工業經營學會誌 第10卷 第15輯 1987年 6月 51

선형계획을 이용한 비선형문제 해법

A New Variant of Successive Linear Programming Algorithm

김 내 헌*

Abstract

This paper describes a new variant of Successive Linear Programming (SLP) algorithm called PSLP (Penalty SLP, 1985) using the multiplicative formulation proposed by Beale (1978). A Fortran computer implementation and its performance are also summarized.

1. 서 론

SLP (Successive Linear Programming) 알고리즘은 선형계획 문제를 반복하여 풀어나감으로써 비선형계획 문제를 해결하는 방법이다. 쉘 석유회사의 Griffith와 Stewart (1961)의 MAP (Method of Approximation Programming)으로부터 연유된 SLP 알고리즘은 선형계획 소프트웨어를 부프로그램으로 사용할 수 있어 쉽게 프로그램을 작성할 수 있을 뿐 아니라 대형의 비선형계획 문제를 다룰 수 있는 장점 때문에 많은 석유화학 회사에서 각기 실정에 맞는 SLP 알고리즘과 이의 소프트웨어를 개발하여 좋은 성과를 거두었다. 이에 대한 자세한 자료는 Lasdon과 Waren (1980)에서 찾아볼 수 있다.

일반적으로 SLP 알고리즘의 효율성은 근사 선형계획 문제로 전개하기 위한 비선형변수의 새로운 시행해를 어떻게 결정하는가와, 근사 선형계획 문제가 원래 비선형계획 문제의 좋은 근사 문제가 되도록 비선형변수의 변화 구간을 어떻게 효과적으로 정해주느냐에 따라 영향을 받게 된다. 이에 대한 가장 간단한 접근방법은 비선형계획 문제의 근사 선형계획 문제로부터 얻은 비선형변수의 값을 다음 근사식을 위한 시행해로 사용하고 비선형변수의 한계 구간은 변화시키지 않는 것이다. 그러나, 이러한 접근방법은 비선형계획 문제의 해가 제약공간의 꼭지점 (vertex)에 놓여 있는 경우에는 때때로 효과가 있으나 그 이외의 경우는 해를 찾지 못하고 실패하는 경우가 대부분이다. 해를 찾지 못하는 경우에는 비선형변수의 한계구간을 감소시킴으로써 안정점으로 근접하도록 조절할 수 있지만, 한계구간의 불필요한 감소는 해로의 수렴속도를 늦추게 하는 폐단이 있다.

효과적으로 한계구간을 확장하고 감소시켜야 하는 비선형변수의 갯수도 SLP 알고리즘의 효율성에 영향을 미친다. 이러한 관점에서 볼 때 Beale (1978)에 의해 제시된 비선형 문제의 형태는 동일한 비선형 문제라도 비선형변수로 지목되는 변수의 수가 적어진다. Baker와 Lasdon (1985)은 Beale의 형태를 이용한 단순한 SLP 알고리즘으로 엑손의 여러 생산계획 모형에서 좋은 결과를 얻었다.

본 연구에서는 Beale의 형태를 이용한 PSLP 알고리즘 (Lasdon et al., 1985)의 변형을 제시하고, 이 알고리즘을 이용하여 개발된 SLP 소프트웨어의 계산결과를 분석하였다.

2. 비선형계획 모형

본 연구에서 다루고 있는 비선형계획 문제의 수학적인 모형은 다음과 같다.

* 아주대학교 산업공학과 교수

본 연구는 한국과학재단의 지원으로 이루어졌읍니다.

문제 NLP

$$\text{Minimize} \quad \sum_{j=1}^{n} a_{oj}(\mathbf{y})\, x_j \quad \cdots\cdots\cdots\cdots \quad (1)$$

subject to

$$\sum_{j=1}^{n} a_{ij}(\mathbf{y})\, x_j \left\{\begin{matrix}=\\ \leqq \\ \geqq\end{matrix}\right\} b_i(\mathbf{y}),\ i \in NC \cdots (2)$$

$$\sum_{j=1}^{n} a_{ij}\, x_j \left\{\begin{matrix}=\\ \leqq \\ \geqq\end{matrix}\right\} b_i,\ i \in LC \cdots\cdots (3)$$

$$0 \leqq x_j \leqq u_j,\ j = 1, \cdots, n \quad \cdots\cdots\cdots\cdots (4)$$

$$l_r \leqq y_r \leqq u_r,\ r = 1, \cdots, k \quad \cdots\cdots\cdots\cdots (5)$$

