JAERI-M
8253

EISPACK-J：固有値問題を解く副プログラム・パッケージ

（ＳＳＬの拡充とベンチマーク・テスト No.8）

1979年5月

藤村　統一郎・筒井　恒夫

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

JAERI－M 8253

EISPACK－J：固有値問題を解く副プログラム・パッケージ
（SSLの拡充とベンチマーク・テスト No.8）

日本原子力研究所東海研究所原子炉工学部
藤村統一郎・筒井恒夫

（1979年4月25日　受理）

固有値問題を解く副プログラム・パッケージEISPACK－Jが開発されるとともに，変った機能を持ついくつかの副プログラムが整備された。これらの副プログラムは複素行列の標準問題や実行列の一般問題を解くほか，必要な固有値や固有ベクトルを求める特殊問題も解くことができる。これらはベンチマーク・テストを通して既存の副プログラムと比較され，その特徴が明らかにされる。

ベンチマーク・テストには実際規模の問題を含む多くのテスト問題が用意され，各副プログラムの計算に要する記憶領域，計算時間，解の精度について検討された。その結果，Householder法，QR法それに逆反復法に基づいているEISPACK－J の副プログラムは計算時間および精度について優れていることが示された。

JAERI-M 8253

EISPACK-J: Subprogram Package for Solving Eigenvalue Problems

(Development and Benchmark Test of SSL, No.8)

Toichiro FUJIMURA and Tsuneo TSUTSUI

Division of Reactor Engineering, Tokai Research Establishment,

JAERI

(Received April 25, 1979)

EISPACK-J, a subprogram package for solving eigenvalue problems, has been developed and subprograms with a variety of functions have been prepared. These subprograms can solve standard problems of complex matrices, general problems of real matrices and special problems in which only the required eigenvalues and eigenvectors are calculated. They are compared to existing subprograms, showing their features through benchmark tests.

Many test problems, including realistic scale problems, are provided for the benchmark tests. Discussions are made on computer core storage and computing time required for each subprogram, and accuracy of the solution. The results show that the subprograms of EISPACK-J, based on Householder, QR and inverse iteration methods, are the best in computing time and accuracy.

Keywords: Eigenvalue Problem, Complex Matrix, General Problem, Benchmark Test, Computing Time, Accuracy, EISPACK-J Subprogram Package, Householder Method, Inverse Iteration.

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隈なくサーベイし，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

固有値問題が離散化されたとき，行列表示を用いて一般に

$$\mathbf{Ax} = \lambda\mathbf{x} \tag{1}$$

と書ける。[1)2)] この固有値や固有ベクトルを求める数値解法は，行列の性質と，必要とする固有値や固有ベクトルの数などにより変わるほか，同一の問題についても異った解法を用いることができる[3)〜8)]。従来は固有多項式（characteristic polynomial）を定め，その根として固有値を求める直接法（direct method）[3)4)6)7)] に依っていたが，現在はHouseholder 法で行列をHessenberg 形（form）に変換した後，QR法で固有値を求めるといったような反復法（iterative method）[5)7)]が優勢である。しかし，どの解法が良いかはあくまでも解かれる問題による。

一方，数値解法はHouseholder 法といった基幹をなす部分のほかに，精度を上げるための軸選択（pivoting）[5)7)] とか，収束を速めるための原点移動（origin shift）[4)5)]などの細かい技法を含む場合が多い。また，同じ技法を用いても，QR法における場合のように，プログラムの作り方により精度に微妙な違いを生じることがある[5)]。

さて，実際に問題が与えられ，これを解くことを想定してみよう。現在原研では3つの科学用サブルーチン（副プログラム）・ライブラリー（SSL）[9)〜12)] が公開されているが，多くの固有値問題を解くには充分と言えず，その拡充が必要である。更に，どのサブルーチン（ルーチン）がどういう問題に向くかの予備知識も必要である。SSLの拡充に際しては米国ANL (Argonne National Laboratory）のパーケージEISPACK−2[13)]を発展させたEISPACK−J を開発するとともに，8件のルーチンを整備し，これに加えた。EISPACK−2 は，ありふれた固有値問題の殆んどを扱うことができるが，多くの場合，行列の変換を行うルーチンとその組合せの（component）固有値を求めるルーチンを呼ぶというふうになっている。EISPACK−J では，このような不便を解消するとともに，行列が縮退した（degenerated）場合の機能が追加された。また，既存のルーチンを含め，これらの特徴を明らかにするために，4件の実際規模の問題を含む12件の問題についてベンチマーク・テストも実施した。

本稿では，これらSSLの拡充について述べられるほか，ベンチマーク・テストの結果の検討も行われる。なお，本論に入るに先立ち，全章を通して使われる慣用的な記号[3)4)7)14)]を予じめ掲げておくが，特に説明を要するものはその都度説明されよう。

Notations

m	number of rows of degenerated matrix $\mathbf{A}$
n	number of columns of degenerated $\mathbf{A}$ or order of $\mathbf{A}$
ℓ	number of columns of matrix $\mathbf{C}$
$\mathbf{I}_n = (\delta_{ij})$	unit matrix of order n
$\mathbf{A} = (a_{ij}) = (\alpha_{ij} + i\beta_{ij})$	(m, n) or (n, n) matrix
$\mathbf{A}^t = (a_{ji})$	transpose of $\mathbf{A}$
$\overline{\mathbf{A}} = (\overline{a}_{ij})$	conjugate of $\mathbf{A}$
$\mathbf{A}^* = (\overline{a}_{ji})$	adjoint of $\mathbf{A}$
$\mathbf{A}^{-1} = (g_{ij})$	inverse or generalized inverse of $\mathbf{A}$
$\mathbf{B} = (b_{ij})$	(n, n) matrix of general problem
$\mathbf{C} = (c_{ij})$	(m, ℓ) or (n, ℓ) matrix of constant vectors
$\lambda_j = \sigma_j + i\tau_j$	eigenvalues
$\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$	diagonal matrix composed of eigenvalues
$\mathbf{X} = (x_{ij}) = (\xi_{ij} + i\eta_{ij})$	matrix of right eigenvectors
$\mathbf{Y} = (y_{ij})$	matrix of left eigenvectors
$\mathbf{U} = (u_{ij})$	(m, n) matrix of singular value decomposition of $\mathbf{A}$
$\mathbf{V} = (v_{ij})$	(n, n) matrix of singular value decomposition of $\mathbf{A}$

2. 固有値問題の型

この章では固有値問題で頻繁に現れる問題の型について考察するが，対応するルーチンが存在するものに主眼がおかれる。また，同じ種類の問題でも，行列の性質により解き方やデータの格納の仕方が変わるのでそれを解くプログラムは違ったものになる。

(1)式において行列$\mathbf{A}$の次数がn であるとしよう。このとき(1)式をみたす固有値λと固有ベクトル$\mathbf{x}$を求める問題をn 次の標準問題（standard problem）という。これに対し，(1)式を特別な場合として含む

$$\mathbf{Ax} = \lambda\mathbf{Bx} \tag{2}$$

は一般問題（generalized problem）と呼ばれる。一般問題はn 次の行列$\mathbf{B}$が正則な場合，(2)式に逆行列$\mathbf{B}^{-1}$を掛けることにより容易に標準問題に帰着されるが，数値計算上このような処理をすることは得策でなく，[7)15)]別の問題として扱われる。

一方，これら2つの方程式（系）は，全ての固有値λ_jと固有ベクトル$\mathbf{x}_j$（j ＝1～n ）を求めるとき完全問題（exact problem），必要な一部の固有値や固有ベクトルを求めるとき特殊問題（special problem）と呼ばれる。実際にも，例えば最大の固有値だけ求めるといったような後者の場合は多い。

これらの問題はその1つを考えてみても，系を表わす行列の性質によりその解法は変ってくる。その主なものは行列が複素係数か実係数か，また対称性や帯構造かどうか等であるが，これらは後で具体的に述べられる。

今回開発されたパッケージEISPACK－J は，元のEISPACK－2[13)]を含め，これらの様々な問題を扱うことができるほか，一般問題の変形（variant ）

$$\mathbf{ABX} = \lambda\mathbf{X} \tag{3}$$

も扱うことができる。

この場合，$\mathbf{A}$，$\mathbf{B}$の一方が対称正定値（symmetric positive definite ）でなければならない。また，行列が縮退した場合，換言すれば$\mathbf{A}$が長方形行列である場合の特異値分解（singular value decomposition ）

$$\mathbf{A} = \mathbf{U}\Lambda\mathbf{V} \tag{4}$$

も可能である。このとき$\mathbf{A}$をm行n 列とすれば，$\mathbf{U}$は同じく（m，n ）行列，$\mathbf{V}$は（n，n ）行列でΛは$\mathbf{A}$の特異値λ_j（j ＝1～n ）からなる対角行列である。この分解は最小自乗法（least squares fit ）

$$\mathbf{Ax} = \mathbf{c} \tag{5}$$

など多方面に応用される。これらの問題では，正方行列に対する連立一次方程式（linear equation ）

$$\mathbf{Ax} = \mathbf{c} \tag{6}$$

を解くことが不可欠となるが，長方形行列に対する同次方程式（ homogeneous equation ）

$$\mathbf{Ax} = \mathbf{o} \tag{7}$$

を解くことも必要となるので，これらの機能も備えられている。

3. 標準問題の数値解法

標準問題(1)式は同じ次数の代数方程式（ algebraic equation ）と対応しているので5次以上の場合は理論的にも有限図の演算では解が得られない。[5)14)] 実際にはある中間的な段階を経て計算を行っているが，古くはまず固有多項式の係数を定め，それから代数方程式として解く直接法[3)4)6)7)]に依っていた。しかしこの方法は係数のわずかな誤差が多項式の根に大きな影響を与えるため，[3)5)] 最近は殆んど反復法[5)7)]に依っている。それは行列をより簡単な3重対角行列やHessenberg 行列に変換し，それから独自の方法を用いて固有値を求める方法である。

固有ベクトルは多くの場合固有値と同時に計算される。[3)~7)] これらの過程を図示したのがFig. 1であるが，これらの基本的概念は一般問題にもあてはまる。[4)7)15)] 以下各解法を詳しく見て行こう。

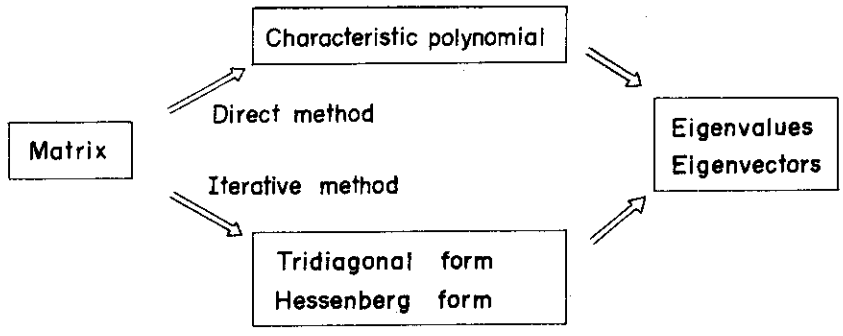

Fig. 1 Process of direct and iterative method

3.1 直接法

この節では複素数の系を考えよう。$\mathbf{A} = (a_{ij})$（ i , j ＝1～n ）とするとき，(1)式から$\mathbf{x}$ を消去した式は

$$\mathbf{Ax} = \mathbf{c} \tag{6}$$

を解くことが不可欠となるが，長方形行列に対する同次方程式（ homogeneous equation ）

$$\mathbf{Ax} = \mathbf{o} \tag{7}$$

を解くことも必要となるので，これらの機能も備えられている。

3. 標準問題の数値解法

標準問題(1)式は同じ次数の代数方程式（ algebraic equation ）と対応しているので5次以上の場合は理論的にも有限回の演算では解が得られない。[5)14)] 実際にはある中間的な段階を経て計算を行っているが，古くはまず固有多項式の係数を定め，それから代数方程式として解く直接法[3)4)6)7)]に依っていた。しかしこの方法は係数のわずかな誤差が多項式の根に大きな影響を与えるため，[3)5)] 最近は殆んど反復法[5)7)]に依っている。それは行列をより簡単な3重対角行列やHessenberg 行列に変換し，それから独自の方法を用いて固有値を求める方法である。

固有ベクトルは多くの場合固有値と同時に計算される。[3)~7)] これらの過程を図示したのがFig. 1であるが，これらの基本的概念は一般問題にもあてはまる。[4)7)15)] 以下各解法を詳しく見て行こう。

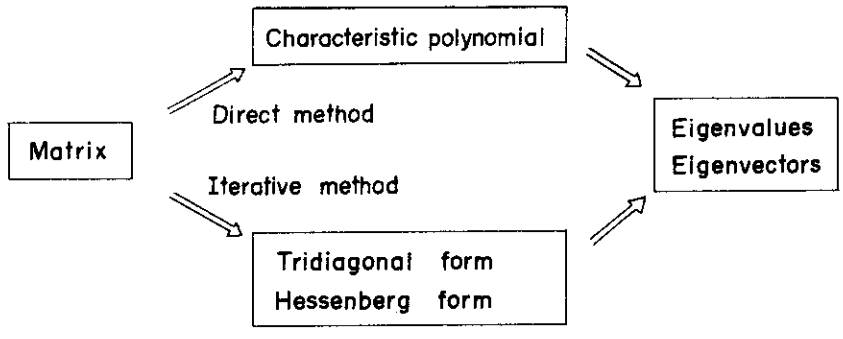

Fig. 1 Process of direct and iterative method

3.1 直接法

この節では複素数の系を考えよう。$\mathbf{A} = (a_{ij})$ (i , j = 1~n) とするとき，(1)式から$\mathbf{x}$ を消去した式は

$$\begin{vmatrix} \lambda - a_{11} & -a_{12} & \cdots\cdots & -a_{1n} \\ -a_{21} & \lambda - a_{22} & \cdots\cdots & -a_{2n} \\ \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots\cdots \\ -a_{n1} & -a_{n2} & \cdots\cdots & \lambda - a_{nn} \end{vmatrix} = 0 \tag{8}$$

となる。[14] 左辺は λ のn 次多項式，即ち$\mathbf{A}$ の固有多項式で

$$f(\lambda) = |\lambda \mathbf{I}_n - \mathbf{A}| = \lambda^n - c_{n-1}\lambda^{n-1} - \cdots\cdots - c_0 \tag{9}$$

と書ける。この式の係数c_i（$i = n-1 \sim 0$）は$\mathbf{A}$のすべての首座小行列式の和[3]であるが，直接この方法で求めようとすると多大な演算を必要とするであろう。

1949年に発表されたFrame 法[3)4)]は$\mathbf{A}$から逐次式により係数を計算するが，理論的にも面白く，よくとり上げられる。その計算手順（アルゴリズム）は

$$\begin{aligned} &\mathbf{A}_0 = \mathbf{I} \\ &c_{n-k} = \mathrm{tr}(\mathbf{A}\mathbf{A}_{k-1})/k \qquad (k = 1 \sim n-1) \\ &\mathbf{A}_k = \mathbf{A}\mathbf{A}_{k-1} - c_{n-k}\mathbf{I} \end{aligned} \tag{10}$$

である。このとき，

$$\begin{aligned} &|\mathbf{A}| = (-1)^{n-1} c_0 \\ &\mathbf{A}^{-1} = \mathbf{A}_{n-1}/c_0 \end{aligned} \tag{11}$$

であるから，$\mathbf{A}$の行列式と逆行列が求まる。また，$\lambda\mathbf{I} - \mathbf{A}$ の随伴行列を$\mathbf{J}(\lambda)$とすると，$\mathbf{A}$の固有値の1つ λ_j に対して

$$\mathbf{A}\mathbf{J}(\lambda_j) = \lambda_j \mathbf{J}(\lambda_j) \tag{12}$$

となるので，固有値 λ_jが求まれば固有ベクトル$\mathbf{x}_j$ も求められる。このとき特にλ_j が単根であれば$\mathbf{J}(\lambda_j)$の各列は比例するので

$$\mathbf{A}_n = \mathbf{A}\mathbf{A}_{n-1} - c_0\mathbf{I} = \mathbf{O} \tag{13}$$

とともに検算に役立つ。しかし，Frame 法はn^4回の乗除算を必要とし，桁落ちもひどく，また接近根の影響を受け易いので注意が必要である。

次に，これよりやや実用的な方法を述べる。

$$\mathbf{F} = \begin{pmatrix} c_{n-1}, & c_{n-2}, & \cdots\cdots\cdots, & c_0 \\ 1 & & O & \\ & 1 & & \\ O & & \ddots & \\ & & 1, & 0 \end{pmatrix} \tag{14}$$

の形の行列をFrobenius の行列（コンパニオン行列）[4)6)]というが，この固有多項式は

$$|\lambda\mathbf{I}-\mathbf{F}|=\begin{vmatrix}\lambda-c_{n-1} & -c_{n-2} & \cdots\cdots & c_0\\ -1 & \lambda & & \\ & -1 & \ddots & O\\ O & & \ddots & \lambda\\ & & -1 & \end{vmatrix} \tag{15}$$

で，元の行列$\mathbf{F}$が固有多項式の係数を陽に含んでいる。Danilevsky 法[4)〜6)]は行列をFrobenius 行列に変換する方法である。

$\mathbf{A}=(a_{ij})$に対し

$$\mathbf{S}_{n-1}=\begin{pmatrix}1 & & & O\\ & \ddots & & \\ O & 1 & & \\ -\dfrac{a_{n1}}{a_{n,n-1}} & \cdots\cdots & \dfrac{1}{a_{n,n-1}} & -\dfrac{a_{nn}}{a_{n,n-1}}\\ 0 & \cdots\cdots & 0 & 1\end{pmatrix} \tag{16}$$

とすれば，$\mathbf{AS}_{n-1}$の第n 行は（0，……，0，1，0）となる。但しこの行列は$\mathbf{A}$に相似ではないので，固有値計算のために

$$\mathbf{S}_{n-1}^{-1}=\begin{pmatrix}1 & & & O\\ & \ddots & & \\ O & & 1 & \\ a_{n1} & \cdots\cdots & a_{n,n-1} & a_{nn}\\ 0 & \cdots\cdots & 0 & 1\end{pmatrix} \tag{17}$$

を左から掛けると，変換後は

$$\mathbf{S}_{n-1}^{-1}\mathbf{AS}_{n-1}=\begin{pmatrix}f_{11} & \cdots\cdots & & f_{1n}\\ \vdots & & & \vdots\\ f_{n-1,1} & \cdots\cdots & f_{n-1,n-1} & f_{n-1,n}\\ 0 & \cdots\cdots & 1 & 0\end{pmatrix} \tag{18}$$

となる。この操作は$f_{n-1,\,n-2} \neq 0$であれば更に続行できる。退化したときは行と列を変換するが，それも不能のときは直和形のFrobenius の標準形に帰着する。この方法の乗除算は（n－1）（n^2+n-1）回[6]であり，軸選択も可能である。そのためには例えば$a_{n,\,n-1}$の代わりに$\max_{1\leq j\leq n-1} |a_{nj}|$なる$a_{nj}$を軸に選べばよい。このとき，$\lambda_j$に対応する固有ベクトルは$\mathbf{F}$の固有ベクトル$\mathbf{w}=(\lambda_j^{n-1}, \cdots\cdots, \lambda_j, 1)^t$を使って$\mathbf{x}_j = \mathbf{S}_{n-1} \cdots\cdots \mathbf{S}_1 \mathbf{w}$より求められる。

以上で2，3の直接法の固有多項式の係数の決め方および固有ベクトルの計算法を述べたが，固有方程式(8)式が解けて初めて固有値が求まり，固有ベクトルが計算できる。そのためまずNewton 法[3)7)16)]から述べよう。求める根をλ_0とし，そのk 回反復後の近似値をμ_iとすれば，λ_0は，

$$\mu_{k+1} = \mu_k - \frac{f(\mu_k)}{f'(\mu_k)} \tag{19}$$

の極限として得られることがFig. 2より分る。ここで，各$f(\mu_k)$, $f'(\mu_k)$は，Horner の方法[6)10)]により

$$\begin{aligned} q_0 &= r_0 \\ q_i &= q_{i-1}\mu_i - c_{n-i} \quad (i=1\sim n) \\ r_i &= r_{i-1}\mu_k + q_i \quad (i=1\sim n-1) \\ f(\mu_k) &= q_n\,,\ f(\mu_k) = r_{n-1} \end{aligned} \tag{20}$$

によって決められる。

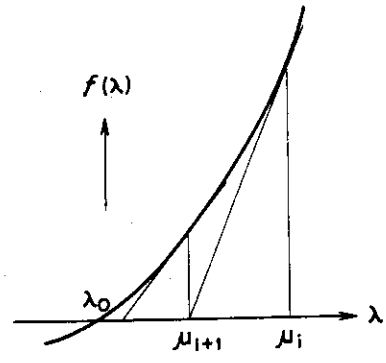

Fig. 2 Convergence by Newton's method

一方Bairstow 法[10)15)16)]は多項式を2次因子に分解する方法である。多項式の次数が奇数次のときは1根だけNewton 法で求める。この方法では2次因子$\lambda^2+d\lambda+e$はd_0, e_0を初期値と

する反復

$$d_{k+1} = d_k + \triangle d$$
$$e_{k+1} = e_k + \triangle e \qquad (21)$$

の極限として得られる。また，△d と△e は

$$q_{-1} = r_{-1} = 0, \quad q_0 = r_0 = 1$$
$$q_i = -q_{i-1} d_k - q_{i-2} e_k - c_{n-i} \quad (i = 1 \sim n)$$
$$r_i = -r_{i-1} d_k - r_{i-2} e_k + q_i \quad (i = 1 \sim n-1)$$

$$\triangle d = \frac{q_{n-1} r_{n-1} - q_n r_{n-3}}{r_{n-2}^2 - r_{n-3}(r_{n-1} - q_{n-1})} \qquad (22)$$

$$\triangle e = \frac{q_n r_{n-2} - q_{n-1}(r_{n-1} - q_{n-1})}{r_{n-2}^2 - r_{n-3}(r_{n-1} - q_{n-1})}$$

より決められる。2次因子が分かれば固有値は根の公式から容易に得られる。

一般に方程式 $f(\lambda) = 0$ を

$$\mu_{k+1} = F(\mu_k) \qquad (23)$$

の形の反復式で解くとき，根 $\lambda_0 = \lim_{k \to \infty} \mu_k$ に対して

$$\lim_{k \to \infty} (\mu_{k+1} - \lambda_0) / (\mu_k - \lambda_0)^h = s \neq 0 \qquad (24)$$

であれば，この収束はh位（h－th order）であるという。Newton 法は2位の収束で，Bairstow 法は2変数の Newton 法になっており，やはり2位の収束である。[16] 手元の文献ではこれらの著しい違いについての記述は見当らなかったが，(19)式にも見られるように，重根や近接根のときは適さないという共通の弱点がある。[16] そのため，これらの改良を中心とした多くの代数方程式の解法が発表されているが，本稿では省略する。しかし，このシリーズの文献16)ではこれらの数値解法プログラムの詳細が述べられ，ベンチマーク・テストが行われている。その結果，Newton 法の変形である Madsen のアルゴリズムによるもの[17]が最も標準的であることが示されている。

3.2 対角行列への変換

ここから3.5節までは固有多項式を経由しないで固有値問題を解く反復法について述べるが，特に，3.4節まではすべての固有値や固有ベクトルを求める完全問題を対象とする。

反復法のはしりは1846年のC. G. Jacobi による。[3] ここでは $\mathbf{A} = (a_{ij})$ を実対称行列としよう。いま a_{pq} が非対角要素の中で絶対値最大であるとすると（p，q）平面の回転

$$\mathbf{S}_{pq}(\theta) = \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & O & \\ & & 1 & & & & \\ & & & \cos\theta & & \sin\theta & \\ & O & & & 1 & & \\ & & & & & \ddots & \\ & & & -\sin\theta & & \cos\theta & \\ & & & & & & 1 \\ & & & & & & & \ddots & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ p) \\ \\ \\ q) \\ \\ \\ \end{matrix} \qquad (25)$$

による交換

$$\mathbf{A}_1 = \mathbf{S}^{t}_{pq}(\theta)\mathbf{A}\mathbf{S}_{pq}(\theta) = (a^{(1)}_{ij}) \qquad (26)$$

により$a^{(1)}_{pq}$を0にできる。次に残りの非対角要素の中から絶対値最大の要素を見つけそれを0にするように変換する。このくり返しにより行列$\mathbf{A}$は対角行列に収束する。[3)]

この方法は各変換が直交変換であるから要素の2乗和は不変であり，$L(\mathbf{A}) = \sum_{i \neq j} a_{ij}^2$ とすると

$$L(\mathbf{A}) + \sum_{i=1}^{n} a_{ii}^2 = L(\mathbf{A}_1) + \sum_{i=1}^{n} a_{ii}^{(1)2} \qquad (27)$$

が成り立つ。ここに

$$\begin{aligned} &a^{(1)2}_{pp} + a^{(1)2}_{qq} = a^2_{pq} + a^2_{qq} + 2a^2_{pq} \\ &a^{(1)}_{ii} = a_{ii} \quad (i \neq p, q) \\ &|a_{pq}| \geqq |a_{ij}| \quad (i \neq j) \\ &r = \frac{n^2 - n - 2}{n^2 - n} < 1 \end{aligned} \qquad (28)$$

であるから

$$L(\mathbf{A}_1) \leqq r\, L(\mathbf{A})$$

となり，この交換を続けると$\mathbf{A}_k$は対角行列に収束する。

いま$L(\mathbf{A}) = 1$に正規化されているとすれば

$$L(\mathbf{A}_k) < \varepsilon \qquad (29)$$

となるには

$$k \sim n^2 \log_{10} \frac{1}{\varepsilon} \qquad (30)$$

回程度の反復が必要となる。この方法は1回の変換には4n回の乗算が必要なので，例えば$\varepsilon = 10^{-h}$程度まで減少させるにはおよそ$4hn^3$回の乗算が必要な訳だが，実際にはもっと速く収束するだろう。[3)]

以上の方法は古典的なJacobi 法と呼ばれているが，毎回最大要素を捜すのはかなり手間を要

すのでa_{12}，a_{13}，……というように連続して要素を選ぶ連続（serial）Jacobi 法，また閾値列$\varepsilon_1 > \varepsilon_2 > \cdots\cdots$ を定め，ε_1 より大きい要素のみをまず変換していき，次にε_2に移るというような閾（threshold）Jacobi 法などの改良がある。[3]また，角θは$\tan 2\theta = 2a_{pq}/(a_{qq}-a_{pp})$となるように決められるが，この計算で平方根の計算を省略したりすることもでき，[3]これらの詳細な誤差解析もなされている。[7]なお，Jacobi 法の固有ベクトルは，連続Jacobi 法を例にとると，$N=n(n-1)/2$回ずつF 周変換したあとでは，全変換行列を変換順に掛けた積

$$\mathbf{X} = \prod_{f=1}^{F} \left(\prod_{p,q} \mathbf{S}_{pq}^{(f)}(\theta) \right) \tag{31}$$

から得られる。

Jacobi 法は，いわば実対称行列$\mathbf{A}$に対し，直交行列$\mathbf{X}$を見つけ出して

$$\mathbf{X}^t \mathbf{A}\mathbf{X} = \Lambda = \mathrm{diag}(\lambda_1, \cdots\cdots, \lambda_n) \tag{32}$$

と対角行列に変換する方法であった。[4]このとき，$\mathbf{X}^t\mathbf{X}=\mathbf{I}$ であるので$\mathbf{A}\mathbf{X}=\mathbf{X}\Lambda$となり，(31)式の$\mathbf{X}$が必然的に固有ベクトルを表わす行列となる訳である。[15] $\mathbf{A}$がHermite のときは，コニタリ行列$\mathbf{X}$を見い出して

$$\mathbf{X}^t \mathbf{A}\mathbf{X} = \Lambda \tag{33}$$

とできる。[15] Greenstadt 法はこの原理に基づいているが，一般的な議論は後述のQR 法の説明のときに詳しくなされる。

このほか回転を用いる変換は種々な方面に応用される。Eberlein 法[8)18)]は回転$\mathbf{R}=(r_{ij})$とずれ$\mathbf{S}=(s_{ij})$で実行列$\mathbf{A}$を

$$\mathbf{A}_1 = (\mathbf{RS})^t \mathbf{A}(\mathbf{RS})$$

と変換し，各列のノルムを減少させて究極的に正規，区画化対角行列にする方法であり，その複素版もある。[19] 各行列はそれぞれ

$$\begin{aligned} &r_{ii} = r_{jj} = \cos\theta, \quad r_{ij} = -r_{ji} = -\sin\theta \\ &s_{ii} = s_{jj} = \cosh\varphi, \quad s_{ij} = s_{ji} = -\sinh\varphi \end{aligned} \tag{34}$$

で定義され，その他の要素は$r_{ij}=s_{ij}=\delta_{ij}$である。

3.3 3重対角行列への変換

同じ反復法でも以下に述べる方法は次の点でJacobi 法と違う。

① 固有値が即座に得られる対角行列でなく，一旦3重対角行列に変換する。

② 3重対角行列の固有値は別の方法で解く。

この方法が考案された動機は，

① 消去法的な手法が利用でき，比較的少ない一定の演算で3重対角形にできる。

② 3重対角行列は形が簡単なため演算時間が短かく，かつ特別な解法が使える。

などであったが特に固有値を求める後半のステップで様々な技法も使えるため，固有値計算ルー

チンとしても，アルゴリズムが簡単で良く使われたJacobi 法にとって代わることになる。[5]

まず，最初に登場したGivens 法[4]について述べよう。これは連続Jacobi法において，実対称行列$\mathbf{A}$の非3重対角部分を0にする方法である。具体的には，$\omega=\sqrt{a_{i-1,i}^2+a_{i-1,j}^2}$とするとき，

$$\begin{aligned}&\omega=0 \text{ ならば} \cos\theta=1,\ \sin\theta=0\\&\omega\neq 0 \text{ ならば} \cos\theta=a_{i-1,i}/\omega,\ \sin\theta=-a_{i-1,j}/\omega\\&\quad (j=i+1\sim n,\ i=2\sim n-1)\end{aligned}\tag{35}$$

によって回転し，$(i-1, j)$要素を0にする。この変換は有限回の$(n-1)(n-2)/2$回で完了する。

その後J.H. Wilkinson はHouseholder 法[4)20]により$(n-2)$回の直交変換で3重対角形にする方法を示した。$\mathbf{A}_1=(a_{ij}^{(1)})=\mathbf{A}$から出発し，$\omega=\sum_{i=j+1}^{n} a_{ij}^{(j)^2}$とするとき

$$\begin{aligned}&\omega_j=0 \quad \text{ならば} \quad \mathbf{T}_j=\mathbf{I}\\&\omega_j\neq 0 \quad \text{ならば} \quad h_j=\omega_j\pm a_{j+1,j}^{(j)}\sqrt{\omega_j}\\&\quad \mathbf{u}_j^t=(0,\cdots\cdots,\ 0,\ a_{j+1,j}^{(j)}\pm\sqrt{\omega_j},\ a_{j+2,j}^{(j)},\ a_{nj}^{(j)})\\&\quad \mathbf{T}_j=\mathbf{I}-\mathbf{u}_j\mathbf{u}_j^t/h_j\end{aligned}\tag{36}$$

によって$\mathbf{T}_j$を定義すれば$\mathbf{A}$が実対称のとき$\mathbf{T}_j$も実対称で直交行列となり，変換$\mathbf{A}_{j+1}=\mathbf{T}_j\mathbf{A}_j\mathbf{T}_j$により$\mathbf{A}$は実対称3重対角行列$\mathbf{A}_{j+1}$に移る。これは一度に1つの行を変換する方法（鏡像変換）であり，Givens 法がいわば各行一つ一つの要素を変換するのに対して行の変換が予想以上に簡単に行えるのが特徴であろう。この変換も平方根の計算等を含んでいるのでJacobi 法と同様に改良できる。[5]

行列が実対称帯行列のときのHouseholder 法は，各1回の変換後の行列がやはり同じ幅の帯行列となるので計算も一層簡単になる。[21] 一般の複素行列にこの方法を適用した場合はHessenberg 行列となる[4]が，これについては次節で論じられる。

次に，ある実非対称3重対角行列を対称3重対角行列に変換する方法を述べよう。いま3重対角行列$\mathbf{A}$が

$$a_{i,i-1}\cdot a_{i-1,i}\geq 0 \qquad (i=2\sim n)\tag{37}$$

をみたすとき，対応する副対角要素の積が非負となるという意味で「性質T_nを持つ」と呼ぶことにしよう。同様に，性質T_nを持ち，

$$a_{i,i-1}\cdot a_{i-1,i}=0 \quad \text{ならば} \quad a_{i,i-1}=a_{i-1,i}=0\tag{38}$$

のとき，「性質T_sを持つ」と呼ぶ。特に後者は，実質的に実3重対角と同じ方法で固有値と固有ベクトルが計算されるので擬対称（quasi－symmetric）3重対角行列と呼ばれる。[7]

性質T_nをもつとき，対角行列$\mathbf{D}=(d_{ii})$を

$$d_{11}=1$$

$$a_{i,i-1} \text{ または } a_{i-1,i} = 0 \text{ ならば } d_{ii} = 1 \quad (i = 2 \sim n) \tag{39}$$

$$a_{i,i-1} \cdot a_{i-1,i} > 0 \text{ ならば } d_{ii} = \sqrt{\prod_{j=2}^{i} a_{i,i-1} / \prod_{j=2}^{i} a_{i-1,i}}$$

で定義すれば，$\mathbf{A}$ は

$$\mathbf{A}_1 = \mathbf{D}^{-1}\mathbf{A}\mathbf{D} \tag{40}$$

より対称3重対角行列に変換される（diagonal similarity transformation）[7)13)]。しかし，$\mathbf{A}$ が性質T_s を持たないときはこの変換を利用して固有ベクトルを求める訳にいかないので注意を要する。

以上のほか，実非対称行列を実非対称3重対角行列に変換する方法[22)]等もあるが，上に述べたように3重対角の利点が生かしにくいので余り実用化されていない。

実対称3重対角行列に変換された後の固有値の計算法は多くが行列の3角分解法（triangular decomposition）[4)6)]に基づいている。そして，この部分はFig. 1 に示されるように，反復法の中心をなすものであり，解の精度の向上や収束の加速など解法上の重要な技法の大部分がここで用いられる。

連立一次方程式の代表的な数値解法にCrout 法[4)6)]というのがあった。それは与えられた実行列$\mathbf{A}$ を対角要素を1とする左下3角行列$\mathbf{L}$ と右上3角行列$\mathbf{R}$ との積に分解して解く方法であり，この分解は一通り（unique）である。いま，行列$\mathbf{A}$ に対して分解と逆順の積（合成）を考えることにより列

$$\begin{aligned}
\mathbf{A}_0 &= \mathbf{A} = \mathbf{L}_1\mathbf{R}_1 \\
\mathbf{A}_1 &= \mathbf{R}_1\mathbf{L}_1 = \mathbf{L}_2\mathbf{R}_2 \\
&\cdots\cdots\cdots\cdots \\
\mathbf{A}_k &= \mathbf{R}_k\mathbf{L}_k = \mathbf{L}_{k+1}\mathbf{R}_{k+1}
\end{aligned} \tag{41}$$

を定義すれば，

$$\mathbf{A}_k = \mathbf{L}_k^{-1}\mathbf{A}_{k-1}\mathbf{L}_k \tag{42}$$

となるから固有値は不変である。また，固有値が

$$|\lambda_1| < |\lambda_2| < \cdots\cdots < |\lambda_n| \tag{43}$$

をみたせばA_kは右上3角行列に収束し，その対角要素が固有値である[5)]。

LR法は1958年 H. Rutishauser により発表された。これが帯行列に用いられると(41)式の分解も合成も帯行列となり有効であるが，非対称行列にそのまま用いると演算回数も多く不安定である[23)24)]。このため変形（modified）LR 法[24)]では部分軸選択により安定化を図っている。一方，元の行列が対称である場合は，より簡単で安定したCholesky 分解[4)6)] $\mathbf{A} = \mathbf{R}^t\mathbf{R}$ に変わる。LR法に似た解法にユニタリ行列を用いて分解する方法がある。以下，現在最も多く使われるこれらの方法について詳しく述べよう。

QR法[5)23)25)]の特徴については，単にアルゴリズムを理解しただけでは不充分で，各種の補助

的手法をうまくプログラミングしたときに初めてその真価を発揮することは既に述べた。いま$\mathbf{A}$を複素行列とする。Gram－Schmidt の直交化（ orthogonalization ）により$\mathbf{A}$はユニタリ行列$\mathbf{Q}$と対角要素がすべて実数で非負の上３角行列$\mathbf{R}$に分解できるので，ＬＲ法と同様な例

$$\begin{aligned}\mathbf{A}_0 &= \mathbf{A} = \mathbf{Q}_1\mathbf{R}_1\\ \mathbf{A}_1 &= \mathbf{R}_1\mathbf{Q}_1 = \mathbf{Q}_2\mathbf{R}_2\\ &\cdots\cdots\\ \mathbf{A}_k &= \mathbf{R}_k\mathbf{Q}_k = \mathbf{Q}_{k+1}\mathbf{R}_{k+1}\end{aligned} \tag{44}$$

を考えることができる。このとき$\mathbf{A}_k$は「本質的に」右上３角行列に収束する。この意味は次のようなものである。いま$\mathbf{A}$の固有値λ_i（ $i=1\sim n$ ）を絶対値の等しいもの同志で組分けし，小さい順にp_r（ $\sum_r p_r = n$ ）個ずつあるとする。このとき$\mathbf{A}_k$は対角上にp_r個の固有値を内蔵したp_r次の区画（ block ）を，左下部分に０の区画を，そして右上部分に一定範囲で振動する非零の区画を持つようになる。このような事情は(43)式が成り立たない場合のＬＲ法にもあてはまる。

ＱＲ法においてもそうであるが，固有値問題を反復法で解くとき，その収束の速さは$|\lambda_i| < |\lambda_j|$に対して$(\lambda_i/\lambda_j)^k \to 0$の速さが支配的な意味をもつことが多い。一般には大きい２つの固有値が$|\lambda_{n-1}| < |\lambda_n|$をみたし，かつ固有値$\lambda$のｋ回反復後の近似値$\lambda^{(k)}$に対して

$$\lambda^{(k)} = \lambda + O\left[\left(\lambda_{n-1}/\lambda_n\right)^{kh}\right] \tag{45}$$

をみたすとき，ｈ次の収束（ h－th order convergence ）であるという。例えば最小固有値が$\lambda_1 = \lambda_2$と重複している場合，古典的 Jacobi 法が２次収束[26]であるのに対し，ＱＲ法は３次収束であり，[28]この方法の収束が速いことが理論的にも示される。

ＱＲ法に似た解法にＱＬ法[28]がある。これは行列を上３角行列に変換する代わりに下３角行列に変換する方法である。ＱＲ法における$\mathbf{A}_k = (a_{ij}^{(k)})$の収束をみると，まず行列の左下部分が小さくなり，それが次第に右下部分に移っていく。それ故，最も小さい固有値λ_1の近似値として$\tilde{\lambda}_1 = a_{nn}^{(k)}$とするのは自然であるが，絶対値の近い固有値がある場合を考えると，２次小行列

$$\begin{pmatrix} a_{n-1,n-1}^{(k)} & a_{n-1,n}^{(k)} \\ a_{n,n-1}^{(k)} & a_{nn}^{(k)} \end{pmatrix}$$

の固有値のうち絶対値の小さい方をとるのもよく[28]，実際にもこの方が多く用いられている。従ってＱＲ法では固有値が対角上に，降下順に並び，右下の小さい値から求まってゆくが，ＱＬ法では上昇順に並び，右下の大きい値から求まることになる。その他機構の対称性によるものはＱＲ法から容易に推論できよう。

詳しい議論は後に譲るが，ＱＲ法がＬＲ法より優れる理由として次のことがあげられる。

① ユニタリ変換は数値的に安定である。

② 実行列のときも複素行列とみなしてＱＲ法を用いると，収束の加速が容易である。

③ 実際の計算は実数の範囲で効率よく行える。

ＱＲ法の収束が$(\lambda_i/\lambda_j)^k \to 0$に支配されていることは既に述べた。いま，最小の固有値λ_1の良い近似値$\tilde{\lambda}_1$が求まった段階で$\tilde{\mathbf{A}} = \mathbf{A} - \tilde{\lambda}_1\mathbf{I}$を考えると，この固有値は$\mu_i = \lambda_i - \tilde{\lambda}_1$（ i

＝1～n ）となるので，$\tilde{\mathbf{A}}$にQR法を適用すると収束は速くなる。この技法は原点移動[4)5)28)]と呼ばれ，LR法やQL法など他の解法にもしばしば用いられる。例えばQL法に用いた場合は

$$\mathbf{A}_k - \tilde{\lambda}_n \mathbf{I} = \mathbf{Q}_k \mathbf{L}_k$$
$$\mathbf{A}_{k+1} = \mathbf{L}_k \mathbf{Q}_k \tag{46}$$

であり，$\tilde{\lambda}_n$ は最大固有値 λ_n の近似値である。また，これに対し

$$\mathbf{A}_k - \lambda_n \mathbf{I} = \mathbf{Q}_k \mathbf{L}_k$$
$$\mathbf{A}_{k+1} = \mathbf{L}_k \mathbf{Q}_k + \tilde{\lambda}_n \mathbf{I} \tag{47}$$

によっても反復できるので，特に後者を陰的（implicit）QL法という[27)29)]。一般に陰的QL法は加速の効果はふつうのQL法ほど顕著でないが，

① 行列の要素の行和が大きく変化する。

② 1行からn行にかけて要素が単調に大きくならない。

場合でも安定であるので，標準的なアルゴリズムとして採用される[28)]。

QR法は(44)式の通りに計算するとQR分解だけでn^3回の乗算および同数の加算を必要とし，誤差も入り易い。いま視点を変えて

$$\mathbf{R}_k \mathbf{Q}_k = \mathbf{Q}_k^{-1} \mathbf{A}_{k-1} \mathbf{Q}_k \tag{48}$$

としてみると，QR法は左から掛けて$\mathbf{A}_{k-1}$ を上3角にするユニタリ行列が求まればよい訳で，それは先に述べた回転変換または（複素数の場合の）鏡像変換の積によってみたされる。即ち，$\mathbf{R}_k$ はその都度別個に作る必要はなく，一つ一つのユニタリ行列を作る過程で簡単に作られれば計算は一層簡潔となる[29)]。QR法の固有ベクトルはJacobi 法やGreenstadt 法と同様な推論から

$$\mathbf{X}_k = \prod_{r=1}^{k} \mathbf{Q}_r \tag{49}$$

より得られるが，$\mathbf{X}_k^* \mathbf{A}_k \mathbf{X}_k$ 自身が対角行列に収束するとは限らないので，区画化対角行列に収束するような場合は，その区画ごとの変換をこの$\mathbf{X}_k$ に掛けることにより，反映させなければならない。

3.4 Hessenberg 行列への変換

前節ではQR法を中心とする3角分解に基づく諸方法の一般的な姿について述べた。しかし，実際には対称3重対角行列の完全問題という特殊な状況を生かし，それなりの簡単化，技法の導入，特徴の生かし方ができた訳である。逆に，一般の行列にこれらを用いる場合，計算量，数値的安定性等を違った立場で配慮せねばならず，それがこの節の主題となる。

簡単のため，まず行列$\mathbf{A}=(a_{ij})$ が実非対称行列であると仮定しよう。このときのみ有効な変換法としてGauss の消去法に基づいた方法[30)31)]があるが，基本操作のために固有値が変らないように注意しなければならない。その手順は以下の操作に従う。

