Journal of Computational Mathematics, Vol.12, No.4, 1994, 380–386.

A QUASI-NEWTON ALGORITHM WITHOUT CALCULATING DERIVATIVES FOR UNCONSTRAINED OPTIMIZATION*[1)]

Sun Lin-ping
(*Department of Mathematics, Nanjing University, Jiangsu, China*)

Abstract

A new algorithm for unconstrained optimization is developed, by using the product form of the OCSSR1 update. The implementation is especially useful when gradient information is estimated by difference formulae. Preliminary tests show that new algorithm can perform well.

1. Introduction

We consider the unconstrained optimization problem

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

where $f(x) : I\!R^n \to I\!R$ is a real continuously differentiable function.

Many algorithms have been proposed for solving (1.1). Typically, given both an approximation H to $[\nabla^2 f(x)]^{-1}$ and g the gradient $\nabla f(x)$ at the current point x, a quasi-Newton algorithm starts each iteration by taking a step

$$x_+ = x - \alpha Hg, \tag{1.2}$$

where the steplenth $\alpha > 0$ is chosen so that

$$f(x) - f(x_+) \geq \sigma\alpha g^T Hg \tag{1.3a}$$

and

$$| g_+^T Hg | \leq \tau g^T Hg \tag{1.3b}$$

are satisfied, where $\sigma \in (0, 1/2)$ and $\tau \in (\sigma, 1)$; and then to form H_+, an estimate of $[\nabla^2 f(x)]^{-1}$ by using an updating formula satisfying the quasi-Newton condition

$$H_+ y = s, \tag{1.4}$$

where $s = x_+ - x$ and $y = g_+ - g$. For the SSR1 update

$$H_+ = \theta H + \frac{(s - \theta Hy)(s - \theta Hy)^T}{(s - \theta Hy)^T y}, \tag{1.5}$$

* Received August 18, 1993.
[1)] The Project Supported by National Natural Science Foundation of China.

Osborne and Sun[9] propose a new algorithm (OCSSR1) with Davidon's optimal condition, in which the scaling factor θ is taken either

$$\theta_1 = c/b - \sqrt{c^2/b^2 - c/a} \tag{1.6a}$$

or

$$\theta_1 = c/b + \sqrt{c^2/b^2 - c/a}, \tag{1.6b}$$

where $a = y^T Hy, b = s^T y$ and $c = s^T H^{-1} s$. Preliminary numerical tests show that the OCSSR1 method compares favourably with good implementations of the BFGS method[5,12,14].

Here, a new implementation of the OCSSR1 algorithm is developed by writing the expression for the SSR1 update in product form. If the derivatives are available, the implementation is equivalent to the OCSSR1 algorithm. Moreover, it is especially useful when gradient information is estimated by finite difference formulae, in this case, the algorithm can perform well. In Section 2, an expression of the SSR1 update in product form is derived. In Section 3, an algorithm using the OCSSR1 update without calculating derivatives is outlined. Numerical results are contained in Section 4.

In this paper, the following notations are used: I is an unit matrix; $\kappa(.)$ denotes the condition number of a matrix; Tr(.) is the trace of a matrix; det(.) denotes the determinant of a square matrix.

2. The SSR1 Formula in Product Form

The SSR1 update (1.5) can be written in product form. That is

$$H_+ = (I + uv^T)\theta H(I + uv^T)^T, \tag{2.1}$$

where

$$u = \mu(s - \theta Hy), \tag{2.2}$$

$$v = (H^{-1}s - \theta y)/\theta \tag{2.3}$$

and

$$\mu = \frac{-\theta \pm \sqrt{(c\theta - b\theta^2)/(b - a\theta)}}{c - 2b\theta + a\theta^2}. \tag{2.4}$$

Let $H_+ = C_+ C_+^T$ and $H = CC^T$, from (2.1) we have

$$C_+ = \sqrt{\theta}[C + u(C^T v)^T]. \tag{2.5}$$

Remark 2.1. Osborne and Sun[9] point out that if $\theta > 0$ and $s^T y > 0$, then H_+ is positive definite if and only if

$$\theta \notin [s^T y/y^T Hy, s^T H^{-1} s/s^T y]. \tag{2.6}$$

Thus, in (2.4), it always holds that $(c\theta - b\theta^2)/(b - a\theta) > 0$. In addition, $c - 2b\theta + a\theta^2 > 0$ because $ac \geq b^2$.

