[03_06]九州大学大型計算機センター広報 : 3(6)

https://doi.org/10.15017/1467971

出版情報：九州大学大型計算機センター広報. 3 (6), pp.1-70, 1970-12-18. 九州大学大型計算機センター
バージョン：

FACOM230—60 LIPS V—1について
——線型計画法のアプリケーション・プログラム——

香 田 徹* 小 野 溪 子**

1. はじめに

一般的な線型計画法を解くアプリケーションプログラムとしてFACOM230−60 LIPS V−1が近くユーザにリリースされる予定である。そこで第2節で線型計画法について、第3節で、V−1の使用法、第4節で使用例について述べる。実際の使い方を知りたい方は、第3節、第4節だけ読んで下さい。

2. 線型計画法の概要

線型計画法については、すでに数多くの文献(1)~(4)があるので、詳しくは、これらを参照されたい。ここでは、第3節で述べるLIPS V−1 の使用法との関連から、シンプレックス法、改訂シンプレックス法に関連する術語の説明を中心にして行なう。(術語は、参考文献(1)にほぼ従う。)

2.1 シンプレックス法

線型計画法とは、いくつかの線型の制約式のもとで線型の目的関数を最小または最大にする非負の最適解を求める問題である。ここで制約式とは、次の3つの型の任意の組合せである。

第一の型(またはL型) $\sum_{j=1}^{n} a_{ij}x_j \leqq b_i \ (b_i \geqq 0,\ i=1,\ \cdots,\ m_1)$ (1)

第二の型(またはG型) $\sum_{j=1}^{n} a_{ij}x_j \leqq b_i \ (b_i \geqq 0,\ i=m_1+1,\ \cdots,\ m_1+m_2)$ (2)

第三の型(またはN型) $\sum_{j=1}^{n} a_{ij}x_j = b_i \ (b_i \geqq 0,\ i=m_1+m_2+1,\ \cdots,\ m_1+m_2+m_3=m)$ (3)

また、目的関数は次のように表わせる。

$$f=\sum_{j=1}^{n} c_j x_j \tag{4}$$

ここで、a_{ij}を**制約係数**, b_iを**制約量**, c_jを**コスト係数**, x_jを**真変数**とよぶ。

また、問題によっては、真変数が、負の値をとりうる場合がある。この時は、x_jを次のように分解して

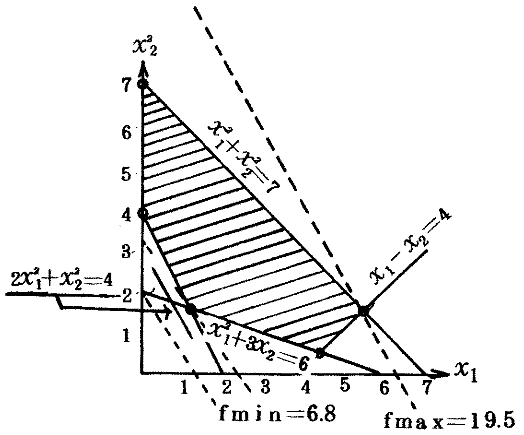
$$x_j=x_j^{+}-x_j^{-},\ x_j^{+}\geqq 0,\ x_j^{-}\geqq 0 \tag{5}$$

(1)~(4)に代入する。以上の事より、真変数x_jを非負の条件の下で解いても一般性を失わない。

まずシンプレックス法の原理を考える。簡単のため、2変数の場合について行なう。

* 九州大学工学研究科通信工学専攻

* * 九州大学大型計算機センター研究開発部



$$\text{制約式}\begin{cases} x_1+x_2 \leqq 7 & (\text{第1の型}) \\ x_1-x_2 \leqq 4 & (\ \text{〃}\ ) \\ x_1+3x_2 \geqq 6 & (\text{第2の型}) \\ 2x_1+x_2 \geqq 4 & (\ \text{〃}\ ) \\ x_1 \geqq 0,\ x_2 \geqq 0 \end{cases}$$

目的関数　$f(x_1)=3x_1+2x_2$

図：制約領域と目的関数の変化

斜線を施した制約領域を**可能解領域**とよぶ。図から明らかなように、目的関数を最適にするのは、少なくとも可能解領域の○印のついた点である。これを端点とよぶ。(最適解が一義的に定まらない場合は、(たとえば $f(x)=x_1-x_2$) 端点と端点を結ぶ直線が最適になる。) ゆえに最適解を求めるには、端点（これを**可能基底解**とよぶ）を知る必要がある。この端点は、**基底解**を求めることにより知ることができる。

この基底解は次のようにして求める。

(1)～(3) に新たな変数（これを**スラック変数**とよぶ）　$s_i\,(i=1,\cdots,m_1+m_2)$, $s_i'\,(i=m_1+m_2+1,\cdots,m)$ を導入する。

第1の型　$\sum_{j=1}^{n} a_{ij}x_j+s_i=b_i$　　$(b_i \geqq 0,\ s_i \geqq 0,\ i=1,\cdots,m_1)$　　(6)

第2の型　$\sum_{j=1}^{n} a_{ij}x_j-s_i=b_i$　　$(b_i \geqq 0,\ s_i \geqq 0,\ i=m_1+1,\cdots,m_1+m_2)$　　(7)

第3の型　$\sum_{j=1}^{n} a_{ij}x_j+s_i'=b_i$　　$(b_i \geqq 0,\ s_i' \geqq 0,\ i=m_1+m_2+1,\cdots,m)$　　(8)

目的関数　$f'=\sum_{j=1}^{n} c_j x_j+M\sum_{i=m_1+m_2+1}^{m} s_i'$　　(9)

ただし、Mは最小値問題に対しては正なる大きな値、最大値問題に対しては負なる大きな値とする。このMは $s_i'\,(i=m_1+m_2+1,\cdots,m)$ が少しでも正の値をとれば目的関数の値を最小値問題に対しては非常に大きな値にし、最大値問題に対しては非常に小さな値とする。これによって $s_i'\,(i=m_1+m_2+1,\cdots,m)$ の値は0になる(罰金法)。わかりやすくするために(6)～(9)を次のように書きかえる。

$y_1=x_1,\cdots\cdots,y_n=x_n,y_{n+1}=s_1,\cdots\cdots\cdots\cdots\cdots\ y_N=s_m'$ $(N=n+m)$　　(10)