① a_{21}，……，a_{n1} のうちa_{i1} が絶対値最大であれば，2行とi 行，2列とi 列を入れ替える。

② 第2行を使って第j 行を消去し，次に第j 列を使って第2列を消去する。以上をj ＝3～

n について行うと，a_{31}，……，a_{n1} が0になる。

③ $a_{i,i-2}$，……，$a_{n,i-2}$（$i=4\sim n$）を0にするために上の①，②を行う。

この操作により$\mathbf{A}$がHessenberg 行列に変換されるが，これは軸選択を行なっているので誤差も少なく，安定化された基本変換（stabilized elementary transformation）とも呼ばれる。途中で軸$a_{i,i-1}$が0になれば，行列は低次のHessenberg 行列の直和に分解される。

この方法は本質的にはLR法と同等であるが，連立一次方程式を解くときのGauss の消去法およびCrout法の関係が，そのまま成り立つ。即ち，前者が度重なる消去で新しい要素を決定するのに比し，後者は積和を用いて一度で計算するため，記憶容量の節約にも役立つ。[5] この類似性は掃き出し（sweep out）法についても同様である。[10]

直交相似変換（orthogonal similarity transformation）による方法のアルゴリズムは Householder 法と全く同じであり，[4] $\mathbf{A}$が実非対称でも(36)式の$\mathbf{T}_j$ は対称となり，変換された行列は実Hessenberg 行列となる。この方法は，幾種かの問題についてはガウスの消去法より高精度であるが，計算速度が遅いため標準的な解法としては消去法が採用される。[13]

行列の要素の大きさにかなりの幅があるとき，例えば，直交変換を行うには，大きな要素を予じめ左上に置くことが望ましい。[13] この概念を一般化したのが規格化（balancing）である。一般に，固有値計算の誤差は，計算機で $1+\varepsilon=1$ となる最大の数（relative machine precision）をεとするとき，およそ$\varepsilon\|\mathbf{A}\|_1$ であるといわれている。ここに，$\|\mathbf{A}\|_1$は$\mathbf{A}$のL_1－ノルムである。規格化は固有値を変えないで，$\|\mathbf{A}\|_1$を小さくすること，換言すれば行列の対応する行と列の絶対和を等しくすることを目的としている。以下その手順を示しておこう。[11)32)]

① $\mathbf{A}_0=\mathbf{A}$，$\mathbf{A}_{10}=\mathbf{A}_0$，$\mathbf{A}_{1i}$（$i=1\sim n$），$\mathbf{A}_1=\mathbf{A}_{1n}$，…… の順で$\mathbf{A}_{k,i-1}=\mathbf{G}=(g_{ij})$ まで求まったとする。

② $\mathbf{G}$の第i 行と第i 列に注目し，

$$\mathbf{R}=\sum_{\substack{j=1\\ j\neq i}}^{n}|g_{ij}|\ ,\quad \mathbf{L}=\sum_{\substack{j=1\\ j\neq i}}^{n}|g_{ji}| \tag{50}$$

を計算する。

③ $RL=0$ のとき，g_{ii} は固有値であり，第1行，第1列と交換する。（permutation similarity transformation）。

④ 2進の計算機では，$p=2$ とし，

$$p^{2h-1}<R/L\leq p^{2h+1} \tag{51}$$

をみたす整数h を求め，$f=p^h$ とする。

⑤ 少しでも$\|\mathbf{G}\|_1$を減らしたいときは$r=1$，大きく減るときのみ規格化を行いたいときは $r<1$ とし，

$$Lf+R/f<r(L+R) \tag{52}$$

をみたせば$d_i^{(k)}=f$ とする。(51)式をみたさないときは変換を行わない。

⑥ i 番目以外の要素が1である対角行列 $\mathbf{D}_{ki}=\mathrm{diag}(1,\cdots\cdots,1,\underset{i}{d_i^{(k)}},1,\cdots\cdots,1)$

により$\mathbf{A}_{ki}=\mathbf{D}_{ki}^{-1}\mathbf{A}_{k,i-1}\mathbf{D}_{ki}$と変換する（exact similarity transformation）。

⑦ どの行と列についても変換を行わなかったとき，$\mathbf{A}_k=\mathbf{A}_{k-1}$として規格化の反復を終了する。

次に一般の複素行列をHessenberg行列に変換する場合を考えてみよう。この場合も本質的には実行列のときと同じ原理で実行できる。Gaussの消去法は既に述べた計算手順①～③において，要素a_{ij}が複素数と思えばそのまま使える。Householder法に因んだ方法では(36)式の直交行列$\mathbf{T}$がユニタリ行列に変わるだけである（unitary similarity transformation）。実際には$\omega_j=\sum_{i=j+1}^{n}|a_{ij}^{(j)}|^2$とするとき，

$$\begin{aligned}
&\omega_j=0 \quad \text{ならば } \mathbf{T}_j=\mathbf{I}\\
&\omega_j\neq 0 \quad \text{ならば } h_j=\omega_j\pm \mathrm{Re}(a_{j+1,j}^{(j)})\sqrt{\omega_j},\\
&\mathbf{u}_j^t=(0,\cdots\cdots,0,a_{j+1,j}^{(j)}\pm\sqrt{\omega_j},a_{j+2,j}^{(j)},\cdots\cdots,a_{nj}^{(j)}),\\
&\mathbf{T}_j=\mathbf{I}-\mathbf{u}_j\mathbf{u}_j^{*}/h_j
\end{aligned}\tag{53}$$

による。しかし，この方法は実非対称の場合の直交変換に対応しているので，一般にはGaussの消去法が望ましいとされている。[31] なお，規格化などの技法は複素行列に対してもそのまま使えることは容易に推察できよう。[13)32)]

Hessenberg行列の固有値はQR法で求められることが多い。それは先に述べた数値的安定性のほかに次のような利点があるからである。[5)]

① Hessenberg行列のユニタリ変換はHessenberg行列である。

② この変換のためのユニタリ行列は簡単に求められ，その個数も少なくて済む。

③ 数回先の反復行列を直接作ることができる。

これらの詳しい議論は省略するとして，次に実行列に対して複素計算を避ける方法を示しておこう。元の行列が実行列でも，ふつうのQR法では$\mathbf{Q}$も$\mathbf{R}$も複素行列になる。しかし，固有値は少なくとも共役の組になるので，例えば最小の固有値λ_1が複素数でその近似値をμ_1とすれば$\bar{\mu}_1$は$\bar{\lambda}_1$の近似値である。従ってアルゴリズム

$$\begin{aligned}
&\mathbf{F}_k=\mathbf{A}_k-\mu_1\mathbf{I}=\mathbf{Q}_k\mathbf{R}_k\\
&\mathbf{F}_{k+1}=\mathbf{R}_k\mathbf{Q}_k,\ \mathbf{A}_{k+1}=\mathbf{F}_{k+1}+\mu_1\mathbf{I},\\
&\mathbf{G}_{k+1}=\mathbf{A}_{k+1}-\bar{\mu}_1\mathbf{I}=\mathbf{Q}_{k+1}\mathbf{R}_{k+1},\\
&\mathbf{F}_{k+2}=\mathbf{R}_{k+1}\mathbf{Q}_{k+1},\ \mathbf{A}_{k+2}=\mathbf{F}_{k+2}+\bar{\mu}_1\mathbf{I}
\end{aligned}\tag{54}$$

を想定すると，$\mathbf{A}_k$が実行列ならば$\mathbf{A}_{k+2}$も実行列となり，

$$\begin{aligned}
&\mathbf{A}_{k+2}=(\mathbf{Q}_k\mathbf{Q}_{k+1})^{-1}\mathbf{A}_k(\mathbf{Q}_k\mathbf{Q}_{k+1})\\
&(\mathbf{A}_k-\mu_1\mathbf{I})(\mathbf{A}_k-\bar{\mu}_1\mathbf{I})=(\mathbf{Q}_k\mathbf{Q}_{k+1})(\mathbf{R}_{k+1}\mathbf{R}_k)
\end{aligned}\tag{55}$$

の関係式を得る。従って，

① $\mathbf{A}_k$を実行列とし$(\mathbf{A}_k-\mu_1\mathbf{I})(\mathbf{A}_k-\bar{\mu}_1\mathbf{I})$を計算する。

② これを$\mathbf{S}=\mathbf{Q}_k\mathbf{Q}_{k+1}$，$\mathbf{T}=\mathbf{R}_{k+1}\mathbf{R}_k$に分解する。

③ $\mathbf{A}_{k+2}=\mathbf{S}^{-1}\mathbf{A}_k\mathbf{S}$ を計算する。

という手順に従えばよく，これを2重（double）QR法という。[5] しかし，$\mathbf{A}_k$がHessenberg行列では(55)式の$\mathbf{A}_k^2$の計算が大変なので，工夫もされている。[5]

QR法の固有ベクトルの計算法は(49)式で既に述べたが，元の行列がHessenberg形（Hessenberg form）にされるとき別の手段が採られたときは，そのときの変換の蓄積（accumulation）が更にこのHessenberg行列の固有ベクトルに掛けられなければならない。QR法に限らず，一般に変換が相似変換（similarity transformation）$\mathbf{T}^{-1}\mathbf{A}\mathbf{T}$ の規則正しい繰り返しで行われる場合，固有ベクトルは常にその積で表わせるので，ベクトルとしての規格化（normalization）以外特に留意する事項はない。

3.5 特殊問題の解法

与えられた行列のすべての固有値や固有ベクトルを求めるのではなく，必要なものだけを計算する場合はそれに応じた独特の解法が用いられる。これら特殊問題に適する解法の殆んどは理論上完全問題を扱うことができるようになっているが，今迄に述べてきた解法に比べ様々な点で不利であることは否めない。本稿で扱われる特殊問題の主なものは

① 絶対値の大きい，または小さい2，3の固有値や固有ベクトルを求める。（これを端（extreme）の固有値を求めるという意味で「E－型」と呼ぶことにし，以下同様とする。）

② 与えられた値λ_0に近い順に，数個の固有値や固有ベクトルを求める。（N－型）

③ ある区間を与えて，それに属する固有値や固有ベクトルを求める。（B－型）

④ 固有値の大小順に従い，j_1番目からj_2番目までの固有値や固有ベクトルを求める。（I－型）

⑤ 固有値が分っているとき，その中から適当に選んでそれに対応する固有ベクトルを求める。（S－型）

などである。特に①～④は系としても固有値が実数となって大小関係が明確となるHermiteまたは実対称行列に限られる。また，固有値のみ必要という場合は実際にも度々起きるが，固有値は求めず固有ベクトルのみという場合は殆んどなく，ここでも除外される。なお，すべての固有値のみを求める問題は完全問題として扱われる。以下項目の順に沿ってその解法を概観しよう。

$\mathbf{A}$が実非対称行列で，その固有値が$|\lambda_1|\leqq\cdots\cdots\leqq|\lambda_{n-1}|<|\lambda_n|$であり，$\lambda_j$に対応する固有ベクトル$\mathbf{x}_j$はともに長さが1であるとする。任意のベクトル$\mathbf{z}^{(0)}=(z_1^{(0)},\cdots\cdots,z_n^{(0)})^t$は$\mathbf{x}_j$の一次結合で表わせるが，$\mathbf{z}^{(0)}=\sum_{j=1}^{n}\rho_j\mathbf{x}_j$ において$\rho_n\neq 0$となっているとしよう。ベクトルとスカラーの反復

$$\begin{aligned}\psi_k &= \max_j |z_j^{(k)}|\\ \mathbf{w}^{(k)} &= \frac{1}{\psi_k}\mathbf{z}^{(k)}\\ \mathbf{z}^{(k+1)} &= \mathbf{A}\mathbf{w}^{(k)}\end{aligned} \qquad (56)$$

を考えると

$$\mathbf{z}^{(k)} = \sum_{j=1}^{n} \rho_j \lambda^k \mathbf{x}_j \Big/ \prod_{r=1}^{k-1} \psi_r = \psi_k \mathbf{w}^{(k)}$$
$$= \frac{\rho_n}{\prod_{r=1}^{k-1} \psi_r} \frac{\lambda_n^k}{} \left\{ \mathbf{x}_n + \sum_{j=1}^{n-1} \frac{\rho_j}{\rho_n} \left(\frac{\lambda_j}{\lambda_n} \right)^k \mathbf{x}_j \right\} \tag{57}$$

となる。仮定からkが十分大きければ，$\mathbf{z}^{(k)}$は$\mathbf{x}_n$とほぼ比例し，その絶対値最大の場所は一致する。一方，$\mathbf{w}^{(k-1)} = \mathbf{z}^{(k-1)} / \psi_{k-1}$も$\mathbf{w}^{(k)} = \mathbf{z}^{(k)} / \psi_k$も$\mathbf{x}_n$とほぼ比例し，絶対値最大の要素が1であるから$\lambda_n / \psi_k$はほぼ1である。即ち

$$\psi_k \to \lambda_n \ , \ \mathbf{w}^{(k)} \to \mathbf{x}_n \quad (k \to \infty) \tag{58}$$

である。このような方法は，行列$\mathbf{A}$のべき乗を用いるのでべき乗法[3)6)8)]という。この方法で，大きい方から数えて2番目の固有値と固有ベクトルを求めるとしよう。$|\lambda_{n-2}| < |\lambda_{n-1}|$として$\rho_n = 0$で始めると，理論上，

$$\psi_k \to \lambda_{n-1} \ , \ \mathbf{w}^{(k)} \to \mathbf{x}_{n-1} \quad (k \to \infty) \tag{59}$$

となる筈であるが，実際は丸めの誤差が$\mathbf{x}_n$成分として現われ，それが増幅して$\psi_k \to \lambda_n$，$\mathbf{w}^{(k)} \to \mathbf{x}_n$となるので注意を要する。べき乗法では$|\psi_k / \psi_n|$，$|\lambda_j / \lambda_n|$が小さい程速く収束する。前者は初期ベクトル$\mathbf{z}^{(0)}$のとり方により，$\mathbf{z}^{(0)}$が$\mathbf{x}_n$に近い程よい。しかし後者は行列$\mathbf{A}$の性質に依存するので，これが1に近いときは他の方法で求めた方がよかろう。この方法の収束性は，厳密には求める固有値がJordanの標準形[7)]でどのような形をとるかに密接に関連している。一般にはひとまず2次因子

$$(\lambda - \lambda_{n-1})(\lambda - \lambda_n) = \lambda^2 + p\lambda + q \tag{60}$$

を求めることになるので，これを解決することにより次のステップに進める。べき乗法で最小の固有値を求めたいとき，$\mathbf{A}^{-1}$が容易に得られれば，この最大固有値として求めればよい。

E－型の問題を解く方法をもう1つあげる。ここでは元の行列が実対称あるいはHermite で，それが既に実対称3重対角行列

$$\mathbf{A}_1 = \begin{pmatrix} a_{11} & a_{12} & & & O \\ a_{21} & a_{22} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & a_{n-1,n} \\ O & & & a_{n,n-1} & a_{nn} \end{pmatrix} \tag{61}$$

に変換されているとしよう。これをQR法で解くことは

$$\mathbf{A}_2 = \begin{pmatrix} a_{11} & a_{12}^2 & & & O \\ 1 & a_{22} & a_{23}^2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & a_{n-1,n}^2 \\ O & & & 1 & a_{nn} \end{pmatrix} \tag{62}$$

をLR法で解くことに相当する。[33)34)] 簡単のため，原点移動が行われない場合を考えてみよう。$\mathbf{A}_2$ を分解して

$$\mathbf{A}_2 = \mathbf{L}_2\mathbf{R}_2 = \begin{pmatrix} \ell_1^{(2)} & & & O \\ 1 & \ddots & & \\ & \ddots & \ddots & \\ O & & 1 & \ell_n^{(2)} \end{pmatrix} \begin{pmatrix} 1 & r_1^{(2)} & & O \\ & \ddots & \ddots & \\ & & \ddots & r_{n-1}^{(2)} \\ O & & & 1 \end{pmatrix} \tag{63}$$

となったとする。このとき，$r_0^{(k)}$，$r_n^{(k)}$ は常に0としてi ＝1～n の順に

$$\begin{aligned} \ell_i^{(k+2)} &= \ell_i^{(k)} + (r_i^{(k)} - r_{i-1}^{(k+1)}) \\ r_i^{(k+1)} &= r_i^{(k)} \cdot (\ell_i^{(k+1)} / \ell_{i+1}^{(k)}) \end{aligned} \tag{64}$$

と計算していくと，根の絶対値がすべて異なるとき，$\ell_i^{(k)}$ は大きい方からi 番目の固有値に，$r_i^{(k)}$ は0に収束する。また絶対値が等しい固有値がある場合は前と同様に2次因子の係数が求まる。このアルゴリズムは(64)式にも現われているように商と差を利用するので特に商差（quotient difference，QD）法[5)]というが，QR法の一つとして有理（rational）QR法と呼ばれることもある[33)34)]。この方法で実非対称行列$\mathbf{A}_1$が性質T_nをもつ場合を扱うときは$\mathbf{A}_2$において$a_{i,i+1}^2$を$a_{i,i+1} \cdot a_{i+1,i}$に置き換えればよい。また，原点移動もそう困難なく導入される。[34)]

固有値計算を簡単化する有力な技法に次数の切り下げ（deflation）[5)7)]がある。これはLR法やQR法のように行列を変換していく過程において，最初に最小の固有値λ_1が（n，n）要素として求まり，第n 行の残りの要素が0に収束する場合，この行列から第n 行と第n 列を除いた（n－1）次の行列に対して残りの（n－1）個の固有値を求める方法である。これはべき乗法などにも使えるが，QR法では残りの固有値も殆んど収束しかけた状態から出発することになるので後の計算も簡単であり，特に対称正定値のときこの傾向が強い。[5)] 例えばN－型の問題を解く場合を考えてみる。目標値λ_0により$\mathbf{A}$の固有値λ_j（j ＝1～n ）は原点移動$\mathbf{A}_0 = \mathbf{A} - \lambda_0\mathbf{I}$により$\lambda_j - \lambda_0$に移るので，この問題は$\mathbf{A}_0$のF－型の問題として原点移動と次数の切り下げを繰り返して簡単に求めることができる。[35)]

次にSturm の定理に基づく2分（bisection）法について述べよう。[4)5)] ここでは$\mathbf{A} = (a_{ij})$を実対称3重対角行列とする。多項式の列

$$\begin{aligned} f_0(\lambda) &= 1 \\ f_1(\lambda) &= \lambda - a_{11} \\ f_i(\lambda) &= (\lambda - a_{ii}) f_{i-1}(\lambda) - a_{i-1,i}^2 f_{i-2}(\lambda) \end{aligned} \tag{65}$$

を考えると，$f_i(\lambda)$は行列$(\lambda\mathbf{I}-\mathbf{A})$のi 次の主行列式であり，特に$f_n(\lambda)=|\lambda\mathbf{I}-\mathbf{A}|$である。また，$a_{i,i-1}$が0のときには$\mathbf{A}$が3重対角行列の直和に縮退するので，$a_{i,i-1}\neq 0$ と考えてよい。このとき関数値の列

$$f_0(\mu),\ f_1(\mu),\ \cdots\cdots,\ f_n(\mu) \tag{66}$$

から0を除いて左から右へ見て行くとき，符号が変化した回数を$V(\mu)$で表わすことにすれば，$p<q$ に対し$f_n(p)\cdot f_n(q)\neq 0$ ならば，区間$[p,\ q]$にある$f_n(\lambda)=0$ の根の数は$V(q)-V(p)$となる。これがSturmの定理であるが，1954年 W. Givensは初めてこれを利用して区間を1/2ずつ狭めることにより解を得た。[5] この方法の特徴は区間を狭めればいくらでも解の精度を上げることができることであるが，逆に余り厳しい精度を要求し過ると余分な分割が入るので注意しなければならない。[36] また，区間の決め方が難しい場合は根の存在の範囲を示すGerschgorinの円[7]やRayleigh 商なども参考になろう。[4] 上の仮定$a_{i,i-1}\neq 0$は$\mathbf{A}$が重複固有値を持たないことに対応しているが，接近根の近くでは$f_i(\lambda)$の計算であふれ（overflow）が起き易いので新しい関数列

$$h_i(\lambda)=f_i(\lambda)/f_{i-1}(\lambda) \tag{67}$$

を用いることが多い。[36] $\mathbf{A}$が実非対称3重対角で$a_{i,i-1}\cdot a_{i-1,i}>0$ のときは，有理QR法のときと同様 (65)式の$a_{i-1,i}^2$ を$a_{i,i-1}\cdot a_{i-1,i}$ で置き換えればよい。以上のことからB－型の問題のときは与えられた区間$[p,\ q]$を出発点とすればよいが，I －型のときは求めようとする固有値の集合に対し，出来る限り多くの情報からその集合の上下界を評価することから始めなければならない。[36]

3重対角行列の固有ベクトルを求めるとき，性質T_pを持つことが重要であることは既に述べた。その係わりをSturm 列(66)式に因んだ方法を通して見てみよう。やはり$\mathbf{A}=(a_{ij})$ を対称な実3重対角行列とし，非対角要素はどれも0でないとする。$\mathbf{A}$の固有値λとその固有ベクトル$\mathbf{x}=(x_1,\ \cdots\cdots,\ x_n)^t$ の間には

$$\begin{aligned}
&(\lambda-a_{11})x_1-a_{12}x_2=0\\
&-a_{i,i-1}x_{i-1}+(\lambda-a_{ii})x_i-a_{i,i+1}x_{i+1}=0\\
&-a_{n,n-1}x_{n-1}+(\lambda-a_{nn})x_n=0
\end{aligned} \tag{68}$$

の関係が成り立つ。仮定から$x_1=1$ としてよく，これらの式は

$$x_i=f_{i-1}(\lambda)/\prod_{r=2}^{i}a_{r-1,r}\quad(i=2\sim n) \tag{69}$$

と書ける。更にべき乗法のときのように，$\lambda_1<\cdots\cdots<\lambda_n$ とし，それぞれに対応する固有ベクトルを$\mathbf{x}_1,\ \cdots\cdots,\ \mathbf{x}_n$ としよう。いま，$\tilde{\lambda}_j$がλ_jの近似値とし$\tilde{\lambda}_j\neq\lambda_j$となっているとする。(69)式を使うと，適当な定数ϕ_rにより

$$\tilde{\mathbf{x}}_j=\sum_{r=1}^{n}\frac{\phi_r}{(\lambda_r-\tilde{\lambda}_j)}\mathbf{x}_r \tag{70}$$

と表わせる。それ故すべてのi （$\neq$j ）に対し$|\phi_j/(\lambda_j-\tilde{\lambda}_j)|\gg|\phi_i/(\lambda_i-\tilde{\lambda}_j)|$

のときのみ$\widetilde{\mathbf{x}}_j$ は$\mathbf{x}_j$ の良い近似ベクトルとなり，単に$\widetilde{\lambda}_j$ がλ_j の良い近似値というだけでは不充分である。

これを一般化し修正したのはJ.H. Wilkinson であり，逆反復法[4)37)]と言われてS一型の問題の始んどの場合に使われている。$(\mathbf{w}^{(o)}, \mathbf{x}_j) \neq 0$ なる$\mathbf{w}^{(o)}=(w_1^{(o)}, \cdots\cdots, w_n^{(o)})^t$をもとに，ベクトル列

$$\mathbf{z}^{(k)}=\mathbf{w}^{(k)}/\max_i w_i^{(k)}$$
$$\mathbf{w}^{(k+1)}=(\mathbf{A}-\widetilde{\lambda}_j\mathbf{I})^{-1}\mathbf{z}^{(j)} \qquad (k=0, 1, \cdots\cdots) \tag{71}$$

を考える。$\mathbf{z}^{(o)}=\sum_{i=1}^{n}\psi_i\mathbf{x}_i$ としたとき，$\mathbf{z}^{(k)}$は$\sum_{i=1}^{n}\frac{\psi_i}{(\lambda_i-\widetilde{\lambda}_j)}\mathbf{x}_i$ に比例し，kが大きくなるに従って$|\psi_j/(\lambda_j-\widetilde{\lambda}_j)^k| \gg |\psi_i/(\lambda_i-\widetilde{\lambda}_j)^k|$となるから，$\mathbf{z}^{(k)}$の極限を$\mathbf{x}_j$とすればよい。$\mathbf{w}^{(o)}$としては$(1, \cdots\cdots, 1)^t$を使うことが多く，(71)式は$(\mathbf{A}-\widetilde{\lambda}_j\mathbf{I})$に3角化法$\mathbf{T}(\mathbf{A}-\widetilde{\lambda}_j\mathbf{I})=\mathbf{R}$を適用して$\mathbf{R}\mathbf{w}^{(k+1)}=\mathbf{T}\mathbf{z}^{(k)}$の形で解かれる。しかし，$\mathbf{A}$が3重対角行列のとき，$\mathbf{R}$は右上5重対角行列である。また，$\mathbf{A}$が実非対称行列のときは，Hessenberg 形にしてから逆反復法が適用されるのが普通である。[38)39)]

固有値λ_jが完全に収束した後の固有ベクトル$\mathbf{x}_j$ の求め方の一つとして，同次方程式

$$(\mathbf{A}-\lambda_j\mathbf{I})\mathbf{x}_j=\mathbf{o} \tag{72}$$

をGauss の消去法などで直接解くのもあろう。

4. 一般問題の数値解法

一般問題(2)式は，物理・工学上で遭遇する固有値問題の連続型の方程式を離散化した場合，標準問題(1)式よりもむしろ自然な形であると言える。このとき，行列$\mathbf{B}$が正則でかつ逆行列$\mathbf{B}^{-1}$が容易に求められるときは$\mathbf{B}^{-1}\mathbf{A}$をJacobi 法で対角化するなど標準問題として扱えるので，この章の主題は$\mathbf{B}$が非正則な場合，あるいは一般問題としての$\mathbf{B}$の効果的な逆転の方法になろう。

(2)式に対する固有多項式を

$$f(\lambda)=|\lambda\mathbf{B}-\mathbf{A}| \tag{73}$$

で定義したとき，$f(\lambda)\equiv 0$ であれば，任意のλが固有値となるので，以下$f(\lambda)\not\equiv 0$ として話を進める。一般に，n次の複素行列$\mathbf{A}$，$\mathbf{B}$に対し，$\mathbf{A}_1=\mathbf{Q}\mathbf{A}\mathbf{Z}$，$\mathbf{B}_1=\mathbf{Q}\mathbf{B}\mathbf{Z}$ が共に右上3角行列となるようなユニタリ行列$\mathbf{Q}$，$\mathbf{Z}$が存在する。[40)] このとき，$\mathbf{x}=\mathbf{Z}\mathbf{y}$ とすると，(2)式は

$$\mathbf{A}_1\mathbf{y}=\lambda\mathbf{B}_1\mathbf{y} \tag{74}$$

に移り，同じ固有値を持つことになる。この関係はユニタリ同値といわれ，この性質に基づく解法をQZ法[40)41)]という。この場合，$\mathbf{A}_1$の対角要素をa_{ii} $(i=1\sim n)$，$\mathbf{B}_1$の対角要素をb_{ii} $(i=1\sim n)$とすれば，(2)式および(74)式の固有値は$\lambda_i=a_{ii}/b_{ii}$ で表わされる。ここに，b_{ii}

のときのみ$\tilde{\mathbf{x}}_j$ は$\mathbf{x}_j$ の良い近似ベクトルとなり，単に$\tilde{\lambda}_j$ がλ_j の良い近似値というだけでは不充分である。

これを一般化し修正したのはJ.H. Wilkinson であり，逆反復法[4)37)]と言われてS－型の問題の始んどの場合に使われている。$(\mathbf{w}^{(o)}, \mathbf{x}_j) \neq 0$ なる$\mathbf{w}^{(o)}=(w_1^{(o)}, \cdots\cdots, w_n^{(o)})^t$をもとに，ベクトル列

$$\mathbf{z}^{(k)}=\mathbf{w}^{(k)}/\max_i w_i^{(k)}$$
$$\mathbf{w}^{(k+1)}=(\mathbf{A}-\tilde{\lambda}_j\mathbf{I})^{-1}\mathbf{z}^{(j)} \qquad (k=0, 1, \cdots\cdots) \tag{71}$$

を考える。$\mathbf{z}^{(o)}=\sum_{i=1}^{n}\psi_i\mathbf{x}_i$ としたとき，$\mathbf{z}^{(k)}$は $\sum_{i=1}^{n}\frac{\psi_i}{(\lambda_i-\tilde{\lambda}_j)}\mathbf{x}_i$ に比例し，k が大きくなるに従って$|\psi_j/(\lambda_j-\tilde{\lambda}_j)^k| \gg |\psi_i/(\lambda_i-\tilde{\lambda}_j)^k|$となるから，$\mathbf{z}^{(k)}$の極限を$\mathbf{x}_j$とすればよい。$\mathbf{w}^{(o)}$としては$(1, \cdots\cdots, 1)^t$を使うことが多く，(71)式は$(\mathbf{A}-\tilde{\lambda}_j\mathbf{I})$に3角化法$\mathbf{T}(\mathbf{A}-\tilde{\lambda}_j\mathbf{I})=\mathbf{R}$を適用して$\mathbf{R}\mathbf{w}^{(k+1)}=\mathbf{T}\mathbf{z}^{(k)}$の形で解かれる。しかし，$\mathbf{A}$が3重対角行列のとき，$\mathbf{R}$は右上5重対角行列である。また，$\mathbf{A}$が実非対称行列のときは，Hessenberg 形にしてから逆反復法が適用されるのが普通である。[38)39)]

固有値λ_jが完全に収束した後の固有ベクトル$\mathbf{x}_j$ の求め方の一つとして，同次方程式

$$(\mathbf{A}-\lambda_j\mathbf{I})\mathbf{x}_j=\mathbf{o} \tag{72}$$

をGauss の消去法などで直接解くのもあろう。

4. 一般問題の数値解法

一般問題(2)式は，物理・工学上で遭遇する固有値問題の連続型の方程式を離散化した場合，標準問題(1)式よりもむしろ自然な形であると言える。このとき，行列$\mathbf{B}$が正則でかつ逆行列$\mathbf{B}^{-1}$が容易に求められるときは$\mathbf{B}^{-1}\mathbf{A}$をJacobi 法で対角化するなど標準問題として扱えるので，この章の主題は$\mathbf{B}$が非正則な場合，あるいは一般問題としての$\mathbf{B}$の効果的な逆転の方法になろう。

(2)式に対する固有多項式を

$$f(\lambda)=|\lambda\mathbf{B}-\mathbf{A}| \tag{73}$$

で定義したとき，$f(\lambda)\equiv 0$ であれば，任意のλが固有値となるので，以下 $f(\lambda)\not\equiv 0$ として話を進める。一般に，n 次の複素行列$\mathbf{A}$，$\mathbf{B}$に対し，$\mathbf{A}_1=\mathbf{Q}\mathbf{A}\mathbf{Z}$，$\mathbf{B}_1=\mathbf{Q}\mathbf{B}\mathbf{Z}$ が共に右上3角行列となるようなユニタリ行列$\mathbf{Q}$，$\mathbf{Z}$が存在する。[40)] このとき，$\mathbf{x}=\mathbf{Z}\mathbf{y}$ とすると，(2)式は

$$\mathbf{A}_1\mathbf{y}=\lambda\mathbf{B}_1\mathbf{y} \tag{74}$$

に移り，同じ固有値を持つことになる。この関係はユニタリ同値といわれ，この性質に基づく解法をQZ法[40)41)]という。この場合，$\mathbf{A}_1$の対角要素をa_{ii} $(i=1\sim n)$，$\mathbf{B}_1$ の対角要素をb_{ii} $(i=1\sim n)$とすれば，(2)式および(74)式の固有値は$\lambda_i=a_{ii}/b_{ii}$ で表わされる。ここに，b_{ii}

$=0$ なるときは $|\mathbf{B}|=0$ であり，f（λ）の次数はnより小さくなるので $\lambda_i=\infty$ を固有値とみなしてよい。これは$\mathbf{B}$ の摂動$\mathbf{B}+\mathbf{F}$ を考えてみれば納得がいく。

QZ 法の手順は概略次の通りである。

① Householder 法で$\mathbf{B}$ を右上3角行列にする。このとき，$\mathbf{A}$にも同じ変換を施す。

② 変換された$\mathbf{A}$をHessenberg 形にする。このとき，$\mathbf{B}$ の右上3角がくずれないようにする。

③ 更に$\mathbf{A}$を擬3角形（quasi－triangular from）にすると同時に$\mathbf{B}$ の右上3角がくずれないようにする。

④ $\mathbf{A}$を右上3角にすると同時に$\mathbf{B}$ の右上3角がくずれないようにする。

ここに擬3角形とは，右上Hessenberg 行列であって，下対角で非零要素が2つと続かないものをいう。$\mathbf{A}$，$\mathbf{B}$がともに実行列であるとき，過程①，②を具体的に述べよう。いま，実ベクトル $\mathbf{u}=(\mathbf{u}_1,\cdots\cdots,\mathbf{u}_n)^t$ において$u_i=0$（$i=1\sim j-1, j+r\sim n$），$u_j=1$とする。$\varphi=\sum_{i=j+1}^{j+r-1} u_i^2$ とおけば，ベクトル $\mathbf{u}_1=\dfrac{-2}{(1+\varphi)}\mathbf{u}$ に対し $\mathbf{u}_1^t\mathbf{u}=-2$ となるので，直交変換の集合

$$H_r(j)=\{\mathbf{I}+\mathbf{u}_1\mathbf{u}^t\} \tag{75}$$

を考えることができる。このとき，

① $i=1\sim n-1$に対して，$\mathbf{B}$の（$i+1, i$）～（n, i）要素を0にするように$\mathbf{Q}_i\in H_{n-i+1}$ を選び，$\mathbf{Q}_{n-1}\cdot\cdots\cdot\mathbf{Q}_1\mathbf{B}\to\mathbf{B}$，　$\mathbf{Q}_{n-1}\cdot\cdots\cdot\mathbf{Q}_1\mathbf{A}\to\mathbf{A}$ とする。

② $i=1\sim n-2$ごとの$j=n-1\sim i+1$に対し，$\mathbf{A}$の（$j+1, i$）要素を0にするよう $\mathbf{Q}_{ij}\in H_2(j)$を選ぶ。続いて$\mathbf{Q}_{ij}$ $\mathbf{B}$ の左下の非零になった（$j+1, j$）要素を再び0にするよう$\mathbf{Z}_{ij}\in H_2(j)$を選び，$\mathbf{Q}=\mathbf{Q}_{n-2,i+1}\cdot\cdots\cdot\mathbf{Q}_{1,n-1}$，$\mathbf{Z}=\mathbf{Z}_{1,n-1}\cdot\cdots\cdot\mathbf{Z}_{n-2,i+1}$ に対して$\mathbf{QAZ}\to\mathbf{A}$，$\mathbf{QBZ}\to\mathbf{B}$

とすればよい。過程③，④は主に2次因子の扱いであるが，ここに原点移動を取り入れることもできる。

QZ 法の特徴は$\mathbf{B}$ が特異（singular）のときでも解けることにあり，特に$\mathbf{B}=\mathbf{I}$ のときはQR法に帰着する。固有ベクトルは右上3角行列$\mathbf{A}_1$，$\mathbf{B}_1$ から得られた固有ベクトルに$\mathbf{Z}$ を掛ければよい。また，$\mathbf{A}$ がHermite または実対称で$\mathbf{B}$ が実対称正定値の場合，$\mathbf{B}=\mathbf{R}^t\mathbf{R}$ とCholesky分解し，

$$(\mathbf{R}^t)^{-1}\mathbf{A}\mathbf{R}^{-1}(\mathbf{R}\mathbf{x})=\lambda\mathbf{R}\mathbf{x} \tag{76}$$

として解　ることが多い。[42] 従って，対称行列$(\mathbf{R}^t)^{-1}\mathbf{A}\mathbf{R}^{-1}$ が求まれば，標準問題に帰着できる訳である。変形された一般問題(3)式も同様に

$$\mathbf{R}\mathbf{A}\mathbf{R}^t\mathbf{R}\mathbf{x}=\lambda\mathbf{R}\mathbf{x} \tag{77}$$

などと変換して簡単に解かれる。[42]

次に一般問題におけるN－型の特殊問題を考えてみよう。[7)13)] ここでは$\mathbf{A}$をHermite，$\mathbf{B}$を実対称正定値とする。λ_0をある目標値とすれば，これに最も近いr 個の固有値を求める問題は

$$\mathbf{A}_1\mathbf{x} = \mu\mathbf{B}\mathbf{x}$$
$$\mathbf{A}_1 = \mathbf{A} - \lambda_0\mathbf{B}, \quad \mu = \lambda - \lambda_0 \tag{78}$$

の最も小さいr 個を求めることに還元できる。$\mathbf{A}_1 = \mathbf{L}\mathbf{D}\mathbf{L}^*$, $\mathbf{B} = \mathbf{R}^t\mathbf{R}$ とCholesky 分解を行い，初期ベクトル $\mathbf{x}_1^{(0)}$, ……, $\mathbf{x}_r^{(0)}$ に対する反復

$$\mathbf{p}_j^{(k)} = \mathbf{R}\,\mathbf{x}_j^{(k)}, \quad \mathbf{p}_j^{(k)} \text{の正規直交化},$$
$$\mathbf{h}_j^{(k)} = \mathbf{R}^t\,\mathbf{p}_j^{(k)}, \quad \mathbf{z}_j^{(k+1)} = \mathbf{L}^{-1}\,\mathbf{h}_j^{(k)} \tag{79}$$
$$\mathbf{w}_j^{(k+1)} = \mathbf{D}^{-1}\,\mathbf{z}_j^{(k+1)}, \quad \mathbf{x}_j^{(k+1)} = (\mathbf{L}^*)^{-1}\,\mathbf{w}_j^{(k+1)}$$

を考えると，$\mathbf{x}_j^{(k)}$ は求めるべき固有ベクトル$\mathbf{x}_j$ に収束する。このとき，$\mathbf{x}_j$ に対応する固有値は

$$\mu_j = \frac{\mathbf{x}_j^*\,\mathbf{A}_1\,\mathbf{x}_j}{\mathbf{x}_j^*\,\mathbf{B}\,\mathbf{x}_j} \tag{80}$$

から得られる。ここに，$\mathbf{x}_j^*\,\mathbf{B}\,\mathbf{x}_j > 0$ であるが，$\mathbf{A}_1$が特異のとき$\mathbf{D}^{-1}$ の計算ができないので，$\mathbf{A}_1$ を更に少し移動させてやればよい。以上のような方法は，多くの初期ベクトルを用いるため多段階（ simultaneous ）逆反復法と呼ばれている。べき乗法もそうであるが，このような解法で精度良く解を求められるのはせいぜい数個であることを指摘しておこう。

5. 縮退した問題の数値解法

系を定める$\mathbf{A}$ が正方行列でない場合を本稿では「縮退している」と呼ぶことにする。一般には $\mathbf{A} = (a_{ij})$ ($i = 1 \sim m$, $j = 1 \sim n$)に対し，$m \gtrless n$ と書けるが，説明の都合上$\mathbf{A}$ は実行列とし，$m \geqq n$ とする。このとき，$\mathbf{A}\mathbf{A}^t$, $\mathbf{A}^t\mathbf{A}$ はそれぞれm次とn 次の対称で非負な定値行列となり，これらの固有値も非負となる。[4)43)] いま，$\mathbf{A}\mathbf{A}^t$ の大きいn 個の固有値に対応する固有ベクトルを$\mathbf{u}_j = (u_{1j}, \cdots\cdots, u_{mj})^t$ ($j = 1 \sim m$) とし，$\mathbf{A}^t\mathbf{A}$ の固有値 λ_j^2 ($\lambda_1 \geqq \cdots \geqq \lambda_n \geqq 0$) に対応する固有ベクトルを$\mathbf{v}_j = (v_{1j}, \cdots\cdots, v_{nj})^t$ ($j = 1 \sim n$) としたとき，これらが正規直交化されているとすれば，$\mathbf{A}$ はこれらにより

$$\mathbf{A} = \mathbf{U}\,\Lambda\,\mathbf{V}^t \tag{81}$$

と分解される。ここに，$\mathbf{U} = (\mathbf{u}_1, \cdots\cdots, \mathbf{u}_n)$, $\Lambda = \mathrm{diag}(\lambda_1, \cdots\cdots, \lambda_n)$, $\mathbf{V} = (\mathbf{v}_1, \cdots\cdots, \mathbf{v}_n)$, $\mathbf{U}^t\mathbf{U} = \mathbf{V}\mathbf{V}^t = \mathbf{V}^t\mathbf{V} = \mathbf{I}_n$ で，λ_i を$\mathbf{A}$ の特異値といい，(81)式を$\mathbf{A}$ の特異値分解という。[43)] この分解は$\mathbf{A}$ の特質を具体的に表わすものとして，最小自乗法などに広く応用される。[43)]

縮退した$\mathbf{A}$ に対し，(n , m) 行列$\mathbf{G}$ が唯一つ存在して

$$\mathbf{A}\mathbf{G}\mathbf{A} = \mathbf{A}, \quad \mathbf{G}\mathbf{A}\mathbf{G} = \mathbf{G},$$
$$(\mathbf{A}\mathbf{G})^t = \mathbf{A}\mathbf{G}, \quad (\mathbf{G}\mathbf{A})^t = \mathbf{G}\mathbf{A} \tag{82}$$

が成り立つ。この$\mathbf{G}$ は$\mathbf{A}$ の一般化逆行列（ generalized inverse ） といわれ，特に$\mathbf{A}$ が正方行列

$$\mathbf{A}_1 \mathbf{x} = \mu \mathbf{B} \mathbf{x}$$
$$\mathbf{A}_1 = \mathbf{A} - \lambda_0 \mathbf{B}, \quad \mu = \lambda - \lambda_0 \tag{78}$$

の最も小さいr 個を求めることに還元できる。$\mathbf{A}_1 = \mathbf{L}\mathbf{D}\mathbf{L}^*$, $\mathbf{B} = \mathbf{R}^t\mathbf{R}$ とCholesky 分解を行い，初期ベクトル $\mathbf{x}_1^{(0)}$, ……, $\mathbf{x}_r^{(0)}$ に対する反復

$$\mathbf{p}_j^{(k)} = \mathbf{R}\,\mathbf{x}_j^{(k)}, \quad \mathbf{p}_j^{(k)} \text{ の正規直交化},$$
$$\mathbf{h}_j^{(k)} = \mathbf{R}^t\,\mathbf{p}_j^{(k)}, \quad \mathbf{z}_j^{(k+1)} = \mathbf{L}^{-1}\,\mathbf{h}_j^{(k)} \tag{79}$$
$$\mathbf{w}_j^{(k+1)} = \mathbf{D}^{-1}\,\mathbf{z}_j^{(k+1)}, \quad \mathbf{x}_j^{(k+1)} = (\mathbf{L}^*)^{-1}\,\mathbf{w}_j^{(k+1)}$$

を考えると，$\mathbf{x}_j^{(k)}$ は求めるべき固有ベクトル$\mathbf{x}_j$ に収束する。このとき，$\mathbf{x}_j$ に対応する固有値は

$$\mu_j = \frac{\mathbf{x}_j^* \mathbf{A}_1 \mathbf{x}_j}{\mathbf{x}_j^* \mathbf{B} \mathbf{x}_j} \tag{80}$$

から得られる。ここに，$\mathbf{x}_j^* \mathbf{B} \mathbf{x}_j > 0$ であるが，$\mathbf{A}_1$ が特異のとき$\mathbf{D}^{-1}$ の計算ができないので，$\mathbf{A}_1$ を更に少し移動させてやればよい。以上のような方法は，多くの初期ベクトルを用いるため多段階（simultaneous）逆反復法と呼ばれている。べき乗法もそうであるが，このような解法で精度良く解を求められるのはせいぜい数個であることを指摘しておこう。

