Measurement of Algorithm Performance in Minimization of Constrained Problems

SALIM A. SALEH
College of Engineering, University of Tikrit, Tikrit, Iraq

Abstract:
In this paper, constrained non-linear programming problems are solved by using the Sequential Unconstrained Minimization Technique (SUMT). The most popular formulas (DFP& BFGS) are used with SUMT in minimization of constrained problems. Numerical comparison shows that the Number of Constrained evaluations (NOC) must be used instead of Number of Function evaluations (NOF) as a main factor in the measurement of algorithm performance.
Keywords: SUMT, Algorithm Performance, NOF, NO

Notations:

n : Dimension of the problem;
m : Number of the constraints;
SUMT: Sequential Unconstrained Minimization Technique;
DFP: Davidon- Fletcher- Powell formula;
BFGS: Broyden- Fletcher- Goldfarb- Shanno formula;
NOC: Number of constraint evaluations;
NOF: Number of function evaluations;
NOI: Number of iterations;
λ: Step size obtained by the line search procedure;
K: K^{th} iteration;
g: n x 1 gradient of f(x);
S: n x 1 difference vector between two successive points;
P: n x 1 search direction vector;
y: n x 1 difference vector between two successive gradients;
H: n x n Hessian matrix;
QN: Quasi- Newton method**.**

Introduction:

Consider the constrained mathematical problem
Minimize f(x) Subject to $C_j(x) \geq 0$; j= 1, 2,…, m
Where $x= (x_1, x_2, \ldots, x_n)$ is an n-dimensional Vector.
The function f(x) is termed the objective or criterion function. The restrictions are stated as nonlinear constraints $C_j(x)$.This problem can be performed with respect to any optimization procedure[1]. Since the constraints $C_j(x)$ are nonlinear it is often particularly advantageous to transform the constrained problem into an unconstrained problem [2].

$$\phi(x,r) = f(x) + r \sum_{j=1}^{j=m} \frac{1}{Cj(x)} \quad .(1)$$

This transformed problem is called the inverse barrier function, which is only suitable for inequality constraints. This problem then can be optimized by the most popular approach in the sequential method, referred to by Fiacco and McCormick in 1968[3] as Sequential Unconstrained Minimization Technique and commonly abbreviated to SUMT.

The defining function $\Phi(x,r)$ becomes infinite at the boundary of the feasible region R, i.e. barriers are constructed on each constraint, and the solution $x_{min}(r) \in R$; then x^*, is approached from the interior of R in a sequence defined by the controlling parameter r, where a sequence of r values tending to zero is used. The growth of ($C_j^{-1}(x)$) can be controlled or " canceled " by decreasing r. Each constraint has its inverse barrier function, which has the necessary property that $C_j^{-1}(x) \rightarrow \infty$ as $C_j(x) \rightarrow 0$. In addition, as $r \rightarrow 0$ the effect of barrier term is steadily reduced to take effect nearer to the boundary of the feasible region. The SUMT algorithm basically consists of the following steps.

Step 1: Select an initial value for r [r_k; where k=0] which tends the decreasing sequence of $r_k \rightarrow 0$ as $k \rightarrow \infty$. Select $x_0 \in R_0$.

Step2: Minimize $\Phi(x,r_k) = f(x) + r_k \sum_{j=1}^{j=m} 1/C_j(x)$

Step 3: From x_k , x_{k+1} , g_k , g_{k+1} , *and* H_k ,the new matrix H_{k+1} is calculated.

Step 4: Increment $k = k+1$ and return to step 2 if convergence is not satisfied.

The initial value given to r [i.e. r_0] is important in reducing the number of iterations to minimize $\Phi(x,r)$. In many problems the value $r_0=1$ is acceptable; however, an initial value for r, suggested by Fiacco and McCormick in 1968 which appears to give good results in general,

$$r = \frac{-\nabla f_{(x)}{}^{T} \nabla Z(X)}{\nabla Z_{(x)}{}^{T} \nabla Z(X)} \text{;where } Z(X) = \sum_{j=1}^{j=m} 1/C_j(x).$$

The usual method of reducing r is simply to define $r_{k+1}=r_k/c$, where $c=10$, though many other sequences have been explored. It should be noted that the current point must remain feasible throughout the calculations. If a non-feasible point is reached at any time, then the calculations continue with a suitably reduced step length from the last feasible point.