$v_1=c_1,\cdots\cdots,v_n=c_n,v_{n+1}=0,\cdots,v_{n+m_1+m_2}=0,\ v_{n+m_1+m_2+1}=M,\cdots,v_n=M$　　(11)

$\sum_{j=1}^{n} a_{ij}y_i+y_{n+i}=b_i$　　$(i=1,\cdots,m_1)$　　(12)

$\sum_{j=1}^{n} a_{ij}y_j-y_{n+i}=b_i$　　$(i=m_1+1,\cdots,m_1+m_2)$　　(13)

$\sum_{j=1}^{n} a_{ij}y_j+y_{n+i}=b_i$　　$(i=m_1+m_2+1,\cdots,m)$　　(14)

$$f' = \sum_{j=1}^{n} v_j y_j \tag{15}$$

これらをベクトル形式に書く。

$$A\mathbf{y} = \mathbf{b} \tag{16}$$

$$f' = \mathbf{v}^T \mathbf{y} \tag{17}$$

$$\mathbf{y} \geqq \mathbf{o} \tag{18}$$

ただし

$$A = \begin{array}{c} \overbrace{\hphantom{a_{11}\,a_{12}\cdots a_{1n}}}^{n}\ \overbrace{\hphantom{1\ 1}}^{m_1}\ \overbrace{\hphantom{-1\ -1}}^{m_2}\ \overbrace{\hphantom{1\ 1}}^{m_3} \\ \begin{bmatrix} a_{11}\,a_{12}\cdots\cdots a_{1n} & 1 & & & & & & \\ a_{21}\,a_{22}\cdots\cdots a_{2n} & & 1 & & & & 0 & \\ \vdots \qquad\qquad \vdots & & & 1 & & & & \\ & & & & -1 & & & \\ & & & & & -1 & & \\ & & 0 & & & & -1 & \\ & & & & & & 1 & \\ a_{m1}\,a_{m2}\cdots a_{mn} & & & & & & & 1 \end{bmatrix} \end{array} = [\mathbf{a}_1, \mathbf{a}_2, \cdots\cdots, \mathbf{a}_n] \tag{19}$$

(16)において、N個の変数のうち、m個を残して、n個を0とおくと変数の数、制約条件の数、ともにm個となり、m個のm次元連立一次方程式になる。この方程式の解を**基底解**とよぶ(この解が一義的に定まらないときがあるが[1]、ここでは触れない)。基底解の数は、たかだか${}_N C_n$個存在する。これらの基底解のうち、各変数の値がすべて非負になっている時、これを**可能基底解(端点)**とよぶ。シンプレックス法はこれら端点のうち、ある**端点**(この最初の端点を**初期端点**とよぶ)から出発して、目的関数を改善するような端点を順次探索して最適解を得る方法である。したがって、最初に初期端点を探し出す必要がある。これを第Ⅰ段階とよぶ。次に順次よりよい端点を見つけ出す操作を第Ⅱ段階とよぶ。初めに第Ⅱ段階について述べる。

i) 第Ⅱ段階　ある一つの端点を$\mathbf{y}^1 = [y_1, y_2, \cdots, y_N]^T$とする。$y_1, \cdots, y_N$のうち、少なくともn個零があるが、このn個の零をうしろに寄せて、残りのm個を前につめ、これを新たに$\mathbf{z}^1$とする。

$$\mathbf{z}^1 = [\overset{1}{z}_1, \overset{2}{z}_2, \cdots, \overset{m}{z}_m, \mathrm{o}, \cdots \overset{n}{\mathrm{o}}]^T \tag{20}$$

この順序を入れ替えることによって、制約係数ベクトル$\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_N$が$\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_N$に、コスト係数$v_1, v_2, \cdots, v_N$が$w_1, w_2, \cdots, w_N$に変わるとする。

$$D_1 \cdot \mathbf{z}^1 = \mathbf{b}, \quad D_1 = [\overset{1}{\mathbf{d}}_1, \cdots\cdots, \overset{m}{\mathbf{d}}_m, \mathbf{d}_k, \overset{n}{\mathbf{d}}_n] \tag{21}$$

$$\tilde{D}_1 \cdot \tilde{\mathbf{z}}^1 = \mathbf{b}, \quad \tilde{D}_1 = [\mathbf{d}_1, \cdots\cdots, \mathbf{d}_m], \quad \tilde{\mathbf{z}}^1 = [\overset{1}{z}_1, z_2, \cdots, \overset{m}{z}_m]^T \tag{22}$$

$$f' = \tilde{\mathbf{w}}\,\tilde{\mathbf{z}}^1 \qquad \tilde{\mathbf{w}} = [w_1, \cdots\cdots, w_m]^T \tag{23}$$

ここで、$\mathbf{d}_1, \cdots\cdots, \mathbf{d}_m$を**基底ベクトル**、$\tilde{D}_1$を**基底行列**、$\mathbf{d}_{m+1}, \cdots, \mathbf{d}_N$を**非基底ベクトル**とよぶ。

$\mathbf{d}_1, \cdots\cdots, \mathbf{d}_N$は$\mathbf{d}_1, \cdots\cdots, \mathbf{d}_m$の一次結合として表わされる。

$$\mathbf{d}_k = \sum_{i=1}^{m} h_{ik}^1\, \mathbf{d}_i = \tilde{D}_1\, \mathbf{h}_k^1 \qquad \mathbf{h}_k^1 = [h_{1k}^1, h_{2k}^1, \cdots, h_{mk}^1]^T \tag{24}$$

ここで$\mathbf{h}_k^1$の上の添字は、基底行列$\tilde{D}_1$の下の添字に対応しているが、混乱のない限り、$\mathbf{h}_k^1$を$\mathbf{h}_k$と書く

任意の非基底ベクトルを $\mathbf{d}_k$ とすると、　(21)－(24)×θ_k　より

$$(z_1-\theta_k h_{1,k})\,\mathbf{d}_1+(z_2-\theta_k h_{2,k})\,\mathbf{d}_2+\cdots\cdots+(z_m-\theta_k h_{m,k})\,\mathbf{d}_m+\theta_k\,\mathbf{d}_k=\mathbf{b} \tag{25}$$

