JAERI-M
7553

連立一次方程式を解くプログラムの開発と整備
(SSLの拡充とベンチマーク・テスト No.4)

1978年3月

藤村統一郎

日本原子力研究所
Japan Atomic Energy Research Institute

連立一次方程式を解くプログラムの開発と整備
（SSLの拡充とベンチマーク・テスト No.4）

日本原子力研究所東海研究所原子炉工学部
藤村 統一郎

（1978年1月30日受理）

科学計算用サブルーチン・ライブラリー(SSL)の拡充のため，連立一次方程式を解くプログラムの整備および開発を行った。

整備されたプログラムは，合同法，乗積型逆行列法，直交化法によるもののほか，疎な系を対象としたクラウト法によるものや反復法を加速するものが主である。

一方，開発されたプログラムは，エスカレーター法，直接的平行残差法，それに帯形の系を対象とした区画化三重対角法によるもので，それらの使用法や改良法が述べられる。更に，テストのための簡単な例題がついたプログラム・リストも与えられる。

Development and Adjustment of Programs for Solving Systems of Linear Equations

(Development and Benchmark Test of SSL, No.4)

Toichiro FUJIMURA

Division of Reactor Engineering,

Tokai Research Establishment, JAERI

(Received January 30, 1978)

Programs for solving the systems of linear equations have been adjusted and developed in expanding the scientific subroutine library SSL.

The principal programs adjusted are based on the congruent method, method of product form of the inverse, orthogonal method, Crout's method for sparse system, and acceleration of iterative methods.

The programs deveroped are based on the escalator method, direct parallel residue method and block tridiagonal method for band system. Described are usage of the programs developed and their future improvement. FORTRAN lists with simple examples in tests of the programs are also given.

Keywords : Computer Programs, Simultaneous Linear Equations, Scientific Subroutine Library, Congruent Method, Product Form, Orthogonalization, Crout's Method, Sparse System, Acceleration, Iterative Method, Escalator Method, Direct Parallel Residue Method, Block Tridiagonal Matrix, Band System

総　　論

ここ数年来，数値解析理論や新しいアルゴリズムばかりでなく，SSL（Scientific Subroutine Library，科学用サブルーチン・ライブラリー）や，特定の問題により深く関係した数値解析プログラム（例えば，Comp. Phys. Comm.）の発表件数は，膨大なものがある。従ってこれらのルーチンを限なくサーベイし，且つ整備しておくことは不可能であるのは勿論，効率的利用という立場から見ても，労多くして，功少なしの感がある。そこで実用上は，各ルーチンの特徴の分析やベンチマークテスト，"stiff"な問題への使用経験など集約された情報が要求され，これらが充分蓄積された時に，ルーチン相互の位置づけや体系化が可能となる。

我々の研究室では，数値解析の広い分野にわたる，これまでのアルゴリズム調査を土台としてベンチマークテストなどの情報集約と評価を行った。対象としたのは，主に最近発表されたプログラムとアルゴリズムで，先に行ったアルゴリズム調査の延長という形でまとめたので御利用いただきたい。

（西田雄彦，藤村統一郎）

目　　次

CONTENTS

1. 序　　言

連立一次方程式の数値解法は通常，直接法[1)2)]と反復法[1)~6)]に大別されている。ガウスの消去法[1)3)6)]や逐次過剰緩和法（SOR法）[1)4)6)7)]などはそれぞれこれらの代表的なものと言えるが，これらのみですべての問題を効率よく解けるとは限らない。一般に，科学計算用サブルーチン・ライブラリー（SSL）には種々な問題に対処するために多くのルーチンが備えられているが，どの問題にどのルーチンが適するかはベンチマーク・テストに待たなければならない。

従来，原研ではSSL・H[8)9)]，GSSL[10)]の2つのライブラリーが使われていたが，これらは所内の問題を解決するに充分とは言えなかった。そこで，まずその拡充のために，必要なルーチンの整備と開発をベンチマーク・テストに先立ち行った。本稿では，整備したルーチンを簡単に紹介するとともに，開発したルーチンの使用法や改良法を述べる。

2. 既存のルーチン

いま，与えられた連立一次方程式を行列表示に従い

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{1}$$

としよう。こゝに$\mathbf{A}=(a_{ij})$ $(i,j=1\sim n)$はn次の行列，$\mathbf{b}=(b_i)^t$ $(i=1\sim n)$と$\mathbf{x}=(x_i)^t$ $(i=1\sim n)$はn次の列ベクトルで，$\mathbf{x}$は未知とする。

これを解くルーチンは，およそ次の4つの項目により特徴づけられる。

① 対象とする系 —— 係数$\mathbf{A}$や定数$\mathbf{b}$の要素が属する数体および$\mathbf{A}$の性質

② 機　　能 —— 計算時の有効桁数，$\mathbf{A}^{-1}\mathbf{b}$ の計算以外の機能の有無

③ 数 値 解 法 —— 解法の種類，軸選択などの技法の有無

④ データの管理 —— $\mathbf{A}$や$\mathbf{b}$の入出力の方法や計算後の保存性，作業領域の必要性

これらは連立一次方程式の分野に限られたことではないが，特に係数が複素数の場合（項目①）と倍精度計算の場合（項目②）は，実数型の単精度計算から容易に変換しうるので，ルーチンの整備や開発も後者の場合に絞られてきた。従って，以後の議論でも単精度実数型のルーチンのみ扱う。

ここで，2つのライブラリーSSL・HとGSSLに属する連立一次方程式のルーチンを見てみよう。Table 1および2は，各ルーチンの対象とする系と解法の特徴を掲げたものであり，以下順を追ってその概要を説明する。

一般の対称でない密な系に対するガウスの消去法はTable 3で示されるが，④~⑨を前進消去，⑩~⑪を後退代入という。⑧の前に④－⑦×c_{12} などを行って一度に解を出すのが掃き出し法[2)5)8)]である。Table 3では①，⑤をもとに消去を行ったがa_{11}，$a_{22,1}$，$a_{33,1}$のことを軸といい，軸を残された小行列の第1列における絶対値最大の要素に選ぶことを部分軸選択，小行列全体におけ

1. 序　　言

連立一次方程式の数値解法は通常，直接法[1)2)]と反復法[1)~6)]に大別されている。ガウスの消去法[1)3)6)]や逐次過剰緩和法（SOR法[1)4)6)7)]）などはそれぞれこれらの代表的なものと言えるが，これらのみですべての問題を効率よく解けるとは限らない。一般に，科学計算用サブルーチン・ライブラリー（SSL）には種々な問題に対処するために多くのルーチンが備えられているが，どの問題にどのルーチンが適するかはベンチマーク・テストに待たなければならない。

従来，原研ではSSL・H[8)9)]，GSSL[10)]の2つのライブラリーが使われていたが，これらは所内の問題を解決するに充分とは言えなかった。そこで，まずその拡充のために，必要なルーチンの整備と開発をベンチマーク・テストに先立ち行った。本稿では，整備したルーチンを簡単に紹介するとともに，開発したルーチンの使用法や改良法を述べる。

2. 既存のルーチン

いま，与えられた連立一次方程式を行列表示に従い

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{1}$$

としよう。こゝに$\mathbf{A}=(a_{ij})$ $(i,j=1\sim n)$はn次の行列，$\mathbf{b}=(b_i)^t$ $(i=1\sim n)$と$\mathbf{x}=(x_i)^t$ $(i=1\sim n)$はn次の列ベクトルで，$\mathbf{x}$は未知とする。

これを解くルーチンは，およそ次の4つの項目により特徴づけられる。

① 対象とする系 —— 係数$\mathbf{A}$や定数$\mathbf{b}$の要素が属する数体および$\mathbf{A}$の性質

② 機　　能 —— 計算時の有効桁数，$\mathbf{A}^{-1}\mathbf{b}$ の計算以外の機能の有無

③ 数 値 解 法 —— 解法の種類，軸選択などの技法の有無

④ データの管理 —— $\mathbf{A}$や$\mathbf{b}$の入出力の方法や計算後の保存性，作業領域の必要性

これらは連立一次方程式の分野に限られたことではないが，特に係数が複素数の場合（項目①）と倍精度計算の場合（項目②）は，実数型の単精度計算から容易に変換しうるので，ルーチンの整備や開発も後者の場合に絞られてきた。従って，以後の議論でも単精度実数型のルーチンのみ扱う。

ここで，2つのライブラリーSSL・HとGSSLに属する連立一次方程式のルーチンを見てみよう。Table 1および2は，各ルーチンの対象とする系と解法の特徴を掲げたものであり，以下順を追ってその概要を説明する。

一般の対称でない密な系に対するガウスの消去法はTable 3で示されるが，④~⑨を前進消去，⑩~⑪を後退代入という。⑧の前に④－⑦×c_{12} などを行って一度に解を出すのが掃き出し法である[2)5)8)]。Table 3では①，⑤をもとに消去を行ったがa_{11}，$a_{22,1}$，$a_{33,1}$のことを軸といい，軸を残された小行列の第1列における絶対値最大の要素に選ぶことを部分軸選択，小行列全体におけ

Table 1 Subroutines for solving systems of linear equations in SSL·H[9]

Name	System	Method
GAUELS	General	Gaussian elimination
SWEEPS	General	Sweep out
SIMEQS	General	Sweep out, Scaling
TRIDGS	Tridiagonal	Gaussian elimination
CHOLES	Symmetric positive definite	Modified Cholesky
CHLSKS	Symmetric positive definite	Modified Cholesky, Revised inner product
BCHSKS	Symmetric positive definite, Sparse	Modified Cholesky, Bandwidth reduction
SCHSKS	Symmetric positive definite, Sparse	Modified Cholosky, Optimal ordering
BANDS	Symmetric positive definite, Band	Modified Cholesky
GAUSES	Diagonally dominant	Gauss-Seidel iteration

Table 2 Subroutines for solving systems of linear equations in GSSL[10]

Name	System	Method
SLERS	General	Sweep out
CROUT	General	Crout
TRIDIA	Tridiagonal	Gaussian elimination
PENTAD	Pentadiagonal	Gaussian elimination
DIVMTX	2-line	Block

る絶対値最大の要素に選ぶことを完全軸選択という。[1)6)] この技法により丸めの誤差が減らせるので，[11) 12)] 多くのルーチンに使われている。

掃き出し法のルーチンのうち，SWEEPS[9)] とSIMEQS[9)] は完全軸選択を行っており，特に後者は，予め係数の大きさを揃えるスケーリング[1)5)] を行い，数値的安定を図っている。一方，SLERS[10)] は逆行列も計算するという機能を持っている。

クラウト法の原理は，行列**A**が下三角行列**L**と上三角行列**U**により

$$\mathbf{A} = \mathbf{L}\,\mathbf{U}$$

と三角分解される性質に基づいている。[2)3)5)] Table 4の例では，$\mathbf{b} = (a_{14}, a_{24}, a_{34})^t$，

$\mathbf{L} = \begin{pmatrix} c_{11} & 0 & 0 \\ c_{21} & c_{22} & 0 \\ c_{31} & c_{32} & c_{33} \end{pmatrix}$，$\mathbf{U} = \begin{pmatrix} 1 & c_{12} & c_{13} \\ 0 & 1 & c_{23} \\ 0 & 0 & 1 \end{pmatrix}$ であり，$\mathbf{c} = (c_{14}, c_{24}, c_{34})^t$ とすると

$$\mathbf{U}\mathbf{x} = \mathbf{L}^{-1}\mathbf{b} = \mathbf{c} \tag{2}$$

から解**x**が簡単に求められる。ガウスの消去法では各要素を何度も書き改めるのに比べ，この方法では積和により一度の書き変えで済ませており，より計算機向きと言えよう。[6)]

三重対角の系（三項方程式）を解くルーチンのうち，TRIDIA[10)] は作業領域を別にとっているので元の係数が計算後も保存される。また，軸選択をしていないので，対角要素が非対角要素より大きい対角優勢[4)] の場合はかえって計算が速い。PENTAD[10)] は同様に五項方程式用であり，DIVM-TX[10)] は微分方程式を離散化したときに導出される方程式で，メッシュ点が2列になっている場合である。これは後で述べる区画化により三項方程式に還元される。

次に正定値対称な系の解法を考えてみよう。先に述べたクラウト法の原理を**A**が対称な場合に応用すると，**U**の要素は**L**の要素により，$c_{ij} = c_{ji}/c_{ii}$ と書ける。従って**L**の第j列を $\sqrt{c_{jj}}$ で割り，**U**の第i行に $\sqrt{c_{ii}}$ を掛けると$\mathbf{A} = \mathbf{L}\mathbf{L}^t$となり，記憶領域も半減できる。これがコレスキー法[2)6)] であるが，**L**および**c**の部分の計算を

$$c_{i1} = a_{i1} \qquad (i = 1 \sim n+1)$$

$$c_{ij} = a_{ij} - \sum_{k=1}^{j-1} c_{ik}\, c_{jk} / c_{kk} \qquad (j = 2 \sim n,\ i = j \sim n+1)$$

