BT/PEF-8811

UM PROGRAMA PARA SOLUÇÃO DO PROBLEMA GENERALIZADO DE AUTOVALORES E AUTOVETORES PARA MATRIZES REAIS DENSAS

(*) Priscila Goldenberg
Reyolando Brasil
Marcia Cimerman
(recebido em 14/06/88)

1 - INTRODUÇÃO

O problema de autovalores e autovetores generalizado, a saber, o da determinação de uma solução não trivial da equação matricial

$$Ax = \lambda Bx \qquad (1)$$

onde A e B são matrizes reais dadas, é um problema comum em aplicações práticas.

Diversos métodos para a resolução deste problema encontram-se descritos na literatura [1,2,3], basicamente dependendo das condições impostas sobre as matrizes A e B.

O presente trabalho se propõe a apresentar um programa computacional AUTOGE(*) para a resolução do problema (1), onde A e B são matrizes reais densas de ordem nxn sendo A e B inversíveis. Tal programa será subdividido em diversas partes, de modo a aproveitar características particulares das matrizes A e B, como por exemplo, ser simétrica, definida positiva, etc.

Gostaríamos de mencionar que diversas das subrotinas aqui utilizadas foram extraídas de [3].

O programa AUTOGE escrito em linguagem FORTRAN foi implementado num microcomputador compatível com PC-XT(IBM) e foi dividido em duas partes:

a) Redução do problema generalizado (1) para a forma padrão

$$Cy = \mu y \qquad (2)$$

(*) O programa AUTOGE encontra-se com os autores à disposição dos interessados

nos casos em que isto é possível.

b) Resolução do problema padrão (2).

Conforme já mencionamos, a escolha do método a ser utilizado, depende das características das matrizes A e B.

Na seção seguinte faremos uma descrição detalhada dos métodos utilizados em ambas as partes acima mencionadas.

2 - DESCRIÇÃO DO MÉTODO

É conveniente neste ponto revermos algumas convenções e definições. A menos de menção contrária, trabalharemos com matrizes reais com nxn elementos, e com vetores reais nx1.

A matriz transposta de $A = (a_{ij})$ será $A^T = (a_{ji})$ e a matriz inversa A^{-1} é uma matriz (quando existir) tal que $A^{-1}A = AA^{-1} = I$ (I: matriz identidade nxn).

Os seguintes tipos de matrizes serão importantes:

A simétrica $\Leftrightarrow A^T = A$.

A triangular superior $\Leftrightarrow a_{ij} = 0$ se $i>j$.

A triangular inferior $\Leftrightarrow a_{ij} = 0$ se $i<j$.

A tridiagonal $\Leftrightarrow a_{ij} = 0$ se $\begin{cases} i \neq j \\ \text{e} \\ i \neq j+1 \\ \text{e} \\ i \neq j-1 \end{cases}$

A diagonal $\Leftrightarrow a_{ij} = 0$ se $i \neq j$.

A definida positiva $\Leftrightarrow x^T Ax > 0$, para todo $x \in R^n$, $x \neq 0$.

A ortogonal $\Leftrightarrow A^T A = I$.

O número de condições de uma matriz inversível A, denotado por cond(A) é definido por

$$\text{cond}(A) = \| A \| \; \| A^{-1} \|$$

onde adotaremos a norma da matriz A, $\| A \|$ como sendo

$$\| A \| = \max_i \sum_j |a_{ij}|$$

2.1 Redução do problema generalizado para o problema padrão

Nós iniciamos o tratamento do problema generalizado $Ax = \lambda Bx$, com a redução do mesmo para a forma padrão $Cy = \mu y$.

Sob o ponto de vista teórico, é imediato que desde que A e B sejam inversíveis, esta redução possa ser efetuada de uma das duas maneiras seguintes:

a) sendo $Ax = \lambda Bx$, podemos escrever

$$B^{-1}Ax = \lambda x$$

Definindo

$$C = B^{-1}A$$

$$\mu = \lambda$$

$$y = x$$

recaímos na forma padrão

b) Sendo $Ax = \lambda Bx$, podemos escrever

$$\frac{1}{\lambda} x = A^{-1}Bx$$

Definindo

$$C = A^{-1}B$$

$$\mu = 1/\lambda$$

$$y = x$$

recaímos na forma padrão.

A escolha de qual das duas formas deve ser utilizada depende das vantagens que podemos obter com a inversão de A ou B. O programa computacional AUTOGE decidirá qual forma utilizar, seguindo o roteiro dos casos que descreveremos a seguir iniciando pelos testes da matriz B e depois da matriz A. Se não houver nenhuma característica que mostre a vantagem de utilizar um caso sobre o outro, a decisão de inverter A ou B ficará a critério do número de condição de ambas, ou seja, inverteremos a matriz cujo número de condição for menor.

Vamos no que segue, descrever os casos considerados e os métodos utilizados em cada caso.

Caso 2.1.1 - Se $B = I$ o problema já está na forma padrão. Se $B \neq I$ e $A = I$ ainda estamos na forma padrão.

Caso 2.1.2 - Suponhamos $B \neq I$, $A \neq I$ e B simétrica definida positiva. Podemos então utilizar a decomposição de Cholesky [7] que consiste em decompor B na forma.

$$B = LL^T$$

sendo L uma matriz triangular inferior não singular.

Desta forma podemos escrever o problema

$$Ax = \lambda Bx$$

da seguinte forma

$$Ax = \lambda LL^T x$$

Daí

$$AL^{-T}L^T x = \lambda LL^T x \qquad \text{onde}$$

$$L^{-T} = (L^T)^{-1}$$

Definindo

$$y = L^T x$$

$$\mu = \lambda$$

$$C = L^{-1} A L^{-T}$$

recaímos no problema padrão.

Observemos que os autovalores do problema padrão e do generalizado são os mesmos e que se y é um autovetor do problema padrão, $x = L^{-T}y$ é autovetor do problema generalizado.

Caso B não verifique as condições de Cholesky, testaremos estas condições para a matriz A. Caso A satisfaça as condições de Cholesky, recaímos no problema padrão de modo análogo; caso contrário passamos ao caso 2.1.3 seguinte.

Caso 2.1.3 - Se A e B não satisfazem as condições de Cholesky a decomposição descrita no caso 2 não poderá ser efetuada. Neste caso o método a ser utilizado é mais trabalhoso que o caso anterior. Vamos supor que a matriz B seja diagonalizável, isto é, B possa ser decomposta sob a forma

$$B = P\Lambda P^T \tag{3}$$

onde Λ é uma matriz diagonal e P é uma matriz ortogonal (se os autovalores de B forem reais e distintos sempre é possível efetuarmos a decomposição (3)).