ここで、$\mathbf{z}^2=[\overset{1}{z_1-\theta_k h_{1,k}},\ \overset{2}{z_2-\theta_k h_{2,k}},\cdots,\ \overset{l}{z_m-\theta_k h_{l,k}},\cdots,\ \overset{m}{z_m-\theta_k h_{m,k}},\ o,\cdots,o,\overset{}{\theta_k},\ o,\cdots\overset{N}{o}]^T$　(26)

とすると、　$D_1\mathbf{z}^2=\mathbf{b}$　(27)

となる。この $\mathbf{z}^2$ が、端点 $\mathbf{z}^1$ に相隣り合う端点になるためには、

$$z_i-\theta_k h_{i,k}\geqq o \qquad i=1,2,\cdots,m \tag{28}$$

かつ、以下に示す正なる θ_k を定めると、$\mathbf{z}^2$ が新たな端点となる。

$$\theta_k=\theta_{kl}=\min_{i\in I}\ z_i/h_{i,k},\quad I=\{i\mid h_{i,k}>o,\ i=1,\cdots,m\} \tag{29}$$

すなわち、$\mathbf{z}^2$ は、ℓ 番目の要素が零となり、k番目の要素が正となる。

((29)において、$h_{k,i}<o\,(i=1,\cdots,m)$ の時は、$\mathbf{z}^2$ は端点とはならず、しいて言えば無限遠点に端点が存在する。この場合、目的関数をどれだけでも改善することができ、最大値、最小値問題に対して、それぞれ、最大値、最小値が有限な値としては存在しない。)

次に2つの端点 $\mathbf{z}^1,\mathbf{z}^2$ に対して、各々の目的関数の値を比較しよう。

$$\begin{aligned} f'(\mathbf{z}^2) &= w_1(z_1-\theta_{k,l}h_{1,k})+w_2(z_2-\theta_{k,l}h_{2,k})+\cdots\cdots+w_m(z_m-\theta_{k,l}h_{m,k})+w_k\theta_{k,l} \\ &=(w_1z_1+w_2z_2+\cdots\cdots+w_mz_m)-\theta_{k,l}(w_1h_{1,k}+w_2h_{2,k}+\cdots\cdots w_mh_{m,k}-w_k) \\ &=f'(\mathbf{z}^1)-\theta_{k,l}(u_k-w_k) \end{aligned} \tag{30}$$

ここで、$u_k=w_1h_{1,k}+w_2h_{2,k}+\cdots\cdots\cdots+w_mh_{m,k}=\tilde{\mathbf{w}}^T\mathbf{h}_k$　(31)

ゆえに、最小値問題に対して、$u_k-w_k>0$、また最大値問題に対して、$u_k-w_k<0$ なる非基底ベクトル $\mathbf{d}_k$ を新しい基底ベクトル（これを **chosen vector** とよぶ）に選び、基底ベクトル $\mathbf{d}_l$ を追い出して、新しい非基底ベクトル（これを **removed vector** とよぶ）にする。ここでより能率よく目的関数を改善するために、$|u_k-w_k|$ の大きな $\mathbf{d}_k$ を選ぶ。（話の都合上、順序は逆になったが、はじめにchosen vector d_k を選び(29)の θ_k を計算することにより、removed vector $\mathbf{d}_l$ が決まる）これらのことを繰り返し行ない、改善しようがなくなったら、（すなわち最小値問題に対しては、$u_k-w_k>0$ を満たす非基底ベクトル $\mathbf{d}_k$ がないとき、最大値問題に対しては、$u_k-w_k<0$ を満たす $\mathbf{d}_k$ がない時）その端点が最適解になる。この第II段階は、実際には、**シンプレックス表**（制約係数行列Aにおいて、ℓ 行k列目の要素を**枢軸要素**（**PIVOTAL　ELEMENT**）にして、**掃出し**を行なう）を用いて行なうが、ここでは述べない。

ii）第I段階　初期端点は、制約式が第1、第3の型のみであるなら、

$$\mathbf{y}^1=[\overset{1}{o},\overset{2}{}\cdots\cdots,\overset{n}{o},\overset{n+1}{b_1},\cdots\cdots\overset{n+m_1}{b_{m_1}},b_{m_1+m_3+1},\cdots\cdots,\overset{N}{b_N}]\quad(m_2=0) \tag{32}$$

とすれば、これが端点となる。しかし、制約式が第2の型を含むときは、簡単には求まらず、次のようにする。

(6)～(9)に導入したスラック変数の他に、さらに新たな変数 μ_i（これを**技巧変数**とよぶ）を導入する。

$$\sum_{j=1}^{n} a_{ij} x_j - s_i + \mu_{i-m_1} = b_i \qquad (b_i \geqq 0, s_i \geqq 0, \mu_{i-m_1} \geqq 0, i = m_1 + 1, \cdots, m_1 + m_2) \tag{33}$$

(6), (8), (33)の条件下で

$$g = \mu_1 + \mu_2 + \cdots\cdots + \mu_{m_2} \tag{34}$$

を最小にする新たな問題を考える。この問題の初期端点は $y_{N+1} = \mu_1, y_{N+2} = \mu_2, \cdots, y_{N+m_2} = \mu_{m_2}$ とすると、

$$Y^1 = [\overset{1}{0}, \overset{2}{0}, \cdots, \overset{n}{0}, \overset{n+1}{b_1}, \cdots, \overset{n+m_1}{b_{m_1}}, \overset{n+m_1+1}{0}, \cdots\cdots, \overset{n+m_1+m_2}{0}, \overset{n+m_1+m_2+1}{b_{m_1+m_2+1}}, \cdots\cdots, \overset{N}{b_N}, \overset{N+1}{b_{m_1+1}}, \cdots, \overset{N+m_2}{b_{m_1+m_2}}] \tag{35}$$

を選び、g＝0となる最適解を第II段階の操作を行なうことによって得る。(g＝0となる解 $\mu_i = 0$ ($i = 1, \cdots, m_2$) がない場合は、原問題は可能解を持たない。) この最適解は原問題の初期端点になっているので、これをもとにして、第II段階の操作を行なえばよい。

また次のようにしてもよい。原問題の目的関数(9)の代りに(36)の g' を目的関数をとり、

$$g' = f' + M' \sum_{i=1}^{m_2} \mu_i \qquad (\text{ただし、} M' \text{の値は(9)と同じようにきめる。}) \tag{36}$$