で行うと $\sqrt{\ }$ の計算が不要となる。これを変形コレスキー法[6)] と呼ぶが，このときは$\mathbf{b} = (a_{n+1,1}, a_{n+1,2}, \cdots, a_{n+1,n})^t$，$\mathbf{c} = (c_{n+1,1}, c_{n+1,2}, \cdots, c_{n+1,n})^t$で，$\mathbf{D} = \begin{pmatrix} c_{11} & & 0 \\ & c_{22} & \\ 0 & & \ddots \; c_{nn} \end{pmatrix}$

とすると，(1)式は$\mathbf{L}\mathbf{D}^{-1}\mathbf{L}^t\mathbf{x} = \mathbf{b}$となり，(2)式の代わりに

$$\mathbf{L}^t\mathbf{x} = \mathbf{D}\mathbf{L}^{-1}\mathbf{b} = \mathbf{c}$$

を解くことになる。

正定値対称な系を解くルーチンのうち，CHOLESとCHLSKS[9)] はデータ（係数a_{ij}や定数b_i）の与え方が異なる。また，前者は対称性のチェックを行っており，後者は積和を倍精度で計算し

Table 3 Algorithm of Gaussian elimination for n = 3

①	a_{11}	a_{12}	a_{13}	b_1	
②	a_{21}	a_{22}	a_{23}	b_2	original equations
③	a_{31}	a_{32}	a_{33}	b_3	
④	1	c_{12}	c_{13}	c_{14}	① ÷ a_{11}
⑤		$a_{22,1}$	$a_{23,1}$	$a_{24,1}$	② − ④ × a_{21}
⑥		$a_{32,1}$	$a_{33,1}$	$a_{34,1}$	③ − ④ × a_{31}
⑦		1	$c_{23,1}$	$c_{24,1}$	⑤ ÷ $a_{22,1}$
⑧			$a_{33,1}$	$a_{34,2}$	⑥ − ⑦ × $a_{32,1}$
⑨			1	x_3	⑧ ÷ $a_{33,2}$
⑩		1		x_2	⑦ − ⑨ × $c_{23,1}$
⑪	1			x_1	④ − ⑩ × c_{12} − ⑨ × c_{13}

Table 4 Algorithm of Crout's method for n = 3

a_{11}	a_{12}	a_{13}	a_{14}		① $c_{i1} = a_{i1}$
a_{21}	a_{22}	a_{23}	a_{24}		② $c_{1j} = a_{1j}/c_{11}$
a_{31}	a_{32}	a_{33}	a_{34}		3 $c_{i2} = a_{i2} - c_{i1}c_{12}$
① c_{11}	② c_{12}	c_{13}	c_{14}		④ $c_{2j} = (a_{2j} - c_{21}c_{1j})/c_{22}$
c_{21}	③ c_{22}	④ c_{23}	c_{24}		⑤ $c_{33} = a_{33} - c_{31}c_{13} - c_{32}c_{23}$
c_{31}	c_{32}	⑤ c_{33}	⑥ c_{34}		⑥ $c_{34} = (a_{34} - c_{31}c_{14} - c_{32}c_{24})/c_{33}$

ている。BCHSKSとSCHSKS[9]も正定値対称用であるが，前者は疎な系の行や列を自動的に入れ替えることにより帯構造に置き換え，その帯幅を減少させる[8]。後者も疎な系に適し，コレスキー分解の過程で発生する非零要素を最小限にとどめるよう最適な入れ替えを行う[8]。BANDS[9]は帯幅の決まった帯構造の系（多項方程式）を対象としている。

GAUSES[9]は唯一の反復法のルーチンであり，解法はガウス・ザイデル法[1)2)4)]である。従って対角優勢な系でないと，収束しなかったり，収束が遅かったりする。

以上概観したところによると，既存のルーチンは，密な系を解く基本的なルーチン，正定値対称な系をコレスキー法で解くルーチン，それに多項方程式を解くルーチンが主であると言える。従って，密な系を更に精度良く解くもの，また反復法ではガウス・ザイデル法以外の解法によるものや反復法を加速するルーチンなどが望まれよう。

3. 整備されたルーチン

SSLを拡充するにあたり，最も手っ取早い方法は必要なルーチンを捜し出すことである。幸い他の機関でプログラムを公開していたり，また論文にプログラムの付いているものも増える傾向にあるので，原研で入手できるものをさし当って整備した。しかし，計算機の機種や言語の違い，プログラムのミスが多々あり，そのまま使えることは少ない。また，アルゴリズムについても改良が望まれる場合があるので，それが簡単な場合はついでに修正を行った。

Table 5はこれらのルーチンについてまとめたものであるが，各ルーチンの詳細は関連する原論文や文献13)を参照するとして，この章では主な特徴と変換時の修正点につき，この表に沿って説明をしてゆく。

GELGはガウスの消去法のルーチンであり，係数と定数を別々の配列に格納しているという特徴がある。また，IBM社のルーチン[14]のため，IBM用のコードを変換する場合も便利である。GUEL1S[13]は行列の次数が変っても良いようにGELGが改良されている。

DECOMMはクラウト法の三角分解を行うルーチンであり，演算がデータの記憶領域への割り付けに沿うように考慮されている[15]。方程式(1)の解は組み合わせの（コンポーネント）ルーチンSOLVE[12)13)]を呼ぶことにより得られる。このルーチンを整備する際，非正則な系の計算を早く中断するよう直された。

単精度計算で得られる直接法の解はどうしても桁落ちが起きやすい。そこで，一度得られた解を$\mathbf{x}_0$とするとき，$\mathbf{r}_0 = \mathbf{A}\mathbf{x}_0 - \mathbf{b}$を倍精度で計算し，再び

$$\mathbf{A}\,\delta\mathbf{x} = \mathbf{r}_0$$

を解いて，$\mathbf{x} = \mathbf{x}_0 + \delta\mathbf{x}$とすれば解の精度は向上する。これをくり返し行うことを解の反復改良[11)12)16)]という。MA 21Aはこれを行うほか，データに誤差がある場合も考慮してある[16]。このルーチンに対しては，n＝1の場合のプログラム・ミスを修正したほか，行列式が大きくなった場合，オーバー・フローを防ぐルーチンOFLOWSおよび内積計算を行うルーチンMC03ASを開発した[13]。また，誤差評価に使う乱数ルーチンもRANDH[13)17)]に変更した。

ている。BCHSKSとSCHSKS[9]も正定値対称用であるが，前者は疎な系の行や列を自動的に入れ替えることにより帯構造に置き換え，その帯幅を減少させる[8]。後者も疎な系に適し，コレスキー分解の過程で発生する非零要素を最小限にとどめるよう最適な入れ替えを行う[8]。BANDS[9]は帯幅の決まった帯構造の系（多項方程式）を対象としている。

GAUSES[9]は唯一の反復法のルーチンであり，解法はガウス・ザイデル法[1)2)4)]である。従って対角優勢な系でないと，収束しなかったり，収束が遅かったりする。

以上概観したところによると，既存のルーチンは，密な系を解く基本的なルーチン，正定値対称な系をコレスキー法で解くルーチン，それに多項方程式を解くルーチンが主であると言える。従って，密な系を更に精度良く解くもの，また反復法ではガウス・ザイデル法以外の解法によるものや反復法を加速するルーチンなどが望まれよう。

3. 整備されたルーチン

SSLを拡充するにあたり，最も手っ取早い方法は必要なルーチンを捜し出すことである。幸い他の機関でプログラムを公開していたり，また論文にプログラムの付いているものも増える傾向にあるので，原研で入手できるものをさし当って整備した。しかし，計算機の機種や言語の違い，プログラムのミスが多々あり，そのまま使えることは少ない。また，アルゴリズムについても改良が望まれる場合があるので，それが簡単な場合はついでに修正を行った。

Table 5はこれらのルーチンについてまとめたものであるが，各ルーチンの詳細は関連する原論文や文献13)を参照するとして，この章では主な特徴と変換時の修正点につき，この表に沿って説明をしてゆく。

GELGはガウスの消去法のルーチンであり，係数と定数を別々の配列に格納しているという特徴がある。また，IBM社のルーチン[14]のため，IBM用のコードを変換する場合も便利である。GUEL1S[13]は行列の次数が変っても良いようにGELGが改良されている。

DECOMMはクラウト法の三角分解を行うルーチンであり，演算がデータの記憶領域への割り付けに沿うように考慮されている[15]。方程式(1)の解は組み合わせの（コンポーネント）ルーチンSOLVE[12)13)]を呼ぶことにより得られる。このルーチンを整備する際，非正則な系の計算を早く中断するよう直された。

単精度計算で得られる直接法の解はどうしても桁落ちが起きやすい。そこで，一度得られた解を$\mathbf{x}_0$とするとき，$\mathbf{r}_0 = \mathbf{A}\mathbf{x}_0 - \mathbf{b}$を倍精度で計算し，再び

$$\mathbf{A}\,\delta\mathbf{x} = \mathbf{r}_0$$

を解いて，$\mathbf{x} = \mathbf{x}_0 + \delta\mathbf{x}$とすれば解の精度は向上する。これをくり返し行うことを解の反復改良[11)12)16)]という。MA 21Aはこれを行うほか，データに誤差がある場合も考慮してある[16]。このルーチンに対しては，n＝1の場合のプログラム・ミスを修正したほか，行列式が大きくなった場合，オーバー・フローを防ぐルーチンOFLOWSおよび内積計算を行うルーチンMC03ASを開発した[13]。また，誤差評価に使う乱数ルーチンもRANDH[13)17)]に変更した。

Table 5 Adjusted subroutines for JSSL[13]

Name	System	Method
GELG	General	Gaussian elimination
GUEL1S	General	Gaussian elimination
DECOMM	General	Crout
MA21A	General	Crout, Scaling, Iterative refinement, Error estimation
PROD	General	Product form of the inverse
ODRPM	General	One-dimension reduction projection
EXACT	Integral	Congruence
LA05A	Sparse	Modified Crout
BAND	Band	Gaussian elimination
ORTHO	Symmetric	Orthogonalization
SYMMLQ	Symmetric	Lanczos
MA16A	Symmetric positive definite	Conjugate gradient
MA22A	Symmetric positive definite	Cholesky, Scaling, Iterative refinement, Error estimation
MA17A	Symmetric positive definite, Sparse	Modified Cholesky
MA15C	Symmetric positive definite, Variable band	Modified Cholesky
CHLSKB	Symmetric positive definite, Band	Cholesky
AAGLIP	General	Adaptive acceleration

次の3つは変った解法によるルーチンである。PRODは乗積型逆行列法[18]によるもので，次のアルゴリズムに従う。いま，行列$\mathbf{A}=(\mathbf{a}_1, \cdots, \mathbf{a}_n)$と単位行列$\mathbf{I}=(\mathbf{e}_1, \cdots, \mathbf{e}_n)$に対して列$\{\mathbf{A}_k\}$を$\mathbf{A}_k=(\mathbf{a}_1, \cdots, \mathbf{a}_k, \mathbf{e}_{k+1}, \cdots, \mathbf{e}_n)$，$\mathbf{A}_0=\mathbf{I}$，$\mathbf{A}_n=\mathbf{A}$で定義する。このとき，$\mathbf{A}_{k-1}\mathbf{y}=\mathbf{a}_k$の解を$\mathbf{y}_k=(y_1, \cdots, y_n)^t$とし，$\mathbf{I}_k=(\mathbf{e}_1, \cdots, \mathbf{e}_{k-1}, \mathbf{y}_k, \mathbf{e}_{k+1}, \cdots, \mathbf{e}_n)$とすれば$\mathbf{A}_k^{-1}=\mathbf{I}_k^{-1}\mathbf{A}_{k-1}^{-1}$が成り立つ。従って，$\mathbf{A}^{-1}=\mathbf{I}_n^{-1}\mathbf{I}_{n-1}^{-1}\cdots\mathbf{I}_1^{-1}$となり，解は$\mathbf{x}=\mathbf{I}_n^{-1}\mathbf{I}_{n-1}^{-1}\cdots\mathbf{I}_1^{-1}\mathbf{b}$と書ける。こゝに各因子は$\mathbf{I}_k^{-1}=(\mathbf{e}_1, \cdots, \mathbf{e}_{k-1}, \mathbf{z}_k, \mathbf{e}_{k+1}, \cdots, \mathbf{e}_n)$，$\mathbf{z}_k=(-y_1/y_k, \cdots, -y_{k-1}/y_k, 1/y_k, -y_{k+1}/y_k, \cdots, -y_n/y_k)^t$であるので計算は簡単である。

