パソコンによる対話型LP教育用プログラムシステムについて

青江　俊夫*・成久　洋之**

*岡山理科大学　情報処理センター
**岡山理科大学　電子理学科

（昭和60年9月26日　受理）

1. はじめに

最適値問題というのは，ある達成したい目的あるいは効果を，いろいろな条件を考慮し，また結果を予想しながら，その上で最良の方策を決定するためにはどうしたらよいのかを追求するものである．たとえば，原料・労働力・エネルギー・……等の資源に制限が加えられた場合に，これらの与えられた資源の組み合わせで製品の生産量を最大にし，さらに利益も最大になるようにして，一方では，投入した原料や労働力を最少にするための方策はどうすればよいのかを探求する問題である．

最適値を求める方法（最適化法）としては，①線形計画法（linear programming; LP）②非線形計画法（non-linear programming; NP）③動的計画法（dynamic programming; DP）……等，種々の方法が考案されて利用されているが，ここでは，制約条件式と目的関数とが共に1次式で表される上記①の線形計画法の手法を用いることにする．

線形計画法は，リニア・プログラミング（Linear Programming）の訳であり，英語の頭文字を取って一般に **LP** と呼ばれている．LPの解法としては，シンプレックス法（simplex method）が最も普及している．今日では，コンピュータの大型化・高速化にともなって，制約条件式や変数の数がそれぞれ数1000個もあるような問題でも比較的に短時間で最適解を見つけることが容易になってきている．しかしながら，シンプレックス法のアルゴリズムに従って，与えられた問題を解こうとすると，単体表（simplex tableau）の作成や基底変数（basic variable）の選択，ピボット（pivot）要素による演算等，複雑な過程を経なければならないので，学校での講義・演習等で手計算（もちろん，電卓等を使用したとしても）で最適解を求めようとするとおのずから，変数や制約条件式の数に制限が加えられる．

そこで我々は，最近，比較的手軽に利用できるようになったパソコン（personal computer）を使用して，パソコンと対話しながら単体表の作成やピボット演算等を行い，最適解を見つける過程が学べるようなシステムをBASIC言語を使用して作成した．

2. LP の手法

LP は，連立 1 次不等式あるいは連立 1 次方程式を制約条件式として持ち，与えられた 1 次式の目的関数の値を最大または最小にするような非負の変数の値を求める方法であり，輸送問題，混合問題，生産計画，仕入管理等に広く用いられている．

LP の代表的なものは，制約条件式

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots\cdots + a_{1x}x_n \leqq b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots\cdots + a_{2n}x_n \leqq b_2 \\ \qquad\qquad \cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots\cdots + a_{mn}x_n \leqq b_m \end{cases} \tag{2. 1}$$

のもとで，目的関数

$$f(x) = c_1x_1 + c_2x_2 + \cdots\cdots + c_nx_n$$

を最大にするような x_1，x_2，……x_n を求める問題である．（2. 1）を，行列とベクトルを導入して表わすと，

$$\mathbf{A} = \begin{pmatrix} a_{11} \cdots\cdots a_{1n} \\ \vdots \qquad\quad \vdots \\ a_{m1} \cdots\cdots a_{mn} \end{pmatrix}, \quad \boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

$$\boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}, \quad \boldsymbol{c} = (c_1 c_2 \cdots\cdots c_n)$$

とすることにより

$$\begin{cases} \text{制約条件式} \\ \qquad \mathbf{A}\boldsymbol{x} \leqq \boldsymbol{b}, \quad \boldsymbol{x} \geqq \boldsymbol{o} \\ \text{目的関数} \\ \qquad f(\boldsymbol{x}) = \boldsymbol{c}\boldsymbol{x} \rightarrow \text{最大} \end{cases} \tag{2. 2}$$

と書ける．

ここで，（2. 1）における制約条件式のおのおのに対して，

$$v_i = b_i - (a_{i1}x_1 + a_{i2}x_2 + \cdots\cdots + a_{in}x_n) \quad (i = 1, 2, \cdots\cdots, m)$$

となるような非負の変数 v_i を導入すると，（2. 1）は

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots\cdots + a_{1n}x_n + v_1 = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots\cdots + a_{2n}x_n + v_2 = b_2 \\ \qquad\qquad \cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots\cdots + a_{mn}x_m + v_n = b_m \end{cases} \tag{2. 3}$$

なる形の連立 1 次方程式に書き表わせる．ここで導入した v_i $(i = 1, 2, \cdots\cdots, m)$ のことを，スラック変数（slack variable）と呼んでいる．（2. 3）を行列，ベクトルで表現すれば，

$$\mathbf{V}=\begin{pmatrix}1 & & 0\\ & \ddots & \\ 0 & & 1\end{pmatrix},\quad \boldsymbol{v}=\begin{pmatrix}v_1\\ \vdots\\ v_m\end{pmatrix}$$

とすると，

$$\begin{cases}\text{制約条件式}\\ \quad [\mathbf{A},\ \mathbf{V}]\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{v}\end{pmatrix}=\boldsymbol{b},\quad \begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{v}\end{pmatrix}\geqq\mathbf{0},\quad \boldsymbol{x}\geqq\mathbf{0},\ \boldsymbol{v}\geqq\mathbf{0}\\ \text{目的関数}\\ \quad f(\boldsymbol{xv})=(\boldsymbol{c},\ \mathbf{0})\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{v}\end{pmatrix}\rightarrow\text{最大}\end{cases}\qquad(2.\ 4)$$

となる．今，あらためて

$$\mathbf{A}=\begin{pmatrix}a_{11}\cdots\cdots a_{1n} & 1 & & 0\\ \vdots\quad\quad\vdots & & \ddots & \\ a_{m1}\cdots\cdots a_{mn} & 0 & & 1\end{pmatrix}$$

$$\boldsymbol{x}=\begin{pmatrix}x_1\\ \vdots\\ x_m\\ v_1\\ \vdots\\ v_m\end{pmatrix},\quad \boldsymbol{c}=(c_1\cdots c_n0\cdots\cdots0)$$