初期端点としては、(35)の $\mathbf{y}^1$ をとり第II段階を用いる。

2．2 改訂シンプレックス法　初期端点の求め方は2.1節と同じである。第II段階は次のようにして行なう。　(22), (24), (31)より

$$\tilde{\mathbf{z}}^{1} = \tilde{D}_1^{-1} \mathbf{d} \tag{37}$$

$$\mathbf{h}_k^1 = \tilde{D}_1^{-1} \mathbf{d}_k \tag{38}$$

$$u_k = \tilde{\mathbf{w}}^T \mathbf{h}_k = (\tilde{\mathbf{w}}^T D_1^{-1})\ \mathbf{d}_k \tag{39}$$

$$u_k - \mathbf{w}_k = \tilde{\mathbf{w}}^T D_1^{-1}\ \mathbf{d}_k - \mathbf{w}_k \tag{40}$$

となる。非基底ベクトル $\mathbf{d}_k$ について、(40)の $\mathbf{u}_k - \mathbf{w}_k$ を計算して、新しい基底ベクトルを探し、そして、基底ベクトル $\mathbf{d}_i$ を追い出して新しい非基底ベクトルとする。これらのベクトルの選び方は2.1節と同じようにして行なう。それにともない、基底行列 $\tilde{D}_1$ が $\tilde{D}_2$ に変わる。ここで $\tilde{D}_1 = [\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_l, \cdots, \mathbf{d}_m]$, $\mathbf{h}_{k1}^1 = \mathbf{h}_k$,

$\tilde{D}_2 = [\mathbf{d}_1, \mathbf{d}_2, \cdots\cdots, \mathbf{d}_k, \cdots\cdots, \mathbf{d}_m]$ で、これらの関係は次の通り、

$$B_1 = \begin{array}{c} \begin{array}{cccc} 1 & 2 & & m \end{array} \\ \begin{bmatrix} 1 & 0 & & 0 \\ 0 & 1 & & 0 \\ 0 & 0 & \mathbf{h}_{k1}^1 & \vdots \\ \vdots & \vdots & & 0 \\ 0 & 0 & & 1 \end{bmatrix} \end{array} \qquad (m\text{次の正方行列}) \tag{41}$$

とすると、

$$\tilde{D}_1 B_1 = \tilde{D}_2 \tag{42}$$

ゆえに、

$$\tilde{D}_2^{-1} = B_1^{-1} \tilde{D}_1^{-1} \tag{44}$$

さらに基底ベクトルの入れ替えを行なうと、

$$\tilde{D}_3^{-1}=B_2^{-1}D_2^{-1} \qquad B_2=\begin{array}{c} \begin{array}{cccc} 1 & 2 & k_2 & m \end{array} \\ \begin{bmatrix} 1 & 0 & & 0 \\ 0 & 1 & & 0 \\ 0 & \vdots & \mathbf{h}_{k_2}^2 & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & 1 \end{bmatrix} \end{array} \tag{45}$$

p回基底ベクトルの入れ替えを行なって最適解（その端点 $\tilde{\mathbf{z}}^{p+1}$ が最適解かどうかの判断は、シンプレックス法と同様に(40)の $\mathbf{u}_k-\mathbf{w}_k$ の符号をすべてkについて調べる。）を得たとすると、その最適解 $\tilde{\mathbf{z}}^{p+1}$ は、

$$\tilde{\mathbf{z}}^{p+1}=\tilde{D}_{p+1}^{-1}\mathbf{b} \tag{46}$$

$$\tilde{D}_{p+1}^{-1}=B_p^{-1}\tilde{D}_p^{-1} \qquad B_p=\begin{array}{c} \begin{array}{cccc} 1 & 2 & k_p & m \end{array} \\ \begin{bmatrix} 1 & 0 & & 0 \\ 0 & 1 & \mathbf{h}_{k_p}^p & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & 1 \end{bmatrix} \end{array} \tag{47}$$

となる。

以上のことから、D_i^{-1},B_i^{-1}を知ると、シンプレックス法と同じように、これらの操作（逆行列の掛算）を繰り返し行なうと最適解が得られることがわかる。この方法を改訂シンプレックス法とよぶ。

3．　FACOM　LIPS　V－1[5]について

3．1　V－1の機能・特長

(1)制約係数・制約量・コスト係数の非零要素のみを係数入力とすればよい。

(2)制約式において、スラック変数・技巧変数を入力としていれる必要はなく、制約条件式の型を指定しさえすればよい。

(3)解法は計算機に適した改訂シンプレックス法である。

(4)基底行列 $\tilde{D}$ の逆行列 $\tilde{D}^{-1}$ の保存は**プロダクトフォーム**である。

2-2節より新しい基底行列は、前の基底行列と、行列Bを知ればよい。p回基底ベクトルの入れ替えを行なえば、(新しい基底行列の逆行列 $\tilde{D}_{p+1}^{-1}$ は、) $\tilde{D}_{p+1}^{-1}=B_p^{-1}\tilde{D}_p^{-1}$ となるのであるが、前の基底行列 $\tilde{D}_p^{-1}$ を計算せずに、

$D_{p+1}^{-1}=B_p^{-1}B_{p-1}^{-1}\cdots\cdots B_1^{-1}D_1^{-1}$ より、$B_1,B_2,\cdots\cdots,B_p$ をもとにして計算を行なえばよいことがわかる。これをプロダクトフォームとよぶ。

すると、

$$B_i=\begin{array}{c} \begin{array}{ccccc} & & k_i & & \end{array} \\ \begin{bmatrix} 1 & 0 & \vdots & & \vdots & 0 \\ 0 & 1 & & & & 0 \\ \vdots & \vdots & & \mathbf{h}_{ki} & & 0 \\ \vdots & \vdots & & & & \vdots \\ 0 & 0 & \vdots & & \vdots & 1 \end{bmatrix} \end{array}$$

より、h^{1}_{k1}，h^{2}_{k2}，………，h^{p}_{kp}だけを記憶しておけばよいことになる。この　h^{i}_{ki}をη **ベクトル**とよぶ。