ODRPMは射影法により次数を1つ減らして解く。いま，ベクトル$\mathbf{r}$，$\mathbf{s}$が正則行列$\mathbf{A}$の初めの$(n-1)$個の列ベクトル$\mathbf{a}_1, \cdots, \mathbf{a}_{n-1}$に直交するとすれば，この2つは平行であるから$\mathbf{s}=t\mathbf{r}$と書ける。これらを残差とする異なったベクトル$\mathbf{y}$，$\mathbf{z}$が見つかり$\mathbf{r}=\mathbf{b}-\mathbf{A}\mathbf{y}$，$\mathbf{s}=\mathbf{b}-\mathbf{A}\mathbf{z}$，$\mathbf{s}=t\mathbf{r}$となれば

$$\mathbf{x}=\mathbf{y}-(\mathbf{y}-\mathbf{z})/(1-t)$$

は方程式(1)の解となる。実際には$\mathbf{y}=\mathbf{y}_0+\delta\mathbf{y}$，$\mathbf{y}_0=\mathbf{o}$，$\delta\mathbf{y}=(\delta\mathbf{y}_1, \cdots, \delta\mathbf{y}_{n-1}, 0)^t$と取り，$\mathbf{r}_0=\mathbf{b}-\mathbf{A}\mathbf{y}_0$，$\mathbf{r}=\mathbf{b}-\mathbf{A}\mathbf{y}$に対して

$$||\mathbf{r}_0||^2-||\mathbf{r}||^2$$

が最大になるように$\delta\mathbf{y}$を決める。この条件は丁度$\mathbf{a}_i^t\mathbf{r}=0$ $(i=1\sim n-1)$のとき満たされるので，これを変形して得られる次の方程式

$$\mathbf{B}\cdot\mathbf{u}=\mathbf{p}$$

を解けばよい。ここに，$\mathbf{B}$の要素は$b_{ij}=\mathbf{a}^t_i\mathbf{a}_j$なので対称正定値であり，ODRPMでは後で述べるORTHOで解かれる。また，$\mathbf{u}=(\delta y_1, \cdots, \delta y_{n-1})^t$は$\delta\mathbf{y}$が縮退したベルトル，$\mathbf{p}$の要素は$p_i=\mathbf{a}^t_i\mathbf{r}_0$である。$\mathbf{z}$についても$\mathbf{z}=\mathbf{z}_0+\delta\mathbf{z}$，$\mathbf{z}_0=\mathbf{e}_n$，$\delta\mathbf{z}=(\delta z_1, \cdots, \delta z_{n-1}, 0)^t$と置くことにより同様に解かれる。このルーチンは，文献19)に4次の場合のプログラムが載っていたのをn次に一般化したもので，余分な配列の削除，入出力の方法，$\mathbf{b}=\mathbf{o}$の場合の計算などについても改良を行った。

EXACT[20]は整数行列に関する合同式

$$\mathbf{A}\mathbf{X}\equiv\mathbf{B}\quad(\mathrm{mod}\ p_i)\qquad(i=1\sim p)$$

によって解かれる[5)21)]。従って，元のデータは有理数でなければならず，かつ通分したあとの整数の計算が素数の積$\prod_{i=1}^{p}p_i$より大きくなってはならないが，計算中の誤差は完全に除かれる。このルーチンでは素数の数を数えるときのミスを修正した。

非対称の疎な系に対してはLA05Aを整備した。これはクラウト法を変形したBartels-Golub法によるもので，行列の分解$\mathbf{A}=\mathbf{L}\mathbf{U}$において，特に$\mathbf{L}^{-1}=\mathbf{M}_r\mathbf{M}_{r-1}\cdots\mathbf{M}_1$，$\mathbf{U}=\mathbf{P}^{-1}\tilde{\mathbf{U}}\mathbf{Q}^{-1}$となっている。各$\mathbf{M}_k$は単位行列と1つだけ要素が異なる行列であり，$\mathbf{U}$は上三角行列$\tilde{\mathbf{U}}$が変換されたものである。これは線形計画法のために開発されたものであり，$\mathbf{A}$のある列$\mathbf{a}_m$を$\bar{\mathbf{a}}_m$に変えたときを$\bar{\mathbf{A}}$とすれば，$\bar{\mathbf{A}}=\mathbf{L}\bar{\mathbf{U}}$で，$\bar{\mathbf{U}}$は$\mathbf{U}$の第m列のみが$\mathbf{L}^{-1}\bar{\mathbf{a}}_m$となるという，PRODのとき

と同様な議論に基づいている。このとき，もし$\mathbf{P\bar{U}Q}$が上三角でなければ，基本操作の行列[2)3)]を付加すれば良い。このルーチンに関しては，疎行列の任意に与えられた非零要素を列の順に並べ変えるルーチンMC20Aを新しく作成した。

BANDは帯幅が余り広がらないよう部分軸選択を行っており，整備に際して100次までという制限が取り払われた。

これからは対称な系を解くルーチンを見て行こう。その殆んどが正定値を仮定しているが，次の2つのルーチンは正定値でなくてもよい。ORTHOはGram-Schnridtの直交化[6)]に似た方法により，

$$\mathbf{x}_1 = \mathbf{e}_1$$

$$\mathbf{x}_2 = \mathbf{e}_2 - \alpha_{12}\mathbf{x}_1$$

$$\cdots\cdots$$

$$\mathbf{x}_n = \mathbf{e}_n - \alpha_{1n}\mathbf{x}_1 - \alpha_{2n}\mathbf{x}_2 - \cdots\cdots - \alpha_{n-1,n}\mathbf{x}_{n-1}$$

なる一次独立で$\mathbf{A}$－直交性$\mathbf{x}_r{}^t\mathbf{A}\mathbf{x}_s = 0$（$r \neq s$）をもつn個のベクトルを作り，解をこれらの一次結合$\mathbf{x} = \sum_{r=1}^{n} c_r\mathbf{x}_r$で表わそうとするもので，

$$\alpha_{rs} = \mathbf{x}_r{}^t\mathbf{A}\mathbf{e}_s / \mathbf{x}_r{}^t\mathbf{A}\mathbf{x}_r \qquad (r < s)$$

$$c_r = \mathbf{x}_r{}^t\mathbf{b} / \mathbf{x}_r{}^t\mathbf{A}\mathbf{x}_r$$

とすればよい[19)]。このルーチンの整備にあたり，無駄な計算と記憶領域の節約のほか，分母の二次形式が小さくなった場合のチェックも加えられた。

一方のSYMMLQはLanczos法[6)23)]に基づいており，$\mathbf{A}$を三重対角行列に変換する方法である。具体的には

$$\mathbf{v}_0 = \mathbf{o}, \quad \beta_1 = ||\mathbf{b}||, \quad \mathbf{v}_1 = \mathbf{b}/\beta_1$$

$$\alpha_j = \mathbf{v}_j{}^t\mathbf{A}\mathbf{v}_j, \quad \beta_{j+1}\mathbf{v}_{j+1} = \mathbf{A}\mathbf{v}_j - \alpha_j\mathbf{v}_j - \beta_j\mathbf{v}_{j-1}$$

により求められる。各$\mathbf{v}_j$（$j=1, 2, \cdots$）は正規化されており，$\mathbf{v}_{k+1} = \mathbf{o}$となったとき止める。このとき，

$$\mathbf{V}_k = \begin{pmatrix} \mathbf{v}_1, \cdots, \mathbf{v}_k \end{pmatrix}, \quad \mathbf{T}_k = \begin{pmatrix} \alpha_1 & \beta_2 & & O \\ \beta_2 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_k \\ O & & \beta_k & \alpha_k \end{pmatrix}$$

であり，$\mathbf{T}_k = \mathbf{V}_k{}^t\mathbf{A}\mathbf{V}_k$となっている。また，$\mathbf{V}_k{}^t\mathbf{b} = \beta_1\mathbf{e}_1$であり，簡単なk次の対称な三項方程式$\mathbf{T}_k\mathbf{y}_k = \beta_1\mathbf{e}_1$を解くことにより$\mathbf{x}_k = \mathbf{V}_k\mathbf{y}_k$として求められる。これらのルーチンは一応正定値でないときにも使えることになっているが，二次形式が零に近くなると数値的不安定の原因ともなるので注意を要する。

対称正定数値である系を変形コレスキー法で解くルーチンは多いが，MA16Aは共役傾斜法[2)24)]で解く。この方法はLanczos法の一般化[23)]でもあり，コレスキー分解とも関係する。先と同様に具体的なアルゴリズムを記すと，$\mathbf{x}_0$を任意の初期ベクトルとして，

$$\mathbf{p}_0=\mathbf{r}_0=\mathbf{b}-\mathbf{A}\mathbf{x}_0,\quad \alpha_{k-1}=||\mathbf{r}_{k-1}||^2/\mathbf{p}_{k-1}^t\mathbf{A}\mathbf{p}_{k-1}$$

$$\mathbf{x}_k=\mathbf{x}_{k-1}+\alpha_{k-1}\mathbf{p}_{k-1},\quad \mathbf{r}_k=\mathbf{r}_{k-1}-\alpha_{k-1}\mathbf{A}\mathbf{p}_{k-1}$$

$$\beta_{k-1}=\|\mathbf{r}_k\|^2/\|\mathbf{r}_{k-1}\|^2,\quad \mathbf{p}_k=\mathbf{r}_k+\beta_{k-1}\mathbf{p}_{k-1}$$

の順で計算する。こゝに$\mathbf{r}$は残差ベクトル，$\mathbf{p}$は修正ベクトルであり，それぞれ$\mathbf{r}_i{}^t\mathbf{r}_j=0$，$\mathbf{p}_i{}^t\mathbf{A}\mathbf{p}_j=0\ (i\neq j)$が成り立つ。計算は$\mathbf{r}_{k-1}=\mathbf{o}$のとき止められ，$\mathbf{x}_{k-1}$が解となる。SYMMLQもそうであったが，このようなアルゴリズムではデータをすべて与える代わりに演算$\mathbf{Ap}$を実行するルーチンを付加して使うことが多く，従って疎行列向きと言えよう。

コレスキー法によるルーチンは4つ整備された。MA22A[16)]はMA21Aと類似ルーチンで解の反復改良を行っており，整備に際しMA21Aと同じ修正がなされた。MA17AとMA15Cも同時に開発されたルーチン[25)]で，ともに変形コレスキー法を用いている。前者は疎な系のために非零要素のみを与えるようになっており，同じ非零パターンの問題も効率よく解けるようになっているのが特色である。このルーチンには非零要素のインデックス計算を行うKB10AS[26)]が用意された。後者のMA15Cは帯幅が一定でない系を対象としており，記憶領域は帯幅分だけ必要であるが，計算はその中の非零要素のみについて行うようになっている。このルーチンは記憶領域を節約できるオプションとして直接アクセスのファイル[27)]を使用しているため，FORTRAN DとH用に別々に整備を行った。

CHLSKB[28)]はSSL・HのBANDSと同じく対称正定値でかつ帯状の系を対象としているが，解法はふつうのコレスキー法で，CHSLBD[13)]とコンポーネントになっている。CHSLBDはコレスキー分解の結果を使って解を出すルーチン[28)]を帯構造用に改めたもので，CHLSKBに課せられている200次までの制限を取り除く他，コレスキー法の特徴である，帯構造のコレスキー分解因子もやはり帯構造という性質が生かされている。

AAGLIP[29)]は反復法に関する唯一の整備されたルーチンで，ガウス・ザイデル法やSOR法など，主要な反復法の殆んどを加速できる。定常一階線型反復法[29)]は方程式(1)を変形して

$$\mathbf{x}_{k+1}=(\mathbf{I}-\mathbf{CA})\mathbf{x}_k+\mathbf{Cb}=\mu(\mathbf{x}_k,\mathbf{b})\qquad (k=0,1,\cdots)\tag{3}$$

で定義される。ここに$\mathbf{C}$は反復に関して不変な行列で，例えばガウス・ザイデル法のときは，$\mathbf{A}$を対角部分と左下部分と右上部分に分け$\mathbf{A}=\mathbf{D}+\mathbf{E}+\mathbf{F}$としたとき$\mathbf{C}=(\mathbf{D}+\mathbf{E})^{-1}$である。(3)式の加速は$\mu(\mathbf{x}_k,\mathbf{b})$を一旦$\mathbf{y}_k$とおき

$$\mathbf{p}_k=\mathbf{y}_k-\mathbf{x}_k,\quad \mathbf{q}_k=\mu(\mathbf{p}_k,\mathbf{o})$$

$$\alpha_k=\mathbf{p}_k{}^t(\mathbf{p}_k-\mathbf{q})/||\mathbf{p}_k-\mathbf{q}_k||^2,\quad \mathbf{x}_{k+1}=\mathbf{y}_k+\alpha_k\mathbf{p}_k$$

とする。これは，元の列$\mathbf{x}_k$の誤差$\mathbf{p}_k$よりも新しい誤差$\mathbf{q}_k$を小さくするように，加速係数αを反

復ごとに決めるところに特徴がある。このルーチンについては $\mathbf{p}_i$ の計算での桁落ちを防ぐため，反復改良[11]と同様な原点移動[6)29)]を用いるなどち密な計算になっていたのを，なるべく反復が進むように簡単化した。

4. 開発されたルーチン

連立一次方程式を解くアルゴリズムの中には，非常に有効なものでありながら，プログラム化がされていなかったり，あるいは我々が入手できなかったものもある。また，さし当っては実用的なものでなくても，研究的な見地から見て有望なものもある。この章ではこれらのアルゴリズムとそのプログラムについて，使用法も含めて幾分詳しく論ずるが，プログラムの細部については付録のFORTRANリスト，あるいは文献13)を参照されたい。

4.1 ESCARSおよびESCASS

数理物理学の分野では，ガレルキンやリッツの方法を用いるとき，(n−1)個の座標関数による系 $\mathbf{A}_{n-1}$ で解が不充分のとき，更にn番目の座標関数を付加した系 $\mathbf{A}_n$ による解が必要となることがある。[3)6)]