とおきなおすと，与えられた問題は，

$$\begin{cases}\text{制約条件式}\\ \quad \mathbf{A}\boldsymbol{x}=\boldsymbol{b},\quad \boldsymbol{x}\geqq\mathbf{0}\\ \text{目的関数}\\ \quad f(\boldsymbol{x})=\boldsymbol{cx}\rightarrow\text{最大}\end{cases}\qquad(2.\ 5)$$

を解けばよいことになる．

ここで，変数 x の $m+n$ 個の要素の中から任意の m 個を取り出して作ったベクトルが，互に１次独立であると仮定すると，(2. 5) の行列 $\mathbf{A}$ の中から上で選択した変数に対応する m 列を取り出して作った正方部分行列の逆行列は常に存在する．

今，変数ベクトル $\boldsymbol{x}$（要素は $m+n$ 個ある）の中から m 個取り出して，x_1，x_2，……，x_m としたものを基底変数，残りの x_{m+1}，x_{m+2}，……，x_{m+n} の n 個の変数を非基底変数と呼ぶ．基底変数，非基底変数の選択に対応して，制約条件式の係数行列 $\mathbf{A}$ および目的関数の係数ベクトル $\boldsymbol{c}$ も２分割して

$$\mathbf{A}=(\mathbf{A}_1\mathbf{A}_2),\quad \boldsymbol{x}=\begin{pmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{pmatrix}$$

$$\boldsymbol{c}=(\boldsymbol{c}_1\,\boldsymbol{c}_2)$$

とすると，制約条件式は

$$\mathbf{A}\boldsymbol{x}=(\mathbf{A}_1\mathbf{A}_2)\begin{pmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{pmatrix}$$

$$=\mathbf{A}_1\boldsymbol{x}_1+\mathbf{A}_2\boldsymbol{x}_2=\boldsymbol{b}$$

$$\therefore \quad \mathbf{A}_1\boldsymbol{x}_1=\boldsymbol{b}-\mathbf{A}_2\boldsymbol{x}_2 \tag{2. 6}$$

となる．ここで，$\mathbf{A}_1$ は $m\times m$ の正方行列であり，しかも正則であることから逆行列が存在し，$\boldsymbol{x}_1$ は（2. 6）より

$$\boldsymbol{x}_1=\mathbf{A}_1^{-1}\boldsymbol{b}-\mathbf{A}_1^{-1}\mathbf{A}_2\boldsymbol{x}_2, \quad \boldsymbol{x}_1\geqq\mathbf{0}, \quad \boldsymbol{x}_2\geqq\mathbf{0} \tag{2. 7}$$

と表現できる．ここで，非基底変数 $\boldsymbol{x}_2$ を $\mathbf{0}$ として得られる解

$$\boldsymbol{x}_1=\mathbf{A}_1^{-1}\boldsymbol{b}, \quad \boldsymbol{x}_2=\mathbf{0} \tag{2. 8}$$

を基底行列 $\mathbf{A}_1$ に対する基底解と呼ぶ．基底解は，たかだか ${}_{m+n}\mathrm{C}_m$ 個である．なぜなら，基底が1個決まると，基底解はクラメルの公式により一意的に決定できるからである．（2. 7）は基底解で，かつ $\mathbf{A}_1^{-1}\boldsymbol{b}\geqq\mathbf{0}$ である．これを許容基底解と呼ぶ．

次に目的関数を書き換えると，

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{c}\boldsymbol{x}=(\boldsymbol{c}_1\boldsymbol{c}_2)\begin{pmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{pmatrix} \\ &= \boldsymbol{c}_1\boldsymbol{x}_1+\boldsymbol{c}_2\boldsymbol{x}_2 \end{aligned} \tag{2. 9}$$

となる．（2. 9）に（2. 7）で求めた $\boldsymbol{x}_1$ を代入すると，

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{c}_1(\mathbf{A}_1^{-1}\boldsymbol{b}-\mathbf{A}_1^{-1}\mathbf{A}_2\boldsymbol{x}_2)+\boldsymbol{c}_2\boldsymbol{x}_2 \\ &= \boldsymbol{c}_1\mathbf{A}_1^{-1}\boldsymbol{b}-\boldsymbol{c}_1\mathbf{A}_1^{-1}\mathbf{A}_2\boldsymbol{x}_2+\boldsymbol{c}_2\boldsymbol{x}_2 \\ &= \boldsymbol{c}_1\mathbf{A}_1^{-1}\boldsymbol{b}-(\boldsymbol{c}_1\mathbf{A}_1^{-1}\mathbf{A}_2-\boldsymbol{c}_2)\boldsymbol{x}_2 \end{aligned} \tag{2.10}$$

となる．ここで，もし

$$\boldsymbol{c}_1\mathbf{A}_1^{-1}\mathbf{A}_2-\boldsymbol{c}_2\geqq\mathbf{0} \tag{2.11}$$

と仮定すると，目的関数 $f(\boldsymbol{x})$ の値は $\boldsymbol{x}$ が $\mathbf{0}$ から増加するに従って減少することになり，

$$\boldsymbol{x}_2=\mathbf{0}$$

としたときに $f(\boldsymbol{x})$ が最大となる．この時の最適解が（2. 8）で求めた

$$\boldsymbol{x}_1=\mathbf{A}_1^{-1}\boldsymbol{b}, \quad \boldsymbol{x}_2=\mathbf{0}$$

である．なお，上記（2.11）の

$$\boldsymbol{c}_1\mathbf{A}_1^{-1}\mathbf{A}_2-\boldsymbol{c}_2$$

は，シンプレックス判定規準と呼ばれるものである．

3. シンプレックス法

LP 問題の最適解は，一般的にはシンプレックス法と呼ばれるアルゴリズムを使用して解かれる．ここでは，簡単な具体例を述べることによりシンプレックス法を使用して問題がどのように解かれるかを説明する．

・例　題：

今2種類の製品 P_1，P_2 を生産すると仮定する．ただし，製品 P_1，P_2 とも原料 A，B を使用するが，原料 A は総量24単位を超えて使用することができず，また原料 B は同じく18単位を超えることができないという制限がある．一方，製品 P_1 には原料 A が6単位，

原料Ｂが3単位必要であり，製品P_2については原料Ａが4単位，原料Ｂが6単位必要であるとする．以上の条件のもとで，製品P_1，P_2をそれぞれx_1，x_2単位生産するときに，生産高x_1+x_2を最大にするようなx_1，x_2を求めなさい．

