プロジェクト選択と有効勾配法

豊田　吉顕　青山学院大学

1. 単一制約下のプロジェクト選択問題

単一制約下のプロジェクト選択問題として，投資の問題を考えてみよう．この投資の問題は次のように定式化される．

$$\text{目的関数：} \max\ f=c_1x_1+c_2x_2+\ldots+c_nx_n$$

$$\text{制約条件：} a_1x_1+a_2x_2+\ldots+a_nx_n\leqq b \qquad (1)$$

$$x_j=0 \quad \text{or} \quad 1,\ j=1,2,\ldots,n$$

$$a_j\geqq 0,\ b,c_j>0,\ j=1,2,\ldots,n$$

ここで変数 x_j は投資案 j を採用するとき1，採用しないとき0とする．また b を利用できる資金量，n を投資案の数，a_j を投資案 j の資金必要量，c_j を投資案 j の限界利益とする．

これは数理計画でナップサック問題と呼ばれ，最適解は列挙法等によっても求められるが，0－1変数の数が多くなると，計算時間が急激に増加する．

ここでは**利益率法**を用いて，有利な投資案を選択することを考えよう．利益率法ではまず各投資案についてその**利益率** $r_j=c_j/a_j$ を計算し，その高い方のプロジェクトから $P_1,P_2,\ldots,P_n$ と名前をつける．もし資金を無制限に使用することができるならば，利益率が正の投資案をすべて選択すればよい．しかし，使用できる資金に制限があるならば，その制限内でできるだけ有利な投資案を選択しなければならない．そこで**図1**に示すように，資金を横軸，利益率を縦軸にとって，利益率の高い案から順に並べれば，利益率の高さと横軸に挟まれた面積が得られる利益となるので，この面積をできる限り大きくすることを考えればよい．したがって，上述の資金制約がない場合には，利益率が正の投資案をすべて選択すれば面積は最大となり，最適解を得る．利益率が正でないものに関しては，選択される可能性のない投資案としてあらかじめ削除しておけばよい．

図2に示すように，資金制約がちょうど投資案 P_t と投資案 P_{t+1} の間にある場合には，P_t まで選択することが他のいかなる組合せよりも面積が大きくなり，これが最適解となる．

図3に示すように，資金制約が投資案 P_t を分割してしまうような場合には，P_t を選択することはできない．そこで P_t を選択しないで，P_1 から P_{t+1} までを選択したとすると，これは必ずしも最適解になるとは限らない．残った資金を P_{t+1} から P_n までの投資案の中のいくつかに投資できるかもしれないし，P_1 から P_{t-1} までの中のいくつかを選択しないで，残った資金を P_t から P_n の中のいくつかに投資した方が有利かもしれない．(もっとも問題の規模が大きい場合には，P_{t-1} まで選択しただけで

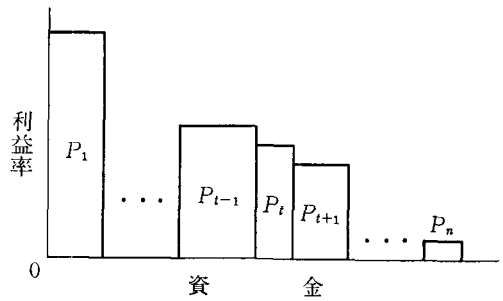

図 1　資金と利益率の関係(1)

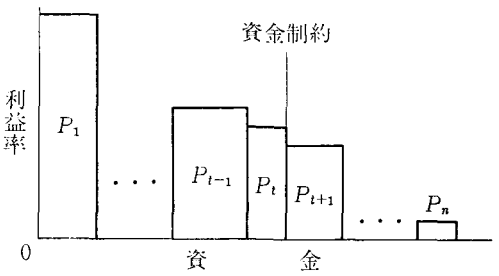

図 2　資金と利益率の関係(2)

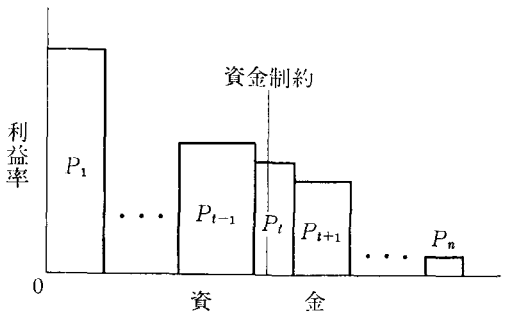

図 3　資金と利益率の関係(3)