一般に，2つの任意の行列を

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \cdots & \mathbf{A}_{1q} \\ \vdots & & \vdots \\ \mathbf{A}_{p1} & \cdots & \mathbf{A}_{pq} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{B}_{11} & \cdots & \mathbf{B}_{1s} \\ \vdots & & \vdots \\ \mathbf{B}_{q1} & \cdots & \mathbf{B}_{qs} \end{pmatrix}$$

と分割(区画化)したとき，各分割された小行列の数と大きさが妥当であればあたかもふつうの要素のように扱うことにより積 $\mathbf{AB}$ が定義できて，その(i，j)小行列は $\sum_{k=1}^{q} \mathbf{A}_{ik}\mathbf{B}_{kj}$ となる。[3)4)]
この法則によれば，$\mathbf{A}_n$ を $\begin{pmatrix} \mathbf{A}_{n-1} & \mathbf{u}_{n-1} \\ \mathbf{v}_{n-1}^t & a_{nn} \end{pmatrix}$ と分割したとき，$\mathbf{A}_n^{-1}$ は $\alpha_n = a_{nn} - \mathbf{v}_{n-1}^t \mathbf{A}_{n-1}^{-1} \mathbf{u}_{n-1}$ を用いて

$$\mathbf{A}_n^{-1} = \begin{pmatrix} \mathbf{A}_{n-1}^{-1} + \mathbf{A}_{n-1}^{-1}\mathbf{u}_{n-1}\mathbf{v}_{n-1}^t\mathbf{A}_{n-1}^{-1}/\alpha_n, & -\mathbf{A}_{n-1}^{-1}\mathbf{u}_{n-1}/\alpha_n \\ -\mathbf{v}_{n-1}^t\mathbf{A}_{n-1}^{-1}/\alpha_n, & 1/\alpha_n \end{pmatrix}$$

と書ける[3)6)]ので，もし $\mathbf{A}_{n-1}^{-1}$ が既知でなければ $\mathbf{A}_{n-2}^{-1}$，…とさかのぼって $\mathbf{A}_n^{-1}$ を求めることになろう。この計算は実際には $-\mathbf{A}_{k-1}^{-1}\mathbf{u}_{k-1}$ が方程式

$$\mathbf{A}_{k-1}\mathbf{f}_{k-1} + \mathbf{u}_{k-1} = \mathbf{o}$$

の解であることを利用して求められる。[3)30)] これを要素を用いて書いたのが下の式である。

復ごとに決めるところに特徴がある。このルーチンについては $\mathbf{p}_i$ の計算での桁落ちを防ぐため，反復改良[11]と同様な原点移動[6)29)]を用いるなどち密な計算になっていたのを，なるべく反復が進むように簡単化した。

4. 開発されたルーチン

連立一次方程式を解くアルゴリズムの中には，非常に有効なものでありながら，プログラム化がされていなかったり，あるいは我々が入手できなかったものもある。また，さし当っては実用的なものでなくても，研究的な見地から見て有望なものもある。この章ではこれらのアルゴリズムとそのプログラムについて，使用法も含めて幾分詳しく論ずるが，プログラムの細部については付録のFORTRANリスト，あるいは文献13)を参照されたい。

4.1 ESCARSおよびESCASS

数理物理学の分野では，ガレルキンやリッツの方法を用いるとき，(n−1)個の座標関数による系 $\mathbf{A}_{n-1}$ で解が不充分のとき，更にn番目の座標関数を付加した系 $\mathbf{A}_n$ による解が必要となることがある。[3)6)]

一般に，2つの任意の行列を

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \cdots & \mathbf{A}_{1q} \\ \vdots & & \vdots \\ \mathbf{A}_{p1} & \cdots & \mathbf{A}_{pq} \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} \mathbf{B}_{11} & \cdots & \mathbf{B}_{1s} \\ \vdots & & \vdots \\ \mathbf{B}_{q1} & \cdots & \mathbf{B}_{qs} \end{pmatrix}$$

と分割(区画化)したとき，各分割された小行列の数と大きさが妥当であればあたかもふつうの要素のように扱うことにより積 $\mathbf{AB}$ が定義できて，その(i，j)小行列は $\sum_{k=1}^{q} \mathbf{A}_{ik}\mathbf{B}_{kj}$ となる。[3)4)]
この法則によれば，$\mathbf{A}_n$ を $\begin{pmatrix} \mathbf{A}_{n-1} & \mathbf{u}_{n-1} \\ \mathbf{v}_{n-1}^t & a_{nn} \end{pmatrix}$ と分割したとき，$\mathbf{A}_n^{-1}$ は $\alpha_n = a_{nn} - \mathbf{v}_{n-1}^t \mathbf{A}_{n-1}^{-1} \mathbf{u}_{n-1}$ を用いて

$$\mathbf{A}_n^{-1} = \begin{pmatrix} \mathbf{A}_{n-1}^{-1} + \mathbf{A}_{n-1}^{-1}\mathbf{u}_{n-1}\mathbf{v}_{n-1}^t\mathbf{A}_{n-1}^{-1}/\alpha_n, & -\mathbf{A}_{n-1}^{-1}\mathbf{u}_{n-1}/\alpha_n \\ -\mathbf{v}_{n-1}^t\mathbf{A}_{n-1}^{-1}/\alpha_n, & 1/\alpha_n \end{pmatrix}$$

と書ける[3)6)]ので，もし $\mathbf{A}_{n-1}^{-1}$ が既知でなければ $\mathbf{A}_{n-2}^{-1}$，…とさかのぼって $\mathbf{A}_n^{-1}$ を求めることになろう。この計算は実際には $-\mathbf{A}_{k-1}^{-1}\mathbf{u}_{k-1}$ が方程式

$$\mathbf{A}_{k-1}\mathbf{f}_{k-1} + \mathbf{u}_{k-1} = \mathbf{o}$$

の解であることを利用して求められる。[3)30)] これを要素を用いて書いたのが下の式である。

$$
\begin{aligned}
&a_{11} f_{1k} + \cdots\cdots \quad + a_{1,k-1} f_{k-1,k} + a_{1k} = 0 \\
&\text{-------} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (k=2\sim n) \\
&a_{k-1,1} f_{1k} + \cdots \quad + a_{k-1,k-1} f_{k-1,k} + a_{k-1,k} = 0
\end{aligned}
\tag{4}
$$

このとき，n次の単位上三角行列 $\mathbf{F} = \begin{pmatrix} 1 & f_{12} & \cdots & f_{1n} \\ & 1 & \ddots & \vdots \\ & & \ddots & f_{n-1,n} \\ O & & & 1 \end{pmatrix}$ を考えると，$\mathbf{A}$との積

$\mathbf{C} = \mathbf{AF} = (c_{ij})$は(4)式から下三角行列となり，

$$
\begin{aligned}
&c_{i1} = a_{i1} \qquad (i=1\sim n) \\
&c_{ij} = a_{i1} f_{1j} + \cdots \quad + a_{i,j-1} f_{j-1,j} + a_{ij} \qquad (j \geqq 2,\ i=j\sim n)
\end{aligned}
\tag{5}
$$

である。更に$\mathbf{C}$は

$$
e_{ij} = c_{ij} / c_{jj} \qquad (j=1\sim n,\quad i=j+1\sim n) \tag{6}
$$

により，対角行列$\mathbf{D} = (c_{jj})$と単位下三角行列$\mathbf{E} = (e_{ij})$により

$$
\mathbf{C} = \mathbf{E}\,\mathbf{D}
$$

と分解される。こゝに$\mathbf{F}^{-1}$はやはり単位上三角行列であって，クラウト法のときと同様な分解$\mathbf{A} = \mathbf{EDF}^{-1}$が得られたことになる。[3)]

次に$\mathbf{A}^t$について同様の操作を行うとしよう。上記のような分解は一意的に定まるので，

$$
\mathbf{A}^t = \mathbf{GDH}^{-1} = (\mathbf{F}^{-1})^t\,\mathbf{DF}^t
$$

から$\mathbf{G} = (g_{ij}) = (\mathbf{F}^{-1})^t$，即ち$\mathbf{GF}^t = \mathbf{I}$が得られる。これを要素を用いて書いたのが次式である。

$$
\begin{aligned}
&g_{k+1,1} + g_{k+1,2} f_{12} + \cdots\cdots \quad + g_{k+1,k} f_{1k} + f_{1,k+1} = 0 \\
&\qquad g_{k+1,2} \quad + \cdots\cdots \quad + g_{k+1,k} f_{2k} + f_{2,k+1} = 0 \qquad (k=1\sim n-1) \\
&\qquad\qquad\qquad\qquad g_{k+1,k} \quad + f_{k,k+1} = 0
\end{aligned}
\tag{7}
$$

転置行列の場合も$\mathbf{GD} = \mathbf{A}^t\mathbf{H}$，$\mathbf{EH}^t = \mathbf{I}$により，それぞれ(5)～(7)式と類似の式が得られる。[3)] このとき(5)および(6)式は$\mathbf{F}$の第j列から$\mathbf{E}$の第j列を，(7)式は$\mathbf{G}$の(k+1)行から$\mathbf{F}$の第(k+1)列を求めるという意味を持っている。

ここで計算の順序について考えてみよう。$\mathbf{E}$の第1列が(5)および(6)式により$\mathbf{A}$の第1列から容易に求められるように，転置行列の場合を考えると，$\mathbf{G}$の第1列は$\mathbf{A}$の第1行から容易に求められる。次に，$\mathbf{E}$および$\mathbf{G}$の第k列までが分ったと仮定すると，これらが単位下三角行列であるこ

とから第(k+1)行まで分ったことになる。すると(7)式により，それぞれ第(k+1)列まで求めることができるので，数学的帰納法によりすべての要素が求められることになる。

Table 6は$\mathbf{A}^{-1}\mathbf{b}$と$(\mathbf{A}^t)^{-1}\mathbf{b}$を計算する順序を示したものであり，$\mathbf{D}$の要素は理解を容易にするために重複して計算されている。(4)～(7)式を拡張し$a_{i,n+1}=a_{n+1,i}=-b_i$とすれば，解はそれぞれ$\mathbf{A}^{-1}\mathbf{b}=(f_{15},\cdots,f_{45})^t$，$(\mathbf{A}^t)^{-1}\mathbf{b}=(h_{15},\cdots,h_{45})^t$として得られる。この解法はエスカレーター法[3)6)30)]と呼ばれ，特に(4)式は検算に応用できるので，たちの悪い系でもかなり精度の良い解が得られると言われている[3)30)]。

文献30)には非対称の4次の場合の計算式が，文献3)には対称の4次の場合の計算図が掲げられているが，ESCARSは前者を一般化し，プログラム化したもので，

CALL ESCARS (A, W, N1, N, IOP, IERR)

で呼び出される。このルーチンは検算はやっていないが，$\mathbf{A}_{n-1}^{-1}$などが分らなくても$\mathbf{A}^{-1}\mathbf{b}$や$(\mathbf{A}^t)^{-1}\mathbf{b}$が解けるようになっている。$(\mathbf{A}^t)^{-1}\mathbf{b}$の計算はオプションで，IOP＝1のとき$\mathbf{A}^{-1}\mathbf{b}$のみを，2のとき双方を計算する。IOP＝2のとき，AとWは大きさ$(n_1\times n_2)$の配列で，n_1, $n_2\geqq n+1$にとられる。データは$A(i,j)=a_{ij}$, $A(i,n+1)=A(n+1,i)=-b_i$と入れられる。Wは作業領域で，E，F，G，Hを格納し，解$\mathbf{A}^{-1}\mathbf{b}$がW(i,n+1)に，$(\mathbf{A}^t)^{-1}$がW(n+1, i)に出力される。N1, Nには上のn_1, nを入れる。正常に計算されたときはIERR＝0であるが，第j番目の軸が0になったとき

PIVOT＝0.　IN ESCARS AT j－TH STAGE

のメッセージが出てRETURNする。IOP＝1のときは$n_1\geqq n$にとれば良く，Aの第(n+1)行に$-\mathbf{b}$を入れる必要はない。

例題としては$\mathbf{A}=\begin{pmatrix}1&2&3\\4&5&6\\7&8&0\end{pmatrix}$，$\mathbf{b}=\begin{pmatrix}14\\32\\23\end{pmatrix}$が取られ，IOP＝2で計算されている。正しい解は$\mathbf{x}=(1,2,3)^t$で，この例では0.093秒以内で8桁の精度が得られている。このルーチンの大きさは578語であり，付属するルーチンは何も無い。

ESCASSは対称な系の場合である。$\mathbf{A}^t=\mathbf{A}$により上のアルゴリズムは$\mathbf{G}=\mathbf{E}$，$\mathbf{H}=\mathbf{F}=(\mathbf{E}^t)^{-1}$となって一層簡単になる[3)]。このルーチンは

CALL ESCASS (A, N1, N, IERR)

で呼び出されるが，Aを大きさ(n+1, n)にとるだけでWやIOPは不要である。データはA(i, j)＝a_{ij}となるよう対角および左下部分のみを入れ，$-\mathbf{b}$はAの第(n+1)行に入れる。このとき，空いた右上部分が作業領域となる。

このルーチンの大きさは354語で，$\mathbf{A}=\begin{pmatrix}1&2&4\\2&3&5\\4&5&6\end{pmatrix}$，$\mathbf{x}=\begin{pmatrix}1\\2\\3\end{pmatrix}$，$\mathbf{b}=\begin{pmatrix}17\\23\\32\end{pmatrix}$の例でESCARSと同様の結果が得られた[13)]。