위의 비선형계획 문제의 모형은 선형계획 문제와 유사한 형태를 갖고 있는데, LC는 순수 선형제약식을 의미하고, NC는 비선형제약식을 의미한다. 비선형제약식의 변수 x_j의 계수 $a_{ij}(\mathbf{y})$와 우변 $b_i(\mathbf{y})$는 상수이거나 또는 k개의 비선형변수 $y_1, \cdots, y_k$의 함수이다. 대형의 비선형계획 문제에서는 대부분의 $a_{ij}(\mathbf{y})$는 상수이며 적은 수의 변수 x_j만이 비선형변수 $\mathbf{y}$의 함수이다. 계산상의 편의를 위하여 혼란이 없는한 $a_{ij}(\mathbf{y})$와 $b_i(\mathbf{y})$도 각각 a_{ij}와 b_i로 표시하기로 한다.

원유의 정유과정에서 얻어진 다음의 예제는 Pooling 문제로 알려져 있는데 모형 (1)~(5)의 기존의 모형과 비교하여 왜 비선형변수로 지목되는 변수의 수가 적어지는가를 보여주고 있다.

예제

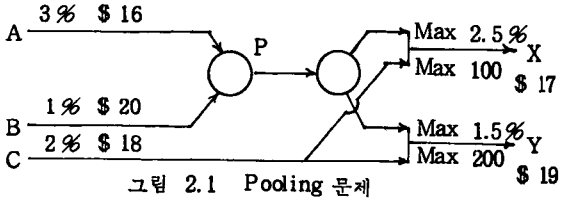

그림 2.1 Pooling 문제

위의 그림 2.1은 정유회사의 실제 생산공정으로서 유황 함유량이 다른 세 종류의 원유 A, B와 C를 혼합하여 유황성분이 다른 두 제품 X와 Y를 생산하는 과정이다. 유황 함유량이 각각 3%와 1%인 원유 A와 B는 같은 탱크에 저장된 후 다시 유황성분이 2%인 원유 C와 혼합되어 유황성분이 2.5%를 넘지 않는 제품 X와 유황성분이 1.5%를 넘지 않는 제품 Y를 생산하는데 사용된다. 원유 A, B 및 C의 배럴당 가격은 각각 16달러, 20달러, 18달러이고, 제품 X와 Y는 각각 배럴당 17달러와 19달러에 판매할 수 있다고 한다. 최대 판매량이 각각 100배럴과 200배럴이라 할 때 최대의 이익을 위한 수학적인 모형은 다음과 같다.

	A	B	PX	CX	X	PY	CY	Y	
PROFIT	−16	−20		−18	17		−18	19	
PLBAL	− 1	−1	1			1			= 0
XBAL			−1	−1	1				= 0
XSULF			P	2	−2.5				≦ 0
XMAX					1				≦100
YBAL						−1	−1	1	= 0
YSULF						P	2	−1.5	≦ 0
YMAX								1	≦200
PDEF	(P−3)			(P−1)					= 0

P는 원유 A와 B가 혼합되어 만들어진 원유의 유황성분으로서 원유 A와 B의 양이 결정되지 않는 한 알지 못하는 값이다. 그런데, 원유 A와 B의 양은 결정변수이므로 P도 또한 결정변수가 되어 비선형이 나타나게 된다. 함수 PROFIT는 목적함수로 판매액에서 비용을 제외한 이익을 표시하며, 제약식 PLBAL은 원유 A와 B의 합은 제품 X와 Y 생산에 소요된 PX와 PY의 합과 같음을 보여준다. 제약식 XBAL과 YBAL은 원유 소용량과 제품생산량에 대한 균형을, 제약식 XSULF와 YSULF는 제품 X와 Y의 유황성분에 대한 제한을, 제약식 PDEF는 원유 A와 B를 혼합했을 때 유황성분의 균형을 나타낸다. XMAX와 YMAX는 제품 X와 Y에 대한 생산 제한을 나타내고 있다.

