HOKKAIDO UNIVERSITY

Title	応用線形計画システムALPS-1の設計と作成
Author(s)	樫原, 敏康; 大内, 東; 加地, 郁夫
Citation	北海道大學工學部研究報告, 95, 63-73
Issue Date	1979-08-31
Doc URL	https://hdl.handle.net/2115/41581
Type	departmental bulletin paper
File Information	95_63-74.pdf

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

北海道大学工学部研究報告
第95号 (昭和54年)

Bulletin of the Faculty of Engineering,
Hokkaido University, No. 95 (1979)

応用線形計画システム ALPS-1 の設計と作成

樫原敏康* 大内 東* 加地郁夫*

(昭和54年3月31日受理)

Applied Linear Programming System ALPS-1

Toshiyasu KASHIHARA Azuma OUCHI Ikuo KAJI

(Received March 31, 1979)

Abstract

The authors are designing and developing an Applied Linear Programming System (ALPS) which is applicable to large scale engineering problems.

This report gives an outline of ALPS, i.e., characteristics, construction and two application examples. In order to solve L.P. effectively, we use those techniques in ALPS such as the revised simplex method using the product form of inverse, the upper-bound technique, the composite primal algorithm, the dual simplex algorithm, the sparsity technique and the reinversion technique by the triangularization method. The first version of ALPS is implemented in FACOM 230-75 in FORTRAN.

1. ALPS-1 の設計思想

数理計画法はシステム工学の重要な技法の1つであり，中でも線形計画法 (LP) は最も強力なアルゴリズムである。現在多くの LP プログラム・パッケージが用意され，ユーザは LP アルゴリズムの詳細を知らなくても，フォーマットに従ってデータを入力しさえすれば，種々の結果を得ることができる。このようなシステムは，誰でもが利用できるという意味で，一般向きである。

しかしながら，近年のシステム工学の発達に伴い，その取り扱うシステムは非常に複雑なものになっている。複雑性の主要な要因の1つは，規模の大規模性にあると言ってよい。すなわち，システム要素間の関係はむしろ単純であるにもかかわらず，その規模が大きいだけで，そのシステムの取扱いが不可能になってしまう。

例えば，逆行列の計算をあげることができる。n 次正方行列の逆行列を求めるアルゴリズムは，任意の次元に対して正確に分っている。$n=2, 3, 4$ 程度は手計算でもできるし，それ以上は電卓，あるいはコンピュータを利用して計算できる。しかし，$n \geq 1000$ では，もはや最新の大型コンピュータを利用しても解けない。しかしながら，n が数千のオーダーの行列は，システム工学で取り扱う問題ではむしろ普通である。すなわち，システム工学で取り扱う問題を解くためには，最新のコンピュータをその性能の極限まで酷使した利用をしなければならない。数理計画法を熟知した者が，最新のアルゴリズムとプログラム技法を駆使して，最新のコンピュータを利用しなければ，問題は解けない。

* 電気工学科 系統工学講座

以上の観点から，著者等は従来の設計思想とは正反対の，専門家向けのプログラム・システムを目的として応用線形計画システム (Applied Linear Programming System ; ALPS) を設計し，作成した。すなわち，LP 計算に必要な種々のアルゴリズムを細かく分類し，それぞれ手続きとして独立させる。さらに LP の拡張として考えられるアルゴリズムはこれらの手続きに必要な手続きを加えて構成する，必要な手続きの取捨選択はユーザが最も効率が良いと思われる構成で行なう，等の設計思想のもとに作成した。もちろん，一般的な使用方法にも耐えうるよう，標準的構成も与えてある。

本報告は，上述の設計思想のもとに作成した ALPS の第1版について，システムの機能と構成を中心に述べる。

2. システムの特徴

本システムは，各種最適化手法にシンプレックス法・擬似シンプレックス法をインプリメントしやすいように，シンプレックス法で頻繁に用いられる基本的な処理を標準化し，基本ルーチンとして独立させている。

