Matematika, 2001, Jilid 17, bil. 1, hlm. 15–23

Limited Modified BFGS Method for Large-Scale Optimization

Malik Hj. Abu Hassan, Mansor B. Monsi & Leong Wah June
Department of Mathematics
Faculty Science and Environmental Studies
Universiti Putra Malaysia
43400 Serdang, Selangor

Abstract In this paper, a limited modified BFGS method for solving large-scale unconstrained optimization problems is proposed. The proposed algorithm generates quasi-Newton directions using a modified BFGS method suggested by Biggs (1973). The modified BFGS method is then extended to the limited memory version. In order to use only minimum storage for the modification, the modification is only applied to the last BFGS corrections. Numerical results indicate that an improvement is achieved.

Keywords Limited modified BFGS method, large-scale unconstrained optimization.

Abstrak *Dalam kertas ini, suatu kaedah BFGS terubahsuai terhad untuk masalah pengoptimuman tak berkekangan yang berskala besar telah dicadangkan. Algoritma yang dicadangkan itu menjanakan arah kuasi-Newton dengan menggunakan kaedah BFGS terubahsuai oleh Biggs (1973). Kaedah BFGS terubahsuai tersebut kemudian dilanjutkan kepada versi ingatan terhad. Pengubahsuaian tersebut hanya dilakukan kepada pembetulan BFGS yang terakhir supaya hanya ruang storan yang minimum bagi pengubahsuaian diperlukan. Keputusan berangka menunjukkan kemajuan telah dicapai.*

Katakunci *Kaedah BFGS terubahsuai terhad, pengoptimuman tak berkekangan berskala besar.*

1 Introduction

Quasi-Newton (QN) methods for unconstrained optimization are a class of numerical techniques for solving the following problem

$$\min_x f(x) \tag{1.1}$$

where $f(x)$ is a nonlinear real-valued function and x is an n-dimensional real vector. At the kth iteration, an approximation point x_k and an $n \times n$ matrix H_k are available. The methods proceed by generating a sequence of approximation points via the equation

$$x_{k+1} = x_k + \lambda_k d_k \tag{1.2}$$

where λ_k is calculated to satisfy certain line search conditions and d_k is a descent direction.

One important feature of QN methods is the choice of the matrix H_k. The methods require H_k to be positive definite and satisfy the QN equation

$$H_{k+1} y_k = \alpha_k s_k, \quad \alpha_k > 0 \tag{1.3}$$

where $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$ with g the gradient of f. One of the best known QN methods is the BFGS method that was proposed independently by Broyden [2], Fletcher [4], Goldfarb [6] and Shanno [9]. The BFGS update is defined by the equation

$$H_{k+1} = H_k + \frac{1}{s_k^T y_k}\left((1 + \frac{y_k^T H_k y_k}{s_k^T y_k}) s_k s_k^T - s_k y_k^T H_k - H_k y_k s_k^T\right). \tag{1.4}$$

with $\alpha_k = 1$.

The BFGS update has been used successfully in many production codes for solving unconstrained optimization problems. In practice, we observe from numerical results of many other papers (see Luksan [7] for instance) that the BFGS out-performed many QN updates in solving practical problems.

Our interest here is the limited memory extension to the QN methods, which will suit for the solution of large-scale optimization problems. Limited memory QN methods has been considered by Nocedal [8], where it is called the SQN method. The user specifies the number m of QN (BFGS, for instance) 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 QN method. For $k > m$, H_k is obtained by applying m QN updates to H_0 using information from the m previous iterations. The limited memory BFGS method (L-BFGS) by Nocedal uses the inverse BFGS formula in the form

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T, \tag{1.5}$$

where

$$\rho_k = 1/y_k^T s_k, \quad V_k = I - \rho_k y_k s_k^T. \tag{1.6}$$

(see Dennis and Schnabel [3].)

In this paper, we try to improve the performance of the BFGS and apply the modified BFGS update to large-scale optimization problems. A description of the technique is presented in Section 2. Section 3 presents implementation of the modified BFGS method to limited memory procedure. Numerical results also included in Section 4. For comparison purposes, numerical results obtained by using L-BFGS method is also given. Discussions

and conclusions are given in Section 5.

2 A modified BFGS algorithm

To improve the performance of the BFGS updates, Biggs [1] first suggested a self-adjustable value for the parameter α_k . Based upon non-quadratic models, he derived the parameter, α_k as

$$\alpha_k = \frac{1}{t_k} \tag{2.1}$$

where

$$t_k = \frac{6}{s_k^T y_k}\left(f(x_k) - f(x_{k+1}) + s_k^T g_{k+1}\right) - 2. \tag{2.2}$$

Hence, a modified BFGS update can be defined by

$$H_{k+1} = H_k + \frac{1}{s_k^T y_k}\left(\left(\alpha_k + \frac{y_k^T H_k y_k}{s_k^T y_k}\right) s_k s_k^T - s_k y_k^T H_k - H_k y_k s_k^T\right). \tag{2.3}$$

