特集　数理計画法

線形計画法の最近の話題

岡本　吉晴・玉井　哲雄

線形計画法は，単体法の発展により，ORの分野でもっとも有力な手法の1つになっている．今日のように，数千の制約式をもつ大規模なLP問題が日常的に解かれるようになったのは，計算機のハードウエアの進展にともなって，ここ数十年の間に，汎用数理計画システムが開発され，その計算技法がいちじるしく進歩してきたからに他ならない．

ここでは，最近の汎用数理計画システムで，大規模なLP問題を効率よく解くために，どのような計算技法の工夫がなされているかを概括的に述べ，また，これらの工夫によって，どの程度の効率アップが得られているかを，若干の数値実験結果をまじえて，紹介してゆきたい．

線形計画法の計算技法の概要

今日では，大抵の大型計算機には数理計画システムがインプリメントされており，少なくとも2,000程度の制約式をもつLP問題が効率よく解けるようにつくられている．このような大規模LP問題では，制約行列は非零要素の密度が1％にも満たないような疎な行列であり，この性質を利用するために，数理計画システムでは基本的に，積型式による改訂単体法が採用されている．

積型式の改訂単体法は，基底逆行列 B^{-1} を，

$$B^{-1}=E_t\cdot\cdots\cdot E_1$$

という形で計算している．E_k は掃きだし行列である．すなわち，掃きだす列ベクトルを $\boldsymbol{a}_j$，枢軸行を r とし，

$$\eta_r=1/a_{rj}$$

$$\eta_i=-a_{ij}/a_{rj}\quad(i\neq r)$$

とすると，E_k は，単位行列の列 r が η ベクトル $\eta\equiv(\eta_1,\cdots,\eta_m)^t$ に置きかわっているような行列である．したがって，実際には，掃きだし行 r と η ベクトルの列を記憶しておけばよい．

改訂単体法の反復計算は，つぎのような手順である．

①〈BTRAN〉 π ベクトル

$$\boldsymbol{\pi}^t\equiv\boldsymbol{C}_B{}^tB^{-1}=\boldsymbol{C}_B{}^tE_t\cdots\boldsymbol{E}_1$$

を計算する．

②〈PRICE〉 非基底変数の reduced cost d_j

$$d_j\equiv C_j-\boldsymbol{C}_B{}^t\boldsymbol{a}_j=C_j-\boldsymbol{\pi}^t\boldsymbol{a}_j$$

を計算し，$d_j<0$ なる適当な j を基底候補に選ぶ．

③〈FTRAN〉 $\boldsymbol{a}_j$ の変換

$$\boldsymbol{a}_j\equiv B^{-1}\boldsymbol{a}_j=E_t\cdots E_1\boldsymbol{a}_j$$

を行なう．

④〈CHUZR〉 変化量 θ と枢軸行 r を求める．

$$\theta=\min_{\alpha_{ij}>0}\beta_i/\alpha_{ij}$$

⑤〈WRETA〉 r を枢軸とする掃きだしベクトル η_{t+1} を η ベクトル列につけ加えるという形で，基底逆行列を更新する．$t=t+1$ として①へゆく．

上記の手順で反復が進むと，η ベクトルの列がふえてきて，BTRAN と FTRAN の計算量が増してくる．このため，適当なタイミングで基底行

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

列を再逆転 (reinversion) してやり，なるべく短いηベクトル列でB^{-1}を表現しなおしてやることが行なわれる．

さて，最近の数理計画システムでは，全体の効率を上げるためにどのような工夫がなされているであろうか．これらの工夫を大別すると，

(1) 改訂単体法自身の効率を上げるもの
(2) 単体法の計算の前の処理を工夫するもの
(3) 問題の構造を利用するもの
(4) 入力データの作成と結果の出力を便利にするもの

改訂単体法の効率を上げる接近法としては，

(a) 1反復あたりの計算時間の短縮
(b) 反復回数の減少

とがある．

(a)の接近法としては，BTRAN，FTRANにおけるB^{-1}の計算を短縮することが重要である．このために，従来より，基底行列の再逆転で，三角化法とよばれる方法により，なるべく基底行列と同程度に疎なηベクトルをつくる工夫がなされている．最近では，基底行列をLU分解してL^{-1}，U^{-1}を積型式で表現することによって，反復が進んでも，ηベクトルの要素の増加がはげしくならないような工夫がなされている．またPRICEの部分でも，1度に制約行列全部をプライシングしないで，一部だけをプライシングする方式（部分プライシングという）や，プライシングで調べた$d_j<0$の非基底変数から$|d_j|$の大きな順にp個まで選んできて，それらをFTRANで変換して，その部分だけをタブロー形式で最適化してしまう方式(suboptimization とか multiple pricing とかいわれている)が従来から工夫されている．