補助記憶装置を使用する場合，データへのアクセス時間が主記憶の場合に比べて桁違いに遅くなり，最適化手法の計算時間は使用コンピュータの機器構成・データへのアクセス方法に強く依存するようになる。このため，アルゴリズムの有効性を考え，大部分のデータを主記憶内で処理するイン・コア処理を原則とする。

実用的な大きさの問題を，高速にかつ高い精度で解くために，アルゴリズムの基礎となるシンプレックス法には，次のような技法を採用している。

1) 積型逆行列を用いた改訂シンプレックス法

基底逆行列を希薄な形で管理するために，ガウス・ジョルダン法を用いる。

2) スパース行列処理技法

係数行列・積型逆行列のスパース性を利用して，計算時間・記憶容量を節約するために，非零要素のみを格納処理する。

3) 有界変数法

LP モデルの縮小及び計算時間の短縮を計るために，有界変数・レンジ制約を変数のタイプとして扱う。

4) 複合主法

実行不可能量の絶対値の和を最小化するための複合目的関数と複数の目的行の重みつき加法和を，同時に最適化する。

5) 双対法

双対実行可能な基底解が与えられた場合（例えば，整数計画法において切断平面法が適用される時），パラメトリック法を用いる場合は主法よりも，双対実行可能性を維持しながら主実行可能（最適）解を求める双対法の方が有効である。

6) 三角化法による再逆転技法

非零要素のフィル・インが少ない積型逆行列を短時間で再計算するために，行と列に対する三角化ソート・修正 Markowitz 法を利用した Smith の三角化法を採用している。

7) 各種プライシング技法

反復回数・計算時間の短縮を計るために，部分プライシング・多重プライシング・

部分最適化を行なうことができる。

この他に，問題の解析・結果の出力をサポートする機能として，次のようなものがある。

- ○ 入力データのチェック・問題の確認のために，係数行列に関する統計量・図式表示を出力するプロセジュアを用意している。
- ○ 構造的に冗長な制約式や変数制約を取り除いて，実質的に解くべき問題の大きささを縮小し，処理時間を短縮したり，データのミス等による矛盾した制約を発見して，無駄な計算を省略する。
- ○ 係数行列の非零要素の値のばらつきによって，精度が落ちないように適当なスケール値を決定する自動スケーリングの機能がある。
- ○ 解の妥当性のチェック，結果の標準的な出力を行なう機能がある。

3. データ構造

3.1 スパース行列

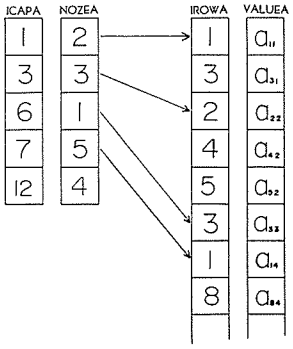
係数行列・積型逆行列のηベクトルのデータ構造としては，非零要素の行番号と要素値を列ごとにパックしたコンパクト形式で格納する行/列インデクス方式を採用している。

係数行列の非零要素を，次のような4つの配列によって表現する。

ICAPA : 各列の先頭の非零要素へのポインタ
NOZEA : 各列の非零要素の個数
IROWA : 各非零要素の行番号
VALUEA : 各非零要素の要素値

同様に，積型逆行列は次のような5つの配列によって表現する。

ICAPT : 各ηベクトルの先頭の非零要素へのポインタ
NOZET : 各ηベクトルの非零要素の個数
IR : 各ηベクトルの位置インデクス
IROWT : 各非零要素の行番号
VALUET : 各非零要素の要素値