Many variants of the Quasi-Newton (QN) methods have been written to solve the problem of minimizing an unconstrained function $\Phi(x,r)$ whose gradient is available. The DFP technique was originally proposed by Davidon (1959) and subsequently improved by Fletcher and Powell (1963)[8]. Later BFGS technique has been devised by Broyden (1970), Fletcher (1970), Goldfarb (1970), and Shanno (1970)[5,6,7,and 8]. The central feature of all QN implementations in the use of successive approximations to the inverse Hessian matrix H of $\Phi(x,r)$. If at the point x_k the gradient is g_k and the inverse Hessian matrix is H_k then a new point x_{k+1} is given by $x_k+s_k=x_k+\lambda_k P_k$ where (λ_k is a scalar chosen to ensure that $\Phi_{k+1} < \Phi_k$, and the search direction $P_k=-H_k g_k$). The inverse Hessian approximation is tend revised by considering the change in gradient ($y_k=g_{k+1}-g_k$) caused by the move ($s_k=x_{k+1}-x_k$). Several formulas for obtaining H_{k+1} from H_k have been used, DFP & BFGS are the most important formulas.

DFP formula was expressed as[4] :

$$H_{k+1} = H_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{H_k y_k \cdot y_k^T H_k}{y_k^T H_k y_k} \quad .(2)$$

BFGS formula is superior in almost all cases[4] was expressed as:

$$H_{k+1} = H_k - \frac{H_k y_k s_k^T + s_k y_k^T H_k}{s_k^T y_k} + \left(1 + \frac{y_k^T H_k y_k}{s_k^T y_k}\right) \frac{s_k s_k^T}{s_k^T y_k} \quad .(3)$$

Each of these formulas can be used with SUMT in minimization of constrained problems.

Measurement of Algorithm Performance:

Usually NOF was used as a main factor in the measurement of algorithm performance in minimization of unconstrained problems[4,9]. In minimization of constrained problems many authors used to adopt NOF to measure the algorithm performance[10,11]. While SUMT is used in minimization of constrained problems, it is based on the strategy of calling objective function (NOF) and constraint functions (NOC) in different stages of computations (see Bunday)[12]. It calls the objective function and constraint functions together in three stages while it calls additionally the constraint functions in the fourth stage. These stages are:

A-To find the current point *P*, the obj. fun. (NOF) and the constraint fun. (NOC) are called together;

B-To find the value of λ_k so that no constraint is violated, only the constraint fun. (NOC) is called;

C-To find the next point *Q*, the obj. fun. (NOF) and the constraint fun. (NOC) are called together;

D-To investigate that no minimum between *P & Q*, and replace *P* by *Q*, (NOF) and (NOC) are called together.

Since the value of λ is founded in stage (B), NOC of this stage can be replaced by NOL. Then the total number of constraint function evaluations given through all stages must be equal to:

NOC= NOL + NOF

From this equation we conclude that (NOF), which was used by some authors as Algorithm performance in unconstrained problems solving, is included in total (NOC).

We propose (NOC) as the performance indicator needed to solve the constrained problems; however, the iteration's number (NOI) and (NOF) are also included.

Each of DFP & BFGS formulas was used with SUMT in minimization of constrained problems. Hence, the efficiency of the SUMT algorithm can be measured by the following rules[4]:

(A) Let R_i= the ratio of new algorithm's NOI to the basis algorithm NOI (i.e. $(NOI)_{new}/(NOI)_{basis}$=*relative iteration*).

(B) Let Rc_i= relative constraint evaluation of new algorithm

$= (NOC/NOI)_{new}$

(C) Find R_{cost}=relative cost of new algorithm

$=R_i * Rc_i$

(D) Find the performance factor of the new algorithm with respect to the basis algorithm as:

$P\%=100 * [1-\{R_{cost}\}_{new}/\{R_{cost}\}_{basis}]$.

Numerical Computation and Conclusions:

The computer program is written in FORTRAN 77 to implement all updating formulas (i.e. DFP &BFGS)with the new proposed indicators NOL & NOC for solving constrained problems. The program which used SUMT algorithm following Bunday[12] was intended to couple DFP and BFGS formulas with our proposed indicators. They are tested by the constrained problems (Appendix A) and compared each one to another. The performance factor was measured by the rules mentioned in section (2).

The same termination criteria are applied in the implemented program, namely that we have convergence if successive minimum of $\Phi_i(x,r)$, $i=1,2,\ldots$ are such that $|(\Phi_i - \Phi_{i+1})/\Phi_i| \leq \varepsilon$; where ε was used to be equal to (*1. E-4*). This condition can of course be modified so that the programming actually terminates when the above condition and

$[r_k \sum_{j=1}^{j=m} 1/C_j(x_k^*) \leq \varepsilon]$ both are hold[12].

Table (1) gives the number of iteration (*NOI*), number of function evaluations (*NOF*), number of constraints evaluations (*NOC*) and the number of constraints evaluations used to find the values of λ_k through the program computation (*NOL)* for each of the test constrained problems by using DFP & BFGS formulas mentioned in equations (2) &(3).