기존의 SLP 알고리즘에서 다루고 있는 비선형계획 모형에서는 변수 A, B, PX, PY 및 P가 모두 비선형변수이지만 모형 (1)~(5)에서는 A, B, PX PY는 계수가 비선형변수 P의 함수인 선형변수로 보게 되어 비선형변수는 P 하나만 남게 된다.

문제 NLP (1)~(5)에 대응하는 L_1 Exact Penalty Function을 다음과 같이 정의한다.

$$P(\mathbf{x}; w) = \sum_{j=1}^{n} a_{oj}\, x_j + w \Big\{ \sum_{i \in INE} \Big| \sum_{j=1}^{n} a_{ij}\, x_j - b_i \Big| + \sum_{i \in INL} \max\Big(0, \sum_{j=1}^{n} a_{ij}\, x_j - b_j\Big) + \sum_{i \in ING} \Big(-\min\Big(0, \sum_{j=1}^{n} a_{ij}\, x_j - b_i\Big)\Big) \Big\} \quad \cdots\cdots\cdots\cdots (6)$$

위 식에서 INE, INL과 ING는 각각 문제 NLP의 비선형 제약식 (2)의 =, ≦, 및 ≧ 제약식을 의

미하고 있으며, w 는 충분히 큰 양의 실수이다. 문제 NLP의 (3)을 제약식으로 갖는 Penalty Function 문제는 아래와 같이 쓸 수 있다.

문제 EPF

Minimize $P(x ; w)$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \left\{ \begin{matrix} = \\ \leq \\ \geq \end{matrix} \right\} b_i, \quad i \in LC$$

$$0 \leq x_j \leq u_j, \quad j = 1, \cdots, n$$

$$l_r \leq y_r \leq u_r, \quad r = 1, \cdots, k$$

여기서 LC는 순수 선형제약식의 집합이다. Han과 Mangasarian (1979)의 결과를 확장한 Lasdon et al. (1985)의 정리 2.1~정리 2.3을 적용하면 문제 EPF의 극소값은 문제 NLP의 안정점과 일치하게 되고 그 역도 성립하므로 문제 EPF를 풀면 문제 NLP를 해결하게 된다.

3. SLP 알고리즘

본 연구에서 개발된 SLP 소프트웨어에 내재된 SLP 알고리즘은 문제 EPF의 1차 Taylor 전개식으로 얻어지는 근사 선형계획 문제에 trust region bound의 제약을 가한 선형계획 문제를 반복적으로 풀어 나감으로써 문제 EPF의 해, 즉 문제 NLP의 해를 구한다.

적절한 w의 값이 결정되었다 가정하고 시행해 (x^0, y^0)에서 식 (6)을 Taylor 급수로 전개한 후 1차 근사식을 취하면 다음과 같이 쓸 수 있다.

$$PL(x, y) = \sum_{j=1}^{n} a_{oj} x_j + \sum_{r=1}^{k} c_{or} y_r - \sum_{r=1}^{k} c_{or} y^0_r + w\{ \sum_{i \in INE} | f_{ij} | + \sum_{i \in INL} \max(0, f_{ij}) + \sum_{i \in ING} \max(0, -f_{ij}) \} \cdots\cdots (7)$$

여기서

$$f_{ij} = \sum_{j=1}^{n} a^0_{ij} x_j + \sum_{r=1}^{k} c_{ir} y_r - (b^0_i + \sum_{r=1}^{k} c_{ir} y^0_r)$$

이고,

$$c_{ir} = \sum_{j=1}^{n} \left(\frac{\partial a_{ij}}{\partial y_r} \right)^0 x^0_j - \left(\frac{\partial b_i}{\partial y_r} \right)^0$$

이며, $(\cdot)^0$는 점 (x^0, y^0)에서 계산된 값을 의미한다.

일반적으로 $\triangle y_r = y_r - y^0_r$이 크지 않으면 식 (7)은 식 (6)의 좋은 근사식이 되므로 편차변수(deviation variable) S^+_i와 S^-_i을 이용하면 다음과 같은 근사 선형계획 문제를 얻는다.