Remark 2.2. Product form updates are well known to be suitable for implementing quasi-Newton methods. However, in the literature mainly the BFGS updates are used. For example, see Osborne and Saunders [8], Davidon [3], Han [4], Powell [10] and Coope [2].

3. The OCSSR1 Algorithm without Calculating Derivatives

Let

$$w = C^T v, \tag{3.1}$$

then

$$u = \theta\mu C w \tag{3.2}$$

and

$$C_+ = \sqrt{\theta} C[I + \theta\mu w w^T]. \tag{3.3}$$

For the vector

$$\hat{g} = C^T g, \tag{3.4}$$

its components are directional derivatives of f(x) in the directions $\{c_j, j = 1, 2, \cdots, n\}$ which are the columns of C. Thus we can use the central difference formula

$$\hat{g}(j) = \frac{f(x + h_j c_j) - f(x - h_j c_j)}{2h_j} \tag{3.5}$$

to estimate this gradient information. Moreover, we have

$$\hat{y} = C^T y = \hat{g}_+ - \hat{g} \tag{3.6}$$

and (3.1) can be written as

$$w = -[\hat{y} + (\alpha/\theta)\hat{g}]. \tag{3.7}$$

In order to calculate θ, the following equations are obtained by using (3.4) and (3.6):

$$a = \hat{y}^T \hat{y}, \tag{3.8a}$$

$$b = -\alpha \hat{g}^T \hat{y} \tag{3.8b}$$

and

$$c = \alpha^2 \hat{g}^T \hat{g}. \tag{3.8c}$$

Based on the above discussion we can establish the implementation of the OCSSR1 algorithm without calculating derivatives.

Algorithm 3.1 (NGOCSSR1).
Step 0. Given $\varepsilon_1 > 0, \varepsilon_2 > 0$ and $x \in I\!R^n$. Set $C = I$ and $k = 0$.
Step 1. If the convergence criterion is achieved, then stop.
Step 2. Calculate the search direction d using the equation

$$d = -C\hat{g}, \tag{3.9}$$

where C satisfies the update (3.3).

Step 3. Select $\alpha > 0$ such that the criterion (1.3a) is satisfied, that is

$$f(x + \alpha d) \le f(x) - \sigma \alpha \hat{g}^T \hat{g}, \tag{3.10}$$

where $\sigma = 0.0001$ is a typical value.

Step 4. Set $x_+ = x + \alpha d$ and calculate $\hat{g}_+$ and $\hat{y}$.

Step 5. If

$$\hat{g}^T \hat{y} > -\epsilon_1 \parallel \hat{g} \parallel_2 \parallel \hat{y} \parallel_2, \tag{3.11}$$

set $C_+ = C$ and go to Step 10.

Step 6. If

$$-(\alpha\hat{g} + \hat{y})^T \hat{y} > \epsilon_1 \parallel \alpha\hat{g} + \hat{y} \parallel_2 \parallel \hat{y} \parallel_2, \tag{3.12}$$

set $\theta = 1$ and go to Step 9. Else continue to Step 7.

Step 7. If

$$\parallel C(\hat{y} + \alpha\gamma\hat{g}) \parallel_2 \le \varepsilon_2, \tag{3.13}$$

where $\gamma = a/b$, set

$$C_+ = (1/\sqrt{\gamma})C \tag{3.14}$$

and go to Step 10. Else calculate θ_1 and θ_2 by using (1.6) and (3.8).

Step 8. Determine θ.

Step 9. Update C by using (3.3).

Step 10. Set $k = k + 1$ and go to Step 1.

Remark 3.2. It is important for the OCSSR1 update to guarantee $s^T y > 0$. Here an equivalent test (3.11) is done from (3.8b).

Remark 3.3. In Algorithm OCSSR1 we use the strategy of forcing $\theta = 1$ if the inequality

$$(s - Hy)^T y > \epsilon_1 \parallel s - Hy \parallel_2 \parallel y \parallel_2$$

is satisfied. This test is implemented here in the equivalent form (3.12) using

$$(s - Hy)^T y = -(\alpha\hat{g} + \hat{y})^T \hat{y}. \tag{3.15}$$

Remark 3.4. We use (3.14) when Hy and s are linearly dependent. This is equivalent to the corresponding test for the OCSSR1 update follows from (3.15).