4.2 DPRM

第3章のODRPMの項で，射影法により次数を1つ減らして解く方法について述べた。それは元の方程式(1)に対して，2つのベクトル$\mathbf{y}=\mathbf{o}+\delta\mathbf{y}$，$\mathbf{z}=\mathbf{e}_n+\delta\mathbf{z}$，$\delta\mathbf{y}=(\delta y_1,\cdots,\delta y_{n-1},0)^t$

Table 6 Algorithms of calculating $A^{-1}b$ and $(A^t)^{-1}b$ in the escalator method

①	②'	④	⑥'	⑧
c_{11}	f_{12}	f_{13}	f_{14}	f_{15}
	③'			
e_{21}	c_{22}	f_{23}	f_{24}	f_{25}
		⑤		
e_{31}	e_{32}	c_{33}	f_{34}	f_{35}
			⑦'	
e_{41}	e_{42}	e_{43}	c_{44}	f_{45}
e_{51}	e_{52}	e_{53}	e_{54}	

①'	②	④'	⑥	⑧'
c_{11}	h_{12}	h_{13}	h_{14}	h_{15}
	③			
g_{21}	c_{22}	h_{23}	h_{24}	h_{25}
		⑤'		
g_{31}	g_{32}	c_{33}	h_{34}	h_{35}
			⑦	
g_{41}	g_{42}	g_{43}	c_{44}	h_{45}
g_{51}	g_{52}	g_{53}	g_{54}	

$\delta \mathbf{z} = (\delta \mathbf{z}_1, \cdots, \delta \mathbf{z}_{n-1}, 0)^t$ を取り，それらの残差ベクトルが射影法の条件をみたすように $\delta \mathbf{y}$，$\delta \mathbf{z}$ を決めるもので，その式は2つの（n－1）次方程式

$$\mathbf{B}\mathbf{u} = \mathbf{p}, \quad \mathbf{B}\mathbf{v} = \mathbf{q}$$

であった[19]。即ち，1つのn次の一般的な方程式が，2つの同一係数の対称正定値な系におきかえられる訳で，以下順次次数を下げて1次の方程式に帰着して解くとき，これを直接的平行残差法という[19]。

この方法は大きな記憶領域を必要とするが，高精度の点で評価されており[19]，DPRMの開発に当り，特に記憶領域を少なくするよう考慮がなされた。その呼出し方は

CALL DPRM（N，A，B，EPS，IER）

である。Nは次数のnであり，IERは計算が正常に行われたとき0，k次の解が生成できなかったときkとなる。Aは1次元の配列で，大きさは $f(n) = \max_{0 \leq i \leq n-1} \{(n-i)^2 + (n-i) \times 2^i\}$ 以上とする。Aはデータを $a_{11}, a_{21}, \cdots, a_{n1}, a_{12}, \cdots, a_{n2}, \cdots, a_{nn}, b_1, \cdots, b_n$ と列に沿って入れるほか，作業領域にも使われる。解 $\mathbf{x}$ は $\mathbf{b}$ の所に入る。Bはやはり大きさf(n)以上の1次元配列で，作業領域である。EPSは零判定値であり，ふつう 10^{-6} くらいにとられる。

このルーチンでは記憶領域を節約するために，作業領域A，Bを交互に使うための付属ルーチンROA，SOLA，SRSを同時に開発したが，作業用ファイル ¥DISK F01 も併わせて使われている[13]。IER＞0 で終ったとき，$\mathbf{A}$ のスケーリングが十分かどうかを再検討してみる必要があろう。また，IER＝0 でも解の精度が良くないときは，EPSを小さくとり直すのも一法であろう。例題としては $\mathbf{A} = \begin{pmatrix} 5 & 1 & 2 & 2 \\ 3 & 5 & -1 & 3 \\ 1 & -3 & 5 & 7 \\ 4 & 1 & 0 & 5 \end{pmatrix}$，$\mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix}$，$\mathbf{b} = \begin{pmatrix} 2 \\ 6 \\ -14 \\ 0 \end{pmatrix}$ が解かれているが，精度は6桁であった。

4.3 MTES

既に多項方程式を対象としたルーチンは多く紹介されたが，その殆んどはガウスの消去法によるものであった。ここでは少し観点を変え，区画化した後三項方程式として解く方法について述べる。

いま帯幅（2m－1）のn次の系があり，n／m＝l でm行，m列ずつに分割できたとしよう。このとき，元の系は区画化三項方程式になるので，前進消去，後退代入の公式

$$\mathbf{F}_1 = \mathbf{A}_{11}$$

$$\mathbf{F}_k = \mathbf{A}_{kk} - \mathbf{A}_{k,k-1} \mathbf{F}_{k-1}^{-1} \mathbf{A}_{k-1,k} \qquad (k = 2 \sim \ell)$$

$$\mathbf{g}_1 = \mathbf{b}_1$$

$$\mathbf{g}_k = \mathbf{b}_k - \mathbf{A}_{k,k-1} \mathbf{F}_{k-1}^{-1} \mathbf{g}_{k-1} \qquad (k = 2 \sim \ell)$$

$$\mathbf{x}_n = \mathbf{F}_\ell^{-1}\mathbf{g}_\ell$$

$$\mathbf{x}_k = \mathbf{F}_k^{-1}(\mathbf{g}_k - \mathbf{A}_{k,k+1}\mathbf{x}_{k+1}) \qquad (k=\ell-1\sim 1)$$

により解くことができる[4]。こゝで注意を要するのは，ふつうの三項方程式の計算と違い，$\mathbf{A}_{k,k-1}\mathbf{F}_{k-1}^{-1}\mathbf{A}_{k-1,k}$ の計算が可換でないことであり，これをルーチン化するときは大きな記憶領域を必要とすることである。このアルゴリズムはそのまゝ用いると非能率であるが，反復法に使えば例えば $\mathbf{F}_k$ の逆転を一度だけで済ますなどの別の効果が期待できよう。また，mがnを整除しないときには $x_{n+1}=0$ などの自明な式を付加すれば良い。

MTESは，記憶領域の節約には工夫がなされているが，アルゴリズムは上記のものを踏襲しており，$\mathbf{F}_k^{-1}$の計算は部分軸選択による掃出し法によっている。その呼び出し方は

CALL MTES (A, W, N, M, NFORM, B, EPS, ILL, X)

である。ここにN，Mは上記のn，mであり，nはmの倍数とする。AとWは2次元の配列で，**A**のAへの入れ方はFig. 1 のように2通りの方法が選択できる。Aの大きさ(n，n_1)はNFORM＝1のとき$n_1 \geqq n$，NFORM＝2のとき$n_1 \geqq 2m-1$とする。Wは作業領域でその大きさは(n，n_2)，$n_2 \geqq 2n+3$である。BとXは大きさn以上の1次元配列であり，Bに**b**を入れるとXに解**x**が得られる。EPSは行列式の零判定値で各$\mathbf{F}_{kk}$や**A**の行列式より少し大きい数を目安として入れる。ILLは計算の終了状態を示す数で，0のとき正常，11～13のときnやmに関するエラー，20のとき行列式がEPSより小さくなったことを示す。最後の場合にはEPSを大きくとりすぎてないかどうか検討する必要があろう。

例題では

$$\mathbf{A} = \begin{pmatrix} 51 & 21 & & & & O \\ 11 & 52 & 22 & & & \\ & 12 & 53 & 23 & & \\ & & 13 & 54 & 24 & \\ & & & 14 & 55 & 25 \\ & O & & & 15 & 56 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 93 \\ 181 \\ 275 \\ 375 \\ 481 \\ 411 \end{pmatrix}$$

を解いており，7桁の精度が得られている。

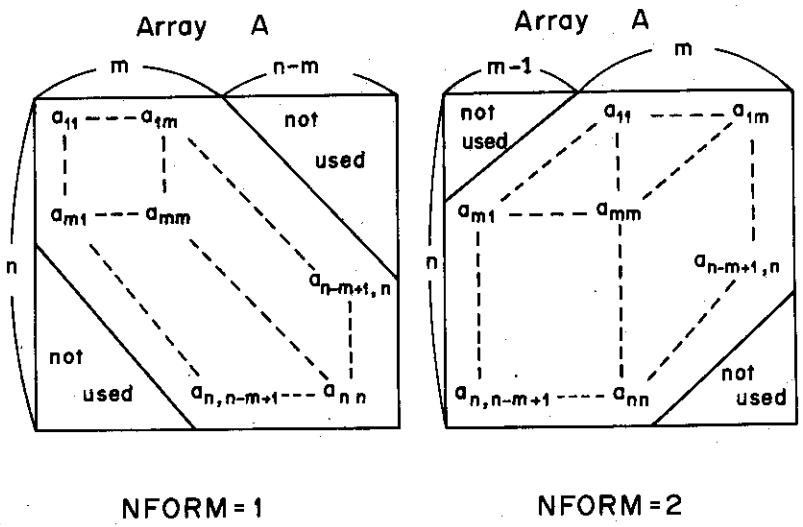

Fig. 1 Data arrangements in array A

4.4 これらの方法の改良について

エスカレーター法のルーチンの改良は既に述べたように検算の項を付け加えれば精度は向上できるであろうが，計算時間が余り長くならないよう考慮すべきだろう。それよりもガウスの消去法などで良く用いられる軸選択の方がより有効と言える[30]。また，具体的に小さい系の結果を知って大きい系を解く必要のある場合はその次数から計算できるよう修正すればよい。そのときは(4)式やTable 6が参考になろう。

DPRMも主に精度に重点を置いたルーチンである。平行比を表わすtは第1要素の比から決めているので，これを例えば最小自乗法[6)31)]のように平均化して決めれば，計算時間は多少かかっても解の各要素間のむらがもっと減らせるだろう。

帯状の系を解くMTESはこのままではガウスの消去法など，他の解法のルーチンに比べうるとは思えない。通常見受けられる反復法の中には，元の行列を区画化することにより反復行列のスペクトル半径を小さくできて，収束が速められる場合がある[4]。従って，大次元の多項方程式を反復法で解く場合，ヤコビ法[2)4)]などは区画化した後の反復行列においても帯幅が不変なので，もし区画化によりスペクトル半径が減少できるとすればMTESの用途も一層開けるだろう。

5. 結　　言

SSLの拡充にあたり，連立一次方程式の分野について既存のルーチンを概観し，不充分な部分の整備や開発についても論じた。既存のライブラリーとしては，富士通で開発されたSSL－Hと原研で開発されたGSSLを基としたが，これらはそれぞれSSL II[32]，JSSL[13]としてより充実したものに改版されつつある。従って，本稿でとり上げられなかったルーチンも幾らかあるが，反復法のルーチンは他のルーチンと同様，汎用性をもった形でもう少し用意されるべきであろう。また，同種の系を対象とした異なったルーチンの比較はベンチマーク・テストに待たなければならないが，そのことも考慮して議論は単精度計算のルーチンに終始した。しかし原研での使用実績をみると，倍精度のルーチンも決して少なくないので，将来は倍精度計算あるいは複素数の系を解くルーチンの充実も不可欠となろう。

謝　　辞

他機関で開発されたルーチンの整備に際し，異種言語などについて核設計研究室，筒井恒夫氏に相談することが多かった。こゝに謝意を表します。

4.4 これらの方法の改良について

エスカレーター法のルーチンの改良は既に述べたように検算の項を付け加えれば精度は向上できるであろうが，計算時間が余り長くならないよう考慮すべきだろう。それよりもガウスの消去法などで良く用いられる軸選択の方がより有効と言える[30]。また，具体的に小さい系の結果を知って大きい系を解く必要のある場合はその次数から計算できるよう修正すればよい。そのときは(4)式やTable 6が参考になろう。

DPRMも主に精度に重点を置いたルーチンである。平行比を表わすtは第1要素の比から決めているので，これを例えば最小自乗法[6)31)]のように平均化して決めれば，計算時間は多少かかっても解の各要素間のむらがもっと減らせるだろう。

帯状の系を解くMTESはこのままではガウスの消去法など，他の解法のルーチンに比べうるとは思えない。通常見受けられる反復法の中には，元の行列を区画化することにより反復行列のスペクトル半径を小さくできて，収束が速められる場合がある[4]。従って，大次元の多項方程式を反復法で解く場合，ヤコビ法[2)4)]などは区画化した後の反復行列においても帯幅が不変なので，もし区画化によりスペクトル半径が減少できるとすればMTESの用途も一層開けるだろう。

5. 結　　言

SSLの拡充にあたり，連立一次方程式の分野について既存のルーチンを概観し，不充分な部分の整備や開発についても論じた。既存のライブラリーとしては，富士通で開発されたSSL－Hと原研で開発されたGSSLを基としたが，これらはそれぞれSSL II[32]，JSSL[13]としてより充実したものに改版されつつある。従って，本稿でとり上げられなかったルーチンも幾らかあるが，反復法のルーチンは他のルーチンと同様，汎用性をもった形でもう少し用意されるべきであろう。また，同種の系を対象とした異なったルーチンの比較はベンチマーク・テストに待たなければならないが，そのことも考慮して議論は単精度計算のルーチンに終始した。しかし原研での使用実績をみると，倍精度のルーチンも決して少なくないので，将来は倍精度計算あるいは複素数の系を解くルーチンの充実も不可欠となろう。