Neste caso podemos escrever o problema generalizado

$$Ax = \lambda Bx$$

sob a forma

$$Ax = \lambda P\Lambda P^T x$$

Daí, lembrando que P é ortogonal, temos

$$P^T A P P^T x = \lambda \Lambda P^T x$$

ou $$\Lambda^{-1}P^{T}APP^{T}x = \lambda P^{T}x$$

Definindo

$$y = P^{T}x$$

$$\mu = \lambda$$

$$C = \Lambda^{-1}P^{T}AP^{T}$$

recaímos no problema padrão.

Neste caso os autovalores do problema padrão e do generalizado são iguais e se y é um autovetor do problema padrão,

$$x = Py$$

é autovetor do problema generalizado.

Se B não é diagonalizável, verificamos se A satisfaz esta condição. Caso isto ocorra podemos repetir o processo utilizando a matriz A para recair no caso padrão.

Caso contrário passamos ao caso 2.1.4 seguinte.

Caso 2.1.4 - Finalmente se nem A nem B satisfizerem as condições descritas nos casos acima, teremos que inverter B ou A diretamente e, neste caso, conforme já mencionamos, a decisão de qual matriz inverter dependerá do número de condição das matrizes.

Comentário final - Se analizarmos, nos casos 2 e 3 também foi efetuada inversão das matrizes B ou A como no caso 4, entretanto nós utilizamos certas características de B ou de A. Observemos por exemplo no caso 3 a ortogonalidade de P, permite reduzir a inversão à simples transposição de P, e sendo Λ diagonal a sua inversão é imediata.

2.2 Resolução do problema padrão

A resolução do problema padrão

$$Cy = \mu y$$

será dividida em 2 casos. O primeiro quando C é simétrica e o outro quando C não é simétrica.

Caso 2.2.1 - C simétrica

No caso em que a matriz C é simétrica o método utilizado foi uma associação do método de tridiagonalização de Householder com o método de decomposição QL. Segundo Wilkinson [1] este método é o mais eficiente na resolução do problema em consideração.

Método de tridiagonalização de Householder

Este método consiste basicamente em reduzir a matriz $C = C_1$ a uma matriz simétrica tridiagonal C_{n-2} por meio de n-2 transformações ortogonais (isto é $C_{i+1} = P_i C_i P_i$, $i = 1,\ldots,n-2$ onde P_i são matrizes ortogonais convenientemente definidas [1,4]).

É claro que se Z é um autovetor da matriz tridiagonal C_{n-1} então $P_1 P_2 \ldots P_{n-2}$ Z é um autovetor de C_1.

Método de decomposiçaõ QL

O método de decomposição QL para matrizes simétricas [1,5,6] é baseado na observação que se

$$C = QL$$

$$\text{e} \qquad F = LQ$$

onde Q é ortogonal e L triangular inferior então

$$F = LQ = Q^T C Q$$

ou seja, F é similar a C. Repetindo sucessivamente o resultado acima, uma sequência de matrizes similares a uma dada matriz C_1 pode ser derivada pelas relações

$$C_s = Q_s L_s, \quad C_{s+1} = L_s Q_s = Q_s^T C_s Q_s$$

onde Q_s é ortogonal e L_s é triangular inferior. Em geral, quando

C_1 possui autovalores com módulos distintos a matriz C_s tende à forma triangular inferior, sendo que a matriz limite C_∞ possui na diagonal os autovalores de C_1. Quando C_1 possui autovalores com mesmo módulo a matriz limite não é em geral triangular mas correspondendo a $|\mu_i|$ de multiplicidade p, C_s possui associada uma matriz bloco diagonal de ordem p no qual os autovalores tendem aos correspondentes p autovalores.

A velocidade de convergência do algoritmo nesta forma é em geral inadequada. A velocidade de convergência pode ser melhorada se trabalharmos com C_s-K_sI (para uma escolha conveniente de K_s) em cada estágio no lugar de C_s. Neste caso o processo resultará

$$C_s-K_sI = Q_sL_s$$

$$C_{s+1} = L_sQ_s+K_sI$$

de onde

$$C_{s+1} = Q_s^TA_sQ_s$$

Caso 2.2.2 - C não simétrica

No caso em que a matriz C não é simétrica o método utilizado foi uma associação do método de redução da matriz C à forma de Hessenberg com o método de decomposição QR.

Método de redução de uma matriz geral à forma de Hesenberg

Diversos métodos para a resolucão do problema de autovalores e autovetores de uma matriz geral,ficam bastante simplificados se a matriz C for inicialmente transformada na forma superior de Hessenberg i.e.transformada a uma matriz H tal que $h_{ij}=0\,(i>i+1)$.A redução pode ser obtida utilizando operações elementares (permutação de linhas e/ou colunas e operações aritméticas sobre as linhas e/ou colunas)na

matriz C.

Algoritmo QR para matrizes reais de Hessenberg

O método de decomposição QR [5,6] é baseado no fato que se

$$C = QR \quad e \quad F = RQ$$

onde Q é unitária e R é triangular superior então

$$F = RQ = Q^T CQ$$

é similar a C. Repetindo sucessivamente o resultado obtemos

$$C_s = Q_s R_s \ , \ C_{s+1} = R_s Q_s = Q_s^T C_s Q_s$$

e em geral C_s tende a forma triangular superior.

Como no algoritmo QL descrito para o caso de matrizes simétricas, este algoritmo será melhor se considerarmos

$$Q_s (C_s - K_s I) = R_s$$

$$C_{s+1} = R_s Q_s^T + K_s I$$

dando

$$C_{s+1} = Q_s C_s Q_s^T$$

onde Q_s é ortogonal, R_s é triangular superior e K_s é um deslocamento da origem. Quando a matriz C_1 é da forma superior de Hessenberg todas as C_s também são. Segundo Wilkinson [2] o volume de trabalho envolvido em cada passo do algoritmo QR fica reduzido se a matriz está na forma de Hessenberg. Este é o motivo da associação dos métodos.

3. DESCRIÇÃO E UTILIZAÇÃO DO PROGRAMA AUTOGE

O programa AUTOGE para resolução do problema de autovalores e autovetores generalizado é constituído das seguintes etapas:

1) Comandos de leitura/impressão de dados.

2) Decisão sobre o problema a ser resolvido e respectivo método de solução, e chamada das subrotinas adequadas.

A seguir se descreve a forma de utilização do programa.

3.1 Leitura e verificação de dados

1) Com o computador no sistema operacional e o disco contendo o programa AUTOGE-EXE, basta digitar e entrar o código AUTOGE.

2) Após a tela de apresentação, é solicitada entrada de N, ordem das matrizes.