・アルゴリズム：

上記の例を一般的な式で表わすと，

$$\begin{cases} \text{制約条件式} \\ \quad 6x_1+4x_2 \leqq 24 \\ \quad 3x_1+6x_2 \leqq 18 \\ \qquad x_1,\ x_2 \geqq 0 \text{ のもとで} \\ \text{目的関数} \\ \quad z=x_1+x_2 \rightarrow \text{最大} \end{cases} \tag{3．1}$$

ということである．（3．1）の関係を図1に示す．図1において，斜線で囲まれた凸多角形の内側（境界線も含む，なぜなら制約条件式およびx_1，x_2は等号を含んでいる．）部分

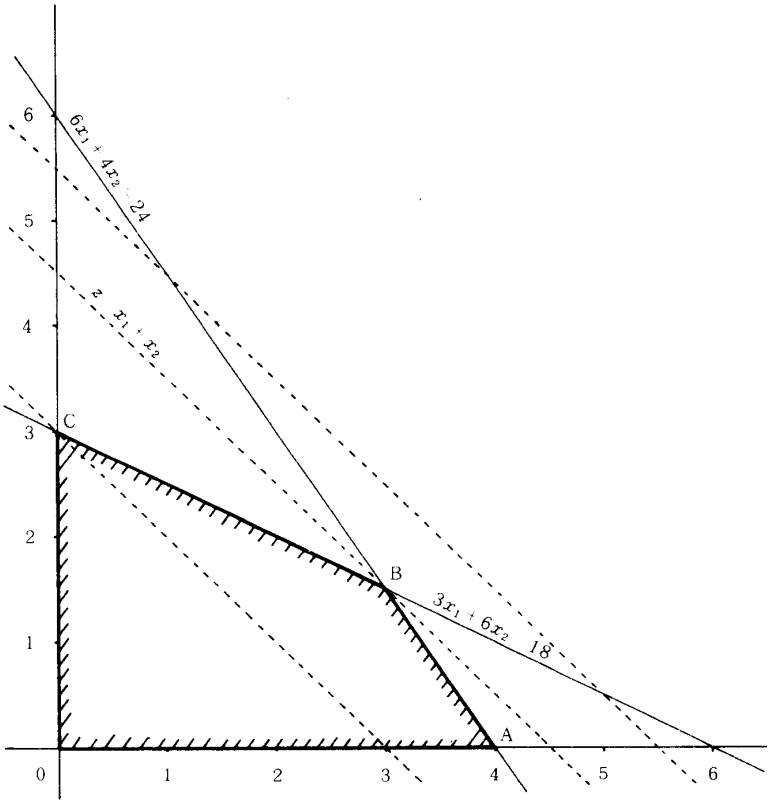

図 1．　条件式と最適解存在領域

が最適解の存在する領域であると考えられる．ここで，制約条件式を等式化するためにスラック変数x_3，x_4を導入すると（3．1）は，

$$\begin{cases} \text{制約条件式} \\ \quad 6x_1+4x_2+x_3 \quad\quad =24 \\ \quad 3x_1+6x_2 \quad\quad +x_4=18 \end{cases} \tag{3．2}$$

$$\left.\begin{array}{l} \qquad x_1,\ x_2,\ x_3,\ x_4 \geqq 0 \text{ のもとで} \\ \text{目的関数} \\ \quad z = x_1 + x_2 + 0 \cdot x_3 + 0 \cdot x_4 \rightarrow \text{最大} \end{array}\right.$$

の形に変形できる．

ここで，仮に各変数の値を

$$x_1 = 0,\quad x_2 = 0,\quad x_3 = 24,\quad x_4 = 18$$

とすると，これらは（3．2）の条件式を満足する1つの解となる．しかし，

$$x_1 = 0,\quad x_2 = 0$$

ということは，製品 P_1，P_2 を共に0単位生産することであり，言いかえれば P_1，P_2 はまったく生産をストップしている状態である．

表1の①の部分は，（3．2）に基づいて作成した単体表（シンプレックス表）と呼ばれるものである．表1の中で，v_j 列にある変数 x_n が現在基底変数として選択されている変

表 1．シンプレックス表

	v_j			0	0	1	1
	v_i	ベクトル	S	x_3	x_4	x_1	x_2
①	0	x_3	24	1	0	6	4
	0	x_4	18	0	1	3	6
		z_j	0	0	0	0	0
		$(z_j - v_j)$		0	0	-1	-1
②	1	x_1	4	1/6	0	1	2/3
	0	x_4	6	-1/2	1	0	4
		z_j	4	1/6	0	1	2/3
		$(z_j - v_j)$		1/6	0	0	-1/3
③	1	x_1	3	1/4	-1/6	1	0
	1	x_2	3/2	-1/8	1/4	0	1
		z_j	9/2	1/8	1/12	1	1
		$(z_j - v_j)$		1/8	1/12	0	0

数であり，z_j 行 S 列にある値が現在の目的関数の値を示している．また，$(z_j - v_j)$ の行は，もし現在の状態で生産活動を開始した時に利益の増加がどのくらい期待できるかが示されている．したがって，$(z_j - v_j)$ 行の値の中に負のものがあれば，それは生産することにより損失が発生していることになるので，その項を含む列に対応する x_n の単位生産量を増して損失を少なくする方向に改良しなければならない．シンプレッ法での重要な処理は，次の3段階である．

(a)　単体表の作成

与えられた制約条件式の係数行列および定数ベクトルから単体表の各要素に数値を埋め込み，初期状態を作る．初期状態あるいは(c)のピボット演算後の単体表から，z_j 行と (z_j-v_j) 行を計算する．

(b)　基底要素の選択

表１の①の場合，(z_j-v_j) 行の値は x_1，x_2 列で共に -1 であるから，x_1 を選択する．

x_1 を１単位実行するためには，x_3 を６単位，x_4 を３単位必要とするから（x_1 列 x_3 行，x_1 列 x_4 行の値を意味している），それぞれ，x_3 については $24/6=4$ 単位分実行可能であり，x_4 については $18/3=6$ 単位分実行可能な計算になる．したがって現実には，小さい方の値に束縛されるので４単位の実行が可能である．この結果，利益を上げるために x_3 の活動は抑止されて，x_1 が代って活動するようになる．したがって①のベクトル列の x_3 を x_1 に変更（基底変数の変更）して②に移る．

