ALM 방법에 의한 수치해석적 최적화

Numerical Optimization Via ALM Method

김 민 수,* 이 재 원**
Min Su Kim, Jae Won Lee

1. 서 론

Numerical Optimization은 Computer Hardware의 발달에 힘입어 최근 급속히 발달하고 있으며, 항공, 선박, 자동차공업, 엔지니어링 분야에서 중량감소, 원가절감과 성능향상 등을 위한 설계기술의 도구로 널리 적용되고 있다.

본고에서는 이러한 추세에 따라서, 보다 효율적인 Optimization Program에 대해서 소개하고자 한다. 사용한 최적화 알고리즘은, ALM(Augmented Lagrange Multiplier) 방법을 적용해서 구속조건이 있는 문제를 구속조건이 없는 문제로 변환한 후, Self-Scaling BFGS(Broydon-Flecher-Goldfarb-Shanno)를 적용한다. BFGS의 각 Descent 방향에서의 Step 길이는, Sequential Search로 Unimodal Point를 구해서, Golden Section방법으로 Refine을 한 후, Cubic Approximation을 적용해서 구한다.

2. 최적화 알고리즘

본 연구에서 적용한 최적화 알고리즘의 전체적인 흐름은 Fig.1과 같다. 알고리즘의 선택은 Constrained Multi-Variable Problem은 ALM, Unconstrained Multi-Variable Problem은 BFGS로 풀고 Unconstrained One Variable Problem은 Line Search로 푼다.

2.1 ALM(Augmented Lagrange Multiplier) Method

ALM방법은 일반적인 Penalty Function방법처럼, Penalty Factor를 ∞ 또는 0.0으로 접근시키지 않아도 Feasible Region내로 수렴을 하기 때문에 Stable하고, Lagrange Multiplier를 Update할 때 마다 수렴이 가

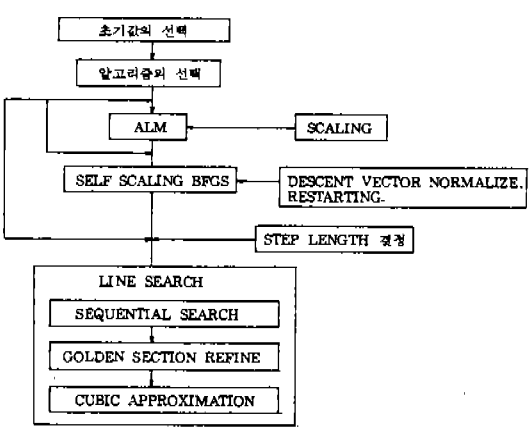


Fig.1 최적화 알고리즘의 흐름도

* 대우자동차㈜ 기술연구소, 연구원
** 대우자동차㈜ 기술연구소, 책임연구원

속된다. Optimum Solution에서 Lagrange Multiplier가 0.0이 아닌 구속조건이 Active하며, Equality 구속조건이 있는 문제에도 강점을 보인다.[1)]

일반적인 최적화 문제를 다음과 같은 형태로 표현하자.

$$\begin{aligned}&\text{Minimize}: F(\underline{x})\\&\text{Subject to}: H_i(\underline{x})=0.0 \quad i=1,\cdots,m\\&\qquad\qquad G_j(\underline{x})\leqq 0.0 \quad j=1,\cdots,n\end{aligned} \tag{1}$$

위와 같은 구속조건이 있는 문제를 Lagrange Multiplier와 Penalty Factor를 적용해서 다음과 같이 구속조건이 없는 Pseudo Function, A로 변환한다.

$$\begin{aligned}&\text{Minimize}: A(\underline{x},\underline{\mu},\underline{\lambda},\underline{r})\\&=F(\underline{x})+\sum_{i=1}^{m}\{\mu_i H_i(\underline{x})+r\cdot H_i(\underline{x})^{**}2\}\\&\quad+\sum_{j=1}^{n}\{\lambda_j\psi_j+r\cdot\psi_j^{**}2\}\end{aligned} \tag{2}$$

여기서,

r : Penalty Factor.

μ_i : Lagrange Multiplier for Equality Constraints.

λ_j : Lagrange Multiplier for Inequality Constraints.

ψ_i : MAX$[G_j(\underline{x}), -\lambda_i/(2\cdot r)]$

이다.

식(2)의 Pseudo Function, A는 Unconstrained Problem이므로 Self Scaling BFGS를 적용해서 풀고, Lagrange Multiplier는 다음과 같이 Update를 한다.

$$\begin{aligned}\mu_i^{k+1}&=\mu_i^k+2^*r^*H_i(\underline{x})\\\lambda_j^{k+1}&=\lambda_j^k+2^*r^*\psi_j\end{aligned} \tag{3}$$

여기서, k는 ALM의 반복횟수이다.