図—1

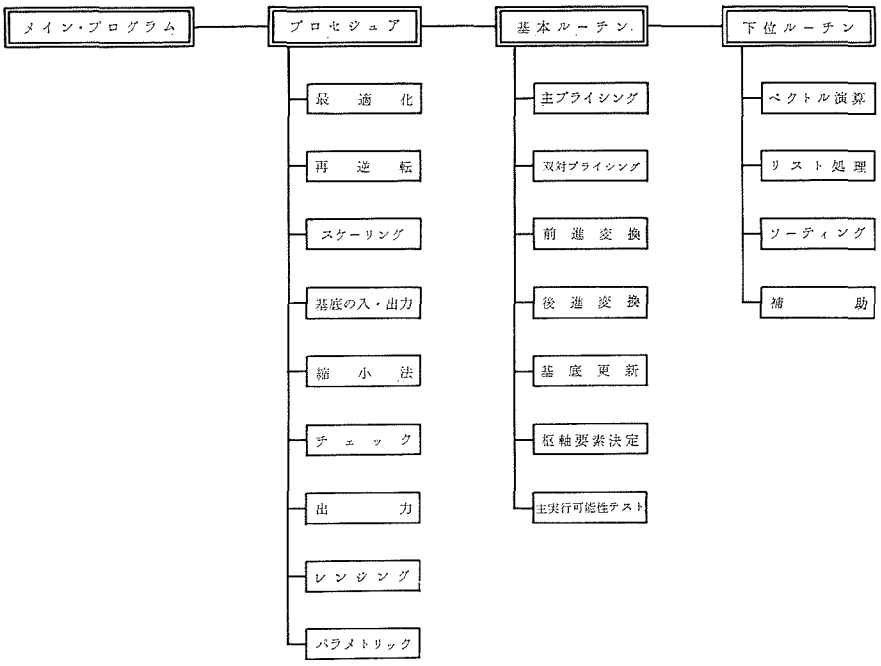
シンプレックス法および多くの科学技術計算で頻繁にあらわれる基本的なベクトル演算のために，標準形式・コンパクト形式ベクトルに対するベクトル演算ルーチンを用意している。

3.2 リスト構造

三角化法による再逆転技法では，基底行列の三角化ソートのために，行に対するノンゼロ・インデクス・リストを必要とする。行リストは，非零要素の列番号と行方向ポインタを持つ要素記述子で構成されたリスト構造を持ち，要素記述子の挿入・削除のために，リスト処理ルーチンを用意している。

4. システムの構成

ALPS-1は，次図のようにプロセジュア・基本ルーチン・下位ルーチンの3つの階層に属するサブルーチン群によって構成されている。各サブルーチンは，常に上位階層のサブルーチンかメイン・プログラムから呼び出され，下位階層からは呼び出されることがないように構成されている。ただし，同レベルの階層から呼び出される場合もある。



図—2

4.1 プロセジュア

通常ユーザの作成するメイン・プログラムから，すでに最適化手法のアルゴリズムが用意されている問題を，標準的なジョブ構成で解く場合に，直接呼び出されるルーチン群である。現在 ALPS-1 では，LP 問題と線形分数計画 (LFP) 問題に対するアルゴリズムが用意されている。

従来の LP システム・数理計画システムにおいて，ユーザのコントロール言語またはデマンドで直接実行を要求されるような，最適化計算・結果の出力等の各種機能を実現するルーチンに対応する。

4.2 基本ルーチン

シンプレックス法のアルゴリズムを構成している基本的な処理を実行するルーチン群で，ユーザが，シンプレックス法を利用した最適化手法のアルゴリズムを実現する際に組み合わされる。

4.3 下位ルーチン

特定のデータ構造 (例えば，ベクトル・リスト等) に依存した処理ルーチンや，入出力等に用いられる補助ルーチンから成り，通常ユーザから直接呼び出されることはない。

5. プロセジュア

5.1 最適化ルーチン

(1) PRIMAL　標準 LP 問題に対して，任意の基底解から出発して，実行可能解と最適解を得る主計算ルーチン。

(2) OPTIMA　PRIMAL と同様，複合主法を用いた主計算ルーチンで，部分プライシング・多重プライシング・部分最適化等の手法を用いて，最適化の能率を上げている。