(5)**マルチプライシング**の機能がある。（新たに基底に入れる非基底ベクトルのいくつかをインコアにバッファリングしてサブセットを作る。そのサブセットをもとにして、最適解を求める。その最適解が、全体の変数で最適になっているかどうかを調べる。なっていなかったら新しいサブセットのもとで同じことを繰り返す。このようにすると、計算がインコアで行なえ、計算時間を短縮できる。）

(6)**リインバーション**の機能がある。　プロダクト法において、基底の入れ替え数pが大きくなると$\widetilde{D}_p^{-1}$を逐次B_1，B_2，……，B_pをもとにして計算するのは時間がかかるので何回か基底を入れ替えると初期基底行列$\widetilde{D}_p^{-1}$を作り直す。するとηベクトルの数を減らすことができる。

(7)**クラッシング**の機能がある。　技巧変数をなるべく早く基底変数から追い出して第I段階での操作を簡単にする。

(8) **Dualな解**が求められる。

(9)マルチＲＨＳ（制約量）、マルチコストができる。（制約量，コストを変えたときの解が求まる）

(10)制約量，コストのレンジングができる。（最適基底の内容が変わることなく、ＲＨＳ；コストの値が変化できる範囲を求め、それらの値を越えたとき、基底から出て行く変数名を求めることができる。）

以上のような特長があるが、しかし次のような制約がある。

(1)入力データ形式がカードのみである。

(2)カードフォーマットには、かなりの制約がある。

(3)出力としては、ラインプリンターのみである。

(4)パラメトリックＬＰなどの感度分析は行なえない。

以上4つが大きな欠点であるが、近い将来、(1)を解決する機能を持ったＶ－2のシステムがユーザにリリースされる予定である。

3．2　Ｖ－１の使用方法

ユーザが用意すべきデータを大別すれば次の三通りである。

(1)コントロールカード

(2)コントロールデータ

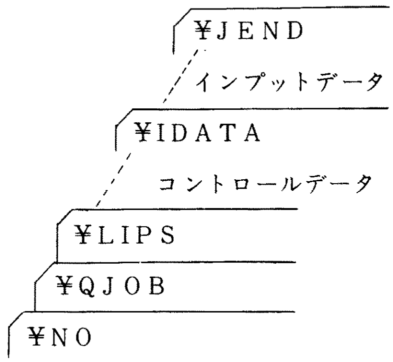
(3)インプットデータ

3．2 1　コントロールカード

コントロールカードはLIPSシステムを動作させるために、モニターに指示を与えるカードである。

col．1から col．72の間に次の要領で書く。

¥NO
¥QJOB
¥LIPS
（コントロールデータ）
¥IDATA
（インプットデータ）
¥JEND



3．2．2　コントロールデータ　col．1から col．72までの間に次の要領で書く。

(1)コメントカード　col．1に“＊”のあるカードでユーザが注釈文として用いる。コントロールデータの中にでも、インプットデータの中にでも入れて用いることができる。

(2)コントロールカード　コントロールデータのはじまりを次のように書く、p1 には“MIN”または“MAX”を入れ、それぞれ最小値問題、最大値問題を示し、p2はインプットデーターをリストするか否かをそれぞれ、“LIST”，“NOLIST”で指示する。p1, p2 のパラメータは省略でき、そのときは、それぞれ前者の方をとる。

CONTROL (p1, p2)

(3)プロセデュアカード　これらのカードを省くと標準のプロセデュアで行なう。

(*a*)クラッシングカード　クラッシングを行なうカード　CRASH

(*b*)制約量レンジングカード　制約量に関する感動分析を行なうカード　RANGERHS

(*c*)コストレンジングカード　コストに　〃　COSTRHS

(4)代入文カード　代入文カードは、LIPS 60システムに問題を解く計算上の指示を与える。

代入数値の書き方は、FORTRAN の記法に従う。

X −変数名	代入数値の型	内　容	省略したときの値
XEPS	Real	誤差判定基準(この値以下は零とみなす)	2.0 E −12
XCRITE	Real	収束判定基準(目的関数の相対係数がこの値以下は零)	2.0 E −10
XNOINV	Integer	インバートを行なう掃出し回数	50
XNOUT	Integer	何回掃出しを行なえば中間結果を出すかを示す数	1
XPRICF	Integer	a_{ij} ベクトルをコア内に読み込む本数	10

例：
XPRICE＝8
XEPS＝1.0E −10

(5)データ転送カード　与える問題の、問題名、目的関数名、制約量名を，“▼……▼”のように両端を囲んだ8文字以内（ブランクも含む）の文字列で与える。

変 数 名	内 容	省略した時の名
XPROB	問題名	▼PROB 1▼
XOBJ	目的関数名	▼OBJ 1▼
XRHS	制約量名	▼RHS 1▼

例：XPROB＝▼PROBLEM▼

(註)"▼" はEL型，H型では "@" を用いる。

(6)タイトルカード　問題に対する標題（ラインプリンターに出力されるとき、初めに打出す。）を "(……)" のように両端を囲んだ64字以内の文字列であらわす。

例：TITLE (SENKEI KEIKAKU)　このカードを省略したとき。 TITLE (LINEAR PROGRAMMING PROBLEM)

(7)コントロールエンドカード　コントロールデータの最後を示す。

CEND

以上7種類のカードがあり、(2) と (7) は必ず用意しなければならない。ほかは省略可能である。

3.2.3 インプットデータ　制約行列の行・列名、制約量・目的関数名、それらの値を与える。

(1)見出しカード　このカードは、その後に続くデータカードの種類を示し、col .1 から書く。

ROWS　このカードの後、制約式・目的関数名を与える。

COLUMNS　〃　〃　、変数名，非零の a_{ij} ベクトル・コスト係数を与える。

RHS　〃　〃　、制約量を与える。

ENDATA　インプットデータの終わりを示す。

(2)データカード

a．NAMEカード　例に示すように col .1 から col .4 の間に "NAME" と書き、col .15から col .22の間に、すでにコントロールカードXPROBで与えた8文字以内と同じ内容を書く。

```
     1 2 3 4 5 ・・・・・ 15 16 17 18 19 20 21 22
例： NAME           PROBLEM
```

b．ROWSデータカード　左に示しているように、col .5 から col .12の間に目的関数名、第1，第2，第3の制約式名を書き、col .2 から col .4 の間に、それぞれに対応して "N", "L", "G", "E" を書く。ただし、目的関数名は、すでにコントロールカードXOBJで与えた8文字以内と同じ内容を書く。ほかに、N，L，G，E に対応する所にDN，DL，DGなどと書くことができるが(5) ここでは省略する。

```
1 2 3 4 5 6 7 8 9 10 11 12
  N   COST
  L   SEIYAKU1
  G   SEIYAKU2
  E   SEIYAKU3
  ↑    ↑
制約式型名  制約式名
```

c．COLUMNSデータカード　a_{ij} ベクトルの列名（すなわち変数名）を定義し、この列名により、制約係数行列要素の値、コスト係数値のうち非零要素を与えるだけでよい。またスケールすることができるが、ここでは省略する。数値の型は、整数、実数型のどちらでもよく、FORTRANの記法に従う。