문제 LPP

Minimize $PL(x, y)$

$$= \sum_{j=1}^{n} a^0_{oj} x_j + \sum_{r=1}^{k} c_{or} y_r - \sum_{r=1}^{k} c_{or} y^0_r + w\{ \sum_{i \in INE} (S^+_i + S^-_i) + \sum_{i \in INL} S^+_i + \sum_{i \in ING} S^-_i \} \cdots\cdots (8)$$

Subject to

$$\sum_{j=1}^{n} a^0_{ij} x_j + \sum_{r=1}^{k} c_{ir} y_r - S^+_i + S^-_i = b^0_i + \sum_{r=1}^{k} c_{ir} y^0_r, \quad i \in INE \cdots\cdots (9)$$

$$\sum_{j=1}^{n} a^0_{ij} x_j + \sum_{r=1}^{k} c_{ir} y_r - S^+_i \leq b^0_i + \sum_{r=1}^{k} c_{ir} y_r, \quad i \in INL \cdots\cdots (10)$$

$$\sum_{j=1}^{n} a^0_{ij} x_j + \sum_{r=1}^{k} c_{ir} y_r + S^-_i \geq b^0_i + \sum_{r=1}^{k} c_{ir} y^0_r, \quad i \in ING \cdots\cdots (11)$$

$$\sum_{j=1}^{n} a_{ij} x_j \left\{ \begin{matrix} = \\ \geq \\ \leq \end{matrix} \right\} b_i, \quad i \in LC \cdots\cdots (12)$$

$$0 \leq x_j \leq u_j, \quad j = 1, \cdots, n \cdots\cdots (13)$$

$$\max(l_r, y^0_r - \rho_r) \leq y_r \leq \min(u_r, y^0_r + \rho_r), \quad r = 1, \cdots, k \cdots\cdots (14)$$

$$S^+_i \geq 0, \quad S^-_i \geq 0 \cdots\cdots (15)$$

ρ_r는 비선형변수 y_r의 변화 범위를 설정해 주는 값으로 trust region bound라 불리운다. trust region의 의의는 $P(x ; w)$에 대한 일차 근사식 $PL(x, y)$은 시행해 (x_0, y_0)에서 가까울수록 좋은 근사값을 주게 되므로 이 두 값 $P(x ; w)$와 $PL(x, y)$의 비에 따라 매 SLP 반복과정마다 y_r의 변동범위 $-\rho_r \leq y_r - y^0_r \leq \rho_r$을 설정해 주는데 있다.

개발된 SLP 소프트웨어에 내재된 SLP 알고리즘의 k번째 반복과정의 주된 단계는 다음과 같다. 매개변수(tolerance, parameter), w, ε, R_l, R_u, m 및 NSTOP 등의 값은 결정되어 있다고 가정한다.

단계 0. 비선형 변수 y_r의 한계구간 $\rho_r^{(k)}$, 시행해 $(x^{(k)}, y^{(k)})$, 시행해에서의 penalty

function의 값 $Pold = P(x^{(k)}; w)$, 일차 근사식의 값 $PL_{old} = PL(x^{(k)}, y^{(k)})$이 있다.

단계 1. 시행해 $(x^{(k)}, y^{(k)})$을 이용해 얻은 문제 LPP를 풀어 최적해 (x^*, y^*)를 구하고 $P_{new} = P(x^*, w)$와 $PL_{new} = PL(x^*, y^*)$를 구한다.

단계 2. $\triangle L = PL_{new} - PL_{old}$의 절대값이 ε (>0)보다 작으면 알고리즘의 시행을 중단한다.

단계 3. 비 $R = (P_{new} - P_{old})/\triangle L$의 값이 $R_l \leq |1-R| \leq R_u$이면 단계 4로, 그렇지 않으면 $\rho_r^{(k)}$를 다음과 같이 수정한 후 단계 5로 간다.

$$\rho_r^{(k+1)} = \begin{cases} \rho_r^{(k)}/m, & |1-R| > R_u \text{ 일때} \\ m \cdot \rho_r^{(k)}, & |1-R| < R_l \text{ 일때} \end{cases}$$

단계 4. Barnes (1967) 방법에 의해 비선형 비기저 변수의 한계구간을 확장한다.

단계 5. 모든 비선형변수의 한계구간이 허용기준 (tolerance)보다 작으면 알고리즘의 수행을 중단한다.

단계 6. $R \leq 0$이면 단계 10으로 간다. penalty function $P(x; w)$의 상대변화가 계속하여 NSTOP SLP 반복과정 동안에 진전이 없으면 알고리즘의 수행을 중단한다.