수렴조건은 Lagrange Multiplier의 값이 변화가 없으면 수렴을 한 것으로 한다. 하지만 Lagrange Multiplier의 값이 때로는 Unstable해서 수렴을 못하는 수가 있기 때문에, 목적함수 $F(\underline{x})$의 값의 상대변화가 없어야 하며, $H_i(\underline{x})=0.0$, $G_j(\underline{x})\leqq 0.0$을 모두 만족시켜야 하는 부가적인 수렴조건을 갖는다. 수렴조건은 다음처럼 적용된다.

```
IF {[ABS(μ_i^(k+1) − μ_i^k).LE. Tol 1].AND.
    [ABS(λ_j^(k+1) − λ_j^k).LE. Tol 1]} then;
     CONVERGE!
Else
  IF {[ABS(H_i(x)).LE. Tol 2].AND.
      [G_j(x).LE. Tol 3].AND.
      [ABS(F^(k+1) − F^k).LE. Tol 4* ABS
      (F^k)]} then;
       CONVERGE!
  Else  Goto Next Step;
```

(4)

여기서, Tol 1, Tol 2, Tol 3, Tol 4는 0.0에 가까운 매우 작은 값이다.

2.2 Self-Scaling BFGS(Broydon-Fletcher-Goldfarb-Shanno) Method

BFGS는 Quasi-Newton Method의 일종으로서 Global Convergence가 보장되며, DFP (Davidon-Fletcher-Powell)보다 Line Search 값의 정확도에 덜 민감하고, Restarting의 횟수가 적은 것으로 알려져 있다.[2)]

BFGS의 알고리즘을 살펴 보면, Pseudo Function, 식(2)의 값이 감소하는 방향(Descent Direction)을 Newton Method에 따라서 다음처럼 정의한다.

$$\underline{d}_k=-\underline{H}_k^*\,\underline{g}_k \tag{5}$$

여기서, $\underline{H}_k$는 Hessian의 Inverse Matrix, $\underline{g}_k$는 Gradient Vector이다. 설계변수는, 식(6)의 방향을 따라서, 적절한 Step Length(α)를 Line Search를 통해서 구한 후 다음과 같이 Update된다.

$$\underline{x}_{k+1}=\underline{x}_k+\alpha^*\,\underline{d}_k \tag{6}$$

이때, $\underline{H}_k$를 직접 구하는 것이 힘들기 때문에, 근사적으로 표현하는 방법에 따라서, BFGS와 DFP로 구분된다. BFGS의 $\underline{H}_k$ 근사식은 다음과 같으며, 매 초기에는 $\underline{H}_k=\underline{I}$

(Identity Matrix)로 가정한다.

$$\underline{H}_{k+1}=\left(\underline{H}_k-\frac{\underline{r}\,\underline{p}^t+\underline{p}\,\underline{r}^t}{\sigma}\right)\frac{\sigma}{\tau}+\frac{\sigma+\tau}{\sigma^2}\,\underline{p}\,\underline{p}^t \quad (7)$$

여기서, $\underline{p}=\underline{x}_{k+1}-\underline{x}_k$, $\underline{y}=A^{k+1}-A^k$, $\underline{r}=\underline{H}_k\cdot\underline{y}$, $\sigma=\underline{p}\cdot\underline{y}$, $\tau=\underline{y}\cdot\underline{r}$이다.

BFGS의 수렴조건은 $\|\underline{g}_k\|$ .LE. Tolc이거나, Line Search가 향상된 해를 구하지 못하면 수렴한 것으로 한다. Tolc는 0.0에 가까운 매우 적은 값이다.

2.3 Line Search

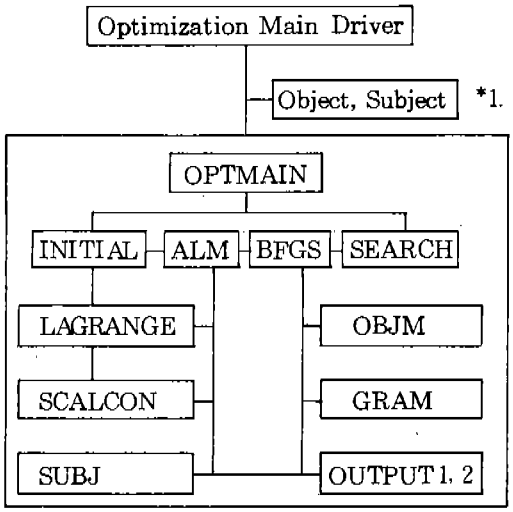
Line Search는 Sequential Search로 개략적인 Unimodal Point를 구해서 Golden Section으로 Refine을 한 후에, Gubic 함수로 Approximation하고, 이 근사함수의 Minimum 값과 Golden Section에 의한 Minimum 값중 최소값을 해로 한다.[1)]

3. 프로그램의 개발

프로그램은 Main Driver와 Optimization Library로 구성되어 있다. Object, Subject는 목적함수와 구속함수의 FUNCTION이다. OPTMAIN은 Library내의 모든 Subroutine을 Control하는 MAIN Routine이다. INITIAL은 초기값을 정하는 Routine이고, ALM은 Augmented Lagrange Multiplier Routine이고, BFGS는 Self-Scaling BFGS Routine이고, Search는 Line Search Routine이다. Lagrange는 Lagrange multiplier의 초기값을 계산하고, Scalcon은 구속함수의 Scaling Factor를 계산하고, Subj는 목적함수와 구속함수의 도함수를 계산하고, GraM은 BFGS를 위한 Pseudo Function, A의 도함수를 만든다.

본 프로그램에서 사용하는 Subroutine 들의 기능을 살펴보자.