5. 縮退した問題の数値解法

系を定める$\mathbf{A}$ が正方行列でない場合を本稿では「縮退している」と呼ぶことにする。一般には $\mathbf{A} = (a_{ij})$ ($i = 1 \sim m$, $j = 1 \sim n$) に対し，$m \gtrless n$ と書けるが，説明の都合上$\mathbf{A}$ は実行列とし，$m \geqq n$ とする。このとき，$\mathbf{A}\mathbf{A}^t$, $\mathbf{A}^t\mathbf{A}$ はそれぞれm次とn 次の対称で非負な定値行列となり，これらの固有値も非負となる。[4)43)] いま，$\mathbf{A}\mathbf{A}^t$ の大きいn 個の固有値に対応する固有ベクトルを$\mathbf{u}_j = (u_{1j}, \cdots\cdots, u_{mj})^t$ ($j = 1 \sim m$) とし，$\mathbf{A}^t\mathbf{A}$ の固有値 λ_j^2 ($\lambda_1 \geqq \cdots \geqq \lambda_n \geqq 0$) に対応する固有ベクトルを$\mathbf{v}_j = (v_{1j}, \cdots\cdots, v_{nj})^t$ ($j = 1 \sim n$) としたとき，これらが正規直交化されているとすれば，$\mathbf{A}$ はこれらにより

$$\mathbf{A} = \mathbf{U}\Lambda\mathbf{V}^t \tag{81}$$

と分解される。ここに，$\mathbf{U} = (\mathbf{u}_1, \cdots\cdots, \mathbf{u}_n)$, $\Lambda = \mathrm{diag}(\lambda_1, \cdots\cdots, \lambda_n)$, $\mathbf{V} = (\mathbf{v}_1, \cdots\cdots, \mathbf{v}_n)$, $\mathbf{U}^t\mathbf{U} = \mathbf{V}\mathbf{V}^t = \mathbf{V}^t\mathbf{V} = \mathbf{I}_n$ で，λ_i を$\mathbf{A}$ の特異値といい，(81)式を$\mathbf{A}$ の特異値分解という。[43)] この分解は$\mathbf{A}$ の特質を具体的に表わすものとして，最小自乗法などに広く応用される。[43)]

縮退した$\mathbf{A}$ に対し，(n, m) 行列$\mathbf{G}$ が唯一つ存在して

$$\mathbf{A}\mathbf{G}\mathbf{A} = \mathbf{A}, \quad \mathbf{G}\mathbf{A}\mathbf{G} = \mathbf{G},$$
$$(\mathbf{A}\mathbf{G})^t = \mathbf{A}\mathbf{G}, \quad (\mathbf{G}\mathbf{A})^t = \mathbf{G}\mathbf{A} \tag{82}$$

が成り立つ。この$\mathbf{G}$ は$\mathbf{A}$ の一般化逆行列（generalized inverse）といわれ，特に$\mathbf{A}$ が正方行列

であれば$\mathbf{A}^{-1}$に一致するので，$\mathbf{G}=\mathbf{A}^{-1}$と記すことができる。$\mathbf{A}$の特異値分解(81)式から$\lambda_j>0$のとき$s_j=1/\lambda_j$，$\lambda_j=0$のとき$s_j=0$として$\mathbf{S}=\mathrm{diag}(s_1, \cdots\cdots, s_n)$とすれば，$\mathbf{A}^{-1}=\mathbf{V}\mathbf{S}\mathbf{U}^t$と表わせる。

連立一次方程式が縮退した場合の

$$\mathbf{A}\mathbf{x}=\mathbf{c} \tag{83}$$

を解いてみよう。これはn元m次の方程式となるので厳密な意味での解$\mathbf{x}$は存在しない。しかし，残差$\mathbf{r}=\mathbf{c}-\mathbf{A}\mathbf{x}$の$\mathbf{L}_2$ーノルムを最小とする$\mathbf{x}$の中で，$\mathbf{x}$自身のノルムも最小にする極小ノルム解(minimal norm solution)は唯一つ存在し，一般化逆行列を用いて

$$\mathbf{x}=\mathbf{A}^{-1}\mathbf{c} \tag{84}$$

と表わせる。

一般に$\mathbf{A}$の階数(rank)はn以下であるが，それは特異値より明らかで，$\lambda_r>\lambda_{r+1}=0$のときrank($\mathbf{A}$)＝rとなる。このときは同次方程式

$$\mathbf{A}\mathbf{x}=\mathbf{o} \tag{85}$$

は(n－r)個の独立解$\mathbf{x}_j$(j＝r＋1～n)を持つ。これは関係式$\mathbf{AV}=\mathbf{US}$より$\mathbf{A}\mathbf{v}_j=\mathbf{o}$となることが直ちに導かれよう。

6. 既存の副プログラム

科学・技術計算の基礎となる汎用副プログラムはなるべく標準的な解法や機能を備えていることが望ましい。原研にもかかる目的のために公開されたライブラリーが3つあり，[9)～12)] 各々固有値問題を扱うルーチンがいくつか含まれている。Table 1～3はこれらをライブラリーごとに表にしたものであるが，引き続きベンチマーク・テストにも活用されるため，下のような観点でまとめられている。

① 例えば，ルーチンSUB1でHessenberg行列に変換し，ルーチンSUB2で固有値を求めるという組合わせになったルーチンは2つを1つにまとめる。

② 固有値が既に分っていて，それを元に固有ベクトルだけ求めるような場合，固有値を求める適当なルーチンと一緒にする。

③ 例えば，規格化を行うルーチンを選択して呼べる場合，呼んだ組と呼ばない組の2つに分ける。

④ 倍精度計算のルーチンのみが掲げてあるが，殆んどの場合対応する単精度のルーチンが用意されている。

⑤ 数値解法は使用手引書に書かれているので，ここでは骨子となる解法のみ記す。

SSL・H[9)10)]では，標準問題，一般問題，それに一般化逆行列を求めるルーチンが一通り揃えてある。固有値は多くの場合，全てを求めるようになっているが，Householder法によるものは数個の大きい固有値を求め，必要に応じて対応する固有ベクトルも求められるようになっている。

であれば$\mathbf{A}^{-1}$に一致するので，$\mathbf{G}=\mathbf{A}^{-1}$と記すことができる。$\mathbf{A}$の特異値分解(81)式から$\lambda_j>0$のとき$s_j=1/\lambda_j$，$\lambda_j=0$のとき$s_j=0$として$\mathbf{S}=\mathrm{diag}(s_1, \cdots\cdots, s_n)$とすれば，$\mathbf{A}^{-1}=\mathbf{V}\mathbf{S}\mathbf{U}^t$と表わせる。

連立一次方程式が縮退した場合の

$$\mathbf{A}\mathbf{x}=\mathbf{c} \tag{83}$$

を解いてみよう。これはn元m次の方程式となるので厳密な意味での解$\mathbf{x}$は存在しない。しかし，残差$\mathbf{r}=\mathbf{c}-\mathbf{A}\mathbf{x}$の$\mathbf{L}_2$ーノルムを最小とする$\mathbf{x}$の中で，$\mathbf{x}$自身のノルムも最小にする極小ノルム解(minimal norm solution)は唯一つ存在し，一般化逆行列を用いて

$$\mathbf{x}=\mathbf{A}^{-1}\mathbf{c} \tag{84}$$

と表わせる。

一般に$\mathbf{A}$の階数(rank)はn以下であるが，それは特異値より明らかで，$\lambda_r>\lambda_{r+1}=0$のときrank($\mathbf{A}$)＝rとなる。このときは同次方程式

$$\mathbf{A}\mathbf{x}=\mathbf{o} \tag{85}$$

は(n－r)個の独立解$\mathbf{x}_j$(j＝r＋1～n)を持つ。これは関係式$\mathbf{AV}=\mathbf{US}$より$\mathbf{A}\mathbf{v}_j=\mathbf{o}$となることが直ちに導かれよう。

6. 既存の副プログラム

科学・技術計算の基礎となる汎用副プログラムはなるべく標準的な解法や機能を備えていることが望ましい。原研にもかかる目的のために公開されたライブラリーが3つあり，[9)～12)] 各々固有値問題を扱うルーチンがいくつか含まれている。Table 1～3はこれらをライブラリーごとに表にしたものであるが，引き続きベンチマーク・テストにも活用されるため，下のような観点でまとめられている。

① 例えば，ルーチンSUB1でHessenberg行列に変換し，ルーチンSUB2で固有値を求めるという組合わせになったルーチンは2つを1つにまとめる。

② 固有値が既に分っていて，それを元に固有ベクトルだけ求めるような場合，固有値を求める適当なルーチンと一諸にする。

③ 例えば，規格化を行うルーチンを選択して呼べる場合，呼んだ組と呼ばない組の2つに分ける。

④ 倍精度計算のルーチンのみが掲げてあるが，殆んどの場合対応する単精度のルーチンが用意されている。

⑤ 数値解法は使用手引書に書かれているので，ここでは骨子となる解法のみ記す。

SSL・H[9)10)]では，標準問題，一般問題，それに一般化逆行列を求めるルーチンが一通り揃えてある。固有値は多くの場合，全てを求めるようになっているが，Householder法によるものは数個の大きい固有値を求め，必要に応じて対応する固有ベクトルも求められるようになっている。

Table 1 Subroutines for eigenproblems in SSL.H[9)10)]

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
DABAD	Standard	Real	All	No	Danilevsky, Bairstow
QREGND	"	"	"	"	Double QR
DANEWD	"	"	"	Corresp.	Danilevsky, Bairstow, sweep out
HESQRD	"	"	"	"	QR, inv. iter.
QREGND GAVE2D	"	"	"	Corresp. right, left	Double QR, inv. iter.
JACOBD	"	Real, sym.	"	Corresp.	Threshold Jacobi
HOUSD	"	"	Exreme	No	Householder
HOUS2D	"	"	"	Corresp.	"
CHSQRD	"	Complex	All	"	QR, inv. iter.
HERMTD	"	Hermite	"	"	Greenstadt
THJACD	"	"	"	"	Threshold Jacobi
HMTQRD	"	"	"	"	QR, inv. iter.
GEIGND	Generalized	A,B: real, sym. B: pos. def.	"	"	QR
GMINVD	Gen. inv.	Degenerated, real			Gram-Schmidt orthog.

Table 2 Subroutines for eigenproblems in SSL-II[11)]

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
DHES1 DHSQR	Standard	Real	All	No	Householder, double QR
DBLNC DHES1 DHSQR	"	"	"	"	Balancing, Householder, double QR
DEIG1	"	"	"	Corresp.	Double QR
DHES1 DHSQR DHVEC DHBK1	"	"	"	Selected	Householder, double QR, inv. iter.
DBLNC DHES1 DHSQR DHVEC DHBK1	"	"	"	"	Balancing, Householder, double QR, inv. iter.
DTRID1 DTRQL	"	Real, sym.	"	No	Householder, QL
DSEIG1	"	"	"	Corresp.	QL
DTRID1 DBSCT1	"	"	Extreme	No	Householder, bisection
DSEIG2	"	"	"	Corresp.	Bisection, inv. iter.
DTRQL	"	Real, sym. tridiag.	All	No	QL

Table 2 (cont'd)

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
DTEIG1	Standard	Real, sym. tridiag.	All	Corresp.	QL
DBSCT1	"	"	Extreme	No	Bisection
DTEIG2	"	"	"	Corresp.	Bisection, inv. iter.
DTRIDH DTRQL	"	Hermite	All	No	Householder, diag. unit. transf., QL
DTRIDH DTEIG1 DTRBKH	"	"	"	Corresp.	"
DTRIDH DBSCT1	"	"	Extreme	No	Householder, diag. unit. transf., bisection
DHEIG2	"	"	"	Corresp.	Householder, bisection, inv. iter.

Table 3 Subroutine for eigenproblem in JSSL[12)]

Entry name	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
BISCTD	Standard	Real, tridiag.(T_p)	Indexed	No	Bisection

(1)式による固有ベクトル$\mathbf{x}$ はλの右ベクトルとも呼ばれるが，GAVE 2Dでは，固有値がすべて求まっている場合にこれらを使ってすべての左ベクトル

$$\mathbf{y}^t \mathbf{A} = \lambda \mathbf{y}^t \tag{86}$$

も計算される。解法も，現在は余り使われなくなった直接法のものも含まれていて，ベンチマーク・テストを行う際非常に興味深い。詳しい条件は各々の使用手引きに参照されるが，このライブラリーのルーチンの多くは収束判定因子の仮引数をもっており，計算の精度と時間の兼合いにかなりの幅を持たせることができよう。

SSL－Ⅱ[11]のライブラリーはSSL・H を全面的に改良し，多機種にわたって使えるよう融通性を高めたものである。このうち固有値問題を解くルーチンは後で述べる EISPACK－2[13]と同様なアルゴリズムを用いているが，次のような改良がなされている。[44]

① 3重対角化等の変換の部分は，多少計算時間を要しても，固有値の精度を高めるため一部精度を上げて計算する。

② ベクトルの内積の計算を速めるため，Assembler 言語を用いる。

ＳＳＬ－Ⅱのルーチンは標準問題に限られているが，よく出くわす系について，完全問題と特殊問題，それに固有ベクトルを求めないものと求めるもの，規格化を行わないものと行うものというふうに系統的に揃っている。このライブラリーの場合，特殊問題は小さい方の数個の固有値も求めることができ，DHVEC のように求まった固有値の中から任意に選んで対応するベクトルを求めるものもある。なお，現在欠けている複素の系の標準問題や一般問題等を解くルーチンは追って開発される予定である。[11]

JSSL[12]の固有値問題を解くルーチンは，これまでのところ1件のみである。「性質T_p」とは，(37)式と同様に副対角要素が

$$a_{i,i-1} \cdot a_{i-1,i} > 0 \qquad (i = 2 \sim n) \tag{87}$$

をみたす場合であり，固有値を「大小順に数えて何番目から何番目まで」というふうに指定できるＩ－型のルーチンである。

既存のルーチンを見ると，実行列やHermite 行列の一般問題を解くルーチンは良く揃っているといえよう。しかし，帯行列を含めた疎行列の一般問題，例えばある区間にある固有値を求めるＢ－型等の特殊問題，それに一般の実行列を対称にした一般問題などは実際の需要が多いにも拘らず，これらのライブラリーには見当らない。そのため，今回はこれらの分野が重点的に拡充されることになったが，それらは後の章で詳しく述べられる。

7. EISPACK-J の開発

既にみてきたように，原研における固有値計算ルーチンは充分と言えず，これを補うために，既に評価が高く多種類の問題が扱える固有値問題専用副プログラム・パッケージEISPACK－2[13]を発展させたEISPACK－J を開発した。

EISPACK（Eigensystem package）[13]は元々米国 ANL で開発されたものであるが，1976

(1)式による固有ベクトル$\mathbf{x}$ はλの右ベクトルとも呼ばれるが，GAVE 2Dでは，固有値がすべて求まっている場合にこれらを使ってすべての左ベクトル

$$\mathbf{y}^t \mathbf{A} = \lambda \mathbf{y}^t \tag{86}$$

も計算される。解法も，現在は余り使われなくなった直接法のものも含まれていて，ベンチマーク・テストを行う際非常に興味深い。詳しい条件は各々の使用手引きに参照されるが，このライブラリーのルーチンの多くは収束判定因子の仮引数をもっており，計算の精度と時間の兼合いにかなりの幅を持たせることができよう。

SSL－Ⅱ[11]のライブラリーはSSL・H を全面的に改良し，多機種にわたって使えるよう融通性を高めたものである。このうち固有値問題を解くルーチンは後で述べる EISPACK－2[13]と同様なアルゴリズムを用いているが，次のような改良がなされている。[44]

① 3重対角化等の変換の部分は，多少計算時間を要しても，固有値の精度を高めるため一部精度を上げて計算する。

② ベクトルの内積の計算を速めるため，Assembler 言語を用いる。

ＳＳＬ－Ⅱのルーチンは標準問題に限られているが，よく出くわす系について，完全問題と特殊問題，それに固有ベクトルを求めないものと求めるもの，規格化を行わないものと行うものというふうに系統的に揃っている。このライブラリーの場合，特殊問題は小さい方の数個の固有値も求めることができ，DHVEC のように求まった固有値の中から任意に選んで対応するベクトルを求めるものもある。なお，現在欠けている複素の系の標準問題や一般問題等を解くルーチンは追って開発される予定である。[11]

JSSL[12]の固有値問題を解くルーチンは，これまでのところ１件のみである。「性質T_p」とは，(37)式と同様に副対角要素が

$$a_{i,i-1} \cdot a_{i-1,i} > 0 \qquad (i = 2 \sim n) \tag{87}$$

をみたす場合であり，固有値を「大小順に数えて何番目から何番目まで」というふうに指定できるＩ－型のルーチンである。

既存のルーチンを見ると，実行列やHermite 行列の一般問題を解くルーチンは良く揃っているといえよう。しかし，帯行列を含めた疎行列の一般問題，例えばある区間にある固有値を求めるＢ－型等の特殊問題，それに一般の実行列を対称にした一般問題などは実際の需要が多いにも拘らず，これらのライブラリーには見当らない。そのため，今回はこれらの分野が重点的に拡充されることになったが，それらは後の章で詳しく述べられる。

7. EISPACK-J の開発

既にみてきたように，原研における固有値計算ルーチンは充分と言えず，これを補うために，既に評価が高く多種類の問題が扱える固有値問題専用副プログラム・パッケージEISPACK－2[13]を発展させたEISPACK－Ｊ を開発した。

EISPACK（Eigensystem package）[13]は元々米国 ANL で開発されたものであるが，1976

年，これを改良した第2版（Release 2）が出された。改良点の主なものは，一般問題や実対称帯行列の標準問題，それに縮退した問題が扱えるようになったこととエラー処理である。第2版にはIBM社の計算機用に作られたものと，そうでないものとの2種類がある。どちらも計算の基本となる，3重対角化や3重対角行列の固有値を計算するルーチン，あるいは固有ベクトルを計算するルーチン等から成り立っているが，前者は問題を規定するパラメーターを送れば，それに必要なルーチンのみ自動的に編集して計算を行うことができる。一方，他の計算機にこの制度を適用すると，すべてのルーチンを編集せねばならず，多大な記憶容量を要する。そのため，後者では必要なルーチンを流れ図[13]に沿って呼び出さなければならず，誤りをおかしやすく手間もかかる。

EISPACK−J はFACOM 230−75計算機のために後者を発展させたものであり，上記の事情を考慮して次のような試みがなされている。

① 一つの問題ごとに一度の呼び出しで済むようルーチンをまとめる。

② 初期設定はまとめたルーチンの中に組込む。

③ 同次方程式，一般化逆行列，極小ノルム解のルーチンを追加する。

④ 元のルーチンも単独で呼び出せるようにする。

⑤ 使用手引きで，系を表わす行列の性質を明確にする。

⑥ エラー状態を示す数の代わりに，求まった固有値や固有ベクトルの数自身を出力する。

また，このパッケージの開発にあたり，次のような変更もなされた。

① EISPACK−2の流れ図[13]で，実対称帯行列の標準問題と実対称行列の一般問題の場合の誤りが訂正された。

② 固有値が分ったあと，固有ベクトルを選んで計算する場合のIMTQLV とTINVIT を含むS −型の特殊問題の流れが省略された。

③ 3重対角行列に変換したあと，副対角要素の2乗を用いて計算するルーチンTQLRATを用いる流れは逐行性が悪いときもあり省略された。[28) 33)]

④ 規格化や収束判定の基礎となる数RADIXとMACHEP[13]がそれぞれ，16，16^{-13}から2，2^{-59}へ変更された。

このパッケージにどのような種類のルーチンがあるかをTable 4で見てみよう。全部で107件になるがEJ……の名で統一されている。この中でのまとまりを良くするため，必ずしも他のライブラリーと同じ並べ方とは限らない。既存のルーチンに見当らない機能としては，実行列に対する一般問題，変形された一般問題，縮退した問題，それにB −型やN −型の特殊問題などである。帯行列を対称としたものもこのパッケージ特有のものである。なお，一次方程式関係のルーチンは既に数多く存在する[9)〜12)45)]が，これらは固有ベクトルを求めるとき使われるので，ついでに掲げてある。一般に解法の名は定まってなく，QR法のことをユニタリ変換と言ったりすることもある。また，2重QR法は実行列に対して意味をもつが，このときも単にQR法と書かれている。例えば，Table 2 のSSL−Ⅱ の解法の欄にQL法とあるのは，ここでいう陰的QL法のことであり，すべて使用手引きで使われた術語がそのまま使われているので，詳しくは解法を示した文献を参照されたい。

このパッケージには実行列の標準問題を解くものとして，基本変換と直交変換との2つの系列

Table 4 Subroutines for eigenproblems in EISPACK-J

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJRNEN	Standard	Real	All	No	Stabil. elem. transf., QR
EJRNEC	"	"	"	Corresp.	"
EJRNES	"	"	"	Selected	Stabil. elem. transf., QR, inv. iter.
EJRNON	"	"	"	No	Orthog. transf., QR
EJRNOC	"	"	"	Corresp.	"
EJRNOS	"	"	"	Selected	Orthog. transf., QR, inv. iter.
EJRBEN	"	"	"	No	Balancing, stabil. elem. transf., QR
EJRBEC	"	"	"	Corresp.	"
EJRBES	"	"	"	Selected	Balancing, stabil. elem. transf., QR, inv. iter.
EJRBON	"	"	"	No	Balancing, orthog. transf., QR
EJRBOC	"	"	"	Corresp.	"
EJRBOS	"	"	"	Selected	Balancing, orthog. transf., QR, inv. iter.
EJRTNA	"	Real, tridiag. (T_n)	"	No	Diag. transf., QL/implicit QL

Table 4 (cont'd) - 1

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJRTNB	Standard	Real, tridiag. (T_n)	Bounded	No	Diag. transf., bisection
EJRTNI	"	"	Indexed	"	"
EJRTNE	"	"	Extreme	"	Diag. transf., rational QR
EJRTPA	"	Real, tridiag. (T_s)	All	Corresp.	Diag. transf., QL/implicit QL
EJRTPB	"	"	Bounded	"	Diag. transf., bisection, inv. iter.
EJRTPI	"	"	Indexed	"	"
EJRTPE	"	"	Extreme	"	Diag. transf., rational QR, inv. iter.
EJSNAN	"	Real, sym.	All	No	Orthog. transf., QL/implicit QL
EJSNAC	"	"	"	Corresp.	"
EJSNBN	"	"	Bounded	No	Orthog. transf., bisection
EJSNBC	"	"	"	Corresp.	Orthog. transf., bisection, inv. iter.
EJSNIN	"	"	Indexed	No	Orthog. transf., bisection
EJSNIC	"	"	"	Corresp.	Orthog. transf., bisection, inv. iter.

Table 4 (cont'd) - 2

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJSNEN	Standard	Real, sym.	Extreme	No	Orthog. transf., rational QR
EJSNEC	"	"	"	Corresp.	Orthog. transf., rational QR, inv. iter.
EJSPAN	"	"	All	No	Orthog. transf., QL/implicit QL
EJSPAC	"	"	"	Corresp.	"
EJSPBN	"	"	Bounded	No	Orthog. transf., bisection
EJSPBC	"	"	"	Corresp.	Orthog. transf., bisection, inv. iter.
EJSPIN	"	"	Indexed	No	Orthog. transf., bisection
EJSPIC	"	"	"	Corresp.	Orthog. transf., bisection inv. iter.
EJSPEN	"	"	Extreme	No	Orthog. transf., rational QR
EJSPEC	"	"	"	Corresp.	Orthog. transf., rational QR, inv. iter.
EJSBAN	"	Real, sym., band	All	No	Orthog. transf., QL/implicit QL
EJSBAC	"	"	"	Corresp.	"
EJSBBN	"	"	Bounded	No	Orthog. transf., bisection

Table 4 (cont'd) - 3

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJSBBC	Standard	Real, sym., band	Bounded	Corresp.	Orthog. transf., bisection, inv. iter.
EJSBIN	"	"	Indexed	No	Orthog. transf., bisection
EJSBIC	"	"	"	Corresp.	Orthog. transf., bisection, inv. iter.
EJSBEN	"	"	Extreme	No	Orthog. transf., rational QR
EJSBEC	"	"	"	Corresp.	Orthog. transf., rational QR. inv. iter.
EJSBNN	"	"	Nearest	No	QR
EJSBNC	"	"	"	Corresp.	QR, inv. iter.
EJSTAN	"	Real, sym., tridiag.	All	No	QL/implicit QL
EJSTAC	"	"	"	Corresp.	"
EJSTBN	"	"	Bounded	No	Bisection
EJSTBC	"	"	"	Corresp.	Bisection, inv. iter.
EJSTIN	"	"	Indexed	No	Bisection
EJSTIC	"	"	"	Corresp.	Bisection, inv. iter.
EJSTEN	"	"	Extreme	No	Rational QR

Table 4 (cont'd) - 4

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJSTEC	Standard	Real, sym., tridiag.	Extreme	Corresp.	Rational QR, inv. iter.
EJCNEN	"	Complex	All	No	Stabil. elem. transf., modified LR
EJCNEC	"	"	"	Corresp.	"
EJCNES	"	"	"	Selected	Stabil. elem. transf., modified LR, inv. iter.
EJCNUN	"	"	"	No	Unit. transf., QR
EJCNUC	"	"	"	Corresp.	"
EJCNUS	"	"	"	Selected	Unit. transf., QR, inv. iter.
EJCBEN	"	"	"	No	Balancing, stabil. elem. transf., modified LR
EJCBEC	"	"	"	Corresp.	"
EJCBES	"	"	"	Selected	Balancing, stabil. elem. transf., modified LR, inv. iter.
EJCBUN	"	"	"	No	Balancing, unit. transf., QR
EJCBUC	"	"	"	Corresp.	"
EJCBUS	"	"	"	Selected	Balancing, unit. transf., QR, inv. iter.

Table 4 (cont'd) - 5

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJHNAN	Standard	Hermite	All	No	Unit. transf., QL/implicit QL
EJHNAC	"	"	"	Corresp.	"
EJHNBN	"	"	Bounded	No	Unit. transf., bisection
EJHNBC	"	"	"	Corresp.	Unit. transf., bisection, inv. iter.
EJHNIN	"	"	Indexed	No	Unit. transf., bisection
EJHNIC	"	"	"	Corresp.	Unit. transf., bisection, inv. iter.
EJHNEN	"	"	Extreme	No	Unit. transf., rational QR
EJHNEC	"	"	"	Corresp.	Unit. transf., rational QR, inv. iter.
EJHPAN	"	"	All	No	Unit. transf., QL/implicit QL
EJHPAC	"	"	"	Corresp.	"
EJHPBN	"	"	Bounded	No	Unit. transf., bisection
EJHPBC	"	"	"	Corresp.	Unit. transf., bisection, inv. iter.
EJHPIN	"	"	Indexed	No	Unit. transf., bisection

Table 4 (cont'd) - 6

Engry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJHPIC	Standard	Hermite	Indexed	Corresp.	Unit. transf., bisection, inv. iter.
EJHPEN	"	"	Extreme	No	Unit. transf., rational QR
EJHPEC	"	"	"	Corresp.	Unit. transf., rational QR, inv. iter.
EJGRGN	Generalized	Real	All	No	Orthog. transf.
EJGRGC	"	"	"	Corresp.	"
EJGSAN	"	A,B: real, sym. B: pos.def.	"	No	Cholesky, orthog. transf., QL/implicit QL
EJGSAC	"	"	"	Corresp.	"
EJGSBN	"	"	Bounded	No	Cholesky, orthog. transf., bisection
EJGSBC	"	"	"	Corresp.	Cholesky, orthog. transf., bisection, inv. iter.
EJGSIN	"	"	Indexed	No	Cholesky, orthog. transf., bisection
EJGSIC	"	"	"	Corresp.	Cholesky, orthog. transf., bisection, inv. iter.
EJGSEN	"	"	Extreme	No	Cholesky, orthog. transf., rational QR

Table 4 (cont'd) - 7

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJGSEC	Generalized	A,B: real, sym. B: pos.def.	Extreme	Corresp.	Cholesky, orthog. transf., rational QR, inv. iter.
EJNABA	$ABx=\lambda x$	A,B: real A or B: pos. def.	All	No	Cholesky, orthog. transf., QL/implicit QL
EJNABB	"	"	Bounded	"	Cholesky, orthog. transf., bisection
EJNABI	"	"	Indexed	"	"
EJNABE	"	"	Extreme	"	Cholesky, orthog. transf., rational QR
EJVABA	"	"	All	Corresp.	Cholesky, orthog. transf., QL/implicit QL
EJVABB	"	"	Bounded	"	Cholesky, orthog. transf., bisection, inv. iter.
EJVABI	"	"	Indexed	"	"
EJVABE	"	"	Extreme	"	Cholesky, orthog. transf., rational QR, inv. iter.
EJSVDP	Sing. value decomp.	Degenerated, real			Householder, QR
EJSVDL	Least square fit	"			"

Table 4 (cont'd) - 8

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EJSVDG	Gen. inv.	Degenerated, real			Householder, QR
EJSVDM	Minmal norm solution	"			"
EJLERB	Linear equation	Real, band			Crout
EJLESB	"	Real, sym., band			Modified Cholesky
EJLERH	Homogeneous equation	Degenerated, real			Householder, QR

のものがある。これらの特徴はベンチマーク・テストで明らかにされるが，このパッケージを開発する段階では，機能や解法が少しでも変っていればすべて取り上げられた。このことは複素行列の場合の基本変換とユニタリ変換についても同じことが言える。EJSNAN とEJSPANは直交変換のあと，固有値を求めるのにQL法と陰的QL法のどちらかを選択できる。また，この2つのルーチンは表で見る限り機能も全く同じであるが，実は後者の場合対称行列の要素を半分だけ1次元配列に圧縮した形（packed form）で格納される。それだけ記憶容量が節約できるが，その反面使いにくいという短所もある。これらは実対称行列のほか，Hermite 行列のグループについても同様である。

今迄はあたかもEISPACK−J が一つのライブラリーのような述べ方をしてきた。しかし，利用や管理上の利点から，各ルーチンはTable 3 のBISCTD と同じ規準でJSSLに登録される。登録にあたり，付録1でJSSLの登録申込書の様式[12]による使用手引きが与えられる。また付録2では，各ルーチンに対して見本となる例題を，ファイル入力形式で実行する方法が示される。

8. その他の整備された副プログラム

SSLの拡充にあたり，第一段階として既に発表されたアルゴリズムやプログラムの調査が行われた。そのとき，最も注目したのがEISPACK−2 であるが，他にも変った解法や他のルーチンでは扱っていない系を解くものもあった。また，QR法の説明のとき述べたように，アルゴリズムは同じでもプログラムとしての性能は全く異なるということも多く，原研で簡単に入手できるルーチンは一応全部整備された。従って，この中にはベンチマーク・テストの結果により，他のルーチンにとって代わられると判断されるものがあるかも知れない。

これらをまとめたのがTable 5 である。EBERLD[8]はEberlein 法によるもので，固有値のほか，右または左のどちらかの固有ベクトルをすべて求める。べき乗法によるPOWERD[8]はアルゴリズムが簡単であるが，0に近い固有値は求めにくい。また，接近根の場合の2次因子の分離も難しいが，このルーチンの中では特別な工夫がなされている。[46] HEVALD[5)47)] は元々Hermite の系を実対称の系に還元して解くことを動機として開発されたが，一般の実対称の系も解けるようになっている。HDIAGD[48]は固有ベクトルの計算が選択できるようになっており，一般問題用のDIRNMD[48]の付属（slave）ルーチンとなっている。EIGNID[49]は固有ベクトルのために配列を余分に取らず，記憶容量低減のために開発されたものである。SIVI とCSIVI[50)51)] はそれぞれ実対称，Hermite 帯行列の一般問題を扱うルーチンであり，大次元を仮定して作業用ファイルを使っている。これらはすべてベンチマーク・テストに取り上げられ，他のルーチンと比較される。また，Eberlein 法は対象とする系によっては$\mathbf{P}$を交代行列として，$(a\mathbf{I}+\mathbf{P})$なる形の区画に収束することもある。[8] EBERLD ではこの処理が省かれており，SSLのルーチンとしての普遍性に欠けるため除かれるが，他はEISPACK−J のルーチンと同様JSSL[12]に登録される。

のものがある。これらの特徴はベンチマーク・テストで明らかにされるが，このパッケージを開発する段階では，機能や解法が少しでも変っていればすべて取り上げられた。このことは複素行列の場合の基本変換とユニタリ変換についても同じことが言える。EJSNAN とEJSPANは直交変換のあと，固有値を求めるのにQL法と陰的QL法のどちらかを選択できる。また，この2つのルーチンは表で見る限り機能も全く同じであるが，実は後者の場合対称行列の要素を半分だけ1次元配列に圧縮した形（packed form）で格納される。それだけ記憶容量が節約できるが，その反面使いにくいという短所もある。これらは実対称行列のほか，Hermite 行列のグループについても同様である。

今迄はあたかもEISPACK-J が一つのライブラリーのような述べ方をしてきた。しかし，利用や管理上の利点から，各ルーチンはTable 3 のBISCTD と同じ規準でJSSLに登録される。登録にあたり，付録1でJSSLの登録申込書の様式[12]による使用手引きが与えられる。また付録2では，各ルーチンに対して見本となる例題を，ファイル入力形式で実行する方法が示される。

8. その他の整備された副プログラム

SSLの拡充にあたり，第一段階として既に発表されたアルゴリズムやプログラムの調査が行われた。そのとき，最も注目したのがEISPACK-2 であるが，他にも変った解法や他のルーチンでは扱っていない系を解くものもあった。また，QR法の説明のとき述べたように，アルゴリズムは同じでもプログラムとしての性能は全く異なるということも多く，原研で簡単に入手できるルーチンは一応全部整備された。従って，この中にはベンチマーク・テストの結果により，他のルーチンにとって代わられると判断されるものがあるかも知れない。

これらをまとめたのがTable 5 である。EBERLD[8]はEberlein 法によるもので，固有値のほか，右または左のどちらかの固有ベクトルをすべて求める。べき乗法によるPOWERD[8]はアルゴリズムが簡単であるが，0に近い固有値は求めにくい。また，接近根の場合の2次因子の分離も難しいが，このルーチンの中では特別な工夫がなされている。[46] HEVALD[5)47)] は元々Hermite の系を実対称の系に還元して解くことを動機として開発されたが，一般の実対称の系も解けるようになっている。HDIAGD[48]は固有ベクトルの計算が選択できるようになっており，一般問題用のDIRNMD[48]の付属（slave）ルーチンとなっている。EIGN1D[49]は固有ベクトルのために配列を余分に取らず，記憶容量低減のために開発されたものである。SIVI とCSIVI[50)51)] はそれぞれ実対称，Hermite 帯行列の一般問題を扱うルーチンであり，大次元を仮定して作業用ファイルを使っている。これらはすべてベンチマーク・テストに取り上げられ，他のルーチンと比較される。また，Eberlein 法は対象とする系によっては$\mathbf{P}$を交代行列として，$(a\mathbf{I}+\mathbf{P})$なる形の区画に収束することもある。[8] EBERLD ではこの処理が省かれており，SSLのルーチンとしての普遍性に欠けるため除かれるが，他はEISPACK-J のルーチンと同様JSSL[12]に登録される。

Table 5 Adjusted subroutines for eigenproblems

Entry names	Problem	Matrix system	Eigenvalues	Eigenvectors	Methods
EBERLD	Standard	Real	All	Corresp. right/left	Eberlein
POWERD	"	"	Extreme	Corresp. right, left	Power
HEVALD	"	Real, sym.	All	No	Householder
HDIAGD	"	"	"	No/corresp.	Jacobi
EIGNID	"	"	"	Corresp.	Householder, QR
DIRNMD	Generalized	A,B: real, sym. B: pos. def.	"	"	Jacobi
SIVI	"	A,B: real, sym., band B: pos. def.	Nearest	"	Cholesky, simul. inv. iter.
CSIVI	"	A,B: band A: Hermite B: real, sym., pos. def.	"	"	"

9. ベンチマーク・テストと検討

固有値問題を解くルーチンに限られたことではないが，一般にＳＳＬのルーチンは次の4つの項目に特徴づけられる。

① 対象とする問題の型や系の性質

② 固有ベクトルの計算の有無や計算時の有効桁数

③ 基幹となる数値解法や細かい技法

④ 行列の入出力の仕方や作業領域の必要性

これらのうち1つでも異なれば，特徴あるルーチンということができるが，これらについて違いの見当らないルーチンでも，実際に問題を解いてみると，計算時間や精度に差が生じることが多い。

この章の目的は現有する公開されたルーチンについて，似た機能を有するルーチンに同じ問題を解かせて，計算に要した記憶容量，時間，それに解の精度によって各々の特徴を明らかにすることにある。この結果，使用者はそれぞれ自己の目的に合った特徴を持つルーチンを選び出すことができよう。また，ここでのベンチマーク・テストには限界があるが，このテストの方法は更に検討を加える場合の一つの指針となろう。

今回のテストでは，演算有効桁としては倍精度計算のルーチンのみが取り上げられる。これらを対象とする問題の型および系の性質によって分類したのがTable 6 であり，一次方程式関係のルーチンは除かれている。各項に属するルーチンの数え方は，コンポーネントになったルーチンを1つに数えるなど，これまでと同様である。この表にも現れている通り，実行列，実対称行列，それにHermite 行列の標準問題は需要が多いためか，それに比例してルーチンの数も多い。実対称3重対角行列や複素行列の標準問題，それに実対称行列の一般問題を解くルーチンがこれらに続くが，実対称帯行列の標準問題と変形された一般問題を解くルーチンはすべてEISPACK－Jのものであり，比較という見地からは興味が薄い。

以上の状況から，異なる系統のライブラリーのものが含まれる項目に対しては1件以上，また使用頻度も高く，多数のルーチンが含まれる項目に対しては2～3の問題を用意してベンチマーク・テストが実施される。このとき，同等に比較するには多少無理があるが，参考のために類似する項目のルーチンが加えられることもある。テスト問題としては，なるべく原研所内で起る実際の問題が取り上げられるか，それが無い項目については理論的に行列の性質が分っているものが採用される。

テストの結果は

① 主プログラムでとる配列の大きさとそのルーチンおよび付属ルーチンの大きさ（単位　語）[52]

② そのルーチンに入ってから出るまでの時間（単位　ミリ秒）

③ 固有ベクトルもr 個求めたとき，

$$\max_{1 \le j \le r} \frac{\|\mathbf{A}\mathbf{x}_j - \lambda_j \mathbf{x}_j\|_1}{\|\mathbf{A}\|_1 \|\mathbf{x}_j\|_1} \qquad (88)$$

Table 6 Number of subroutines contained in the same type of problem and the same matrix system

Problem	Matrix system	Number of subroutines
Standard	Real	24
"	Real, tridiag. $(T_n)^\dagger$	4
"	Real, tridiag. $(T_s)^\dagger$	4
"	Real, tridiag. $(T_p)^\dagger$	1
"	Real, sym.	26
"	Real, sym., band	10
"	Real, sym. tridiag.	12
"	Complex	13
"	Hermite	23
Generalized	Real	2
"	A,B: real, sym. B: pos. def.	10
"	A,B: real, sym., band B: pos. def.	1
"	A,B: band A: Hermit B: real, sym., pos. def.	1
$ABx = \lambda x$	A,B: real, sym. A or B: pos. def.	8
Sing. value decomp.	Degenerated, real	1
Gen. inv.	"	2
Least squares fit	"	1
Minmal norm solution	"	1

† The property "T_n" or "T_p" means that products of corresponding pairs of off-diagonal elements become all non-negative or positive, respectively. Similarly, the property "T_s" means in addition to the property T_n that the products become zero only when both elements are zero.