단계 7. 점 (x^*, y^*)가 문제 NLP의 실행가능해가 아니면 단계 10으로 간다. 문제 NLP의 목적함수의 상대변화가 계속하여 NSTOP SLP 반복과정 동안에 진전이 없으면 알고리즘의 수행을 중단한다.

단계 8. 양의 값을 갖는 편차변수가 없으면 최적해 조건을 검토한 후, 만족되면 알고리즘의 수행을 중단한다.

단계 9. $x^{(k+1)} = x^*$, $y^{(k+1)} = y^*$로 둔 후 단계 0으로 간다.

단계 10. $x^{(k+1)} = x^{(k)}$, $y^{(k+1)} = y^{(k)}$로 둔 후 단계 0으로 간다.

단계 4의 한계구간 변경은 비기저 변수가 한쪽 방향으로 움직여 가려는 경향이 있으면 이를 포착하여 움직여 가려는 방향의 구간한계를 확장시키는 방법으로 경험적인 방법이다. 이 SLP 알고리즘의 특징은 문제 NLP의 실행가능해를 찾으려는 시도가 없다는 것이다. 따라서, 실행가능해를 찾으려고 시도하는 알고리즘이 제약공간의 경계를 가까이 움직여감으로써 해로의 수렴이 늦어지는 폐단을 극복할 수 있다.

4. SLP 소프트웨어

매개변수 w의 값(penalty weight)은 문제 NLP의 해를 구하기 전에는 미리 알 수 없는 값이므로, SLP 알고리즘으로 문제 NLP를 풀기 위하여는 w의 예측값을 사용할 수밖에 없다. 이 값은 문제 NLP의 최적 쌍대변수(dual variable)의 절대값들의 최대치보다 큰 값이면 된다. 그러나 필요 이상으로 큰 w의 값은 문제의 해를 구하지 못하는 경우도 있으므로 (Lasdon et al., 1985), 적절한 w값을 선정하는 방법이 요구되나 아직은 경험에 의한 예측치 혹은 예측치를 이용한 계산 결과를 분석하여 보정된 값을 사용하므로써 문제를 해결하고 있다.

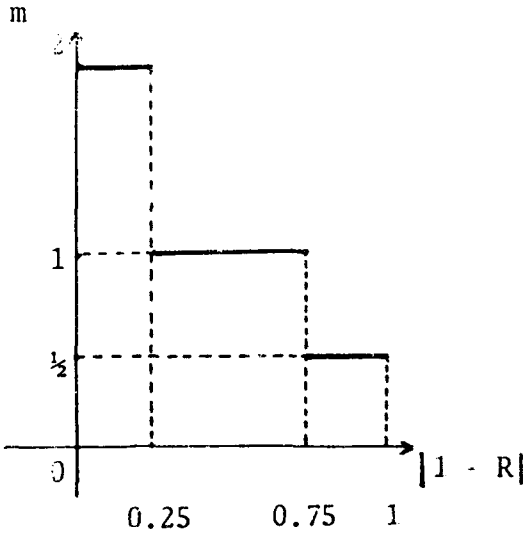

그림 4.1 m과 R의 관계도(불연속)

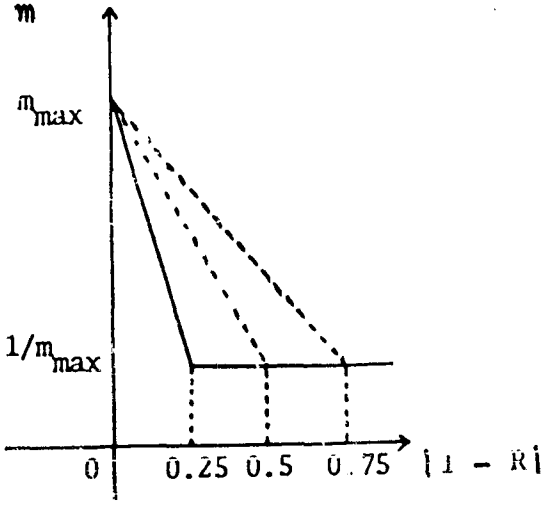

그림 4.2 m과 R의 관계도(연속)

trust region과 밀접한 관련이 있는 매개변수 R_l, R_u 및 m의 값은 각각 0.25, 0.75 및 2가 사용된다. 비 R값의 범위에 따라 비선형변수의 한계구간을 단순히 m배로 확장하거나 1/m로 축소하는 그림 4.1과 같은 이산적인 방법이 주로 사용되고 있으나, 그림 4.2와 같은 연속적인 구간 변화방법도 생각할 수 있다. 두 경우를 비교한 계산 결과는 표 5.5와 표 5.6에 요약되어 있다.