1. OPTMAIN : 문제의 형태에 따른 알고리즘, 각 알고리즘에서 사용하는 Stop Criterion과 Output Option 등을 선택한다.



*1. 은 사용자가 정의해야만 한다.

Fig.2 STRUCTURE OF PROGRAM

2. INITIAL : 초기점을 결정한다. User는 직접 Key-in하거나, 적절한 초기값을 구할 수 없을 경우는, Random Search를 선택하므로서, 설계변수의 Bound 값만 Key-in하면 된다.
3. ALM : ALM의 Main Routine
4. LAGRANGE : Lagrange Multiplier 의 초기값을 결정한다.
5. SCALCON : Constraints의 Scaling Factor를 결정한다.
6. OUTPUT1 : ALM의 계산 내용을 출력한다.
7. BFGS : BFGS의 Main Routine
8. OBJM : Unconstrained Problem의 목적함수값을 계산하거나, ALM의 Pseudo Function값을 계산한다.
9. GRAM : Unconstrained Problem의 목적함수의 Gradient나, ALM의 Pseudo Function의 Gradient를 계산한다.
10. OUTPUT 2 : BFGS의 계산 내용을 출력한다.
11. SEARCH : Line Search를 수행한다.

사용자가 정의 해야만 하는 Object와 Subject의 Format과 프로그램의 실행에는 APPENDIX A를 통해서 알아 보기로 한다.

4. NUMERICAL TEST

프로그램에 다음과 같은 Test문제를 적용해서 해석능력을 알아본다. 수렴한 계 값인 Tol 1=1.0E-5, Tol 2=1.0E-3, Tol 3=1.0E-5, Tol 4=1.0E-5, Tol 5=1.0E-5로 하고, Line Search에서 Golden Section의 Refine비는 1.0E-3으로 한다.

〔예제 1〕 Rosen-Suzuki Problem[3,4]

많은 Local Minimum을 포함하고 있는 문제로 형태는 다음과 같다.

Minimize : $F(\underline{x})=x1^2+x2^2+2\cdot x3^2+x4^2-5\cdot x1-5\cdot x2-21\cdot x3+7\cdot x4$

Subject to :

$G1(\underline{x})=(x1^2+x2^2+x3^2+x4^2+x1-x2-x3-x4)/8.0-1.0\leqq 0.0$

$G2(\underline{x})=(x1^2+2\cdot x2^2+x3^2+2\cdot x4^2-x1-x4)/10.0-1.0\leqq 0.0$

$G3(\underline{x})=(2\cdot x1^2+x2^2+x3^2+2\cdot x1-x2-x3)/5.0-1.0\leqq 0.0$

프로그램이 초기점을 (0, 0, 0, 0)으로 해서 구한 최적값은 (0.363, 1.08, 2.07, −0.651)이고, $F=-44.923$으로서, 〔3〕에서 초기점을 (0, 0, 0, 0)으로 할 때 Optimum Point (0, 1, 2, −1), $F=-44$보다 향상된 값을 구했다. 총 반복횟수는 ALM=3, BFGS=7, LINE=211이다. 여기서 LINE은 Line Search중에 함수계산 횟수를 의미한다.

〔예제 2〕 Miele et al. Problem[6,9]

구속조건이 Equality로만 구성되어 있기 때문에, 일반적으로 수렴이 까다롭다.

Minimize : $F(\underline{x})=(x1-1)^2+(x1-x2)^2+(x3-1)^2+(x4-1)^4+(x5-1)^6$

Subject to :

$H1(\underline{x})=x1^2\ x2+\sin(x4-x5)-2\sqrt{2}=0.0$

$H2(\underline{x})=x2+x3^4\cdot x4^2-8-\sqrt{2}=0.0$

프로그램이 구한 최적값은 $F=0.24149$이고, 설계변수값은 (1.17, 1.18, 1.38, 1.51, 0.611)이다. 〔9〕에서의 최적값은 $F=0.24149$이다. 총 반복횟수는 ALM=4, BFGS=41, LINE=511이다.

〔예제 3〕 Luus, R Problem[7]

많은 Local Maximum Point를 가지고 있는 문제로 형태는 다음과 같다.

Maximize : $F(\underline{x})=x1^2+x2^2+x3^2+(x3-x4)^2$

Subject to :

$G1(\underline{x})=16-4(x1-1/2)^2-2(x2-0.2)^2-x3-0.1\cdot x1\cdot x2-0.2\cdot x2\cdot x3\geqq 0.0$

$G2(\underline{x})=2\,x1^2+x2^2-2\,x3^2-2\geqq 0.0$

$G3(\underline{x})=x2+x4\geqq 0.0$

$-2.3\leqq xi\leqq 2.7 \quad i=1,\cdots,4$

프로그램에서는 Random Search를 이용해서 초기값을 (1.8702, 2.6166, −1.3938, 2.1283)로 구했다. 최적값은 (1.4022, 2.3833, −1.9510, 2.700)이고, 이때 $F=33.0841$이다. 〔7〕에서의 Optimal Solution은 $F=11.68$이었다.

프로그램에서 구한 값이 대단히 향상된 값임을 알 수 있으며, 〔7〕에서의 Optimum Solution은 Global Maximum값이 될 수 없음도 알 수 있다. 총 반복횟수는 ALM=6, BFGS=30, LINE=691이며 Active한 구속조건은 $G1$ (2.9279E-6)과 $G2$(8.7198E-6)이다.