による精度

の3つについて比較される。但し，③の精度は標準問題の場合の評価式であり，L_2ーノルムなどを使うこともできる。これは他の問題の場合，類似の式に置き換えられるが，何れの場合も固有値のみしか計算しない場合は評価できないことになる。

また，細かな事項として，

① 2次の区画から固有値をとり出すなど，簡単な変数の置き換え

② 求まった固有値の大小順による並べ替え

③ 固有ベクトルの正規化

④ エラー・チェックや後処理

の有無などにより，記憶容量や計算時間に多少の差が出ることに注意する必要がある。

以下，各テストの問題と結果について検討しよう。

Problem SR−1 これは標準問題の，実行列の系に対する最初のテスト問題で，SR－1という識別番号が付けてある。たちの悪い連立一次方程式の系として有名なLotkin の行列[3)53)54)] $\mathbf{A}$ は

$$\mathbf{A}=(a_{ij})=1 \quad (i=1,\ j=1\sim n)$$
$$=\frac{1}{i+j-1} \quad (i=2\sim n,\ j=1\sim n) \tag{89}$$

で与えられるが，固有値問題としては1つの相対的に小さい固有値を持つという意味で正確に解きにくい。ここでは6次の場合を取り上げ，規格化の効果をみるために更に第3行と第3列に10^6が掛けられる。

この計算結果を表にしたのがTable 7である。実行列の標準問題を解くルーチンは多いため，固有ベクトルの計算の有無，規格化，得意とする問題の型などにより組分けし，比較を容易にしている。参考のため，各ルーチンが属するライブラリーが略号で示されているが，EISPACK−JのものはJSSLに入れてある。

これらのルーチンのうち，固有ベクトルの正規化が保証されているものはDHVEC を含んだものとPOWERD である。注釈の欄は，主に各ルーチンで選択されたパラメーターの値であり，必要に応じてその概要が説明されるが，ルーチンごとに意味が異なるので，詳しくは各ルーチンの使用手引きを参照されたい。解法はライブラリーごとの表で示されているので，ここでは省略されている。

この問題では，固有値はすべて求め，固有ベクトルはすべて求めるか，すべて求めないかのどちらかになっており，最後の組では左固有ベクトルも計算されている。

固有ベクトルを計算しない組についてみると，規格化を行う方は多少とも記憶容量も大きく，計算時間も長い。EJRNBN だけ大きいのはQR法のプログラムが大きいためで，6次くらいの問題では，主プログラムでとられる配列の大きさは殆んど無視される。収束の判定はDABADの場合，Newton 法の反復式(19)式に還元して$|\mu_{i+1}-\mu_i|<$EPS で行われるが，SSL−Ⅱ[11)]やJSSL[12)]のルーチンはHessenberg 形に変換した後，(55)式の反復を30回行うまでに所望の精度[13)]に達したときに収束したと判定される。QREGND では独自の収束判定を行っている。[9)10)]

固有ベクトルを求める組についても，規格化の影響については同じようなことが言える。固有

Table 7 Computational results for Problem SR-1
(modified Lotkin's matrix of order 6)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy	Comment
All values, no vectors, no balancing	H	DABAD	2210	6		EPS=10^{-16}
	II	DHES1 DHSQR	2030	7		
	J	EJRNEN	1836	4		
	J	EJRNON	1900	5		
All values, no vectors, balancing	H	QRENGD	2814	6		
	II	DBLNC DHES1 DHSQR	2548	7		
	J	EJRBEN	5900	5		
	J	EJRBON	2492	6		
All values, corresp. vectors, no balancing	H	DANEWD	4626	13	1.6×10^{-11}	EPS=10^{-14}
	II	DEIG1	4710	10	2.4×10^{-19}	
	J	EJRNEC	3782	8	6.4×10^{-19}	
	J	EJRNOC	3886	9	4.9×10^{-19}	
All values, corresp. vectors, balancing	H	HESQRD	7264	15	4.9×10^{-19}	
	II	DEIG1	4710	11	4.9×10^{-19}	
	J	EJRBEC	6000	8	5.2×10^{-19}	
	J	EJRBOC	4740	10	3.1×10^{-19}	
All values, corresp. vectors (S-type)	II	DHES1 DHSQR DHVEC DHBK1	5732	11	1.0×10^{-18}	
	II	DBLNC DHES1 DHSQR DHVEC DHBK1	6250	12	1.1×10^{-18}	
	J	EJRNES	4918	8	1.4×10^{-19}	
	J	EJRNOS	4864	11	1.2×10^{-18}	

Table 7 (cont'd)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy	Comment
	J	EJRBES	5772	9	3.9×10^{-19}	
	J	EJRBOS	5844	10	1.1×10^{-18}	
All values, corresp. vectors (right/left)	H	QREGND GAVE2D	9354	28	6.3×10^{-19}	
	J	POWERD	4006	120	2.2×10^{-1}	E-type, L=14, LIM=200, IMAG=50
		EBERLD	1324	76	2.8×10^{-12}	EP=10^{-18}

ベクトルを使って(88)式により評価した精度をみると，DANEWD が著しく悪いのは，規格化というよりもむしろDanilevsky 法という解法の所為だろう。規格化は対応する行と列の要素の和を等しくするよう行われるので，むしろ問題としてはLotkin の行列の第3行だけを小さくした方が良かったかも知れない。EISPACK－J の基本変換と直交変換によるものの違いについては，一般に「前者が後者より速いが，ある場合において後者がより精度が良い」と言われている。[13)31)] このことは，このテストにも現れており，基本変換の方がより標準ルーチン向きと言えよう。DEIG1 は2箇所に出ているが，これは規格化を選択できるためである。

S－型の特殊問題を解くルーチンの組では，固有値を求めたあと，すべての固有ベクトルを計算するよう指定されている。このような場合のベクトルの計算は，SSL－Ⅱ，JSSL のルーチンとも，(71)式における反復を最大5回まで行う。[13)] DBLNC は規格化を行うルーチンであり，この組の中を，更に規格化の有無に注目して見ることができる。この組の中での際立った差は見られないが，初めからすべての固有ベクトルを求めることが分っている場合，S－型のルーチンよりも完全問題用のルーチンを使った方が有利であることが明白である。このテストでは，例えばEJRNOC とEJRNOS というふうに計算時間を比べてみると分る。

左固有ベクトルも求める組は当然ながら計算時間は長い。EBERLD では一度の呼出しで左または右固有ベクトルしか求められないので，2度呼ばれており，区画化対角行列に変換する計算が2重に行われている。また，これらの精度は，左固有ベクトルを使った精度と右固有ベクトルを使った精度の大きい方が採られている。POWERD はE－型の特殊問題のために開発されたルーチンであり，べき乗法のため最も小さい固有値の精度が悪くなっている。このため，この固有値を除いて

$$\max\left\{\max_{1\leq j\leq 5}\frac{\|\mathbf{A}\mathbf{x}_j-\lambda_j\mathbf{x}_j\|_1}{\|\mathbf{A}\|_1\|\mathbf{x}_j\|_1},\ \max_{1\leq j\leq 5}\frac{\|\mathbf{y}_j^{\dagger}\mathbf{A}-\lambda_j\mathbf{y}_j^{\dagger}\|_1}{\|\mathbf{A}\|_1\|\mathbf{y}_j\|_1}\right\} \tag{90}$$

で精度をみると 3.8×10^{-14} まで改善される。この問題に対しては，固有値の精度14桁，最大反復回数200，複素根の検査の間隔50回で計算されているが，計算時間が長いのは次数の切り下げを行っていないことのほか，計算の経過を示すプリントが多数出ることにもよる。しかし，Eberlein 法と同様アルゴリズムが簡単なため，プログラムは小さい。

Problem SR－2　これは最適制御則の決定の際，Ricatti の方程式を解くときに現れてくる問題である。付録3で示されるように，動特性システム行列の次数は　であるが，システムの安定度を示す固有値は異符号の組となる。[46)]

ここでは，実行列の標準問題を解くルーチンのうち，先のProblem SR－1の結果を考慮して，組分けしたルーチンの代表的なものが取り上げられる。ここでいう「代表的」とは，精度がよく，記憶容量や計算時間についてあまり難点が無いもの，各ライブラリーを代表するもの，解法が特徴的であったり，なるべく多くの技法を用いているもの，変った機能のもの等である。同種の系に対して2件以上の問題が用意される場合，今後も同様な選択が行われる。

Table 8 は，Problem SR－2 に付き，ルーチンを選んでテストした結果である。固有ベクトルを求めない組でみると，前の問題よりも系の次数が高くなったため，EJRBEN の大きさが目

Table 8 Computational results for Problem SR-2
(Ricatti's equation of order 16)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy	Comment
All values, no vectors	J	EJRNEN	2336	46		
	H	QRENGD	3734	62		
	II	DBLNC DHES1 DHSQR	3068	95		
	J	EJRBEN	6420	58		
All values, corresp. vectors	J	EJRNEC	4722	88	5.0×10^{-19}	
	H	HESQRD	8624	179	5.8×10^{-15}	
	II	DEIG1	5650	141	4.8×10^{-18}	MODE≠1
	J	EJRBEC	6960	110	2.4×10^{-19}	
	J	EJRBOC	5700	130	2.4×10^{-18}	
	II	DBLNC DHES1 DHSQR DHVEC DHBK1	7750	133	7.5×10^{-16}	S-type
	J	EJRBES	7672	90	4.2×10^{-17}	S-type
All values, corresp. vectors (right/left)	H	QREGND GAVE2D	12514	315	2.8×10^{-15}	
	J	POWERD	6386	1391	1.8×10^{-6}	E-type, L=8, LIM=200, IMAG=50
		EBERLD	2204	3172	2.1×10^{-7}	EP=10^{-18}

立たなくなっている。またSSL－Ⅱのルーチンは一部精度を上げて計算しているため，はっきりと時間が長くなっている。

固有ベクトルを求めるものは，SSL－Hのルーチンが10^{-15}の次数，他は10^{-16}から10^{-19}の次数で精度良く求まっている。DEIG1は規格化が行われた場合である。この例におけるPOWERDは，2つまでなら10^{-14}の次数で求まっている。また，EBERLDは2つを除いて10^{-18}まで求まっているが，計算時間がかかり過ぎているのは，収束条件が厳し過ぎたこともあろう。

Problem SR－3　これは原子炉物理におけるアクティノイドの消滅の問題[55]である。この系の要素は付録4で与えられるが，固有値の大きさは10^{-5}から10^{-9}の範囲に亘り，うち4つが複素固有値である。

Table 9を見ると，やはりSSL－Ⅱのルーチンは，一部精度を上げて計算しているため，計算時間が長い。固有ベクトルの計算では，系が22次となったためか，概して精度が下っている。また，HESQRDおよびS－型のルーチンでは$6n^2$語以上の配列を必要とするため，プログラムの大きさに比べ，配列の大きさの比率が意味をもつ程大きくなっていることが分る。左右の固有ベクトルを求める組をみると，POWERDでは右ベクトルと左ベクトルの固有値が一致せず計算が打ち切られている。EBERLDでは，複素根の虚数部が10^{-9}以下で，これを実根として取り出したため精度が低く評価されている。

Problem SRT　実非対称3重対角の系を解くルーチンは少なく，すべてのルーチンの対象となる性質T_pをもつ系が取り上げられる。この系[13]は付録5で与えられるように15次であり，固有値はすべて実数である。

計算結果はTable 10に示されるが，BISCTDのみ既存のルーチンであり，I－型であるため，主にこの系統のものと比較される。EJRTPAは固有ベクトルが正規化されている。

BISCTDは配列を少なく抑えてあり，記憶容量は小さいが，計算時間が長い。EPS 1はどのルーチンにおいても誤差の許容範囲を表わすものであり，0のときは標準値が採られる。特に2分法では余り小さくとり過ぎると，余分な分割を行うので注意を要する。完全問題用のルーチンでは，どちらも陰的QL法が選択されており，計算時間は他のルーチンよりはっきりと短くなっている。

Problem SRS－1　実対称の標準問題を解くルーチンは多く，2つの問題が用意されるが，これはその1つである。系としては，8次であるRosserの行列[53)54)]が選ばれる。これは次の問題と同様厳密解が分っているので，固有値自身の精度を確かめることもできるが，ここでは省略される。Rosserの行列は

$$\mathbf{A}=\begin{pmatrix} 611 & 196 & -192 & 407 & -8 & -52 & -49 & 29 \\ & 899 & 113 & -192 & -71 & -43 & -8 & -44 \\ & & 899 & 196 & 61 & 49 & 8 & 52 \\ & & & 611 & 8 & 44 & 59 & -23 \\ & & & & 411 & -599 & 208 & 208 \\ & & 対 & 称 & & 411 & 208 & 208 \\ & & & & & & 99 & -911 \\ & & & & & & & 99 \end{pmatrix} \tag{91}$$

Table 9 Computational results for Problem SR-3 (incineration of 22 actinoids)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy	Comment
All values, no vectors	J	EJRNEN	2828	108		
	H	QRENGD	4670	156		
	II	DBLNC DHES1 DHSQR	3572	211		
All values, corresp. vectors	J	EJRNEC	5670	223	9.8×10^{-19}	
	H	HESQRD	10016	433	3.4×10^{-15}	
	II	DEIG1	6598	313	1.0×10^{-15}	MODE≠1
	J	EJRBEC	7920	204	3.1×10^{-16}	
	J	EJRBOC	6660	113	1.5×10^{-15}	
	II	DBLNC DHES1 DHSQR DHVEC DHBK1	9226	296	2.4×10^{-13}	S-type
	J	EJRBES	9580	175	1.9×10^{-14}	S-type
All values, corresp. vectors (right/left)	H	QREGND GAVE2D	15754	768	1.0×10^{-12}	
	J	POWERD	8774	346	(Not obtained)	E-type, L=10, LIM=200, IMAG=50
		EBERLD	3116	3106	2.5×10^{-5}	EP=10^{-18}

Table 10 Computational results for Problem SRT (a certain tridiagonal matrix of order 15 which has property T_p)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy $\max_{1\leq j\leq n} \frac{\|Ax_j-\lambda_j x_j\|_1}{\|A\|_1\|x_j\|_1}$	Comment
All values, no vectors	J	BISCTD	660	120		I-type, EPS1=10^{-16}
	J	EJRTNA	2700	17		IMP=1
	J	EJRTNI	1632	73		I-type, EPS1=0
All values, corresp. vectors	J	EJRTPA	3272	43	4.8×10^{-18}	IMP=1
	J	EJRTPI	3604	83	4.3×10^{-17}	I-type, EPS1=0

で与えられ，$p=\sqrt{10405}$，$q=\sqrt{26}$ に対して

$$\begin{aligned}\lambda_1 &= -\lambda_8 = 1020.04901843 = 10p\\ \lambda_2 &= 1020\\ \lambda_3 &= 1019.90195136 = 510+100q\\ \lambda_4 &= \lambda_5 = 1000\\ \lambda_6 &= 0.09804864072 = 510-100q\\ \lambda_7 &= 0\end{aligned} \tag{92}$$

という固有値をもつ。この固有ベクトルの厳密解も同様に与えられる。

実行列のときと同様，最初はすべてのルーチンがテストされ，その結果がTable 11に示される。実対称行列の場合，系の要素の半分だけを与えれば良いので，固有ベクトルの計算の有無とこのような圧縮型になっているかどうかで組分けされる。SSL・Hのルーチンは非圧縮型，SSL－Ⅱのルーチンは圧縮型に統一されているが，EISPACK－J のルーチンには双方がある。また，固有値が実数となるため，特殊問題のB－型のルーチンも登場する。

まず，計算条件では，完全問題用のEJSNAN などはすべて陰的QL法が選ばれている。特殊問題用のルーチンの収束条件の多くは零としてあるが，これは標準値に置き換えられるので，ほぼ $10^{-16}\sim10^{-18}$ くらいとみてよかろう。B－型ではすべての固有値の存在する範囲を与えなければならないが，これは前以って知ることはできないので，一応，区間（10^{-10}，10^{10}）が与えられる。E－型ではSSL・Hのルーチンが元々大きい順になっているほかは，すべて小さい順に求めるよう指定される。また，EJSNEN などの場合，定値性が利用できるようになっているが，ここでは不明であるとして計算される。

表から固有値のみの計算をみると，8次のため，わずかながら圧縮型のルーチンが対応するルーチンに比べて記憶容量が小さくなっている。これは高次になると，n^2 対 $n^2/2$ の割合で効いてくるので，なるべく圧縮型を使うことが望ましい。この場合，使われるアルゴリズムは殆んど同じなので，計算時間に差は出ない。HOUSD とHOUS 2Dの記憶容量が大きいのは，プログラムが大きいためである。EIGN1Dは固有ベクトルを，初めに与える行列**A** の場所に格納するため記憶容量が節約されており，より高次の問題になると，他の圧縮型のルーチンより，はっきりと小さくなる筈である

次に，完全問題用と特殊問題用のルーチンを計算時間について比較してみよう。例えば，QL法や陰的QL法はすべての固有値を求める場合，2分法より約25%速い[13)36)]と言われているが，ここでは3～4倍の差が出ている。この点，HDIAG は時間がかかり過ぎているし，EIGN1D は精度もいま一歩と言えよう。

Problem SRS－2　Peiの行列 $\mathbf{A}=(a_{ij})$ は

$$\begin{aligned}a_{ij} &= d \quad (i=j)\\ &= 1 \quad (i\neq j)\end{aligned} \tag{93}$$

で定義される[53)54)]が，ここでは $d=1+10^{-5}$ で正定値であり，24次とする。この固有値は

$$\begin{aligned}\lambda_i &= d-1 \quad (i=1\sim n-1)\\ &= d+n-1 \quad (i=n)\end{aligned} \tag{94}$$

Table 11 Computational results for Problem SRS-1
(Resser's matrix whose order is 8)

Test	Library	names	Core (words)	Time (msec)	Accuracy $\max_{i\leq j\leq n} \frac{\|Ax_j-\lambda_j x_j\|_1}{\|A\|_1\|x_j\|_1}$	Comment
All values no vectors, no packed form	H	HOUSD	2962	27		E-type
	J	HDIAGD	1454	48		
	J	HEVALD	1560	4		
	J	EJSNAN	1848	4		IMP=1
	J	EJSNBN	1826	22		B-type, LB=10^{-10}, UB=10^{10}, EPS1=0
	J	EJSNIN	1922	22		I-type, EPS1=0
	J	EJSNEN	1720	17		E-type, TYPE=.TRUE., IDEF=0, EPS1=0
All values, no vectors, packed form	II	DTRID1 DTRQL	1234	6		
	II	DTRID1 DBSCT1	1964	24		E-type, M=-n, EPST=10^{-16}
	J	EJSPAN	1572	5		IMP=1
	J	EJSPBN	1548	22		B-type, LB=10^{-10}, UB=10^{10}, EPS1=0
	J	EJSPIN	1648	22		I-type, EPS1=0
	J	EJSPEN	1442	17		E-type, TYPE=.TRUE., IDEF=0, EPS1=0

Table 11 (cont'd)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy $\max_{i\leq j\leq n} \frac{\|Ax_j-\lambda_j x_j\|_1}{\|A\|_1\|x_j\|_1}$	Comment
All values, all vectors, no packed form	H	JACOBD	1732	24	7.1×10^{-16}	EPS=10^{-14}
	H	HOUS2D	4496	32	1.0×10^{-18}	E-type
	J	HDIAGD	1582	53	1.5×10^{-18}	
	J	EIGN1D	1626	14	3.6×10^{-13}	RHO=10^{-18}
	J	EJSNAC	3860	8	1.5×10^{-18}	IMP=1
	J	EJSNBC	3110	26	1.3×10^{-17}	B-type, LB=10^{-10}, UB=10^{10}, EPS1=0
	J	EJSNIC	3142	27	1.3×10^{-17}	I-type, EPS1=0
	J	EJSNEC	3392	22	8.4×10^{-18}	E-type, TYPE=.TRUE., IDEF=0, EPS1=0
All values, corresp. vectors, packed form	II	DSEIG1	2104	10	9.9×10^{-19}	
	II	DSEIG2	4032	29	1.2×10^{-17}	E-type, M=-n,
	J	EJSPAC	3018	9	1.6×10^{-18}	IMP=1
	J	EJSPBC	2776	27	8.0×10^{-18}	B-type, LB=10^{-10}, UB=10^{10}, EPS1=0
	J	EJSPIC	2844	27	8.0×10^{-18}	I-type, IPS1=0
	J	EJSPEC	4358	22	7.1×10^{-18}	E-type, TYPE=.TRUE., IDEF=0, EPS1=0

で与えられ，固有ベクトルの厳密解も分っている。

Table 12 は，この計算結果である。EISPACK−J 以外にB−型とI −型のルーチンが無いため，これらの型のものが除かれている。計算条件は前回と殆んど変わらないが，EISPACK−JのE−型のルーチンにより，正定値性を利用せずに，小さい順にすべての固有値と固有ベクトルを計算したときの収束が悪かった。これは小さい根が重根のためか，固有ベクトルを求めない場合の計算時間にも現れている。このため，固有ベクトルも求めるとき，大きい順に求めてある。特に，EJSNEC のときは解が求まっていない。

この例でもEIGN1Dは記憶容量が少ないが，計算時間はやはり長い。しかし，前の問題で計算時間の長かったHDIAGDは，例えばHEVALDやEJSNAC と比べると，逆に短かくなっている。また，EISPACK−J のルーチンも，前の問題では計算時間が短かったにも拘らずこの問題では概して長くなっている。このように，各ルーチンは問題により結構変動することが伺えよう。

Problem SRST　n 次のEberlein の行列$\mathbf{A}=(a_{ij})$の一般式は

$$\begin{aligned} a_{ii} &= -\{(2i-1)(n-1)+(i-1)s+2(i-1)^2\} \quad (i=1\sim n) \\ a_{i-1,i} &= (i-1)(n+1+s-i) \quad (i=2\sim n) \\ a_{i,i-1} &= (i-1)(n+1-i) \quad (i=2\sim n) \end{aligned} \tag{95}$$

である[53]。対称にするためにs ＝0 とし，n＝40 とする。この場合の固有値は

$$\lambda_j = -(j-1)(s+j) \quad (j=1\sim n) \tag{96}$$

であり，固有ベクトルもやはり解析的に与えられる。

この計算結果がTable 13 である。ここでは完全問題と特殊問題のE−型のルーチンがテストされる。E−型ではどれも小さい順に固有値が計算されているが，P_{ei} の行列のときとは逆にEISPACK−J のルーチンの方が，計算時間が短かくなっている。DTRQL はEJSTAN より記憶容量が小さくなっているが，そのうちの配列の大きさは，前者が6n語に対し後者が4n語である。この現象は，後者が多数の付属ルーチンを組合わせて作られたため，その制御のための余分な命令によるもので，アルゴリズムが複雑なためではない。

この問題の固有値は，(96)式をみると明らかなように整数なので，固有値自身の絶対誤差を調べることが可能である。表の中の精度の欄は，固有ベクトルを求めるときは従来通り(88)式による相対誤差を，固有ベクトルを求めないときは，固有値が何桁まで合っていたかの最小数で示す。これは少数例にすぎないが，やはりSSL−Ⅱのルーチンが「固有値だけは高精度で計算している」ことの一つの現れといえよう。

Problem SC　標準問題で一般の複素の系の例は実例が少ない。これもテスト行列のテキスト[53]から引用されたWilkinson の行列であり，Hessenberg 形で10次である。各要素の具体的な数字は付録6で与えられるが，固有値の大きさも揃っており，近接根もなく，固有値問題としては良条件である。

Table 14 に見られるように，ここでは固有ベクトルも求める完全問題のルーチンばかりがテストされる。良条件の問題のためか，どれも精度良く求まっている。このうち，CHSQRD とEJCNEC，EJCNUC が規格化を行っていない。ここでも，基本変換の方が速くなっているが，精度は若干ながらユニタリ変換の方が良いように見える。[13)31)]

Table 12 Computational results for Problem SRS-2
(Pei's matrix whose order is 24)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy $\max_{i\leq j\leq n} \frac{\|Ax_j-\lambda_j x_j\|_1}{\|A\|_1\|x_j\|_1}$	Comment
All values, no vectors, no packed form	H	HOUSD	3986	26		E-type
	J	HDIAGD	2542	32		
	J	HEVALD	2744	39		
	J	EJSNAN	2936	28		IMP=1
	J	EJSNEN	2872	430		E-type, TYPE=.TRUE., IDEF=0, EPS1=0
All values, no vectors, packed form	II	DTRID1 DTRQL	1858	14		
	II	DTRID1 DBSCT1	2684	18		E-type, M=-n, EPST=10
	J	EJSPAN	2164	34		IMP=1
	J	EJSPEN	2098	438		E-type, TYPE=.TRUE., IDEF=0, EPS1=0
All values, corresp. vectors, no packed form	H	JACOBD	3780	30	7.4×10^{-19}	EPS=10^{-14}
	H	HOUS2D	6544	88	1.0×10^{-18}	E-type
	J	HDIAGD	3694	37	1.2×10^{-18}	
	J	EIGN1D	2810	144	1.6×10^{-17}	RHO=10^{-18}
	J	EJSNAC	5972	63	1.7×10^{-18}	IMP=1
	J	EJSNEC	5792	422	(Not obtained)	E-type, TYPE=.FALSE., IDEF=0, EPS1=0

Table 12 (cont'd)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy $\max_{1\leq j\leq n} \frac{\|Ax_j-\lambda_j x_j\|_1}{\|A\|_1\|x_j\|_1}$	Comment
All values, corresp. vectors, packed form	II	DSEIG1	3752	21	3.3×10^{-18}	
	II	DSEIG2	5840	29	2.2×10^{-18}	E-type, M=-n,
	J	EJSPAC	4634	72	5.7×10^{-19}	IMP=1
	J	EJSPEC	6262	467	1.2×10^{-18}	E-type, TYPE=.FALSE., IDEF=0, EPS1=0

Table 13 Computational results for Problem SRST
(Eberlein's matrix of order 40)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy	Comment
All values, no vectors	II	DTRQL	708	95	16	
	II	DBSCT1	1630	516	17	E-type, M=-n, EPST=0
	J	EJSTAN	1940	95	16	IMP=1
	J	EJSTEN	1216	197	15	E-type, TYPE=.TRUE., IDEF=0, EPS1=0
All values, corresp. vectors	II	DTEIG1	4096	478	3.5×10^{-18}	
	II	DTEIG2	6334	549	3.4×10^{-17}	E-type, M=-n
	J	EJSTAC	5380	517	4.4×10^{-18}	IMP=1
	J	EJSTEC	6104	282	1.3×10^{-16}	E-type, TYPE=.TRUE., IDEF=0, EPS1=0

Table 14 Computational results for Problem SC
(Wilkinson's matrix of order 16)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy $\max_{1\leq j\leq n} \frac{\|Ax_j-\lambda_j x_j\|_1}{\|A\|_1\|x_j\|_1}$	Comment
All values, corresp. vectors	H	CHSQRD	6660	288	9.9×10^{-19}	
	J	EJCNEC	4462	89	1.3×10^{-18}	
	J	EJCNUC	5024	132	2.6×10^{-19}	
	J	EJCBEC	5570	103	1.2×10^{-17}	
	J	EJCBUC	7670	135	4.1×10^{-19}	

Problem SH−1　Hermite の標準問題は2つ与えられるが，これは手初めのMuller の行列であり，5次である。[54] 行列の要素は付録7で示されるが，この固有値はおよそ−15.9，−5.1，−0.8，5.6，15.2というように，絶対値が比較的近い値になっている。

この計算結果はTable 15に示される。実対称のときのように，SSL・Hのルーチンは非圧縮型になっており，すべて固有ベクトルも計算する。従って，全体を3組に分けてあるが，このうち計算条件で変っているのは，EJSPACK−J の完全問題用のルーチンでQL法も選択されていること，Greenstadt 法のHERMTD では収束条件が甘くとられていること，それに特殊問題のB−型のルーチンの範囲が（-10^5，10^5）と指定されていることである。記憶容量をみると幾分差があるが，もう少し高次の問題になると，また変った結果が出る可能性が強い。計算時間は完全問題用のルーチンが短く，それなりに傾向が現れていると言える。しかし，QL法と陰的QL法の違いは，この例では判断できない。HERMTD は収束解が得られにくく，収束判定条件も甘くされた。それだけに精度は劣るが，計算も速く終っている。その点，THJACD は高精度ではあるが，幾分計算時間が長いと言えよう。

Problem SH−2　この問題はα−マンガンの結晶中のd−電子のエネルギーの固有値を求める問題である。[56] 固有値の精度は3〜5桁，固有ベクトルのそれは2〜3桁必要であり，少なくとも2，3の小さい固有値，できればすべての固有値と固有ベクトルを求めたい。系は密なHermite行列であるが，正定値とは限らず，近接固有値も多い。次数も145次，290次，580次と拡大したい。

これらの要求に沿って解かれたものの一部がTable 16 であるが，実はこれには以下のような経緯があった。まず，精度の要求がそう高くないこともあって，古くからあるSSL・HのGreenstadt 法のルーチン HERMTD が使われた。しかし，これでは収束解が得られなかったので，Hermite の系を実対称の問題に還元し，それからHouseholder 法で固有値を求めるHEVALDが開発された。即ち，(1)式で$\mathbf{A}=\mathbf{P}+i\,\mathbf{Q}$とし，$\lambda_j$に対応する固有ベクトルを$\mathbf{x}_j=\mathbf{r}_j+i\,\mathbf{s}_j$（$j=1\sim n$）とすれば，(1)式は2n次の実対称な系

$$\begin{pmatrix}\mathbf{P} & -\mathbf{Q}\\ \mathbf{Q} & \mathbf{P}\end{pmatrix}\mathbf{t}=\mu\begin{pmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{pmatrix}\mathbf{t} \tag{97}$$

に移る。ここに，λ_jは実数で，$\mu_{2j-1}=\mu_{2j}=\lambda_j$，$\mathbf{t}_{2j-1}=\begin{pmatrix}\mathbf{r}_j\\ \mathbf{s}_j\end{pmatrix}$，$\mathbf{t}_{2j}=\begin{pmatrix}-\mathbf{s}_j\\ \mathbf{r}_j\end{pmatrix}^{(j=1\sim n)}$

となる。表では，HEVALDのみこのように倍の次数で解かれており，対称性も利用せず，かつ圧縮型でもないので，記憶容量，計算時間とも大きくなっているが，一応固有値だけは得られた。

このような状況の後で，SSL−ⅡやEISPACK−J が開発され，すべての固有値や最小の固有値とその固有ベクトルを計算した結果，精度や計算時間等の問題も無くなったので，すべての固有値と固有ベクトルを計算するに至った。1組の固有値と固有ベクトルしか求めないとき，その精度は88)式で$r=1$として計算されるが，このような典型的なE−型の問題では，特に高次のとき，完全問題用のルーチンよりE−型のルーチンの方が，記憶容量や計算時間において相当節約できることが伺えよう。

Table 15 Computational results for Problem SH-1
(Mueller's matrix of order 5)

Test	Library	Entry names	Core (words)	Time (msec)	Accuracy $\max_{1\leq j\leq n} \frac{\|Ax_j-\lambda_j x_j\|_1}{\|A\|_1\|x_j\|_1}$	Comment
All values, no vectors, packed form	II	DTRIDH DTRQL	2378	6		
	II	DTRIDH DBSCT1	3090	15		E-type, M=-n, EPST=10^{-16}
	J	EJHPAN	2131	3		IMP=1
	J	EJHPAN	2131	3		IMP=0
	J	EJHPBN	1694	13		B-type, LB=-10^5,UB=10^5, EPS1=0
	J	EJHPIN	2298	13		I-type, EPS1=0
	J	EJHPEN	2120	4		E-type, TYPE=.TRUE., IDEF=0, EPS1=0
All values, corresp. vectors, no packed form	H	HERMTD	1222	13	2.8×10^{-10}	EPS=10^{-8}
	H	THJACD	2202	49	2.9×10^{-16}	EPS=10^{-14}
	H	HMTQRD	4410	9	5.2×10^{-19}	
	J	EJHNAC	4196	6	3.3×10^{-19}	IMP=1
All values, corresp. vectors, packed form	II	DHEIG2	5382	19	4.0×10^{-19}	E-type, M=-n
	II	DTRIDH DTEIG1 DTRBKH	3302	8	4.6×10^{-19}	
	J	EJHPAC	3142	6	3.3×10^{-19}	IMP=1
	J	EJHPAC	3142	6	4.8×10^{-19}	IMP=0
	J	EJHPBC	3982	16	6.6×10^{-19}	B-type, LB=-10^5,UB=10^5, EPS1=0
	J	EJHPIC	4074	16	6.6×10^{-19}	I-type, EPS1=0
	J	EJHPEC	3948	7	2.1×10^{-18}	E-type, TYPE=.TRUE., IDEF=0, EPS1=0

Table 16 Computational results for Problem SH-2
(energy matrix for electron, order 145)

Test	Library	names	Core (words)	Time (msec)	Accuracy	Comment
All values, no vectors	II	DTRIDH DTRQL	45778	22737		
	J	EJHPAN	45296	16095		IMP=1
	J	HEVALD	172472	47636		Real sym. version, no packed form
Smallest value, corresp. vector (E-type)	II	DHEIG2	50662	21897	6.5×10^{-20}	
	J	EJHPEC	49849	15405	4.0×10^{-19}	IDEF=0, EPS1=0
All values, corresp. vectors	H	HERMTD	169222	900000	(Not obtained)	No packed form, EPS=10^{-4}
	II	DTRIDH DTEIG1 DTRBKH	130632	56731	2.7×10^{-20}	
	J	EJHPAC	130262	51795	2.4×10^{-20}	IMP=1

Problem GH　これは核融合におけるプラズマの安定性を調べる問題であり，一般問題の係数行列**A**，**B**はプラズマ変位から得られる。[51] このとき，固有値は不安定モードの成長率を表わし，固有ベクトルは不安定モードの変位を表わす。もとの問題は322次で，**A**がHermite，**B**が実対称正定値で，ともに帯幅が約120の帯行列である。このままだと，数値実験が大変なため，ここでは上の行列の性質を壊さないよう形式的に縮約し，付録8に示されるように帯幅が21の46次の問題として扱われる。

Table 17はこの結果である。この問題に最も適した形になっているルーチンはCSIVIであり，**A**，**B**が帯行列で，**A**がHermite，**B**が実対称正定値という性質がそのまま生かされる。SIVIも帯状の一般問題を扱う，特殊問題のN－型用のルーチンであるが，**A**が実対称でないと解けないため，(97)式と同様の還元をして，実対称行列の一般問題として解く。他のルーチンもSIVIと同様Hermite 行列は直接扱えないため，**A**が実対称，**B**が実対称正定値で92次という問題として解く。これらは更に帯行列という特徴も生かせず，対称であるということに対してすら圧縮型になっていない。

表の最初の組は，特殊問題として46次の4つの固有値とその固有ベクトルを求めたもので，SIVI とEJGSEC では92次の8つの固有値と固有ベクトルを計算することになる。従って，CSIVI の精度が

$$\max_{1\leqq j\leqq 4}\frac{\|\mathbf{A}\mathbf{x}_j-\lambda_j\mathbf{B}\mathbf{x}_j\|_1}{\|\mathbf{A}\|_1\ \|\mathbf{x}_j\|_1} \tag{98}$$

でなされるのに対し，CSIVI 以外では8つの固有値と固有ベクトルを使って評価される。この46個の固有値はすべて正であるため，N－型で零に最も近い4個と指定するのと，E－型で最も小さい4個と指定するのは同等である。CSIVI とSIVI における他のパラメーターは，収束条件を10^{-5}，計算機の精度を10^{-16}，(78)式の最大反復回数を20としている。これらは多少変えても結果に大きな変動を与えない。EJGSECでは正定値性が利用される。これらの結果をみると，CSIVI に対して，他の2つのルーチンはやはり記憶容量が大きくなっている。SIVIの計算時間がCSIVI より長いのは当然であるが，EJGSECと比べた場合，時間，精度とも悪くなっている。しかし，CSIVI とSIVI で使われている多段階逆反復法のアルゴリズムは，もっと高次の問題を分割して処理しながら解くとき，その長所を発揮するだろう。

表の後半の組は完全問題としてとらえたもので，注釈もHermite の完全問題を解く圧縮型のルーチンを基準にして付けられている。計算条件も前の組と同様で，EJGSEC では固有値の小さい順に計算されている。CSIVI とSIVI はN－型のため，すべての固有値や固有ベクトルを求めるには向かないようで，記憶容量も大きく，計算も収束していない。DIRNMD も，記憶容量は小さいが，精度も悪く，満足な解が得られたとは言えない。残りのルーチンは，記憶容量，計算時間，それに精度とも十分であるがやはり完全問題用のEJGSACが最善と言えよう。

Problem GI　縮退した問題では一般化逆行列の計算がとり上げられる。テスト問題としては

$$\mathbf{A}=(a_{ij})=1+(i-1)\times(j-1)^2 \tag{99}$$

なる（9，5）行列[13]が与えられるが，この階数は5である。

この結果はTable 18に示されるが，この場合の精度は

$$\frac{\|\mathbf{A}\mathbf{A}^{-1}\mathbf{A}-\mathbf{A}\|_1}{\|\mathbf{A}\|_1} \tag{100}$$

で評価される。EJSVDGの特異値は，10^{-16}より小さいとき零とみなされるが，この例ではすべて正である。掲げられた2つのルーチンは，ともにそん色なく，よほど高次か悪条件でなければ使用に耐え得よう。

Table 17 Computational results for Problem GH (modified matrices made from coefficients of plasma displacements, order 46)

Test	Library	Entry name	Core (words)	Time (msec)	Accuracy	Comment
Four smallest values, corresp. vectors	J	CSIVI	8830	2420	1.3×10^{-11}	AL(1)=0. EPSCON=10^{-5}, EPSMAC=10^{-16}, NPIN(5)=20
	J	SIVI	26906	7253	2.7×10^{-11}	Real sym. version, AL(1)=0, EPSCON=10^{-5}, EPSMAC=10^{-16}, NPIN(5)=20
	J	EJGSEC	40620	2416	3.2×10^{-19}	Real sym. version, no packed form, E-type, IDEF=1, EPS1=0
All values, corresp. vectors	J	CSIVI	33206	151365	3.5×10^{-3}	N-type, AL(1)=0, EPSCON=10^{-5}, EPSMAC=10^{-16}, EPIN(5)=20
	J	SIVI	75658	292888	1.2×10^{-2}	Real sym. version, N-type, AL(1)=0, EPSCON=10^{-5}, EPSMAC=10^{-16}, NPIN(5)=20
	J	EJGSEC	56812	8263	1.1×10^{-15}	Real sym. version, no packed form, E-type, TYPE=.TRUE., IDEF=1, EPS1=0
	H	GEIGND	54712	9432	1.5×10^{-18}	Real sym. version, no packed form
	J	DIRNMD	51974	129424	1.7×10^{-8}	Real sym. version, no packed form
	J	EJGSAC	55128	7367	1.4×10^{-18}	Real sym. version, no packed form, IMP=1

Table 18 Computational results for Problem GI
(a degenerated (9,5) matrix)

Test	Library	Entry name	Core (words)	Time (msec)	Accuracy $\frac{\|AA^{-1}A-A\|_1}{\|A\|_1}$	Comment
Gen. inv.	H	GMINVD	5024	5	7.0×10^{-19}	
	J	EJSVDG	4736	6	1.6×10^{-18}	TV=10^{-16}

10. 結　　言

本稿では，広く固有値問題と言われる様々な形の問題について立体的に考察し，それに則って既存のルーチンを概観した。既存のルーチンは，SSL・H，SSL－Ⅱ，JSSLの3つの科学計算用ライブリーに登録されているが，これらで実際に起こり得る固有値問題に対処するには充分と言えず，原研で入手できるプログラムを中心に，その拡充が試みられた。

米国ANLで開発された，固有値問題専用パッケージEISPACK－2は，計算速度も速く高精度で多種類の問題が扱え，この分野での評価も高かった。しかし，ＳＳＬとしてこれらをそのまま現有の計算機で使用する場合，非常に使いにくく，所内では従来も余り利用された形跡は見当らない。このパッケージは，流れ図に従い，基本となるルーチンを使用者が組合わせて使うようになっているが，このままでは分りにくく，かつ間違い易い。

EISPACK－J はかかる不便を取り払い，かつ縮退した問題に関しても開発を加えて，EISPACK－2を発展させたものである。これにより，既存のルーチンに比べて，実3重対角行列と実対称帯行列の標準問題，実行列等の一般問題，それに縮退した問題などが扱えるようになった。また，EISPACK－J の開発と平行して，原研で入手可能なルーチンの整備も行った。これらの多くは上記のパッケージと共通するものが多いが，ベンチマーク・テストにより，その特徴が明らかにされた。また，実対称行列の標準問題で記憶容量を節約したり，帯行列の一般問題を解くなど，特徴あるものも含まれている。

これら全ルーチンに使われている解法や技法は3つの章で概要を述べたが，同じ解法でも，プログラムの仕方，系の次数，行列の性質等により計算の状況は大きく変わる。このため，少数ではあるが身近かな例題を使い，ベンチマーク・テストが実施された。テストにはすべて倍精度計算が用いられたが，テストを容易にするため，どうしても低次の問題が使われがちであった。しかし，Hermite の標準問題で 145 次の系を解いたときも含めて，すべての固有値と固有ベクトルの相対誤差が 10^{-18}～10^{-20} 程度に得られた。このように，SSL－ⅡやEISPACK－J のルーチンはまず標準ルーチンとして安心して使えよう。変った解法としては，Danilevsky 法，Greenstadt 法，Eberlein 法，べき乗法等もテストされたが，これらは標準ルーチンとしてこ

Table 18 Computational results for Problem GI
(a degenerated (9,5) matrix)

Test	Library	Entry name	Core (words)	Time (msec)	Accuracy $\frac{\|AA^{-1}A-A\|_1}{\|A\|_1}$	Comment
Gen. inv.	H	GMINVD	5024	5	7.0×10^{-19}	
	J	EJSVDG	4736	6	1.6×10^{-18}	TV=10^{-16}

10. 結　　言

本稿では，広く固有値問題と言われる様々な形の問題について立体的に考察し，それに則って既存のルーチンを概観した。既存のルーチンは，SSL・H，SSL－Ⅱ，JSSLの3つの科学計算用ライブリーに登録されているが，これらで実際に起こり得る固有値問題に対処するには充分と言えず，原研で入手できるプログラムを中心に，その拡充が試みられた。

米国ANLで開発された，固有値問題専用パッケージEISPACK－2は，計算速度も速く高精度で多種類の問題が扱え，この分野での評価も高かった。しかし，ＳＳＬとしてこれらをそのまま現有の計算機で使用する場合，非常に使いにくく，所内では従来も余り利用された形跡は見当らない。このパッケージは，流れ図に従い，基本となるルーチンを使用者が組合わせて使うようになっているが，このままでは分りにくく，かつ間違い易い。

EISPACK－J はかかる不便を取り払い，かつ縮退した問題に関しても開発を加えて，EISPACK－2を発展させたものである。これにより，既存のルーチンに比べて，実3重対角行列と実対称帯行列の標準問題，実行列等の一般問題，それに縮退した問題などが扱えるようになった。また，EISPACK－J の開発と平行して，原研で入手可能なルーチンの整備も行った。これらの多くは上記のパッケージと共通するものが多いが，ベンチマーク・テストにより，その特徴が明らかにされた。また，実対称行列の標準問題で記憶容量を節約したり，帯行列の一般問題を解くなど，特徴あるものも含まれている。

これら全ルーチンに使われている解法や技法は3つの章で概要を述べたが，同じ解法でも，プログラムの仕方，系の次数，行列の性質等により計算の状況は大きく変わる。このため，少数ではあるが身近かな例題を使い，ベンチマーク・テストが実施された。テストにはすべて倍精度計算が用いられたが，テストを容易にするため，どうしても低次の問題が使われがちであった。しかし，Hermite の標準問題で145次の系を解いたときも含めて，すべての固有値と固有ベクトルの相対誤差が10^{-18}～10^{-20}程度に得られた。このように，SSL－ⅡやEISPACK－J のルーチンはまず標準ルーチンとして安心して使えよう。変った解法としては，Danilevsky 法，Greenstadt 法，Eberlein 法，べき乗法等もテストされたが，これらは標準ルーチンとしてこ

のまま用いるにはやはり難が有るようで，それなりの場面で用いるか，より標準化するかの工夫が望まれる。

テスト問題も，できるだけ実際問題を採用すべく，ここでも4件が採用された。しかし，手元にない場合は多くは文献53)，54) 等のテキストに頼ることが多かった。これらはそれなりに理論的な性質，例えば固有値の厳密解などが分っていて有利なこともあるが，記憶容量や計算時間の評価があいまいとなる。

具体例について述べると，Problem SR−1 のLotkin の行列の修正では，第3行に 10^{-6} を掛けただけで，第3列はそのままの方が規格化のテストには良かったかも知れない。[57] Problem SRS−1，SH−1，それにGI ではもっと次数を高くすべきであった。また，Problem SRS−2 のEJSNEC のように，2，3の収束解が得られなかった場合の追求が充分でなかった。これらは将来の研究課題となろう。

SSL の拡充という意味では，一般問題では実帯行列の系を解くもの，帯状でない疎な系を解くものなどがさし当って望まれる。また，Problem GRS のテスト結果から察せられるように，EJGSEC のアルゴリズムを帯行列用に修正できれば，もっと効果的なルーチンが開発できよう。この時点より新しい発展として，べき乗法を改良したものがあり，[5] これらも今後の拡充にとって大いに期待される。

謝　　辞

ベンチマーク・テストで取り上げる実際問題については大いに苦労した。しかし，原子炉制御研究室の島崎潤也氏，高速炉物理研究室の三谷浩氏，固体物理第2研究室の佐々木健氏，理論解析研究室の常松俊秀氏の各氏には手持ちの問題を提供していただき，計算結果の検討にも加わっていただいた。富士通（株）の田子精男氏，杉本南海夫氏，三上次郎氏の各氏には，SSL−IIなどについて御教示をいただくことが多かった。また，富士通（株）の松浦俊彦氏，原子炉システム研究室の中原康明氏の各氏からは，本報告書作成にあたり有益な助言をいただいた。ここに謝意を表します。

のまま用いるにはやはり難が有るようで，それなりの場面で用いるか，より標準化するかの工夫が望まれる。

テスト問題も，できるだけ実際問題を採用すべく，ここでも4件が採用された。しかし，手元にない場合は多くは文献53)，54) 等のテキストに頼ることが多かった。これらはそれなりに理論的な性質，例えば固有値の厳密解などが分っていて有利なこともあるが，記憶容量や計算時間の評価があいまいとなる。

具体例について述べると，Problem SR−1 のLotkin の行列の修正では，第3行に 10^{-6} を掛けただけで，第3列はそのままの方が規格化のテストには良かったかも知れない。[57] Problem SRS−1，SH−1，それにGI ではもっと次数を高くすべきであった。また，Problem SRS−2 のEJSNEC のように，2，3の収束解が得られなかった場合の追求が充分でなかった。これらは将来の研究課題となろう。

SSL の拡充という意味では，一般問題では実帯行列の系を解くもの，帯状でない疎な系を解くものなどがさし当って望まれる。また，Problem GRS のテスト結果から察せられるように，EJGSEC のアルゴリズムを帯行列用に修正できれば，もっと効果的なルーチンが開発できよう。この時点より新しい発展として，べき乗法を改良したものがあり，[5] これらも今後の拡充にとって大いに期待される。

謝　　辞

ベンチマーク・テストで取り上げる実際問題については大いに苦労した。しかし，原子炉制御研究室の島崎潤也氏，高速炉物理研究室の三谷浩氏，固体物理第2研究室の佐々木健氏，理論解析研究室の常松俊秀氏の各氏には手持ちの問題を提供していただき，計算結果の検討にも加わっていただいた。富士通（株）の田子精男氏，杉本南海夫氏，三上次郎氏の各氏には，SSL−IIなどについて御教示をいただくことが多かった。また，富士通（株）の松浦俊彦氏，原子炉システム研究室の中原康明氏の各氏からは，本報告書作成にあたり有益な助言をいただいた。ここに謝意を表します。

参考文献

1) 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（1968）．
2) クーラン　R．，ヒルベルト　D．：“数理物理学の方法　1”（斉藤利称監訳），東京図書（1959）．
3) 一松　信：“数値計算”，至文堂（1966）．
4) 新谷尚義：“数値計算Ⅰ”朝倉書店（1972）．
5) 戸川隼人：“マトリクスの数値計算”，オーム社（1971）．
6) Faddeeva V.N.: "Computational Methods of Linear Algebra" (translated by Benster C.D.), Dover Publications, Inc. (1959) および，小国力（訳）：“線型代数の計算法（上），（下）”，産業図書（1970）．
7) Wilkinson J.H.: "The Algebraic Eigenvalue Problem", Oxford University Press (1965).
8) 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）．
9) 富士通（株）：“FACOM FORTRAN SSL 使用手引書（99 SP－0040－1）”富士通（1976）．
10) 富士通（株）：“FACOM SSL（科学用サブルーチン・ライブラリ）解法解説書（000－301～309－003－5）”，富士通（1972）．
11) 富士通（株）：“FACOM FORTRAN SSL Ⅱ使用手引書（99 SP－0050－2）”，富士通（1977）．
12) 藤村統一郎，西田雄彦，浅井清（編）：“JSSL（原研版・科学用サブルーチン・ライブラリー）マニュアル”，JAERI－M 7102（1977）．
13) Smith B.T., et al.: "Matrix Eigensystem Routines - EISPACK Guide", Springer-Verlag (1974).
14) 古屋　茂：“行列と行列式”，培風館（1967）．
15) 山内二郎，森口繁一，一松　信：“電子計算機のための数値計算法Ⅰ”培風館（1965）．
16) 朝岡卓見：“高次代数方程式の数値解法プログラム（SSL の拡充とベンチマーク・テスト No 1）”，JAERI－M 7335（1977）．
17) Madsen K.: "A Root-Finding Algorithm Based on Newton's Method", BIT, 13, 71 (1973).
18) Eberlein P.J., Boothroyd J.: "Solution to the Eigenproblem by a Norm Reducing Jacobi Type Method", Numer. Math., 11, 1 (1968).
19) Eberlein P.J.: "Solution to the Complex Eigenproblem by a Norm Reducing Jacobi Type Method", Numer. Math., 14, 232 (1970).
20) Martin R.S., Reinsch C., Wilkinson J.H.: "Householder's Tridiagonalization of a Symmetric Matrix", Numer. Math., 11, 181 (1968).

