544 大韓機械學會論文集 第 15 卷 第 2 號, pp. 544~556, 1991.

〈論 文〉

구속조건식이 있는 비선형 최적화 문제를 위한 ALM방법의 성능향상

김민수* · 김한성** · 이재원*** · 최동훈****

(1990년 9월 7일 접수)

Computational Enhancement to the Augmented Lagrange Multiplier Method for the Constrained Nonlinear Optimization Problems

Min-Soo Kim, Han-Sung Kim, Jae-Won Lee and Dong-Hoon Choi

Key Words: Optimization(최적화), Efficiency(효율성), Transformation Method(변환방법), Penalty Function(벌칙함수), Pseudo Function(假函數), Lagrang Multiplier(라그랑지 乘數), Dynamic Stopping Criterion(가변 수렴조건), Restarting(감소방향 벡터의 방향 수정)

Abstract

The optimization of many engineering design problems requires a nonlinear programming algorithm that is robust and efficient. A general-purpose nonlinear optimization program IDOL (Interactive Design Optimization Library) is developed based on the Augmented Lagrange Mulitiplier (ALM) method. The ideas of selecting a good initial design point, using resonable initial values for Lagrange multipliers, constraints scaling, descent vector restarting, and dynamic stopping criterion are employed for computational enhancement to the ALM method. A descent vector is determined by using the Broydon-Fletcher-Goldfarb-Shanno (BFGS) method. For line search, the Incremental-Search method is first used to find bounds on the solution, then the bounds are reduced by the Golden Section method, and finally a cubic polynomial approximation techanique is applied to locate the next design point. Seven typical test problems are solved to show IDOL efficient and robust.

기 호 설 명

$A(X)$: 구속조건이 있는 문제를 구속조건이 없는 문제로 변환한 가함수(pseudo function)

d_k : 가함수를 줄일 수 있는 감소 방향 벡터

$F(X)$: 일반적인 비선형 최적화 문제의 목적함수

$G_j(X)$: j번째 비선형 부등식 구속조건 함수

$H_i(X)$: i번째 비선형 등식 구속조건 함수

k : 최적화 알고리즘의 반복횟수(상부첨자) k번째 설계변수(하부첨자)

ℓ : 부등식 구속조건 함수의 갯수

m : 등식 구속조건 함수의 갯수

n : 설계변수의 갯수

r : 가함수에서 구속조건의 가중치를 반영하기 위한 벌칙 계수(penalty parameter)

$U(X)$: 구속조건이 없는 최적화 문제의 목적함수

X : 최적화 알고리즘에서의 설계변수

* 정회원, 대우자동차 기술연구소
** 한양대학교 대학원
*** 정회원, 대신전산센터 시스템 관리부
**** 정회원, 한양대학교 공과대학 기계설계학과

α^* : 선탐색(line search)을 통하여 구한 설계 변수의 이동거리

λ_i : i번째 등식 구속조건을 위한 라그랑지 乘數(lagrange multiplier)

μ_j : j번째 부등식 구속조건을 위한 라그랑지 乘數

ζ_i : i번째 등식 구속조건의 스케일링 인자(scaling factor)

η_j : j번째 부등식 구속조건의 스케일링 인자(scaling factor)

1. 서 론

1960년 Schmit[1]가 自動 設計를 위하여 最適化 技法을 도입한 이래로 컴퓨터 하드웨어와 最適化 技法의 발달에 힘입어 最適設計 분야는 급속히 발달하고 있다. 1987년 Waren등[2]이 행한 조사에 의하면 18종류의 最適化 技法 소프트웨어가 商業用으로 개발되어 있다. 이들 소프트웨어들은 해당분야의 해석 소프트웨어와 결합되어 航空, 船舶, 自動車, 플랜트 엔지니어링 분야등에서 重量減少, 原價節減과 性能向上등을 위한 設計記述의 道具로 널리 적용되고 있다[3].

본 연구에서는 拘束條件式이 있는 非線型最適化 문제의 최적해를 구할 수 있는 범용 最適化 技法 소프트 웨어 IDOL(Interactive Design Optimization Library)을 개발하고자 한다. 사용하는 알고리즘은 非線型 最適化 問題의 해결에 많이 사용되는 方法중의 하나인 ALM(Augmented Lagrange Multiplier)方法에 安定性과 收斂性을 높일 수 있는 여러가지 技法들을 도입하였다.

본 論文에서는 먼저, ALM의 일반적인 理論과 이를 보강한 여러가지 技法을 기술하고, IDOL의 構造와 使用方法을 알아본다. 또한 각종 例題에 적용하여 IDOL의 有用性을 보이려 한다.

2. ALM 方法

拘束條件式이 있는 非線型 최적화 문제를 풀기 위한 방법은 크게 구속조건식을 직접 취급하는 직접 방법(direct method)과 구속조건이 있는 문제를 구속조건이 없는 문제로 변환한 후, 잘 발달된 구속조건이 없는 문제를 푸는 최적화 기법(unconstrained optimization method)을 적용하는 변환 방법(transformation method)으로 나눌 수 있다.

변환 방법은 여러가지 형태의 벌칙함수(penalty function)를 적용하는 SUMT(Sequential Unconstrained Minimization Technique)가 주종을 이룬다. ALM은 이와같은 SUMT에 최적점 조건(optimality condition)을 도입하여 信賴性과 效率性을 높인 變換 方法이다.

