

Choose

$$\delta_k = 1, m_k = \min\{k+1, m\},$$

$$P_k = Q_k = V^T_{k-m_k+1,k}, \text{ and}$$

$$w_{ik} = z_{ik} = \frac{V^T_{k-m_k+i+1,k} s_{k-m_k+i}}{\sqrt{t_{k-m_k+i} s^T_{k-m_k+i} y_{k-m_k+i}}} \text{with all } t_k > 0.$$

These choices clearly fit the general form. Furthermore,

$$P_k y_i = \begin{cases} 0, & \text{if } j = k-m_k, k-m_k+1, \dots, k \\ y_j, & \text{if } j = 0, 1, \dots, k-m_k-1 \end{cases},$$

so this variation satisfies condition (6.1) of Theorem 6.1.

(2) This is a special case of the first variant. Note that the BFGS update is equals to $t_k = 1$.

(3) This is a special case of the second variant with the choices:

$$\delta_k = 1, \hat{m} = \min\{k+1, m_k\},$$

$$P_k = Q_k = V^T_{k-\hat{m}_k+1,k}, \text{ and}$$

$$w_{ik} = z_{ik} = \frac{V^T_{k-\hat{m}_k+i+1,k} s_{k-\hat{m}_k+i}}{\sqrt{t_{k-\hat{m}_k+i} s^T_{k-\hat{m}_k+i} y_{k-\hat{m}_k+i}}} \text{with all } t_k > 0.$$

These choices will also fit the general form. Moreover,

$$P_k y_i = \begin{cases} 0, & \text{if } j = k-\hat{m}_k, k-\hat{m}_k+1, \dots, k \\ y_j, & \text{if } j = 0, 1, \dots, k-\hat{m}_k-1 \end{cases}.$$

After $\hat{m}$ iterations, choose $\hat{m}$ =min$\{k+1, m\}$ and $t_i = 1$ for $i = \hat{m}, \hat{m}+1, \dots, m_k$.

□

Note that the considered cases in Corollary 6.3 cover both limited memory methods with full of partially modified BFGS updates.

7. Conclusions

We have attempted, in this paper, to develop two numerical methods for large-scale unconstrained optimization that are based on different technique of approximating the objective function. We applied both the Biggs' and Yuan's BFGS updates partially in the limited memory scheme replacing the standard BFGS update.

We tested these methods on a set of standard test functions from Moré et al. [7]. Our test results show that on the set of problems we tried, our partially modified L-BFGS methods require fewer iterations and function/gradient evaluations than L-BFGS by Nocedal [8]. Numerical tests also suggest that these partially modified L-BFGS methods are more superior than the standard L-BFGS method.

Thus for large problems where space limitations do not preclude using the full quasi-Newton updates, these methods are recommended.

References

[1] Biggs, M.C. *Minimization algorithms making use of non-quadratic properties of the objective function.* Journal of the Institute of Mathematics and Its Applications, 8:315-327, 1971.

[2] Biggs, M.C. *A note on minimization algorithms making use of non-quadratic properties of the objective function.* Journal of the Institute of Mathematics and Its Applications, 12:337-338, 1973.

[3] Dennis, J.E.Jr. and Moré, J.J. *Quasi-Newton method, motivation and theory.* SIAM Review, 19:46-89, 1977.

[4] Dennis, J.E.Jr. and Schnabel, R.B. *Numerical Methods for Nonlinear Equations and Unconstrained Optimization.* Prentice-Hall, New Jersey, 1983.

[5] Kolda, T.G., Dianne, P.O. and Nazareth, L. *BFGS with update skipping and varying memory.* SIAM Journal on Optimization, 8/4: 1060-1083, 1998.

[6] Liu, D.C. and Nocedal, J. *On the limited memory BFGS method for large-scale optimization.* Mathematical Programming, 45:503-528, 1989.

[7] Moré, J.J., Garbow, B.S. and Hillstrom, K.E. *Testing unconstrained optimization software.* ACM Transactions on Mathematical Software, 7:17-41, 1981.

[8] Nocedal, J. *Updating quasi-Newton matrices with limited storage.* Mathematics of Computation, 35:773-782, 1980.

[9] Phua, P.K.H. and Setiono, R. *Combined quasi-Newton updates for unconstrained optimization.* National University of Singapore, Dept. of Information Systems and Computer Science, TR 41/92, 1992.

[10] Powell, M.J.D. *Some global convergence properties of a variable metric algorithm for minimization without exact line searches*, in: R.W. Cottle and C.E. Lemke, eds., Nonlinear Programming SIAM-AMS Proceeding, SIAM Publications, 9:53-72, 1976.

[11] Wang, H.J. and Yuan, Y. *A quadratic convergence method for one-dimensional optimization.* Chinese Journal of Operations Research, 11:1-10, 1992.

[12] Yuan, Y. *A modified BFGS algorithm for unconstrained optimization.* IMA Journal Numerical Analysis, 11:325-332, 1991.

[13] Nash, S.G. and Nocedal, J. *A numerical study of a limited memory BFGS method and the truncated Newton method for large-scale optimization*. SIAM J. Optimization, 1:358-372, 1991.

Leong Wah June
Department of Mathematics
Faculty of Science
University Putra Malaysia
Serdang, 43400 Malaysia
e-mail address: leongwj@putra.upm.edu.my

Malik Abu Hassan
Department of Mathematics
Faculty of Science
University Putra Malaysia
Serdang, 43400 Malaysia
e-mail address: malik@fsas.upm.edu.my

(*Received April 26, 2005*)