(c)　ピボット演算

(b)の処理で基底に選ばれた変数 x_1 について，x_1 行の全要素を x_1 行 x_1 列の要素で割ることにより②の x_1 行が得られる．次に他の基底行について，x_1 列の要素が０になるような演算を工夫して，同じ演算を同一基底行の他の列の要素についても実施し，単体表を変更する（ピボット演算と呼ばれている）．

以上の操作の後，再度(a)の単体表再構成を実行することにより表１の②が得られる．ここで (z_j-v_j) 行にまだ負の要素があれば，その要素に対応する変数 x_n の活動を活性化することにより利益の増大を計ることができるので，前述の(a)，(b)，(c)の処理を再度実行する．この例の場合，②の段階では (z_j-v_j) の値は変数 x_2 で負になっているので繰返し実行することになる．

最終的に，(z_j-v_j) 行に負の要素がなくなった時，最適解に到達したことになる．この例の場合，最適解は図１でいえば，目的関数 z をパラメータとする直線（図では点線で示した平行な直線群）が凸多角形 OABC 内に在り，しかも原点から一番遠く離れている点を通る直線と多角形との交点と考えられる．本例の場合，それは頂点 B を通る場合であるから最適解は B 点（$x_1=3$，$x_2=1.5$），すなわち製品 P_1 を３個，製品 P_2 を 1.5 個生産すればよいことになる．

4．作成システムの概要

本システムは対話形に作成してあるので，利用者は画面上に表示された指示にしたがって必要な情報を入力すればよい．本システムは，次のような５つの部分から構成されており，概略処理フローを図２に示す．

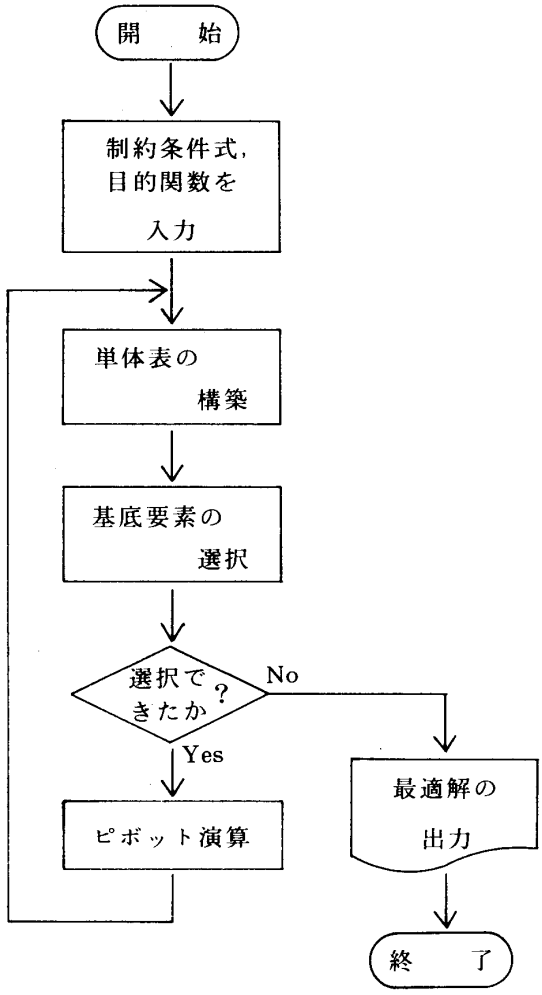

図 2. 作成システムのブロック・チャート

1. データ入力
2. 単体表の作成
3. 基底要素の選択
4. ピボット演算
5. 最適解出力

構成部分のおのおのについて，以下に解説する．

4．1 データの入力

単体表の初期状態を作成するのに必要な情報を入力する．

単体表を構成するには，制約条件式において各変数が持つ係数，条件式の等・不等号の区別，定数項が必要である．そこで本システムでは，なるべく制約条件式に忠実な形で入力ができるように考慮してある．制約条件式の入力時には，図５のような画面が順次表示されるので，それに対する応答として条件式を入力すればよい（図５で，下線の部分が利用者の応答である）． 全制約条件式の入力後に，左辺の変数の係数行列と右辺の定数ベク