2.1 ALM 알고리즘

일반적인 최적화 문제를 다음과 같은 형태로 기술하여 ALM 알고리즘을 기술하고자 한다.

$$\text{Minimize } F(X) \tag{1.a}$$

$$\text{Subject to } H_i(X)=0.0 \quad i=1, \cdots, m \tag{1.b}$$

$$G_j(X) \le 0.0 \quad j=1, \cdots, \ell \tag{1.c}$$

$$X_k^l \le X_k \le X_k^u \quad k=1, \cdots, n \tag{1.d}$$

위의 최적화 문제에 라그랑지 乘數와 벌칙계수(penalty parameter)를 도입하여 아래와 같은 假函數(pseudo function)를 구성한다.

$$\text{Minimize } A(X, \lambda_i, \mu, r)$$

$$=F(X)+\sum_{i=1}^{m}\{\lambda_i \cdot H_i(X)+r \cdot H_i(X)^2\}$$

$$+\sum_{j=1}^{\ell}\{\mu_j \cdot \Psi_j + r \cdot \Psi_j^2\} \tag{2}$$

여기서, $\Psi_j=\text{maximum}\{G_j(X), \ -\mu_j/(2 \cdot r)\}$이다. 위의 假函數는 구속조건이 없는 최적화 문제이다.

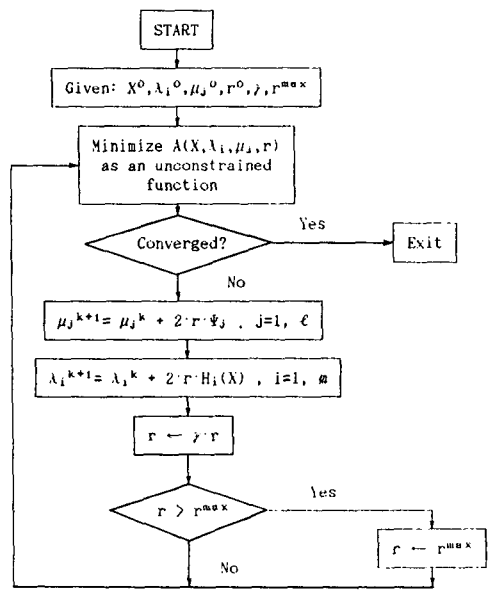

Fig. 1 Flow chart for the ALM method

구속조건이 없는 최적화 문제의 해를 구하는 알고리즘은 다음 절에서 설명하기로 하고 본 절에서는 Fig. 1을 통하여 ALM알고리즘을 간략하게 살펴보기로 한다. 초기단계에서 설계변수 벡터의 초기값 (X^0), 라그랑지 乘數의 초기값(λ_i^0, μ_j^0), 벌칙계수의 초기값(r^0), 벌칙계수의 곱수(γ)와 벌칙계수의 한계값(r^{max})을 기본적으로 정의한다. 이 값들을 기초로하여 식 (2)의 假函數가 최소가 되는 새로운 설계점을 구하고, 그 설계점에서 수렴의 여부를 판정한다. 수렴 조건이 만족되지 않는 경우는 라그랑지 乘數와 벌칙계수값을 개선하여 재실행 시킨다. ALM방법은, 일반적인 변환방법에 비하여, 벌칙계수값에 상대적으로 덜 민감하기 때문에 벌칙계수값을 無限大(∞)까지 증가시킬 필요가 없으며, 라그랑지 乘數를 개선함에 따라 수렴이 가속화된다. 라그랑지 乘數의 개선 규칙은 아래와 같다[4].

$$\lambda_i^{k+1}=\lambda_i^k+2\cdot r\cdot H_i(X)$$
$$\mu_j^{k+1}=\mu_j^k+2\cdot r\cdot \Psi_j \tag{3}$$

여기서, k는 ALM의 반복 횟수이다.

2.2 拘束條件이 없는 最適化 問題를 푸는 일반적인 節次

구속조건이 없는 최적화 문제를 다음과 같이 기술하자.

$$\text{Minimize } U(X) \tag{4}$$

식 (4)의 최적해를 구하기 위한 일반적인 절차는 아래와 같다.

단계 0 : 초기 설계점을 가정한다.

단계 1 : 현 설계점에서 목적함수 값을 줄일 수 있는 감소방향 벡터(descent vector) d_k를 구한다.

단계 2 : 감소방향 벡터를 따라서 목적함수 값이 최소가 되는 거리 α^*를 구한다.

단계 3 : 단계 2에서 구한 α^*를 이용하여 새로운 설계변수를 구한다.

$$X^{k+1}=X^k+\alpha^*\cdot d_k \tag{5}$$

단계 4 : 새로운 설계점 X^{k+1}에서 수렴조건을 만족하면 수렴한 것으로 하고, 그렇지 않으면 단계 1로 간다.

단계 1의 감소방향 벡터를 결정하는 방법에는 많은 알고리즘이 있다. 본 연구에서는 가장 효율적인 방법중의 하나인 BFGS(Broyden-Fletcher-Goldfarb-Shanno)방법[7]을 이용한다. 또 감소방향에서의 최적거리 α^*를 결정하기 위하여 다음과 같은 안정성과 수렴성이 우수한 선탐색(line search) 알고리즘을 이용한다[4].

단계 0 : 증분 탐색(incremental search)방법을 이용하여 해가 포함되어 있는 개략적인 구간을 정한다.

단계 1 : 황금 분활(golden section)방법을 적용하여 단계 0에서 구한 구간을 줄인다.

단계 2 : 줄여진 구간에서 3차함수로 근사화 한 후, 이 함수의 최소값을 이용하여 최적거리 α^*를 구한다.

3. ALM 方法의 效率 向上을 위한 技法

앞 장에서 기술한 일반적인 ALM 방법에 구속조건식의 스케일링(scaling), 설계변수의 초기점 선정, 라그랑지 乘數의 초기값 선택, 가변 수렴조건(dynamic stopping criterion)과 감소방향 벡터의 방향 수정(restaring)을 적용한다.