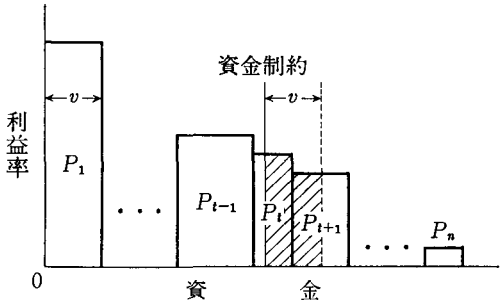

図 4　資金と利益率の関係(4)

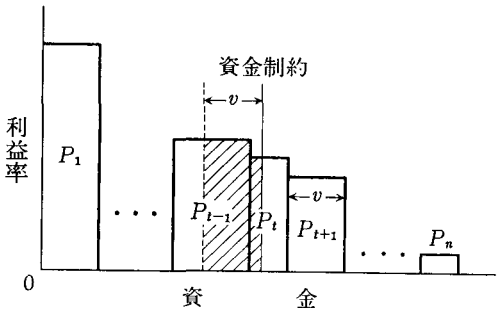

図 5　資金と利益率の関係(5)

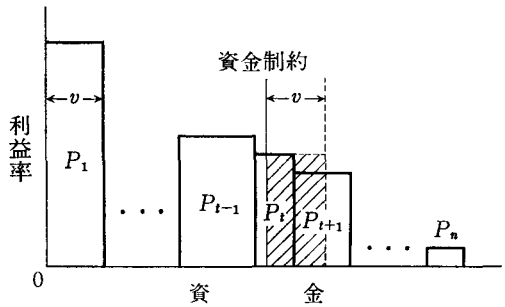

図 6　資金と利益率の関係(6)

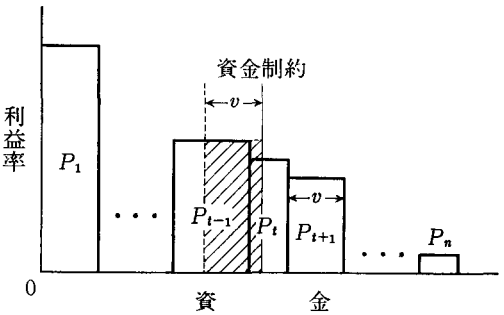

図 7　資金と利益率の関係(7)

も実用上十分な精度の近似解を得られることが多い.)

P_t を分割するような資金制約の場合,最適解は次のようにして求めることができる. すなわち, 現在の可能最良既知解は P_{t-1} まで選択したときであり,そのときの目的関数 f の値を $f^{(1)}$ とする. もし P_t を分割できることにすると, f の最大は P_t を利用可能資金量 b のところで分割した場合であり, そのときの f の値を $f^{(2)}$ とする. そうすると, 最適解における f の値 f^* は必ず $f^{(1)} \leqq f^* \leqq f^{(2)}$ を満足する. たとえばここで P_1 が最適解の中にないとすると, $f^{(2)}$ から P_1 の限界利益を取り除く代わりに増加する利益は高々**図 4** に示す斜線部分の面積である. そこでそのときの f の値 $f^{(3)}$ が $f^* \leqq f^{(3)} \leqq f^{(1)}$ となるならば上記の条件を満たさなくなり, したがって P_1 は必ず最適解の中に含まれていなければならない. 同様のことを $k<t$ のすべての P_k について調べ, 必ず最適解の中に含まれる投資案と, 含まれるか含まれないか未だわからない投資案とを分離する.

次に, もし最適解の中に P_{t+1} が含まれているとする. すると $f^{(2)}$ に P_{t+1} の限界利益が加わる代わりに減少する利益は少なくとも**図 5** に示す斜線部分の面積である. そこでそのときの f の値 $f^{(3)}$ が $f^* \leqq f^{(3)} \leqq f^{(1)}$ となるならば前述の条件を満たさなくなり, したがって P_{t+1} は最適解の中に決して含まれない. 同様のことを $k>t$ のすべての P_k について調べ, 決して最適解の中に含まれない投資案と, 含まれるか含まれないか未だわからない投資案とを分ける.