謝　　辞

他機関で開発されたルーチンの整備に際し，異種言語などについて核設計研究室，筒井恒夫氏に相談することが多かった。こゝに謝意を表します。

4.4 これらの方法の改良について

エスカレーター法のルーチンの改良は既に述べたように検算の項を付け加えれば精度は向上できるであろうが，計算時間が余り長くならないよう考慮すべきだろう。それよりもガウスの消去法などで良く用いられる軸選択の方がより有効と言える[30]。また，具体的に小さい系の結果を知って大きい系を解く必要のある場合はその次数から計算できるよう修正すればよい。そのときは(4)式やTable 6が参考になろう。

DPRMも主に精度に重点を置いたルーチンである。平行比を表わすtは第1要素の比から決めているので，これを例えば最小自乗法[6)31)]のように平均化して決めれば，計算時間は多少かかっても解の各要素間のむらがもっと減らせるだろう。

帯状の系を解くMTESはこのままではガウスの消去法など，他の解法のルーチンに比べうるとは思えない。通常見受けられる反復法の中には，元の行列を区画化することにより反復行列のスペクトル半径を小さくできて，収束が速められる場合がある[4]。従って，大次元の多項方程式を反復法で解く場合，ヤコビ法[2)4)]などは区画化した後の反復行列においても帯幅が不変なので，もし区画化によりスペクトル半径が減少できるとすればMTESの用途も一層開けるだろう。

5. 結　　言

SSLの拡充にあたり，連立一次方程式の分野について既存のルーチンを概観し，不充分な部分の整備や開発についても論じた。既存のライブラリーとしては，富士通で開発されたSSL－Hと原研で開発されたGSSLを基としたが，これらはそれぞれSSL II[32]，JSSL[13]としてより充実したものに改版されつつある。従って，本稿でとり上げられなかったルーチンも幾らかあるが，反復法のルーチンは他のルーチンと同様，汎用性をもった形でもう少し用意されるべきであろう。また，同種の系を対象とした異なったルーチンの比較はベンチマーク・テストに待たなければならないが，そのことも考慮して議論は単精度計算のルーチンに終始した。しかし原研での使用実績をみると，倍精度のルーチンも決して少なくないので，将来は倍精度計算あるいは複素数の系を解くルーチンの充実も不可欠となろう。

謝　　辞

他機関で開発されたルーチンの整備に際し，異種言語などについて核設計研究室，筒井恒夫氏に相談することが多かった。こゝに謝意を表します。

参 考 文 献

1) Westlake J.R.: "A Handbook of Numerical Matrix Inversion and Solution of Linear Equations", John Wiley & Sons, Inc., New York (1968)

2) 新谷尚義："数値計算I"，朝倉書店（1972）

3) Faddeeva V.N.: "Computational Methods of Linear Algebra" (Benster C.D. tr.) Dover, New York (1959)

4) Varga R.S.: "Matrix Iterative Analysis", Printice-Hall (1962)またはR.S.バーガ："大型行列の反復解法"（渋谷政昭他訳），サイエンス社（1974）

5) 一松信："数値計算"，至文堂（1966）

6) 戸川隼人："マトリックスの数値計算"，オーム社（1971）

7) フォーサイス，ワソー："偏微分方程式の差分法による近似解法　上，下"（藤野精一訳），吉岡書店（1967）

8) 富士通（株）："FACOM SSL（科学用サブルーチン・ラリブラリ）解法解説書（000－301～309－003－5）"（1972）

9) 富士通（株）："FACOM FORTRAN SSL使用手引書（99SP－0040－1）"（1976）

10) 計算センター："GSSL（原研版サブルーチン・ライブラリ）マニュアル"（所内資料）（1972）

11) Forsythe G.E., Moler C.B.: "Computer Solution of Linear Algebraic Systems"（渋谷政昭，田辺国士訳），培風館（1970）

12) 堀上邦彦："連立一次代数方程式の精密計算プログラム（SLINER，DLINER）"(所内資料)（1973）

13) 藤村統一郎，西田雄彦，浅井清（編）："JSSL（原研版・科学用サブルー・ライブラリー）マニュアル"，JAERI－M 7102（1977）

14) IBM（株）："SL－MATH（科学計算ライブラリー）（1974）

15) Moler C.B.: "Algorithm 423, Linear Equation Solver [F4]", Comm. ACM, 15, 274 (1972)

16) Malow S., Reid J.K.: "Fortran Subroutines for the Solution of Linear Equations, Inversion of Matrices and Evaluation of Determinants", AERE-R 6899 (1971)

17) 堀上邦彦：私信

18) 磯田和男，大野豊（監）："FORTRANによる数値計算ハンドブック"，オーム社（1971）

19) Harms D.W., Keller R.F.: "A Direct Method Based on Projections for Solving Systems of Linear Equations", IS-3396 (1974)

20) Howell J.A.: "Algorithm 406, Exact Solution of Linear Equations Using Residual Arithmetic [F4]", Comm. ACM, 14, 180 (1971)

21) 鈴木忠和，森口欽一："MODULUS算法によるマトリックスの反転コード（EXACT）"（所内資料）（1968）

22) Reid J.K.: "Fortran Subroutines for Handling Sparse Linear Programming Bases", AERE-R 8269 (1976)

23) Paige C.C., Saunders M.A.: "Solution of Sparse Indefinite Systems of Equations and Least Squares Problems", SU-326-P30-29 (1974)

24) Reid J.K.: "A Fortran Subroutine for the Solution of Large Sparse Sets of Linear Equations by Conjugate Gradients", AERE-R 6545 (1970)

25) Reid J.K.: "Two Fortran Subroutines for Direct Solution of Linear Equations whose Matrix is Sparse, Symmetric and Positive-Definite", AERE-R 7119 (1972)

26) 朝岡卓見："連立非線形方程式の数値解法プログラム"，JAERI－Mレポート（公刊予定）

27) 富士通（株）："FACOM 230 M－VII FORTRAN IV－H 使用手引書（75SP－0280－1)"（1976）

28) Schwarz H.R., Rutishauser H., Stiefel E.: "Numerical Analysis of Symmetric Matrices" (Hertelendy P. tr.), Printice-Hall, Inc. (1973)

29) 田辺国士："定常一階線型反復法を加速するプログラム"，情報処理，13，263（1972）

30) Morris J.: "An Escalator Process for the Solution of Linear Simultaneous Equations", Philos. Mag., 37, 106 (1946)

31) 伊勢武治，西田雄彦，鈴木忠和："最近の最小自乗法の計算コード"，日本原子力学会誌，18，89（1976）

32) 富士通（株）："FACOM FORTRAN SSL II 使用手引書（99SP－0050－2)"（1977）

付録 1. ESCARS の例題と FORTRAN リスト

```
ISN   ST-NO                    SOURCE PROGRAM                                  SEQUENCE

      C          EX. FOR   ESCARS
  1         DIMENSION A(5,6),W(5,6),XE(3)
  2         NDIM=5
  3         N=3
  4         NP1=N+1
  5         DO 21 I=1,N
  6         XE(I)=I
  7         DO 21 J=1,N
  8         A(I,J)=N*(I-1)+J
  9      21 CONTINUE
 10         A(N,N)=0.
 11         WRITE (6,66) ((A(I,J),J=1,N),I=1,N)
 12         DO 35 I=1,N
 13         A(I,NP1)=0.
 14         DO 31 J=1,N
 15         A(I,NP1)=A(I,NP1)-A(I,J)*XE(J)
 16      31 CONTINUE
 17         A(NP1,I)=A(I,NP1)
 18      35 CONTINUE
 19         WRITE (6,66)  (A(I,NP1),I=1,N)
 20         IOP=2
 21         CALL ESCARS (A,W,NDIM,N,IOP,IERR)
 22         WRITE (6,44) NDIM,N,IOP,IERR
 23         WRITE (6,66)  (W(I,NP1),I=1,N)
 24         WRITE (6,66) (W(NP1,I),I=1,N)
 25      44 FORMAT (2X,10I12)
 26      66 FORMAT (2X,1P3E20.8)
 27         STOP
 28         END

ISN   ST-NO                    SOURCE PROGRAM                                  SEQUENCE

  1         SUBROUTINE ESCARS (A,W,NA,N,IOP,IERR)
      C*****
      C     SOLVE LINEAR EQUATION   A*X=B, AT*Y=B   BY
      C     ESCALATOR METHOD   FOR REAL MATRIX A
      C     IF A=E*D*FI, AT=G*D*HI THEN ET*H=I, GT*F=I
      C         WHERE, FI=INV. OF F, AT=TRANSP. OF A
      C         REFER TO   V.N. FADDEEVA (1959)
      C         IOP=1  SOLVE A*X=B,  IOP=2  BOTH
      C***  ENTER -B(I) TO A(I,N+1) IF IOP=1,
      C         FURTHER TO A(N+1,I) IF IOP=2
      C         NA.GE.N IF IOP=1, NA.GE.(N+1) IF IOP=2
  2         DIMENSION A(NA,1),W(NA,1)
  3         IERR=0
  4         NP1=N+1
  5         NE=NP1
  6         IF (IOP.EQ.1) NE=N
  7         DO 81 J=1,N
  8         JM1=J-1
  9         JP1=J+1
      C-----    D(J,J)
      C***  NEED NOT STORE D(J,J)
 10         DJJ=A(J,J)
 11         IF (J.EQ.1) GO TO 17
 12         DO 15 K=1,JM1
 13         DJJ=DJJ+A(J,K)*W(K,J)
      C***  NEED NOT CALCULATE FOR H
 14      15 CONTINUE
 15      17 CONTINUE
 16         IF (DJJ.EQ.0.) GO TO 91
      C-----            G(I,J)
      C***          G(I,J) STORED IN        W(J,I)
 17         DO 29 I=JP1,NP1
 18         GIJ=A(J,I)
 19         IF (J.EQ.1) GO TO 27
 20         DO 21 K=1,JM1
 21         GIJ=GIJ+A(K,I)*W(J,K)
 22      21 CONTINUE
 23      27 CONTINUE
 24         W(J,I)=GIJ/DJJ
 25      29 CONTINUE
      C-----                F(I,J+1)
      C***          F(I,J) STORED IN W(I,J)
 26         DO 39 I=1,J
 27         W(I,JP1)=-W(I,JP1)
 28         IF (I.EQ.J) GO TO 39
 29         DO 35 K=I+1,J
 30         W(I,JP1)=W(I,JP1)-W(K,JP1)*W(I,K)
 31      35 CONTINUE
 32      39 CONTINUE
 33         IF (IOP.EQ.1 .AND. J.EQ.N) GO TO 83
      C-----    E(I,J)    STORED IN W(I,J)
 34         DO 59 I=JP1,NE
 35         EIJ   =A(I,J)
 36         IF (J.EQ.1) GO TO 57
 37         DO 51 K=1,JM1
 38         EIJ=EIJ+A(I,K)*W(K,J)
 39      51 CONTINUE
```

```
ISN   ST-NO                    SOURCE PROGRAM      ( ESCARS )                    SEQUENCE

40       57 CONTINUE
41          W(I,J)=E(J/DJJ
42       59 CONTINUE
     C-----    H(I,J+1) IN W(J+1,I)
43          DO 69 I=1,J
44          W(JP1,I)=-W(JP1,I)
45          IF (I.EQ.J) GO TO 69
46          DO 65 K=I+1,J
47          W(JP1,I)=W(JP1,I)-W(JP1,K)*W(K,I)
48       65 CONTINUE
49       69 CONTINUE
50       81 CONTINUE
     C***  F(I,N+1) (W(I,N+1)) IS THE SOLUTION A*X=B
51       83 CONTINUE
     C***  H(I,N+1) (W(N+1,I)) IS THE SOLUTION AT*Y=B
52          GO TO 95
53       91 CONTINUE
54          WRITE (6,92) J
55       92 FORMAT (2X,' PIVOT=0. IN ESCARS AT ',I5,' -TH STAGE ')
56          IERR=J
57       95 CONTINUE
58          RETURN
59          END
```

```
 9.99999999E-01      1.99999999E+00      3.00000000E+00
 3.99999999E+00      5.00000000E+00      6.00000000E+00
 7.00000000E+00      7.99999999E+00      0.0
-1.40000000E+01     -3.20000000E+01     -2.30000000E+01
     5                3                   2                 0
 9.99999999E-01      1.99999999E+00      3.00000000E+00
 2.23333334E+01     -7.33333337E+00      3.00000000E+00
```