SLP 알고리즘에 대한 컴퓨터 프로그램은 FORTRAN Ⅳ로 작성되었으며, 현재 PRIME 550 Ⅱ 컴퓨터에서 사용되고 있다. PRIME 550 Ⅱ는 32비트 기계와 동등한 정도의 유호 숫자 정확도를 갖고 있지만 보다 정확한 도함수 계산을 위하여 DOUBLE PRECISION을 사용하였다. 만일 비선형함수의 도함수가 SUBROUTINE PARSH에 의해 사용자가 입력하는 경우는 32비트 기계의 SINGLE PRECISION도 허용될 수 있다.

SLP 시스템은 수정된 MINOS(Murtagh, Saunders, 1977) 코드를 부 프로그램으로 사용하여 선형계획 문제를 해결하고 있는데, 입출력 데이터나 사용하는 파일은 IBM에서 개발한 수리계획법 시스템(Mathematical Programming System)의 MPS/360, MPSX 및 MPSX/370 등과 흡사한 구조를 갖고 있다.

5. 테스트문제의 계산결과

본 연구에서 개발된 SLP 소프트웨어와 비교되는 SLP 소프트웨어는 다른 알고리즘과 비교하여 좋은 실행결과를 보여준 PSLP 알고리즘(Lasdon et al., 1985)이다. 두 SLP 소프트웨어는 모두 PRIME 550 Ⅱ에서 동일한 조건아래 실행되도록 하였다. 동일한 초기시행해, 비선형변수의 초기 한계구간, 동일한 도함수 계산방법 및 가능한한 동일한 매개변수 값들을 사용함으로써 주변의 영향을 극소화하도록 하였다.

테스트에 사용된 비선형계획 문제는 두 종류로, 첫번째 테스트 문제는 Himmelblau (1972)의 문제로 화학공학 공정으로부터 얻어지는 최적화 모형이고, 두번째 종류의 문제는 비선형성이 큰 소형의 기하계획(geometric programming) 문제로 Dembo (1974) 문제이다.

표 5.1 Himmelblau 문제의 계산 결과

- o K.C : Kuhn-Tucker 조건
- o F.C : 목적함수의 변화가 적으면 중단
- o S.S : 한계구간이 적으면 중단

문제	실행시간	SLP 반복	GCOMP 호출회수	LP pivot	수행 중단조건	super-basic	목적함수	SUMINF
H 1	0.2667	3	10	2	K.C	0	1.393364802	0.5425 E-4.
H 4	7.82422	40	371	182	F.C	7	−47.75144326	0
H 4a	4.32424	21	192	71	F.C	7	−47.75726596	0.1460 E-3
H 5	1.06061	12	40	13	F.C	1	961.6798298	0.3228 E-1
H 7	3.51818	25	128	33	F.C	1	1162.027464	0.2705 E-5
H 10	2.67273	21	82	33	S.S	1	−32.34867855	0
H 11	0.54242	2	13	13	K.C	0	−30665.56479	0.7521 E-3
H 12	2.46969	7	33	22	K.C	0	1.905155260	0.4649 E-6
H 13	0.34545	3	16	7	K.C	0	−5.193700125	0
H 16	5.36065	19	155	66	F.C	2	−0.8660269839	0.1096 E-4
H 18	10.26364	27	358	111	F.C	4	32.35788084	0
H 20	19.05454	8	201	114	K.C	0	−0.0565804983	0.4671 E-5
H 23	123.93029	19	1820	474	F.C	30	−1733.48	0
H 24	0.22121	2	7	1	K.C	0	1.0	0

표 5.1은 Himmelblau 문제의 계산 결과를 주고 있는데 대부분의 문제가 안정점으로 수렴하고 있음을 알 수 있다. GCOMP은 부 프로그램으로 비선형함수의 값을 제공하는데 사용자에 의해 주어진다.