3.1 拘束函數의 스케일링(Scaling)

拘束函數들간의 함수값 또는 구배값의 크기 차이가 심하게 되면, 최적화 알고리즘에서 각 구속함수들의 重要度가 다르게 반영되므로 정확한 최적해에 수렴하기 힘들다. IDOL은 이를 보완하기 위하여 구속함수의 구배값과 목적함수의 구배값의 크기가 같아지도록 하는 스케일링 方法을 적용한다. 초기점에서 정한 스케일링 계수를 비선형성이 심한 최적화 문제에 적용하면, 최적화가 진행됨에 따라 수렴에 나쁜 영향을 줄 수 있다. 따라서 IDOL에서는 ALM의 반복횟수에 따라 가함수값의 크기 변화가 심하면 非線型性이 심한 것으로 간주하고, 다시 스케일링 한다. 스케일링 계수를 식 (2)에 적용하면 아래와 같다.

$$\begin{aligned}\text{Minimize } A(X, \lambda_i, \mu_j, r) &= F(X)+\sum_{i=1}^{m}\{\lambda_i\cdot H_i(X)+r\cdot\zeta_i\cdot H_i(X)^2\}\\ &+\sum_{j=1}^{l}\{\mu_j\cdot\Psi_j+r\cdot\eta_j\cdot\Psi_j^2\}\end{aligned} \tag{6}$$

여기서,

$$\zeta_1=\frac{\|\nabla F(X^0)\|}{\|\nabla H_i(X^0)\|} \qquad i=1,\cdots,m$$

$$\eta_j=\frac{\|\nabla F(X^0)\|}{\|\nabla G_j(X^0)\|} \qquad j=1,\cdots,\ell$$

이다.

3.2 설계변수의 초기값 선정

비선형 최적화 문제에서 초기값의 위치는 최적점을 구하는 데 매우 큰 영향을 끼친다. IDOL에서는 설계변수의 초기값을 사용자가 직접 정의하거나, 초기값 선정이 여의치 않은 경우는 설계변수의 상한값과 하한값을 정의해 주면 IDOL은 랜덤 탐색(random search)[5]을 통하여 최적의 초기점을 자동적으로 선정한다. 랜덤 탐색의 횟수는 설계변수의 갯수를 고려하여 프로그램에서 자동으로 정할 수도 있고 사용자가 정하여 줄 수도 있다.

3.3 라그랑지 乘數의 초기치 選定

ALM 알고리즘은 라그랑지 乘數값에 따라 수렴속도가 많은 영향을 받는다. 따라서, 초기의 라그랑지 乘數값의 적절한 선정은 빠른수렴을 위하여 매우 중요하다. IDOL에서는 Kuhn-Tucker의 최적점 조건[4]을 이용하여 다음과 같이 초기값을 선정하다.

Table 1 Initial choice of lagrange multiplier

Condition	Lagrange multiplier
$\nabla F \cdot \nabla H_i \leq 0.0$	$\lambda_i = 1.0$
$\nabla F \cdot \nabla H_i > 0.0$	$\lambda_i = -1.0$
$G_j(X^0) \leq 0.0$	$\mu_j = 0.0$
$\nabla F \cdot \nabla G_j > 0.0$	$\mu_j = 0.0$
$\nabla F \cdot \nabla G_j \leq 0.0$	$\mu_j = 1.0$

3.4 可變收斂條件의 채택

앞에서 기술한 바와 같이 ALM의 k번째 실행에서 구한 최적해는 $k+1$번째 실행을 위한 초기점으로 사용된다. 따라서 ALM의 초기 실행 단계에서는 정확한 해를 구하기 위하여 많은 함수계산을 할 필요가 없다. ALM에서 해를 구하기 위한 함수 계산량은 식 (6)의 구속조건이 없는 假函數의 최적해를 구하는 알고리즘(본 연구에서는 BFGS)의 수렴조건과 선탐색(line search)의 수렴조건에 따라서 많이 좌우된다. 일반적으로 선탐색(line search)의 수렴조건이 최적해의 정확도에 상대적으로 많은 영향을 끼치므로 수렴조건을 느슨하게 잡을 수 없다. 따라서 IDOL에서는 BFGS의 수렴조건을 가변적으로 적용함으로써 함수 계산량을 줄인다. IDOL에서 채택한 可變 수렴조건은 0.1에서 부터 사용자가 정의한 값까지 ALM이 반복됨에 따라 점차적으로 줄여 나간다. 그러나, 이와 같은 수렴조건을 수렴이 쉬운 문제(예제 3, 4)에 적용하면 오히려 함수계산량이 증가할 수도 있다. 따라서 IDOL에서는 ALM의 첫번째 실행에서 수렴한 BFGS의 반복횟수가 설계변수의 갯수보다 적거나 같을 때는, BFGS의 수렴조건을 보다 엄밀하게 줄여서 재실행을 한다. 본 연구의 가변 수렴조건을 적용한 결과에 따르면 동일한 최적해를 구하는데 소요된 함수계산량이 크게 줄어 들었다.