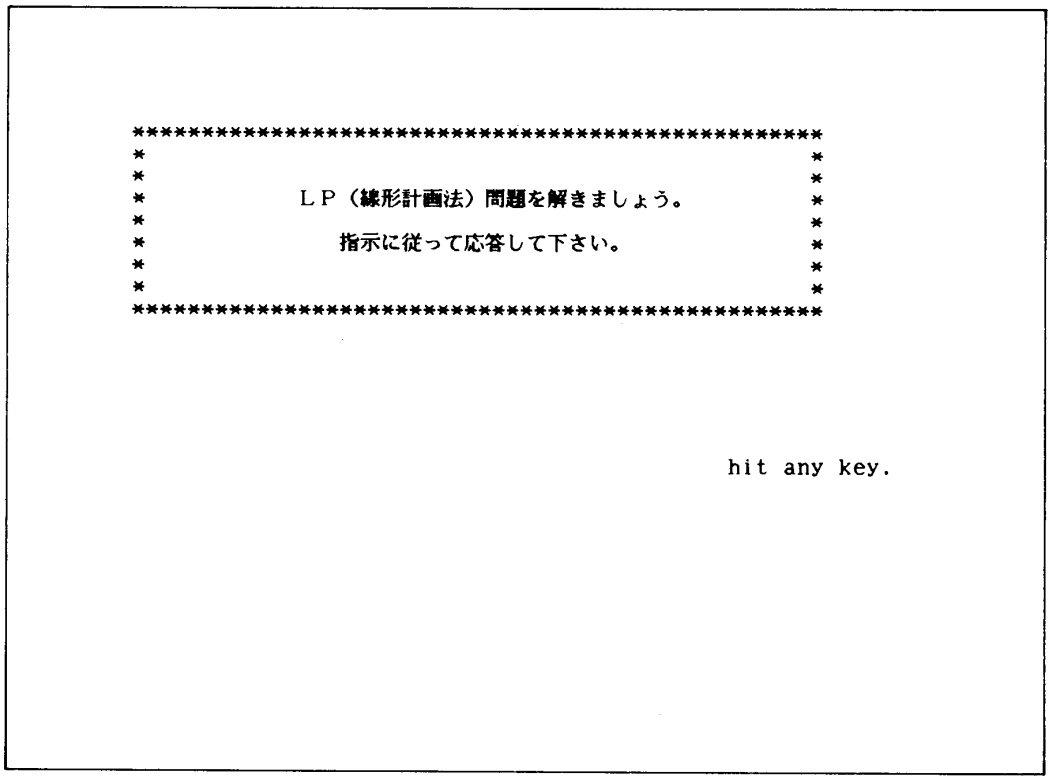

図 3.　画面表示例－１（初期画面）

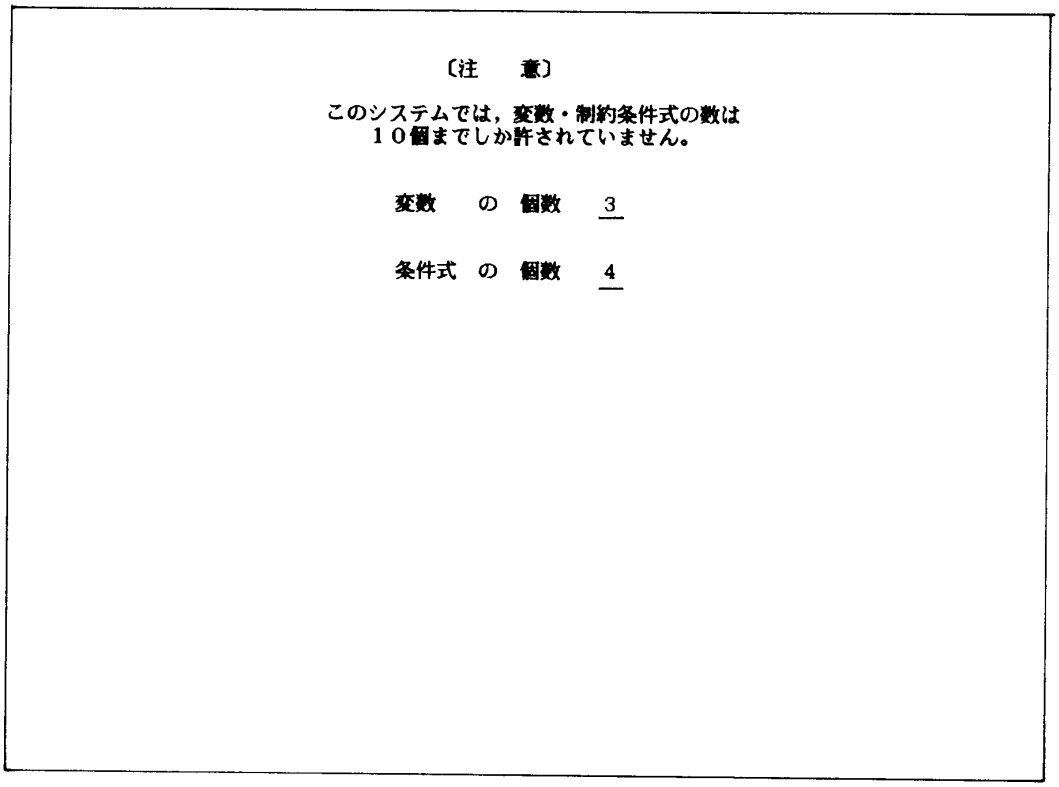

図 4.　画面表示例－２

```
条件式の係数，等・不等号，定数を入力して下さい

条件式 2 の係数，等・不等号，定数を入力

   X( 1 )=? 2
   X( 2 )=? 3
   X( 3 )=? 1

等・不等号? =<

定数? 60

制約条件式

 2 X( 1 ) + 3 X( 2 ) + 1 X( 3 ) =< 60
```

図 5. 画面表示例－3（条件式入力）

```
確認して下さい。正しければ Y 訂正は N ? Y

条件式の係数行列

   3.00     2.00     1.00
   2.00     3.00     1.00
   0.00     4.00     3.00
   1.00     0.00     2.00
```