3) O programa solicita informações sobre a matriz A ser ou não identidade

- SE FOR MATRIZ IDENTIDADE ENTRE 0
- SE NÃO ENTRE 1
- SE NÃO SABE ENTRE 2 QUE VAMOS VERIFICAR

Caso a resposta seja 0 o programa salta para 7).

4) O programa solicita informações sobre a matriz A ser ou não simétrica

- SE FOR SIMÉTRICA ENTRE 0
- SE NÃO ENTRE 1
- SE NÃO SABE ENTRE 2 QUE VAMOS VERIFICAR

5) Caso a resposta ao ítem 4) seja 0, é solicitada a entrada dos elementos de A, linha a linha, a partir da diagonal, para aproveitamento de sua simetria.

Caso contrário a entrada dos elementos de A é feita por linha completa. O formato dos elementos de A é totalmente livre.

6) É feita impressão dos elementos de A para conferência, e é feita a pergunta

QUER MUDAR ALGUM ELEMENTO DA MATRIZ A

(1 = SIM, 0 = NÃO)

na resposta positiva, é solicitado

ENTRE I, J DO ELEMENTO A MUDAR. Entrados os dois índices, é solicitado o novo valor de A(I,J). Essa sequência de correção se repete até resposta negativa à pergunta inicial.

7) O programa solicita informações sobre a matriz B quanto a ser ou não identidade e simétrica na mesma sequência de 3) e 4), e lê os elementos de B na mesma forma de 5), imprimindo-os para conferência e admitindo correção como em 6).

3.2 Impressão dos resultados

O programa AUTOGE imprime inicialmente as matrizes A e B de entrada segundo o comando FORMAT 160 e 370. Se A ou B forem singulares é impressa uma mensagem e o programa termina.

A seguir são impressos os autovalores e autovetores calculados em formato livre.

Os números de comando FORMAT 160 e 370 devem ser mudados dependendo da ordem das matrizes.

3.3 Desenvolvimento do Programa

As etapas do programa seguem passo a passo as etapas descritas na secção 2.

As subrotinas utilizadas são:

TRED2 , TQL2 , DIAGO, MULMA, FADEEV, COND, ELMHES,ELTRAN, HQR2, REDUC1 , REBAKA, RG, DENT, DEFPO, SIM, AINV.

A listagem do programa encontra-se em anexo no final.

Descrição das subrotinas

1) Subrotina COND

Objetivo: Obtenção do número de condição ρ da matriz inversível A, definido por

$$\rho = \| A \|_\infty \cdot \| A^{-1} \|_\infty$$

$$\| A \|_\infty = \max_i \sum_j |aij|$$

Utilização: CALL COND (N,A,RO)

Argumentos: N = ordem da matriz A

A = Matriz real de entrada de ordem N

RO = Variável real de saída, contendo o número da condição.

Subrotina Requerida - AINV.

2) Subrotina MULMA

Objetivo: Multiplicação de matrizes quadradas

Utilização: CALL MULMA (NA,NB,A,B,C)

Argumentos: NA = ordem da matriz A

NB = ordem da matriz B

A = Matriz real de entrada de ordem NA

B = Matriz real de entrada de ordem NB

C = Matriz real de saída, contendo produto A*B

3) Subrotina DEFPO

Objetivo: Saber se a matriz de entrada A é definida positiva

Utilização: CALL DEFPO (N,A,IPO)

Argumentos: N = ordem da matriz A

A = matriz real de entrada de ordem N

IPO = variável inteira de saída contendo a informação sobre a possibilidade da matriz ser definida positiva.

Se IPO = 0 a matriz é definida positiva.

Se IPO = 1 a matriz não é definida positiva.

Algoritmo Utilizado: É testado se os subdeterminantes principais de A são positivos.

Subrotinas requeridas: FADEEV e MULMA.

4) Subrotina DENT

Objetivo: Saber se a matriz de entrada A é a matriz identidade.

Utilização: CALL DENT (N,A,IDENT)

Argumentos: N = ordem da matriz A

A = matriz real de entrada de ordem N

IDENT = variável inteira de saída contendo a informação sobre a possibilidade da matriz A ser a matriz identidade.

Se IDENT = 0 a matriz é identidade.

Se IDENT = 1 a matriz não é identidade

Subrotinas requeridas: não há.

5) Subrotina TRED2

Objetivo: Redução de uma Matriz Real Simétrica a uma Matriz Tridiagonal Simétrica, usando e acumulando transformações ortogonais (Algoritmo de Househouder).

Utilização: CALL TRED2 (NM,N,A,D.E.Z)

Argumentos:NM = variável inteira de entrada igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem da ma

triz A: N deve ser menor que NM.

A = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo menos N. Contém a matriz A a ser operada.

D = variável real unidimensional de entrada com dimensão mínima N contendo os elementos da diagonal da matriz tridiagonal resultante.

E = variável real unidimensional de saída com dimensão mínima N contendo, em suas últimas N - 1 posições, os elementos subdiagonais da matriz tridiagonal. E(1) é nulo.

Z = variável bidimensional real com dimensão de linhas igual a NM e de coluna pelo menos N. Contém a matriz de transformação ortogonal produzida na reduçaõ à forma tridiagonal.

Subrotinas requeridas: nenhuma.

6) Subrotina TQL2

Objetivo: Determinar os autovalores e autovetores de uma matriz tridiagonal simétrica usando o método QL.

Utilização: CALL TQL2 (NM,D,E,Z,IERR)

Argumentos: NM = variável inteira de entrega igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem da matriz A. N deve ser menor que NM.

D = variável real unidimensional com dimensão mínima N contendo na entrada os elementos da diagonal da matriz tridiagonal simétrica. Na saída, contém os autovalores da matriz em ordem crescente.

E = variável real unidimensional com dimensão mínima N contendo, na entrada, nas últimas N - 1 posições os elementos subdiagonais da matriz tridiagonal. E(1) é nulo.

Z = variável bidimensional real com dimensão de linhas igual a NM e de coluna pelo menos N. Contém a matriz de transformação ortogonal produzida na redução à forma tradicional por TRED2 na entrada, e na saída contém os autovetores auto normais da matriz simétrica.

IERR = variável inteira de saída nula se o processo se completou normalmente e igual ao número de ordem de autovetor que não convergiu em 30 iterações.

Subrotinas requeridas: nenhuma.

7) Subrotina ELMHES

Objetivo: Reduz uma matriz real geral à forma superior de Hessenberg usando transformações elementares estabilizadas.

Utilização: CALL ELMHES (NM,N,LOW,IGH,A,INT)

Argumentos: NM = variável inteira de entrada igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem da matriz A. N deve ser menor que NM.