(3) CRASH クラッシング技法を用いて，ほとんど論理列で構成された基底から，基底に入る候補ベクトルを順次基底に入れて，できるだけ早く意味のある基底を作り出す。

(4) DUAL 双対法を適用して，双対実行可能な基底から出発し，すべての変数タイプを許す標準 LP 問題に対して，最適実行可能解を与える。

(5) FRACT Martos のアルゴリズムを用いて，任意の符号の分母を持つ，LFP 問題に対する最適解を与える。無効点・特異点が存在すれば，解の状態を出力する。

(6) BITRAN Bitran & Novaes のアルゴリズムを用いて，正の分母を持つ，LFP 問題に対する最適解を与える。LP 化問題の解法には，PRIMAL を使用している。

5.2 再逆転ルーチン

(1) INVERT 指定した任意の基底に対する新しい積型逆行列を計算する再逆転ルーチンで，速度・精度・逆行列のスパース性の点ですぐれた Smith の三角化法を用いている。

5.3 スケーリング・ルーチン

(1) ASCALE 変数の変域を標準タイプの変域に変換し，かつ，精度を維持して計算誤差が大きくならないように，自動的にスケール値を決定して，スケールを行なう，自動スケーリング・ルーチン。

(2) SSCALE 変数の変域を標準タイプの変域に変換し，かつ，ユーザの与えたスケール値でスケールを行なう，標準スケーリング・ルーチン。

(3) SOLTN 標準タイプの基底解から，スケールと平行移動を除いて，本来の変域における値に戻す，逆スケーリング・ルーチン。

5.4 基底の入・出力

(1) INIT 論理基底を出発基底として与え，基底の初期化を行なう。

(2) INIT 2 変数のタイプを考慮して，論理基底より有効な基底を，初期基底として設定する。

(3) BSIN ユーザの指定したファイルから，カード・イメージで，初期基底を入力する。

(4) BSOUT ユーザの指定したファイルに，カード・イメージで現在の基底を BSIN の入力フォーマットと同じフォーマットで出力する。

5.5 縮 小 法

(1) REDUCE 変数・制約式が任意の変域を持つ LP 問題・線形制約式の冗長度・実行可能性等の解析を行ない，問題の縮小化を計る。

5.6 チェック・ルーチン

(1) CHEQUE 解の妥当性をチェックする機能で，計算誤差を評価して，許容値より大きければ，チェック・エラーの警告が出力される。

5.7 出力ルーチン

(1) ENTITL ページのヘッドに印刷される見出しを指定する。

(2) PAGE 改ページを行ない，見出しを印刷する。

(3) STATUS 問題の係数行列に関する統計量（非零要素の最大値・最小値・度数分布等），各変数・制約式のタイプと対応する行・列の非零要素の個数等を印刷する。

(4) NOTYPE 各変数・制約式のタイプ別の個数を印刷する。

(5) PICTUR 係数行列を簡略化した図式表示（ピクチャー）を行なう。

(6) OUTPUT　現在の基底解の主・双対変数の値，主変数の上・下限を印刷する標準出力ルーチン。

5.8 レンジング・ルーチン

(1) RNGRHS*　対応する論理変数が基底に入っていない右辺の要素を変えたとき，最適解の近傍に対して主実行可能性の範囲を計算する。