図 6. 画面表示例－4（係数行列確認）

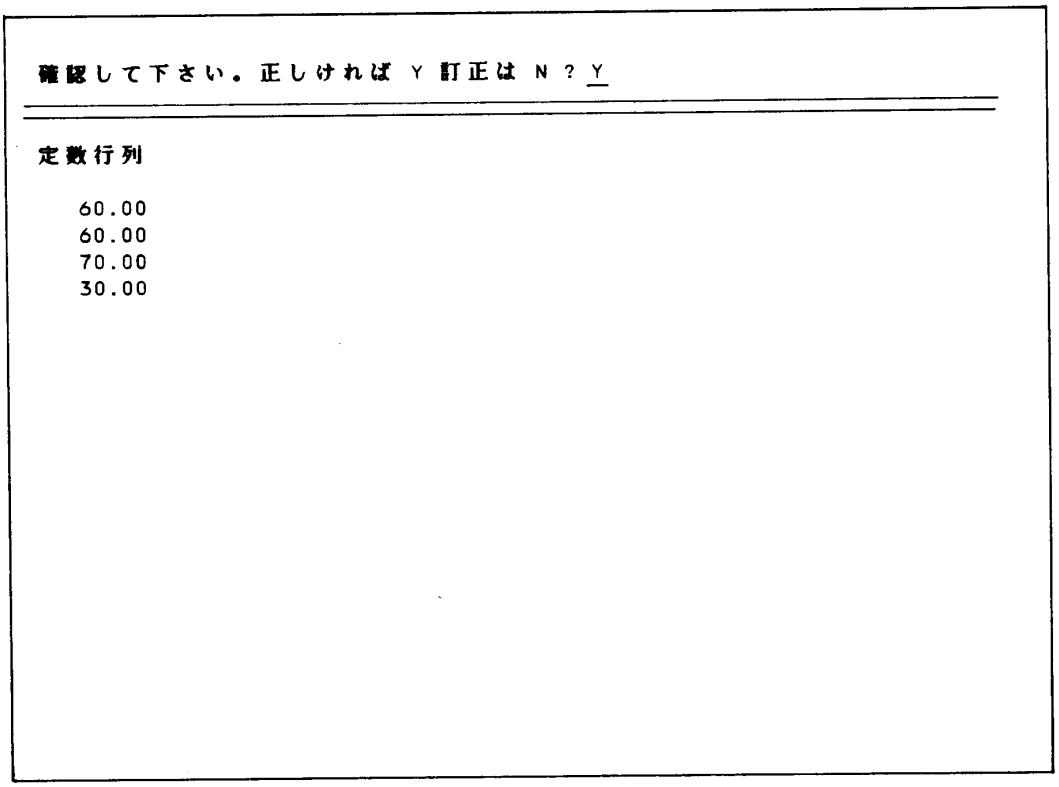

図 7.　画面表示例－５（定数ベクトル確認）

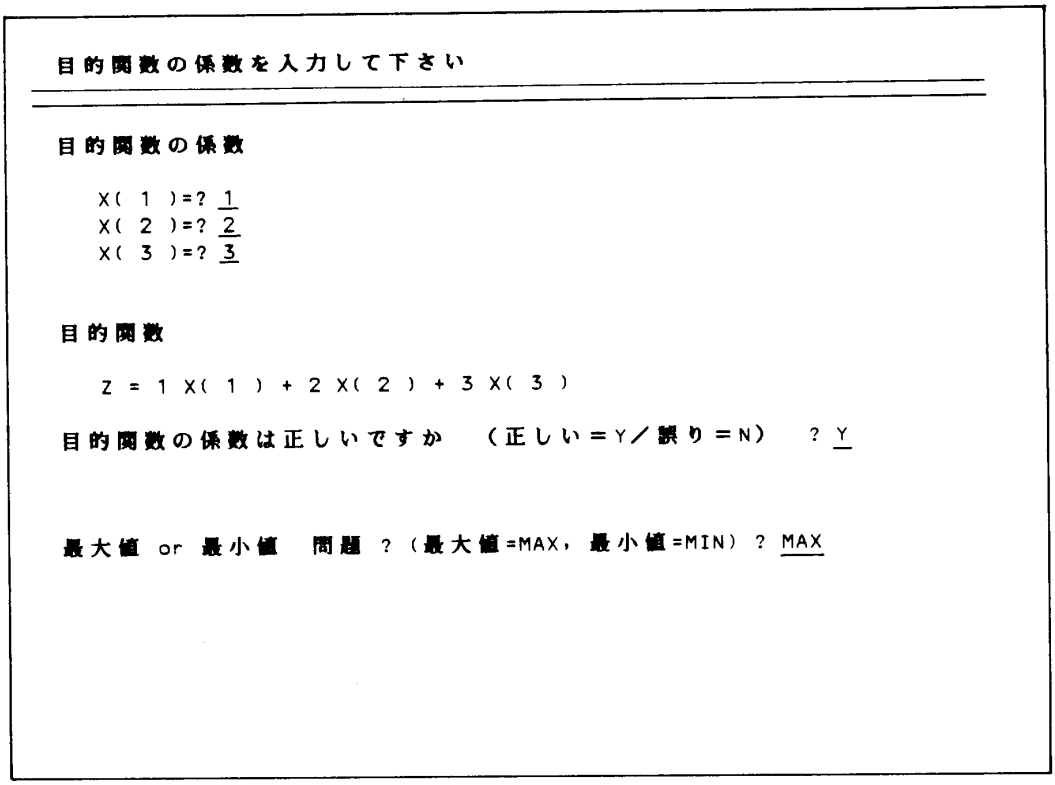

図 8.　画面表示例－６（目的関数入力）

トルについて入力の誤りがないかどうか確認を求めるようになっているので，もし入力ミスがあればそれぞれ行番号と列番号を与えて修正すればよい.(図６，図７）次に目的関数の各変数に対する係数を入力する.(図８）制約条件式，目的関数の双方について，該当の変数が無い場合（すなわち，係数が０で式から省略されている場合）があるが，その変数についてはそのまま０を入力すればよい．目的関数の係数を入力後，目的関数の式が正しいか否かの確認を求めてくるので，もし誤っていれば修正が可能である．最後に，最大値問題か最小値問題かを問い合わせてくるので応答すれば，入力は終了である．

4．2 単体表の作成

4．1で入力したデータを使用して単体表を作成する．単体表の大きさは，制約条件式の数，変数の数および条件式の等・不等号の数により決定する．

行数＝制約条件式の数＋４

列数＝変数の数＋スラック変数の数＋人為変数の数＋３

行数の ＋4 というのは，表１において v_j の行，すなわち目的関数の変数の係数値を入れておく行，ベクトル行，目的関数値を表わす z_j 行および利益を表わす (z_j-v_j) 行である．列数の ＋3 は，表１の左端の基底変数の係数列，基底変数列 v_j それと基底変数および目的関数の現在値を含むＳ列である．列数の人為変数というのは，制約条件式が例えば，

$$a_{k1}x_1+a_{k2}x_2+\cdots\cdots+a_{kn}x_n\geqq b_k \qquad (4.2.1)$$

で与えられたとき，条件式を等式化するためにスラック変数 v_k を導入すると

$$a_{k1}x_1+a_{k2}x_2+\cdots\cdots+a_{kn}x_n-v_k=b_k \qquad (4.2.2)$$

となる．ここで $\boldsymbol{x}\geqq\boldsymbol{0}$ であるから各 x_i $(i=1,2,\cdots, n)$ をすべて０と仮定する．すなわち

$$x_1,\ x_2,\ x_3,\ \cdots\cdots,\ x_n=0$$

として（4．2．2）に代入すると

$$a_{k1}\cdot 0+a_{k2}\cdot 0+\cdots\cdots+a_{kn}\cdot 0-v_k=b_k$$

$$\therefore\quad v_k=-b_k \qquad (4.2.3)$$

となる．ところが，スラック変数は（2．3）に導入したときの経緯から非負でなければならないが（4．2．3）では負になっている．これはスラック変数の定義に反するので(4．2．1）のような不等式の場合は，さらに非負の変数を加えて