21) Schwarz H.R.: "Tridiagonalization of a Symmetric Band Matrix", Numer. Math., 12, 231 (1968).

22) Budde C.D.: "The Reduction of an Arbitrary Real Square Matrix to Tridiagonal Form Using Similarity Transformations", Math. Comp. 17, 433 (1963).

23) Francis J.G.F.: "The QR Transformation, A Unitary Analogue to the LR Transformation - Part I", Comp. J., 4, 265 (1961).

24) Martin R.S., Wilkinson J.H.: "The Modified LR Algorithm for Complex Hessenberg Matrices", Numer. Math., 12, 369 (1968).

25) Francis J.G.F.: "The QR Transformation - Part II", Comp. J., 4, 332 (1962).

26) Kempen H.P.M.: "On the Convergence of the Classical Jacobi Method for Real Symmetric Matrices with Non-Distinct Eigenvalues", Numer. Math., 9, 11 (1966).

27) Wilkinson J.H.: "The QR Algorithm for Real Symmetric Matrices with Multiple Eigenvalues", Comp. J., 8, 85 (1965).

28) Bowdler H., et al.: "The QR and QL Algorithms for Symmetric Matrices", Numer. Math., 11, 293 (1968).

29) Martin R.S., Wilkinson J.H.: "The Implicit QL Algorithm", Numer. Math., 12, 377 (1968).

30) Businger P.A.: "Reducing a Matrix to Hessenberg Form", Math. Comp., 23, 819 (1969).

31) Martin R.S., Wilkinson J.H.: "Similarity Reduction of a General Matrix to Hessenberg Form", Numer. Math., 12, 349 (1968).

32) Parlett B., Reinsch C.: "Balancing a Matrix for Calculation of Eigenvalues and Eigenvectors", Numer. Math., 13, 293 (1969).

33) Reinsch C.H.: "A Stable, Rational QR Algorithm for the Computation of the Eigenvalues of an Hermitian, Tridiagonal Matrix", Math. Comp., 25, 591 (1971).

34) Reinsch C., Bauer F.L.: "Rational QR Transformation with Newton Shift for Symmetric Tridiagonal Matrices", Numer. Math., 11, 264 (1968).

35) Martin R.S., Reinsch C., Wilkinson J.H.: "The QR Algorithm for Band Symmetric Matrices", Numer. Math., 16, 85 (1970).

36) Barth W., Martin R.S., Wilkinson J.H.: "Calculation of the Eigenvalues of a Symmetric Tridiagonal Matrix by the Method of Bisection", Numer. Math., 4, 362 (1962) and 9, 386 (1967) (reviced).

37) Wilkinson J.H.: "Calculation of the Eigenvectors of a Symmetric Tridiagonal Matrix by Inverse Iteration", Numer. Math., 4, 368 (1962).

38) Grad J., Brebner M.A.: "Eigenvalues and Eigenvectors of a Real General Matrix", Comm. ACM, 11, 820 (1968) and 13, 122 (1970).

39) Knoble H.D.: "Eigenvalues and Eigenvectors of a Real General Matrix", Comm. ACM, 13, 122 (1970).

40) Stewart G.W.: "On the Sensitivity of the Eigenvalue Problem Ax=λBx", SIAM J. Numer. Anal., 9, 669 (1972).

41) Moler C.B., Stewart G.W.: "An Algorithm for Generalized Matrix Eigenvalue Problems", SIAM J. Numer. Anal., 10, 241 (1973).

42) Martin R.S., Wilkinson J.H.: "Reduction of the Symmetric Eigenproblem Ax=λBx and Related Problems to Standard Form", Numer. Math., 11, 99 (1968).

43) Golub G.H., Reinsch C.: "Singular Value Decomposition and Least Squares Solutions", Numer. Math., 14, 403 (1970).

44) 杉本南海夫：私信

45) 藤村統一郎："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）", JAERI－M 7553（1978）.

46) 島崎潤也："動特性のモード解析用べき乗法の改良", JAERI－M（公刊予定）.

47) 藤田恵一：私信

48) 東京大学大型計算機センター（編）："東京大学大型計算機センター ライブラリー・プログラム 第Ⅰ集", 東京大学出版会（1967）.

49) Dobosh P.A.: "DIGN1M: A Matrix Diagonalization Subroutine with Minimum Storage Requirements", Comp. Chem., 1, 295 (1977).

50) Gruber R.: "HYMN1A - Band Matrix Package for Solving Eigenvalue Problems", Comp. Phys. Comm., 10, 30 (1975).

51) Tsunematsu T., Takeda T.: "A New Iteration Method for Solution of a Large-Scale General Eigenvalue Problem", J. Comp. Phys., 28, 287 (1978).

52) 富士通（株）："FACOM 230 M－Ⅶ FORTRAN－H 使用手引書（75 SP－0281－1）", 富士通（1976）.

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

54) Gregory R.T., Karney D.L.: "A Collection of Matrices for Testing Computational Algorithms", John Wiley & Sons, Inc. (1969).

55) Mitani H., et al.: "Sensitivity Analysis for Actinide Production and Depletion in Fast Reactor", JAERI-M 8133 (1979).

56) Sasaki K.: "Band Structure of α-Manganese Metal", JAERI-Report (to be published).

57) 星野　聰：“行列の固有値固有ベクトル計算ライブラリの問題点”，数理解析研究所講究録 269，70，京都大学（1976）．

付録1　EISPACK-J 使用手引書

付録1　EISPACK—J 使用手引書

ここでは各ルーチンの呼び出し方が説明される。各ルーチンが扱う問題や系の性質，解法などはTable 4を中心とした，第7章を参照されたい。また，計算時の記憶容量，時間，精度については，第9章のベンチマーク・テストを参照するか，EISPACK－Jに付けられている例題を検証されたい。なお，更に解法やプログラミング上の詳しいことはEISPACK－2の説明書に見ることができる。

下記のリストは，Table 4に対応する全ルーチンの一覧表である。この呼び出し方の説明において，各仮引数は，引数名，内容，型，配列のとり方，入出力，注釈の順に簡単に記述される。例えば配列でA（NM，N）とある場合，Aの第1の寸法をNM，第2の寸法をN以上とることを意味する。また入出力の項で，入力とはそのルーチンを呼ぶ前に必要な値を入れなければならないことを意味し，出力とはそのルーチンに入ってから一度以上値が変えられることを意味している。なお，類似のルーチンの間で同じ説明文が繰り返される場合，説明が省略されるので，関連するルーチンの説明を参照されたい。

```
EJRNEN (NM,N,A,WR,WI,INT,LA)  ------------------------------------------------------------  73
EJRNEC (NM,N,A,WR,WI,Z,INT,LA,LE)  ------------------------------------------------------  73
EJRNES (NM,MM,N,A,AS,WR,WI,Z,SELECT,INT,RM1,RV1,RV2,LA,LC)  -----------------------------  74
EJRNON (NM,N,A,WR,WI,ORT,LA)  ------------------------------------------------------------  74
EJRNOC (NM,N,A,WR,WI,Z,ORT,LA,LE)  ------------------------------------------------------  74
EJRNOS (NM,MM,N,A,AS,WR,WI,Z,SELECT,ORT,RM1,RV1,RV2,LA,LC)  -----------------------------  75
EJRBEN (NM,N,A,WR,WI,INT,SCALE,LA)  ------------------------------------------------------  75
EJRBEC (NM,N,A,WR,WI,Z,INT,SCALE,LA,LE)  ------------------------------------------------  76
EJRBES (NM,MM,N,A,AS,WR,WI,Z,SELECT,INT,SCALE,RM1,RV1,RV2,LA,LC)  -----------------------  76
EJRBON (NM,N,A,WR,WI,ORT,SCALE,LA)  ------------------------------------------------------  77
EJRBOC (NM,N,A,WR,WI,Z,ORT,SCALE,LA,LE)  ------------------------------------------------  77
EJRBOS (NM,MM,N,A,AS,WR,WI,Z,SELECT,ORT,SCALE,RM1,RV1,RV2,LA,LC)  -----------------------  77
EJRTNA (NM,N,A,E,W,LA,IMP)  --------------------------------------------------------------  77
EJRTNB (NM,MM,N,A,D,E,E2,W,IND,RV4,RV5,LA,L,LB,UB,EPS1)  ---------------------------------  78
EJRTNI (NM,MM,N,A,D,E,E2,W,IND,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)  -------------  79
EJRTNE (NM,MM,N,A,E,W,BD,IND,LA,TYPE,IDEF,EPS1)  -----------------------------------------  79
EJRTPA (NM,N,A,E,W,Z,L,IMP)  -------------------------------------------------------------  80
EJRTPB (NM,MM,N,A,D,E,E2,W,Z,RV1,RV2,RV3,RV4,RV5,RV6,LA,L,LB,UB,EPS1)  -------------------  81
EJRTPI (NM,MM,N,A,D,E,E2,W,Z,IND,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)  -----------  81
EJRTPE (NM,MM,N,A,D,E,E2,W,Z,BD,IND,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,TYPE,IDEF,EPS1)  ---------  82
EJSNAN (NM,N,A,E,W,LA,IMP)  --------------------------------------------------------------  82
EJSNAC (NM,N,A,E,W,Z,L,IMP)  -------------------------------------------------------------  83
EJSNBN (NM,MM,N,A,D,E,E2,W,IND,RV4,RV5,LA,L,LB,UB,EPS1)  ---------------------------------  83
EJSNBC (NM,MM,N,A,D,E,E2,W,Z,RV1,RV2,RV3,RV4,RV5,RV6,LA,L,LB,UB,EPS1)  -------------------  84
EJSNIN (NM,MM,N,A,D,E,E2,W,IND,RV4,RV5,LA,M11,LB,UB,EPS1)  -------------------------------  84
EJSNIC (NM,MM,N,A,D,E,E2,W,Z,IND,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)  ----------  85
EJSNEN (NM,MM,N,A,E,W,BD,IND,LA,TYPE,IDEF,EPS1)  -----------------------------------------  85
EJSNEC (NM,MM,N,A,D,E,E2,W,Z,BD,IND,RV1,RV2,RV3,RV4,RV6,LA,LE,TYPE,IDEF,EPS1)  ------------  86
EJSPAN (N,A,E,W,LA,IMP)  -----------------------------------------------------------------  86
EJSPAC (NM,N,A,E,W,Z,L,IMP)  -------------------------------------------------------------  86
EJSPBN (MM,N,A,D,E,E2,W,IND,RV4,RV5,LA,L,LB,UB,EPS1)  ------------------------------------  87
EJSPBC (NM,MM,N,A,D,E,E2,W,Z,RV1,RV2,RV3,RV4,RV5,RV6,LA,L,LB,UB,EPS1)  -------------------  88
EJSPIN (MM,N,A,D,E,E2,W,IND,RV4,RV5,LA,M11,LB,UB,EPS1)  ----------------------------------  88
EJSPIC (NM,MM,N,A,D,E,E2,W,Z,IND,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)  -----------  88
EJSPEN (MM,N,A,E,W,BD,IND,LA,TYPE,IDEF,EPS1)  --------------------------------------------  89
EJSPEC (NM,MM,N,A,D,E,E2,W,Z,BD,IND,RV1,RV2,RV3,RV4,RV6,LA,LE,TYPE,IDEF,EPS1)  ------------  89
EJSBAN (NM,N,MB,A,E,W,LA,IMP)  -----------------------------------------------------------  90
EJSBAC (NM,N,MB,A,E,W,Z,L,IMP)  ----------------------------------------------------------  90
EJSBBN (NM,MM,N,MB,A,D,E,E2,W,IND,RV4,RV5,LA,L,LB,UB,EPS1)  ------------------------------  91
EJSBBC (NM,MM,N,MB,A,AS,D,E,E2,W,Z,IND,RV1,RV6,RV,LA,LE,L,LB,UB,EPS1)  -------------------  91
EJSBIN (NM,MM,N,MB,A,D,E,E2,W,IND,RV4,RV5,LA,M11,LB,UB,EPS1)  ----------------------------  92
EJSBIC (NM,MM,N,MB,A,AS,D,E,E2,W,Z,IND,RV4,RV5,RV6,RV,LA,LE,M11,LB,UB,EPS1)  -------------  92
EJSBEN (NM,MM,N,MB,A,E,W,BD,IND,LA,TYPE,IDEF,EPS1)  --------------------------------------  93
EJSBEC (NM,MM,N,MB,A,AS,E,W,Z,BD,IND,RV6,RV,LA,LE,TYPE,IDEF,EPS1)  ------------------------  93
EJSBNN (NM,MM,N,MB,A,W,RV,LA,T,R)  -------------------------------------------------------  94
EJSBNC (NM,MM,N,MB,A,AS,W,Z,RV6,RV,LA,LE,T,R)  -------------------------------------------  94
EJSTAN (N,E,W,LA,IMP)  -------------------------------------------------------------------  94
EJSTAC (NM,N,E,W,Z,L,IMP)  ---------------------------------------------------------------  95
EJSTBN (MM,N,D,E,E2,W,IND,RV4,RV5,LA,L,LB,UB,EPS1)  --------------------------------------  95
EJSTBC (NM,MM,N,D,E,E2,W,Z,RV1,RV2,RV3,RV4,RV5,RV6,LA,L,LB,UB,EPS1)  ---------------------  96
EJSTIN (MM,N,D,E,E2,W,IND,RV4,RV5,LA,M11,LB,UB,EPS1)  ------------------------------------  96
EJSTIC (NM,MM,N,D,E,E2,W,Z,IND,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)  -------------  97
EJSTEN (MM,N,E,W,BD,IND,LA,TYPE,IDEF,EPS1)  ----------------------------------------------  97
EJSTEC (NM,MM,N,D,E,E2,W,Z,BD,IND,RV1,RV2,RV3,RV4,RV6,LA,LE,TYPE,IDEF,EPS1)  --------------
```

```
EJCNEN (NM,N,AR,AI,WR,WI,INT,LA)  ----------                                                    98
EJCNEC (NM,N,AR,AI,WR,WI,ZR,ZI,INT,LA,LE)  ----------                                          99
EJCNES (NM,MM,N,AR,AI,ARS,AIS,WR,WI,ZR,ZI,SELECT,INT,RM1,RM2,RV1,RV2,LA,LE)  ----------       99
EJCNUN (NM,N,AR,AI,WR,WI,ORTR,ORTI,LA)  ----------                                             100
EJCNUC (NM,N,AR,AI,WR,WI,ZR,ZI,ORTR,ORTI,LA,LE)  ----------                                    100
EJCNUS (NM,MM,N,AR,AI,ARS,AIS,WR,WI,ZR,ZI,SELECT,ORTR,ORTI,RM1,RM2,RV1,RV2,LA,LE)  ---------  100
EJCBEN (NM,N,AR,AI,WR,WI,INT,SCALE,LA)  ----------                                             100
EJCBEC (NM,N,AR,AI,WR,WI,ZR,ZI,INT,SCALE,LA,LE)  ----------                                    101
EJCBES (NM,MM,N,AR,AI,ARS,AIS,WR,WI,ZR,ZI,SELECT,INT,SCALE,RM1,RM2,RV1,RV2,LA,LE)  ---------  101
EJCBUN (NM,N,AR,AI,WR,WI,ORTR,ORTI,SCALE,LA)  ----------                                       102
EJCBUC (NM,N,AR,AI,WR,WI,ZR,ZI,ORTR,ORTI,SCALE,LA,LE)  ----------                              102
EJCBUS (NM,MM,N,AR,AI,ARS,AIS,WR,WI,ZR,ZI,SELECT,ORTR,ORTI,SCALE,RM1,RM2,RV1,RV2,LA,LE)  ---  103
EJHNAN (NM,N,AR,AI,E,W,TAU,LA,IMP)  ----------                                                 103
EJHNAC (NM,N,AR,AI,E,W,ZR,ZI,TAU,LA,LE,IMP)  ----------                                        103
EJHNBN (NM,MM,N,AR,AI,D,E,E2,W,IND,TAU,RV4,RV5,LA,L,LB,UB,EPS1)  ----------                    104
EJHNBC (NM,MM,N,AR,AI,D,E,E2,W,ZR,ZI,IND,TAU,RV1,RV2,RV3,RV4,RV5,RV6,LA,L,LB,UB,EPS1)  -----  104
EJHNIN (NM,MM,N,AR,AI,D,E,E2,W,IND,TAU,RV4,RV5,LA,M11,LB,UB,EPS1)  ----------                   105
EJHNIC (NM,MM,N,AR,AI,D,E,E2,W,ZR,ZI,IND,TAU,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)--  105
EJHNEN (NM,MM,N,AR,AI,E,W,BD,IND,TAU,LA,TYPE,IDEF,EPS1)  ----------                            106
EJHNEC (NM,MM,N,AR,AI,D,E,E2,W,ZR,ZI,BD,IND,TAU,RV1,RV2,RV3,RV4,RV6,LA,LE,TYPE,IDEF,EPS1) --  106
EJHPAN (NM,N,A,E,W,TAU,LA,IMP)  ----------                                                     107
EJHPAC (NM,N,A,E,W,ZR,ZI,TAU,L,IMP)  ----------                                                107
EJHPBN (NM,MM,N,A,D,E,E2,W,IND,TAU,RV4,RV5,LA,L,LB,UB,EPS1)  ----------                        107
EJHPBC (NM,MM,N,A,D,E,E2,W,ZR,ZI,IND,TAU,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,L,LB,UB,EPS1)  ------  108
EJHPIN (NM,MM,N,A,D,E,E2,W,IND,TAU,RV4,RV5,LA,M11,LB,UB,EPS1)  ----------                      109
EJHPIC (NM,MM,N,A,D,E,E2,W,ZR,ZI,IND,TAU,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)  ----  109
EJHPEN (NM,MM,N,A,E,W,BD,IND,TAU,LA,TYPE,IDEF,EPS1)  ----------                                109
EJHPEC (NM,MM,N,A,D,E,E2,W,ZR,ZI,BD,IND,TAU,RV1,RV2,RV3,RV4,RV6,LA,LE,TYPE,IDEF,EPS1)  ------  110
EJGRGN (NM,N,A,B,ALFR,ALFI,BETA,LA)  ----------                                                110
EJGRGC (NM,N,A,B,ALFR,ALFI,BETA,Z,LA,LE)  ----------                                           111
EJGSAN (NM,N,A,B,E,W,LA,IMP)  ----------                                                       111
EJGSAC (NM,N,A,B,E,E2,W,Z,LA,M,IMP)  ----------                                                112
EJGSBN (NM,MM,N,A,B,D,E,E2,W,IND,RV4,RV5,LA,L,LB,UB,EPS1)  ----------                          112
EJGSBC (NM,MM,N,A,B,D,E,E2,W,Z,DL,RV1,RV2,RV3,RV4,RV5,RV6,LA,L,LB,UB,EPS1)  ----------        113
EJGSIN (NM,MM,N,A,B,D,E,E2,W,IND,RV4,RV5,LA,M11,LB,UB,EPS1)  ----------                        114
EJGSIC (NM,MM,N,A,B,D,E,E2,W,Z,IND,DL,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1)  -------  114
EJGSEN (NM,MM,N,A,B,E,W,BD,IND,LA,TYPE,IDEF,EPS1)  ----------                                  114
EJGSEC (NM,MM,N,A,B,D,E,E2,W,Z,BD,IND,DL,RV1,RV2,RV3,RV4,RV6,LA,LE,TYPE,IDEF,EPS1)  --------  115
EJNABA (NM,N,A,B,E,W,LA,IMP)  ----------                                                       116
EJNABB (NM,MM,N,A,B,D,E,E2,W,IND,RV4,RV5,LA,L,LB,UB,EPS1)  ----------                          116
EJNABI (NM,MM,N,A,B,D,E,E2,W,IND,RV4,RV5,LA,M11,LB,UB,EPS1)  ----------                        117
EJNABE (NM,MM,N,A,B,E,W,BD,IND,LA,TYPE,IDEF,EPS1)  ----------                                  117
EJVABA (NM,N,A,B,E,E2,W,Z,LA,LE,IAB,IMP)  ----------                                           118
EJVABB (NM,MM,N,A,B,D,E,E2,W,Z,DL,RV1,RV2,RV3,RV4,RV5,RV6,LA,L,LB,UB,EPS1,IAB)  ----------    118
EJVABI (NM,MM,N,A,B,D,E,E2,W,Z,IND,DL,RV1,RV2,RV3,RV4,RV5,RV6,LA,LE,M11,LB,UB,EPS1,IAB)  ---  119
EJVABE (NM,MM,N,A,B,D,E,E2,W,Z,BD,IND,DL,RV1,RV2,RV3,RV4,RV6,LA,LE,TYPE,IDEF,EPS1,IAB)  ----  120
EJSVDP (NM,M,N,A,W,U,V,RV1,L)  ----------                                                      121
EJSVDL (NM,M,N,IP,A,C,W,RV1,L)  ----------                                                     121
EJSVDG (NM,M,N,A,W,U,V,RV1,TV,IERR)  ----------                                                121
EJSVDM (NM,M,N,IP,A,C,W,Z,RV1,TV,IERR)  ----------                                             122
EJLERB (NM,N,MBW,IP,A,C,RV1,RV6,RV,IERR)  ----------                                           123
EJLESB (NM,N,MB,IP,A,C,RV1,RV6,RV,IERR)  ----------                                            123
EJLERH (NM,M,N,A,W,RV1,TV,LA,LE)  ----------                                                   123
```

CALL **EJRNEN** (NM, N, A, WR, WI, INT, LA)

NM ―― 2次元配列Aの大きさを示す第1の整合寸法。整数型，入力。NM≧nとする。

N ―― 行列**A**の次数n。整数型，入力。N≧3とする。

A ―― 行列**A**を格納する。2次元倍精度実数型配列A（NM，N），入出力。行列$\mathbf{A}=(a_{ij})$をA(i,j)$=a_{ij}$と入れる。

WR―― 固有値の実数部を格納する。1次元倍精度実数型配列WR(N)，出力。LA＞0のとき，LA個の固有値の実数部がWR(j)（j＝n－LA＋1～n）に入れられる。共役固有値のとき続けて入れられる（WR(j)＝WR(j＋1))ほか特に格納の順序はない。

WI ―― 固有値の虚数部を格納する。1次元倍精度実数型配列WI(N)，出力。LA＞0のとき，LA個の固有値の虚数部がWI(j)（j＝n－LA＋1～n）に入れられる。共役固有値のとき，虚数部が正のものを先にして続けて入れられる（WR(j)＝－WR(j＋1)＞0）ほか特に格納の順序はない。

INT―作業領域。1次元整数型配列INT(N)，出力。

LA ―― 求まった固有値の数。整数型，出力。0≦LA≦nとなる。

CALL **EJRNEC** (NM, N, A, WR, WI, Z, INT, LA, LE)

NM ―― 2次元配列AやZの大きさを決める第1の整合寸法。整数型，入力。NM≧nとする。

N, A, WR, WI, INT, LA

―― EJRNEN参照。

Z ―― 固有ベクトルを格納する。2次元倍精度実数型配列Z（NM，N），出力。LE＞0のとき，固有値WR(j)＋i×WI(j)に対応する固有ベクトル（$\xi_{ij}+i\,\eta_{ij}$）は，固有値が実数であればZ(i,j)＝ξ_{ij}，複素数であればZ(i,j)＝ξ_{ij}，Z(i,j＋1)＝η_{ij}と入れられる。従って，共役固有値WR(j＋1)＋i×W(j＋1)に対応する固有ベクトルは(Z(i,j)－i×Z(i,j＋1))から求められる。これらのベクトルは正規化されていない。

LE ―― 求まった固有ベクトルの数。整数型，出力。LE＝0またはLE＝LAとなる。

CALL **EJRNES** (NM, MM, N, A, AS, WR, WI, Z, SELECT, INT, RM1, RV1, RV2, LA, LC)

NM ―― 2次元配列A，AS，ZおよびRM1の大きさを示す第1の整合寸法。整数型，入力。NM≧nとする。

MM ―― 2次元配列Zの大きさを示す第2の整合寸法。MM≧LCとする。

N, A, WR, WI, INT, LA

―― EJRNEN参照。

AS ——作業領域。2次元倍精度実数型配列 AS (NM, N), 出力。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列 Z (NM, MM), 出力。LC＞0のとき, 最終的に SELECT(j)=. TRUE. とされた j に対し WR(j)+i×WI(j)に対応する固有ベクトルが入れられる。格納の仕方は j の数の若い方から, 実ベクトルのときは1つの欄, 複素ベクトルのときは2つの欄を使って Z の第1欄から詰められる。なお, 複素ベクトルのときは実数部を先に詰める。

SELECT

——固有ベクトルの計算を指示する。1次元論理型配列 SELECT(N), 入出力。WR(j)+i×WI(j)に対する固有ベクトルを必要とするとき SELECT(j)=. TRUE., 必要としないとき. FALSE. とする。共役固有値 (WR(j)+i×WI(j), WR(j+1)+i×WI(j+1))の双方に対して. TRUE. としたとき, 最初の方の固有ベクトルのみ格納され, SELECT(j+1)=. FALSE. とされる。

固有値を確認してからその固有ベクトルを指定したいとき, このルーチンを2度呼べばよい。

RM1—作業領域。2次元倍精度実数型配列 RM1 (NM, N), 出力。

RV1—作業領域。1次元倍精度実数型配列 RV1(N), 出力。

RV2—作業領域。1次元倍精度実数型配列 RV2(N), 出力。

LC ——固有ベクトルの格納のため, 配列Zで実際に使われた欄の数。整数型, 出力。0≦LC≦MMとなる。但し, 指定通り固有ベクトルが求まらなかったときは LC＞MMの状態になった場合もある。

CALL **EJRNON** (NM, N, A, WR, WI, ORT, LA)

NM, N, A, WR, WI, LA

——EJRNEN参照。

ORT—作業領域。1次元倍精度実数型配列 ORT(N), 出力。

CALL **EJRNOC** (NM, N, A, WR, WI, Z, ORT, LA, LE)

NM, Z, LE

——EJRNEC 参照。

N, A, WR, WI, LA

——EJRNEN参照。

ORT—EJRNON参照。

CALL **EJRNOS** (NM, MM, N, A, AS, WR, WI, Z, SELECT, ORT, RM1, RV1, RV2, LA, LC)

NM, MM, AS

——EJRNES 参照。

N, A, WR, WI, LA

——EJRNEN参照。

AS, Z, SELECT, RM1, RV1, RV2, LC

——EJRNES 参照。

ORT—EJRNON参照。

CALL **EJRBEN** (NM, N, A, WR, WI, INT, SCALE, LA)

NM ——配列Aの大きさを示す第1の整合寸法。整数型，入力。NM≧n とする。

N ——行列**A**の次数 n 。整数型，入力。N≧3とする。

A ——行列**A**を格納する。2次元倍精度実数型配列A (NM, N)，入出力。行列$\mathbf{A}=(a_{ij})$をA(i, j) $=a_{ij}$ と入れる。

WR——固有値の実数部を格納する。1次元倍精度実数型配列 WR(N)，出力。LA＞0のとき，LA個の固有値の実数部がWR(j) (j =n −LA+1〜n) に入れられる。共役固有値のとき続けて入れられる (WR(j)= WR(j + 1)) ほか特に格納の順序はない。

WI ——固有値の虚数部を格納する。1次元倍精度実数型配列 WI(N)，出力。LA＞0のとき，LA個の固有値の虚数部が WI (j) (j =n −LA+ 1 〜n) に入れられる。共役固有値のとき，虚数部が正のものを先にして続けて入れる (WI (j)=− WI (j + 1)＞0) ほか特に格納の順序はない。

INT—作業領域。1次元整数型配列INT(N)，出力。

SCALE

——作業領域。1次元倍精度実数型配列 SCALE(N)，出力。

LA ——求まった固有値の数。整数型，出力。0≦LA≦n となる。N＞NMとした場合にもLA＝0となる。

CALL **EJRBEC** (NM, N, A, WR, WI, Z, INT, SCALE, LA, LE)

NM ——2次元配列AやZの大きさを決める第1の整合寸法。整数型，入力。NM≧n とする。

N, A, WR, WI, INT, SCALE, LA

——EJRBEN参照。

N ——固有ベクトルを格納する。2次元倍精度実数型配列Z (NM, N)，出力。LE＞0のとき，固有値 WR(j)+i × WI (j)に対応する固有ベクトル ($\xi_{ij}+i\,\eta_{ij}$) は，固有値が実数であればZ (i, j) $=\xi_{ij}$，複素数であればZ(i, j) $=\xi_{ij}$，Z (i, j +1) $=\eta_{ij}$ と入れられる。従って，共役固有値 WR (j+1) + i ×WI (j +1) に対応する固有ベクトルは (Z(i, j) − i ×Z (i, j +1)) から求められる。これらのベクトルは正規化されていない。

LE ——求まった固有ベクトルの数。整数型，出力。LE＝0 またはLE＝LAとなる。

CALL **EJRBES** (NM, MM, N, A, AS, WR, WI, Z, SELECT, INT, SCALE, RM1, RV1, RV2, LA, LC)

NM —— 2次元配列A，AS，Zおよび RM1の大きさを示す第1の整合寸法。整数型，入力。NM≧nとする。

MM —— 2次元配列Zの大きさを示す第2の整合寸法。整数型，入力。MM≧LCとする。

N, A, WR, WI, INT, SCALE

——EJRBEN参照。

AS —— 作業領域。2次元倍精度実数型配列 AS (NM, N)，出力。

Z —— 固有ベクトルを格納する。2次元倍精度実数型配列 Z (NM, N)，出力。LC＞0のとき，最終的に SELECT(j)＝.TRUE.とされたjに対し WR(j)＋i×WI(j) に対応する固有ベクトルが入れられる。格納の仕方はjの数の若い方から，実ベクトルのときは1つの欄，複素ベクトルのときは2つの欄を使ってZの第1欄から詰められる。なお，複素ベクトルのときは実数部を先に詰める。

SELECT

——固有ベクトルの計算を指示する。1次元論理型配列 SELECT(N)，入力。WR(j)＋i×WI(j) に対応する固有ベクトルを必要とするとき SELECT(j)＝.TRUE.，必要としないとき.FALSE.とする。共役固有値 (WR(j)＋i×WI(j＋1)，WR(j＋1)＋i×WI(j＋1)) の双方に対して.TRUE.としたとき，最初の方の固有ベクトルのみ格納され，SELECT(j＋1)＝.FALSE.とされる。

固有値を確認してからその固有ベクトルを指定したいとき，このルーチンを2度呼べばよい。

RM1 —作業領域。2次元倍精度実数型配列 RM1 (NM, N)，出力。

RV1 —作業領域。1次元倍精度実数型配列 RV1(N)，出力。

RV2 —作業領域。1次元倍精度実数型配列 RV2(N)，出力。

LA —— 求まった固有値の数。整数型，出力。0≦LA≦nとなる。

LC —— 固有ベクトルの格納のため，配列Zで実際に使われた欄の数。整数型，出力。0≦LC≦MMとなる。但し，指定通り固有ベクトルが求まらなかったときはLC＞MMの状態になった場合もある。

CALL **EJRBON** (NM, N, A, WR, WI, ORT, SCALE, LA)

NM, N, A, WR, WI, SCALE

——EJRBEN参照。

ORT —作業領域。1次元倍精度実数型配列 ORT(N)，出力。

LA ——EJRBES参照。

CALL **EJRBOC** (NM, N, A, WR, WI, Z, ORT, SCALE, LA, LE)

NM, Z, LE
——EJRBEC参照。
N, A, WR, WI
——EJRBEN参照。
ORT—EJRBON参照。
LA —EJRBES参照。

CALL **EJRBOS** (NM, MM, N, A, AS, WR, WI, Z, SELECT, ORT, SCALE, RM1, RV1, RV2, LA, LC)

NM, MM, AS, Z, SELECT, RM1, RV1, RV2, LA, LC
——EJRBES参照。
N, A, WR, WI, SCALE
——EJRBEN参照。
ORT—EJRBON参照。

CALL **EJRTNA** (NM, N, A, E, W, LA, IMP)

NM —— 2次元配列Aの大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。
N ——行列**A**の次数n。整数型，入力。N≧3とする。
A ——行列**A**を格納する。2次元倍精度実数型配列A（NM, 3），入力。
A(i, j−i+2)＝a_{ij} ($1 \leqq i \leqq n$, $1 \leqq i-1 \leqq j \leqq i+1 \leqq n$) と入れる。
E ——作業領域。1次元倍精度実数型配列E(N)，出力。
W ——固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0のとき，W(1)からW(LA)までに入れられる。特にLA＝nのときは小さい順に並んでいる。
LA——求まった固有値の数。整数型，出力。$0 \leqq LA \leqq n$となる。あるi ($2 \leqq i \leqq n$) について$a_{i-1,i}\ a_{i,i-1} < 0$となったときやIMP＝1でNM＜Nとしたときにも LA＝0となる
IMP——解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJRTNB** (NM, MM, N, A, D, E, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1)

NM, N, A, E

—— EJRTNA参照。

MM ——区間〔LB, UB〕に存在する固有値の数の上限。整数型，入力。MM≧Lとする。

D ——作業領域。1次元倍精度実数型配列D(N)，出力。

E2 ——作業領域。1次元倍精度実数型配列E2(N)，出力。

W ——固有値を格納する。1次元倍精度実数型配列W(MM)，出力。固有値は小さい順にW(1)からW(LA)までに入れられる。

IND ——作業領域。1次元整数型配列IND(N)，出力。

RV4 ——作業領域。1次元倍精度実数型配列RV4(N)，出力。

RV5 ——作業領域。1次元倍精度実数型配列RV5(N)，出力。

LA ——求まった固有値の数。整数型，出力。L＞MMのときLA＝0となるほかはLA＝Lとなる。

L ——区間〔LB, UB〕に存在する固有値の数。整数型，出力。$0 \leqq L \leqq n$となる。あるi（$2 \leqq i \leqq n$）に対し，$a_{i-1,i}\, a_{i,i-1} < 0$となったときにもL＝0となる。

LB ——固有値を捜す範囲の下界を与える。倍精度実数型，入力。LB＜UB。LB≧UBとしたときL＝0となる。

UB ——固有値を捜す範囲の上界を与える。倍精度実数型，入力。

EPS1 ——固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0とする。計算機で$1+\varepsilon=1$となる最大の数εをMACHEP（倍精度計算では約10^{-18}）とするとき，行列**A**の相対的ずれ（誤差）がMACHEP程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1をおよそMACHEP×$\|\mathbf{A}\|_1$にとる。しかし，EPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0としたときは，ほぼ上の値に置き換えられる。

CALL **EJRTNI** (NM, MM, N, A, D, E, E2, W, IND, RV4, RV5, LA, M11, LB, UB, EPS1)

NM, N, A, E

—— EJRTNA参照。

MM —— 求める固有値の数を指定する。整数型，入力。$1 \leqq MM \leqq n$とする。

D, E2, W, IND, RV4, RV5, EPS1

—— EJRTNB参照。

LA ——求まった固有値の数。整数型，出力。あるi（$2 \leqq i \leqq n$）に対し$a_{i-1,i}\, a_{i,i-1} < 0$となったとき求める固有値の一方の端（LBまたはUB）が重複固有値となったときにLA＝0となるがその他のときはLA＝MMとなる。

M11 ——求める固有値の最初の番号。整数型，入力。$1 \leqq M11 \leqq n$とする。固有値は小さい方から数えてM11番目から（M11＋MM－1）番目まで求めることになる。

LB ——求めるMM個の固有値の最初の値。倍精度実数型，出力。LB＜UBとなる。

UB ——求めるMM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJRTNE** (NM, MM, N, A, E, W, BD, IND, LA, TYPE, IDEF, EPS 1)

NM, N, A, E
——EJRTNA参照。

MM ——EJRTNI 参照。

W ——固有値を格納する。1次元倍精度実数型配列 W(N)，出力。LA＞0のとき，TYPE＝.TRUE.ならば小さい順に，.FALSE.ならば大きい順にW(1)からLA個の固有値が入れられる。

BD ——固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列 BD(N)，出力。W(j)に対する誤差は，第j段階で出力されたEPS 1に対しBD(j)≦j×EPS 1 (j＝1～LA)となっている。

IND ——EJRTNB参照。

LA ——求まった固有値の数。整数型，出力。あるiに対して$a_{i-1,i}\ a_{i,i-1}<0$となったとき，**A**が正定値でないのにIDEF＝1，TYPE＝.TRUE.としたとき，**A**が負定値でないのにIDEF＝－1，TYPE＝.FALSE.としたとき，それにMM＞nとしたときにLA＝0となるが，それ以外はLA＝MMとなる。

TYPE－固有値の大小順を指定する。論理型，入力。小さい順のとき.TRUE.，大きい順のとき.FALSE.とする。

IDEF—行列**A**の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS 1－固有値の理論的な絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS 1≧0とする。計算機で$1+\varepsilon=1$となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，W(0)＝0と仮定してEPS 1＜$\mathrm{MACHEP}\times\sum_{k=1}^{i}|W(k)-W(k-1)|$となる度にEPS 1は右辺で置き換えられる。

CALL **EJRTPA** (NM, N, A, E, W, Z, L, IMP)

NM——2次元配列AやZの大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。

N ——行列**A**を格納する。2次元倍精度実数型配列A（NM，3），入力。A(i，j－i＋2)＝a_{ij}（$1\leqq i\leqq n$，$1\leqq i-1\leqq j\leqq i+1\leqq n$）と入れる。従ってA(1，1)とA(n，3)は使用されない。

E ——作業領域。1次元倍精度実数型配列E(N)，出力。

W ——固有値を格納する。1次元倍精度実数型配列 W(N)，出力。L＞0のとき，W(1)からW(L)までに入れられる。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列 Z（NM，N），出力。L＞0の

とき，Zの第L欄までに入れられる。これらは，IMP＝0のとき正規直交化されているが，IMP＝1のとき正規化されていない。

L ——求まった固有値や固有ベクトルの数。整数型，出力。$0 \leqq L \leqq n$ となる。あるiに対し，$a_{i-1,i}\, a_{i,i-1} < 0$ または $a_{i-1,i}\, a_{i,i-1} = 0$，$a^2_{i-1,i} + a^2_{i,i-1} > 0$ となった場合やIMP＝1でN＞NMとした場合にもL＝0となる。

IMP ——解法を選択する。整数型，入力。QL法を用いるとき0，陰的QL法を用いるとき1とする。

CALL **EJRTPB** (NM, MM, N, A, D, E, E2, W, Z, RV1, RV2, RV3, RV4, RV5, RV6, LA, L, LB, UB, EPS1)

NM, N, A, E
——EJRTPA参照。

MM ——区間〔LB，UB〕に存在する固有値の数の上限。整数型，入力。MM≧LA。

D ——作業領域。1次元倍精度実数型配列D(N)，出力。

E2 ——作業領域。1次元倍精度実数型配列E2(N)，出力。

W ——固有値を格納する。1次元倍精度実数型配列W(MM)，出力。L＞0のとき，W(1)からW(L)までに入れられる。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, MM)，出力。L＞0のとき，Zの第L欄までに入れられる。これらは正規されていない。

RV1 ——作業領域。1次元倍精度実数型配列RV1(N)，出力。

RV2 ——作業領域。1次元倍精度実数型配列RV2(N)，出力。

RV3 ——作業領域。1次元倍精度実数型配列RV3(N)，出力。

RV4 ——作業領域。1次元倍精度実数型配列RV4(N)，出力。

RV5 ——作業領域。1次元倍精度実数型配列RV5(N)，出力。

RV6 ——作業領域。1次元倍精度実数型配列RV6(N)，出力。

LA ——区間〔LB，UB〕に存在する固有値の数。整数型，出力。$0 \leqq LA \leqq n$ となる。あるi（$2 \leqq i \leqq n$）に対し $a_{i-1,i}\, a_{i,i-1} < 0$ または $a_{i-1,i}\, a_{i,i-1} = 0$，$a^2_{i-1,i} + a^2_{i,i-1} > 0$ となったときにもLA＝0となる。

L ——求まった固有値と固有ベクトルの数。整数型，出力。$0 \leqq L \leqq MM$ となる。LA＞MMとなったときにもL＝0となる。

LB ——固有値を捜す範囲の下界を与える。倍精度実数型，入力。LB＜UBとする。LB≧UBとしたときLA＝0となる。

UB ——固有値を捜す範囲の上界を与える。倍精度実数型，入力。

EPS1 ——固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0とする。計算機で $1 + \varepsilon = 1$ となる最大の数 ε をMACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，行列**A**の相対的ずれ（誤差）がMACHEP程度であるときの固有値のずれ程度の精度で求めようとすればEPS1をおよそ MACHEP×$\|\mathbf{A}\|_1$ にとる。しかし

EPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0としたときはほぼ上の値に置き換えられる。

CALL **EJRTPI** (NM, MM, N, A, D, E, E2, W, Z, IND, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1)

NM, N, A, E
——EJRTPA参照。

MM——求める固有値および固有ベクトルの数。整数型，入力。1≦MM≦nとする。

D, E2, RV1, RV2, RV3, RV4, RV5, RV6, EPS1
——EJRTPB参照。

W——固有値を格納する。1次元倍精度実数型配列W(MM)，出力。LA＞0のとき，小さい順にW(1)から入れられる。

Z——固有ベクトルを格納する。2次元倍精度実数型Z(NM, MM)，出力。W(j)に対応する固有ベクトルはZの第j欄に入れられる。これらは正規化されていない。

IND——作業領域。1次元倍精度実数型配列IND(N)，出力。

LA——求まった固有値の数。整数型，出力。あるiに対し，$a_{i-1,i}\ a_{i,i-1} < 0$ または $a_{i-1,i}\ a_{i,i-1} = 0$，$a^2_{i-1,i} + a^2_{i,i-1} > 0$ となった場合か，求める固有値の一方の端(LBまたはUB)が重複固有値となった場合にLA＝0となるほかはLA＝MMとなる。

LE——求まった固有ベクトルの数。整数型，出力。LE＝0またはLE＝MMとなる。

M11——求める固有値の最初の番号。整数型，入力。1≦M11≦nとする。固有値は小さい方から数えてM11番目から(M11＋MM－1)番目まで求めることになる。

LB——求めるMM個の固有値の最初の値。倍精度実数型，出力。LB＜UBとする。

UB——求めるMM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJRTPE** (NM, MM, N, A, D, E, E2, W, Z, BD, IND, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, TYPE, IDEF, EPS1)