最後に上記の 2 つの手順で未だ最適解に含まれるかどうか決まらなかった投資案と残っている資金とを用いて問題を作成しなおすと規模がかなり小さくなり, これをなんらかの方法で解けばよい. 筆者の経験では, 乱数を用いて作成した n が数千の問題の場合, 70～80%以上の投資案が最適解の中に含まれるかどうかを決定することができる.

この方法は Nauss[1] の考え方にしたがったものであるが, 原論文では図 4, 図 5 の斜線部分の代わりにそれぞれ**図 6**, **図 7** の斜線部分を用いており, いくらか精度は落ちるが, そのぶん計算時間が短くなる.

利益率を用いる利点は, 最適解を求めることの他に, 各投資案を利益率という尺度で評価できることである. このことは意思決定をするさいに柔軟性をもたせることができる.

2. 多重制約下のプロジェクト選択問題

本節では制約条件が複数ある場合のプロジェクト選択

問題を考えよう．a_{ij}, b_i, c_j をそれぞれプロジェクト j が制約資源 i を必要とする量，制約資源 i の利用できる量，プロジェクト j の限界利益とする．するとプロジェクト選択問題は次のように定式化できる．

$$
\begin{aligned}
&\text{目的関数：} \max\ f=c_1x_1+c_2x_2+\ldots+c_nx_n\\
&\text{制約条件：} a_{11}x_1+a_{12}x_2+\ldots+a_{1n}x_n\leqq b_1\\
&\qquad a_{21}x_1+a_{22}x_2+\ldots+a_{2n}x_n\leqq b_2\\
&\qquad\qquad\vdots\\
&\qquad a_{m1}x_1+a_{m2}x_2+\ldots+a_{mn}x_n\leqq b_m\\
&\qquad x_j=0\ \text{ or }\ 1,\quad j=1,2,\ldots,n\\
&\qquad a_{ij}\geqq 0,\ b_i,\ c_j>0,\quad i=1,2,\ldots,m,\\
&\qquad\qquad\qquad j=1,2,\ldots,n
\end{aligned}
\tag{2}
$$

これは数理計画で多次元ナップサック問題と呼ばれている．理論的には列挙法等でこの問題の最適解を求めることができるが，単一制約下の場合と同様，0－1変数の数が多くなると，計算時間が急激に増加する．

そこで単一制約下の利益率法が，多重制約下の問題に対して拡張できれば，非常に都合が良いが，現在までのところそのまま拡張する方法は開発されていない．しかしその考え方を用いた方法として，**有効勾配法**が提案されている．有効勾配法でプロジェクト選択問題を解いた場合，必ずしも最適解が得られるとは限らないが，大規模問題に対しても，実行可能な計算時間で精度の良い近似解を求めることができる．

上述の問題(2)の制約式の両辺を b_i で割ってできた問題も原問題と等価である．そこで $h_{ij}=a_{ij}/b_i$ として原問題を正規化すると次のようになる．

$$
\begin{aligned}
&\text{目的関数：} \max\ f=c_1x_1+c_2x_2+\ldots+c_nx_n\\
&\text{制約条件：} h_{11}x_1+h_{12}x_2+\ldots+h_{1n}x_n\leqq 1\\
&\qquad h_{21}x_1+h_{22}x_2+\ldots+h_{2n}x_n\leqq 1\\
&\qquad\qquad\vdots\\
&\qquad h_{m1}x_1+h_{m2}x_2+\ldots+h_{mn}x_n\leqq 1\\
&\qquad x_j=0\ \text{ or }\ 1,\qquad j=1,2,\ldots,n\\
&\qquad h_{ij}\geqq 0,\ b_i,\ c_j>0,\quad i=1,2,\ldots,m\\
&\qquad\qquad\qquad j=1,2,\ldots,n
\end{aligned}
\tag{3}
$$

ここでは**表1**に示すような $m=2$，$n=8$ の簡単な数値例を用いて有効勾配法を説明することにしよう．有効勾配法には実行可能解から出発してその解を改良していき，最良解に到達する**主有効勾配法**と，実行不可能解から出発して実行可能解を探索していき，最良解に到達する**双対有効勾配法**とがある．