LOW, IGH = variáveis inteiras de entrada dando índices a serem utilizados quando se faz balanceamento prévio da matriz a ser operada. Neste programa LOW = 1 e IGH = N.

A = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo

menos N. Contém a matriz A a ser reduzida à forma de Hessenberg na entrada, e a matriz superior reduzida na saída.

INT = variável inteira unidimensional de saída de ordem igual a IGH indicando linhas e colunas que foram reordenadas no processo

Subrotinas requeridas: nenhuma.

8) Subrotina ELTRAN

Objetivo: Acumular as transformações elementares de similaridade usadas na redução de uma matriz real geral a forma de Hessenberg por ELMHES.

Utilização: CALL ELTRAN (NM,N,LOW,IGH,A,INT,Z)

Argumentos: NM = variável inteira de entrada igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem da matriz A. N deve ser menor que NM.

LOW, IGH = variáveis inteiras de entrada dando índices a serem utilizados quando se faz balanceamento prévio da matriz a ser operada. Neste programa LOW = 1 e IGH = N.

A = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo menos N. Contém no triangulo inferior os multiplicadores utilizados na redução à forma de Hessenberg.

INT = variável inteira unidimensional de dimensão mínima igual a IGH, identificando as linhas e colunas reordenadas pela subrotina ELMHES.

Z = variável bidimensional real com dimensão de li

nhas igual a NM e de coluna pelo menos N. Contém a matriz de transformação produzida na redução a forma de Hessenberg por ELMHES.

Subrotinas requeridas: nenhuma.

9) Subrotina HQR2

Objetivo: Computar os autovalores e autovetores de uma matriz real superior de Hessenberg usando o método QR.

Utilização: CALL HQR2 (NM,N,LOW,IGH,H,WR,WI,Z,IERR)

Argumentos: NM = variável inteira de entrada igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem da matriz H. N deve ser menor que NM.

LOW, IGH = variáveis inteiras de entrada dando índices a serem utilizados quando se faz balanceamento prévio da matriz a ser operada. Neste programa LOW = 1 e IGH = N.

H = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo menos N. Contém a matriz a ser operada.

WR, WI = variáveis unidimensionais reais de dimensão pelo menos igual a N, contendo as partes reais e imagiárias, respectivamente, dos autovalores procurados no problema.

Z = variável bidimensional real com dimensão de linhas igual a NM e de coluna pelo menos N. Contém a matriz de transformação produzida na redução à forma de Hessenberg pela subro

tina ELTRAN na entrada, e na saída contém as partes reais e imaginárias dos autovetores da matriz real geral original.

IERR = variável inteira de saída igual a código de erro em completar o processo.

Subrotinas requeridas: nenhuma.

10) Subrotina REDUC1

Objetivo: Redução do problema generalizado, simétrico, de autovalores a forma standard utilizando a decomposiçao de Cholesky.

Utilização: CALL REDUC1 (NM,N,A,B,DL,IERR)

Argumentos: NM = variável inteira de entrada igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem das matrizes A e B. N deve ser menor que NM.

A = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo menos N. Contém a matriz A a ser operada na entrada, e o triangulo inferior da matriz do problema standard correspondente na saída.

B = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo menos N. Contém a matriz B a ser operada, positivo definida, na entrada, e os elementos subdiagonais da matriz triangular inferior L na saída.

DL = variável real unidimensional de saída, com dimensão mínima N contendo os elementos da

diagonal da matriz triangular inferior L produzida na redução de Cholesky.

IERR = variável inteira de saída usada se B, talvez por problemas de arredondamento, não é positivo definida.

Subrotinas requeridas: nenhuma.

11) Subrotina REBAKA

Objetivo: Computar os autovetores do problema generalizado simétrico original, a partir dos autovetores do problema standard correspondente, quando a redução de um para outro foi feita por Cholesky utilizando a subrotina REDUC1.

Utilização: CALL REBAKA (NM,N,B,DL,Z)

Argumentos: NM = variável inteira de entrada igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem da matriz B. N deve ser menor que NM.

B = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo menos N. Contém os elementos subdiagonais da matriz L, triangular inferior, obtida na subrotina REDUC 1

DL = variável real unidimensional de entrada com dimensão mínima N contendo os elementos da diagonal da matriz triangular inferior L, obtida na subrotina da REDUC1

Z = variável bidimensional real com dimensão de linhas igual a NM e de coluna pelo menos N. Contém na entrada os autovetores

do problema standard obtido por REDUC1 , e na saída os do problema generalizado original.

Subrotinas requeridas: nenhuma.

12) Subrotina RG

Objetivo: Chamar a sequência recomendada de subrotinas para determinar os autovalores e autovetores de matriz real geral utilizando redução à forma de Hessenberg e o método QR.

Utilização: CALL RG (NM,N,A,WR,WI,MATZ,Z,IV1,FV1,IERR)

Argumentos: NM = variável inteira de entrada igual à dimensão das linhas das matrizes A, B e Z do programa principal.

N = variável inteira de entrada igual à ordem da matriz A. N deve ser menor que NM.

A = variável bidimensional real de entrada com dimensão de linhas igual a NM e de colunas pelo menos N. Contém a matriz A a ser operada.

WR, WI = variáveis unidimensionais reais de dimensão pelo menos igual a N, contendo as partes reais e imagiárias,respectivamente,dos autovalores procurados no problema.

MATZ = variável inteira de entrada igual a 1 neste programa.

Z = variável bidimensional real com dimensão de linhas igual a NM e de coluna pelo menos N. Contém as partes reais e imaginárias dos autovetores procurados no problema.

IVl = variável inteira unidimensional temporária de dimensão mínima igual a N.

FVl = variável real unidimensional temporária de dimensão mínima igual a N.

IERR = variável inteira de saída igual a código de erro em completar os cálculos.

Subrotinas requeridas: ELMHES, ELTRAN e HQR2 .

13) Subrotina FADEEV

Objetivo: Obtenção do determinante de uma matriz quadrada A qualquer.

Utilização: CALL FADEEV (N,A,DETA)

Argumentos: N = ordem da matriz A.

A = matriz real de entrada de ordem N.

DETA = variável real de saída contendo o valor do determinante de A

Algoritmo utilizado: O algoritmo utilizado foi o de Fadeev |8| baseado no fato que

$$adj(-A) = det(-A)(-A)^{-1}$$

onde

adj(-A) é a matriz adjunta de (-A)

Subrotina requerida: MULMA

14) Subrotina SIM

Objetivo: Saber se a matriz de entrada A é simétrica.

Utilização: CALL SIM (N,A,ISI)

Argumentos: N = ordem da matriz A.