3.5 減少方向 벡터의 방향 수정(Restarting)

減少方向을 따라서 식 (6)의 가함수가 최소가 되는 거리 α^*를 구할때 감소방향 벡터의 크기 변화가 심하면 수치해석적으로 불안정해진다. 따라서, 감소방향 벡터는 아래와 같이 정규화(normalize)한다.

$$d_k = \frac{d_k}{\|d_k\|} \tag{7}$$

IDOL에서는 식 (6)의 최적해를 구하기 위하여 채택한 BFGS알고리즘은 Quasi-Newton 방법의 일종이다. Quasi-Newton방법은 Hessian의 역행렬을 근사적으로 구하기 때문에 알고리즘이 반복됨에 따라 감소방향이 나빠질 수가 있다. 따라서, 적절한 때에 감소방향을 수정할 필요가 있다. IDOL에서는 가함수의 구배와 감소방향 벡터가 이루는 내각이 직각보다 작거나, 선탐색(line search)이 향상된 해를 구하는데 실패했을 때 감소방향 벡터를 급강하 기울기(steepest descent) 벡터로 재 정의한다[4].

4. 프로그램의 구조

IDOL은 크게 MAIN DRIVER, OPTIMIZATION LIBRARY, 다른 프로그램과 연결하여 작업하기 위한 SUBMAIN$와 사용자가 정의하여야 하는 목적함수와 구속함수를 위한 OBJECT$, SUBJECT$로 구성되어 있다. IDOL은 사용자의 편의에 따라 대화식 작업과 배치(batch) 작업을 수행할 수 있고 IBM-PC 및 메인 프레임(VAX... 등)에서도 적용 가능하다[6]. 전체적인 프로그램의 구성은 Fig. 2와 같다.

IDOL을 구성하는 부함수들의 기능에 대해서 알아보자.

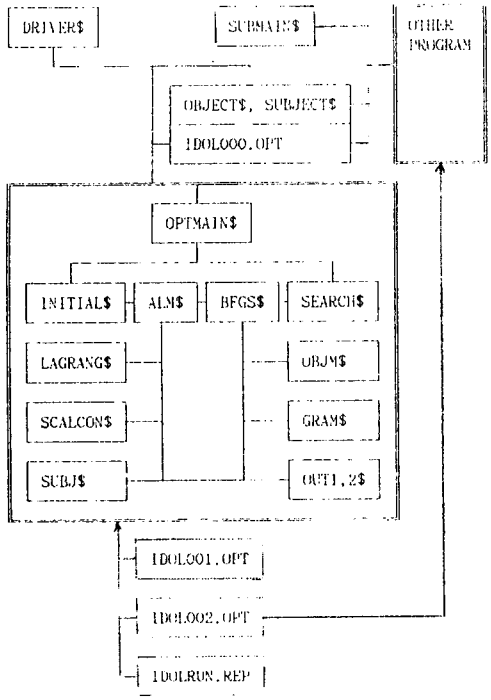

Fig. 2 Simple flow chart for the IDOL

OPTMAIN$:
문제의 형태에 따라서 적용할 방법(ALM, BFGS, SEARCH)을 자동으로 구분하고, 사용하는 수렴조건, 출력을 위한 선택과 초기 설계변수값의 입력방법 선택을 한다. 이러한 작업을 대화식으로 할 수도 있고 배치(batch)식으로도 할 수 있다.

INITIAL$: OPTMAIN$에서 선택한 방법에 따라 초기값을 사용자가 직접 입력하거나 랜덤탐색(random search)를 위한 상한값과 하한값을 입력한다.

RAN$: 0.0과 1.0사이의 난수(random number)를 만든다.

ALM$: ALM의 主 프로그램으로 라그랑지 乘數와 벌칙계수를 개선한다.

SUBJ$: 목적함수와 구속함수의 구배값을 구한다.

SCALCON$: 구속함수의 스케일링 인자(scaling factor)를 계산한다

LAGRANGE$: 라그랑지 乘數의 초기값을 선정한다.

BFGS$: BFGS의 主 프로그램이다.

OBJM$: 구속조건이 없는 문제의 목적함수와 ALM$의 가함수를 구성한다.

GRAM$: OBJM$의 구배를 계산한다.

SEARCH$: 선탐색(line search)모듈로 유니모달(unimodal) 함수가 되도록 개략적인 범위를 정하고, 황금 분활법으로 개선한 후에 3차함수 근사화 기법을 적용한다.

OBJL$: 선탐색(line search)를 위한 목적함수를 구성한다.

OUT1,2$: ALM$과 BFGS$의 계산결과를 사용자가 알아보기 쉬운 형식으로 출력한다.

이상이 모듈으로 구성된 IDOL은 독자적인 프로그램으로 사용할 수도 있지만, 다른 프로그램과 접속(interface)을 위해서 SUBMAIN$이란 부함수를 제공한다. SUBMAIN$는 DRIVER$ 모듈의 기능과 같다. IDOL을 실행하면 몇개의 부가적인 화일이 생성되는데 이 화일의 역할을 알아보자.

IDOLOUT.C : ALM$의 결과가 일정한 형식으로 출력된 화일이다.

IDOLOUT.U : BFGS$의 결과가 일정한 형식으로 출력된 화일이다.

IDOL000.OPT : OPTMAIN$에서 대화식으로 작업한 데이터와 설계변수의 초기값이 저장되어 있다. 따라서 사용자는 이 화일의 내용을 변경해서 Batch작업을 실행하거나 다른 프로그램과 접속(interface)할 때 사용 가능하다.

IDOL001.OPT : IDOL에서 사용하는 구속함수의 스케일링 인자값과 설계변수의 스케일링을 위한 값이 저장된다. 설계 변수값의 차이가 심해서 정확한 해를 구하지 못하였을 때는 이 화일의 설계변수 스케일링 인자를 이용해서 문제의 구성을 다시한다. IDOL은 설계변수의 스케일링에 덜 민감하기 때문에 대부분의 경우는 사용하지 않아도 될 것이다.