NM, N, A, E
——EJRTPA参照。

MM, Z, IND, LE
——EJRTPI参照。

D, E2, RV1, RV2, RV3, RV4, RV5, RV6
——EJRTPB参照。

W——固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0のとき，TYPE＝.TRUB.ならば小さい順に，.FALSE.ならば大きい順にW(1)から入れられる。

BD ―― 固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列BD(N)，出力。W(j)に対する誤差は，第j段階で出力されたEPS 1に対しBD(j)≦j ×EPS 1(j＝1～MM)となっている。

LA ―― 求まった固有値の数。整数型，出力。あるiに対し$a_{i-1,i}\ a_{i,i-1}<0$または$a_{i-1,i}\ a_{i,i-1}=0$，$a^2_{i-1,i}+a^2_{i,i-1}>0$となったとき，行列**A**が正定値でないのにIDEF＝1，TYPE＝.TRUE.または負定値でないのにIDEF＝－1，TYPE＝.FALSE.としたときにLA＝0となるほかはLA＝MMとなる。

TYPE―固有値の大小順を指定する。論理型，入力。小さい順のとき.TRUE.，大きい順のとき.FALSE.とする。

IDEF―行列**A**の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS 1―固有値の理論的な絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS 1≧0とする。計算機で$1+\varepsilon=1$となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，W(0)＝0と仮定してEPS 1＜MACHEP×$\sum_{k=1}^{j}|W(k)-W(k-1)|$となる度にEPS 1は右辺で置き換えられる。

CALL **EJSNAN** (NM, N, A, E, W, LA, IMP)

NM―― 2次元配列Aの大きさを定める第1の整合寸法。整数型，入力。MM≧nとする。

N ―― 行列**A**の次数n。整数型，入力。N≧3とする。

A ―― 行列**A**を格納する。2次元倍精度実数型配列A(NM, N)，入出力。**A**の左下3角部分をA(i, j)＝a_{ij} (1≦j≦i ≦n)と入れる。

E ―― 作業領域。1次元倍精度実数型配列E(N)，出力。

W ―― 固有値を格納する。1次倍精度実数型配列W(N)，出力。LA＞0のとき，固有値はW(j) (j＝1～LA)に入れられる。

LA ―― 求まった固有値の数。整数型，出力。0≦LA≦nとなる。

IMP―― 解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJSNAC** (NM, N, A, E, W, Z, L, IMP)

NM―― 2次元配列AやZの大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。

N, A, E, IMP
―― EJSNAN参照。

W ―― 固有値を格納する。1次元倍精度実数型配列W(N)，出力。L＞0のとき，固有値はW(j) (j＝1～L)に入れられる。

Z ―― 固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, N)，出力。L＞0のとき，W(j) (j＝1～L)に対応する固有ベクトルがZの第j欄に入れられる。これらは

正規直交化されている。

L ——求まった固有値および固有値ベクトルの数。整数型，出力。$0 \leqq L \leqq n$ となる。IMP＝0でN＞NMとした場合にもL＝0となる。

CALL **EJSNBN** (NM, MM, N, A, D, E, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1)

NM, N, A, E
——EJSNAN参照。

MM——求める固有値の数の上限。整数型，入力。$L \leqq MM \leqq n$ とする。

D ——作業領域。1次元倍精度実数型配列D(N)，出力。

E2——作業領域。1次元倍精度実数型配列E2(N)，出力。

W ——固有値を格納する。1次元倍精度実数型配列W（MM），出力。固有値は小さい順にW(1)からW(LA）までに入れられる。

IND —作業領域。1次元整数型配列IND(N)，出力。

RV4 —作業領域。1次元倍精度実数型配列RV4(N)，出力。

RV5 —作業領域。1次元倍精度実数型配列RV5(N)，出力。

LA——求まった固有値の数。整数型，出力。L＞MMのときLA＝0となるほかはLA＝Lとなる。

L ——区間〔LB, UB）に存在する固有値の数。整数型，出力。$0 \leqq L \leqq n$ となる。

LB ——固有値を捜す範囲の下界を与える。倍精度実数型，入力。LB＜UBとする。LB≧UBとしたときはL＝0となる。

UB——固有値を捜す範囲の上界を与える。倍精度実数型，入力。

EPS1—固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0とする。計算機で$1+\varepsilon=1$となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，行列**A**の相対的ずれ（誤差）がMACHEP程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1をおよそMACHEP×$\|\mathbf{A}\|_1$にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0としたときは,ほぼ上の値に置き換えられる。

CALL **EJSNBC** (NM, MM, N, A, D, E, E2, W, Z, RV1, RV2, RV3, RV4, RV5, RV6, LA, L, LB, UB, EPS1)

NM,W—EJSNAC参照。

MM ——求める固有値や固有ベクトルの数の上限。整数型，入力。LA≦MM≦nとする。

N,A,E—EJSNAN参照。

D, E2, RV4, RV5, LB, UB, EPS1
——EJSNBN参照。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, MM)，出力。L＞0のとき，W(j)（j＝1～L）に対応する固有ベクトルがZの第j欄に入れられる。これは正規直交化されている。

RV1——作業領域。1次元倍精度実数型配列RV1(N)，出力。

RV2——作業領域。1次元倍精度実数型配列RV2(N)，出力。

RV3——作業領域。1次元倍精度実数型配列RV3(N)，出力。

RV6——作業領域。1次元倍精度実数型配列RV6(N)，出力。

LA ——区間〔LB, UB）に存在する固有値の数。整数型，出力。0≦LA≦nとなる。

L ——求まった固有値および固有ベクトルの数。整数型，出力。0≦L≦MMとなる。
LA≧MMとなったときにもL＝0となる。

CALL **EJSNIN** (NM, MM, N, A, D, E, E2, W, IND, RV4, RV5, LA, M11, LB, UB, EPS1)

NM, N, A, E
——EJSNAN参照。

MM——求める固有値の数。整数型，入力。1≦MM≦nとする。

D, E2, W, IND, RV4, RV5, EPS1
——EJSNBN参照。

LA ——求まった固有値の数。整数型，出力。求める固有値の一方の端（LBまたはUB）が重複固有値となったときLA＝0となるほかはLA＝MMとなる。

M11——求める固有値の最初の番号。整数型，入力。1≦M11≦nとする。固有値は小さい方から数えてM11番目から（M11＋MM－1）番目まで求めることになる。

LB ——求めるMM個の固有値の最初の値。倍精度実数型，出力。LB＜UBとなる。

UB ——求めるMM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJSNIC** (NM, MM, N, A, D, E, E2, W, Z, IND, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1)

NM——EJSNAC参照。

MM——求める固有値や固有ベクトルの数。整数型，入力。1≦MM≦nとする。

N, A, E
——EJSNAN参照。

D, E2, W, IND, RV4, RV5, EPS1
——EJSNBN参照。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, MM)，出力。LE＞0のとき，W(j)（j＝1～LE）に対応する固有ベクトルがZの第j欄に入れられる。これらは正規直交化されている。

RV1, RV2, RV3, RV6,

——EJSNBC参照

LA, M11, LB, UB

——EJSNIN参照。

LE——求まった固有ベクトルの数。整数型，出力。LE＝0またはLE＝MMとなる。

CALL **EJSNEN** (NM, MM, N, A, E, W, BD, IND, LA, TYPE, IDEF, EPS1)

NM, N, A, E

——EJSNAN参照。

MM——EJSNIN参照。

W——固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0のとき，TYPE＝.TRUE.ならば小さい順に，.FALSE.ならば大きい順にW(1)からLA個の固有値が入れられる。

BD——固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列BD(N)，出力。W(j)に対する誤差は，第j段階で出力されたEPS1に対しBD(j)≦j×EPS1(j＝1～LA)となっている。

IND——EJSNBN参照。

LA——求まった固有値の数。整数型，出力。0≦LA≦nとなる。行列**A**が正定値でないのにIDEF＝1，TYPE＝.TRUE.としたとき，**A**が負定値でないのにIDEF＝－1，TYPE＝.FALSE.としたとき，それにMM＞nとしたときLA＝0となる。

TYPE－固有値の大小順を指定する。論理型，入力。小さい順のとき.TRUE.，大きい順のとき.FALSE.とする。

IDEF—行列**A**の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS1—固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS1≧0とする。計算機で$1+\varepsilon=1$となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，W(0)＝0と仮定してEPS1＜MACHEP×$\sum_{k=1}^{j}|W(k)-W(k-1)|$となる度にEPS1は右辺で置き換えられる。

CALL **EJSNEC** (NM, MM, N, A, D, E, E2, W, Z, BD, IND, RV1, RV2, RV3, RV4, RV6, LA, LE, TYPE, IDEF, EPS1)

NM——EJSNAC参照。

MM, Z, LE

——EJSNIC参照。

N,A,E－EJSNAN参照。

D, E2, IND, RV4
——EJSNBN参照。
W, BD, LA, TYPE, IDEF, EPS1
——EJSNEN参照。
RV1, RV2, RV3, RV6
——EJSNBC参照。

CALL **EJSPAN** (N, A, E, W, LA, IMP)

N ——行列**A**の次数n。整数型，入力。N≧3とする。
A ——行列**A**を格納する。1次元倍精度実数型配列A(NN)（NN＝n(n+1)／2)，入出力。**A**の左下3角部分を行に沿ってa_{11}，a_{21}，a_{22}，a_{31}，……，a_{33}，……，a_{n1}，…a_{nn}の順にA(1)から入れる。
E ——作業領域。1次元倍精度実数型配列E(N)，出力。
W ——固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0のとき，固有値はW(j)（j＝1～LA）に入れられる。
LA——求まった固有値の数。整数型，出力。0≦LA≦nとなる。
IMP——解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJSPAC** (NM, N, A, E, W, Z, L, IMP)

NM——2次元配列Zの大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。
N, A, E, IMP
——EJSPAN参照。
W ——固有値を格納する。1次元倍精度実数型配列W(N)，出力。L＞0のとき，固有値はW(j)（j＝1～L）に入れられる。
Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z(NM，N)，出力。L＞0のとき，W(j)（j＝1～L）に対応する固有ベクトルがZの第j欄に入れられる。これは正規直交化されている。
L ——求まった固有値および固有ベクトルの数。整数型，出力。0≦L≦nとなる。IMP=0でN＞NMとした場合にもL＝0となる。

CALL **EJSPBN** (MM, N, D, E, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1)

MM——求める固有値の数の上限。整数型，入力。L≦MM≦nとする。
N, A, E
——EJSPAN参照。

D ——作業領域。1次元倍精度実数型配列D(N)，出力。

E2 ——作業領域。1次元倍精度実数型配列E2(N)，出力。

W ——固有値を格納する。1次元倍精度実数型配列W(MM)，出力。固有値は小さい順にW(1)からW(LA)までに入れられる。

IND ——作業領域。1次元整数型配列IND(N)，出力。

RV4 ——作業領域。1次元倍精度実数型配列RV4(N)，出力。

RV5 ——作業領域。1次元倍精度実数型配列RV5(N)，出力。

LA ——求まった固有値の数。整数型，出力。L＞MMのときLA＝0となるほかはLA＝Lとなる。

L ——区間〔LB，UB〕に存在する固有値の数。整数型，出力。$0 \leqq L \leqq n$ となる。

LB ——固有値を捜す範囲の下界を与える。倍精度実数型，入力。LB＜UBとする。LB≧UBとしたときはL＝0となる。

UB ——固有値を捜す範囲の上界を与える。倍精度実数型，入力。

EPS1 ——固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0とする。計算機で$1+\varepsilon=1$となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，行列**A**の相対的ずれ（誤差）がMACHEP程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1をおよそ$\mathrm{MACHEP} \times \|\mathbf{A}\|_1$にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0としたときは，ほぼ上の値に置き換えられる。

CALL **EJSPBC** (NM, MM, N, A, D, E, E2, W, Z, RV1, RV2, RV3, RV4, RV5, RV6, LA, L, LB, UB, EPS1)

NM, W——EJSPAC参照。

MM ——求める固有値や固有ベクトルの数の上限。整数型，入力。LA≦MM≦nとする。

N, A, E

——EJSPAN参照。

D, E2, RV4, RV5, LB, UB, EPS1

——EJSPBN参照。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, MM)，出力。L＞0のとき，W(j)（j＝1～L）に対応する固有ベクトルがZの第j欄に入れられる。これらは正規直交化されている。

RV1——作業領域。1次元倍精度実数型配列RV1(N)，出力。

RV2 ——作業領域。1次元倍精度実数型配列RV2(N)，出力。

RV3——作業領域。1次元倍精度実数型配列RV3(N)，出力。

RV6 ——作業領域。1次元倍精度実数型配列RV6(N)，出力。

LA ——区間〔LB，UB〕に存在する固有値の数。整数型，出力。$0 \leqq LA \leqq n$ となる。

L ——求まった固有値および固有ベクトルの数。整数型，出力。$0 \leqq L \leqq LA$。L＜LAと

なるのは，LA＞MMとなったときと一部の解が収束しなかったときである。

CALL **EJSPIN**（MM, N, A, D, E, E2, W, IND, RV4, RV5, LA, M11, LB, UB, EPS1）

MM ——求める固有値の数。整数型，入力。1≦MM≦n とする。

N, A, E
——EJSPAN参照。

D, E2, W, IND, RV4, RV5, EPS1
——EJSPBN参照。

LA ——求まった固有値の数。整数型，出力。求める固有値の一方の端（LB またはUB）が重複固有値となったときLA＝0となるほかはLA＝MMとなる。

M11——求める固有値の最初の番号。整数型，入力。1≦M11≦n とする。固有値は小さい方から数えてM11番目から（M11＋MM－1）番目まで求めることになる。

LB ——求めるMM個の固有値の最初の値。倍精度実数型，出力。LB＜UB となる。

UB ——求めるMM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJSPIC**（NM, MM, N, A, D, E, E2, W, Z, IND, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1）

NM ——EJSPAC参照。

MM ——求める固有値や固有ベクトルの数。整数型，入力。1≦MM≦n とする。

N, A, E—EJSPAN参照。

D, E2, W, IND, RV4, RV5, EPS1
——EJSPBN参照。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z（NM, MM），出力。LE＞0のとき，W(j)（j＝1～LE）に対応する固有ベクトルがZの第j欄に入れられる。これらは正規直交化されている。

RV1, RV2, RV3, RV6
——EJSPBC参照。

LA, M11, LB, UB
——EJSPIN参照。

LE ——求まった固有ベクトルの数。整数型，出力。LE＝0　またはLE＝MMとなる。

CALL **EJSPEN**（MM, N, A, E, W, BD, IND, LA, TYPE, IDEF, EPS1）

MM ―― EJSPIN参照。

N,A,E ― EJSPAN参照。

W ―― 固有値を格納する。1次元倍精度実数型配列 W(N)，出力。LA＞0のときTYPE＝.TRUE.ならば小さい順に，.FALSE.ならば大きい順に W(1)からLA個の固有値が入れられる。

BD ―― 固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列BD(N)，出力。W(j)に対する誤差は，第j段階で出力されたEPS1に対しBD(j)≦ j ×EPS1（j＝1～LA）となっている。

IND ―― EJSPBN参照。

LA ―― 求まった固有値の数。整数型，出力。0≦LA≦n となる。行列**A**が正定値でないのに IDEF＝1，TYPE＝.TRUE.としたとき，**A**が負定値でないのに IDEF＝－1，TYPE＝.FALSE.としたとき，それにMM＞n としたときLA＝0となる。

TYPE ― 固有値の大小順を指定する。論理型，入力。小さい順のとき.TRUE.，大きい順のとき.FALSE.とする。

IDEF ― 行列**A**の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS1 ― 固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS1≧0とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，W(0)＝0と仮定してEPS1＜MACHEP×$\sum_{k=1}^{j}|W(k)-W(k-1)|$ となる度にEPS1は右辺で置き換えられる。

CALL **EJSPEC** (NM, MM, N, A, D, E, E2, W, Z, BD, IND, RV1, RV2, RV3, RV4, RV6, LA, LE, TYPE, IDEF, EPS1)

NM ―― EJSPAC参照。

MM, Z, LE
―― EJSPIC参照。

N,A,E ― EJSPAN参照。

D, E2, IND, RV4
―― EJSPBA参照。

W, BD, LA, TYPE, IDEF, EPS1
―― EJSPEN参照。

RV1, RV2, RV3, RV6
―― EJSPBC参照。

CALL **EJSBAN** (NM, N, MB, A, E, W, LA, IMP)

NM ―― 2次元配列Aの大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。

N ――行列**A**の次数n。整数型，入力。N≧3とする。

MB――行列**A**の帯の半幅。整数型，入力。（2×MB－1）が帯幅で，2≦MB≦nとする。

A ――行列**A**を格納する。2次元倍精度実数型配列A(NM, MB)，入出力。行列**A**はA(i, j＋MB－i)＝a_{ij}（1≦j≦i≦n，i－j≦MB－1）と入れる。

E ――作業領域。1次元倍精度実数型配列E(N)，出力。

W ――固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0のときW(j)（j＝1～LA）に入れられる。特にLA＝nのときは小さい順に並んでいる。

LA ――求まった固有値の数。整数型，出力。0≦LA≦nとなる。

IMP――解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJSBAC**（NM, N, MB, A, E, W, Z, L, IMP）

NM ――2次元配列AやZの大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。

N, MB, A, E, IMP
――EJSBAN参照。

W ――固有値を格納する。1次元倍精度実数型配列W(N)，出力。L＞0のときW(j)（j＝1～L）に入られる。特にL＝nのときは小さい順になっている。

Z ――固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, N) 出力。W(j)(j=1～L）に対応する固有ベクトルがZの第j欄に入れられる。これらは正規直交化されている。

L ――求まった固有値や固有ベクトルの数。整数型，出力。0≦L≦nとなる。IMP＝0でN＞NMとしたときにもL＝0となる。

CALL **EJSBBN**（NM, MM, N, MB, A, D, E, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1）

NM, N, MB, A, E
――EJSBAN参照。

MM ――区間〔LB，UB）に存在する固有値の数の上限。整数型，入力。L≦MM≦nとする。

D ――作業領域。1次元倍精度実数型配列D(N)，出力。

E2 ――作業領域。1次元倍精度実数型配列E2(N)，出力。

W ――固有値を格納する。1次元倍精度実数型配列W(MM)，出力。LA＞0のとき，W(j)（j＝1～LA）に小さい順に入れられる。

IND ――作業領域。1次元整数型配列IND(N)，出力。

RV4 ――作業領域。1次元倍精度実数型配列RV4(N)，出力。

RV5 — 作業領域。1次元倍精度実数型配列RV5(N)，出力。

LA — 求まった固有値の数。整数型，出力。L＞MM のときLA＝0となるほかはLA＝Lとなる。

L — 区間〔LB，UB〕に実際に存在する固有値の数。整数型，出力。0≦L≦n となる。

LB — 固有値を捜す範囲の下界。倍精度実数型，入力。LB＜UB とする。LB≧UB としたときはLA＝0となる。

UB — 固有値を捜す範囲の上界。倍精度実数型，入力。

EPS1 — 固有値の絶対誤差の許容範囲。倍精度実数型，入出力。EPS1＞0とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，行列**A**の相対的ずれ（誤差）が MACHEP 程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1をおよそ MACHEP×$\|\mathbf{A}\|_1$ にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0としたときは，ほぼ上の値に置き換えられる。

CALL **EJSBBC**（NM, MM, N, MB, A, AS, D, E, E2, W, Z, IND, RV1, RV6, RV, LA, LE, L, LB, UB, EPS1）

NM — EJSBAC 参照。

MM, D, E2, IND, LA, L, LB, LU, EPS1, W
— EJSBBN参照。

N, MB, A, E, W
— EJSBAN参照。

AS — 作業領域。2次元倍精度実数型配列 AS（NM, MB），出力。

Z — 固有ベクトルを格納する。2次元倍精度実数型配列 Z（NM, MM），出力。LE＞0のとき，W(j)(j＝1～LE）に対応する固有ベクトルがZの第j欄に入れられる。これらは正規直交化されている。

RV1 — 作業領域。1次元倍精度実数型配列RV1(N)，出力。

RV6 — 作業領域。1次元倍精度実数型配列RV6(N)，出力。

RV — 作業領域。1次元倍精度実数型配列RV(NV)，出力。NV＝n×(2×MB－1）とする。

LE — 求まった固有ベクトルの数。整数型，出力。LE＝0 またはLE＝LAとなる。

CALL **EJSBIN**（NM, MM, N, NB, A, D, E, E2, W, IND, RV4, RV5, LA, M11, LB, UB, EPS1）

NM, N, MB, A, E
—EJSBAN参照。

MM ―― 求める固有値の数。整数型，入力。1≦MM≦n とする。

D, E2, IND, RV4, RV5, EPS1

――EJSBBN参照。

W ―― 固有値を格納する。1次元倍精度実数型配列W(MM)，出力。LA＞0のとき，W(j)(j＝1～LA)に小さい順に入れられる。

LA ―― 求まった固有値の数。整数型，出力。求める固有値となったときLA＝0となるほかはLA＝MMとなる。

M11 ―― 求める固有値の最初の番号。整数型，入力。1≦M11≦n とする。固有値は小さい方から数えてM11番目から(M11＋MM－1)番目まで求めることになる。

LB ―― 求めるMM個の固有値の最初の値。倍精度実数型，出力。LB＜UB となる。

UB ―― 求めるMM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJSBIC** (NM, MM, N, MB, A, AS, D, E, E2, W, Z, IND, RV4, RV5, RV6, RV, LA, LE, M11, LB, UB, EPS1)

NM ―― EJSBAC 参照。

MM ―― 求める固有値や固有ベクトルの数。整数型，入力。1≦MM≦n とする。

N, MB, A, E

――EJSBAN参照。

AS, Z, RV6, RV, LE

――EJSBBC 参照。

D, E2, IND, RV4, RV5, EPS1

――EJSBBN参照。

W, LA, M11, LB, UB

――EJSBIN参照。

CALL **EJSBEN** (NM, MM, N, MB, A, E, W, BD, IND, LA, TYPE, IDEF, EPS1)

NM, N, MB, A, E

――EJSBAN参照。

MM ―― EJSBIN参照。

W ―― 固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0のとき，TYPE＝.TRUE. なら小さい順に，TYPE＝.FALSE. ならば大きい順にW(1)からLA個の固有値が入れられる。

BD ―― 固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列BD(N)，出力。W(j)に対する誤差は，第j 段階で出力されたEPS1に対し，BD(j)≦j×EPS1 (j＝1～LA)となっている。

IND——EJSBBN参照。

LA——求まった固有値の数。整数型，出力。行列**A**が正定値でないのにIDEF＝1，TYPE＝.TRUE.としたとき，**A**が負定値でないのにIDEF＝−1，TYPE=.FALSE.としたとき，それにMM＞nとしたときLA＝0なるほかはLA＝MMとなる。

TYPE—固有値の大小順を指定する。論理型，入力。小さい順のとき．TRUE．，大きい順のとき．FALSE．とする。

IDEF—行列**A**の定値性を示す。整数型，入力。正定値のとき1，負定値のとき−1，定値でないときや不明のとき0とする。

EPS1—固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS1≧0とする。計算機で$1+\varepsilon=1$となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，W(0)＝0と仮定してEPS1＜MACHEP$\times\sum_{k=1}^{j}|W(k)-W(k-1)|$となる度にEPS1は右辺で置き換えられる。

CALL **EJSBEC** (NM, MM, N, MB, A, AS, E, W, Z, BD, IND, RV6, RV, LA, LE, TYPE, IDEF, EPS1)

NM——EJSBAC参照。

MM——EJSBIC参照。

N, MB, A, E
——EJSBAN参照。

AS, Z, RV6, RV, LB
——EJSBBC参照。

W, BD, LA, TYPE, IDEF, EPS1
——EJSBEN参照。

IND——EJSBBN参照。

CALL **EJSBNN** (NM, MM, N, MB, A, W, RV, LA, T, R)

NM, N, MB, A
——EJSBAN参照。

MM——EJSBIN参照。

W——固有値を格納する。1次元倍精度実数型配列W(MM)，出力。LA＞0のとき，求まったTに最も近いLA個の固有値がW(1)から小さい順に並べ替えられている。

RV——作業領域。1次元倍精度実数型配列RV(NV)，出力。$NV=2\times MB^2+4\times MB-3$とする。

LA——求まった固有値の数。整数型，出力。0≦LA≦MMとなる。

T——求めようとする固有値の目標となる値。倍精度実数型，入出力。

R——行列変換時の零判定値。倍精度実数型，入出力。R≧0とする。

CALL **EJSBNC** (NM, MM, N, MB, A, AS, W, Z, RV6, RV, LA, LE, T, R)

NM ── EJSBAC 参照。

MM ── EJSBIC 参照。

N, MB, A

── EJSBAN参照。

AS, Z, RV6, LE

── EJSBBC 参照。

W, LA, T, R

── EJSBNN参照。

RV ──作業領域。1次元倍精度実数型配列 RV (NV)，出力。NV＝max {2×MB^2 +4 ×MB−3，n×(2×MB−1)}とする。

CALL **EJSTAN** (N, E, W, LA, IMP)

N ──行列**A**の次数 n 。整数型，入力。N≧3 とする。

E ──**A**の下対角部分を格納する。1次元倍精度実数型配列 E(N)，入出力。E(i)＝$a_{i,i-1}$ (i ＝2〜n) とする。

W ──**A**の対角部分や固有値を格納する。1次元倍精度実数型配列 W(N)，入出力。W(i)＝a_{ii} (i ＝1〜n) と対角部分を入れると求まった固有値が W(j) (j＝1〜LA) に出力される。LA＝n のときに限り，固有値は小さい順に並んでいる。

LA ──求まった固有値の数。整数型，出力。0≦LA≦n とする。

IMP──解法を選択する。整数型，入力。QL 法のとき 0，陰的QL 法のとき 1 とする。

CALL **EJSTAC** (NM, N, E, W, Z, L, IMP)

NM ── 2次元配列 Z の大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

N, E, IMP

── EJSTAN参照。

W ──**A**の対角部分や固有値を格納する。1次元倍精度実数型配列 W(N)，入出力。W(i)＝a_{ii} (i ＝1〜n) と対角部分を入れると求まった固有値が W(j) (j＝1〜L) に出力される。L ＝n のときに限り，固有値は小さい順に並んでいる。

Z ──固有ベクトルを格納する。2次元倍精度実数型配列 Z (NM, N)，出力。W(j) に対応する固有ベクトルが Z の第 j 欄に入れられる。これらは正規直交化されている。

L ──求まった固有値や固有ベクトルの数。整数型，出力。0≦L ≦n となる。IMP ＝1 で N＞NMとしたときにも L＝0 となる。

CALL **EJSTBN** (MM, N, D, E, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1)

MM ―― 区間〔LB, UB〕に存在する固有値の数の上限。整数型，入力。L≦MM≦n とする。

N,E ―― EJSTAN参照。

D ―― 行列**A**の対角部分を格納する。1次元倍精度実数型配列D(N)，入力。$D(i)=a_{ii}$ (i＝1～n) とする。

E2 ―― 作業領域。1次元倍精度実数型配列E2(N)，出力。

W ―― 固有値を格納する。1次元倍精度実数型配列W(MM)，出力。LA＞0のとき，W(j) (j＝1～LA) に小さい順に入れられる。

IND ―― 作業領域。1次元整数型配列IND(N)，出力。

RV4 ―― 作業領域。1次元倍精度実数型配列RV4(N)，出力。

RV5 ―― 作業領域。1次元倍精度実数型配列RV5(N)，出力。

LA ―― 求まった固有値の数。整数型，出力。L＞MMのときLA＝0となるほかはLA＝Lとなる。

L ―― 区間〔LB, UB〕に実際に存在する固有値の数。整数型，出力。0≦L≦n となる。

LB ―― 固有値を捜す範囲の下界。倍精度実数型，入力。LB＜UB とする。LB≧UB としたときはLA＝0となる。

UB ―― 固有値を捜す範囲の上界。倍精度実数型，入力。

EPS1 ― 固有値の絶対誤差の許容範囲。倍精度実数型，入出力。EPS1＞0とする。計算機で 1＋ε＝1 となる最大の数εを MACHEP（倍精度計算ではおよそ 10^{-18} ）とするとき，行列**A**の相対的ずれ（誤差）が MACHEP 程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1をおよそ MACHEP×$\|\mathbf{A}\|_1$ にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0としたときは，ほぼ上の値に置き換えられる。

CALL **EJSTBC** (NM, MM, N, D, E, E2, W, Z, RV1, RV2, RV3, RV4, RV5, RV6, LA, L, LB, UB, EPS1)

NM ―― EJSTAC参照。

MM, D, E2, RV4, RV5, LB, UB, EPS1
―― EJSTBN参照。

N,E ―― EJSTAN参照。

W ―― 固有値を格納する。1次元倍精度実数型配列W(MM)，出力。L＞0のとき，W(j) (j＝1～L) に小さい順に入れられる。

Z ―― 固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, MM)，出力。L＞0のとき，W(j)に対応する固有ベクトルがZの第j欄に入れられる。これらは正規直交化されている。

RV1 ―― 作業領域。1次元倍精度実数型配列RV1(N)，出力。

RV2 ——作業領域。1次元倍精度実数型配列 RV2(N), 出力。
RV3 ——作業領域。1次元倍精度実数型配列 RV3(N), 出力。
RV6 ——作業領域。1次元倍精度実数型配列 RV6(N), 出力。
LA ——区間〔LB, UB〕に実際に存在する固有値の数。整数型, 出力。$0 \leqq LA \leqq n$ となる。
L ——求まった固有値や固有ベクトルの数。整数型, 出力。$0 \leqq L \leqq LA$ となる。$LA > MM$ となったときにもLA＝0となる。

CALL **EJSTIN** (MM, N, D, E, E2, W, IND, RV4, RV5, LA, M11, LB, UB, EPS1)

MM ——求める固有値の数。整数型, 入力。$1 \leqq MM \leqq n$ とする。
N,E ——EJSTAN参照。
D, E2, W, IND, RV4, RV5, EPS1
——EJSTBN参照。
LA ——求まった固有値の数。整数型, 出力。求める固有値の一方の端（LBまたはUB）が重複固有値となったときLA＝0, そうでないときLA＝MMとなる。
M11 ——求める固有値の最初の番号。整数型, 入力。$1 \leqq M11 \leqq n$ とする。固有値は小さい方から数えてM11番目から（M11＋MM－1）番目まで求めることになる。
LB ——求めるMM個の固有値の最初の値。倍精度実数型, 出力。LB＜UBとなる。
UB ——求めるMM個の固有値の最後の値。倍精度実数型, 出力。

CALL **EJSTIC** (NM, MM, N, D, E, E2, W, Z, IND, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1)

NM ——EJSTAC参照。
MM ——求める固有値や固有ベクトルの数。整数型, 入力。$1 \leqq MM \leqq n$ とする。
N,E ——EJSTAN参照。
D, E2, W, IND, RV4, RV5, EPS1
——EJSTBN参照。
Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z（NM, MM）, 出力。LE＞0のとき, W(j) に対応する固有ベクトルがZの第j欄に入れられる。これらは正規直交化されている。
RV1, RV2, RV3, RV6
——EJSTBC参照。
LA, M11, LB, UB
——EJSTIN参照。

LE ——求まった固有ベクトルの数。整数型，出力。LE＝0またはLE＝LAとなる。

CALL **EJSTEN**（MM, N, E, W, BD, IND, LA, TYPE, IDEP, EPS 1）

MM——EJSTIN参照。

N,E——EJSTAN参照。

W ——行列**A**の対角部分や固有値を格納する。1次元倍精度実数型配列 W(N)，入出力。W(i)＝a_{ii}（i ＝1～n）と対角部分を入れると求まった固有値がW(j)（j ＝1～LA）に出力される。TYPE＝. TRUE . なら小さい順に，TYPE＝. FALSE . なら大きい順に並んでいる。

BD ——固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列 BD(N)，出力。W(j)に対する誤差は，第 j 段階で出力されたEPS 1に対し，BD(j)≦j ×EPS 1(j＝1～LA）となっている。

IND——EJSTBN参照。

LA ——求まった固有値の数。整数型，出力。行列**A**が正定値でないのに IDEF＝1，TYPE＝. TRUE . としたとき，**A**が負定値でないのに IDEF＝−1，TYPE＝. FALSE . としたとき，それに MM＞n としたときLA＝0 となるほかはLA＝MMとなる。

TYPE－固有値の大小順を指定する。論理型，入力。小さい順のとき . TRUE . ，大きい順のとき . FALSE . とする。

IDEF －行列**A**の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS 1－固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS 1≧0。計算機で1＋ε＝1となる最大の数εをMACHEP（倍精度計算ではおよそ 10^{-18} ）とするとき，W(0)＝0と仮定してEPS 1＜MACHEP×$\sum_{k=1}^{j}|W(k)-W(k-1)|$ となる度にEPS 1は右辺で置き換えられる。

CALL **EJSTEC**（NM, MM, N, D, E, E 2, W, Z, BD, IND, RV 1, RV 2, RV 3, RV 4, RV 6, LA, LE, TYPE, IDEF, EPS 1）

NM——EJSTAS 参照。

MM, Z, LE
——EJSTIC参照。

N,E——EJSTAN参照。

D, E 2, IND, RV 4
——EJSTBN参照。

W ——固有値を格納する。1次元倍精度実数型配列 W(N)，出力。LA＞0のとき，TYPE＝. TRUE . なら小さい順に，TYPE＝. FALSE . なら大きい順にW(1)からLA個の

固有値が入れられる。

BD, LA, TYPE, IDEF, EPS1

——EJSTEN参照。

RV1, RV2, RV3, RV6

——EJSTBC参照。

CALL **EJCNEN** (NM, N, AR, AI, WR, WI, INT, LA)

NM ——2次元配列ARやAIの大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。

N ——行列**A**の次数n。整数型，入力。N≧3とする。

AR ——行列**A**の実数部を格納する。2次元倍精度実数型配列AR（NM，N），入出力。$\mathbf{A}=(a_{ij})=(\alpha_{ij}+i\ \beta_{ij})$ に対し，AR（i，j）$=\alpha_{ij}$ と入れる。

AI ——行列**A**の虚数部を格納する。2次元倍精度実数型配列AI（NM，N），入出力。AI（i，j）$=\beta_{ij}$ と入れる。

WR ——固有値の実数部を格納する。1次元倍精度実数型配列WR(N)，出力。$\lambda_j=\sigma_j+i\tau_j$ とするとき，LA＞0に対してWR(j)$=\sigma_j$（j＝n－LA＋1～n）と入る。

WI ——固有値の虚数部を格納する。1次元倍精度実数型配列WI(N)，出力。LA＞0に対してWI(j)$=\tau_j$（j＝n－LA＋1～n）と入る。

INT ——作業領域。1次元整数型配列INT(N)，出力。

LA ——求まった固有値の数。整数型，出力。$0\leqq LA\leqq n$ となる。

CALL **EJCNEC** (NM, N, AR, AI, WR, WI, ZR, ZI, INT, LA, LE)

NM ——2次元配列AR，AIおよびZR，ZIの大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

N, AR, AI, WR, WI, INT, LA
—— EJCNEN 参照。

ZR —— 固有ベクトルの実数部を格納する。2次元倍精度実数型配列 ZR(NM, N)，出力。WR(j)＋i×WI(j) に対応する固有ベクトルを $\mathbf{x}_j = (\xi_{ij} + i\,\eta_{ij})$ とするとき ZR(i, j)＝ξ_{ij} と入る。

ZI —— 固有ベクトルの虚数部を格納する。2次元倍精度実数型配列 ZI(NM, N)，出力。WR(j)＋i×WI(j) に対応する固有ベクトル $\mathbf{x}_j$ に対し，ZI(i, j)＝η_{ij} と入る。

LE —— 求まった固有ベクトルの数。整数型，出力。LE＝0 または LE＝n となる。これらは正規化されていない。

CALL **EJCNES** (NM, MM, N, AR, AI, ARS, AIS, WR, WI, ZR, ZI, SELECT, INT, RM1, RM2, RV1, RV2, LA, LE)

NM—— 2次元配列 AR，AI，ARS，AIS，ZR，ZI の大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

MM—— 求める固有ベクトルの数の上限。整数型，入力。MM≧LE とする。

N, AR, AI, WR, WI, INT, LA
—— EJCNEN 参照。

ARS —作業領域。2次元倍精度実数型配列 ARS(NM, N)，出力。

AIS —作業領域。2次元倍精度実数型配列 AIS(NM, N)，出力。

ZR ——固有ベクトルの実数部を格納する。2次元倍精度実数型配列 ZR(NM, MM)，出力。j の番号の若い方から，SELECT(j)＝.TRUE. と指定したベクトル $\mathbf{x}_j$ の実数部をZRの第1欄から詰めて入れる。

ZI ——固有ベクトルの虚数部を格納する。2次元倍精度実数型配列 ZI(NM, N)，出力。j の番号の若い方から，指定したベクトル $\mathbf{x}_j$ の虚数部をZI の第1欄から詰めて入れる。

SELECT
——求める固有ベクトルを指定する。1次元論理型配列SELECT(N)，入力。WR(j)＋i×WI(j) に対応する固有ベクトルを求めるとき SELECT(j)＝.TRUE.，求めないとき .FALSE. とする。しかし 1≦j≦n－LA の範囲のものを指定すると，すべてのベクトルが正常に求まらない可能性がある。

RM1—作業領域。2次元倍精度実数型配列 RM1(N, N)，出力。

RM2— 作業領域。2次元倍精度実数型配列 RM2(N, N)，出力。

RV1 ―― 作業領域。1次元倍精度実数型配列 RV1(N)，出力。
RV2 ―― 作業領域。1次元倍精度実数型配列 RV2(N)，出力。
LE ―― 求まった固有ベクトルの数。整数型，出力。指定したベクトルの数をsとするとき，
LE＝0 または LE＝min(s，MM) となる。

CALL **EJCNUN** (NM，N，AR，AI，WR，WI，ORTR，ORTI，LA)

NM，N，AR，AI，WR，WI，LA
―― EJCNEN 参照。
ORTR― 作業領域。1次元倍精度実数型配列 ORTR(N)， 出力。
ORTI ― 作業領域。1次元倍精度実数型配列 ORTI(N)， 出力。

CALL **EJCNUC** (NM，N，AR，AI，WR，WI，ZR，ZI，ORTR，ORTI，LA，LE)

NM，ZR，ZI，LE
―― EJCNEC 参照。
N，AR，AI，WR，WI，LA
―― EJCNEN 参照。
ORTR，ORTI
―― EJCNUN 参照。

CALL **EJCNUS** (NM，MM，N，AR，AI，ARS，AIS，WR，WI，ZR，ZI，SELECT，
ORTR，ORTI，RM1，RM2，RV1，RV2，LA，LE)

NM，MM，ARS，AIS，ZR，ZI，SELECT，RM1，RM2，RV1，RV2，LE
―― EJCNES 参照。
N，AR，AI，WR，WI，LA
―― EJCNEN 参照。

CALL **EJCBEN** (NM，N，AR，AI，WR，WI，INT，SCALE，LA)

NM ―― 2次元配列 AR や AI の大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。
N ―― 行列 **A** の次数n。整数型，入力。N≧3 とする。
AR ―― 行列 **A** の実数部を格納する。2次元倍精度実数型配列 AR(NM，N)，入出力。
A＝(a_{ij})＝$(\alpha_{ij}+i\beta_{ij})$ に対し，AR(i，j)＝α_{ij} と入れる。

AI —— 行列 **A** の虚数部を格納する。2次元倍精度実数型配列 AI (NM, N)，入出力。
AI(i, j) $=\beta_{ij}$ と入れる。

WR —— 固有値の実数部を格納する。1次元倍精度実数型配列 WR(N)，出力。$\lambda_j = \sigma_j + i\tau_j$ とするき，LA > 0 に対して WR(j) $=\sigma_j$ (j = n − LA + 1 ～ n) と入る。

WI —— 固有値の虚数部を格納する。1次元倍精度実数型配列 WI(N)，出力。LA > 0 に対して WI(j) $=\tau_j$ (j = n − LA + 1 ～ n) と入る。

INT—— 作業領域。1次元整数型配列 INT(N)，出力。

SCALE
—— 作業領域。1次元倍精度実数型配列 SCALE(N)，出力。

LA ——求まった固有値の数。整数型，出力。 $0 \leqq LA \leqq n$ となる。

CALL **EJCBEC** (NM, N, AR, AI, WR, WI, ZR, ZI, INT, SCALE, LA, LE)

NM —— 2次元配列 AR，AI および ZR，ZI の大きさを定める第1の整合寸法。整数型，入力。
$NM \geqq n$ とする。

N, AR, AI, WR, WI, INT, SCALE, LA
—— EJCBEN 参照。

ZR —— 固有ベクトルの実数部を格納する。2次元倍精度実数型配列 ZR (NM, N)，出力。
WR(j) + i × WI(j) に対応する固有ベクトルを $\mathbf{x}_j = (\xi_{ij} + i\ \eta_{ij})$ とするとき
ZR(i, j) $=\xi_{ij}$ と入る。

ZI —— 固有ベクトルの虚数部を格納する。2次元倍精度実数型配列 ZI (NM, N)，出力。
WR(j) + i × WI(j) に対応する固有ベクトル $\mathbf{x}_j$ に対し，ZI(i, j) $=\eta_{ij}$ と入る。

LE —— 求まった固有ベクトルの数。整数型，出力。LE = 0 または LE = n となる。これらは正規化されていない。

CALL **EJCBES** (NM, MM, N, AR, AI, ARS, AIS, WR, WI, ZR, ZI, SELECT, INT, SCALE, RM1, RM2, RV1, RV2, LA, LE)

NM —— 2次元配列 AR，AI，ARS，AIS，ZR，ZI の大きさを定める第1の整合寸法。整数型，入力。$NM \geqq n$ とする。

MM —— 求める固有ベクトルの数の上限。整数型，入力。$MM \geqq LE$ とする。

N, AR, AI, WR, WI, INT, SCALE, LA
—— EJCBEN 参照。

ARS —— 作業領域。2次元倍精度実数型配列 ARS(NM, N)，出力。

AIS —— 作業領域。2次元倍精度実数型配列 AIS (NM, N)，出力。

ZR —— 固有ベクトルの実数部を格納する。2次元倍精度実数型配列 ZR(NM, MM)，出力。j の番号の若い方から，SELECT(j) = .TRUE. と指定したベクトル $\mathbf{x}_j$ の実数部を ZR の第1欄から詰めて入れる。

ZI —— 固有ベクトルの虚数部を格納する。2次元倍精度実数型配列 ZI(NM, N)，出力。j の番号の若い方から，指定したベクトル $\mathbf{x}_j$ の虚数部を ZI の第1欄から詰めて入れる。

SELECT

—— 求める固有ベクトルを指定する。1次元論理型配列 SELECT(N)，入力。WR(j) + i × WI(j) に対応する固有ベクトルを求めるとき SELECT(j) = .TRUE.，求めないとき .FALSE. とする。しかし $1 \leqq j \leqq n - LA$ のものを指定すると，すべてのベクトルが正常に求まらない可能性がある。

RM1 — 作業領域。2次元倍精度実数型配列 RM1(N, N)，出力。

RM2 — 作業領域。2次元倍精度実数型配列 RM2(N, N)，出力。

RV1 — 作業領域。1次元倍精度実数型配列 RV1(N)，出力。

RV2 — 作業領域。1次元倍精度実数型配列 RV2(N)，出力。

LE —— 求まった固有ベクトルの数。整数型，出力。指定したベクトルの数を s とするとき，LE = 0 または LE = min(s, MM) となる。

CALL **EJCBUN** (NM, N, AR, AI, WR, WI, ORTR, ORTI, SCALE, LA)