(2) RNGOBJ*　基底中の目的関数の係数を変えたとき，最適解の近傍に対する双対実行可能性の範囲を計算する。

(3) FORCE　換算費用係数(双対変数)がゼロに近い変数を基底に入れたとき，最適解が受ける影響を計算し，準最適主実行可能解を求める。

(4) REMOVE　アクティビティ・レベル（主変数）がゼロに近い変数を基底から除いたとき，最適解が受ける影響を計算し，準最適双対実行可能解を求める。

5.9 パラメトリック・ルーチン

(1) PARRHS*　増分列(変分右辺)をパラメータ倍したもので，右辺のパラメトリック計算を行なう。

(2) PAROBJ*　増分行(変分目的行)をパラメータ倍したもので，目的関数のパラメトリック計算を行なう。

6. 基本ルーチン

6.1 主プライシング・ルーチン

(1) PRICE　双対実行不可能な非基底変数で，最大の $|\alpha^r_j|$ を与える j を枢軸列 s として選択する。

(2) PPRICE　有効な候補ベクトルが q 個発見したとき，あるいは，全非基底変数を調べた時，プライシングを打ち切る部分プライシングを行なう。

(3) MPRICE　部分プライシングと同時に，有効な候補列の中から，$|\alpha^r_j|$ の大きい $p(<q)$ 個の候補列を選択する多重プライシングを行なう。

(4) CPRICE　基底変数 β^r の値を固定しながら，目的関数 u^r を増加させる候補列を選択する。

(5) RPRICE　基底変数 β^r と目的関数 u^r の変化の比率が最も大きい候補列を選択する。

6.2 双対プライシング・ルーチン

(1) DPRICE　基底変数 β^r を基底から除いた時，双対実行可能性を維持するために，基底に入るべき非基底変数 X_s を決定する。

6.3 前進変換ルーチン

(1) FTRAN　係数列（構造変数ならば構造列，論理変数ならば論理列）に対して，前進変換を行なう。

(2) FTRANH　任意の列ベクトルに対して，前進変換を行なう。

(3) FTRANM　多重プライシングで選ばれた複数の係数列に対して，前進変換を行なう。

6.4 後進変換ルーチン

(1) BTRAN　行ベクトル（例えば，目的関数行）に対して，後進変換を行なう。

(2) BTRAN 2　2つの行ベクトル（例えば，目的関数行と枢軸行）に対して，後進変換を行なう。

---

* 現在，まだインプリメントされていない。

6.5 基底更新ルーチン

(1) WRETA 枢軸要素 α_s^r で消去を行ない，基底を更新する。

(2) WRETAM 基底の更新と同時に，α ファイルの掃き出し計算を行なう。

6.6 枢軸要素決定ルーチン

(1) CHUZR 変数 X_s を基底に取り入れた時，主実行可能性を維持するために，基底から除かれる基底変数 β^r を決定する。

(2) SUBOPT 部分最適化を行なうために，多重プライシングで選ばれた複数の候補列から，最も目的関数を増加させる枢軸要素 α_s^r を選択する。

6.7 主実行可能性テスト・ルーチン

(1) PHASE 1 複合主法の phase 1 で用いられ，各基底変数の実行不可能量の絶対値の和を表わす複合目的関数に対応するプライシング・ベクトルを作成する。