A = matriz real de entrada de ordem N.

ISI = variável inteira de saída contendo a informação sobre a possibilidade da simetria da matriz.

Se ISI = 0 a matriz é simétrica

Se ISI = 1 a matriz não é simétrica

Subrotinas requeridas: nenhuma.

15) Subrotina AINV

Objetivo: Obtenção da matriz A^{-1} inversa da matriz A.

Utilização: CALL AINV (N,A,AM1)

Argumentos: N = ordem da matriz A.

A = matriz real de entrada de ordem N.

AM1 = matriz real de saída contendo os elementos da matriz inversa da matriz A.

Algoritmo utilizado: Algoritmo triangularização de Gauss [9].

Subrotinas requeridas: nenhuma.

16) Subrotina DIAGO

Objetivo: Obtenção da matriz $C=\Lambda^{-1}P^{T}AP$ no caso em que a matriz A é diagonalizável.

Utilização: CALL DIAGO (N,A,X,Y,DINV,ATIL,IDI)

Argumentos: N = ordem da matriz A.

A = matriz real de entrada de ordem N.

X = vetor de saída real contendo os autovalores de A (quando estes forem reais).

Y = matriz real de saída contendo os autovetores de A (no caso em que a matriz A é diagonalizável).

DINV = matriz real de saída contendo Λ^{-1}.

ATIL = matriz real de saída contendo a matriz C.

IDI = variável inteira de saída contendo a informação sobre a possibilidade da diagonalização.

Se IDI = 0 a matriz é diagonalizável.

Se IDI = 1 a matriz não é diagonalizável.

Subrotinas requeridas: FADEEV, SIM, MULMA, TRED2, TQL2, RG, AINV.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] WILKINSON, J.H. & REINSCH, C. Handbook for Automatic Computation Linear Algebra. Part 2. Berlin, New York, Springer-Verlag,1971.

[2] WILKINSON, J.H. The Algebraic Eingevalue Problem.Oxford,Claredon Press, 1965.

[3] SMITH, B.T., BOYLE, J.M., DANGORRA, J.J., GARBOW, B.S., IKEBE Y., KLEMA, V.C., MOLER, C.B. Matrix EigensystemRoutines: EISPACK GUIDE. 2nd ed. Lecture Notes in Computer Science 6, Springer-Verlag, New York, 1976

[4] WILKINSON, J.H. Householder's method for symetric matrices.Numer. Math 4, 354-361, 1962.

[5] FRANCIS, J.G.F. The QR Transformation. Parts I and II.Comput. Jour, 4, 265-271, 332-345, 1961, 1962.

[6] KUBLANOSKAYA, V.N. On Some Algorithms for the Solution of the Complete Eingevalue Problem. Z. Vycisl. Mat. Fiz 1,555-570 1961.

[7] BATHE, K.J. & WILSON, E.L. Numerical Methods in Finite Element Analysis Englewood Cliffs. Prentice-Hall, 1976.

[8] FADDEEV, D.K., FADDEEVA, V.N., Computacional Methods of Linear Algebra, by W.H. Freeman and Company, 1963.

[9] BARROS, I.Q. Métodos Numéricos de Álgebra Linear. Notas de aula IMEUSP, Março 1970.