Remark 3.5. θ is chosen by a heuristic strategy which aims at making $\kappa(C_+C_+^T)$ smaller. We choose $\theta = \theta_1$ except when the following conditions satisfied:

$$\mathrm{Tr}(C_+C_+^T)_{\theta=\theta_1} \ge \mathrm{Tr}(C_+C_+^T)_{\theta=\theta_2}. \tag{3.16}$$

In fact, from Osborne and Sun [9] we know that

$$\det(C_+C_+^T)_{\theta=\theta_1} \le \det(C_+C_+^T)_{\theta=\theta_2},$$

which implies

$$\prod_{i=1}^{n} \lambda_i^{\theta_1} < \prod_{i=1}^{n} \lambda_i^{\theta_2},$$

where λ_i^θ are the eigenvalues of the matrix $(C_+C_+^T)_\theta$ at $\theta = \theta_j (j = 1, 2)$ with $\lambda_1^\theta \le \lambda_2^\theta \le \cdots \le \lambda_n^\theta$. On the other hand,

$$\mathrm{Tr}(C_+C_+^T)_\theta = \sum_{i=1}^{n} \lambda_i^\theta.$$

Thus, $\kappa(C_+C_+^T)_{\theta=\theta_2}$ is probably smaller than $\kappa(C_+C_+^T)_{\theta=\theta_1}$ when (3.16) holds. It is not difficult to calculate the values $Tr(C_+C_+^T)$. For the initial $C = I$, we have

$$\mathrm{Tr}(C_+C_+^T) = \theta[\mathrm{Tr}(CC_+^T) + r^T r / w^T \hat{y}], \tag{3.17}$$

where $r = Cw$.

Remark 3.6. In the update (3.3), if

$$\det(I + \theta\mu ww^T) > 0$$

is required then

$$\mu > -1/\theta w^T w. \tag{3.18}$$

Thus, in (2.4) we practically choose

$$\mu = \frac{-\theta + \sqrt{(c\theta - b\theta^2)/(b - a\theta)}}{c - 2b\theta + a\theta^2}. \tag{3.19}$$

Remark 3.7. The criterion (3.10) is a efficient weak line search strategy to guarantee termination (see Osborne [7]).

Remark 3.8. It should be stated that if we consider how to choose θ so that the condition number of the matrix $C^{-1}C_+$ becomes optimal, then the derived value of θ is the same as the formula (1.6).

4. Numerical Results

In this section, we report some numerical experiments performed by the NGOCSSR1 algorithm. The test functions are outlined as follows:

TF.1 Beale x_0=(1,1),
TF.2 Brown Badly Scaled $x_0 = (1, 1)$,
TF.3 Brown and Dennis $x_0 = (25, 5, -5, -1)$,
TF.4 Broyden Tridiagonal $x_0 = (-1, \cdots, -1)$,
TF.5 Modified Cragg $x_0 = (1, 2, 2, 2)$,
TF.6 Dixon $x_0 = (-2, \cdots, -2)$,
TF.7 Extended Powell $x_0 = (3, -1, 0, 1, \cdots, 3, -1, 0, 1)$,
TF.8 Helical $x_0 = (-1, 0, 0)$,
TF.9 Hilbert $x_0 = (-4, -2, -1.333, -1)$,
TF.10 Powell $x_0 = (1, 1, 1, 1)$,
TF.11 Penalty $x_0 = (1, 2, \cdots, n)$,
TF.12 Rosenbrock $x_0 = (-1.2, 1)$,
TF.13 Tridia $x_0 = (1, \cdots, 1)$,

TF.14 Trigonometric $x_0 = \left(\frac{1}{n}, \cdots, \frac{1}{n}\right)$

TF.15 Variably Dimensioned $x_0 = (1 - \frac{1}{n}, \cdots, 1 - \frac{i}{n}, \cdots, 0)$

TF.16 Wood $x_0 = (-3,-1,-3,-1)$

where TF.1, TF.2, TF.3, TF.4, TF.5, TF.7, TF.8, TF.10, TF.12, TF.14, TF.15 and TF.16 appear in More, Carbow and Hillstrom [6]; TF.6 appears in Wolfe [15]; Tf.9 appears in Schittkowski [11]; TF.11 and TF.13 appears in Shanno [13] .