(2) PHASE 2 双対法で用いられ，実行不可能量の絶対値が最大の基底変数 β^r を，基底から除かれる変数として選択する。

7. 下位ルーチン

7.1 ベクトル演算ルーチン

(1) SMA (Scalar Multiply and Add)・ルーチン $A_i \leftarrow A_i + S \times B_i$

a SMA 1 A: 標準形式，B: コンパクト形式

b SMA 2 A, B: 標準形式

(2) SM (Scalar Multiply)・ルーチン $A_i \leftarrow S \times B_i$

a SM 1 A: 標準形式，B: コンパクト形式

b SM 2 A, B: 標準形式

c SM 3 A: コンパクト形式，B: 標準形式

(3) VIP (Vector Inner Product)・ルーチン $Z \leftarrow \Sigma A_i \times B_i$

a VIP A: 標準形式，B: コンパクト形式

b ASUM* A, B: 標準形式

(4) MOVE** $A_i \leftarrow B_i$

(5) IVALUE** $A_i \leftarrow S$

7.2 リスト処理ルーチン

(1) INSERT 構造列の非零要素で使用可能な行に存在する要素を記述する要素記述子を行リストに挿入して，行カウントを更新する。

(2) DELETE 行リストから，構造列の非零要素に対応する要素記述子を削除して，行カウントを更新する。

7.3 ソーティング・ルーチン

(1) SORTS キーが昇順にソートされたレコード（キーと項目）のリストに，straight insertion sort を用いて，レコードを挿入する。

(2) SORTB キーが昇順にソートされたレコードのリストに，binary insertion sort を用いて，レコードを挿入する。

---

* SSL II

** FORTRAN 基本サブルーチン

7.4 補助ルーチン

(1) PRINT　反復計算の中間結果の印刷を行なう。
(2) TYPE　変数・制約式の変域のタイプを示す。
(3) IERROR　最適解あるいは反復途中での解の誤差の度数分布表を作るために，誤差の分類を行なう。
(4) STATE　変数の状態を示す。
(5) BOUND　変数のタイプから，上・下限を示す。
(6) RANGE　制約式のタイプから，上・下限を示す。
(7) ISYMBL　係数行列の要素の度数分布表を作るために，要素の分類を行なう。
(8) SYMBL　係数行列の要素を，絶対値の大きさ・符号によって記号化する。

8. 標準的なジョブ構成

ALPS-1 のプロセジュアを用いて，標準 LP 問題を解く場合，次のように構成すればよい。

Step 1 (データの入力)

ALPS-1 は，係数行列等のデータ入力自体をサポートする標準的なプロセジュアを含んでいないので，ユーザの作成したプログラムで，ファイルから問題のデータを入力しなければならない。

Step 2 (スケーリング)

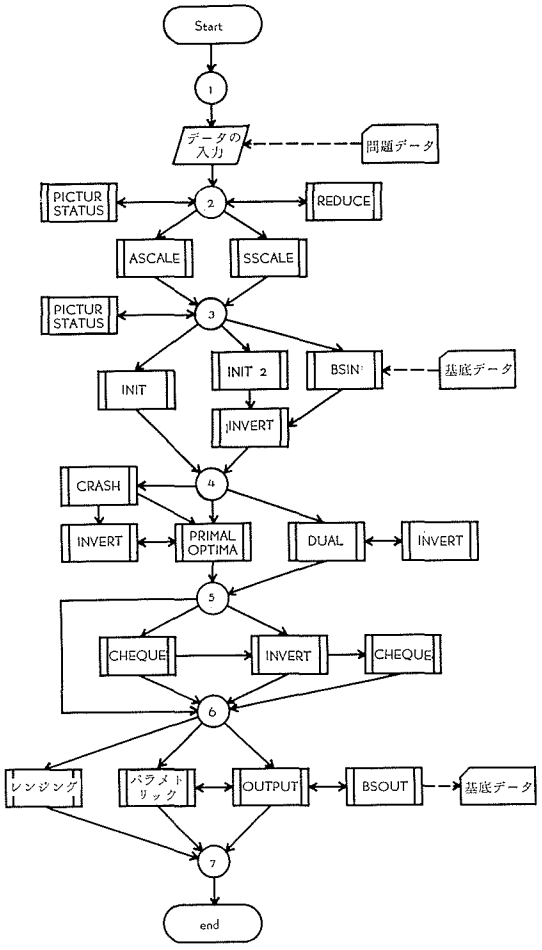
必要に応じて，スケーリングの前後に，係数行列の表示ルーチン PICTUR，STATUS を呼ぶことができる。問題の縮小法 REDUCE は解析のアルゴリズムの制限から，スケーリング前にのみ実行可能で，通常 REDUCE の実行によって，変数の変域は変更される。スケーリングには，自動スケーリングか標準スケーリングを使用する。

Step 3 (出発基底の入力)