APÊNDICE

```
C     PROGRAMA AUTOGE
C     ---------------------------------------------
C     DETERMINACAO DE TODOS AUTOVALORES E AUTOVETORES
C     PROBLEMA GENERALIZADO
C     ---------------------------------------------
C
C     DEPARTAMENTO DE ESTRUTURAS E FUNDACOES
C        ESCOLA POLITECNICA-USP
C
C          PRISCILA GOLDENBERG
C          REYOLANDO BRASIL
C          MARCIA CIMERMAN
C
C               ---- ABRIL 1988----
C
C              VERSAO 1.1
C     ---------------------------------------------
      REAL  A(10,10),B(10,10),WR(10),WI(10),Z(10,10),ZT(10,10)
      DIMENSION  FV1(10),IV1(10),Y(10,10),ATIL(10,10),Q(10,10)
      DIMENSION AM1(10,10),BM1(10,10),AA(10,10),BB(10,10)
      DIMENSION AL(10,10),BL(10,10)
       OPEN(5,FILE='PRN')
      NM = 10
   10 WRITE(5,*)'    PROGRAMA AUTOGE'
      WRITE(5,*)'DETERMINACAO AUTOVALORES AUTOVETORES'
      WRITE(5,*)' PROBLEMA GENERALIZADO'
      WRITE(5,*)'-------------------------------------'
      WRITE(5,*)'  DEP. ESTRUTURAS E FUNDACOES-EPUSP'
      WRITE(5,*)'     PRISCILA GOLDENBERG'
      WRITE(5,*)'     REYOLANDO BRASIL'
      WRITE(5,*)'     MARCIA CIMERMAN'
      WRITE(5,*)'-------------------------------------'
      WRITE(*,*)'ENTRE ORDEM DA(S) MATRIZ(ES) N='
      READ(*,*) N
      IF (N .LE. 0) STOP
      IF (N .GT. 10) STOP
      WRITE(*,*)'ENTRE INFORMACOES SOBRE A MATRIZ A'
      WRITE(*,*)'-SE FOR MATRIZ IDENTIDADE ENTRE 0'
      WRITE(*,*)'-SE NAO ENTRE  1'
      WRITE(*,*)'-SE NAO SABE ENTRE 2 QUE VAMOS VERIFICAR'
      READ(*,*) IDENTA
      IF (IDENTA .EQ. 0) GO TO 250
      WRITE(*,*)'-SE FOR SIMETRICA ENTRE  0'
      WRITE(*,*)'-SE NAO ENTRE  1'
      WRITE(*,*)'-SE NAO SABE ENTRE 2 QUE VAMOS VERIFICAR'
      READ(*,*) ISIA
      IF (ISIA .NE. 0) GO TO 120
      WRITE(*,*)'ENTRE MATRIZ SIMETR. A POR LINHA A PARTIR DIAGONAL'
          READ(*,*)((A(I,J),J=I,N),I=1,N)
      DO  110  I = 1, N
          DO  110  J = 1, N
  110        A(J,I) = A(I,J)
      GO TO 145
  120 WRITE(*,*)'ENTRE MATRIZ A NAO SIMETR. LINHA A LINHA'
         READ(*,*)((A(I,J),J=1,N),I=1,N)
  145 WRITE(5,150) N
  150 FORMAT(///21H ORDEM DAS MATRIZES E,I4//22H ELEMENTOS DA MATRIZ A)
          WRITE(5,160)((A(I,J),J=1,N),I=1,N)
  160     FORMAT(1X,1P4E14.6)
  190 WRITE(*,*)'QUER MUDAR ALGUM ELEMENTO DA MATRIZ A?(1=SIM,0=NAO)'
      READ(*,*) IND
      IF (IND .EQ. 0) GO TO 275
      WRITE(*,*)'ENTRE I,J DO ELEMENTO A MUDAR'
      READ(*,*) I,J
      WRITE(*,240) I,J
  240 FORMAT(//13H ENTRE NOVO A,I2,I2)
      READ(*,*) A(I,J)
      GO TO 190
  250 DO 270 I=1, N
      DO 260 J=1, N
      A(I,J)=0.0
  260 CONTINUE
      A(I,I)=1.0
  270 CONTINUE
      ISIA=0
  275 WRITE(*,*)'ENTRE INFORMACOES SOBRE MATRIZ B :'
      WRITE(*,*)'-SE FOR MATRIZ IDENTIDADE ENTRE  0'
      WRITE(*,*)'-SE NAO ENTRE  1'
      WRITE(*,*)'-SE NAO SABE ENTRE 2 QUE VAMOS VERIFICAR'
      READ(*,*) IDENTB
      IF (IDENTB .NE. 0) GO TO 300
      DO 290 I=1, N
      DO 280 J=1, N
      B(I,J)=0.0
  280 CONTINUE
      B(I,I)=1.0
  290 CONTINUE
      ISIB=0
      GO TO 460
  300 WRITE(*,*)'-SE FOR SIMETRICA ENTRE 0'
      WRITE(*,*)'-SE NAO ENTRE 1'
      WRITE(*,*)'-SE NAO SABE ENTRE 2 QUE VAMOS VERIFICAR'
      READ(*,*) ISIB
      IF (ISIB .NE. 0) GO TO 330
      WRITE(*,*)'ENTRE MATRIZ SIMETR. B POR LINHA A PARTIR DIAGONAL'
          READ(*,*)((B(I,J),J=I,N),I=1,N)
```

```
      DO  320  I = 1, N
          DO  320  J = 1, N
  320         B(J,I) = B(I,J)
      GO TO 360
  330     WRITE(*,*)'ENTRE MATRIZ NAO SIMETR. B LINHA A LINHA'
          READ(*,*)((B(I,J),J=1,N),I=1,N)
  360 WRITE(5,361)
  361 FORMAT(///22H ELEMENTOS DA MATRIZ B)
          WRITE(5,370)((B(I,J),J=1,N),I=1,N)
  370     FORMAT(1X,1P4E14.6)
  400 WRITE(*,*)'QUER MUDAR ALGUM ELEMENTO DA MATRIZ B?(1=SIM,0=NAO)'
      READ(*,*) IND
      IF (IND .EQ. 0) GO TO 460
      WRITE(*,*)'ENTRE I,J DO ELEMENTO A MUDAR'
      READ(*,*) I,J
      WRITE(*,450) I,J
  450 FORMAT(/13H ENTRE NOVO B,I2,I2)
      READ(*,*) B(I,J)
      GO TO 400
  460 WRITE(5,*)'DADOS COMPLETOS*****PROCESSANDO'
      DO 2000 I=1,N
      DO 2000 J=1,N
      AA(I,J)=A(I,J)
 2000 BB(I,J)=B(I,J)
      IF(IDENTA .LT. 2) GOTO 480
      CALL DENT(N,A,IDENTA)
  480 IF(ISIA .LT. 2) GOTO 490
      CALL SIM(N,A,ISIA)
  490 IF(IDENTB .LT. 2) GOTO 500
      CALL DENT(N,B,IDENTB)
  500 IF(ISIB .LT. 2) GO TO 510
      CALL SIM(N,B,ISIB)
  510 CALL FADEEV(N,A,DETA)
      IF(DETA .EQ. 0.0) GO TO 600
      CALL FADEEV(N,B,DETB)
      IF(DETB .EQ. 0.0) GO TO 700
      IF(IDENTB .EQ. 0) GO TO 1000