反復回数を減らすためには，1回の反復でなるべく目的関数値の変化量の大きなものを基底に入れるように選ぶ必要がある．このため，従来よりsuboptimization のとき，p個の基底候補全部の枢軸行を計算し，目的関数値の変化量のもっとも大きいものを基底に入れるようにしている（最大増加法）．最近では，PRICEに際しても，目的関数値の変化量が大きいと期待できる変数を選ぶために，DEVEX法と称する方法が考えられ，ある程度の成果が得られているようである．

単体法の前処理の工夫としては，クラッシング技法と問題の縮小化技法がある．

クラッシング技法は，つぎのような方針で，なるべくいい初期基底を，なるべく短時間で得ることを目的としている．

① なるべく多くのスラック変数でない変数を基底に入れる．
② なるべく基底行列の形を三角化された状態に保っておく．
③ 実行不可能な行をなるべく減少させるような変数を基底に入れる．
④ 基底に入る変数の値がなるべくゼロにならないように配慮する．

問題の縮小化技法は，与えられた問題から冗長な制約式や不必要な有界変数条件をとり除き，サイズを縮小する技法である．最近の数理計画システムでは，大抵この機能がとり入れられている．

問題の構造を利用する方法としては，最近ではとくにGUB(Generalized Upper Bounding)法が数理計画システムにとり入れられて，非常に効果をあげている．たとえば Tomlin[8] によれば，50，215行（49，588 GUB行），282，468列の超大規模なLP問題が，MPSⅢのGUBプログラムを用いたら，IBM 370/165 で56分で解けたそうである．また，GUB行を数行束ねた型はDantzig-Wolfe の分解型の形になるが，この形の問題に対してはGUB法を拡張した拡張GUB法を適用できる．この方法も，筆者らの開発したプログラムでは，かなりの成果をおさめている[5]．

大規模なLP問題では，その入力データを誤りなく作成したり，結果を見やすいリポートにまとめたりするために，かなりの人手と時間が費やされている．その作業をサポートするために，多くのマトリックス・ジェネレータやリポート・ジェ

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

ネレータが開発されている．

数値実験結果

ＩＢＭの数理計画システムMPSX/370は，ＬＵ分解法による逆行列を採用しており，DEVEX法によるプライシング機能，問題の縮小化機能も有している．表１は縮小化機能の実験結果である．表２は，MPSX/360との比較（MPSX/360は，通常の積型式の逆行列を採用している）を通じて，ＬＵ分解法と DEVEX 法の有効性をテストした結果である．使用機種は，ＩＢＭ 370/168である．

表１より，縮小化技法によって，大規模なＬＰ問題の制約式は，行，列，それぞれ１～２割程度は，サイズが縮小化される．表１で，NORMAL, FREE, FIXED, BOUNDEDは，変数，あるいは制約式(スラック変数)のタイプで，

NORMAL	$0 \leq x < \infty$
FREE	$-\infty < x < \infty$
FIXED	$x = a$
BOUNDED	$l \leq x \leq u$

の意味である．

表２のηベクトルの非零要素数を見ると，ＬＵ分解法によって，基底行列の再逆転直後でも，従来のものより少なくなっているが，反復が進んで再逆転の直前になると，従来のものと比べていちじるしく少なくなっており，1/2から1/4程度である．とくに，問題のサイズが大きくなるにつれて，この差が大きくなっている．また，１反復あたりのＣＰＵ時間も，従来のものに比べて，2/3程度に短縮化されている．

DEVEX 法については，MODEL-C では，有効であったが，MODEL-B では，それほど効果はあがっておらず，MODEL-X では，逆にいちじるしく効率を落としている．

縮小化された問題に対しては，最適化の計算時間が短縮されており，縮小化に要する計算時間を加えても，全体の効率はあがるようである．とくに，問題のサイズが大きくなるにつれて，有効であろう．