付録2. ESCASSの例題とFORTRANリスト

```
ISN   ST-NO                SOURCE PROGRAM                                    SEQUENCE

      C          EX. FOR   ESCASS
 1          DIMENSION A(5,6)
 2          NDIM=5
 3          N=3
 4          NP1=N+1
 5          FL=0.
 6          DO 23 I=1,N
 7          DO 21 J=1,I
 8          FL=FL+1.0
 9          A(I,J)=FL
10       21 CONTINUE
11          WRITE (6,66) (A(I,J),J=1,I)
12       23 CONTINUE
13          DO 37 I=1,N
14          BI=0.
15          DO 35 J=1,N
16          IF (J.LE.I) AK=A(I,J)
17          IF (J.GT.I) AK=A(J,I)
18          BI=BI+AK*J
      C***       EXACT SOL.     X(I)=I
19       35 CONTINUE
20          A(NP1,I)=-BI
21       37 CONTINUE
22          WRITE (6,66) (A(NP1,J),J=1,N)
23          CALL ESCASS (A,NDIM,N,IERR)
24          WRITE (6,44) NDIM,N,NP1,IERR
25          WRITE (6,66) (A(I,NP1),I=1,N)
26       44 FORMAT (2X,10I12)
27       66 FORMAT (2X,1P3E20.8)
28          STOP
29          END
```

```
ISN   ST-NO                SOURCE PROGRAM                                    SEQUENCE

 1          SUBROUTINE ESCASS (A,NA,N,     IERR)
      C=====
      C                    SOLVE    LINEAR EQUATION   A*X=B   BY
      C     ESCALATOR METHOD   FOR REAL SYMMETRIC MATRIX A
      C     A=E*D*FI   WHERE ET*F=I
      C         REFER TO   V.N. FADDEEVA (1959)
      C***  ENTER      DIAGONAL ØND LEFT LOWER  PART FOR MATRIX A
      C***  ENTER -B(I) TO A(N+1,I)
 2          DIMENSION A(NA,1)
 3          IERR=0
 4          NP1=N+1
 5          DO 81 J=1,N
 6          JM1=J-1
 7          JP1=J+1
      C-----     D(J,J)
      C***  NEED NOT STORE D(J,J)
 8          DJJ=A(J,J)
 9          IF (J.EQ.1) GO TO 33
10          DO 31 K=1,JM1
11          DJJ=DJJ+A(J,K)*A(K,J)
12       31 CONTINUE
13       33 CONTINUE
14          IF (DJJ.EQ.0.) GO TO 91
      C-----     E(I,J)
      C***       E (I,J) IS STORED IN A(J,I)
15          DO 39 I=JP1,NP1
16          EIJ=A(I,J)
17          IF (J.EQ.1) GO TO 39
18          DO 37 K=1,JM1
19          EIJ    =EIJ    +A(I,K)*A(K,J)
20       37 CONTINUE
21       39 A(J,I)=EIJ/DJJ
      C-----     F(I,J+1)
      C***       F(I,J+1) IS STORED IN A(I,J+1) DELETING     E(J+1,I)
22          DO 43 I=1,J
23          A(I,JP1)=-A(I,JP1)
24          IF (I.EQ.J) GO TO 43
25          DO 41 K=I+1,J
26       41 A(I,JP1)=A(I,JP1)-A(K,JP1)*A(I,K)
27       43 CONTINUE
28       81 CONTINUE
      C***  A(I,N+1) IS THE SOLUTION X(I)
29          GO TO 95
      C-----                ERROR
30       91 CONTINUE
31          WRITE (6,92) J
32       92 FORMAT (2X,' PIVOT =0. IN ESCASS. AT ',I5,' -TH STAGE ')
33          IERR=J
34       95 CONTINUE
35          RETURN
36          END
```

```
 9.99999999E-01
 1.99999999E+00      3.00000000E+00
 3.99999999E+00      5.00000000E+00      6.00000000E+00
-1.70000000E+01     -2.30000000E+01     -3.20000000E+01
      5            3            4            0
 9.99999999E-01      1.99999999E+00      3.00000000E+00
```

付録 3. DPRM の例題と FORTRAN リスト

```
ISN   ST-NO                SOURCE PROGRAM                                SEQUENCE

      C          EX. FOR   DPRM
      C      EX. 9 ON H3 (P48)
 1           DIMENSION A(4,5),B(4,5),X(4)
 2           N=4
 3           EPS=1.0E-6
 4           READ (5,12) ((A(I,J),J=1,N),I=1,N)
 5           READ (5,12)  (A(I,5),I=1,N)
 6           READ (5,12)   X
 7           WRITE (6,46) ((A(I,J),J=1,N),I=1,N)
 8           WRITE (6,46)  (A(I,5),I=1,N)
 9           CALL DPRM (N,A,B,EPS,IER)
10           WRITE (6,44) N,IER
11           WRITE (6,42)
12           WRITE (6,46) X
13           WRITE (6,46)  (A(I,5),I=1,N)
14        12 FORMAT (4F6.2)
15        42 FORMAT (2X,' EXACT SOL. ')
16        44 FORMAT (2X,10I12)
17        46 FORMAT (2X,1P4E20.8)
18           STOP
19           END
```

```
ISN   ST-NO                SOURCE PROGRAM                                SEQUENCE

      CV         REFER TO   H3
 1           SUBROUTINE DPRM (N,A,B,EPS,IER)
 2           DIMENSION A(1),B(1)
 3           NM=N-1
      C-----     REDUCTION
 4           DO 11 L=1,NM
 5           NL=N-L+1
 6           L2=2**(L-1)
 7           IF (MOD(L,2).EQ.1) CALL ROA (A,NL,L2,B)
 8           IF (MOD(L,2).EQ.0) CALL ROA (B,NL,L2,A)
 9        11 CONTINUE
      C-----     1-DIM.
10           IF (MOD(N,2).EQ.1) CALL SOLA (N,A,EPS,IER)
11           IF (MOD(N,2).EQ.0) CALL SOLA (N,B,EPS,IER)
12           IF (IER.NE.0) GO TO 91
      C-----     SOLVE REDUCED SYSTEM
13           DO 85 L=1,NM
14           NL=N-L
15           L2=2**NL
16           IF (MOD(NL,2).EQ.1) CALL SRS (B,L,L2,A,EPS,IER)
17           IF (MOD(NL,2).EQ.0) CALL SRS (A,L,L2,B,EPS,IER)
18           IF (IER.NE.0) GO TO 91
19        85 CONTINUE
20           GO TO 95
21        91 WRITE (6,92) IER
22        92 FORMAT (2X,' ERROR IN DPRM, IER= ',I12)
23        95 CONTINUE
24           RETURN
25           END
```

```
ISN   ST=NO                    SOURCE PROGRAM                                  SEQUENCE

      CV          REFER TO   H3
  1         SUBROUTINE RGA (A,NA,LA,B)
      C           REDUCE A TO B FOR PROJECTION METHOD
  2         DIMENSION A(1),B(1)
  3         NB=NA-1
  4         MA=(NA+LA)*NA
  5         WRITE (1) (A(M),M=1,MA)
      C-----      MATRIX
  6         DO 36 I=1,NB
  7         DO 36 J=I,NB
  8         IJ=(J-1)*NB+I
  9         JI=(I-1)*NB+J
 10         KI=(I-1)*NA+1
 11         KJ=(J-1)*NA+1
 12         CALL MCO3AS (A(KI),1,A(KJ),1,0.,B(IJ),NA,0)
 13      36 B(JI)=B(IJ)
      C-----      VECTOR
 14         DO 40 J=1,LA
 15         DO 40 I=1,NB
 16         IJ1=(NB+2*(J-1))*NB+I
 17         IJ2=IJ1+NB
 18         KI=(I-1)*NA+1
 19         KJ=(NA+J-1)*NA+1
 20         KA=(NA-1)*NA+1
 21         CALL MCO3AS (A(KI),1,A(KJ),1,0.,B(IJ1),NA,0)
 22         CALL MCO3AS (A(KI),1,A(KA),1,0.,B(IJ2),NA,0)
 23      40 B(IJ2)=B(IJ1)-B(IJ2)
 24         RETURN
 25         END
```

```
ISN   ST=NO                    SOURCE PROGRAM                                  SEQUENCE

      CV          REFER TO   H3
  1         SUBROUTINE SOLA (N,A,EPS,IER)
      C           SOLVE 1-DIM. PROBLEM
  2         DIMENSION A(1)
  3         N2=2**(N-1)
  4         IER=0
  5         IF (ABS(A(1)).GT.EPS) GO TO 31
  6         IER=1
  7         GO TO 51
  8      31 CONTINUE
  9         DO 41 J=1,N2
 10         JA=1+J
 11         A(JA)=A(JA)/A(1)
 12      41 CONTINUE
 13      51 CONTINUE
 14         RETURN
 15         END
```

```
ISN   ST-NO                SOURCE PROGRAM                                SEQUENCE

      CV        REFER TO   H3
  1         SUBROUTINE SRS (B,NB,LB,A,EPS,IER)
      C     SOLVE REDUCED SYSTEM
  2         DIMENSION A(1),B(1)
  3         NA=NB+1
  4         LA=LB/2
  5         MA=(NA+LA)*NA
  6         BACK SPACE 1
  7         READ (1) (A(M),M=1,MA)
  8         BACK SPACE 1
  9         DO  81 J=1,LA
 10         JB1=(NB+2*(J-1))*NB+1
 11         JB2=JB1+NB
 12         IZC=0
 13         DO  45 I=1,NA
 14         IJ=(NA+J-1)*NA+I
 15         KA=(NA-1)*NA+I
 16         CALL MC03AS (A(I) ,NA,B(JB1),1,A(IJ),R1,NB,1)
 17         IF (ABS(R1).GT.EPS) GO TO 41
 18         IZC=IZC+1
 19         GO TO 45
 20      41 CONTINUE
 21         CALL MC03AS (A(I),NA,B(JB2),1,A(IJ),R2,NB,1)
 22         R2=R2-A(KA)
 23         T=R2/R1
 24         T1=1.0-T
 25         IF (ABS(T1).GT.EPS) GO TO 71
 26      45 CONTINUE
 27         IF (IZC.GE.NA) GO TO 51
 28         IER=NA
 29         GO TO 91
 30      51 CONTINUE
      C-----     SOL. FOR VECTOR R1 IS ZERO
 31         DO 55 I=1,NB
 32         IA=(NA+J-1)*NA+I
 33         IB=JB1+I-1
 34         A(IA)=B(IB)
 35      55 CONTINUE
 36         IA=(NA+J)*NA
 37         A(IA)=0.
 38      71 CONTINUE
      C-----     SOL.   FOR T.NE.1.
 39         DO 75 I=1,NB
 40         IA=(NA+J-1)*NA+I
 41         IB1=JB1+I-1
 42         IB2=IB1+NB
 43         A(IA)=B(IB1)-(B(IB1)-B(IB2))/T1
 44      75 CONTINUE
 45         IA=(NA+J)*NA
 46         A(IA)=1.0/T1
 47      81 CONTINUE
 48      91 CONTINUE
 49         RETURN
 50         END
```

```
     5.00000000E+00      9.99999999E-01      1.99999999E+00      1.99999999E+00
     3.00000000E+00      5.00000000E+00     -1.00000000E+00      3.00000000E+00
     9.99999999E-01     -3.00000000E+00      5.00000000E+00      7.00000000E+00
     3.99999999E+00      9.99999999E-01      0.0                 5.00000000E+00
     1.99999999E+00      6.00000000E+00     -1.40000000E+01      0.0
        4            0
EXACT SOL.
     9.99999999E-01      9.99999999E-01     -1.00000000E+00     -1.00000000E+00
     9.99999836E-01      1.00000020E+00     -9.99999642E-01     -1.00000000E+00
```