      IF(IDENTA .EQ. 0) GO TO 1100
      IF(ISIB .EQ. 0) GO TO 1200
  520 IF(ISIA .EQ. 0) GO TO 1300
      GO TO 8000
  600 WRITE(*,*)' MATRIZ A E SINGULAR'
      GO TO 10
  700 WRITE(*,*)' MATRIZ B E SINGULAR'
      GO TO 10
 1000 IF(ISIA .EQ. 0) GO TO 1010
      CALL RG(NM,N,AA,WR,WI,1,ZT,IV1,FV1,IERR)
       GO TO 8300
 1010 CALL TRED2(NM,N,AA,WR,FV1,ZT)
      CALL TQL2(NM,N,WR,FV1,ZT,IERR)
      DO 1020 I = 1, N
 1020 WI(I) = 0
      GOTO 8300
 1100 IF(ISIB .EQ. 0) GOTO 1110
      CALL RG(NM,N,BB,WR,WI,1,ZT,IV1,FV1,IERR)
 1001 DO 1004 I=1,N
      SQ1=WR(I)*WR(I)+WI(I)*WI(I)
      WR(I)=WR(I)/SQ1
 1004 WI(I)=-WI(I)/SQ1
      GO TO 8300
 1110 CALL TRED2(NM,N,BB,WR,FV1,ZT)
      CALL TQL2(NM,N,WR,FV1,ZT,IERR)
      DO 1120 I = 1, N
      WR(I) = 1./WR(I)
 1120 WI(I) = 0.
      GO TO 8300
 1200 CALL DEFPO(N,B,IPO)
      IF(IPO .NE. 0) GO TO 520
      IF(ISIA .NE. 0) GO TO 1230
      CALL REDUC1(N,AA,BB)
      CALL TRED2(NM,N,AA,WR,FV1,ZT)
      CALL TQL2(NM,N,WR,FV1,ZT,IERR)
      CALL REBAKA(N,BB,ZT)
      DO 1220 I = 1, N
 1220 WI(I)=0.0
      GO TO 8300
 1230 CALL REDUC1(N,AA,BB)
      CALL RG(NM,N,AA,WR,WI,1,ZT,IV1,FV1,IERR)
      CALL REBAKA(N,BB,ZT)
      GO TO 8300
 1300 CALL DEFPO(N,A,IPO)
      IF(IPO .NE. 0) GO TO 8000
      IF(ISIB .NE. 0) GO TO 1330
      CALL REDUC1(N,BB,AA)
      CALL TRED2(NM,N,BB,WR,FV1,ZT)
      CALL TQL2(NM,N,WR,FV1,ZT,IERR)
      CALL REBAKA(N,AA,ZT)
      DO 1320 I=1, N
      WR(I)=1./WR(I)
 1320 WI(I)=0.0
      GO TO 8300
 1330 CALL REDUC1(N,BB,AA)
      CALL RG(NM,N,BB,WR,WI,1,ZT,IV1,FV1,IERR)
      CALL REBAKA(N,AA,ZT)
      GO TO 1001
 8000 DO 2200 I=1,N
      DO 2200 J=1,N
```

```
      AL(I,J)=A(I,J)
 2200 BL(I,J)=B(I,J)
         CALL DIAGO(N,BL,AL,ISIB,Y,ATIL,IDI)
      IF(IDI)8401,8501,8401
 8501 NM=10
      CALL RG(NM,N,ATIL,WR,WI,1,Q,IV1,FV1,IERR)
      IF(IERR.NE.0)GOTO 8888
      GO TO 8889
 8888 WRITE(*,8500) IERR
 8500 FORMAT(1X,'NAO CONVERGIU O AUTOVALOR',I4)
      GO TO 8601
 8889 CALL MULMA(N,N,Y,Q,ZT)
 8300  WRITE(5,*)'****SAIDA DOS AUTOVALORES E AUTOVETORES***'
      I=1
 8330 CONTINUE
        WRITE(5,8320) WR(I),WI(I)
 8320   FORMAT(1X,'AUTOVALOR=',E14.8,'+i',E14.8,/)
       IF(WI(1))8321,8322,8321
 8321 WRITE(5,8323)(ZT(J,I),J=1,N)
 8323 FORMAT(10X,'PARTE REAL DO AUTOVETOR',/,(1X,E14.8))
      WRITE(5,8324)(ZT(J,I+1),J=1,N)
      WRITE(5,*)'------------------------------------'
 8324  FORMAT(10X,'PARTE IMAGINARIA DO AUTOVETOR',/,(1X,E14.8))
      DO 8329 KI=1,N
 8329     ZT(KI,I+1)=-ZT(KI,I+1)
      WI(I)=-WI(I)
       WRITE(5,8320) WR(I),WI(I)
       WRITE(5,8323)(ZT(J,I),J=1,N)
       WRITE(5,8324)(ZT(J,I+1),J=1,N)
       WRITE(5,*)'---------- ----------'
        I=I+2
      IF(I-N)8330,8330,8325
 8322 WRITE(5,8326)(ZT(J,I),J=1,N)
      WRITE(5,*)'------------------------------------'
 8326 FORMAT(1X,E14.8)
      I=I+1
      IF(I-N)8330,8330,8325
 8325 CONTINUE
      WRITE(5,*)'*****FIM DO PROBLEMA*****'
      GO TO 8601
 8401 DO 2009 I=1,N
      DO 2009 J=1,N
      AL(I,J)=A(I,J)
 2009  BL(I,J)=B(I,J)
      CALL DIAGO(N,AL,BL,ISIA,Y,ATIL,IDI)
      IF(IDI)8402,8502,8402
 8502 MATZ=1
      NM=10
      CALL RG(NM,N,ATIL,WR,WI,MATZ,Q,IV1,FV1,IERR)
      IF(IERR.NE.0)GO TO 8888
      DO 8887 I=1,N
      SQ=WR(I)**2.+WI(I)**2.
      WR(I)=WR(I)/SQ
 8887 WI(I)=-WI(I)/SQ
      GO TO 8889
 8402 CONTINUE
      CALL COND(N,B,RB)
      CALL COND(N,A,RA)
      IF(RA-RB)8700,8701,8701
 8700 CALL AINV(N,A,AM1)
       CALL MULMA(N,N,AM1,B,ATIL)
      GO TO 8502
 8701 CALL AINV(N,B,BM1)
      CALL MULMA(N,N,BM1,A,ATIL)
      GO TO 8501
 8601 CONTINUE
      GO TO 10
      END
      SUBROUTINE AINV(N,A,AM1)
      DIMENSION A(10,10),AM1(10,10)
      REAL AUX
      INTEGER I,J,K
      DO 1 KI=1,N
      DO 1 KJ=1,N
 1    AM1(KI,KJ)=A(KI,KJ)
      DO 50 K=1,N
      AUX=AM1(K,K)
      AM1(K,K)=1.0
       DO 20 J=1,N
 20   AM1(K,J)=AM1(K,J)/AUX
      DO 40 I=1,N
      IF(I.EQ.K)GO TO 40
      AUX=AM1(I,K)
      AM1(I,K)=0.
      DO 30 J=1,N
 30   AM1(I,J)=AM1(I,J)-AUX*AM1(K,J)
 40   CONTINUE
 50   CONTINUE
       RETURN
      END
      SUBROUTINE MULMA(NA,NB,A,B,C)
      REAL A(10,10),B(10,10),C(10,10)
      DO 1 I=1,NA
      DO 1 J=1,NB
 1    C(I,J)=0.
      DO 2 I=1,NA
      DO 2 J=1,NB
```

```
      DO 2 K=1,NB
    2 C(I,J)=C(I,J)+A(I,K)*B(K,J)
      RETURN
      END
      SUBROUTINE COND(N,A,RO)
      DIMENSION A(10,10),DB(10),AM(10,10)
      DO 1 I=1,N
    1 DB(I)=0
      DO 2 I=1,N
      DO 2 J=1,N
    2 DB(I)=DB(I)+ABS(A(I,J))
      RO1=DB(1)
      DO 3 I=2,N
      IF(DB(I)-RO1)3,3,4
    4 RO1=DB(I)