表１，表２の結果からだけでは，明確なことは

表 1　問題の縮小化の実験結果

問題	モデル名	行列	TOTAL	NORMAL	FREE	FIXED	BOUNDED	ELEMENTS	DENSITY%
I	MODEL-B	行	309	63	1	245	0	2993	1.01
		列	641	548	0	61	32		
II	MODEL-B（縮小化）	行	273	55	1	217	0	2869	1.17
		列	625	428	93	83	21		
III	MODEL-C	行	715	218	2	478	17	9315	0.59
		列	1477	1445	0	32	0		
IV	MODEL-C（縮小化）	行	564	135	2	410	17	8585	0.74
		列	1472	1000	155	278	39		
V	MODEL-X	行	1645	314	1	1330	0	41048	0.28
		列	7229	5999	0	1115	115		
VI	MODEL-X（縮小化）	行	1319	229	1	1089	0	38025	0.34
		列	7079	4406	498	2090	85		
VII	MODEL-Y	行	1646	315	2	1329	0	41051	0.28
		列	7229	5979	0	1131	119		
VIII	MODEL-Y（縮小化）	行	1359	228	2	1129	0	38256	0.33
		列	7115	4389	591	2048	87		
IX	MODEL-Z	行	2561	44	1	2516	0	12765	0.08
		列	3478	2380	0	87	1011		
X	MODEL-Z（縮小化）	行	2288	44	1	2343	0	11817	0.09
		列	3400	935	1400	290	775		

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

いえないが，筆者らの経験も加味すると，つぎのようなことがいえる．

① ＬＵ分解法によって，従来のものより単体法による最適化の計算時間が，２～３割短縮される．

② 問題の縮小化は行なって損なことはない．とくに，問題のサイズが大きくなると有効である．

③ DEVEX法の有効性は，よくわからない．今後，どのようなタイプの問題に対して有効であるかを調査すべきであろう．

ＬＵ分解法，問題の縮小化技法，DEVEX法については，第５回シンポジウムの予稿集，あるいは，ＬＵ分解法は文献[2]，問題の縮小化技法は文献[1]，DEVEX 法は文献[3] を参照されたい．

ＬＰコード

最近のＬＰコード(汎用数理計画システム)は，ＬＰの最適化のみならず，入力や出力におけるさまざまなデータ処理機能，最適化後の感度分析機能，問題の保存・回復・変更機能などや，さらには，混合整数計画問題やある種の非線形計画問題の最適化機能なども備えている．代表的なＬＰコードには，つぎのようなものがある．

- MPSX ……対象機種はＩＢＭ．とくにＩＢＭ 370シリーズ用に改良されたMPSX/370が最新である
- UMPIRE …対象機種 UNIVAC
- OPHELIE…対象機種 CDC
- FMPS……Bonner & Moore 社により開発されたもので，種々の機種にインプリメントされている

これらは機能的にみると，大体似たようなものを備えており，性能的にもそれほどの違いはないと思われる．日本では，各国産機メーカーで開発しているものがある．なお，筆者らが開発に参画したものに，IMPS（日本情報処理振興協会の委託開発）があり，これは，とくに混合整数計画問題に対するヒューリスティックな解法の導入と，構造をもった線形計画問題の効率よい取りあつか

表 2　ＬＵ分解法と DEVEX 法の効率評価　　注）＊時間打ち切り

		最適化時間計算 CPU(分)	反復回数 (CRASH)	CPU時間(分) 反復	平均間隔 INV	平均 η-elements Invert 前	平均 η-elements Invert 後	縮小化計算に要した時間(分)
MODEL-B	MPSX/360	0.37	727(370)	0.0009	73	8172	2069	
	MPSX/370	0.30	699(286)	0.0006	70	L 2105 U2471	L 1050 U 758	
	MPSX/370 DEVEX	0.33	697(286)	0.0007	39	L 1619 U1500	L 1008 U 765	
縮小化 MODEL-B	MPSX/370	0.30	716(174)	0.0005	119	L 2919 U5094	L 899 U 721	0.04
MODEL-C	MPSX/360	3.79	2943(742)	0.0017	84	23205	5629	
	MPSX/370	3.44	3098(792)	0.0012	79	L 4265 U4348	L 3212 U1765	
	MPSX/370 DEVEX	2.89	2370(792)	0.0017	52	L 4162 U3498	L 3170 U1749	
縮小化 MODEL-C	MPSX/370	2.27	2500(701)	0.0012	114	L 4921 U7295	L 2401 U1876	0.14
MODEL-X	MPSX/360	2.56	674	0.0030	96	45023	7224	
	MPSX/370	2.21	817	0.0027	136	L 6049 U6074	L 4699 U2316	
	MPSX/370 DEVEX	4.96*	870	0.0056	145	L 6951 U6956	L 4718 U2367	

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

いとを特徴としている[6),7)].