Table (1): Performance Parameters for the Standard Algorithms

Test Problem No.	Method							
	DFP				BFGS			
	NOI	NOF	NOL	NOC	NOI	NOF	NOL	NOC
1	29	100	1456	1556	34	118	1572	1690
2	20	70	1993	2063	21	73	1741	1814
3	34	111	4017	4128	36	117	3690	3807
4	42	135	6627	6762	45	142	6573	6715
5	38	125	5412	5537	39	128	5125	5253
6	21	74	2110	2184	23	77	1885	1962
7	25	83	1486	1569	24	80	1251	1331
8	19	70	1852	1922	17	63	1471	1534
Total	**228**	**768**	**24953**	**25721**	**239**	**798**	**23308**	**24106**

In order to compare the effiecncies of such formulas the rules mentioned in section 2 are used to measure the performance of each algorithm based on the standard DFP formula.

Table (2) gives the performance factors of DFP & BFGS formulas based on (*NOC)* parameter.

Table (3) gives the performance factors of DFP & BFGS formulas based on (*NOL)* parameter.

Table (4) gives the performance factors of DFP & BFGS formulas based on (*NOF)* parameter.

Table (2): Performance Factors of Updating formulas based on (NOC) parameter

Optimization Algorithm	**Total**		**Performance Factors**			
	NOI	NOC	R_i	Rc_i	R_{cost}	P %
DFP *basis*	228	25721	1	112.81	112.81	0
BFGS	239	24106	1.05	100.86	105.91	6.122

Table (3): Performance Factors of Updating formulas based on (NOL) parameter

Optimization Algorithm	**Total**		**Performance Factors**			
	NOI	**NOL**	**Ri**	**Rci**	**Rcost**	**P%**
DFP basis	228	24953	1	109.44	109.44	0
BFGS	239	23308	1.05	97.523	102.4	6.436

Table (4): Performance Factors of Updating formulas based on (NOF) parameter

Optimization Algorithm	**Total**		**Performance Factors**			
	NOI	**NOF**	$\mathbf{R_i}$	$\mathbf{Rc_i}$	$\mathbf{R_{cost}}$	**P%**
DFP *basis*	228	768	1	3.368	3.368	0
BFGS	239	793	1.05	3.339	3.506	-4.084

Examining tables (1) through (3) we notice that (*NOL*) takes a high percentage of total (*NOC*) for each algorithm. This indicates that step size λ_k requires a more constrained function evaluations (*NOC*) to check that no constraint is violated through the computations. There is a need to find a powerful method to optimize the step size of each iteration. This method must reduce the total (*NOC*), which proved to be used as a main factor in the measurement of algorithm performance in minimization of constrained problems (see section 2).

Table (2) agrees that BFGS formula is to be more effective than DFP formula and improves the performance factor by 6.122%.

Table (3) shows that BFGS formula was decreased (*NOL*) by 6.436% compared with DFP formula.

Comparing tables (2) & (4) shows that when using (*NOF*)as a main factor of the measurement, DFP formula appeared to be more efficient than BFGS formula which was not true as it had seen in section (1).

References:

1. Rao, S.S. ; and Hati, S.K.; " Computerized selection of optimum machining conditions for a job Requiring Multiple operations" ASME, J. of Engineering for Indus., Vol. 100, Aug. (1978) PP. 356 – 362.
2. Feiring, B.R.; Phillips, D.T., and Hogg, G.L., "Computational experience with an exact penalty function technique (EPT) ", Comput. And Indus. Engineering, Vol. 5, No. 3, (1981) PP. 205 – 216.
3. Gill, P.E.; and Murray, W. (eds.), "Numerical methods for constrained optimization ", Academic Press. Inc., London, (1974).
4. Scales, L.E., "Introduction to Non-linear optimization ", Macmillan, London, (1985).
5. Broyden, C. G., "The convergence of a class of double- rank minimization algorithms ", J.I.M.A., 6, (1970).
6. Fletcher, R., "Practical methods of optimization ", 2nd ed., John Willey and Sons, (1987).
7. Goldfarb, D., "A family of variable metric methods derived by Variational means ", Math. Comp., 24 (1970).
8. Shanno, D. F., "Conditioning of Quasi- Newton methods for function minimization ", Math. Comp., 24, (1970).
9. Taha, D. B., " A study of different programming Languages Implementiry a New Quasi- Newton Method ", M.Sc. Thesis, University of Mosul, College of Science, (1995).
10. Muhanad, M.S., "Investigation on the use of different numerical techniques in CAD systems ", Ph.D. Thesis Brunel University, Dept. of Manufacturing and Eng. System, UK (1991).
11. Tassopoulos, A., "The use of Non- Quadratic Models in Optimization ", Ph.D. Thesis University of Technology, Loughborugh, UK, (1982).
12. Bunday, B.D., "Basic Optimization Methods ", Edward Arnold Pub. Ltd., London, (1985).