表1 プロジェクト選択問題の簡単な数値例

プロジェクト	a_{1j}	a_{2j}	c_j	h_{1j}	h_{2j}
1	6	2	100	0.250	0.067
2	2	8	400	0.083	0.267
3	6	5	600	0.250	0.167
4	4	6	800	0.167	0.200
5	9	3	300	0.375	0.100
6	3	2	200	0.125	0.067
7	5	6	400	0.208	0.200
8	1	7	500	0.042	0.233
合計	36	39	3300	1.500	1.300
制約	24	30		1.000	1.000

2.1 主有効勾配法[3]

プロジェクト選択を行なうさいには，制約資源の使用量が少なく，かつ限界利益が多いものを選択した方が有利なことはいうまでもない．しかし制約資源が複数の場合には，使用量の評価が問題である．あるプロジェクトにおいて，そのプロジェクトの制約資源使用量の評価は，その時までの制約資源の使われ具合によって変ってしかるべきである．プロジェクト1の資源使用量をベクトルを用いて $\boldsymbol{P}_1=(0.250, 0.067)$ とし，現時点までの制約資源の使われ具合を仮に $\boldsymbol{U}=(0.6, 0.3)$ であったとすると，制約資源1の方が2よりも多く使われているので，制約資源1をたくさん使うプロジェクトは2をたくさん使うプロジェクトよりも不利になるように評価しなければならない．そこで制約資源の使われ具合の割合から，制約資源1の1単位につき0.6，2の1単位につき0.3のペナルティをかけることにする．すなわち $\boldsymbol{P}_1\cdot\boldsymbol{U}=(0.250, 0.067)(0.6, 0.3)=0.250\cdot0.6+0.067\cdot0.3=0.170$ となり，これを**相対資源使用量**と呼ぶこととし，また $\boldsymbol{U}$ をペナルティベクトルと呼ぶことにする．同じプロジェクトでも $\boldsymbol{U}=(0.3, 0.6)$ の場合には $\boldsymbol{P}_1\cdot\boldsymbol{U}=(0.250, 0.067)(0.3, 0.6)=0.250\cdot0.3+0.067\cdot0.6=0.115$ となる．限界利益が同じならこの相対資源使用量が少ないほど有利なプロジェクトということができる．

ここで計算した相対資源使用量は $\boldsymbol{U}$ の大きさと方向によって変化する．そこでこれを $\boldsymbol{U}$ の大きさ $|\boldsymbol{U}|$ で割った量を考える．すると $(\boldsymbol{P}_1\cdot\boldsymbol{U})/|\boldsymbol{U}|=\boldsymbol{P}_1(\boldsymbol{U}/|\boldsymbol{U}|)$ となり，$\boldsymbol{U}/|\boldsymbol{U}|$ は $\boldsymbol{U}$ と同じ方向の単位ベクトルを表わすことから，**図8**に示すように $\boldsymbol{P}_1\cdot\boldsymbol{U}/|\boldsymbol{U}|$ は $\boldsymbol{P}_1$ の $\boldsymbol{U}$ 上への正射影の長さを表わすことになる．この量を**実質資源使用量**と定義すると，利益率法の考え方を用いて $G_1=c_1/(\boldsymbol{P}_1\cdot\boldsymbol{U}/|\boldsymbol{U}|)$ がプロジェクトの有利さを表わす尺度になるで