〔예제 4〕 Optimal Design of a through circulation system for drying catalyst pellets[8]

Through circulation system의 production rate를 최대로 하는 Fluid velocity, $x1$과 Bed depth, $x2$를 구하는 문제로 Formulation 과정에서 설계변수의 Weight를 잘못 정한 대표적인 Case이다.

Minimize : $F(\underline{x})=0.0064\cdot x1(\exp(-0.184\cdot x1\cdot x2)-1.0)$

Subject to :

$G1(\underline{x})=(3,000+x1)\,x1^2\,x2/1.2E+13-1.0\leqq 0.0$

$G2(\underline{x})=\exp(0.184\,x1^{0.3}\cdot x2)/4.1-1.0\leqq 0.0$

수치해석의 Stability를 높이기 위해서 설계변수를 Scaling하자. $x1=x1^*\,10000.0$

Minimize : $F(\underline{x})=64.0\,x1\cdot(\exp(-1,840\,x1$

$x2)-1.0)$

Subject to :

$G1(\underline{x})=(0.3+x1)\,x1^2\,x2/12.0-1.0 < 0.0$

$G2(\underline{x})=\exp(2.9162\,x1^{0.3}\,x2)/4.1-1.0<0.0$

초기값을 (3.0, 0.25)로 했을 때, 최적값은 (3.1770, 0.3421)이고, $F=-153.71$이다. 총 반복횟수는 ALM=4, BFGS=30, LINE=547이다.

〔예제 5〕 Optimal Design of Vibration Absorbers[10]

Main System의 Amplitude를 최소로 하는 Absorber의 ζ_2와 상대 진동수 Ω_r를 구하는 문제이다. μ, ζ_1와 Ω는 입력으로 주어진다.

Minimize : $F(\underline{x})=x_1/(f/k_1)$

Subject to : $G1(\underline{x})=x_2/x_1-R\leqq 0.0$

$0.0\leqq\Omega_r\leqq\Omega_r^u$

$0.0\leqq\zeta_2\leqq\zeta_2^u$

여기서,

$$\frac{x_1}{f/k_1}=\frac{1}{A}\sqrt{(\Omega_r^2-\Omega^2)^2+4(\zeta_2\Omega\Omega_r)^2}$$

$$\frac{x_2}{f/k_1}=\frac{1}{A}\sqrt{\Omega_r^4+4(\zeta_2\Omega\Omega_r)^2}$$

$$\omega_i=k_i/m_i \quad i=1,2$$

$$\zeta_i=\frac{C_i}{2\sqrt{m_i\,k_i}} \quad i=1,2$$

$$\Omega=\omega/\omega_r,\ \Omega_r=\omega_2/\omega_1,\ \mu=m_2/m_1$$

$$A^2=[\Omega^4-\Omega^2-\Omega^2\Omega_r^2(1+\mu)-4\zeta_1\zeta_2\Omega^2\Omega_r+\Omega_r^2]^2+4\Omega^2\Omega_r^2[\zeta_1\Omega^2+\zeta_2\Omega^2(1+\mu)-\zeta_2-\zeta_1\Omega_r]^2$$

위의 문제에서 초기점에 관한 자료가 없기 때문에, 본 프로그램에서는 Random Search를 이용해서 구했다. 본 프로그램의 결과는 아래와 같고, [10]의 Optimum Solution에 비해서, 밑줄 친 부분의 값이 30% 향상된 minimum 값이 구해졌고, 나머지는 동일한 값이 구해졌다.

이상과 같은 예제를 통해서 알아본 결과 본 연구에서 제안한 방법이 매우 효과적임을 알 수 있었다. 하지만 문제 Formulation 과정에서 설계변수 값의 Order 차이가 심한 경우는 수렴은 하지만 정확한 Optimum Point를 구하지 못하였다. 본 프로그램의 Numerical Test를 위한 추가 예제는 APPENDIX B에 수록하였다.

5. 결 론

본 연구에서 개발한 프로그램은 Test Problem을 통해서 효율적임을 알 수 있었다. 다만 문제의 Formulation에서 설계변수 Order의 차이가 큰 경우는 정확한 Optimum Point를 구하지 못했지만, Optimum에 가까운 값에 수렴을 한다. 따라서, 일반적으로 안전성이 있음을 알 수 있다.

본 연구에서 개발한 프로그램에 Structure Analysis와 Dynamic Analysis 프로그램을 적용하면 Shape Optimization, Vehicle Dynamic System Optimization과 Weight Reduction 등 다양한 분야의 설계자동화에 기여할 것이다.