通常は論理基底を与える INIT を用いて，出発基底の入力を行なう。また，ファイルから基底データを入力する BSIN，より有効な基底を与える INIT 2 を使用する場合は，対応する積型逆行列を計算するために再逆転を実行する必要がある。

Step 4 (最適化計算)

論理基底より有効な基底解をすばやく求める CRASH，あるいは再



図—3

逆転を行なう INVERT 等を利用しながら，複合主法 PRIMAL あるいは OPTIMA によって，最適解に到達する。出発基底として，双対実行可能解が与えられた場合，双対法 DUAL を利用できる。

Step 5 (解のチェック)

再逆転によって基底解の正確な値を再計算し，必要に応じて解のチェックを行なう。チェックの結果によっては，再逆転が不要になったり，変数の変域の与え方を調整して再計算を行なう必要が生じたりする。

Step 6 (結果の出力)

ユーザの作成した出力ルーチン，あるいは標準出力ルーチン OUTPUT を用いて，結果を出力する。最適化後の解析を行なう場合，レンジング・ルーチンあるいはパラメトリック・ルーチンを使用する。後の計算に現在の基底解が必要な場合，あるいは現在の基底を復元する必要がある場合には，BSOUT で基底データをファイルに出力する。

9. 応 用 例

9.1 ミニマックス型最適制御問題[4]

問題： 次の制約のもとで，最大加速度を最小化する状態・制御トラジェクトリーを求めよ。

状態方程式 $\dot{s}(t)=v(t)$; 速度

$\dot{v}(t)=a(t)$; 加速度

制御条件 $|a(t)|\leq z$

境界条件 $s(0)=v(0)=v(24)=0$, $s(24)=144$

$u(t)=a(t)+z$ とおいて，離散化すると次のような LP 問題に変形される。

$$
\begin{cases}
z \rightarrow \min & \\
\quad \text{subject to}: & s(k)+\Delta t\cdot v(k)-s(k+1)=0 \quad ; k=0\cdots N-1 \\
& v(k)+\Delta t\cdot u(k)-v(k+1)-\Delta t\cdot z=0 \quad ; k=0\cdots N-1 \\
& u(k)-2z\leq 0 \quad ; k=0\cdots N-1 \\
& s(k)\geq 0, \quad 0\leq v(k)\leq V_{\max}, \quad u(k)\geq 0 \quad ; k=0\cdots N-1 \\
& s(0)=v(0)=v(N)=0, \quad s(N)=144
\end{cases}
$$

ただし， $\Delta t=24/N$

$N=48$ の場合，145 行・147 列・係数行列の非零要素 433 個・密度 2.03% の LP 問題になり，計算結果は次頁のようになった。

9.2 線形分数計画法

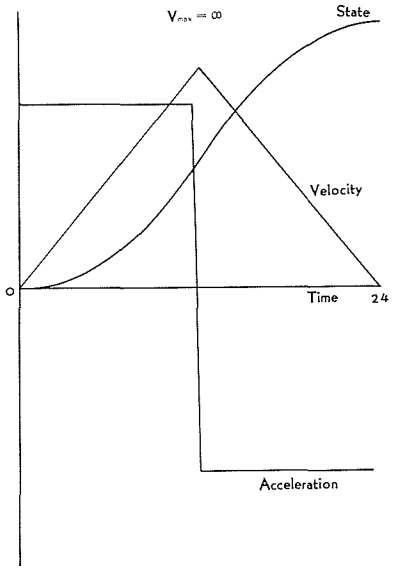
線形分数計画 (LFP) 問題は，次のように定式化される。

$$
\begin{cases}
z=\dfrac{c^TX+\alpha}{d^TX+\beta} \longrightarrow \max \\
\quad \text{subject to}: \quad AX\leq b, \quad X\geq 0
\end{cases}
$$