A more useful form of (2.3) when apply to large-scale problems can be written as follows:

$$H_{k+1} = V_k^T H_k V_k + \alpha_k \rho_k s_k s_k^T. \tag{2.4}$$

Therefore, given any initial approximate inverse Hessian H_0, a recursive formula for (2.4) at any iteration k can be expressed as follows:

$$\begin{aligned} H_{k+1} = &\left(V_k^T \cdots V_0^T\right) H_0 \left(V_0 \cdots V_k\right) \\ &+\alpha_0\rho_0 \left(V_k^T \cdots V_1^T\right) s_0 s_0^T \left(V_1 \cdots V_k\right) \\ &+\alpha_1\rho_1 \left(V_k^T \cdots V_2^T\right) s_1 s_1^T \left(V_2 \cdots V_k\right) \\ &\quad\vdots \\ &+\alpha_k\rho_k s_k s_k^T. \end{aligned} \tag{2.5}$$

3 Limited modified BFGS method

Based upon the recursive formula (2.5) and limited memory updating procedures developed by Nocedal [8], we can now state a limited memory modified BFGS algorithm with inexact line searches as follows.

Algorithm 3.1. LmBFGS method

Step 1. Choose x_0, $0 < \beta' < \frac{1}{2}$, $\beta' < \beta < 1$,and initial matrix $H_0 = I$. Set $k = 0$.

Step 2. Compute

$$d_k = -H_k g_k$$

and

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

where λ_k satisfies

$$f(x_k + \lambda_k d_k) \leq f(x_k) + \beta' \lambda_k g_k^T d_k, \tag{3.1}$$

$$g(x_k + \lambda_k d_k)^T d_k \geq \beta g_k^T d_k \tag{3.2}$$

(the steplength $\lambda = 1$ is tried first).

Step 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

$$\begin{aligned} H_{k+1} = & \left(V_k^T \cdots V_{k-\hat{m}}^T\right) H_0 \left(V_{k-\hat{m}} \cdots V_k\right) \\ & + \alpha_{k-\hat{m}} \rho_{k-\hat{m}} \left(V_k^T \cdots V_{k-\hat{m}+1}^T\right) s_{k-\hat{m}} s_{k-\hat{m}}^T \left(V_1 \cdots V_k\right) \\ & + \alpha_{k-\hat{m}+1} \rho_{k-\hat{m}+1} \left(V_k^T \cdots V_{k-\hat{m}+2}^T\right) s_{k-\hat{m}+1} s_{k-\hat{m}+1}^T \left(V_{k-\hat{m}+2} \cdots V_k\right) \\ & \quad \vdots \\ & + \alpha_k \rho_k s_k s_k^T. \end{aligned} \tag{3.3}$$

with α_k calculated by (2.1)-(2.2), and

if $\alpha_k \leq 0.01$ then set $\alpha_k = 0.01$,

if $\alpha_k \geq 100$ then set $\alpha_k = 100$.

However, in order to calculate the approximation inverse Hessian, H using (3.3), we need additional $\hat{m}$ storage for α . We try to avoid this by introducing new updating formula in Step 3 of Algorithm 3.1 as follows:

$$\begin{aligned} H_{k+1} = & \left(V_k^T \cdots V_{k-\hat{m}}^T\right) H_0 \left(V_{k-\hat{m}} \cdots V_k\right) \\ & + \rho_{k-\hat{m}} \left(V_k^T \cdots V_{k-\hat{m}+1}^T\right) s_{k-\hat{m}} s_{k-\hat{m}}^T \left(V_{k-\hat{m}+1} \cdots V_k\right) \\ & + \rho_{k-\hat{m}+1} \left(V_k^T \cdots V_{k-\hat{m}+2}^T\right) s_{k-\hat{m}+1} s_{k-\hat{m}+1}^T \left(V_{k-\hat{m}+2} \cdots V_k\right) \\ & \vdots \\ & + \alpha_k \rho_k s_k s_k^T. \end{aligned} \tag{3.4}$$

By doing so, we only need to calculate α_k instead of $\alpha_{k-\hat{m}}$, $\alpha_{k-\hat{m}+1}$, $\cdots$, α_k. Therefore formula (3.4) is preferred.

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

Table 1: Comparison of LmBFGS and LBFGS with $m = 5$

Test Problems	LmBFGS($m = 5$)		LBFGS($m = 5$)	
	n_I	n_f	n_I	n_f
Penalty I				
$n = 8$	42	55	55	66
$n = 200$	61	71	60	73
$n = 1000$	62	75	64	79
Trigonometric				
$n = 8$	25	32	24	31
$n = 200$	47	53	40	45
$n = 1000$	46	54	48	45
Rosenbrook				
$n = 8$	36	44	38	49
$n = 200$	36	45	36	45
$n = 1000$	38	52	37	48
Powell				
$n = 8$	36	47	46	54
$n = 200$	36	45	37	46
$n = 1000$	45	57	67	78
Beale				
$n = 8$	13	15	15	17
$n = 200$	14	17	15	16
$n = 1000$	14	18	15	16
Wood				
$n = 8$	107	140	91	118
$n = 200$	90	120	91	121
$n = 1000$	96	126	99	128