$R=5.0,\ \mu=0.2,\ \zeta_1=0.2$

	Ω	0.2	0.6	1.0	1.4	1.8
계산 결과	Ω_r	0.1938	0.5661	0.9920	1.460	1.872
	ζ_2	0.09353	0.07574	0.1009	0.09755	0.09824
	$x_1/(f/k_1)$	0.77822	0.7778	0.71666	0.28034	0.14189

참 고 문 헌

1. Vanderplaats, G.N. "Numerical Optimization Techniques for Engineering Design" McGraw-Hill (1984)
2. Luenberger, D.G. "Linear and nonlinear Programming" 2nd Ed. Addison-Wesley (1984)
3. Reklaitis, G.V., Ravindran, A. and Ragsdell, K.M. "Engineering Optimization Methods and Application" John Wiley and Sons (1983)
4. Rosen, J.B. and Suzuki, S., "Construction of nonlinear Programming Test Problems", Communication of Association for Computing Machinery, Vol.8 pp.113 (1965)
5. Miele et al., J. Optim. Theory Appl., 10, 1 (1972)
6. Svanberg, K. "The Method of Moving Asymptotes - A New Method for Structural optimization" International Journal for Numerical Methods in Engineering Vol.24, pp.359 (1987)
7. Luus, R. AICHE J. 20, pp.608 (1974)
8. Thygeson, J.R. and E.D. Grossmann, AICHE J., 16, pp.749 (1970)
9. Scott A. Burns "Generalized Geometric Programming with many Equality Constraints" International Journal for Numerical methods in Engineering Vol.24, pp. 725-741 (1987)
10. Nader, D. Ebrahimi "Optimum Design of Vibration Absorbers" Transaction of the ASME Technical Briefs Vol.109, April (1987)
11. Gill, P.E., Walter Murray "Pratical Optimization" Academic press, Inc. (1981)
12. Waren, A.D., Hung, M.S. and Lasdon, L.S. "The Status of Nonlinear programming Software: An Update" Operations Research Vol.35, No.4 (1987)
13. Sandgren, E., Ragsdell, K.M. "The Utility of Nonlinear Programming Algorithms: A Comparative Study - Part I,II" Transactions of the ASME (Journal of Mechanical Design) pp.540-551 (1980)
14. Lasdon, L.S. Beck, P.O. "Scaling Nonlinear Programs" Operations Research Letters, pp.6-9 October (1981)
15. Parkinson, A., Wilson, M. "Development of a Hybrid SQP-GRG Algorithm for Constrained Nonlinear Programming" Transaction of the ASME (Journal of Mechanisms, Transmissions, and Automation in Design) pp.308-315 (1988)
16. Belegundu, A.D. "Probabilistic Optimal Design Using Second Moment Criteria" Transaction of the ASME (Journal of Mechanisms, Transmissions, and Automation in Design) pp.324-329 (1988)

APPENDIX A: 프로그램의 사용방법과 실행 예

〔예제 1〕의 문제를 프로그램에 적용하기 위해서는 Table 1과 같은 Format으로 작성해야만 한다. 사용자는 Table 1. BOX 부분에 목적함수와 구속함수를 기입한다. 이때 주의할 것을 BOX를 제외한 부분은 프로그램에서 사용하는 Standard Format이므로 변경해서는 안된다.

사용자는 Table 1과 같은 Source Code를 작성후에 Table 2의 Batch File을 실행시키면 된다. 이때 사용자는 Table 2의 밑줄 친 부분에 있는 사용 Fortran Compiler와 Library의 이름과 Directory Path를 수정해야만 한다. 편의상 Table 1의 내용이 저장된 File의 이름을 "TEST 1. FOR"라 하고서 실행을 시켜보자.

Active Path : >HRUN TEST1

Table 2의 Batch Program은 IBM PC를 위한 것으로, 본 연구에서 제공하는 Routine은 Main Routine으로 DRIVER. OBJ, Numerical Optimization Library로 IDOL.

```
C ---------------------------------------------------------------- C
C   PROBLEM 1: ROSEN - SUZUKI PROBLEM.                             C
C        This is purely a mathematical problem for testing         C
C       optimization algorithm                                     C
C        (Rosen J. B. and S. Suziki, Commun. ACM 8,113 1965).      C
C ---------------------------------------------------------------- C
      FUNCTION OBJECT(x)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (Imax=50)
      DIMENSION x(Imax)
      Object=x(1)*x(1)+x(2)*x(2)+2*x(3)*x(3)+x(4)*x(4)-5*x(1)-
     &        5*x(2)-21*x(3)+7*x(4)
      Return
      End

      SUBROUTINE Subject(x)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (Imax=50)
      DIMENSION x(Imax)
C ----------------------- Using COMMON Block ---------------------- C
      COMMON/Block3/Hi(Imax),Gj(Imax),Pusai(Imax),Q(Imax),W(Imax),
     +               r,Scale,Number,Neq,Nneq
C ------------------------------------------ C
C     Hi(i) : Equality Subject Function.     C
C     Gj(j) : Inequality Subject Function.   C
C ------------------------------------------ C
      Gj(1)=(x(1)*x(1)+x(2)*x(2)+x(3)*x(3)+x(4)*x(4)+x(1)-
     &      x(2)+x(3)-x(4))/8.0-1.0
      Gj(2)=(x(1)*x(1)+2*x(2)*x(2)+x(3)*x(3)+2*x(4)*x(4)-
     &      x(1)-x(4))/10.0-1.0
      Gj(3)=(2*x(1)*x(1)+x(2)*x(2)+x(3)*x(3)+2*x(1)-x(2)-x(3))/5.0-1.0
C
      Return
      End
```