표 5.2는 PSLP 알고리즘과의 총체적인 비교를 보여주고 있는데 두 소프트웨어의 실행 결과가 큰 차이를 보이지 않고 있다. 본 연구의 SLP 알고리즘이 사용하는 수학적인 모형에 적합한 문제는 오직 H 13 하나 뿐으로 만일 이러한 유형의 문제가 많이

포함되어 있으면 좋은 결과를 얻을 것으로 기대된다.

표 5.2 Himmelblau 문제 실행결과 대비

	PSLP	SLP
총 실행시간 (초)	183.12	181.85
총 SLP iteration	209	209
총 GCOMP 호출회수	3,429	3,426

표 5.3은 Dembo 문제의 실행결과로 D5와 D6은 penalty weight w의 값에 매우 민감하여서, w의 값을 필요 이상으로 크게하였을 때는 안정점을 찾는데 실패하였다. D4A, D5와 D6의 해를 구하기 위하여 목적함수의 상대적인 개선을 10^{-5} 정도까지 고려하였다.

표 5.3 Dembo 문제의 계산결과

문제	실행시간	SLP 반복	GCOMP 호출회수	LP pivot	수행 중단조건	super-basic 수	목적함수	SUMINF
D1B	5.3576	25	158	110	S.S	10	3.168437682	0.9058 E-5
D2	0.7242	4	25	9	K.C	0	30,669.71111	0.1567 E-6
D4A	10.2788	57	434	184	F.C	4	3.951166734	0.7935 E-6
D4C	10.3727	47	390	156	F.C	4	3.952138047	0.5789 E-6
D5	8.6242	64	433	106	S.S	3	7,048.873486	0.2532 E-4
D6	11.8727	65	430	83	S.S	3	102.0189666	0.6887 E-4
D7	4.8845	12	125	49	K.C	1	174.911058	0.6257 E-4

비선형이 강한 소형문제에 대해 두 알고리즘은 거의 동일한 결과를 갖는 것이 표 5.4에 나타나 있다. 이는 문제의 형태가 어느 한 SLP 알고리즘에 유리한 영향을 주지 않을 뿐 아니라, 문제의 크기도 작기 때문이라 하겠다.

표 5.4 Dembo 문제 실행결과 대비

	PSLP	SLP
총 실행시간 (초)	52.65	52.12
총 SLP iteration	274	274
총 GCOMP 호출회수	1,995	1,995

표 5.5와 표 5.6은 그림 4.2의 세 종류의 연속함수를 사용하여 비선형변수의 한계구간을 변화시키며 문제를 푼 결과를 보여주고 있다. 0.25, 0.5 및 0.75는 그림 4.2에서 기울기가 각각 다른 함수에 대응하는 것이며, m_{max}의 값은 2로 하였다.

*표는 PSLP와 같은 한계구간과 가중치 w를 사용하였을 때 최적해로의 수렴이 실패한 경우이다. 0.75의 경우는 모든 문제를 전부 해결하였으나 소요시간의 감소는 두드러지지 않다. 이 결과는 경험에 의한 이산적인 한계구간 변화방법이 다른 방법과 비교하여 결코 열등하지 않음을 보여준다.

6. 결 론

비선형계획 문제를 다룰 수 있는 효율적인 소프트웨어의 부족으로 인하여 비선형 모형의 사용이 많은 제약을 받아왔다. 그러나, SLP 알고리즘과 같이 사용 가능한 효율적인 소프트웨어가 쉽게 접근 가능하다면 이와같은 제약은 감소된다. SLP 소프트웨어는 선형 문제에 가까운 대형의 비선형 문제에는 더욱 효율적이며, 알고리즘 자체가 선형계획 문제만을 풀기 때문에 임의로 큰 비선형 문제를 다룰 수가 있다.

표 5.5 한계구간 변화방법에 따른 효율성 비교(Himmelblau)