Table 2: Comparison of LmBFGS and LBFGS with $m = 10$

Test Problems	LmBFGS($m = 10$)		LBFGS($m = 10$)	
	n_I	n_f	n_I	n_f
Penalty I				
$n = 8$	43	57	45	55
$n = 200$	61	73	58	70
$n = 1000$	60	73	63	75
Trigonometric				
$n = 8$	20	26	24	29
$n = 200$	47	55	48	55
$n = 1000$	49	56	50	61
Rosenbrook				
$n = 8$	36	43	37	44
$n = 200$	37	50	37	49
$n = 1000$	38	49	35	44
Powell				
$n = 8$	41	43	38	43
$n = 200$	43	48	31	33
$n = 1000$	43	50	51	58
Beale				
$n = 8$	14	16	14	16
$n = 200$	14	17	16	17
$n = 1000$	15	20	15	16
Wood				
$n = 8$	83	113	89	118
$n = 200$	82	113	89	118
$n = 1000$	80	110	86	117

Table 3. Comparison of LmBFGS and LBFGS with $m = 30$

Test Problems	LmBFGS($m = 30$)		LBFGS($m = 30$)	
	n_I	n_f	n_I	n_f
Penalty I				
$n = 8$	43	57	45	55
$n = 200$	61	73	58	70
$n = 1000$	60	73	63	75
Trigonometric				
$n = 8$	19	26	23	28
$n = 200$	46	55	46	55
$n = 1000$	45	57	45	53
Rosenbrook				
$n = 8$	36	44	37	44
$n = 200$	37	50	37	49
$n = 1000$	37	48	35	44
Powell				
$n = 8$	40	45	36	39
$n = 200$	38	41	45	46
$n = 1000$	40	44	36	41
Beale				
$n = 8$	14	16	14	16
$n = 200$	14	17	16	17
$n = 1000$	15	20	15	16
Wood				
$n = 8$	84	113	87	113
$n = 200$	83	114	87	115
$n = 1000$	80	100	85	114

4 Numerical Results

All routines are written in FORTRAN 77 and computational results are obtained on a Pentium II machine. The required accuracy is set as 10^{-5}. That is, convergence is assumed if the following criterion is satisfied at the point x_k

$$\|g_k\| < 10^{-5} \times \max\{1, \|x_k\|\} \tag{4.1}$$

where $\|\cdot\|$ is the l_2 (Euclidean) norm.

A total of 6 standard functions have been chosen for evaluation purposes. Several of these functions are given by Gill and Murray [5]. Each function is tested with dimensions varied from 8 to 1000 variables. For the purpose of comparison, the LBFGS methods are also evaluated over the same set of test problems with $m = 5$, 10 and 30. Numerical results obtained by LmBFGS and LBFGS algorithms are summarized in Table 1-3. In these tables n_I denotes the number of iteration and n_f denotes the number of function/gradient evaluations.

As can be seen from Table 1-3 for most of the problems especially problems with large number of variables, the number of iterations and function/gradient evaluations required by LmBFGS is less than the corresponding numbers required by LBFGS. This indicates that LmBFGS is a better choice for solving large-scale optimization problems.

5 Conclusions

We have presented a limited modified BFGS method for solving unconstrained large-scale optimization problems. The proposed algorithm generates quasi-Newton directions using a modified BFGS method suggested by Biggs [1]. This modified BFGS method is then extended to the limited memory version. In order to save storage, the modification is only applied to the last corrections. Numerical results indicate an overall improvement on the number of iteration and function/gradient evaluation.

References

[1] M.C. Biggs, *A note on minimization algorithms which make use of non-quadratic properties of the objective function*, J. Inst. Maths Applics., 12 (1973), 337-338.

[2] C.G. Broyden, *The convergence of a class of double-rank minimization algorithms*, J. Inst. Maths Applics., 6 (1970), 76-90.

[3] J.E. Dennis, Jr. & R.B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations,* Prentice-Hill Inc., New Jersey, 1983.

[4] R. Fletcher, *A new approach to variable metric algorithm*, Computer Journal,13 (1970), 392-399.

[**5**] P.E. Gill & W. Murray, *Conjugate-gradient methods for large scale nonlinear optimization*, Technical report SOL 79-15, Department of Operation Research, Stanford University, Stanford, 1979.

[**6**] D. Goldfarb, *A family of variable metric methods derived by variational means*, Mathematics of Computations, 24 (1970), 23-26.

[**7**] L. Luksan, *Computational experience with known variable metric updates*, Journal of Optimization Theory and Applications, 83 (1994), 27-47.

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

[**9**] D.F. Shanno, *Conditioning of quasi-Newton methods for function minimization*, Mathematics of Computation 24, (1970), 647-656.