これらのＬＰコードにとって，制約式の数が1,000程度の問題は中位の規模といってよく，制約式2,000～3,000，変数10,000程度までの問題は実用的に解くことができ，また実際に日常的に解かれているようである．このほかにＬＰコードの機能として比較的最近話題となっている項目として，つぎのようなものがあげられる．

(1) ＧＵＢ．ＧＵＢ型の問題に対して，ＧＵＢ技法を適用して効率的に解く機能は多くのＬＰコードに導入されており，さらに一般のＬＰ問題に対して，ＧＵＢの構造を発見して，その形にデータを配列し直す機能を備えているものもある．

(2) ＭＩＰ．混合整数計画問題を解く機能は，多くのＬＰコードが備えている．ほとんどが分枝限定法を解法として採用している．最近，需要が増えつつあるようである．

(3) データ管理．各ＬＰコードは，ＬＰ問題を入力するデータの形式をもっており，これを内部的なデータに変換して最適化を行ない，また得られた結果をリストやファイルに出力するという入出力機能をもつ．これらを補助する意味での，行列生成(Matrix Generator)やレポート生成(Report Generator)のプログラムも，かなり以前から，これらＬＰコードと別に，あるいはＬＰコードに付随してつくられてきたが，最近ますますその必要性が高くなっているようである．この他に，問題や解の保存・回復・修正なども，データ管理に含まれる．現在，このデータ管理機能の比重が増し，計算時間(費用)で，最適化部分の３倍位がこちらにかかっているという人もいる[4)].

(4) 構造のあるＬＰ．とくに分解型（ＬＰ行列の大部分がブロック三角化されているもの）の線形計画問題を効率よく解く方法の研究は多いが，ＬＰコードに採用されているのは非常に少ない．

(5) 会話型ＬＰ．時分割方式の処理用にインプリメントされたＬＰで，最近，とくにＬＰ計算にともなうデータ管理の複雑化の傾向にともない，有効性が認識されつつあるらしい．

(6) イン・コア最適化．最適化の速度を上げるためになるべくデータを主記憶内において処理しようとする工夫である．とくに行列の同じ値をもつ要素を重複してもたないという工夫により，行列を主記憶内に納める技法が FMPS の改良版に使われているのが，目新しい．

参考文献

1) Brearley, A. L., G. Mitra and H. P. Williams, "Analysis of Mathematical Programming Problems prior to Applying the Simplex Algorithm," Mathematical Programming, 8 (1975), 54—83.

2) Forrest, J. H. and J. A. Tomlin, "Updating Triangular Factors of the Basis to Maintain Sparsity in the Product Form Simplex Method," Mathematical Programming, 2 (1972), 263—278.

3) Harris, P. M. J., "Pivot Selection Methods of the DEVEX LP Code," Mathematical Programming, 5 (1973), 1—28.

4) Hirsheld, D. S., "The Practice of Linear Programming : Status and Outlook, XXII International Meeting TIMS, 1975, Kyoto, Japan.

5) 反町洋一，武川博臣，岡本吉晴，"上限値付変数を含む拡張ＧＵＢ法，" 1972年度秋季研究発表会アブストラクト集，日本ＯＲ学会，1972.

6) 反町洋一，岡本吉晴，玉井哲雄，"数理計画システムの開発，" 情報処理，15 (1974)，710—716.

7) Sorimachi, Y., H. Mukawa, Y. Okamoto and T. Tamai, "Implementation of the Heuristic Mixed Integer Program to the Mathematical Programming System," Proceedings, IX International Symposium on Mathematical Programming, Hungary, 1976, to appear.

8) Tomlin, J. A., "Survey of Computational Methods for Solving Large Scale Systems," TR 72—25, Operations Research House, Stanford Univ., 1972.

おかもと・よしはる　1946年生　三菱総合研究所
たまい・てつお　1948年生　三菱総合研究所

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．