	비 R의 연속함수									본 연구의 SLP			PSLP		
	0.25			0.5			0.75								
	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수
HIMM01	0.2303	10	3	0.2576	10	3	0.2424	10	3	0.2667	10	3	0.2818	10	3
HIMM04	0.9455	39	8	2.8030	149	18	4.0030	193	22	7.8242	371	40	7.9212	371	40
HIMM04A	1.7242	89	8	2.5152	124	13	4.5242	213	22	4.3242	192	21	4.2909	192	21
HIMM05	0.7758	35	10	0.8970	36	11	0.8667	36	11	1.0606	40	12	1.0727	40	12
HIMM07	*			*			3.3697	148	27	3.5182	128	25	3.4970	128	25
HIMM10	1.0242	29	8	1.1364	32	11	2.0030	69	18	2.6727	82	21	3.2030	82	21
HIMM11	0.4667	13	2	0.5333	13	2	0.5242	13	2	0.5424	13	2	0.5636	13	2
HIMM12	2.0939	27	6	3.2515	39	8	2.7818	33	7	2.4697	33	7	2.6512	33	7
HIMM13	0.2242	11	2	0.2636	11	2	0.2061	11	2	0.3455	16	3	0.3485	19	3
HIMM16	*			3.4152	103	12	4.8758	163	18	5.3607	155	19	5.3394	155	19
HIMM18	*			9.5999	435	29	14.1212	631	45	10.2636	358	27	10.2788	358	27
HIMM20	*			*			16.7576	201	8	19.0545	201	8	20.2091	201	8
HIMM24	0.2091	7	2	0.2030	7	2	0.1999	7	2	0.2212	7	2	0.2546	7	2
합 계	—	—	—	—	—	—	54.4756	1728	187	57.9242	1606	190	59.9118	1609	190

표 5.6 한계구간 변화방법에 따른 효율성 비교(Dembo)

	비 R의 연속함수									본 연구의 SLP			PSLP		
	0.25			0.5			0.75								
	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수	실행시간	GCOMP 호출회수	SLP 반복회수
DEMB01B	2.6515	94	9	4.0879	165	20	4.2364	155	22	5.3576	158	25	5.3394	158	25
DEMB02	0.7455	25	4	0.6667	25	4	0.6970	25	4	0.7242	25	4	0.7273	25	4
DEMB04A	*			8.6758	473	56	10.9576	515	66	10.2788	434	57	10.4818	434	57
DEMB04C	0.8182	23	4	0.8394	23	4	10.8152	420	51	10.3727	390	47	10.4848	390	47
DEMB05	*			*			4.4424	337	40	8.6242	433	64	8.6212	433	64
DEMB06	*			*			16.8030	516	73	11.8727	430	65	11.5030	430	65
DEMB07	3.3303	87	6	4.1909	141	12	4.0576	123	10	3.8848	125	12	4.8151	125	12
합 계	—	—	—	—	—	—	52.0092	2091	266	52.1150	1995	274	52.6365	1995	274

References

Barnes, G. K., "A Comparative Study of Nonlinear Optimization Codes." Unpublished Masters Thesis. Univ. of Texas at Austin, 1967.

Beale, E. M. L. "Nonlinear Programming Using a General Mathematical Programming System." In Design and Implementation of Optimization Software. Ed. H. J. Greenberg. Netherlands : Sijthoff and Noordnoff, 1978, pp. 259 ~ 279.

Dembo, R. "A Set of Geometric Programming Test Problems and Their Solutions." Working Paper No. 87, Dept. of Management Sciences, Univ. of Waterloo, Ontario, Dec. 1974.

Griffith, R. E., and R. A. Stewart. "A Nonlinear Programming Technique for the Optimization of Continuous Processing Systems." Management Science, Vol. 7 (1961), pp. 379 ~ 392.

Han, S. P. and O. L. Mangasarian. "Exact Penalty Function in Nonlinear Programming." Mathematical Programming, Vol. 17 (1979), pp. 251 ~ 269.

Himmelblau, D. M. Applied Nonlinear Programming. New York : McGraw-Hill, 1972.

Lasdon, L. S., And A. D. Waren. "Survey of Nonlinear Programming Applications." Operations Research, Vol. 28, No. 5 (Sept.-Oct. 1980), pp. 1029 ~ 1073.

Lasdon, L. S., And T. E. Baker. "Successive Linear Programming at EXXON." Management Science, Vol. 31, No. 3, March 1985, pp. 264 ~ 274.

Lasdon, L. S., Jianzhong Zhang, and Kim, N. H. "An Improved Successive Linear Programming Algorithm." Management Science, Vol. 31, No. 10, October 1985, pp. 1312 ~ 1331.

Murtagh, B. A. and M. A. Saunders. "MIONS User's Guide." Technical Report SOL 77 - 9. Stanford, California : Stanford Univ., 1977.