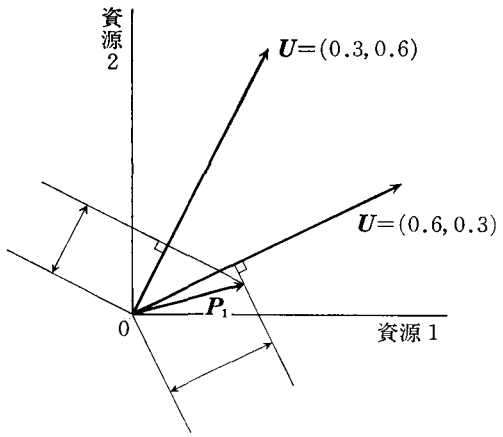

図 8　ペナルティベクトル上への正射影の長さ

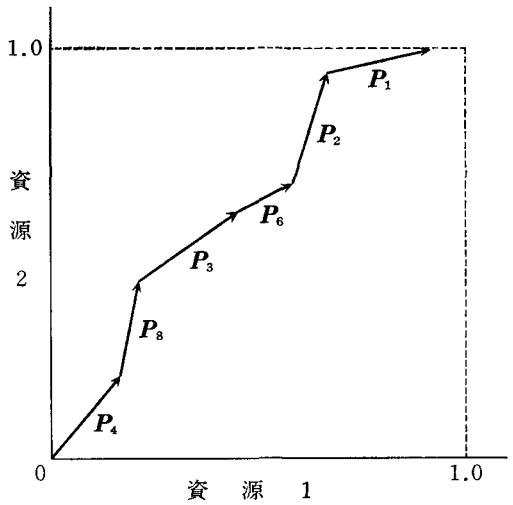

図 9　プロジェクトを選択していく過程

あろう．この G_1 を**主有効勾配法における有効勾配**と定義する．なお，ここで初期状態においてはプロジェクトを1つも選択していないので $\boldsymbol{U}=(0,0)$ となり，有効勾配の計算ができない．そこですべての制約資源の使われ具合が等しいという意味で，$\boldsymbol{U}=(1,1)$ を代用する．

このようにしてプロジェクトを選択していくと**表2**および**図9**に示すように最終的に選択するプロジェクトは4, 8, 3, 6, 2, 1であり，限界利益の和は2600となる．

2.2　主有効勾配法における原点移動法[3]

主有効勾配法において最初に $\sum_j h_{ij}>1$ なる資源 i は，プロジェクトを選択していく過程でもはや制約とはなりえない状態になっても，最後まで制約として残ってしまう．また $\boldsymbol{U}$ の長さが長くなると，U の方向の変化が緩慢になる．これらのことは問題の規模が大きくなったときに近似解の精度を悪くする傾向がある．そこでこれらの点を是正するため**原点移動法**を導入する．

たとえば $\boldsymbol{U}=(0.6,0.3)$, $\boldsymbol{P}_1=(0.2,0.4)$, $c_1=100$, $\boldsymbol{P}_2=(0.4,0.2)$, $c_2=130$ とすると，$G_1=280$, $G_2=291$ となり，プロジェクト2を選択後のペナルティベクトルは $\boldsymbol{U}=(0.6+0.4, 0.3+0.2)=(1.0, 0.5)$ となって，制約資源2にはかなり余裕があるにもかかわらず，1を少しでも使うプロジェクトは選択できなくなる．

ここで**図10**に示すように，原点を $\boldsymbol{Q}=(0.2, 0.2)$ だけ移動したとすると，ペナルティベクトルは $\boldsymbol{U}'=(0.6-0.2, 0.3-0.2)=(0.4, 0.1)$ となる．有効勾配を計算するさいに $\boldsymbol{U}$ の代りに $\boldsymbol{U}'$ を用いると，$G_1=344$, $G_2=298$ となって，$\boldsymbol{P}_1$ を選択後 $\boldsymbol{U}=(0.6+0.2, 0.3+0.4)=(0.8, 0.7)$ となる．このように原点を正の方向に移動すると，資源間の使われ具合のバランスをとる力が強くなる．ここで原点移動ベクトル $\boldsymbol{Q}$ の要素 q が0.3よりも大きな場合，たとえば $q=0.4$ の場合を考えてみよう．すると $\boldsymbol{Q}=(0.4, 0.4)$ となるので，ペナルティベクトル $\boldsymbol{U}''=(0.6-0.4, 0.3-0.4)=(0.$

表 2　プロジェクトを選択していく過程（主有効勾配法の有効勾配）

ステップ / プロジェクト	1	2	3	4	5	6
1	446	473	596	488	446	**501**
2	1616	1549	1447	1524	**1565**	
3	2036	2083	**2320**			
4	**3085**					
5	893	947	1187			
6	1475	1524	1750	**1552**		
7	1386	1394	1479	1402	1390	
8	2571	**2429**				
$\boldsymbol{U}$	(0.167, 0.200)	(0.209, 0.433)	(0.458, 0.600)	(0.583, 0.667)	(0.667, 0.933)	(0.917, 1.000)

2, −0.1) となるが，実際には**図11**に示すように $U'=(0.2,0)$ を用いるべきである．したがって，ペナルティベクトル U' の要素 u'_i は

$$u'_i=\begin{cases}u_i-q & (u_i>q)\\ 0 & (u_i\leqq q)\end{cases}$$

となる．

ただし，$q\geqq\max(u_i)$ なる値に対する原点移動は意味がないことは明らかである．筆者の経験では，$q=\{\max(u_i)\}^2$ 移度の移動を行なうのが良いようである．