1・・・・4	5・・・・12	13　14	15・・・・22	23　24	25・・・・36	37 38 39	40・・・・47	48　49	50・・・・61	62・・72・・80
フィールド1	フィールド2	BL.	フィールド3	BL.	フィールド4	BL.	フィールド5	BL.	フィールド6	BLANK
ブランク	変数名	ブランク	制約式名1	ブランク	値　1	ブランク	制約式名2	ブランク	値　2	ブランク

←オプションフィールド→

このデータカードの順序は次のような規則がある。

ある列（変数）の名前を定義し、すでにROWSデータカードで定義した行名のうち、非零要素のみについて、行名、数値をそれぞれ、フィールド3･4、次にまた同様に、フィールド5･6に書く。（但し、フィールド5・6はオプションフィールド、その列に関して、2つ以上の非零要素があれば再びフィールド2でその列名を書き、残りの非零要素を同じようなフォーマットで書く。このようにして非零要素の制約係数を書いたあと、その変数に対する目的函数が非零であれば、フィールド3・4またはフィールド5・6に書く。このことをそれぞれの変数に対応してカードを作る。詳しくは、第4章の実例を参照のこと。

インプット・データの順序が上のような規則と違うと、エラーメッセージを打ち出さずに、読み飛ばす場合があるから十分注意を払う必要がある。

4．V－1の使用例

```
F230-60  LP  CONTROL  SOURCE  STATEMENT  LIST  AND  ERROR  LIST                    DATE 45. 8.21.     PAGE   1

1          CONTROL(MAX,LIST)
2          XPROB='PROB-002'
3          XRHS='DUALRHSN'          コントロールデータ
4          XOBJ='DUALSUMV'
5          CEND

FACOM 230-60 LP  V-1 L-1            LINEAR  PROGRAMMING  PROBLEM               45-08-21     PAGE    1
                                                      └→TITLE Type out:
      - CONTROL RESULT CONDITION -

             PROBLEM TYPE            ....  MAX,LIST

             PROBLEM NAME            ....  PROB-002

             OBJECT.FUNC NAME        ....  DUALSUMV

             RHS NAME                ....  DUALRHSN

             PROBLEM TITLE           ....          LINEAR  PROGRAMMING  PROBLEM  コントロールデータでタイトルカードを省
                                                                                 いたので省略した場合のTITLEとなる

         *** OPTIONAL PROCEDURE ***   プロセデュアカードを省いたのでSTANDARDとなる。

             STANDARD

         *** VALUE OF CONSTANT  ***   省略したときの値となる。

             VALUE OF EPSIRON      .... 0.20000000E-11          VALUE OF CRITERION   .... 0.20000000E-09

             NUMBER OF INVERT      ....             50          NUMBER OF PRICE      ....             10

             NUMBER OF OUTPUT      ....              1          NUMBER OF SAVE       ....          10000

ELAPSE  TIME **** 00H 00M 11S       TOTAL  TIME **** 00H 00M 11S  前者はコア占有時間、後者は累計コア占有時間
```

```
* INPUT DATA LIST *
        *                                                                  L 2     1
        *                                                                  L 2     2
        NAME           PROB-002                                            L 2     3  －コントロールデータで与えたのと同じ内容にする。
        ROWS                                                               L 2     4
         N  DUALSUMV                                                       L 2     5  －コントロールデータで与えた目的関数名と同じ内容
         G  DUALRW 0                                                                   にする。
         L  DUALRW 1
         L  DUALRW 2
         G  DUALRW 3
         L  DUALRW 4
        COLUMNS                                                            L 2    11
            DUALV  1  DUALRW 0      1.00000                                L 2    12
            DUALV  1  DUALSUMV      1.0
            DUALV  2  DUALRW 0     -1.00000   DUALRW 1      1.00000        L 2    13
            DUALV  2  DUALSUMV      1.0
            DUALV  3  DUALRW 0     -1.00000   DUALRW 2      1.00000        L 2    14
            DUALV  3  DUALSUMV      1.0
            DUALV  4  DUALRW 0     -1.00000   DUALRW 2      1.00000        L 2    15
            DUALV  4  DUALRW 1      1.00000                                L 2    16
            DUALV  4  DUALSUMV      1.0
            DUALV  5  DUALRW 0      1.00000   DUALRW 3      1.00000        L 2    17
            DUALV  5  DUALSUMV      1.0
            DUALV  6  DUALRW 0      1.00000   DUALRW 3      1.00000        L 2    18
            DUALV  6  DUALRW 1     -1.00000                                L 2    19  コスト係数は、制約係数の後に変数
            DUALV  6  DUALSUMV      1.0                                                ごとに入れる。
            DUALV  7  DUALRW 0      1.00000   DUALRW 3      1.00000        L 2    20
            DUALV  7  DUALRW 2     -1.00000                                L 2    21
            DUALV  7  DUALSUMV      1.0
            DUALV  8  DUALRW 0      1.00000   DUALRW 3      1.00000        L 2    22
            DUALV  8  DUALRW 2     -1.00000   DUALRW 1     -1.00000        L 2    23
            DUALV  8  DUALSUMV      1.0
            DUALV  9  DUALRW 0     -1.00000   DUALRW 4      1.00000        L 2    24
            DUALV  9  DUALSUMV      1.0
            DUALV 10  DUALRW 0     -1.00000   DUALRW 4      1.00000        L 2    25
            DUALV 10  DUALRW 1      1.00000                                L 2    26
            DUALV 10  DUALSUMV      1.0
            DUALV 11  DUALRW 0     -1.00000   DUALRW 4      1.00000        L 2    27
            DUALV 11  DUALRW 2      1.00000                                L 2    28
            DUALV 11  DUALSUMV      1.0
            DUALV 12  DUALRW 0     -1.00000   DUALRW 4      1.00000        L 2    29
            DUALV 12  DUALRW 2      1.00000   DUALRW 1      1.00000        L 2    30
            DUALV 12  DUALSUMV      1.0
```

```
 DUALV 13  DUALRW 0     -1.00000   DUALRW 4     1.00000          L 2   31
 DUALV 13  DUALRW 3     -1.00000                                 L 2   32
 DUALV 13  DUALSUMV      1.0
 DUALV 14  DUALRW 0     -1.00000   DUALRW 4     1.00000          L 2   33
 DUALV 14  DUALRW 3     -1.00000   DUALRW 1     1.00000          L 2   34
 DUALV 14  DUALSUMV      1.0
 DUALV 15  DUALRW 0     -1.00000   DUALRW 4     1.00000          L 2   35
 DUALV 15  DUALRW 3     -1.00000   DUALRW 2     1.00000          L 2   36
 DUALV 15  DUALSUMV      1.0
 DUALV 16  DUALRW 0     -1.00000   DUALRW 4     1.00000          L 2   37
 DUALV 16  DUALRW 3     -1.00000   DUALRW 2     1.00000          L 2   38
 DUALV 16  DUALRW 1      1.00000                                 L 2   39
 DUALV 16  DUALSUMV      1.0                                     L 2   40
*
RHS
 DUALRHSN  DUALRW 0      1.00000                                 L 2   41
 DUALRHSN  DUALRW 1      1.00000                                 L 2   42
 DUALRHSN  DUALRW 2      1.00000                                 L 2   43
 DUALRHSN  DUALRW 3      1.00000                                 L 2   44
 DUALRHSN  DUALRW 4      1.00000                                 L 2   45
*
ENDATA
```