Table1. Format Statement of OBJECT, SUBJECT

```
ECHO OFF
CLS
ECHO        ****************************************************
ECHO        *       MAKE EXECUTION FILE  FOR HARD DISK.        *
ECHO        *--------------------------------------------------*
ECHO        *     INTERACTIVE DESIGN OPTIMIZATION LIBRARY      *
ECHO        *                  VERSION 1.0                     *
ECHO        ****************************************************
ECHO
IF EXIST X1.OBJ GOTO STEP2
FORTRAN X1.FOR /LE/Z/Y3
:STEP2
LINK X1+DRIVER,X1,NUL,IDOL+C:\FORTRAN\PLIB\,NUL
CLS
ECHO       ******************************************
ECHO       *                                        *
ECHO       *     LINK IS COMPLETED  AND RUN !       *
ECHO       *                                        *
ECHO       ******************************************
ECHO
CLS
X1.EXE
CLS
ECHO      ******************************************
ECHO      *                                        *
ECHO      *  NUMERICAL OPTIMIZATION                *
ECHO      *                    IS COMPLETED !      *
ECHO      *                                        *
ECHO      ******************************************
```

Table2. Structure of HRUN. BAT(HRUN. CMD)

LIB이다. Table 2의 DRIVER. OBJ와 IDOL. LIB는 IBM PS/2의 FORTRAN/2를 위한 Version이며, 사용 Operating System과 Fortran Compiler의 종류에 따라서 제공한다.

Table 2의 Batch File을 실행시키면 Table 3과 같은 화면이 떠오른다. 이때 사용자는 화면상의 Command를 따라서 수행해 가면 된다.

Table 3의 OUTPUT Option에서 Output을 Screen에 출력하지 않고 Output File을 만들게 했다. 이는 출력을 Screen상에 출력을 하면 CPU Time의 손실이 많다. IDOL은

```
+------------------------------------------------------------+
|  +------------------------------------------------------+  |
|  |  ##########  ########     #####     ##               |  |
|  |      ##      ##     ##   ##   ##    ##               |  |
|  |      ##      ##      ##  ##   ##    ##               |  |
|  |      ##      ##      ##  ##   ##    ##               |  |
|  |      ##      ##     ##   ##   ##    ##               |  |
|  |  ##########  ########     #####     #########        |  |
|  +------------------------------------------------------+  |
|                                                            |
|          INTERACTIVE DESIGN OPTIMIZATION LIBRARY           |
|                                                            |
|                        VERSION 1.0                         |
|                                                            |
|                  CAD/CAE RESEARCH SECTION                  |
|                     TECHANICAL CENTER                      |
|                   DAEWOO MOTOR Co. Ltd.                    |
|                    (032) 520 - 2437                        |
+------------------------------------------------------------+
 == GO TO NEXT STEP : TYPE <RETURN> =====> __
```

```
#-----------------------------------------------------#
|                                                     |
|          CONFIGURATION OF THE PROBLEM               |
|                                                     |
| 1. NUMBERS OF VARIABLE                    =    0    |
| 2. NUMBERS OF EQUALITY CONSTRAINT         =    0    |
| 3. NUMBERS OF INEQUALITY CONSTRAINT       =    0    |
|                                                     |
| 4. GOTO NEXT STEP =======>                          |
|                                                     |
#-----------------------------------------------------#
======== SELECT the NUMBER ====>  1
```

```
#-----------------------------------------------------#
|                                                     |
|          CONFIGURATION OF THE PROBLEM               |
|                                                     |
| 1. NUMBERS OF VARIABLE                    =    4    |
| 2. NUMBERS OF EQUALITY CONSTRAINT         =    0    |
| 3. NUMBERS OF INEQUALITY CONSTRAINT       =    0    |
|                                                     |
| 4. GOTO NEXT STEP =======>                          |
|                                                     |
#-----------------------------------------------------#
======== SELECT the NUMBER ====>  3

====== INPUT the Value  ===>  3
```

```
#-----------------------------------------------------#
|                                                     |
|          CONFIGURATION OF THE PROBLEM               |
|                                                     |
| 1. NUMBERS OF VARIABLE                    =    4    |
| 2. NUMBERS OF EQUALITY CONSTRAINT         =    0    |
| 3. NUMBERS OF INEQUALITY CONSTRAINT       =    3    |
|                                                     |
| 4. GOTO NEXT STEP =======>                          |
|                                                     |
#-----------------------------------------------------#
======== SELECT the NUMBER ====>  4
```

```
#-----------------------------------------------------#
|          CONVERGENCE CRITERION for ALM.             |
|                                                     |
|  1.LAGRANGE MULTIPLIER     =  1.000E-05             |
|  2.EQUALITY CONSTRAINT     =  1.000E-05             |
|  3.INEQUALITY CONSTRAINT   =  1.000E-05             |
|  4.OBJECT FUNCTION         =  1.000E-05             |
|                                                     |
|  5. GOTO NEXT STEP                                  |
#-----------------------------------------------------#
======= SELECT NUMBER ========>  5
```

```
+--------------------------------------------------+
|      CONVERGENCE CRITERION for BFGS.             |
|                                                  |
|  1.NORM of GRADIENT     =  1.000E-05             |
|  2.LINE SEARCH RESTART =  1.000E-04              |
|                                                  |
|  3. GOTO NEXT STEP                               |
+--------------------------------------------------+
======= SELECT NUMBER ========>  3
```

```
+--------------------------------------------------+
|    CONVERGENCE CRITERION for SEARCH.             |
|                                                  |
|  1.MAXIMUM ITERATE          =  30                |
|  2.GOLDEN SECTION  REFINE =  1.000E-03           |
|                                                  |
|  3. GOTO NEXT STEP                               |
+--------------------------------------------------+
======= SELECT NUMBER ========>  3
```

```
+--------------------------------------------------+
|              OUTPUT OPTION                       |
|                                                  |
|       ON  : SCREEN                               |
|      OFF  : OUPUT FILE                           |
|                                                  |
|  1.CONSTRAINED OPT.     =  ON                    |
|  2.UNCONSTRAINED OPT.  =  ON                     |
|  3.LINE SEARCH          =  ON                    |
|                                                  |
|  4. GOTO NEXT STEP                               |
+--------------------------------------------------+
======== SELECT the NUMBER ====>  1
```

```
+--------------------------------------------------+
|            OUTPUT SELECTION                      |
|                                                  |
|  1.PRINT OUT OPTIMAL SOLUTION, ONLY.             |
|  2.PRINT THE CALCULATE PROCESS, ALL.             |
|  3.PRINT THE ITERATE NUMBER,ONLY (SCREEN).       |
|                                                  |
+--------------------------------------------------+
======== SELECT the NUMBER ====>  2
```