Table (Numerical Tests for NGOCSSR1)

Function	n	N_t	N_f	f^*	f_t
TF.1	2	14	81	0	0.339×10^{-10}
TF.2	2	10	58	0	0.575×10^{-14}
TF.3	4	18	209	85822.2	0.858222×10^{5}
TF.4	10	39	845	0	0.158×10^{-11}
TF.5	4	49	455	0	0.726×10^{-10}
TF.6	10	45	971	0	0.353×10^{-10}
TF.7	4	41	387	0	0.543×10^{-10}
	32	46	3062	0	0.588×10^{-10}
	64	48	6327	0	0.680×10^{-10}
TF.8	3	42	314	0	0.408×10^{-11}
TF.9	4	4	47	0	0.598×10^{-11}
TF.10	20	35	1480	0	0.699×10^{-10}
	50	7	819	0	0.183×10^{-13}
TF.11	4	70	653	0.000022499	$0.22499799 \times 10^{-4}$
	10	189	4231	0.000070876	$0.70876530 \times 10^{-4}$
TF.12	2	22	124	0	0.148×10^{-10}
TF.13	10	11	255	0	0.188×10^{-11}
	50	49	5053	0	0.647×10^{-18}
TF.14	5	31	355	0	0.845×10^{-11}
TF.15	20	16	896	0	0.210×10^{-12}
	50	15	1871	0	0.210×10^{-12}
TF.16	4	37	354	0	0.266×10^{-11}

All tests used double precision on an IBM PC/AT alone. The corresponding machine precision is of the order of 10^{-16}.

For each test function, the Table reports the dimension of the objective function argument (n), the number of iteration (N_t), the number of function evaluations (N_f), the value of the objective function at the minimum x^* (f^*) and the value of the objective function at the termination point x_t (f_t).

The convergence criterion is

$$| f - f^* | < 10^{-10} \max(1, | f |). \tag{4.1}$$

For the size of the differencing interval h_j, in (3.5), we suggest that

$$h_j = 10^{-8} \| C_j \|_2, \qquad j = 1, \cdots, n. \tag{4.2}$$

Remark 4.1. In order to estimate derivative information by the central difference formula, 2n extra function evaluation must be made at every iteration. Thus an ideal amount of calculation is that $\hat{N}_f = (2n+1)N_t$.

References

[1] K.W. Brodlie, A.R. Goulay and J.Greenstadt, Rank-one and Rank-two corrections to positive definite matrices expressed in product form, *J. Inst. Maths. Applics.*, 11 (1973), 73–82.

[2] I.D.Coope, A conjugate direction implementation of the bfgs algorithm with automatic scaling, Research Report, No. 42, Department of mathematics, University of Canterbury, 1987.

[3] W.C. Davidon, Optimally conditioned optimization algorithms without line searches, *Math. Prog.*, 9 (1975), 1–30.

[4] S-P.Han, Optimization by updated conjugate subspace, in "Numerical Analysis: Pitman Research Notes in Mathematics Series", 140, 82–97.

[5] Ladislav Lukšan, Computational experience with improved variable metric methods for unconstrained optimization, *Kybernetika,* 26 (1990), 415–430.

[6] J.J. More, B.S. Garbow and K.E. Hillstrom, Testing unconstrained optimization software, *ACM Transaction on Mathematical Software,* 7 (1981), 17–41.

[7] M.R.Osborne, An efficient weak line search with guaranteed termination, MRC Report No.1870, Mathematics Reseach Center, University of Wisconsin, 1978.

[8] M.R. Osborne and M.A. Saunders, Descent methods for minimization, in "Optimization", Eds: R.S. Andersen, L.S. Jennings and D.M. Ryan, University of Queensland Press, 1972, 221–237.

[9] M.R. Osborne and L.P. Sun, A new approach to the symmetric rank one updating algorithm, CSTR Report NMO/02, Stat.Res.Sec., SMS, ANU, 1988.

[10] M.J.D. Powell, Updating conjugate direction by the BFGS formula, *Math. Prog.*, 38 (1987), 29-46.

[11] K. Schittkowski, More Test Examples for Nonlinear Programming Codes, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, 282, 1987.

[12] R.B. Schnabel, Sequential and Parallel Methods for Unconstrained Optimization, in Mathematical Programming Recent Developments and Applications, Eds: Masao Iri and Kunio Tanabe, Kluwer Academic Publishers, The 13th International Symposium on Mathematical Programming, 1989, 227–262.

[13] D.F.Shanno, Conjugate gradient methods with inexact searches, *Math. Comp.*, 34 (1978), 499-514.

[14] L.P.Sun, Scaling Rank-one updating formula and its application in unconstrained optimization, Ph.D Thesis, ANU, 1990.

[15] M.A.Wolfe and C.Viazminsky, Supermemory descent methods for unconstrained minimization, *JOTA,* 18 (1976), 455–469.