—制約量名はコントロールデータで与えたのと同じ名前にする。

	制約式名／変数名	DUALV1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		DUALRHSN
制約式	DUALRW0	1	-1	-1	-1	-1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	≤	1
	DUALRW1		1		1		-1		-1		1		1		1		1	≤	1
	DUALRW2			1	1			-1	-1			1	1			1	1	≤	1
	DUALRW3					1	1	1	1					-1	-1	-1	-1	≥	1
	DUALRW4									1	1	1	1	1	1	1	1	≤	1
																		≤	
目的関数	DUALSUMV	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		

NUMBER OF ELEMENTS IN NONCONSTRAINED ROW(S). —目的関数名とそのコスト係数の非零要素数

1. DUALSUMV .... 16 —"1"は目的関数の番号を示す。(マルチコストをさせていないので)"16"は非零要素数

NUMBER OF ELEMENTS IN CONSTRAINED ROWS. —制約係数名とその係数の非零要素数)

COUNT	ROW	COUNT	ROW	COUNT	ROW	COUNT	ROW	COUNT	ROW	COUNT	ROW
16	DUALRW 0	8	DUALRW 1	8	DUALRW 2	8	DUALRW 3	8	DUALRW 4		

└→制約式DUALRW0については、非零の数は16である

NUMBER OF ELEMENTS BY COLUMN ORDER. —変数の方からみた、制約係数の非零の数

COUNT	NAME	COUNT	NAME	COUNT	NAME	COUNT	NAME	COUNT	NAME	COUNT	NAME
1	DUALV 1	2	DUALV 2	2	DUALV 3	3	DUALV 4	2	DUALV 5	3	DUALV 6
3	DUALV 7	4	DUALV 8	2	DUALV 9	3	DUALV 10	3	DUALV 11	4	DUALV 12
3	DUALV 13	4	DUALV 14	4	DUALV 15	5	DUALV 16				

└→変数DUALV11は3つの制約式について非零係数がある。

DENSITY OF PROBLEM IN PERCENT 60.000 —制約係数の非零要素が占める割合(充満率)が60%である。

THIS PROBLEM HAS 48 ELEMENTS, 16 VECTOR AND 5 ROWS. —この問題は非零要素が48個あり16変数で5個の制約式をもつ。

$$\left(\frac{48}{16\times 5}\times 100=60\right)$$

REFERENCE TABLE -- VARIABLE NAME AND COLUMN NUMBER -- —変数名と制約式名に番号をつけ、制約式名については、スラック変数・技巧変数の必要の有無をつける。("S")　("A")

NUMBER	A.S	NAME	NUMBER	A.S	NAME	NUMBER	A.S	NAME	NUMBER	A.S	NAME	NUMBER	A.S	NAME
1		DUALV 1	2		DUALV 2	3		DUALV 3	4		DUALV 4	5		DUALV 5
6		DUALV 6	7		DUALV 7	8		DUALV 8	9		DUALV 9	10		DUALV 10
11		DUALV 11	12		DUALV 12	13		DUALV 13	14		DUALV 14	15		DUALV 15
16		DUALV 16	17	A.S	DUALRW 0	18	S	DUALRW 1	19	S	DUALRW 2	20	A.S	DUALRW 3
21	S	DUALRW 4												

└→DUALV1の変数には"1"の番号をつける。

└→DUALRW0の制約式には"17"の番号をつけ、この制約式は"G"型であるのでスラック変数と技巧変数を附加する

└→DUALRW4の制約式には"21"の番号をつけ、この制約式は"L"型であるので、スラック変数だけを附加すればよい。

ELAPSE TIME **** 00H 00M 21S　　TOTAL TIME **** 00H 00M 33S

** PHASE 1 BEGIN ** —技巧変数を入れたので第I段階をやる必要がある。第I段階開始

NO OF ITER	NO OF INFEAS	NO OF ETAS	CURRENT VALUE (W)	CURRENT VALUE (Z)	CHOSEN VECTOR	REMOVED VECTOR	R.H.S ELEMENT	PIVOTAL ELEMENT	CURRENT CRITERION
1	1	1	0.0	0.1000000E 01	5	20 A	0.1000000E 01	0.1000000E 01	-0.2000000E 01
2	0	2	0.0	0.1000000E 01	1	17 A	0.0	0.1000000E 01	-0.1000000E 01
注1	注2	注3	注4	注5	注6	注7	注8	注9	注10