IDOL002.OPT : IDOL과 다른 프로그램을 접속해서 사용할 때는 최적값이 특정 형식으로 출력되면 이용하기가 힘들다. 이를 보완하기 위해서 無型式으로 최적값만 출력한다.

IDOLRUN.REP : 부록 1과 같이 수렴과정을 표 형식으로 출력한다.

5. 소형 전산기를 이용한 수학적 검증

IDOL을 다음과 같이 예제들에 적용하여 본다.

예제들은 각종 자료에서 문제의 특성에 따라 다양하게 선정하였다. IDOL은 사용자의 편의에 따라 목적함수와 구속함수의 구배값을 사용자가 기술하여 줄 수도 있고 자동적으로 중앙 차분법을 이용하여 구할 수도 있으나, IDOL의 안정성을 보이기 위하여 중앙 차분법을 이용하여 구했으며 모든 예제들에서 사용된 수렴조건은 1.0×10^{-5}로 하였다. IDOL과 다른 프로그램들의 결과 비교는 아래와 같고 자세한 결과는 부록 1에 수록하였다. 사용기종은 IBM-PC 386 호환기종이다.

〔예제 1〕 ROSEN & SUZUKI 문제[8,9]

국부 최소점을 많이 포함하는 수학적인 문제다. 이 문제는 일반적으로 최적화 프로그램/알고리즘의 평가를 위하여 많이 사용된다. IDOL과 참고문헌 (8)과 (9)의 결과 비교는 Table 2과 같다.

Table 2 Summary of results for problem 1

Program	Algorithm	Final cost
CONMIN	Feasible direction	56.15
OPTDYN	Feasible direction	56.001
LINRM	Recursive quadratic programming	56.00
GRP-UI	Gradient projection	56.00
SUMT	Exterior penalty	56.00
IDOL	Augmented lagrange multiplier	56.00

〔예제 2〕 WELDED BEAM DESIGN[9]

최적화 프로그램의 상대 평가를 위하여 많이 사용되는 문제다. 설계목적은 鎔接費의 최소화이고, 설계를 위한 구속조건은 鎔接應力, 棒의 굽힘응력, 棒의 좌굴, 棒의 처짐과 설계변수의 크기 제한이다. 설계변수는 Fig. 3에 도시한 용접 사이즈(h), 용접길이(l), 棒의 높이(t), 棒의 폭(b)이다. 적용 결과는 Table 3과 같다.

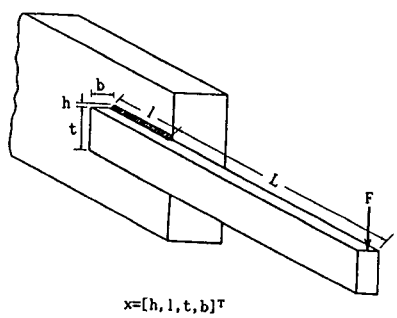

Fig. 3 Welded beam

Table 3 Summary of results for problem 2

	IDOL	BIAS[9]
Initial value	$F=15.8154$ $h=1.0000$ $l=7.000$ $t=4.0000$ $b=2.0000$	$F=15.8154$ $h=1.0000$ $l=7.0000$ $t=4.0000$ $b=2.0000$
Optimum value	$F=2.3809$ $h=0.2443$ $l=6.2181$ $t=8.2916$ $b=0.2443$	$F=2.3811$ $h=0.2444$ $l=6.2187$ $t=8.2915$ $b=0.2444$
Iterated number	ALM = 5 BFGS = 49	MOM = 6 DFP = 76

BIAS[9]는 MOM(Method of Multiplier)과 DFP (Davidon-Fletcher-Powell)을 적용한 변환 방법의 일종이다. Table 3의 상대비교는 BIAS의 결과 〔참고문헌(9)의 pp. 247~249〕를 이용하였다. Table 3의 결과에서 볼 수 있는 바와 같이 IDOL이 BIAS보다 적은 반복 횟수만에 좋은 최적해로 수렴하였다.

〔예제 3〕 THROUGH CIRCULATION DRIVER DESIGN[8]

이송 순환계(through circulation system)의 생산 비율을 최대로 하는 유체속도, X_1과 베드 깊이 (bed depth), X_2를 구하는 문제로 문제의 구성상 설계변수의 크기 차이가 심하고, 가함수(pseudo function)의 Hessian 행렬의 찌그러짐(distorison)

Table 4 Summary of results for problem 3

	Reference(8)	Scaling	Non-scaling
F	−1.5370E+02	−1.5371E+02	−1.5371E+02
X_1	3.1766E+04	3.1766E+00	3.1766E+04
X_2	3.4200E+00	3.4207E+00	3.4207E+00

이 매우 심하기 때문에 최적해로의 수렴이 까다롭다. 참고문헌 (8)의 결과와 IDOL에서 설계변수 X_1을 $10000X_1$으로 스케일링 하여 구한 최적해와 스케일링 하지않고 구한 최적해의 비교는 Table 4와 같다.

Table 4의 결과를 볼때 IDOL은 설계변수의 크기 차이에 의한 假函數의 찌그러짐에 영향을 받지 않는 것을 알 수 있다. 따라서 IDOL은 스케일링을 하지않고 최적해를 구하여도 안정성이 있다는 것을 알 수 있다.