قياس أداء خوارزميات التقليل للمسائل المقيدة

سالم عبدا لله صالح الدليمي

كلية الهندس، جامعة تكريت، تكريت، جمهورية العراق

الملخص:

يتم تحديد النتائج المثلى للمشاكل المقيدة اللاخطية باستخدام تقنية التقليل المتتابع اللامقيد (*SUMT*). من الخوارزميات الشائعة الاستخدام بهذه التقنية خوارزميات (*DFP & BFGS*). يستخدم معيار عدد مرات تقييم الدالة (*NOF*) عادة للمقارنة بين الخوارزميات المعتمدة في حل المشاكل اللامقيدة. بنفس السياق اعتاد بعض الباحثين على استخدام هذا المعيار في تقييم الخوارزميات المعتمدة في حل المشاكل المقيدة لا سيما تلك التي تستخدم مع تقنية (*SUMT*). لأجل التحقق من ذلك تناولت هذه الورقة المقارنة التحليلية بين هذا المعيار ومعايير مقترحة أخرى. أظهرت تلك المقارنة إن معيار عدد مرات تقييم القيود (*NOC*) المقترح هو المعيار الشامل والمعول عليه في تقييم الخوارزميات المستخدمة مع تقنية (*SUMT*) في حل المشاكل المقيدة.

الكلمات الدالة: المشاكل المقيدة، كفاءة الخوارزميات، تقنية التقليل المتتابع اللامقيد.

Appendix A

Constrained Test Problems

Problem 1:

Min. $F(X)= (X_1-1)(X_1-2)(X_1-3)+ X_3$

S.T.

$-X_1^2 - X_2^2 + X_3^2 \geq 0$

$X_1^2 + X_2^2 + X_3^2 - 4 \geq 0$

$X_3 \geq 0$

$X_i \geq 0$

$X_0 =(0.1, 2.0, 2.1)$; $X^* =(0, \sqrt{2}, \sqrt{2})$; $F(X^*)= -6 + \sqrt{2}$

Problem 2:

Min. $F(X)= -X_1 X_2 X_3$

S.T.

$X_1^2 + 2X_2^2 + 4X_3^2 \leq 0$

$X_i \geq 0$

$X_0 =(1.0, 1.0, 1.0)$; $X^* =(4.0, 2.83, 2.0)$; $F(X^*)= -22.627$

Problem 3:

Min. $F(X)= -X_1 X_2 X_3$

S.T.

$X_1 + 2X_2 + 2X_3 \leq 72$

$X_i \leq 42$

$X_i \geq 0$

$X_0 =(20.0, 10.0, 10.0)$; $X^* =(24.0, 12.0, 12.0)$; $F(X^*)= -3456.0$

Problem 4:

Min. $F(X)= -X_1 X_2 X_3$

S.T.

$X_1 < 20$

$X_2 \leq 11$

$X_3 \leq 42$

$X_i \geq 0$

$X_0 =(15.0, 10.0, 20.0)$; $X^* =(20.0, 11.0, 42.0)$; $F(X^*)= -9240.0$

Problem 5:

Min. $F(X)= -X_1 X_2 X_3$

S.T.

$X_1 + 2X_2 + 2X_3 \leq 72$

$X_1 \leq 20$

$X_2 \leq 11$

$X_3 \leq 42$

$X_i \geq 0$

$X_0 =(15.0, 10.0, 15.0)$; $X^* =(20.0, 11.0, 15.0)$; $F(X^*)= -3300.0$

Problem 6:

Min. $F(X)= -X_1 X_2 X_3$

S.T.

$2X_1^2 + X_2^2 + 3X_3^2 \leq 51$

$X_i \geq 0$

$X_0 =(1.0, 1.0, 1.0)$; $X^* =(2.9155, 4.1231, 2.3805)$; $F(X^*)= -28.6153$

Problem 7:

Min. $F(X)= X_1^2 + X_2^2 + X_3^2$

S.T.

$X_1 + X_2 + X_3 \geq 3$

$X_1 \ X_2 \ X_3 \geq 3$

$X_i \geq 0$

$X_0 =(1.0, 2.0, 3.0)$; $X^* =(1.4422, 1.4422, 1.4422)$; $F(X^*)= 6.2403$

Problem 8:

Min. $F(X) = -(X_2^3 / 27\sqrt{3}) (9 - (X_1 - 3)^2)$

S.T.

$0 \leq X_1 + \sqrt{3} X_2 \leq 6$

$0 \leq X_2 \leq X_1 / \sqrt{3}$

$X_i \geq 0$

$X_0 = (1.0, 0.5)$; $X^* = (3.0, \sqrt{3})$; $F(X^*) = -1.0$