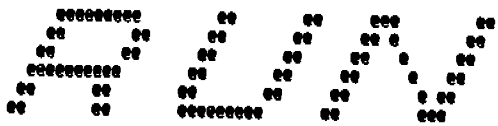
```
+--------------------------------------------------+
|              OUTPUT OPTION                       |
|                                                  |
|       ON  : SCREEN                               |
|      OFF  : OUPUT FILE                           |
|                                                  |
|  1.CONSTRAINED OPT.     = OFF                    |
|  2.UNCONSTRAINED OPT.  =  ON                     |
|  3.LINE SEARCH          =  ON                    |
|                                                  |
|  4. GOTO NEXT STEP                               |
+--------------------------------------------------+
======== SELECT the NUMBER ====>  _
```

```
+--------------------------------------------------+
|              OUTPUT OPTION                       |
|                                                  |
|       ON  : SCREEN                               |
|      OFF  : OUPUT FILE                           |
|                                                  |
|  1.CONSTRAINED OPT.     = OFF                    |
|  2.UNCONSTRAINED OPT.  = OFF                     |
|  3.LINE SEARCH          = OFF                    |
|                                                  |
|  4. GOTO NEXT STEP                               |
+--------------------------------------------------+
======== SELECT the NUMBER ====>  4
```

```
+--------------------------------------------------+
|            PENALTY FACTOR OPTION                 |
|                                                  |
| COMMENT : Initial value of Penalty factor        |
|           is calculated by Program.              |
|           Maximum value is multiple of           |
|           initial value.                         |
| EXAMPLE  :                                       |
|              Pmax = P0 * 10.0                    |
|                                                  |
|           1.Fixed Pmax    =     1.000E+11        |
|           2.Fixed P0      =     1.000E+00        |
|                                                  |
|           3.No Change, Goto NEXT                 |
+--------------------------------------------------+
==== SELECT THE NUMBER   =======> 3
```

```
+--------------------------------------------------------+
|            INITIAL GUESSING VALUE OPTION               |
|                                                        |
|              1. USER DEFINE.                           |
|              2. RANDOM SEARCH.                         |
|                                                        |
|  NOTE : If You have not Good Initial Guessing          |
|         Values, then Select  " 2 ".                    |
+--------------------------------------------------------+
=========== SELECT THE NUMBER =======>   1
```

```
==== INPUT the INITIAL DESIGN VARIABLE X( 1) ===> 0
==== INPUT the INITIAL DESIGN VARIABLE X( 2) ===> 0
==== INPUT the INITIAL DESIGN VARIABLE X( 3) ===> 0
==== INPUT the INITIAL DESIGN VARIABLE X( 4) ===> 0
```



```
<>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<>
<>                                           <>
<>       FIND  the OPTIMAL  SOLUTION  !      <>
<>                                           <>
<>   * OBJECT VALUE =     -4.49234E+01       <>
<>                                           <>
<>  Do You Want to do final Optimization ?   <>
<>                                           <>
<>>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<>

====== SELECT  (Y/N) ==============> n
```

Table 3. Example of PROGRAM Execution

Output File의 이름이 정해져 있다. Constrained Optimization Problem의 Output File 이름은 "CONST. OUT"이고 Unconstrained Multi-Variable Optimization Problem의 Output File 이름은 "UNCON. OUT"이고 Unconstrained One-Dimensional Optimization Problem의 이름은 "LINE. OUT"이다. 따라서 예제의 Output File은 CONST. OUT 이고 내용은 Table 4와 같다.

```
**********************************************
*                                            *
*        CONSTRAINED OPTIMIZATION            *
*  (Augmented Lagrange Multiplier Method)    *
*                                            *
*        FIND the OPTIMAL SOLUTION !         *
*                                            *
*           ITERATE =  3                     *
*                                            *
**********************************************
```

Number of Iterate	Constrained Optimization	= 3
	Unconstrained Optimization	= 7
	One Dimension Search	= 211
Stop Criterion	Constrained Optimization	= 0.100D-04
	Unconstrained Optimization	= 0.100D-02
	One Dimensional Search	= 0.100D-02