〔예제 4〕 VIBRATION ABSORBER DESIGN[13]

Fig. 4와 같이 主시스템(질량 M_1)의 진동을 최소화하는 副시스템(질량 m_2)의 감쇠비 ζ_2와 고유진동수비 Ωr를 정하는 문제이다. 초기점에 대한 자료가 없기 때문에 IDOL에서는 랜덤 탐색(random search)을 이용했다. 계산결과는 Table 5와 같고, 질량 M_1의 감쇠비 ζ_1가 0.2일 때 IDOL에서 구한 목적함수의 최소값은 0.77819이고 설계변수들(ζ_2, Ψr)의 최적해는 각각 0.1938, 0.09352로 참고문헌 (13)의 목적함수값 2.317보다 약 30% 향상되었다. ζ_1의 나머지 값에서는 동일한 최적해가 구해졌다.

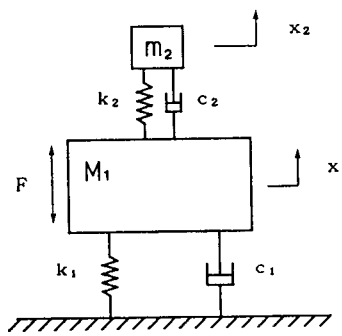

Fig. 4 Vibration absorber

Table 5 Summary of results for problem 4

Ω	0.2	0.6	1.0	1.4	1.8
Ωr	0.1938	0.5661	0.9920	1.460	1.872
ζ_2	0.09352	0.07574	0.1009	0.09755	0.9824
F	0.77819	0.7778	0.71666	0.28038	0.14189

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

〔예제 5〕 TENSION-COMPRESSION SPRING DESIGN[14]

Fig. 5에 도시한 바와 같은 처짐량, 전단응력, 서지 진동수(surge frequency), 외경과 설계변수의 허용 범위에 따른 구속조건이 있는 引張/壓縮 스프링의 질량을 최소화하는 스프링의 감긴 횟수(n), 코일의 직경(D)과 와이어(wire)의 직경(d)를 구하는 문제이다. 초기치를 $n=3$, $D=2$, $d=1$로 했을 때, 다른 프로그램과 IDOL의 최적해 비교는 Table 6과 같다.

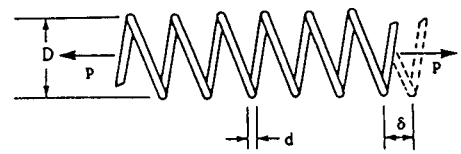

Fig. 5 Tension/compression spring

Table 6 Summary of results for problem 5

Program	Optimum value $F(n,\ D,\ d)$
CONMIN	Failed
OPTDYN	0.01543(2.959, 0.7488, 0.0644)
LINRM	0.01543(2.945, 0.7502, 0.0645)
GRP-UI	Failed
SUMT	0.01470(13.36, 0.3455, 0.0526)
IDOL	0.01267(11.29, 0.3569, 0.0517)

Table 6에서 볼 수 있는 바와 같이 IDOL이 다른 상용 프로그램에 비하여 우수한 결과를 주었다.

〔예제 6〕 THREE BAR TRUSS DESIGN[14]

Fig. 6과 같이 세개의 荷重條件〔참고문헌 (14)의 p. 175〕下에, 처짐량, 응력, 고유진동수, 좌굴 하

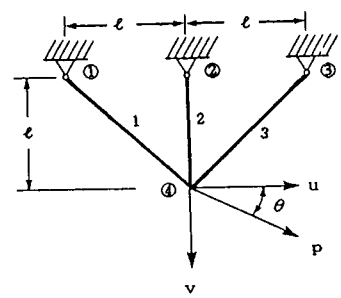

Fig. 6 Three bar truss design

중, 설계변수의 한계 범위등에 종속된 3개 棒의 무게를 최소화하는 단면적을 구하는 문제이다. 탄성계수 (E)는 1.0×10^5, 밀도(ρ)는 0.1ibs/in, σ_1과 σ_3의 허용응력은 5000psi, σ_2의 허용응력은 20,000 psi, δx와 δy의 허용 처짐량은 0.005in이다. 단면적들(A_1, A_2, A_3)의 초기치를(10, 5, 5)로 선정했을 때 다른 프로그램과 IDOL의 최적해 비교는 Table 7과 같다.

Table 7 Summary of results for problem 6

Program	Optimum value $W(n,\ D,\ d)$
CONMIN	21.047(8.981, 2.289, 4.283)
OPTDYN	20.542(8.914, 1.934, 4.245)
LINRM	20.542(8.910, 1.930, 4.251)
GRP-UI	20.542(8.911, 1.931, 4.249)
SUMT	20.525(8.902, 1.930, 4.248)
IDOL	20.543(8.910, 1.929, 4.251)

Table 7의 결과에서 IDOL이 다른 프로그램들보다 우수하거나 비슷한 결과를 주었음을 알 수 있다.

〔예제 7〕 Shell (Colville) Primal 문제[21]

Hock, W.와 Schittkowski, K.(1980)의 비선형 프로그래밍 코드(Nonlinear Programing Codes)를 위한 검증 예제중 하나로 일반적으로 최적화 프로그램/알고리즘의 평가를 위하여 많이 사용된다. 참고문헌과 동일한 초기점(0.1, 0.1, 0.1, 0.1, 0.1,)을 IDOL에 적용하여 구한 결과는 Table 8과 같다.

Table 8 Summary of results for problem 7

	Reference(21)	IDOL
F	−32.35	−32.35
X_1	0.3	0.30001
X_2	0.3335	0.33347
X_3	0.4	0.40001
X_4	0.4283	0.42830
X_5	0.2240	0.22395

Table 8의 결과에서 IDOL이 참고문헌 (21)과 동일한 결과를 얻었음을 알 수 있다.