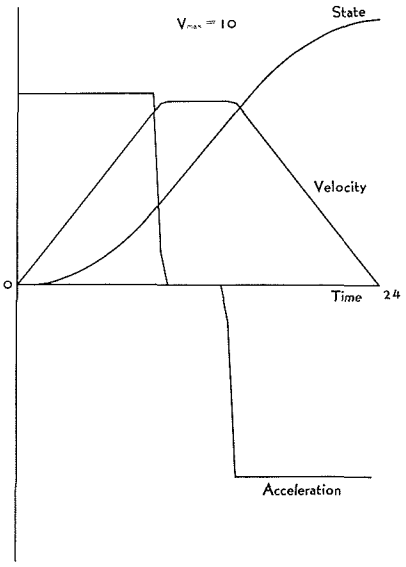
LFP に対する解法としては，次の 3 つのアルゴリズムが提案されている。

1) Charnes & Cooper のアルゴリズム[5] LP 問題への変換
2) Martos のアルゴリズム[6] パラメトリック LP
3) Bitran & Novaes のアルゴリズム[7] Martos のアルゴリズムの変形

Martos のアルゴリズムと Bitran & Novaes のアルゴリズムを，ALPS-1 でインプリメント



図—4

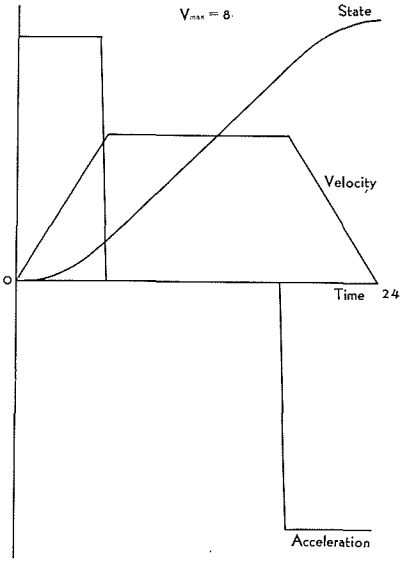


図—5

表—1

最大速度 (V_{max})	PRIMAL		DUAL	
	反復回数	計算時間*	反復回数	計算時間*
∞	119	4.1	166	4.2
11	124	4.0	154	3.5
10	128	4.4	153	3.5
9	135	4.9	143	3.1
8	142	5.6	143	2.9
7	155	7.0	151	3.1

* 単位 [sec]



図—6

表—2

問題のサイズ			Martos		Bitran & Novaes		
式	列	密　度*	反復回数	計算時間**	反復回数	計算時間**	LP の個数
400	800	1.25	604	47.6	907	98.7	7
200	400	1.25	126	6.6	793	52.5	6
200	400	4.0	132	4.9	320	16.2	4

* 単位 [%]　　** 単位 [sec]

して，乱数で係数行列を作成した問題で数値実験を行なった[8]。

10. あとがき

ALPS-1 の今後の拡張として，Glassey の多段決定形 LP，一般上限法，分解原理等のアルゴリズムを追加し，機能を充実させる予定である。

文　献

1) W. オーチャード・ヘイズ： コンピュータによる線形計画法 (昭 48), 培風館.
2) 岡本吉晴他： 数理計画法 (昭 52), 第 5 回日本 OR 学会シンポジウム・プログラム.
3) Brameller, A. 他： Sparsity (1976), Pitman Publishing.
4) Lack, G. N. T. 他： IEEE trans. on Automatic Control (1967), p. 749–p. 752.
5) Charnes, A. 他： Nav. Res. Log. Quart. Vol. 9 (1962), p. 181–p. 186.
6) Martos, B.： Nav. Res. Log. Quart. Vol. 11 (1964), p. 134–p. 155.
7) Bitran, G. R. 他： Oper. Res. Vol. 21 (1973), p. 22–p. 29.
8) 樫原敏康他： 電気学会情報処理研究会資料 IP 78-70 (昭 53).
9) 樫原敏康： 応用線形計画システム ALPS-1 の作成 (昭 54), (修士論文).