NM, N, AR, AI, WR, WI, SCALE

—— EJCBEN 参照。

ORTR — 作業領域。1次元倍精度実数型配列 ORTR(N)，出力。

ORTI — 作業領域。1次元倍精度実数型配列 ORTI(N)，出力。

LA —— 求まった固有値の数。整数型，出力。$0 \leqq LA \leqq n$ となる。$N > NM$ としたときも 0 となる。

CALL **EJCBUC** (NM, N, AR, AI, WR, WI, ZR, ZI, ORTR, ORTI, SCALE, LA, LE)

NM, ZR, ZI, LE,

—— EJCBEC 参照。

N, AR, AI, WR, WI, SCALE

—— EJCBEN 参照。

ORTR, ORTI, LA

—— EJCBUN 参照。

CALL **EJCBUS** (NM, MM, N, AR, AI, ARS, AIS, WR, WI, ZR, ZI, SELECT, ORTR, ORTI, SCALE, RM1, RM2, RV1, RV2, LA, LE)

NM, MM, ARS, AIS, ZR, ZI, SELECT, RM1, RM2, RV1, RV2, LE
—— EJCBES 参照。

N, AR, AI, WR, WI, SCALE, LA
—— EJCBEN 参照。

CALL **EJHNAN** (NM, N, AR, AI, E, W, TAU, LA, IMP)

NM —— 2次元配列AR, AIの大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

N —— 行列**A**の次数n。整数型，入力。N≧3とする。

AR —— 行列**A**の実数部を格納する。2次元倍精度実数型配列AR(NM, N)，入出力。$\mathbf{A}=(a_{ij})=(\alpha_{ij}+i\beta_{ij})$に対し，AR(i, j)$=\alpha_{ij}$ $(1\leqq j\leqq i\leqq n)$ と入れる。

AI —— 行列**A**の虚数部を格納する。2次元倍精度実数型配列AI(NM, N)，入出力。AI(i, j)$=\beta_{ij}$ $(1\leqq j<i\leqq n)$ と入れる。

E —— 作業領域。1次元倍精度実数型配列 E(N)，出力。

W —— 固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0 のとき，W(j) (j=1～LA)に入れられる。特にLA=n のときは小さい順に並んでいる。

TAU — 作業領域。2次元倍精度実数型配列 TAU(2, N)，出力。

LA —— 求まった固有値の数。整数型，出力。0≦LA≦n となる。

IMP —— 解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJHNAC** (NM, N, AR, AI, E, W, ZR, ZI, TAU, LA, LE, IMP)

NM —— 2次元配列AR, AI, ZR, ZI の大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

N, AR, AI, E, W, TAU, IMP
—— EJHNAN 参照。

ZR —— 固有ベクトルの実数部を格納する。2次元倍精度実数型配列ZR(NM, N)，出力。W(j) に対応する固有ベクトルを$\mathbf{x}_j=(\xi_{ij}+i\eta_{ij})$とするとき，ZR(i, j)$=\xi_{ij}$ と入れられる。ベクトルは正規直交化されている。

ZI —— 固有ベクトルの虚数部を格納する。2次元倍精度実数型配列ZI(NM, N)，出力。W(j)に対応する固有ベクトル$\mathbf{x}_j$に対し，ZI(i, j)$=\eta_{ij}$ と入れられる。

LA —— 求まった固有値の数。整数型，出力。0≦LA≦n となる。IMP=0 のとき，N＞NM とした場合にもLA=0 となる。

LE —— 求まった固有ベクトルの数。整数型，出力。IMP=0で 0＜LA＜n となった場合にLE=0 となるほかは LE=LAとなる。

CALL **EJHNBN** (NM, MM, N, AR, AI, D, E, E2, W, IND, TAU, RV4, RV5, LA, L, LB, UB, EPS1)

NM, N, AR, AI, E, TAU
—— EJHNAN 参照。

MM —— 区間〔LB, UB〕に存在する固有値の数の上限。整数型，入力。$L \leqq MM \leqq n$ とする。

D —— 作業領域。1次元倍精度実数型配列 D(N)，出力。

E2 —— 作業領域。1次元倍精度実数型配列 E2(N)，出力。

W —— 固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。LA＞0 のとき W(j)（j＝1～LA）に小さい順に入れられる。

IND —— 作業領域。1次元整数型配列 IND(N)，出力。

RV4 —— 作業領域。1次元倍精度実数型配列 RV4(N)，出力。

RV5 —— 作業領域。1次元倍精度実数型配列 RV5(N)，出力。

LA —— 求まった固有値の数。整数型，出力。L＞MM のとき LA＝0 となるほかは LA＝L となる。

L —— 区間〔LB, UB〕に存在する固有値の数。整数型，出力。$0 \leqq L \leqq n$ となる。

LB —— 固有値を捜す範囲の下界。倍精度実数型，入力。LB＜UB とする。LB≧UB としたときは LA＝0 となる。

UB —— 固有値を捜す範囲の上界。倍精度実数型，入力。

EPS1 —— 固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0 とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，行列 **A** の相対的ずれ（誤差）が MACHEP 程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1 をおよそ MACHEP × $\|\mathbf{A}\|_1$ にとる。しかし EPS1 を小さくとり過ぎると余分な分割を行うので注意を要する。
EPS1≦0 としたときは，ほぼ上の値に置き換えられる。

CALL **EJHNBC** (NM, MM, N, AR, AI, D, E, E2, W, ZR, ZI, IND, TAU, RV1, RV2, RV3, RV4, RV5, RV6, LA, L, LB, UB, EPS1)

NM —— EJHNAC 参照。

MM, D, E2, IND, RV4, RV5, LB, UB, EPS1
—— EJHNBN 参照。

N, AR, AI, E, TAU
—— EJHNAN 参照。

W —— 固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。L＞0 のとき W(j)（j＝1～L）に入れられる。

ZR —— 固有ベクトルの実数部を格納する。2次元倍精度実数型配列 ZR(NM, MM)，出力。

W(j) に対応する固有ベクトルを $\mathbf{x}_j = (\xi_{ij} + i\,\eta_{ij})$ とするとき，ZR(i, j) $= \xi_{ij}$ と入れられる。ベクトルは正規直交化されている。

ZI —— 固有ベクトルの虚数部を格納する。2次元倍精度実数型配列 ZI(NM, MM)，出力。W(j) に対応する固有ベクトル $\mathbf{x}_j$ に対し，ZI(i, j) $= \eta_{ij}$ と入れられる。

RV1 —— 作業領域。1次元倍精度実数型配列 RV1(N)，出力。

RV2 —— 作業領域。1次元倍精度実数型配列 RV2(N)，出力。

RV3 —— 作業領域。1次元倍精度実数型配列 RV3(N)，出力。

RV6 —— 作業領域。1次元倍精度実数型配列 RV6(N)，出力。

LA —— 区間〔LB, UB〕に存在する固有値の数。整数型，出力。$0 \leqq LA \leqq n$ となる。

L —— 求まった固有値や固有ベクトルの数。整数値，出力。$0 \leqq L \leqq MM$。LA>MM となった場合にも L = 0 となる。

CALL **EJHNIN** (NM, MM, N, AR, AI, D, E, E2, W, IND, TAU, RV4, RV5, LA, M11, LB, UB, EPS1)

NM, N, AR, AI, E, TAU

—— EJHNAN 参照。

MM —— 求める固有値の数。整数型，入力。$1 \leqq MM \leqq n$ とする。

D, E2, W, IND, RV4, RV5, EPS1

—— EJHNBN 参照。

LA —— 求まった固有値の数。整数型，出力。求める固有値の一方の端（LB または UB）が重複固有値となったとき LA = 0 となるほかは LA=MM となる。

M11 —— 求める固有値の最初の番号。整数型，入力。$1 \leqq M11 \leqq n$ とする。固有値は小さい方から数えて M11番目から（M11 + MM − 1）番目まで求めることになる。

LB —— 求める MM 個の固有値の最初の値。倍精度実数型，出力。LB < UB となる。

UB —— 求めるMM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJHNIC** (NM, MM, N, AR, AI, D, E, E2, W, ZR, ZI, IND, TAU, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1)

NM —— EJHNAC 参照。

MM —— 求める固有値や固有ベクトルの数。整数型，入力。$1 \leqq MM \leqq n$ とする。

N, AR, AI, E, TAU

—— EJHNAN 参照。

D, E2, W, IND, RV4, RV5, EPS1

—— EJHNBN 参照。

ZR, ZI, RV1, RV2, RV3, RV6

—— EJHNBC 参照。

LE ―― 求まった固有ベクトルの数。整数型，出力。LE＝0 または LE＝LA となる。

CALL **EJHNEN** (NM, MM, N, AR, AI, E, W, BD, IND, TAU, LA, TYPE, IDEF, EPS1)

NM, N, AR, AI, E, TAU
―― EJHNAN 参照。

MM ― EJHNIN 参照。

W ―― 固有値を格納する。1次元倍精度実数型配列 W(N)，出力。TYPE＝.TRUE. のとき小さい順に，.FALSE. のとき大きい順に W(1) から LA 個の固有値が入れられる。

BD ―― 固有値の理論的誤差範囲を格納する。1次元倍精度実数型配列 BD(N)，出力。W(j) に対する誤差は第 j 段階で出力された EPS1 に対し $BD(j) \leqq j \times EPS1$ ($j=1 \sim LA$) となっている。

IND ― EJHNBN 参照。

LA ―― 求まった固有値の数。整数型，出力。行列 **A** が正定値でないのに IDEF＝1，TYPE＝.TRUE. としたとき，**A** が負定値でないのに IDEF＝－1，TYPE＝.FALSE. としたとき，それに MM＞n としたとき LA＝0 となるが，そのほかのときは LA＝MM となる。

TYPE ― 固有値の大小順を指定する。論理型，入力。小さい順のとき .TRUE.，大きい順のとき .FALSE. とする。

IDEF ― 行列 **A** の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS1 ― 固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。$EPS1 \geqq 0$ とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP (倍精度計算ではおよそ 10^{-18}) とするとき，W(0)＝0 と仮定して $EPS1 < MACHEP \times \sum_{k=1}^{j} |W(k)-W(k-1)|$ となる度に EPS1 は右辺で置き換えられる。

CALL **EJHNEC** (NM, MM, N, AR, AI, D, E, E2, W, ZR, ZI, BD, IND, TAU, RV1, RV2, RV3, RV4, RV6, LA, LE, TYPE, IDEF, EPS1)

NM ― EJHNAC 参照。

MM, LE
―― EJHNIC 参照。

N, AR, AI, E, TAU
―― EJHNAN 参照。

D, E2, IND, RV4
―― EJHNBN 参照。

W, BD, LA, TYPE, IDEF, EPS1

—— EJHNEN 参照。

ZR, ZI, RV1, RV2, RV3, RV6

—— EJHNBC 参照。

CALL **EJHPAN** (NM, N, A, E, W, TAU, LA, IMP)

NM —— 2次元配列Aの大きさを定める第1の整合寸法。整数型，入力。MM≧n とする。

N —— 行列 **A** の次数n。整数型，入力。N≧3 とする。

A —— 行列 **A** を格納する。2次元倍精度実数型配列 A(NM, N)，入出力。$\mathbf{A}=(a_{ij})=(\alpha_{ij}+i\beta_{ij})$ に対し，A(i, j)$=\alpha_{ij}$ $(1\leqq j\leqq i\leqq n)$，A(i, j)$=\beta_{ij}$ $(1\leqq i<j\leqq n)$ と入れる。

E —— 作業領域。1次元倍精度実数型配列 E(N)，出力。

W —— 固有値を格納する。1次元倍精度実数型配列 W(N)，出力。LA>0 のとき，W(j) (j=1～LA) に入れられる。特に LA=n のときは小さい順に並んでいる。

TAU — 作業領域。2次元倍精度実数型配列 TAU(2, N)，出力。

LA — 求まった固有値の数。整数型，出力。$0\leqq LA\leqq n$ となる。

IMP — 解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJHPAC** (NM, N, A, E, W, ZR, ZI, TAU, L, IMP)

NM — 2次元配列 A，ZR，ZI の大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

N, A, E, TAU, IMP

—— EJHPAN 参照。

W —— 固有値を格納する。1次元倍精度実数型配列 W(N)，出力。L>0 のとき，W(j) (j=1～L) に入れられる。特に L=n のときは小さい順に並んでいる。

ZR —— 固有ベクトルの実数部を格納する。2次元倍精度実数型配列 ZR(NM, N)，出力。W(j)に対応する固有ベクトルを $\mathbf{x}_j=(\xi_{ij}+i\ \eta_{ij})$ とするとき，ZR(i, j)$=\xi_{ij}$ と入れられる。ベクトルは正規直交化されている。

ZI —— 固有ベクトルの虚数部を格納する。2次元倍精度実数型配列 ZI(NM, N)，出力。W(j) に対応する固有ベクトル $\mathbf{x}_j$ に対し，ZI(i, j)$=\eta_{ij}$ と入れられる。

L —— 求まった固有値や固有ベクトルの数。整数型，出力。$0\leqq L\leqq n$ となる。

CALL **EJHPBN** (NM, MM, N, A, D, E, E2, W, IND, TAU, RV4, RV5, LA, L, LB, UB, EPS1)

NM, N, A, E, TAU

—— EJHPAN 参照。

MM ——区間〔LB, UB〕に存在する固有値の数の上限。整数型，入力。$L \leqq MM \leqq n$ とする。

D ——作業領域。1次元倍精度実数型配列 D(N)，出力。

E2 ——作業領域。1次元倍精度実数型配列 E2(N)，出力。

W ——固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。LA＞0のとき W(j)（j＝1～LA）に小さい順に入れられる。

IND ——作業領域。1次元整数型配列 IND(N)，出力。

RV4 ——作業領域。1次元倍精度実数型配列 RV4(N)，出力。

RV5 ——作業領域。1次元倍精度実数型配列 RV5(N)，出力。

LA ——求まった固有値の数。整数型，出力。L＞MM のとき LA＝0 となるほかは LA＝L となる。

L ——区間〔LB, UB〕に実際に存在する固有値の数。整数型，出力。$0 \leqq L \leqq n$ となる。

LB ——固有値を捜す範囲の下界。倍精度実数型，入力。LB＜UB とする。$LB \geqq UB$ としたときは LA＝0 となる。

UB ——固有値を捜す範囲の上昇。倍精度実数型，入力。

EPS1 ——固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，行列 **A** の相対的ずれ（誤差）が MACHEP 程度であるときの固有値のずれ程度で求めようとすれば，EPS1をおよそ MACHEP×$\|\mathbf{A}\|_1$ にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。$EPS1 \leqq 0$ としたときは，ほぼ上の値に置き換えられる。

CALL **EJHPBC** (NM, MM, N, A, D, E, E2, W, ZR, ZI, IND, TAU, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, L, LB, UB, EPS1)

NM ——EJHPAC 参照。

MM, D, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1

—— EJHPBN 参照。

N, A, E, TAU

—— EJHPAN 参照。

ZR ——固有ベクトルの実数部を格納する。2次元倍精度実数型配列 ZR(NM, MM)，出力。W(j) に対応する固有ベクトルを $\mathbf{x}_j=(\xi_{ij}+i\,\eta_{ij})$ とするとき，$ZR(i,j)=\xi_{ij}$ と入れられる。ベクトルは正規直交化されている。

ZI ——固有ベクトルの虚数部を格納する。2次元倍精度実数型配列 ZI(NM, MM)，出力。W(j) に対応する固有ベクトル $\mathbf{x}_j$ に対し，$ZI(i,j)=\eta_{ij}$ と入れられる。

RV1 ——作業領域。1次元倍精度実数型配列 RV1(N)，出力。

RV2 ——作業領域。1次元倍精度実数型配列 RV2(N)，出力。

RV3 —作業領域。1次元倍精度実数型配列 RV3(N)，出力。
RV6 —作業領域。1次元倍精度実数型配列 RV6(N)，出力。
LE —求まった固有ベクトルの数。整数型，出力。LE＝0 または LE＝MM となる。

CALL **EJHPIN** (NM, MM, N, A, D, E, E2, W, IND, TAU, RV4, RV5, LA, M11, LB, UB, EPS1)

NM, N, A, E, TAU
— EJHPAN 参照。
MM —求める固有値の数。整数型，入力。1≦MM≦n とする。
D, E2, W, IND, RV4, RV5, EPS1
— EJHPBN 参照。
LA —求まった固有値の数。整数型，出力。求める固有値の一方の端（LB または UB）が重複固有値となったとき LA＝0 となるほかは LA＝MM となる。
M11 —求める固有値の最初の番号。整数型，入力。1≦M11≦n とする。固有値は小さい方から数えて M11 番目から（M11＋MM−1）番目まで求めることになる。
LB —求める MM個の固有値の最初の値。倍精度実数型，出力。LB＜UB となる。
UB —求める MM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJHPIC** (NM, MM, N, A, D, E, E2, W, ZR, ZI, IND, TAU, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1)

NM —EJHPAC 参照。
MM —求める固有値や固有ベクトルの数。整数型，入力。1≦MM≦n とする。
N, A, E, TAU
— EJHPAN 参照。
D, E2, W, IND, RV4, RV5, EPS1
— EJHPBN 参照。
ZR, ZI, RV1, RV2, RV3, RV6, LE
— EJHPBC 参照。
LA, M11, LB, UB
— EJHPIN 参照。

CALL **EJHPEN** (NM, MM, N, A, E, W, BD, IND, TAU, LA, TYPE, IDEF, EPS1)

NM, N, A, E, TAU
— EJHPAN 参照。

MM ―― EJHPIN 参照。

W ―― 固有値を格納する。1次元倍精度実数型配列W(N)，出力。TYPE＝.TRUE. のとき小さい順に，.FALSE. のとき大きい順にW(1)からLA個の固有値が入れられる。

BD ―― 固有値の理論的誤差範囲を格納する。1次元倍精度実数型配列 BD(N)，出力。W(j)に対する誤差は第j段階で出力されたEPS1に対しBD(j)≦j×EPS1（j＝1~LA）となっている。

IND ―― EJHPBN 参照。

LA ―― 求まった固有値の数。整数型，出力。行列**A**が正定値でないのにIDEF＝1，TYPE＝.TRUE. としたとき，**A**が負定値でないのにIDEF＝－1，TYPE＝.FALSE. としたとき，それにMM＞n としたときLA＝0となるが，それ以外のときはLA＝MM となる。

TYPE－固有値の大小順を指定する。論理型，入力。小さい順のとき .TRUE.，大きい順のとき .FALSE. とする。

IDEF ― 行列**A** の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS1 ― 固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS1≧0。計算機で1＋ε＝1となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，W(0)＝0 と仮定してEPS1＜MACHEP×$\sum_{k=1}^{j}$|W(k)－W(k－1)| となる度にEPS1は右辺で置き換えられる。

CALL **EJHPEC** (NM, MM, N, A, D, E, E2, W, ZR, ZI, BD, IND, TAU, RV1, RV2, RV3, RV4, RV6, LA, LE, TYPE, IDEF, EPS1)

NM ―― EJHPAC 参照。

MM ―― EJHPIC 参照。

N, A, E, TAU
―― EJHPAN 参照。

D, E2, IND, RV4
―― EJHPBN 参照。

W, BD, LA, TYPE, IDEF, EPS1
―― EJHPEN 参照。

ZR, ZI, RV1, RV2, RV3, RV6, LE
EJHPBC 参照。

CALL **EJGRGN** (NM, N, A, B, ALFR, ALFI, BETA, LA)

NM ― 2次元配列A，Bの大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

N ―― 行列**A** および**B** の次数。整数型，入力。N ≧ 3 とする。

A ―― 行列**A**を格納する。2次元倍精度実数型配列A(NM, N)，入出力。$\mathbf{A}=(a_{ij})$とするとき，A(i, j)=a_{ij}と入れる。

B ―― 行列**B**を格納する。2次元倍精度実数型配列B(NM, N)，入出力。$\mathbf{B}=(b_{ij})$とするとき，B(i, j)=b_{ij}と入れる。

ALFR－固有値を分数表示で与えるときの分子の実数部を格納する。1次元倍精度実数型配列ALFR(N)，出力。

ALFI－固有値を分数表示で与えるときの分子の虚数部を格納する。1次元倍精度実数型配列ALFI(N)，出力。

BETA－固有値を分数表示するときの分母を格納する。1次元倍精度実数型 BETA(N)，出力。LA>0 のとき，一般に固有値 λ_j は $\lambda_j=(ALFR(j)+i\times ALFI(j))/BETA(j)$ (j=n－LA＋1～n)で表わされる。特に，BETA(j)=0 のときは，対応する固有ベクトル $\mathbf{x}_j$ に対し，$(ALFR(j)+i\times ALFI(j))\times \mathbf{B}\,\mathbf{x}_j=0$ である。また，格納の順序は特にないが，共役な固有値は続けて入れられ，虚数部が正のものが先になっている。

LA ―― 求まった固有値の数。整数型，出力。$0\leqq LA\leqq n$ となる。

CALL **EJGRGC** (NM, N, A, B, ALFR, ALFI, BETA, Z, LA, LE)

NM ―― 2次元配列A，B，Zの大きさを定める第1の整合寸法。整数型，入力。$NM\geqq n$ とする。

N, A, B, ALFR, ALFI, BETA, LA

―― EJGRGN 参照。

Z ―― 固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, N)，出力。j番目に入れられた固有値 λ_j が実数のとき，その固有ベルトル $\mathbf{x}_j$ はZの第j欄に入れられる。一方，λ_j が複素数のとき，虚数部の正の方の固有値がj番目に入れられ，それに対応する固有ベクトルは実数部と虚数部がそれぞれZの第j欄と(j＋1)欄に入れられる。従って(j＋1)番目に入れられた固有値 $\bar{\lambda}_j$ に対応する固有ベクトルは(Z(i, j)－i×Z(i, j＋1))である。これらは $\max_{1\leqq i\leqq n}|x_{ij}|=1$ となるよう正規化されている。

LE ―― 求まった固有ベクトルの数。整数型，出力。LE＝0 または LE＝n となる。

CALL **EJGSAN** (NM, N, A, B, E, W, LA, IMP)

NM ―― 2次元配列A，Bの大きさを定める第1の寸法。整数型，入力。$NM\geqq n$ とする。

N ―― 行列**A**や**B**の次数n。整数型，入力。$N\geqq 3$ とする。

A ―― 行列**A**を格納する。2次元倍精度実数型配列A(NM, N)，入出力。$\mathbf{A}=(a_{ij})$を$A(i, j)=a_{ij}$ $(1\leqq i\leqq j\leqq n)$と入れる。

B ―― 行列**B**を格納する。2次元倍精度実数型配列B(NM, N)，入出力。$\mathbf{B}=(b_{ij})$を$B(i, j)=b_{ij}$ $(1\leqq i\leqq j\leqq n)$と入れる。

E ——作業領域。1次元倍精度実数型配列E(N)，出力。

W ——固有値を格納する。1次元倍精度実数型配列W(N)，出力。LA＞0 のとき，W(j)（j＝1～LA）と入る。特にLA=n のときは小さい順に並んでいる。

LA ——求まった固有値の数。整数型，出力。$0 \leqq LA \leqq n$ となる。行列 **B** が正定値でないときにもLA＝0 となる。

IMP ——解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJGSAC** (NM, N, A, B, E, E2, W, Z, LA, LE, IMP)

NM ——2次元配列A，B，Z の大きさを定める第1の整合寸法。整数型，入力。$NM \geqq n$ とする。

N, A, B, E, W, IMP

——EJGSAN 参照。

E2 ——作業領域。1次元倍精度実数型配列 E2(N)，出力。

Z ——固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, N)，出力。LE＞0 のとき，W(j) に対応する固有ベクトル $\mathbf{x}_j$ がZの第j欄に入れられる。これらは $\mathbf{x}_j^t \mathbf{B} \mathbf{x}_j = 1$ となるよう正規化されている。

LA ——求まった固有値の数。整数型，出力。$0 \leqq LA \leqq n$ となる。行列 **B** が正定値でなかった場合のほか，IMP＝0 でN＞NM としたときにもLA＝0 となる。

LE ——求まった固有ベクトルの数。整数型，出力。IMP＝0 で 0＜LA＜n のとき LE＝0 となるが，その他のときはLE=LA である。

CALL **EJGSBN** (NM, MM, N, A, B, D, E, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1)

NM, N, A, B, E

——EJGSAN 参照。

MM ——区間〔LB，UB）に存在する固有値の数の上限。整数型，入力。$L \leqq MM \leqq n$ とする。

D ——作業領域。1次元倍精度実数型配列 D(N)，出力。

E2 ——EJGSAC 参照。

W ——固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。LA＞0 のとき W(j)（j＝1～LA）に小さい順に入れられる。

IND ——作業領域。1次元整数型配列 IND(N)，出力。

RV4 ——作業領域。1次元倍精度実数型配列 RV4(N)，出力。

RV5 ——作業領域。1次元倍精度実数型配列 RV5(N)，出力。

LA ——求まった固有値の数。整数型，出力。L＞MM のとき LA＝0 となるほかはLA=Lとなる。

L ——区間〔LB，UB）に存在する固有値の数。整数型，出力。$0 \leqq L \leqq n$ となる。行列 **B** が正定値でないときにもL＝0 となる。

LB ―― 固有値を捜す範囲の下界を与える。倍精度実数型，入力。LB＜UB とする。LB≧UB としたときはLA＝0 となる。

UB ―― 固有値を捜す範囲の上界を与える。倍精度実数型，入力。

EPS1 ― 固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0 とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，行列 **A** の相対的ずれ（誤差）が MACHEP 程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1をおよそ MACHEP × $\|\mathbf{A}\|_1$ にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0 としたときは，ほぼ上の値に置き換えられる。

CALL **EJGSBC** （NM，MM，N，A，B，D，E，E2，W，Z，DL，RV1，RV2，RV3，RV4，RV5，RV6，LA，L，LB，UB，EPS1）

NM ―― EJGSAC 参照。

MM ―― 求める固有値や固有ベクトルの数の上限。整数型，入力。LA≦MM≦n とする。

N，A，B，E

―― EJGSAN 参照。

D，E2，RV4，RV5，LB，UB，EPS1

―― EJGSBN 参照。

W ―― 固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。L＞0 のとき，W(j)（j＝1～L）に入れられる。

Z ―― 固有ベクトルを格納する。2次元倍精度実数型配列 Z(NM，MM)，出力。L＞0 のとき，W(j)（j＝1～L）に対応する固有ベクトル $\mathbf{x}_j$ が Z の第 j 欄に入れられる。これらは $\mathbf{x}_j^t \mathbf{B}\, \mathbf{x}_j = 1$ となるよう正規化されている。

DL ―― 作業領域。1次元倍精度実数型配列 DL(N)，出力。

RV1 ― 作業領域。1次元倍精度実数型配列 RV1(N)，出力。

RV2 ― 作業領域。1次元倍精度実数型配列 RV2(N)，出力。

RV3 ― 作業領域。1次元倍精度実数型配列 RV3(N)，出力。

RV6 ― 作業領域。1次元倍精度実数型配列 RV6(N)，出力。

LA ― 区間〔LB，UB）に存在する固有値の数。整数型，出力。0≦LA≦n となる。行列 **B** が正定値でなかったときにもLA＝0 となる。

L ―― 求まった固有値および固有ベクトルの数。整数型，出力。0≦L≦LA となる。LA＞MM となったときL＜LA となるほかは L＝LA となる。

CALL **EJGSIN** (NM, MM, N, A, B, D, E, E2, W, IND, RV4, RV5, LA, M11, LB, UB, EPS1)

NM, N, A, B, E
—EJGSAN 参照。

MM —求める固有値の数。整数型，入力。$1 \leqq MM \leqq n$ とする。

D, E2, W, IND, RV4, RV5, EPS1
—EJGSBN 参照。

LA —求まった固有値の数。整数型，出力。行列 **B** が正定値でないときと，求める固有値の一方の端（LB または UB）が重複固有値となったとき LA＝0 となるが，そのほかのときは LA＝MM となる。

M11 —求める固有値の最初の番号。整数型，入力。$1 \leqq M11 \leqq n$ とする。固有値は小さい方から数えて M11番目から（M11＋MM－1）番目まで求めることになる。

LB —求める MM個の固有値の最初の値。倍精度実数型，出力。LB＜UB となる。

UB —求める MM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJGSIC** (NM, MM, N, A, B, D, E, E2, W, Z, IND, DL, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1)

NM —EJGSAC 参照。

MM —求める固有値や固有ベクトルの数。整数型，入力。$1 \leqq MM \leqq n$ とする。

N, A, B, E
—EJGSAN 参照。

D, E2, W, IND, RV4, RV5, EPS1
—EJGSBN 参照。

Z —固有ベクトルを格納する。2次元倍精度実数型配列 Z (NM, MM)，出力。LE＞0のとき，W(j)（j＝1～LE）に対応する固有ベクトル $\mathbf{x}_j$ が Z の第 j 欄に入れられる。これらは $\mathbf{x}_j^t \mathbf{B} \mathbf{x}_j = 1$ となるよう正規化されている。

DL, RV1, RV2, RV3, RV6
—EJGSBC 参照。

LA, M11, LB, UB
—EJGSIN 参照。

LE —求まった固有ベクトルの数。整数型，出力。LE＝0 または LE＝LA となる。

CALL **EJGSEN** (NM, MM, N, A, B, E, W, BD, IND, LA, TYPE, IDEF, EPS1)

NM, N, A, B, E
—EJGSAN 参照。

MM ——EJGSIN 参照。

W ——固有値を格納する。1次元倍精度実数型配列 W(N)，出力。TYPE＝.TRUE．のとき小さい順に，.FALSE．のとき大きい順に W(1) から LA 個の固有値が入れられる。

BD ——固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列 BD(N)，出力。W(j) に対する誤差は，第 j 段階で出力された EPS1 に対し BD(j)≦j×EPS1（j＝1～LA）となっている。

IND ——EJGSBN 参照。

LA ——求まった固有値の数。整数型，出力。行列 **B** が正定値でないとき，行列 $\mathbf{B}^{-1}\mathbf{A}$ が正定値でないのに IDEF＝1，TYPE＝.TRUE．としたとき，$\mathbf{B}^{-1}\mathbf{A}$ が負定値でないのに IDEF＝－1，TYPE＝.FALSE．としたとき，それに MM＞n としたとき LA＝0 となるが，それ以外のときは LA＝MM となる。

TYPE－固有値の大小順を指定する。論理型，入力。小さい順のとき．TRUE．，大きい順のとき．FALSE．とする。

IDEF ―行列 $\mathbf{B}^{-1}\mathbf{A}$ の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS1 ―固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS1≧0 とする。計算機で 1＋ε＝1 となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，W(0)＝0 と仮定して EPS1＜MACHEP×$\sum_{k=1}^{j}$ |W(k)－W(k－1)| となる度に EPS1 は右辺で置き換えられる。

CALL **EJGSEC** (NM, MM, N, A, B, D, E, E2, W, Z, BD, IND, DL, RV1, RV2, RV3, RV4, RV6, LA, LE, TYPE, IDEF, EPS1)

NM ——EJGSAC 参照。

MM, Z, LE
——EJGSIC 参照。

N, A, B, E
——EJGSAN 参照。

D, E2, IND, RV4
——EJGSBN 参照。

BD, LA, TYPE, IDEF, EPS1
——EJGSEN 参照。

DL, RV1, RV2, RV3, RV6
——EJGSBC 参照。

W ——固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。TYPE＝.TRUE．のとき小さい順に，.FALSE．のとき大きい順に W(1) から LA 個の固有値が入れられる。

CALL **EJNABA** (NM, N, A, B, E, W, LA, IMP)

NM ―― 2次元配列A，Bの大きさを定める第1の寸法。整数型，入力。NM≧n とする。

N ――行列 **A** や **B** の次数n。整数型，入力。N≧3 とする。

A ――行列 **A** を格納する。2次元倍精度実数型配列 A(NM, N)，入出力。$\mathbf{A}=(a_{ij})$ を $A(i,j)=a_{ij}$ $(1\leqq i\leqq j\leqq n)$ と入れる。

B ――行列 **B** を格納する。2次元倍精度実数型配列 B(NM, N)，入出力。$\mathbf{B}=(b_{ij})$ を $B(i,j)=b_{ij}$ $(1\leqq i\leqq j\leqq n)$ と入れる。

E ――作業領域。1次元倍精度実数型配列 E(N)，出力。

W ――固有値を格納する。1次元倍精度実数型配列 W(N)，出力。LA>0 のとき，W(j) (j＝1～LA) と入れる。特にLA＝n のときは小さい順に並んでいる。

LA ――求まった固有値の数。整数型，出力。0≦LA≦n となる。行列 **B** が正定値でないときにもLA＝0 となる。

IMP ― 解法を選択する。整数型，入力。QL法のとき0，陰的QL法のとき1とする。

CALL **EJNABB** (NM, MM, N, A, B, D, E, E2, W, IND, RV4, RV5, LA, L, LB, UB, EPS1)

NM, N, A, B, E

――EJGSAN 参照。

MM ―― 区間〔LB, UB〕に存在する固有値の数の上限。整数型，入力。L≦MM≦n とする。

D ――作業領域。1次元倍精度実数型配列 D(N)，出力。

E2 ――作業領域。1次元倍精度実数型配列 E2(N)，出力。

W ――固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。LA>0 のとき W(j) (j＝1～LA) に小さい順に入れられる。

IND ― 作業領域。1次元整数型配列 IND(N)，出力。

RV4 ― 作業領域。1次元倍精度実数型配列 RV4(N)，出力。

RV5 ― 作業領域。1次元倍精度実数型配列 RV5(N)，出力。

LA ― 求まった固有値の数。整数型，出力。L＞MM のとき LA＝0 となるほかはLA＝L となる。

L ――区間〔LB, UB〕に存在する固有値の数。整数型，出力。0≦L≦n となる。行列 **B** が正定値でないときにも L＝0 となる。

LB ――固有値を捜す範囲の下界を与える。倍精度実数型，入力。LB＜UB とする。LB≧UB としたときはLA＝0 となる。

UB ――固有値を捜す範囲の上界を与える。倍精度実数型，入力。

EPS1 ―固有値の絶対誤差の許容範囲を与える。倍精度実数型，入出力。EPS1＞0 とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP (倍精度計算ではおよそ 10^{-18}) とす

るとき，行列**A**の相対的ずれ（誤差）がMACHEP程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1をおよそMACHEP×$\|\mathbf{A}\|_1$にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1≦0としたときは，ほぼ上の値に置き換えられる。

CALL **EJNABI** (NM, MM, N, A, B, D, E, E2, W, IND, RV4, RV5, LA, M11, LB, UB, EPS1)

NM, N, A, B, E
—— EJNABA 参照。

MM —— 求める固有値の数。整数型，入力。1≦MM≦n とする。

D, E2, W, IND, RV4, RV5, EPS1
—— EJNABB 参照。

LA —— 求まった固有値の数。整数型，出力。行列**B**が正定値でないときと，求める固有値の一方の端（LB または UB ）が重複固有値となったときLA＝0となるが，そのほかのときはLA＝MMとなる。

M11 —— 求める固有値の最初の番号。整数型，入力。1≦M11≦n とする。固有値は小さい方から数えてM11番目から（M11＋MM－1）番目まで求めることになる。

LB —— 求めるMM個の固有値の最初の値。倍精度実数型，出力。LB＜UB となる。

UB —— 求めるMM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJNABE** (NM, MM, N, A, B, E, W, BD, IND, LA, TYPE, IDEF, EPS1)

NM, N, A, B, E
—— EJNABA 参照。

MM —— EJNABI 参照。

W —— 固有値を格納する。1次元倍精度実数型配列 W(N)，出力。TYPE＝.TRUE. のとき小さい順に，.FALSE. のとき大きい順にW(1)からLA個の固有値が入れられる。

BD —— 固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列 BD(N)，出力。W(j)に対する誤差は，第j段階で出力されたEPS1に対しBD(j)≦j×EPS1（j＝1～LA）となっている。

IND —— EJNABB 参照。

LA —— 求まった固有値の数。整数型，出力。行列**B**が正定値でないとき，行列**AB**が正定値でないのにIDEF＝1，TYPE＝.TRUE. としたとき，**AB**が負定値でないのにIDEF＝－1，TYPE＝.FALSE. としたとき，それにMM＞n としたときLA＝0となるが，それ以外はLA=MM となる。

TYPE －固有値の大小順を指定する。論理型，入力。小さい順のとき .TRUE.，大きい順のとき .FALSE. とする。

IDEF — 行列 **AB** の定値性を示す。整数型，入力。正定値のとき1，負定値のとき−1，定値でないときや不明のとき0とする。

EPS1 — 固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS1≧0 とする。計算機で1＋ε＝1となる最大の数εをMACHEP（倍精度計算ではおよそ10^{-18}）とするとき，W(0)＝0と仮定して EPS1＜MACHEP×$\sum_{k=1}^{j}$ |W(k)−W(k−1)|となる度にEPS1は右辺で置き換えられる。

CALL **EJVABA** （NM，N，A，B，E，E2，W，Z，LA，LE，IAB，IMP）

NM — 2次元配列A，B，Z の大きさを定める第1の整合寸法。整数型，入力。NM≧nとする。

N — 行列 **A** や **B** の次数 n。整数型，入力。N≧3 とする。

A — 行列 **A** を格納する。2次元倍精度実数型配列 A（NM，N），入出力。**A**＝(a_{ij}) を A(i，j)＝a_{ij} （1≦i≦j≦n）と入れる。

B — 行列 **B** を格納する。2次元倍精度型配列 B（NM，N），入出力。**B**＝(b_{ij}) をB(i，j)＝b_{ij}（1≦i≦j≦n）と入れる。

E — 作業領域。1次元倍精度実数型配列 E(N)，出力。

E2 — 作業領域。1次元倍精度実数型配列 E2(N)，出力。

W — 固有値を格納する。1次元倍精度実数型配列 W(N)，出力。LA＞0のとき，W(j)（j＝1～LA）に入れられる。特にLA＝n のときは小さい順に並んでいる。

Z — 固有ベクトルを格納する。2次元倍精度実数型配列Z（NM，N），出力。LE＞0 のとき，W(j) に対応する固有ベクトル $\mathbf{x}_j$ がZの第j欄に入れられる。これらは $\mathbf{x}_j^t \mathbf{B} \mathbf{x}_j$ ＝1となるよう正規化されている。

LA — 求まった固有値の数。整数型，出力。0≦LA≦n となる。行列 **B** が正定値でなかった場合のほか，IMP＝0 で N＞NM としたときにも LA＝0 となる。

LE — 求まった固有ベクトルの数。整数型，出力。IMP＝0 で 0＜LA＜n のとき LE＝0 となるが，その他のときは LE＝LA である。

IAB — 正定値行列を指示する。整数型，入力。行列 **A** が正定値のとき1，**B** が正定値のとき0とする。

IMP — 解法を選択する。QL法のとき0，陰的QL法のとき1とする。

CALL **EJVABB** （NM，MM，N，A，B，D，E，E2，W，Z，DL，RV1，RV2，RV3，RV4，RV5，RV6，LA，L，LB，UB，EPS1，IAB）

NM，N，A，B，E，E2，IAB
— EJVABA 参照。

MM — 求める固有値や固有ベクトルの数の上限。整数型，入力。LA≦MM≦n とする。

D —— 作業領域。1次元倍精度実数型配列 D(N)，出力。

W —— 固有値を格納する。1次元倍精度実数型配列 W(MM)，出力。L＞0のとき，W(j)（j＝1～L）に入れられる。

Z —— 固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, MM)，出力。L＞0 のとき，W(j)（j＝1～L）に対応する固有ベクトル $\mathbf{x}_j$ がZの第j欄に入れられる。これらは $\mathbf{x}_j^t \mathbf{B} \mathbf{x}_j = 1$ となるよう正規化されている。

DL —— 作業領域。1次元倍精度実数型配列 DL(N)，出力。

RV1 —— 作業領域。1次元倍精度実数型配列 RV1(N)，出力。

RV2 —— 作業領域。1次元倍精度実数型配列 RV2(N)，出力。

RV3 —— 作業領域。1次元倍精度実数型配列 RV3(N)，出力。

RV4 —— 作業領域。1次元倍精度実数型配列 RV4(N)，出力。

RV5 —— 作業領域。1次元倍精度実数型配列 RV5(N)，出力。

RV6 —— 作業領域。1次元倍精度実数型配列 RV6(N)，出力。

LA —— 区間〔LB，UB〕に存在する固有値の数。整数型，出力。$0 \leqq LA \leqq n$ となる。行列**B**が正定値でなかったときにも LA＝0 となる。

L —— 求まった固有値および固有ベクトルの数。整数型，出力。LA＞MMとなったときL＞LA となるほかは L＝LA となる。

LB —— 固有値を捜す範囲の下界。倍精度実数型，入力。LB＜UB とする。LB≧UB としたときはLA＝0 となる。

UB —— 固有値を捜す範囲の上界。倍精度実数型，入力。

EPS1 —— 固有値の絶対誤差の許容範囲。倍精度実数型，入出力。EPS1＞0とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，行列**A** の相対的ずれ（誤差）がMACHEP 程度であるときの固有値のずれ程度の精度で求めようとすれば，EPS1 をおよそ MACHEP × $\|\mathbf{A}\|_1$ にとる。しかしEPS1を小さくとり過ぎると余分な分割を行うので注意を要する。EPS1としたときは，ほぼ上の値に置き換えられる。

CALL **EJVABI** (NM, MM, N, A, B, D, E, E2, W, Z, IND, DL, RV1, RV2, RV3, RV4, RV5, RV6, LA, LE, M11, LB, UB, EPS1, IAB)

NM, N, A, B, E, E2, W, IAB

—— EJVABA 参照。

MM —— 求める固有値や固有ベクトルの数。整数型，入力。$1 \leqq MM \leqq n$ とする。

D, IND, DL, RV1, RV2, RV3, RV4, RV5, RV6, EPS1

—— EJVABB 参照。

Z —— 固有ベクトルを格納する。2次元倍精度実数型配列Z(NM, MM)，出力。LE＞0 のとき，W(j)（j＝1～LE）に対応する固有ベクトル $\mathbf{x}_j$ がZの第j欄に入れられる。これらは $\mathbf{x}_j^t \mathbf{B} \mathbf{x}_j = 1$ となるよう正規化されている。

LA —求まった固有値の数。整数型，出力。行列 **B** が正定値でないときと，求める固有値の一方の端（LB または UB ）が重複固有値となったとき LA＝0 となるが，その他のときは LA＝MM となる。

LE —求まった固有ベクトルの数。整数型，出力。LE ＝0 または LE＝MM となる。

M11 —求める固有値の最初の番号。整数型，入力。1≦M11≦n とする。固有値は小さい方から数えて M11番目から（M11＋MM－1）番目まで求めることになる。

LB —求める MM個の固有値の最初の値。倍精度実数型，出力。LB＜UB となる。

UB —求める MM個の固有値の最後の値。倍精度実数型，出力。

CALL **EJVABE** (NM, MM, N, A, B, D, E, E2, W, Z, BD, IND, DL, RV1, RV2, RV3, RV4, RV6, LA, LE, TYPE, IDEF, EPS1, IAB)

NM, N, A, B, E, E2, IAB
—EJVABA 参照。

MM, Z, LE
—EJVABI 参照。

D, IND, DL, RV1, RV2, RV3, RV4, RV6
—EJVABB 参照。

W —固有値を格納する。1次元倍精度実数型配列 W（MM），出力。TYPE＝.TRUE. のとき小さい順に，.FALSE. のとき大きい順に W(1)から LA 個の固有値が入れられる。

BD —固有値の理論的な誤差範囲を格納する。1次元倍精度実数型配列 BD(N)，出力。W(j) に対する誤差は，第 j 段階で出力された EPS1 に対し BD(j)≦j×EPS1（j=1～LA）となっている。

LA —求まった固有値の数。整数型，出力。行列 **B** が正定値でないとき，行列 **AB** が正定値でないのに IDEF＝1，TYPE＝.TRUE. としたとき，**AB** が負定値でないのに IDEF＝－1，TYPE＝.FALSE. としたとき，それに MM＞n としたとき LA＝0 となるが，その他のときは LA＝MM となる。