-- FEASIBLE SOLUTION --

註1．イタレーション数（基底ベクトルの入れ替え回数）

註2．基底に入っている技巧変数の数。この数が“0”となるまでベクトルの入れ替えを必要とする。

註3．ηベクトルの数。この数はそれまでにリインバーションを行なわない限り“NO OF INTER”の数と一致する。

註4．第Ⅰ段階での仮の目的関数の値（第2節(34)のgに相当する。）

註5．本来の目的関数値（第2節(9)のf′に相当する。）

註6．基底にとり入れられた変数番号

註7．基底から追い出された変数番号（第Ⅰ段階では“A”のついた変数番号（技巧変数）がすべて追い出されなくてはいけない。）

註8．ピボット行のRHSの値

註9．ピボット要素の値

註10．基底にとり入れられた変数のシンプレックス基準

** PHASE 1 END ** —第Ⅰ段階の終了を示す。（実行可能解を得た。）これで解の存在の有無がわかる。もし解がなければ“***** OBJECT FUNCTION (W) IS NOT ZERO *****”、“***** INFEASIBLE SOLUTION ***** ABNORMAL END を打出す。

** PHASE 2 BEGIN ** —第Ⅱ段階の開始

NO OF ITER	NO OF INFEAS	NO OF ETAS	CURRENT VALUE (W)	CURRENT VALUE (Z)	CHOSEN VECTOR	REMOVED VECTOR	R.H.S ELEMENT	PIVOTAL ELEMENT	CURRENT CRITERION
3	0	3	.	0.3000000E 01	2	18 S	0.1000000E 01	0.1000000E 01	-0.2000000E 01
4	0	4	.	0.5000000E 01	3	19 S	0.1000000E 01	0.1000000E 01	-0.2000000E 01
5	0	5	.	0.7000000E 01	9	21 S	0.1000000E 01	0.1000000E 01	-0.2000000E 01
6	0	6	.	0.1100000E 02	8	5	0.1000000E 01	0.1000000E 01	-0.4000000E 01

└→第Ⅱ段階では、第Ⅰ段階での仮の目的関数は不要なので“・”で示す。

*** UNBOUNDED SOLUTION *** COL.NO 17 —解が無限解になっている。“17”はその時のベクトル番号である。計算は続行されない。もしこれが最適解であれば“*** OPTIMAL SOLUTION ***”と打ち出し、その後“PHASE 2 END”と打つ。

***** SOLUTION OF MAXIMUM PROBLEM ***** —解を示す。

OBJECT = DUALSUMV RHS = DUALRHSN

OPTIMUM IN 6 ITERATIONS AND 0 REINVERSIONS 6回のイタレーションでリインバーションを1回もせずに解を得た。

* * * * * * *

VALUE OF THE OBJECT FUNCTION 11.00000000 目的関数値

NO. └→行番号	BASIS NAME	VALUE OF SOLUTION └→各変数の解の値	ROW NAME └→制約式の行名	SHADOW PRICE └→双対問題の解	└→制約式の型	ORIGINAL R.H.S. └→最初に与えた制約量
1	DUALV 1	5.00000000	- DUALRW 0	1.00000000	GE	1.00000000
2	DUALV 2	2.00000000	+ DUALRW 1	2.00000000	LE	1.00000000
3	DUALV 3	2.00000000	+ DUALRW 2	2.00000000	LE	1.00000000
4	DUALV 8	1.00000000	- DUALRW 3	0.0	GE	1.00000000
5	DUALV 9	1.00000000	+ DUALRW 4	2.00000000	LE	1.00000000

└→基底変数の名前。変数名に“+”、“−”がついているときには、それぞれその変数名（行名）の正のスラック変数、負のスラック変数を意味する。

*** UNSCRAMBLE *** —スクランブルな状態の解を最初に与えた変数の順序に分類して解を打出したものである。

***** SOLUTION OF MAXIMUM PROBLEM *****

OBJECT = DUALSUMV RHS = DUALRHSN

OPTIMUM IN 6 ITERATIONS AND 0 REINVERSIONS

* * * * * * *

VALUE OF THE OBJECT FUNCTION 11.00000000

NO.	BASIS NAME	VALUE OF SOLUTION	ROW NAME	SHADOW PRICE		ORIGINAL R.H.S.
1	DUALV 1	5.00000000	- DUALRW 0	1.00000000	GE	1.00000000
2	DUALV 2	2.00000000	+ DUALRW 1	2.00000000	LE	1.00000000
3	DUALV 3	2.00000000	+ DUALRW 2	2.00000000	LE	1.00000000
4	DUALV 8	1.00000000	- DUALRW 3	0.0	GE	1.00000000
5	DUALV 9	1.00000000	+ DUALRW 4	2.00000000	LE	1.00000000

*** REDUCED COST *** —全部の変数に対する双対解

DUALV 1	0.0	DUALV 2	0.0	DUALV 3	0.0	DUALV 4	2.00000000
DUALV 5	0.0	DUALV 6	-2.00000000	DUALV 7	-2.00000000	DUALV 8	0.0
DUALV 9	0.0	DUALV 10	2.00000000	DUALV 11	2.00000000	DUALV 12	4.00000000
DUALV 13	0.0	DUALV 14	2.00000000	DUALV 15	2.00000000	DUALV 16	4.00000000
- DUALRW 0	1.00000000	+ DUALRW 1	2.00000000	+ DUALRW 2	2.00000000	- DUALRW 3	0.0
+ DUALRW 4	2.00000000						

ELAPSE TIME **** 00H 00M 13S TOTAL TIME **** 00H 00M 52S

5．参考文献

(1) 関根　泰次著　　数理計画法 I（岩波講座基礎工学）

(2) 森口　繁一、宮下藤太郎著　　線型計画法（岩波講座現代応用数学）

(3) 森口　繁一著　　線型計画法入門（日科技連ライブラリー）

(4) 刀根　薫監訳　　電子計算機のための数理計画法（H.P.キュンチ、H.G.チャッハ、C.A.チュンダー共著）（日科技連）

(5) 富士通　　FACOM 230—60　LP(V－1) 解説書

LIPS V－1のユーザーへのリリースの時期につきましては、おってセンターニュースでお知らせせします。

なお、LIPS V－1は特殊ジョブ扱いとなります。