6. 결 론

일반적인 ALM방법에 收斂性, 效率性 그리고 安定性을 높이기 위하여 본 연구의 여러가지 技法(초기값의 선택, 구속함수들의 스케일링(scaling), 감소 방향 벡터의 방향수정, 可變收斂條件)을 적용하여 프로그램 IDOL을 개발하였다. 본 프로그램은 각종 대표적인 예제 들을 통하여 일반적으로 효율적임을 알 수 있다. 따라서, 본 연구에서 개발한 프로그램은 다양한 工業分野의 最適設計를 위한 道具로서 편리하게 적용이 가능할 것으로 본다.

참 고 문 헌

(1) Schmit, L.A., 1960, "Structural Design by Systematic Synthesis", Proc. 2nd Conference on Electronic Computation, ASCE, New York, pp. 105~122.

(2) Waren, A.D., Hung, M.S. and Lasdon, L.S., 1987, "The Status of Nonlinear Programming Software", An Update Operations Research, Vol.35, No.4.

(3) Vanderplaats, G.N., 1986, "Trends in Structural Optimization : Some Consideration in Using Standard Finite Element Software SAE 860801.

(4) Vanderplaats, G.N., 1984, Numerical Optimization Techaniques for Engineering Design, McGraw Hill.

(5) Johnson, R.C., 1980, Optimal Design of Mechanical Elements, John Wiley & Sons.

(6) 김민수, 김한성, 이재원, 최동훈, 1990, "IDOL v. 1.5 User's Guide", Optimal Design Lab. Mechanical Design & Production Engineering in Han Yang University.

(7) Luenberg, D.G., 1984, "Linear and Nonlinear Programming, Addison-Wesley.

(8) Reklaitis, G.V., 1983, Ravindren, A. and Ragsdell, K.M. Engineering Optimization Method and Application, John Wiley & Sons.

(9) Rosen, J. B. and Suzuki, S., 1965, "Construction of Nonlinear Programming Test Problems", Communication of Association for Computing Machinery, Vol.8, p. 113.

(10) Miele et al., 1972, "Journal of Optimization Theory and Applications", Vol.10, No.1, p .24.

(11) Luus, R. AICHE J. 20, p. 608, 1974.

(12) Scott, A.B., 1987, "Generalized Geometric Programming With many Equality Constraints", International Journal for Numerical Methods in Engineer-

ing, Vol.24, p. 725.

(13) Nader, D.E., 1987, "Optimum Design of Vibration Absorbers", Techanical Briefs, Trans. of the ASME, Vol.109.

(14) Belegundu, A.S. and Arora, J.S., 1985, "A Study of Mathematical Programming Methods for Structural Optimization Part Ⅰ, Ⅱ", International Journal for Numerical Methods in Engineering, Vol.21, pp. 1583~1623.

(15) Haug, E.J. and Arora, J.S., 1979, Applied Optimal Design, John Wiley & Sons.

(16) Gill, P.E. and Murry, W., 1981, Practical Optimization Academic Press.

(17) Sandgren, E. and Ragsdell, K.M., 1980, "The Utility of Nonlinear Programming Algorithm: A Comparative Study Part Ⅰ, Ⅱ", Trans. of the ASME (J. of Mechanical Design), pp. 540~551.

(18) Lasdon, L.S. and Beck, P.O., 1981, "Scaling Nonlinear Programs", Operations Research Letters, pp. 6~9.

(19) Parkinson, A. and Wilson, M., 1988, "Dvelopment of A Hybrid SQP-GRG Algorithm for Constrained Nonlinear Programming", Trans. of the ASME (J. of Mechanisms, Transmission and Automation in Design), p. 308.

(20) Belegundu, A.D., 1988, "Probilistic Optimal Design Using Second Moment Criteria", Trans. of the ASME (J. of Mechanism, Transmissions and Automation in Design), p. 324.

(21) Lim, O.K. and Arora, J.S., 1985, "An RQP Algorithm with Active Set Strategy for Optimum Design", Technical Report No. ODL. 85.13.

(22) Moris, A.J., 1980, Fundations for Structural Optimization: An Unified Approach, John Wiley & Sons.

Table A1 Interation history for problem 1

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	1.00000E+02	IN THE FEASIBLE REGION	NA
1	7	71	5.47978E+01	Gj(1) = 4.98262E−02 Gj(3) = 8.53875E−02	9.26285E+00 8.10054E+00
2	13	102	5.59864E+01	Gj(1) = −5.90492E−04 Gj(3) = 1.84771E−03	8.16511E+00 9.85343E+00
3	21	159	5.59999E+01	Gj(1) = −8.79870E−06 Gj(3) = 1.53030E−05	8.00154E+00 9.99861E+00
4	22	162	5.60000E+01	Gj(1) = −1.74615E−05 Gj(3) = 7.45085E−07	7.29843E+00 1.00226E+01

NIA : No. of iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation

Table A2 Iteration history for problem 2

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	1.58155E+01	IN THE FEASIBLE REGION	NA

1	24	169	2.00757E+00	Gj(1) = 6.20498E−03 Gj(2) = 3.86013E−02 Gj(3) = 1.41247E−02 Gj(4) = 6.74318E−01 Gj(11) = 3.62049E−03	6.49702E−01 9.27505E−01 8.17699E−01 4.69994E−0. 1.17167E−01
2	38	262	2.38474E+00	Gj(1) = 1.99890E−03 Gj(3) = −7.88922E−03 Gj(4) = 9.15733E−03	6.85904E−01 6.15614E−01 3.88910E−01
3	47	313	2.38097E+00	Gj(1) = 1.81063E−05 Gj(2) = −1.90646E−05 Gj(3) = −7.63392E−05 Gj(4) = 9.22999E−07	6.89184E−01 2.06972E−01 5.96059E−01 3.89255E−01
4	48	315	2.38097E+00	Gj(1) = 1.81063E−05 Gj(2) = −1.90646E−05 Gj(3) = −7.63392E−05 Gj(4) = 9.22999E−07	6.13179E−01 2.43337E−01 5.48744E−01 7.13478E−01
5	49	317	2.38097E+00	Gj(1) = −6.48129E−05 Gj(2) = −2.60374E−05 Gj(3) = −4.29675E−05 Gj(4) = −1.99370E−05	0.00000E−01 0.00000E−01 0.00000E−01 4.29598E−01