TYPE－固有値の大小順を指定する。論理型，入力。小さい順のとき .TRUE.，大きい順のとき .FALSE. とする。

IDEF —行列 **AB** の定値性を示す。整数型，入力。正定値のとき1，負定値のとき－1，定値でないときや不明のとき0とする。

EPS1 —固有値の理論的誤差の許容範囲を与える。倍精度実数型，入出力。EPS1≧0 とする。計算機で $1+\varepsilon=1$ となる最大の数 ε を MACHEP（倍精度計算ではおよそ 10^{-18}）とするとき，W(0)＝0 と仮定して EPS1＜MACHEP×$\sum_{k=1}^{j}$ |W(k)－W(k－1)| となる度に EPS1 は右辺で置き換えられる。

CALL **EJSVDP** (NM, M, N, A, W, U, V, RV1, L)

NM ——2次元配列A，U，Vの大きさを示す第1の整合寸法。整数型，入力。NM≧mとする。

M ——行列**A**や**U**の行の数m。整数型，入力。M≧n とする。

N ——行列**A**や**U**の列の数，および**V** の次数を示すn。整数型，入力。N≧3 とする。

A ——行列**A** を格納する。2次元倍精度実数型配列A(NM, N)，入力。$\mathbf{A}=(a_{ij})$ (i =1～m, j=1～n)をA(i, j) = a_{ij} と入れる。

W ——行列**A**の特異値を格納する。1次元倍精度実数型配列W(N)，出力。L＞0 のとき，W(j) (j =n−L+1～n) にL個の固有値が入れられる。

U ——行列**U**を格納する。2次元倍精度実数型配列**U**(NM, N)，出力。L＞0のとき，W(j) (j＝n－L＋1～n)に対詰する**U**のベクトルが**U**の第 j 欄に入えられ。

V ——行列**V**を格納する。2次元倍精度実数型配列V(NM, N)出力。L＞0 のとき，W(j) (j＝n－L＋1～n)に対応する**V**のベクトルがVの第j欄に入れられる。

RV1 — 作業領域。1次元倍精度実数型配列 RV1(N)，出力。

L ——求まった特異値や対応するベクトルの数。整数型，出力。0≦L≦n となる。

CALL **EJSVDL** (NM, M, N, IP, A, C, W, RV1, L)

NM —— 2次元配列AやCの大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

M ——行列**A**や**C**の行の数 m。整数型，入力。M≧ n とする。

N, W, RV1, L

—— EJSVDP 参照。

IP ——行列**C**の列の数 p 。整数型，入力。IP≧1 とする。

A ——行列**A** を格納する。2次元倍精度実数型配列 A(NM, N)，入出力。$\mathbf{A}=(a_{ij})$ (i =1～m, j =1～n)をA(i, j)= a_{ij} と入れる。L＞0 のとき，W(j) (j＝n－L＋1～n) に対応する**V**のベクトルはAの第 j 欄に入れられる。

C ——行列**C**を格納する。2次元倍精度実数型配列C(NM, IP)，入出力。$\mathbf{C}=(c_{ij})$ (i = 1～m, j =1～p)をC(i, j)=c_{ij} と入れる。L＞0のとき，W(j) (j＝n－L＋1～n)に対応する $\mathbf{U}^t\mathbf{C}$ のベクトルはCの第 j 行に入れられる。

CALL **ETSVDG** (NM, M, N, A, W, U, V, RV1, TV, IERR)

NM, M, N, RV1

— EJSVDP 参照。

A ——行列**A** を格納する。2次元倍精度実数型配列 A(NM, N)，入出力。行列 $\mathbf{A}=(a_{ij})$ (i =1～m, j =1～n)をA(i, j)= a_{ij} と入れると，一般化逆行列 $\mathbf{A}^{-1}=(g_{ij})$ がA(j, i)= g_{ij} (i =1～n, j =1～m)と出力される。

W —— 行列 **A** の特異値を格納する。1次元倍精度実数型配列 W(N)，出力。

U —— 行列 **U** を格納する。2次元倍精度実数型配列 U（NM，N），出力。$\mathbf{U}=(u_{ij})$ （i =1～m，j =1～n ）とするとき，U（i，j ）$=u_{ij}$ と入れられる。

V —— 行列 **V** を格納する。2次元倍精度実数型配列 V（NM，N），出力。$\mathbf{V}=(v_{ij})$ （i，j =1～n ）とするとき，V（i，j ）$=v_{ij}$ と入れられる。

TV —— 特異値の零判定値。倍精度実数型，入力。TV＞0 とする。TV より小さい特異値は零とみなされる。

IERR —— 終了状態を示す。整数型，出力。正常のとき0，そうでないとき非零となる。

CALL **EJSVDM** （NM，M，N，IP，A，C，W，Z，RV1，TV，IERR）

NM —— 2次元配列 A，C，Z の大きさを定める第1の整合寸法。整数型，入力。NM≧m とする。

M，IP —— EJSVDL 参照。

N —— 行列 **A** の列の数 n 。整数型，入力。N≧3 とする。

A —— 行列 **A** を格納する。2次元倍精度実数型配列 A（NM，N），入出力。$\mathbf{A}=(a_{ij})$ （i＝1～m，j＝1～n ）を A（i，j ）$=a_{ij}$ と入れる。

C —— 行列 **C** を格納する。2次元倍精度実数型配列 C（NM，IP），入出力。$\mathbf{C}=(c_{ij})$ （i＝1～m，j＝1～p ）を C（i，j ）$=c_{ij}$ と入れる。

W，TV，IERR
—— EJSVDG 参照。

Z —— 最小ノルム解 **X** を格納する。2次元倍精度実数型配列 Z（NM，IP），出力。$\mathbf{X}=(x_{ij})$（i =1～n，j＝1～p ）とするとき，Z（i，j ）$=x_{ij}$ と入れられる。

RV1 —— EJSVDP 参照。

CALL **EJLERS** （NM，N，MBW，IP，A，C，RV1，RV6，RV，IERR）

NM —— 2次元配列 A や C の大きさを定める第1の整合寸法。整数型，入力。NM≧n とする。

N —— 行列 **A** の次数 n 。整数型，入力。N≧3 とする。

MBW —— 行列 **A** の帯幅。整数型，入力。MBW≧3 とする。

IP —— 行列 **C** の列の数。整数型，入力。IP≧1 とする。

A —— 行列 **A** を格納する。2次元倍精度実数型配列 A（NM，MBW），入力。$\mathbf{A}=(a_{ij})$ を A（i，j＋MB－i ）$=a_{ij}$ （$1\leqq i，j\leqq n$，$|i-j|\leqq MB-1$）と入れる。ここに MB ＝（MBW＋1）／2 である。

C —— 行列 **C** を格納する。2次元倍精度実数型配列 C（NM，IP），入出力。$\mathbf{C}=(c_{ij})$ （i =1～n，j＝1～p ）を C（i，j ）$=c_{ij}$ と入れると，解行列 $\mathbf{X}=(x_{ij})$ が C（i，j ）$=x_{ij}$ と入れられる。

RV1 ―作業領域。1次元倍精度実数型配列 RV1(N)，出力。
RV6 ―作業領域。1次元倍精度実数型配列 RV6(N)，出力。
RV ―作業領域。1次元倍精度実数型配列 RV(NV)，出力。NV＝n×MBW とする。
IERR―計算の終了状態を示す。整数型，出力。正常のとき0，そうでないとき非零となる。

CALL **EJLESB** (NM, N, MB, IP, A, C, RV1, RV6, RV, IERR)

NM, N, IP, C, RV1, RV6, RV, IERR
EJLERB 参照。
MB ― 行列 **A** の帯の半幅。整数型，入力。MB≧2 とする。2×MB－1が帯幅である。
A ― 行列 **A** を格納する。2次元倍精度実数型配列 A(NM, N)，入力。$\mathbf{A}=(a_{ij})$ を $A(i, j+MB-i)=a_{ij}$ $(1\leqq i\leqq j\leqq n, i-j\leqq MB-1)$ と入れる。

CALL **EJLERH** (NM, M, N, A, W, RV1, TV, LA, LE)

NM ― 2次元配列Aの大きさを定める第1の整合寸法。整数型，入力。NM≧m とする。
M ―行列 **A** の行の数 m。整数型，入力。M≧n とする。
N ―行列 **A** の列の数 n。整数型，入力。N≧3 とする。
A ― 行列 **A** を格納する。2次元倍精度実数型配列 A(NM, N)，入出力。$\mathbf{A}=(a_{ij})$ $(i=1\sim m, j=1\sim n)$ を $A(i, j)=a_{ij}$ と入れる。LE＞0 のとき W(j)＜TV $(j=n-LA+1\sim n)$ に対応する独立解がAの第1欄から入れられる。
W ― 行列 **A** の特異値を格納する。1次元倍精度実数配列 W(N)，出力。LA＞0 のとき，W(j) $(j=n-LA+1\sim n)$ にLA個の特異値が入れられる。
RV1 ―EJLERB 参照。
TV ―特異値の零判定値。倍精度実数型，入力。TV＞0 とする。TVより小さい特異値は零とみなされる。
LA ―求まった特異値の数。整数型，出力。0≦L≦n となる。
LE ―求まった独立解の数。整数型，出力。0≦LE≦LA となる。

付録2　EISPACK-J の見本の例題の実行

例えば，実対称行列の一般問題を解くルーチン EJGSBN を CPS（Conversational Programming System）を用いて，ファイル入力形式で実行するとき，

```
.HFORT      SF=EJDRIVER, ELM=GS
.HLIED      RF=EISPACKRB, EDIT=ON, A=NOMAP
.DISKTO     F01. 001. (RSDATAI/ELM02)
.DISKTO     F02. 001. (RSGDATAI/ELM12)
.DISKTO     F05. 001. (EJDRIVER/GSDATA)
.DISKTN     F06. 001, OUTPUT
.HRUN       A=(SIZE=10)
```

のようにする。これにより，Table 4 のルーチンのうち，EJGSAN から EJGSEC までの8個のルーチンに対して，上の例題を実行した結果が OUTPUT に得られる。

他のルーチンに対する見本の例題を実行する場合，下の表から該当するプログラムや入力データを探し，上の例の下線の部分を書き替える。この表の最初の欄は各ルーチンが属する組を意味し，Table 4 から容易に推察できるように，RN には EJRNEN から EJRNOS までが含まれ，以下同様に，VAB には EJVABA から EJVABE が含まれている。但し，SVD1 には EJSVDP と EJSVDL，SVD2 には FJSVDG と EJSVDM，LE1 には EJLERB と EJLESB，そして LE2 には EJLERH が含まれる。

Driver	Data on unit 1	Data on unit 2	Data on unit 5
RN	RGDATAI/ELM1666		RNDATA
RB	RGDATAI/ELM1666		RBDATA
RTN	RTDATAI/ELM32		RTNDATA
RTP	RTDATAI/ELM32		RTPDATA
SN	RSDATAI/ELM02		SNDATA
SP	RSDATAI/ELM02		SPDATA
SB	RSDATAI/ELM32		SBDATA
ST	RSTDATAI/ELM27		STDATA
CN	CGDATAI/ELM01		CNDATA
CB	CGDATAI/ELM01		CBDATA
HN	CHDATAI/ELM08		HNDATA
HP	CHDATAI/ELM08		HPDATA
GRG	RGDATAI/DATA1	RGGDATA/DATA2	
GS	RSDATAI/ELM02	RSGDATAI/ELM12	GSDATA
NAB	RSDATAI/ELM02	RSGDATAI/ELM12	NABDATA
VAB	RSDATAI/ELM02	RSGDATAI/ELM12	VABDATA
SVD1	RLDATAI/ELM28		
SVD2	RLDATAI/ELM28		
LE1	RBLDATAI/ELM13		LE1DATA
LE2	RLDATAI/ELM28		

付録3　問題 SR-2 の行列

ROW	COLUMN							
	J= 1	J= 2	J= 3	J= 4	J= 5	J= 6	J= 7	J= 8
I= 1	-1.81D+06	1.27D-02	3.17D-02	1.15D-01	3.11D-01	1.40D+00	3.87D+00	0.0
I= 2	5.62D+04	-1.27D-02	0.0	0.0	0.0	0.0	0.0	0.0
I= 3	3.71D+05	0.0	-3.17D-02	0.0	0.0	0.0	0.0	0.0
I= 4	3.37D+05	0.0	0.0	-1.15D-01	0.0	0.0	0.0	0.0
I= 5	7.05D+05	0.0	0.0	0.0	-3.11D-01	0.0	0.0	0.0
I= 6	2.70D+05	0.0	0.0	0.0	0.0	-1.40D+00	0.0	0.0
I= 7	7.12D+04	0.0	0.0	0.0	0.0	0.0	-3.87D+00	0.0
I= 8	1.81D+06	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-1.00D+00

ROW	COLUMN							
	J= 9	J= 10	J= 11	J= 12	J= 13	J= 14	J= 15	J= 16
I= 1	1.00D+00	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 9	1.81D+06	-5.62D+04	-3.71D+05	-3.37D+05	-7.05D+05	-2.70D+05	-7.12D+04	-1.81D+06
I= 10	-1.27D-02	1.27D-02	0.0	0.0	0.0	0.0	0.0	0.0
I= 11	-3.17D-02	0.0	3.17D-02	0.0	0.0	0.0	0.0	0.0
I= 12	-1.15D-01	0.0	0.0	1.15D-01	0.0	0.0	0.0	0.0
I= 13	-3.11D-01	0.0	0.0	0.0	3.11D-01	0.0	0.0	0.0
I= 14	-1.40D+00	0.0	0.0	0.0	0.0	1.40D+00	0.0	0.0
I= 15	-3.87D+00	0.0	0.0	0.0	0.0	0.0	3.87D+00	0.0
I= 16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

付録4　問題 SR-3 の行列

ROW	COLUMN										
	J= 1	J= 2	J= 3	J= 4	J= 5	J= 6	J= 7	J= 8	J= 9	J= 10	J= 11
I= 1	-1.44D-08	2.70D-12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	9.00D-13
I= 2	3.14D-09	-4.32D-09	0.0	0.0	3.76D-06	0.0	0.0	0.0	0.0	0.0	0.0
I= 3	0.0	3.67D-09	-1.20D-06	8.48D-12	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 4	0.0	0.0	2.27D-09	-1.89D-09	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 5	0.0	0.0	0.0	0.0	-8.77D-06	5.40D-13	0.0	0.0	0.0	0.0	0.0
I= 6	0.0	0.0	1.19D-06	0.0	1.99D-09	-6.23D-09	0.0	0.0	0.0	0.0	0.0
I= 7	0.0	0.0	0.0	0.0	0.0	4.24D-09	-3.84D-06	0.0	0.0	0.0	0.0
I= 8	0.0	0.0	0.0	1.64D-09	0.0	0.0	1.99D-09	-3.42D-06	0.0	0.0	0.0
I= 9	0.0	0.0	0.0	0.0	4.99D-06	0.0	0.0	0.0	-1.67D-08	0.0	0.0
I= 10	0.0	0.0	0.0	0.0	0.0	0.0	3.82D-06	0.0	0.0	-9.13D-09	1.11D-12
I= 11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.41D-06	0.0	1.24D-09	-1.30D-08
I= 12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.79D-09
I= 13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 21	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 22	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

ROW	COLUMN										
	J= 12	J= 13	J= 14	J= 15	J= 16	J= 17	J= 18	J= 19	J= 20	J= 21	J= 22
I= 1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 2	3.25D-12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 4	0.0	0.0	5.80D-14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 6	0.0	0.0	0.0	5.07D-11	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 8	0.0	0.0	0.0	0.0	0.0	0.0	2.87D-12	0.0	0.0	0.0	0.0
I= 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.92D-08	0.0	0.0	0.0
I= 11	2.83D-12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	6.86D-10	0.0	0.0
I= 12	-4.27D-09	1.16D-11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.21D-09	0.0
I= 13	2.30D-09	-1.77D-08	3.10D-12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.66D-12
I= 14	0.0	2.39D-09	-3.44D-09	0.0	0.0	2.16D-06	0.0	0.0	0.0	0.0	0.0
I= 15	0.0	1.50D-09	0.0	-8.10D-09	7.70D-12	7.70D-12	0.0	0.0	0.0	0.0	0.0
I= 16	0.0	0.0	0.0	8.89D-10	-1.25D-08	0.0	0.0	0.0	0.0	0.0	0.0
I= 17	0.0	0.0	0.0	4.60D-09	1.45D-10	-1.20D-05	0.0	0.0	0.0	0.0	0.0
I= 18	0.0	0.0	1.89D-09	0.0	2.23D-09	2.23D-09	-4.59D-09	0.0	0.0	0.0	0.0
I= 19	0.0	0.0	0.0	0.0	0.0	9.84D-06	0.0	-5.36D-08	2.08D-12	0.0	0.0
I= 20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.11D-09	-4.68D-09	0.0	0.0
I= 21	0.0	0.0	0.0	0.0	0.0	0.0	3.07D-09	0.0	2.22D-09	-5.56D-09	0.0
I= 22	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.07D-09	-1.58D-08

付録5　問題 SRT の行列

ROW	COLUMN		
	J=I-1	J=I	J=I+1
I= 1	0.0	2.63D+04	-1.97D+04
I= 2	-3.75D+03	-9.07D+03	-2.91D+04
I= 3	-2.02D+04	2.66D+04	6.69D+03
I= 4	3.39D+03	-2.00D+04	2.46D+04
I= 5	1.71D+04	1.31D+04	1.79D+04
I= 6	1.44D+04	2.62D+04	2.04D+04
I= 7	7.25D+02	2.81D+04	-2.74D+04
I= 8	-1.95D+04	2.64D+04	9.02D+03
I= 9	8.61D+03	-2.09D+04	5.20D+03
I= 10	1.86D+04	2.12D+04	-2.86D+04
I= 11	-2.76D+04	-1.32D+03	6.43D+02
I= 12	2.22D+04	3.06D+04	-1.71D+04
I= 13	-7.49D+02	1.87D+04	1.98D+04
I= 14	1.27D+04	2.12D+04	-1.64D+04
I= 15	-2.29D+04	-3.13D+02	0.0

付録6　問題 SC の行列

REAL　　PART

ROW \ COLUMN	J= 1	J= 2	J= 3	J= 4	J= 5	J= 6	J= 7	J= 8	J= 9	J=10
I= 1	2	3	0	0	0	0	0	0	0	0
I= 2	3	-2	1	0	0	0	0	0	0	0
I= 3	5	1	2	-1	0	0	0	0	0	0
I= 4	2	-2	3	-4	5	0	0	0	0	0
I= 5	1	2	-3	1	2	1	0	0	0	0
I= 6	5	0	1	-8	4	7	4	0	0	0
I= 7	5	1	6	8	4	-1	3	-4	0	0
I= 8	-4	1	1	2	3	1	1	6	7	0
I= 9	5	2	1	1	-4	1	1	2	0	3
I=10	5	2	1	7	4	-7	3	5	6	2

IMAGINARY　PART

ROW \ COLUMN	J= 1	J= 2	J= 3	J= 4	J= 5	J= 6	J= 7	J= 8	J= 9	J=10
I= 1	3	1	0	0	0	0	0	0	0	0
I= 2	2	-1	2	0	0	0	0	0	0	0
I= 3	-3	2	1	4	0	0	0	0	0	0
I= 4	6	3	-1	2	5	0	0	0	0	0
I= 5	4	2	7	5	-3	6	0	0	0	0
I= 6	-1	4	5	-1	7	1	-2	0	0	0
I= 7	2	4	-5	4	-4	5	0	6	0	0
I= 8	-3	3	6	-4	1	2	4	3	-1	0
I= 9	0	2	3	1	-2	6	2	5	1	2
I=10	2	6	-3	4	1	0	-3	-4	3	5

付録7　問題 SH-1 の行列

REAL PART

ROW	COLUMN				
	J=1	J=2	J=3	J=4	J=5
I=1	-0.8450	5.2000	0.3010	-9.6000	0.0734
I=2	5.2000	-6.2000	-3.3900	0.1220	4.1900
I=3	0.3010	-3.3900	0.0190	0.9350	-0.0572
I=4	-9.6000	0.1220	0.9350	7.2100	0.3370
I=5	0.0734	4.1900	-0.0572	0.3370	-1.2300

IMAGINARY PART

ROW	COLUMN				
	J=1	J=2	J=3	J=4	J=5
I=1	0.0	0.1030	-0.0454	0.9360	7.2600
I=2	-0.1030	0.0	-0.4070	0.9100	-3.6600
I=3	0.0454	0.4070	0.0	-0.2710	2.8200
I=4	-0.9360	-0.9100	0.2710	0.0	0.0603
I=5	-7.2600	3.6600	-2.8200	-0.0603	0.0

付録8　問題 GH の行列

MATRIX　A　　(REAL　　PART)

ROW	COLUMN										
	J=I	J=I+1	J=I+2	J=I+3	J=I+4	J=I+5	J=I+6	J=I+7	J=I+8	J=I+9	J=I+10
I= 1	1.8D+05	2.7D-12	1.7D+02	-6.0D-14	-1.6D+00	-4.5D+04	-6.0D-13	2.2D-16	1.8D-16	1.8D-17	0.0
I= 2	3.1D+05	6.1D-15	3.1D-15	2.8D-17	-2.2D+04	-5.6D+04	-2.8D-16	-3.5D-16	2.8D-17	0.0	0.0
I= 3	6.3D+03	-8.0D+01	-9.6D+00	-9.9D-15	1.1D+03	4.2D+02	-4.4D+01	-7.4D-01	0.0	0.0	0.0
I= 4	6.0D+03	-4.5D+01	-3.9D+01	-3.3D+01	3.6D+02	1.1D+03	-4.2D+01	0.0	0.0	0.0	0.0
I= 5	1.1D+00	2.0D-16	3.0D-15	-1.5D+01	-1.2D+01	7.1D-01	0.0	0.0	0.0	0.0	0.0
I= 6	1.1D+05	3.9D-12	-2.2D-16	-1.9D+01	-6.7D+03	-3.3D+04	-1.8D-12	2.6D-17	1.4D-17	3.8D-17	0.0
I= 7	9.1D+04	1.5D-13	-6.5D+01	6.6D-12	-1.3D+04	-1.9D+04	-9.1D-14	-3.3D+01	3.5D-15	0.0	0.0
I= 8	1.2D+03	-2.3D+01	-2.1D+01	8.5D-16	2.2D+02	2.5D+02	-1.2D+01	-1.0D+01	0.0	0.0	0.0
I= 9	1.8D+03	-6.3D+01	8.3D-16	-4.3D+01	6.8D+02	2.2D+00	-3.2D+01	0.0	0.0	0.0	0.0
I= 10	1.1D+04	1.0D-11	-8.7D-12	-4.2D+01	4.4D-02	-4.8D+03	-5.6D-12	5.0D-12	9.0D-17	4.1D-18	-1.1D-18
I= 11	8.0D+04	1.0D-11	9.1D-16	4.3D-17	-9.6D+03	-2.4D+04	-5.6D-12	-1.8D-16	1.5D-17	3.7D-17	0.0
I= 12	3.6D+04	-3.0D-16	-8.6D+01	4.4D-12	-9.5D+03	-4.1D+03	-3.3D+01	-1.0D+01	3.4D-15	0.0	0.0
I= 13	1.6D+03	-3.2D+00	-8.4D+01	9.9D-16	1.2D+01	6.5D+02	-1.2D+01	-3.2D+01	0.0	0.0	0.0
I= 14	5.3D+00	-3.1D-18	-2.6D-17	-4.4D+01	1.5D+00	8.4D-01	-3.0D-16	0.0	0.0	0.0	0.0
I= 15	2.5D+04	7.4D-12	-6.4D-12	-4.2D+01	7.2D-01	-1.1D+04	-1.0D-12	6.0D-17	1.3D-16	6.3D-18	0.0
I= 16	5.9D+04	7.7D-12	4.1D-16	4.2D-17	-7.4D+03	-1.9D+04	-3.4D-16	-2.0D-16	4.6D-17	0.0	0.0
I= 17	9.7D+03	-6.5D+01	-2.3D+01	3.6D-12	-3.5D+03	2.8D+02	-4.3D+01	-8.4D-01	0.0	0.0	0.0
I= 18	1.2D+03	-2.1D+01	-6.4D+01	-2.0D+01	3.3D+01	4.6D+02	-4.3D+01	0.0	0.0	0.0	0.0
I= 19	3.6D+00	4.2D-16	-8.5D-16	-1.5D+01	-6.0D+00	1.1D+00	0.0	0.0	0.0	0.0	0.0
I= 20	3.4D+04	1.5D-12	3.1D-15	-3.2D+01	2.9D+00	-1.5D+04	-1.4D-13	5.4D-17	6.2D-17	3.0D-17	0.0
I= 21	4.1D+04	1.2D-13	-4.0D+01	-1.1D-14	-6.1D+03	-1.2D+04	-7.3D-14	-2.0D+01	-9.4D-15	0.0	0.0
I= 22	5.8D+02	-4.6D+01	-9.3D+00	2.8D-16	1.9D+02	7.5D+01	-2.4D+01	-3.8D+00	0.0	0.0	0.0
I= 23	8.4D+02	-7.8D+01	4.6D-16	-4.0D+01	1.8D+02	1.9D+02	-3.9D+01	0.0	0.0	0.0	0.0
I= 24	8.9D+00	-1.9D-17	-1.9D-15	-2.4D+01	-1.9D+01	4.8D+00	0.0	0.0	0.0	0.0	0.0
I= 25	3.9D+04	3.2D-13	2.6D-16	1.6D-16	-5.1D+03	-1.3D+04	-8.9D-13	-7.5D-17	2.2D-17	3.7D-17	0.0
I= 26	2.3D+04	-6.9D-14	-7.9D+01	-3.3D-13	-5.1D+03	-4.8D+03	-2.0D+01	-2.0D+01	-8.2D-15	0.0	0.0
I= 27	4.8D+02	-8.1D+00	-4.7D+01	4.6D-16	3.2D+01	1.9D+02	-7.9D+00	-1.9D+01	0.0	0.0	0.0
I= 28	3.5D+02	-3.9D+01	1.6D-16	-4.4D+01	1.6D+02	3.1D+00	-1.9D+01	0.0	0.0	0.0	0.0
I= 29	9.5D+03	-1.4D-12	2.7D-12	-4.3D+01	5.2D-01	-4.4D+03	-7.6D-13	1.4D-17	6.1D-17	6.3D-18	-4.3D-18
I= 30	3.3D+04	-1.4D-12	3.0D-16	5.4D-17	-4.4D+03	-1.1D+04	-2.9D-16	-8.4D-17	2.7D-17	2.1D-17	0.0
I= 31	1.0D+04	-4.0D+01	-4.8D+01	-2.9D-13	-4.4D+03	3.0D+02	-4.0D+01	-4.0D+00	-9.5D-15	0.0	0.0
I= 32	6.5D+02	-7.8D+00	-7.8D+01	3.5D-16	-1.4D+01	2.9D+02	-2.3D+01	-2.0D+01	0.0	0.0	0.0
I= 33	7.8D+00	-1.1D-15	-1.3D-17	-2.5D+01	-1.6D-01	1.2D+00	-1.1D-15	0.0	0.0	0.0	0.0
I= 34	1.7D+04	2.3D-13	-1.9D-15	-3.9D+01	3.0D+00	-7.7D+03	-9.1D-14	2.8D-17	4.3D-17	3.5D-17	0.0
I= 35	2.9D+04	2.7D-16	1.8D-16	7.7D-17	-3.9D+03	-9.7D+03	-1.1D-16	-1.5D-17	-1.6D-17	0.0	0.0
I= 36	5.7D+02	-7.9D+01	-9.7D+00	-1.1D-14	1.9D+02	7.1D+01	-4.4D+01	-8.2D-01	0.0	0.0	0.0
I= 37	5.6D+02	-4.5D+01	-3.9D+01	-3.2D+01	5.4D+01	2.0D+02	-4.2D+01	0.0	0.0	0.0	0.0
I= 38	8.9D+00	1.8D-15	3.0D-15	-1.5D+01	-1.2D+01	4.0D+00	0.0	0.0	0.0	0.0	0.0
I= 39	2.2D+04	8.3D-13	-9.4D-16	-2.0D+01	3.3D+00	-4.3D-17	5.0D-17	3.5D-17	0.0	0.0	0.0
I= 40	1.9D+04	3.0D-14	-6.5D+01	1.0D+02	6.7D+01	-3.2D+01	4.3D-15	0.0	0.0	0.0	0.0
I= 41	2.5D+02	-2.2D+01	-2.2D+01	8.0D+01	2.7D+01	-1.2D+01	0.0	0.0	0.0	0.0	0.0
I= 42	3.8D+02	-8.4D+01	-2.4D+01	1.8D+02	-3.2D+01	0.0	0.0	0.0	0.0	0.0	0.0
I= 43	1.2D+02	-4.0D+00	-5.9D+01	8.2D+00	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 44	1.5D+02	-2.3D+01	-4.8D+00	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 45	2.2D+02	-3.9D+01	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 46	8.4D+00	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

MATRIX A (IMAGINARY PART)

ROW	COLUMN										
	J=I	J=I+1	J=I+2	J=I+3	J=I+4	J=I+5	J=I+6	J=I+7	J=I+8	J=I+9	J=I+10
I= 1	0.0	-1.4D+00	-1.6D+04	-1.3D-02	-4.2D-03	3.7D-15	5.0D+03	2.8D+03	-2.2D+02	4.5D-15	0.0
I= 2	0.0	2.8D+00	1.6D+04	-2.5D-02	-4.2D-03	3.3D-13	-2.8D+03	-5.0D+03	2.3D+02	0.0	0.0
I= 3	0.0	2.8D-16	2.3D-18	8.4D+03	-5.6D+02	6.0D-17	2.5D-17	2.0D-18	0.0	0.0	0.0
I= 4	0.0	1.9D-18	-1.9D+02	-7.8D+03	2.8D-17	1.1D-18	-3.7D-19	0.0	0.0	0.0	0.0
I= 5	0.0	-5.0D+01	1.5D+02	8.4D-19	1.6D-18	1.7D-20	0.0	0.0	0.0	0.0	0.0
I= 6	0.0	-1.9D+03	-1.3D+03	-4.2D-02	-8.4D-03	5.3D-15	4.7D+03	3.4D+02	-2.4D+02	1.9D-14	0.0
I= 7	0.0	1.3D+03	2.0D+03	1.7D+03	1.3D+03	-2.5D-13	-5.0D+03	1.5D+01	2.2D+02	0.0	0.0
I= 8	0.0	2.0D-17	9.1D-14	1.0D+03	-1.0D+03	1.8D-17	1.8D-18	3.2D-19	0.0	0.0	0.0
I= 9	0.0	-8.0D+01	-1.6D+02	-3.0D+03	3.9D-17	3.4D-18	-1.5D-18	0.0	0.0	0.0	0.0
I= 10	0.0	4.4D+01	-3.0D+02	-1.3D-13	-1.3D-02	1.9D-16	1.5D-15	1.2D+03	-8.0D+01	-3.1D-15	1.2D-15
I= 11	0.0	-8.6D+02	2.0D+02	-7.5D-02	-1.3D-02	5.3D-15	2.4D+03	-7.7D+02	-5.3D+01	1.6D-14	0.0
I= 12	0.0	1.2D+03	2.1D-16	2.9D+03	4.8D+02	-8.3D-13	-2.8D+03	4.6D+01	1.4D+02	0.0	0.0
I= 13	0.0	1.0D-17	2.3D-13	-2.1D+03	-1.2D+03	2.3D-17	2.3D-18	-1.4D-18	0.0	0.0	0.0
I= 14	0.0	-1.9D+02	-3.5D+01	2.3D-14	3.7D-18	8.2D-20	5.2D-20	0.0	0.0	0.0	0.0
I= 15	0.0	1.4D+02	-5.6D+02	-3.3D-02	-1.7D-02	8.2D-16	9.2D+02	1.3D+03	-1.9D+02	-1.4D-14	0.0
I= 16	0.0	-1.6D+02	5.4D+02	-1.0D-01	-1.7D-02	1.1D-12	8.6D+00	-1.3D+03	1.4D+02	0.0	0.0
I= 17	0.0	2.7D+02	8.0D-17	2.8D+03	1.1D+00	3.6D-13	-9.2D+02	7.8D+01	0.0	0.0	0.0
I= 18	0.0	3.2D-18	-1.4D+02	-2.8D+03	-7.7D-18	6.1D-18	-3.7D-19	0.0	0.0	0.0	0.0
I= 19	0.0	-9.7D+01	9.1D+01	3.1D-19	1.6D-18	1.1D-19	0.0	0.0	0.0	0.0	0.0
I= 20	0.0	-3.2D+01	-3.4D+02	-8.4D-02	-2.1D-02	8.8D-16	1.8D+03	4.5D+02	-2.4D+02	-3.6D-14	0.0
I= 21	0.0	1.1D+02	4.2D+02	-1.3D-01	7.5D+02	2.7D-13	-1.5D+03	-7.5D+02	2.3D+02	0.0	0.0
I= 22	0.0	1.2D-17	4.0D-18	1.4D+03	-3.0D+02	1.2D-17	1.3D-18	9.9D-19	0.0	0.0	0.0
I= 23	0.0	6.4D-19	-2.2D+02	-1.8D+03	2.0D-17	4.3D-19	-2.0D-18	0.0	0.0	0.0	0.0
I= 24	0.0	-2.7D+00	2.2D+02	-5.4D-19	1.1D-18	2.3D-19	0.0	0.0	0.0	0.0	0.0
I= 25	0.0	-2.6D+02	-4.4D+01	-1.5D-01	-2.5D-02	1.9D-15	1.9D+03	-2.0D+02	-1.5D+02	-4.6D-14	0.0
I= 26	0.0	2.2D+02	1.2D+02	1.1D+03	3.8D+02	2.8D-13	-1.7D+03	3.1D+01	1.9D+02	0.0	0.0
I= 27	0.0	-6.5D-18	-5.9D-14	-3.7D+02	-5.0D+02	3.5D-18	5.0D-18	-1.6D-18	0.0	0.0	0.0
I= 28	0.0	-1.4D+02	-9.8D+01	-6.3D+02	1.9D-17	1.8D-18	-1.8D-18	0.0	0.0	0.0	0.0
I= 29	0.0	9.2D+01	-1.8D+01	-2.9D-02	-2.9D-02	2.3D-16	1.0D-15	9.8D+02	-1.4D+02	7.6D-15	2.2D-14
I= 30	0.0	-1.0D+02	1.1D+02	-1.8D-01	-2.9D-02	2.0D-15	5.5D+02	-5.8D+02	4.2D+01	-9.3D-14	0.0
I= 31	0.0	1.6D+02	1.5D-16	1.5D+03	1.1D+02	1.9D-13	-9.7D+02	6.2D+01	7.8D+01	0.0	0.0
I= 32	0.0	6.6D-18	-7.9D+01	-1.6D+03	-6.4D-18	7.5D-18	-2.5D-20	1.2D-18	0.0	0.0	0.0
I= 33	0.0	-1.4D+02	2.8D+01	2.1D-18	1.4D-18	2.8D-19	-7.6D-21	0.0	0.0	0.0	0.0
I= 34	0.0	1.9D+02	-1.9D+02	-1.0D-01	-3.3D-02	8.0D-16	8.5D+02	4.8D+02	-2.2D+02	4.5D-15	0.0
I= 35	0.0	2.8D+00	2.0D+02	-2.0D-01	-3.3D-02	5.8D-14	-4.6D+02	-8.6D+02	2.3D+02	0.0	0.0
I= 36	0.0	1.9D-17	5.4D-18	1.4D+03	-9.2D+01	7.4D-18	8.1D-18	-2.5D-18	0.0	0.0	0.0
I= 37	0.0	2.1D-18	-1.9D+02	-1.3D+03	6.0D-18	1.2D-17	-4.5D-18	0.0	0.0	0.0	0.0
I= 38	0.0	-4.9D+01	1.5D+02	-5.1D-19	2.3D-18	-5.3D-19	0.0	0.0	0.0	0.0	0.0
I= 39	0.0	-1.8D+01	-6.2D+01	-1.9D-01	4.2D+02	8.5D+02	-2.2D+02	-1.7D+01	0.0	0.0	0.0
I= 40	0.0	6.8D+01	1.0D+02	-1.9D-01	-2.4D+02	-9.9D+02	2.3D+02	0.0	0.0	0.0	0.0
I= 41	0.0	1.4D-17	1.9D-18	2.8D-18	3.7D-18	3.9D-19	0.0	0.0	0.0	0.0	0.0
I= 42	0.0	-6.7D-19	1.3D-17	8.2D-18	-1.9D-19	0.0	0.0	0.0	0.0	0.0	0.0
I= 43	0.0	-3.1D-18	5.0D-18	1.4D-18	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 44	0.0	5.0D-18	1.3D-18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 45	0.0	-6.3D-19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 46	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

MATRIX B

ROW	COLUMN J=I	J=I+1	J=I+2	J=I+3	J=I+4	J=I+5	J=I+6	J=I+7	J=I+8	J=I+9	J=I+10
I= 1	1.7D+01	2.4D-16	1.0D-02	5.7D-18	5.2D-03	4.2D+00	6.0D-17	0.0	0.0	0.0	0.0
I= 2	2.9D+01	0.0	0.0	0.0	2.1D+00	5.2D+00	0.0	0.0	0.0	0.0	0.0
I= 3	2.9D-01	2.4D-18	0.0	0.0	3.1D-02	7.8D-02	5.5D-18	0.0	0.0	0.0	0.0
I= 4	2.9D-01	3.7D-17	-3.2D-17	1.9D-18	3.1D-02	7.8D-02	4.4D-18	0.0	0.0	0.0	0.0
I= 5	2.9D-01	3.8D-17	0.0	0.0	3.1D-02	7.8D-02	0.0	0.0	0.0	0.0	0.0
I= 6	1.0D+01	3.6D-16	0.0	-1.5D-17	6.4D-01	3.1D+00	1.8D-16	0.0	0.0	0.0	0.0
I= 7	8.5D+00	-2.4D-17	4.8D-17	-6.2D-16	1.3D+00	1.9D+00	9.1D-18	0.0	0.0	0.0	0.0
I= 8	5.9D-01	-2.4D-17	0.0	0.0	5.2D-02	1.3D-01	0.0	0.0	0.0	0.0	0.0
I= 9	5.9D-01	1.8D-17	0.0	3.3D-18	5.2D-02	1.3D-01	1.5D-18	0.0	0.0	0.0	0.0
I= 10	1.6D+00	9.8D-16	-8.2D-16	0.0	5.2D-02	5.5D-01	5.2D-16	-4.7D-16	0.0	0.0	0.0
I= 11	7.5D+00	9.4D-16	0.0	0.0	9.0D-01	2.2D+00	5.2D-16	0.0	0.0	0.0	0.0
I= 12	3.7D+00	7.2D-18	0.0	-4.1D-16	9.0D-01	6.0D-01	2.1D-18	0.0	0.0	0.0	0.0
I= 13	8.8D-01	0.0	0.0	0.0	7.3D-02	1.8D-01	0.0	0.0	0.0	0.0	0.0
I= 14	8.8D-01	7.2D-18	0.0	0.0	7.3D-02	1.8D-01	1.3D-17	0.0	0.0	0.0	0.0
I= 15	2.9D+00	7.0D-16	-6.0D-16	4.5D-18	7.3D-02	1.1D+00	9.8D-17	0.0	0.0	0.0	0.0
I= 16	5.6D+00	7.2D-16	0.0	0.0	7.0D-01	1.7D+00	0.0	0.0	0.0	0.0	0.0
I= 17	1.8D+00	3.6D-17	0.0	-3.4D-16	4.0D-01	2.4D-01	1.3D-17	0.0	0.0	0.0	0.0
I= 18	1.2D+00	-4.9D-17	9.6D-17	-4.6D-17	9.4D-02	2.4D-01	1.6D-17	0.0	0.0	0.0	0.0
I= 19	1.2D+00	-4.9D-17	0.0	0.0	9.4D-02	2.4D-01	0.0	0.0	0.0	0.0	0.0
I= 20	3.5D+00	1.4D-16	0.0	5.9D-18	9.4D-02	1.4D+00	1.7D-17	0.0	0.0	0.0	0.0
I= 21	4.0D+00	1.9D-16	-1.6D-16	0.0	5.7D-01	1.2D+00	6.7D-17	-6.0D-17	0.0	0.0	0.0
I= 22	1.5D+00	1.8D-16	0.0	0.0	1.2D-01	2.9D-01	6.7D-17	0.0	0.0	0.0	0.0
I= 23	1.5D+00	1.2D-17	0.0	-5.3D-17	1.2D-01	2.9D-01	3.4D-18	0.0	0.0	0.0	0.0
I= 24	1.5D+00	0.0	0.0	0.0	1.2D-01	2.9D-01	0.0	0.0	0.0	0.0	0.0
I= 25	3.7D+00	3.1D-17	0.0	0.0	4.8D-01	1.2D+00	8.4D-17	0.0	0.0	0.0	0.0
I= 26	2.9D+00	2.2D-16	-1.9D-16	3.0D-17	4.8D-01	6.9D-01	1.9D-17	0.0	0.0	0.0	0.0
I= 27	1.8D+00	2.3D-16	0.0	0.0	1.4D-01	3.4D-01	0.0	0.0	0.0	0.0	0.0
I= 28	1.8D+00	5.4D-17	0.0	-6.7D-17	1.4D-01	3.4D-01	1.9D-17	0.0	0.0	0.0	0.0
I= 29	2.2D+00	-1.3D-16	2.6D-16	-6.7D-17	1.4D-01	6.2D-01	7.3D-17	0.0	0.0	0.0	0.0
I= 30	3.2D+00	-1.3D-16	0.0	0.0	4.2D-01	1.0D+00	0.0	0.0	0.0	0.0	0.0
I= 31	2.4D+00	6.3D-17	0.0	2.6D-17	4.2D-01	3.9D-01	4.4D-18	0.0	0.0	0.0	0.0
I= 32	2.1D+00	2.7D-16	-2.2D-16	0.0	1.6D-01	3.9D-01	9.1D-17	-8.2D-17	0.0	0.0	0.0
I= 33	2.1D+00	2.6D-16	0.0	0.0	1.6D-01	3.9D-01	9.1D-17	0.0	0.0	0.0	0.0
I= 34	2.5D+00	2.4D-17	0.0	-7.2D-17	1.6D-01	8.2D-01	1.1D-17	0.0	0.0	0.0	0.0
I= 35	2.8D+00	0.0	0.0	0.0	3.7D-01	9.2D-01	0.0	0.0	0.0	0.0	0.0
I= 36	2.3D+00	2.0D-17	0.0	0.0	1.8D-01	4.4D-01	3.1D-17	0.0	0.0	0.0	0.0
I= 37	2.3D+00	2.9D-16	-2.5D-16	1.1D-17	1.8D-01	4.4D-01	2.5D-17	0.0	0.0	0.0	0.0
I= 38	2.3D+00	3.0D-16	0.0	0.0	1.8D-01	4.4D-01	0.0	0.0	0.0	0.0	0.0
I= 39	2.4D+00	7.7D-17	0.0	-8.7D-17	8.9D-02	0.0	0.0	0.0	0.0	0.0	0.0
I= 40	2.5D+00	-1.1D-16	2.2D-16	9.9D-02	9.9D-02	3.4D-17	0.0	0.0	0.0	0.0	0.0
I= 41	2.6D+00	-1.1D-16	-1.2D-16	6.0D-01	9.9D-02	0.0	0.0	0.0	0.0	0.0	0.0
I= 42	2.6D+00	1.1D-16	-3.5D-18	6.0D-01	9.9D-02	0.0	0.0	0.0	0.0	0.0	0.0
I= 43	2.5D+00	1.8D-16	-1.4D-16	6.0D-01	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 44	1.4D+00	1.7D-16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 45	1.4D+00	1.0D-17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
I= 46	1.4D+00	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0