$$a_kx_1+a_{k2}x_2+\cdots\cdots+a_{kn}x_n-v_k+w_k=b_k \qquad (4.2.4)$$

とすれば，もはや前述のような矛盾は起らない．ここに導入された変数 w_k のことを人為変数（artificial variable）と呼んでいる．

与えられた制約条件式および目的関数から，この単体表に数値を埋め込めばよい．

4．3 基底変数の選択とピボット演算

この部分では，次のような処理を行う．（表１参照）

```
処理過程，第 1 段階
---------------------------------------------------------------

タブローの内容


A( 1, 1) - A( 8,10)
  0.000  0.000  0.000  0.000  0.000  0.000  0.000  1.000  1.000  1.000
  0.000  0.000  0.000  4.000  5.000  6.000  7.000  1.000  2.000  3.000
  0.000  4.000 60.000  1.000  0.000  0.000  0.000  3.000  2.000  1.000
  0.000  5.000 60.000  0.000  1.000  0.000  0.000  2.000  3.000  1.000
  0.000  6.000 70.000  0.000  0.000  1.000  0.000  0.000  4.000  3.000
  0.000  7.000 30.000  0.000  0.000  0.000  1.000  1.000  0.000  2.000
  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
  0.000  0.000  0.000  0.000  0.000  0.000  0.000 -1.000 -1.000 -1.000


次のブロックに行く時はどれかキーを押して下さい ！
```

図 9. 画面表示例－7（単体表の表示）

```
処理過程，第 1 段階                                    hit any key.
---------------------------------------------------------------

基底変数選択
    目的関数の値が大きくなるように改善するために，
    次の値のうち負の最小値を選択します。

    X( 4 )=0.0000E+00    X( 5 )=0.0000E+00    X( 6 )=0.0000E+00
    X( 7 )=0.0000E+00    X( 1 )=-.1000E+01    X( 2 )=-.1000E+01
    X( 3 )=-.1000E+01

変数 X( 1 ) を選びました


ピボット選択の段階
    上で選んだ X( 1 ) を，実行するために制限となる正の最小値を選択

     X( 4 ) = 0.6000E+02 / 0.3000E+01 = 0.2000E+02
     X( 5 ) = 0.6000E+02 / 0.2000E+01 = 0.3000E+02
     X( 6 ) = 0.7000E+02 / 0.0000E+00 = 9.9999E+99
     X( 7 ) = 0.3000E+02 / 0.1000E+01 = 0.3000E+02


基底変数 X( 4 ) の代りにX( 1 ) を新たに採用します。
```

図10. 画面表示例－8（基底変数の選択）

(1) 基底変数の候補になり得る変数を探す.

(z_j-v_j) 行の中から負の最小値を探してマークする（x_1 列）. もし負の要素がなければ計算を終了する. その時点の基底変数の持っている値が, 求める最適解となる.

(2) 上でマークした列の各数値でS列の対応した数値を割り, その結果の中で正の最小のものを探してマークする（x_3 行）.

上記(1), (2)の処理において, 各要素がどのように選択されるかを, 指示を与えることにより画面上に表示して確認することができる. 同様に単体表についても, 変化した様子を画面上に表示することが可能である.（図9, 図10）

(3) 表1の②に移って, 基底変数 x_3 を追い出して, (1) でマークした変数x_1を新たに基底変数として組み入れると共に x_1 の係数を v_i に移す. 次に, x_1 行 x_1 列の値（表1では6に相当）で, 同一行の他の列の要素を割算する.

(4) 基底変数を変更しなかったすべての行について, x_1 行の要素が0になるように演算を施す. 同一の演算を同じ行の他の要素についても行う（掃き出しを行う）.

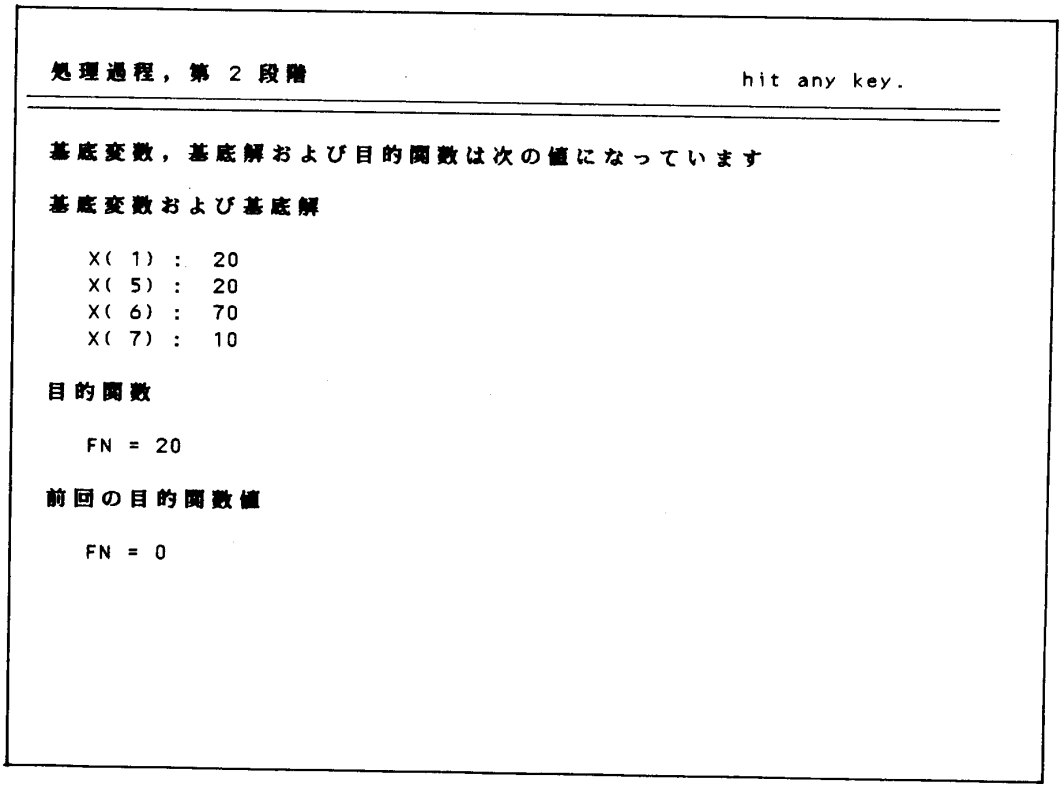

図11. 画面表示例－9（基底解の中間結果表示）

以下, 同様に上記(1)～(4)の手順を繰り返す. 最終的に (z_j-v_j) 行の各要素の値が非負になるまで行う.