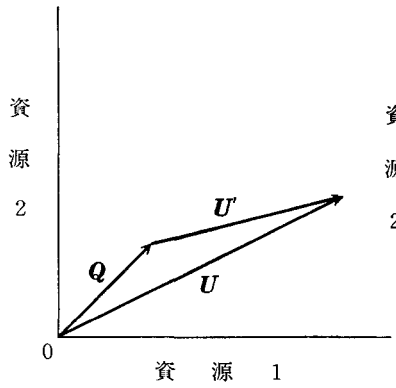

図10 原点移動とペナルティベクトル

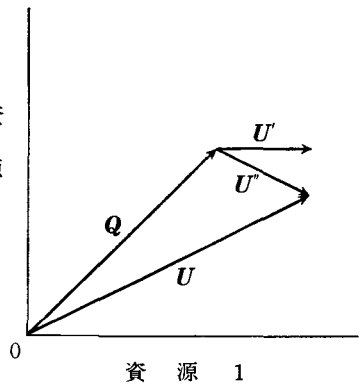

図11 ペナルティベクトルの方向

3.3 双対有効勾配法[2]

プロジェクトを選択するさい，制約資源がない場合には限界利益が正のプロジェクトをすべて選択すればよい．しかし資源が制約されていて，すべてのプロジェクトを選択できない場合には，いくつかのプロジェクトの選択を止めることによって資源の制約を満足させるとともに，その時の失う限界利益を最少にしなければならない．

主有効勾配法で用いた表1に示された数値例をここでも用いて，双対有効勾配法を説明することにしよう．すべてのプロジェクトを選択したとすると必要制約資源量は (1.5, 1.3) となり，**図12**で示す点 R となる．これは使用できる制約資源の量を超過しているので，点 R が可能領域内に入るようにプロジェクトを落していかなければならない．点 R と可能領域を結ぶ最短ベクトル S を超過ベクトルと呼ぶことにする．可能解を得るためには超過ベクトルの方向に大きく進む，すなわち**図13**に示すように S 上への正射影の長さが長いプロジェクトを落したほうが効率が良いことになる．超過ベクトルの方向に進む大きさは，そのベクトルと超過ベクトルと同じ方向の単位ベクトルとの内積で与えられる．たとえばプロジェクト1の場合 $S=(1.5-1.0, 1.3-1.0)=(0.5, 0.3)$ であるから，$P_1\cdot S/|S|=0.249$ となる．ただし，たとえ可能領域の方向へ大きく進んだとしても，その時に失う限界利益が大きくては意味がない．そこで利益率法の考えを用いて，$G_1=c_1/(P_1\cdot S/|S|)$ がプロジェクト1の有利さを表わす1つの尺度となるであろう．これを**双対有効勾配法における有効勾配**と定義する．このようにして限界利益の減少が少なくかつ可能領域へ進む距離の大きなプロジェクトの選択を止めればよい．

ここで点 R が**図14**に示すような位置になったとしよう．そうすると可能領域までの最短のベクトルは S' であるから，超過ベクトルとして S の代りに S' を用いるべきである．したがって，