NIA : No. of iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation

Table A3-1 Iteration history for ploblem 3 (Scaling)

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	−1.22326E+02	IN THE FEASIBLE REGION	NA
1	5	29	−1.66632E+02	Gj(1) = 2.29894E−01 Gj(2) = 3.40075E−02	5.17300E+01 7.14872E+00
2	11	63	−1.53899E+02	Gj(1) = 3.09669E−03 Gj(2) = 3.57172E−04	5.86980E+01 7.89953E+00
3	16	98	−1.53715E+02	Gj(1) = −2.49181E−05 Gj(2) = −409650E−05	5.81373E+01 7.03840E+00
4	17	100	−1.53715E+02	Gj(1) = 5.01457E−06 Gj(2) = 2.37636E−07	6.54597E+01 1.56191E+01
5	18	102	−1.53715E+02	Gj(1) = −2.54048E−05 Gj(2) = −4.05556E−05	3.03372E+01 0.00000E−01

NIA : No. of iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation

Table A3-2 Iteration history for problem 3 (Non-Scaling)

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	−1.22326E+02	IN THE FEASIBLE REGION	NA
1	2	8	−1.51843E+02	Gj(2) = 1.66141E−01	3.44449E+01
2	7	49	−1.55037E+02	Gj(1) = 2.43136E−02	5.55043E+01
3	125	631	−1.53722E+02	Gj(1) = 1.46263E−04 Gj(2) = −1.15812E−04	5.88432E+01 7.88256E+00
4	129	642	−1.53716E+02	Gj(1) = 2.75503E−06 Gj(2) = −3.64289E−05	5.94722E+01 3.30004E−01

NIA : No. of Iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation

Table A4 Iteration history for problem 4

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	9.62471E−01	IN THE FEASIBLE REGION	NA
1	1	3	9.56686E−01	IN THE FEASIBLE REGION	NA
2	9	50	7.68255E−01	Gj(1) = 5.18646E−02	1.89658E−01
3	16	94	7.78191E−01	Gj(1) = −9.83856E−05	1.86060E−01
4	17	96	7.78191E−01	Gj(1) = −1.00279E−04	1.49390E−01

NIA : No. of iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation

Table A5 Iteration history for problem 5

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	1.00000E+01	Gj(1) = 9.99666E−01 Gj(4) = 1.00000E+00	0.00000E−01 0.00000E−01
1	10	171	3.88280E−02	Gj(2) = 7.72412E−06	0.00000E−01

2	15	214	1.30961E−02	Gj(1) = 9.03749E−02 Gj(2) = 4.54373E−02	1.29002E−02 2.41664E−02
3	44	364	1.26913E−02	Gj(1) = −1.49158E−03 Gj(2) = 1.41682E−04	1.07549E−02 2.44131E−02
4	47	372	1.26787E−02	Gj(1) = 7.81004E−07 Gj(2) = 6.12985E−07	1.07661E−02 2.44238E−02

NIA : No. of iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation

Table A6 Iteration history for problem 6

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	2.62132E+01	IN THE FEASIBLE REGION	NA
1	5	34	1.77286E+01	Gj(10) = 1.89739E−01 Gj(13) = 1.17411E−01	9.32801E+00 6.93056E+00
2	12	81	2.04594E+01	Gj(10) = 5.14146E−03 Gj(13) = 2.70401E−03	1.18557E+01 8.52669E+00
3	16	127	2.05434E+01	Gj(10) = 2.45501E−05 Gj(13) = 8.08398E−06	1.19764E+01 8.57440E+00
4	18	134	2.05438E+01	Gj(10) = −2.45534E−07 Gj(13) = 8.64372E−08	1.19643E+01 8.57951E+00

NIA : No. of iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation

Table A7 Iteration history for problem 7

INTERACTIVE DESIGN OPTIMIZATION LIBRARY
VERSION 1.5
OPTIMIZATION HISTORY

NIA	NIB	NF	OBJECT VALUE	VIOLATED/ACTIVE CONSTRAINT	LAGRANGE MULTIPLIER
0	0	0	−1.02700E+01	Gj(9) = 7.00000E−01 Gj(10) = 5.00000E−01	0.00000E−01 0.00000E−01
1	7	39	−4.05384E+01	Gj(3) = 1.59095E−01 Gj(5) = 2.67826E−01 Gj(6) = 4.44518E−01	1.00123E+00 1.12223E+01 1.00864E+01
2	17	93	−3.24833E+01	Gj(3) = 4.91898E−03 Gj(5) = 2.64894E−03 Gj(6) = 8.08800E−03 Gj(9) = 6.94364E−04	1.28654E+00 1.22453E+01 1.17779E+01 4.37830E−01

3	23	139	−3.23490E+01	Gj(3)= 1.19598E−05 Gj(5)=−2.22507E−07 Gj(6)= 2.93290E−05 Gj(9)= 1.28974E−05	1.29348E+00 1.22444E+01 1.18392E+01 5.19154E−01

NIA : No. of iteration for ALM
NIB : No. of iteration for BFGS
NF : No. of calls for function evaluation