付録 4. MTESの例題とFORTRANリスト

```
ISN   ST-NO                SOURCE PROGRAM                                SEQUENCE

      C          EX. FOR   MTES
      C      EXAMPLE   1 - 1
  1          DIMENSION A(6,3),B(2,15),V(6),X(6)
  2          DIMENSION XE(6)
  3          N=6
  4          M=2
  5          L=N/M
  6          NFORM=2
  7          EPS=1.0E-20
  8          A(1,1)=0.
  9          A(6,3)=0.
 10          DO 21 I=1,6
 11          XE(I)=I
 12          A(I,2)=50+I
 13          IF (I.LE.5) A(I,3)=20+I
 14          IF (I.GE.2) A(I,1)=10+I-1
 15          V(I)=A(I,1)*(I-1)+A(I,2)*I+A(I,3)*(I+1)
 16       21 CONTINUE
 17          DO 51 I=1,N
 18          WRITE (6,56) (A(I,J),J=1,L),XE(I),V(I)
 19       51 CONTINUE
 20          CALL MTES (A,B,6,2,2,V,EPS,ILL,X)
 21          WRITE (6,*) N,M,L,NFORM,ILL
 22          WRITE (6,32)
 23       32 FORMAT (2X,' EXACT SOL. , X(I)=I ')
 24          WRITE (6,66) (X(I),I=1,N)
 25       56 FORMAT (2X,1P5E16.8)
 26       66 FORMAT (2X,1P6E16.8)
 27          STOP
 28          END
```

```
ISN   ST-NO                SOURCE PROGRAM                                SEQUENCE

  1          SUBROUTINE MTES (A,B,N,M,NFORM,V,EPS,ILL,X)
      C
      CCCCCCCCCC   MULTI TERM EQUATION   CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      C                    SEEK THE SOLUTION OF MULTI-DIAGONAL LINEAR EQUATION
      C                              (SINGLE PRECISION)
      C                                        MAR. 27 '72   T.FUJIMURA
      C
      CONDITIONS -----
      C            ABOUT THE (N,N)-MATRIX
      C     1.   N .GE. 1
      C     1.   IT IS (2M-1)-PLE (1.LE.M.LE.N) DIAGONAL MATRIX SUCH THAT
      C     1.   M IS A DIVISOR OF N (LET N/M=L)
      C     1.   ANY (M,M) DIAGONAL SUBMATRIX IS NON-SINGULAR
      C
      COMMENTS
      C     B ------ COEFFICIENTS ARE STORED
      C              IT SHOULD BE DECLARED 'DIMENSION B(M,2N+3)' IN
      C                 USERS PROGRAM
      C              MULTI DIAGONAL PART OF (N,N)-MATRIX IS STORED IN
      C                 FIRST 2N-L-M+1 COLUMNS
      C              LAST M+L+2 COL. ARE WORK AREA
      C     N ------ ORDER OF SOURCE MATRIX
      C     M ------ NUMBER SUCH THAT 2M-1 = NO. OF TERM
      C                                  (= MULTIPLICITY OF DIAGONAL)
      C     NFORM -- PRESENT DATA FORM
      C                 NFORM = 1   IN DECLARED SQUARE ARRAY A(N,N) AS USUAL
      C                 NFORM = 2   IN DECLARED RECTANGULAR ARRAY A(N,2M-1) AS
      C                                FOLLOWS
      C                                        A(I,J) IS (I,I+J-M) ELEMENT
      C                                           FOR (M+1).LE.(I+J).LE.(N+M)
      C     V ------ CONSTANT VECTOR DECLARED 'V(N)'
      C     EPS ---- SEVERITY OF NON-SINGULARITY CHECK
      C     ILL ---- PARAMETER FOR ILL CONDITION STOP
      C                 ILL = 11   N.LE.0
      C                 ILL = 12   M.LE.0 OR M.GT.N
      C                 ILL = 13   MOD(N,M).NE.0
      C                 ILL = 20   SINGULARITY CHECKED
      C     X ------ SOL. VECTOR DECLARED 'X(N)'
      C
      CAUTION      B IS DELETED AFTER CALLING
      C                 THIS ROUTINE
      C
  2          DIMENSION A(N,1),B(M,1),X(1),V(1)
      C
  3          ILL=0
      C
      C========= ARG. CHECK
  4          IF (N.LE.0) ILL=11
  5          IF (M.LE.0 .OR. M.GT.N) ILL=12
  6          IF (MOD(N,M).NE.0) ILL=13
  7          IF (ILL.GT.0) GO TO 411
      C
      C========= CONSTANTS
  8          L=N/M
  9          NMM=N-M
 10          MM1=M-1
 11          LM1=L-1
```

```
ISN   ST-NO                 SOURCE PROGRAM      ( MTES   )                SEQUENCE

 12         NC=N+MM1*LM1
      C                     END COL. OF C AT B
 13         NF=NC+M
      C                     END COL. OF F AT B
 14         NG=NF+L
      C                     END COL. OF G AT B
 15         NT=NG+1
      C                     END COL. OF T AT B
 16         NP=NT+1
      C                     END COL. OF P AT B
      C
      C========= DATA ARRANGEMENT
      C                                (ASSIGN A TO B)
 17         DO 31 I=1,N
 18         J1=I-M+1        +MM1
 19         J2=I+M-1          +MM1
 20         IB=MOD(I-1,M)+1
 21         IK=(I-IB)/M +1
 22         DO 31 JPLUS=J1,J2
 23         J=JPLUS-MM1
 24         IF (J.LT.1 .OR. J.GT.N) GO TO 31
      C          --- (I,J) ELEMENT
 25         ETIJ=A(I,J)
      C                        FROM RECTANGULAR
 26         IF (NFORM.EQ.2) ETIJ=A(I,J-I+M)
      C          --- BLOCK
 27         JA=MOD(J-1,M) +1
 28         JK=(J-JA)/M +1
 29         IF (IK.GT.JK) GO TO 11
 30         IF (IK.LT.JK) GO TO 21
      C          --- DIAGONAL SUBMATRIX
 31         B(IB,J)=ETIJ
 32         GO TO 31
      C          --- UPPER TRIANG. SUBMAT.  C(IK,IK-1)
 33      11 CONTINUE
 34         JB=JA-1 +N+(JK-1)*MM1
 35         B(IB,JB)=ETIJ
 36         GO TO 31
      C          --- LOWER TRIANG. SUBMAT.  C(IK,IK+1)
 37      21 CONTINUE
 38         JB=JA +N+(JK-2)*MM1
 39         B(IB,JB)=ETIJ
 40      31 CONTINUE
      C
      C========= F,G CAL.
 41         DO 71 K=1,L
 42         KM1=K-1
 43         KM2=K-2
 44         K1M=KM1*M
 45         K2M=KM2*M
 46         KL=N+KM2*MM1
 47         KG=NF+K
 48         IF (K.GT.1) GO TO 41
      C          --- G(1) CAL.
 49         DO 33 I=1,M
 50         B(I,NF+1)=V(I)
 51      33 CONTINUE
 52         GO TO 69
      C          --- C(K,K-1)*C(K-1,K-1)**(-1) CAL.
 53      41 CONTINUE
 54         DO 45 I=1,M
 55         IP1=I+1
 56         DO 45 J=1,M
 57         JB=NC+J
 58         B(I,JB)=0.
 59         IF (I.GE.M) GO TO 45
 60         DO 43 IJ=IP1,M
 61         B(I,JB)=B(I,JB)+B(I,KL+IJ-1)*B(IJ,K2M+J)
 62      43 CONTINUE
 63      45 CONTINUE
      C          --- G(K) CAL.
 64         DO 53 I=1,M
 65         B(I,KG)=V(K1M+I)
 66         DO 53 J=1,M
 67         B(I,KG)=B(I,KG)-B(I,NC+J)*B(J,KG-1)
 68      53 CONTINUE
      C          --- F(K) CAL.
 69         DO 65 I=1,M
 70         DO 65 J=1,M
 71         IF (J.GE.M) GO TO 65
 72         JP1=J+1
 73         DO 63 IJ=JP1,M
 74         B(I,K1M+J)=B(I,K1M+J)-B(I,NC+IJ)*B(IJ,KL+J)
 75      63 CONTINUE
 76      65 CONTINUE
 77      69 CONTINUE
      C
      C          --- F(K)**(-1) CAL.
      C                                (M,M)=MATRIX INVERSION
 78         DO 101 I=1,M
 79         B(I,NP)=I+0.3
 80         DO 101 J=1,M
 81         JC=K1M+J
 82         JF=NC+J
 83         B(I,JF)=B(I,JC)
 84     101 CONTINUE
 85         DO 1000 MM=1,M
 86         IJK=1
 87         R=B(1,NC+1)
 88         IF (MM.EQ.M) GO TO 10
 89         IJV=M-MM+1
 90         DO 102 I=2,IJV
 91         IC=NC+I
 92         S=B(I,IC)
 93         IF (ABS(S).LE.ABS(R)) GO TO 102
 94         IJK=I
 95         R=S
 96     102 CONTINUE
 97      10 CONTINUE
 98         IF (ABS(R).LT.EPS) GO TO 999
 99         IF (IJK.EQ.1) GO TO 20
100         IJL=B(1,NP)
101         B(1,NP)=B(IJK,NP)
102         B(IJK,NP)=IJL+0.3
```

```
ISN   ST-NO                SOURCE PROGRAM      ( MTES   )                    SEQUENCE

103          DO 103 I=1,M
104          TEMP=B(I,NC+1)
105          B(I,NC+1)=B(I,NC+IJK)
106          B(I,NC+IJK)=TEMP
107      103 CONTINUE
      C
108       20 CONTINUE
109          B(1,NT)=1./R
110          IF (M.LE.1) GO TO 333
111          DO 104 I=2,M
112          B(I,NT)=B(1,NC+I)/R
113      104 CONTINUE
114          DO 106 I=1,MM1
115          DO 105 J=1,MM1
116          B(I,NC+J)=B(I+1,NC+J+1)-B(J+1,NT)*B(I+1,NC+1)
117      105 CONTINUE
118          B(I,NF)=-B(1,NT)*B(I+1,NC+1)
119      106 CONTINUE
120          IJL=B(1,NP)
121          DO 107 I=1,MM1
122          B(I,NP)=B(I+1,NP)
123          B(M,NC+I)=B(I+1,NT)
124      107 CONTINUE
125          B(M,NP)=IJL+0.3
126      333 CONTINUE
127          B(M,NF  )=B(1,NT)
128     1000 CONTINUE
      C
129          DO 110 I=1,MM1
130          IPV=B(I,NP)
131          IF (I.EQ.IPV) GO TO 110
132          IP1=I+1
133          DO 108 J=IP1,M
134          JPV=B(J,NP)
135          IF (I.EQ.JPV) GO TO 40
136      108 CONTINUE
137       40 CONTINUE
138          IJL=B(I,NP)
139          B(I,NP)=B(J,NP)
140          B(J,NP)=IJL+0.3
141          DO 109 MM=1,M
142          TEMP=B(I,NC+MM)
143          B(I,NC+MM)=B(J,NC+MM)
144          B(J,NC+MM)=TEMP
145      109 CONTINUE
146      110 CONTINUE
147          DO 200 I=1,M
148          DO 200 J=1,M
149          B(I,K1M+J)=B(I,NC+J)
150      200 CONTINUE
      C---
151       71 CONTINUE
      C
      C========= X CAL.
152          DO 381 K=1,L
153          KV=L-K+1
154          IF (KV.LT.L) GO TO 321
      C
      C            --- X(L) CAL.
155          DO 315 I=1,M
156          I1=NMM      +I
157          X(I1)=0.
158          DO 315 J=1,M
159          J1=NMM+J
160          X(I1)=X(I1)+B(I,J1)*B(J,NG)
161      315 CONTINUE
162          GO TO 381
      C========= X(KV) CAL.
      C            --- F(KV)**(-1) * G(KV) CAL.
163      321 CONTINUE
164          KV1=(KV-1)*M
165          KVM=KV*M
166          KVN=(KV-1)*MM1
167          DO 355 I=1,M
168          I1=KV1+I
169          X(I1   )=0.
170          DO 355 J=1,M
171          J1=KV1+J
172          X(I1)=X(I1)+B(I,J1)*B(J,NF+KV)
173      355 CONTINUE
      C            --- F(KV)**(-1) * C(KV,KV+1) CAL.
174          DO 363 I=1,M
175          DO 363 J=1,M
176          J1=NC+J
177          B(I,J1   )=0.
178          IF (J.GE.M) GO TO 363
179          JP1=J+1
180          DO 361 IJ=JP1,M
181          B(I,J1)=      B(I,J1)+B(I,KV1+IJ)*B(IJ,N+KVN+J)
182      361 CONTINUE
183      363 CONTINUE
      C            --- X(KV) CAL.
184          DO 373 I=1,M
185          I1=KV1+I
186          DO 373 J=1,M
187          J1=NC+J
188          X(I1)=X(I1)-B(I,J1)*X(KVM+J)
189      373 CONTINUE
190      381 CONTINUE
191          GO TO 199
      C
      C========= ERROR CONDITION
      C            --- ARG.ERROR
192      411 CONTINUE
193          WRITE (6,412) ILL
194      412 FORMAT (10X,  'ILL COND. ON ARG. IN SUBR. MTES , ILL= ',I5 )
195          GO TO 199
      C            --- FAIL INVERSION
196      999 CONTINUE
197          ILL=20
198          WRITE (6,1001) ILL,K
199     1001 FORMAT (10X,  'UNABLE INVERSION IN SUBR. MTES , ILL= ',
           1     I5,5X,  'F(',I5,  '  )' )
200          IJK=M/8 +1
201          DO 112 IJL=1,IJK
```

```
ISN   ST-NO                 SOURCE PROGRAM      ( MTES   )                    SEQUENCE

202          IJK1=1+(IJL-1)*7
203          IJK2=IJK1+6
204          IF (IJL.EQ.IJK) IJK2=M
205          WRITE (6,1002) ((I),I=IJK1,IJK2)
206     1002 FORMAT (2X,7(8X,I7))
207          DO 111 I=1,M
208          WRITE (6,1003) I,(B(I,NC+J),J=IJK1,IJK2)
209      111 CONTINUE
210     1003 FORMAT (5H0     ,I5,7E15.5)
211          WRITE (6,1004)
212     1004 FORMAT (//)
213      112 CONTINUE
214      199 CONTINUE
215          RETURN
216          END
```

```
 0.0              5.09999999E+01  2.09999999E+01  9.99999999E-01  9.29999999E+01
 1.09999999E+01   5.19999999E+01  2.19999999E+01  1.99999999E+00  1.80999999E+02
 1.19999999E+01   5.29999999E+01  2.29999999E+01  3.00000000E+00  2.74999999E+02
 1.29999999E+01   5.39999999E+01  2.39999999E+01  3.99999999E+00  3.74999999E+02
 1.39999999E+01   5.49999999E+01  2.49999999E+01  5.00000000E+00  4.80999999E+02
 1.49999999E+01   5.59999999E+01  0.0             6.00000000E+00  4.10999999E+02
        6              2              3            2              0
EXACT SOL. , X(I)=I
 9.99999970E-01   2.00000005E+00  2.99999994E+00  4.00000011E+00  5.00000000E+00  6.00000000E+00
```