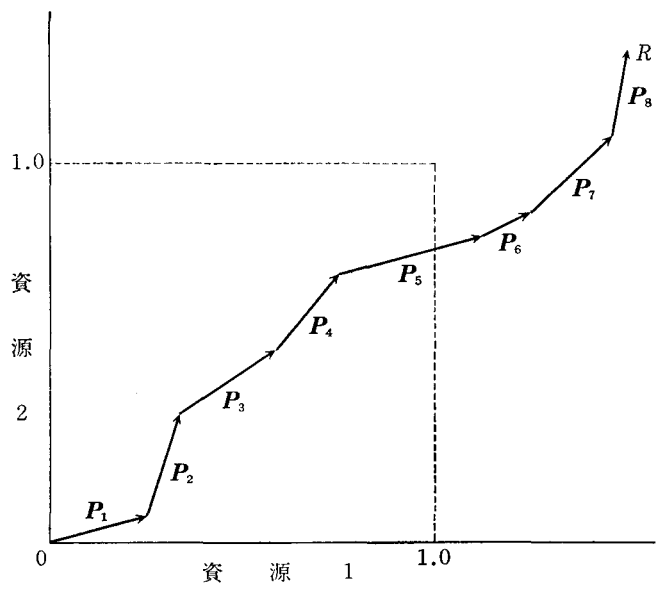

図12 すべてのプロジェクトを選択した場合の制約資源必要量

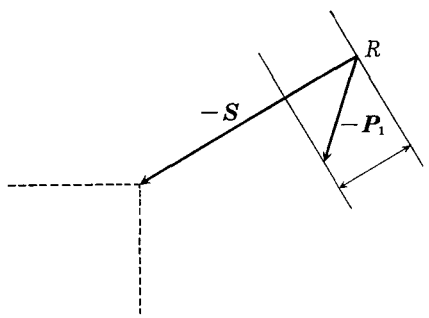

図13 超過ベクトル上への正射影の長さ

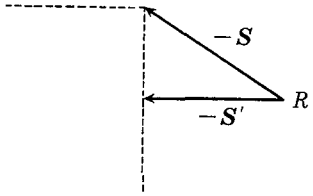

図 14 超過ベクトルの方向

超過ベクトルS'の要素S'_tは

$$s'_t = \begin{cases} s_t & (s_t > 0) \\ 0 & (s_t \leqq 0) \end{cases}$$

となる.

表1に示す数値例について，上述の方法でプロジェクトを落していく過程が**表3**および**図15**に示されている．この結果，落すプロジェクトは1,5,2であり，その時の限界利益の損失の和は800となる．このことはプロジェクト3,4,6,7,8を選択し，その時の限界利益の和は2500となることを表わしている．

いちど可能解を得た後，残った資源を制約とし，いちど落したプロジェクトの中でこの制約資源で選択できる可能性のあるものを候補プロジェクトとして，再度アルゴリズムをくりかえすことにより，解を改良することができる．

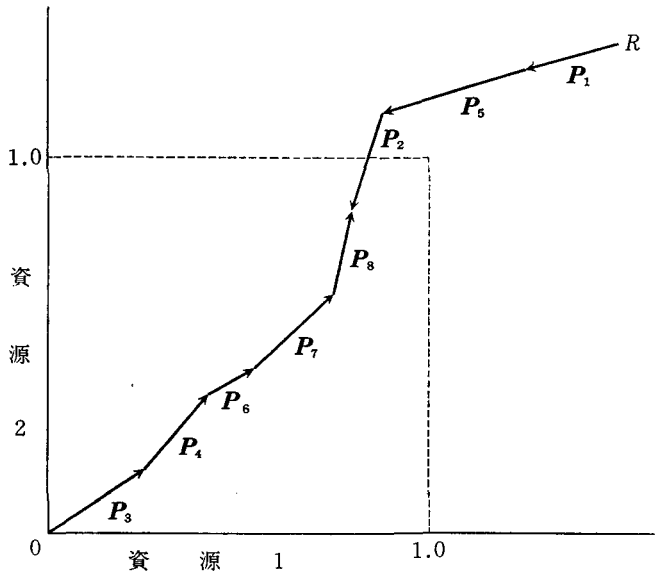

図 15 プロジェクトを落していく過程

参 考 文 献

[1] Nauss, R. M., "An Efficient Algorithm for the 0-1 Knapsack Problem," *Management Science*, Vol. 23, No.1, (September, 1976), pp. 27–31.

[2] Senju, S., and Toyoda, Y., "An Approach to Linear Programming with 0-1 Variables," *Management Science*, Vol. 15, No.4,(December, 1968), pp. B196–B207.

[3] Toyoda, Y., "A Simplified Algorithm for Obtaining Approximate Solutions to Zero-One Programming Problems, "*Management Science*, Vol. 21, No.12, (August, 1975), pp.1417–1427.

表 3 プロジェクトを落していく過程（双対有効勾配の有効勾配）

ステップ / プロジェクト	1	2	3	4
1	**402**			
2	1917	1647	**1500**	
3	1999	2024	3600	
4	3254	3097	4000	
5	804	**876**		
6	1414	1461	3000	
7	1421	1385	2000	
8	3210	2636	2143	
S	(0.500, 0.300)	(0.250, 0.233)	(−0.125, 0.133)	(−0.208, −0.133)