4. 4 最適解出力

前述 4. 3 の基底変数選択の段階で, (z_j-v_j) 行に負の要素が無くなった時がシンプ

レックス法が終了した時，すなわち最適解が見つかった時であることはすでに述べた．図12は，最適解を出力した例であり，基底変数の値とその時の目的関数の値が表示される．

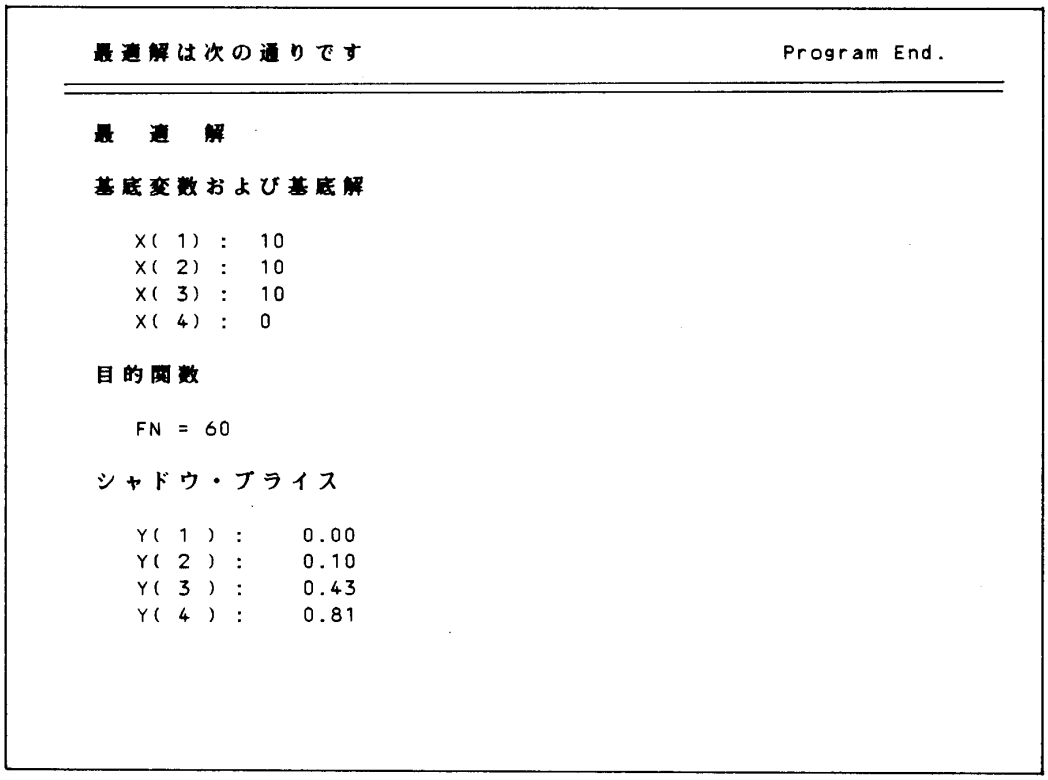

図12．　画面表示例－10（最適解の表示）

出力された結果の中のシャドウ・プライス（shadow price：替在価格）というのは，一般の線形計画問題において，制約条件式の定数ベクトルと，目的関数の係数ベクトルとを変換した時の，いわゆる双対問題（dual problem）を解く時に有用な役割を果す情報である．

5．まとめ

LP 教育用に，パソコン利用による対話型 LP 解法システムについて記述したが，LP 手法の簡易化と，より一層の普及を計るためにはパソコン利用に適したアルゴリズムの開発が不可欠であろう．本論では主として，教育用のシステムとしているが，対話形式にすることにより，より一般向きのソフトとなり得ることから，LP の細かな内容を知らずとも配分計画，混合計画，輸送計画等に利用できるものである．

LP 教育上，最も重要なピボット操作の計算法と，単体表（シンプレックス・タブロー）の作成とをわかり易く画面表示できるようにしているので，最適化の途中のプロセスが任意に取り出せるなど教育効果の向上も十分に狙ったものである．画面上に表示する関係上，取り扱う変数の数ならびに制約条件式の数には制限を受けるが，ただ結果の最適解のみを求めようと思えば，内蔵メモリの許容範囲内でかなり大規模な問題まで処理できる．しか

し現実的には，パソコンを利用するという手軽さを活かす観点からは変数の数も精々100個，条件式の数は50個程度のものが限界であろう．

今後，さらに，これに感度分析，あるいは双対解の対照表などが表示できるようにし，計画作成の分析検討アルゴリズムを付加したDSSシステム（Decision Support System）へと拡張する予定である．

参 考 文 献

[1] 牧野都治，牧野京子：「パソコンによるOR」，朝倉書店
[2] 竹中淑子：「最適値問題」，培風館
[3] 関根泰次：「数理計画法」，岩波書店
[4] 森口繁一：「数理計画法入門」，日科技連

Interactive Linear Programming Systems for Education on the Personal Computer

Toshio AOE* and Hiroyuki NARIHISA**

*Information Processing Center
Okayama University of Science
**Department of Electronics Science
Okayama University of Science
Ridaicho 1-1, Okayama 700, JAPAN

Mathematical programming is one of the optimization technipue with constraints. Linear programming is the most prevailed mathematical programming algorithms among them in accordance with the progress of the electrical computer.

In the actual planning areas, various kinds of problems can be formulated as a linear programming problem. At present, large scale of linear programming problem can be solved within reasonable economic times. However, there are many small scale of linear programming problems around our daily living or our daily social and economic activities, too.

From the above mentioned circumstances, we developped the basic linear programming algorithm for personal computer, which is also applicable to the education of the linear programming lecture.

Here, we present the interactive problem-oriented simplex algorithm of linear programming technique which is usable simplex method without advanced detail knowledge of the theory of linear programming.