Object Function Value	= -4.49234E+01
Penalty Factor	= 1.00000E+02
Number of Design Variable	= 4
Number of Equality Constraint	= 0
Number of InEquality Constraint	= 3

	EQUALITY		INEQUALITY	
X value	Constraint	Multiplier	Constraint	Multiplier
3.6287E-01	0.000E-01	0.000E-01	3.1071E-09	1.981E+01
1.0752E+00	0.000E-01	0.000E-01	-2.1406E-01	0.000E-01
2.0692E+00	0.000E-01	0.000E-01	-3.4351E-01	0.000E-01
-6.5049E-01	0.000E-01	0.000E-01	0.0000E-01	0.000E-01

Table4. Content of CONST.OUT

APPENDIX B : Additional Numerical Test

1. Simple Beam Design[6]

[6]의 Test Problem으로 [6]에서의 초기점을 프로그램에 적용해본 결과 동일한 Optimum solution이 구해졌다.

Minimize : $F(\underline{x})=0.0624(x1+x2+x3+x4+x5)$

Subject to :

$$G1(\underline{x}) = 61/x1+37/x2+19/x3+7/x4+1/x5-1.0 \leqq 0.0$$

$$xi \geqq 0.0 \quad i=1,\cdots,5$$

초기값을 (5, 5, 5, 5, 5)로 했을 때, 최적값은 (6.0159, 5.3092, 4.4944, 3.5016, 2.1536)이고, $F=1.33996$이다. 총 반복횟수는 ALM=4, BFGS=21, LINE=469이며, Active한 구속조건은 $G1(-4.0158E-7)$이다.

2. Two Bar Truss Design[6]

문제1과 같이 [6]의 Test Problem이며, [6]에서의 초기점을 프로그램에 적용해 본 결과 동일한 Optimum solution이 구해졌다.

Minimize : $F(\underline{x})=x1\sqrt{1+x2}$

Subject to :

$$G1(\underline{x})=0.124(1+x2^2)\cdot(8/x1+1/x1\,x2)-1.0<0.0$$

$$G2(\underline{x})=0.124(1+x2^2)\cdot(8/x1-1/x1\,x2)-1.0<0.0$$

$$0.2 \leqq x1 \leqq 4.0$$

$$0.1 \leqq x2 \leqq 1.6$$

초기값을 (1.5, 0.5)로 했을 때, 최적값은 (1.4116, 0.37707)이고, $F=1.50865$이다. 총 반복횟수는 ALM=4, BFGS=12, LINE=304이며, Active한 구속조건은 $G1(2.6372E-9)$이다.

3. Bracken and Mc Cormick Problem[9]

SUMT IV를 위한 Test Problem으로, 일반적으로 Optimization Program의 상대평과를 위해서 많이 사용된다. 문제의 형태는 다음과 같다.

Minimize : $F(\underline{x})=x1^2+x2^2+5.0-4\,x1-2\,x2$

Subject to :

$$H1(\underline{x})=2\,x2-x1-1.0=0.0$$

$$G1(\underline{x})=1/4\,x1^2+x2^2-1.0 \leqq 0.0$$

$$0.0 \leqq xi \quad i=1,\ 2$$

초기값을 (2.0, 2.0)로 했을 때, 최적값은 (0.82288, 0.91144)이고, $F=1.39346$이다. 총 반복횟수는 ALM=5, BFGS=12, LINE=273이며 Active한 구속조건은 $H1(-9.986E-8)$과 $G1(3.1027E-8)$이다.

4. Welded Beam Design[3]

일반적으로 Optimization Program의 상대평가를 위해서 많이 사용되는 문제로, Local Minimum Point가 많고 수렴하기가 매우 까다롭다.

Minimize : $F(\underline{x})=1.10471\,x1^2\cdot x2+0.04811\,x3\,x4(14+x2)$

Subject to :

$$G1(\underline{x})=(D1+D2+D3/D4)^{**}(15/34)^{**}2-1.0=<0.0$$

$$G2(\underline{x})=1.0-x4^{*}\,x3^2/16.8=<0.0$$

$$G3(\underline{x})=1.0-x4/x1=<0.0$$

$$G4(\underline{x})=1.0-x3\cdot x4^2(1-0.02823\,x3)/0.09267=<0.0$$

$$G5(\underline{x})=1.0-x3^3\cdot x4/8.7808=<0.0$$

$0.125 =< x1 =< 10.0$

$0.1 =< xi =< 10.0 \quad i = 2, 3, 4$

여기서,

$D1 = 1.0/(2.0 * x1 * x2 * x2)$

$D2 = 3.0*(28+x2)/(x1*x1*x2*(x2*x2+3*(x1+x3)**2))$

$D3 = 4.5*(28+x2)**2*(x2*x2+(x1+x3)**2)$

$D4 = x1*x1*x2*x2(x2**2+3*(x1+x3)**2)**2$

초기값을 (1, 7, 4, 2)로 했을 때, 최적값은 (0.23756, 6.4567, 8.3025, 0.24429)이고, $F=$ 2.39834이다. 총 반복횟수는 ALM=4, BFGS =64, LINE=1212이며 Active한 구속조건은 $G1$(4.0796E-8)과 $G4$(7.6056E-8)이다.