    3 CONTINUE
      CALL AINV(N,A,AM)
      DO 5 I=1,N
    5 DB(I)=0
      DO 6 I=1,N
      DO 6 J=1,N
    6 DB(I)=DB(I)+ABS(AM(I,J))
      RO2=DB(1)
      DO 7 I=2,N
      IF(DB(I)-RO2)7,7,8
    8 RO2=DB(I)
    7 CONTINUE
      RO=RO1*RO2
      WRITE(5,*)RO
      RETURN
      END
      SUBROUTINE SIM(NA,A,ISI)
      REAL A(10,10),DDB(10,10)
      ISI=0
      DO 10 I=1,NA
      DO 10 J=1,NA
      DDB(I,J)=A(J,I)
      IF(A(I,J).EQ.DDB(I,J)) GO TO 10
      GO TO 20
   10 CONTINUE
      GO TO 100
   20 ISI=1
  100 CONTINUE
      RETURN
      END
      SUBROUTINE DIAGO(N,B,A,IS,Y,ATIL,IDI)
      DIMENSION A(10,10),X(10),Y(10,10),DINV(10,10),ATIL(10,10)
      DIMENSION YT(10,10),V1(10),V2(10),WR(10),WI(10)
      DIMENSION TEMP1(10,10),TEMP2(10,10),IV1(10),B(10,10)
      INTEGER I,J,N,NM
      NM=10
      IDI=0
      IF(IS)1,2,1
    2 CALL TRED2(NM,N,B,X,V2,Y)
      CALL TQL2(NM,N,X,V2,Y,IERR)
      IF(IERR.NE.0)GO TO 10
      DO 100 J=1,N
      XMOD=X(J)
      DO 100 I=J+1,N
      IF(X(I)-XMOD)100,10,100
  100 CONTINUE
      DO 4 I=1,N
      DO 4 J=1,N
    4 DINV(I,J)=0.
      DO 5 I=1,N
    5 DINV(I,I)=1./X(I)
   12 DO 6 I=1,N
      DO 6 J=1,N
    6 YT(I,J)=Y(J,I)
      CALL MULMA(N,N,DINV,YT,TEMP1)
      CALL MULMA(N,N,TEMP1,A,TEMP2)
      CALL MULMA(N,N,TEMP2,Y,ATIL)
      GO TO 20
    1 MATZ=1
      CALL RG(NM,N,B,WR,WI,MATZ,Y,IV1,V1,IERR)
      IF(IERR.NE.0)GO TO 10
      DO 7 I=1,N
      IF(WR(I)) 10,7,10
    7 CONTINUE
      DO 101 J=1,N
      XMOD=WR(J)
      DO 101 I=J+1,N
      IF(WR(I)-XMOD)101,10,101
  101 CONTINUE
      DO 9 I=1,N
      DO 9 J=1,N
    9 DINV(I,J)=0
      DO 11 I=1,N
   11 DINV(I,I)=1./WR(I)
      GO TO 12
   10 IDI=1
   20 CONTINUE
      RETURN
      END
      SUBROUTINE FADEEV(N,A,DETA)
      REAL ADENT(10,10),S(10,10),COEFA(10),SA(10,10),A(10,10)
      DO 3 I=1,N
      DO 3 J=1,N
    3 ADENT(I,J)=0.
```

```
      DO 4 I=1,N
    4 ADENT(I,I)=1.
      DO 5 I=1,N
      DO 5 J=1,N
    5 S(I,J)=ADENT(I,J)
      DO 6 JJ=1,N
      SAUX=0.
      CALL MULMA(N,N,S,A,SA)
      DO 7 I=1,N
    7 SAUX=SAUX-SA(I,I)
      COEFA(JJ)=SAUX/FLOAT(JJ)
      DO 8 I=1,N
      DO 8 J=1,N
    8 S(I,J)=ADENT(I,J)*COEFA(JJ)
      DO 9 I=1,N
      DO 9 J=1,N
    9 S(I,J)=SA(I,J)+S(I,J)
    6 CONTINUE
      DETA=(-1.)**N*COEFA(N)
      RETURN
      END
      SUBROUTINE DENT(NA,A,IDENT)
      REAL A(20,20)
      IDENT=0
      DO 30 I=1,NA
      IF(A(I,I).EQ.1) GO TO 30
      GO TO 20
   30 CONTINUE
      DO 40 I=1,NA
      DO 40 J=I+1,NA
      IF(A(I,J).EQ.0) GO TO 40
      GO TO 20
   40 CONTINUE
      DO 50 J=1,NA
      DO 50 I=J+1,NA
      IF(A(I,J).EQ.0) GO TO 50
      GO TO 20
   50 CONTINUE
      GO TO 100
   20 IDENT=1
  100 CONTINUE
      RETURN
      END
      SUBROUTINE DEFPO(NA,A,IPO)
      REAL A(10,10),BD(10,10)
      IPO=0
      DO 10 M=1,NA
      DO 15 ID=1,M
      DO 15 JD=1,M
      BD(ID,JD)=A(ID,JD)
   15 CONTINUE
      CALL FADEEV(M,BD,DETB)
      IF (DETB)20,20,10
   10 CONTINUE
      GO TO 30
   20 IPO=1
   30 CONTINUE
      RETURN
      END
      SUBROUTINE REDUC1(N,A,B)
      INTEGER I,J,K,IM1,JM1,IP1,N
      REAL A(10,10),B(10,10),AL(10,10),ALI(10,10)
      REAL SI,SJ
      REAL SQRT
      DO 10 I=1, N
      DO 10 J=1, N
      AL(I,J)=0.0
   10 ALI(I,J)=0.0
      DO 70 I=1, N
      SI=0.0
      IF(I .EQ. 1) GO TO 60
      IM1=I-1
      DO 40 J=1, IM1
      SJ=0.0
      IF(J .EQ. 1) GO TO 30
      JM1=J-1
      DO 20 K=1, JM1
   20 SJ=SJ+AL(I,K)*AL(J,K)
   30 AL(I,J)=(B(I,J)-SJ)/AL(J,J)
   40 CONTINUE
      DO 50 K=1, IM1
   50 SI=SI+AL(I,K)*AL(I,K)
   60 AL(I,I)=SQRT(B(I,I)-SI)
   70 CONTINUE
      DO 90 I=1, N
      ALI(I,I)=1./AL(I,I)
      IF(I .EQ. N) GO TO 90
      IP1=I+1
      DO 85 J=IP1, N
      SJ=0.0
      JM1=J-1
      DO 80 K=I, JM1
   80 SJ=SJ-AL(J,K)*ALI(K,I)
      ALI(J,I)=SJ/AL(J,J)
   85 CONTINUE
   90 CONTINUE
      DO 100 I=1, N
      DO 100 J=1, N
  100 B(I,J)=ALI(J,I)
```

```
      CALL MULMA(N,N,ALI,A,AL)
      CALL MULMA(N,N,AL,B,A)
      RETURN
      END
      SUBROUTINE REBAKA(N,B,Z)
      INTEGER N,I,J
      REAL B(10,10),Z(10,10),ZI(10,10)
      DO 10 I=1, N
      DO 10 J=1, N
   10 ZI(I,J)=Z(I,J)
      CALL MULMA(N,N,B,ZI,Z)
      RETURN
      END
      SUBROUTINE  TRED2(NM,N,A,D,E,Z)
      INTEGER  I,J,K,L,N,II,NM,JP1
      REAL  A(NM,N),D(N),E(N),Z(NM,N)
      REAL  F,G,H,HH,SCALE
      REAL  SQRT,ABS,SIGN

      DO  100  I = 1, N
          DO  100  J = 1, I
              Z(I,J) = A(I,J)
  100 CONTINUE
      IF  (N .EQ. 1)  GO  TO  320
      DO  300  II = 2, N
          I = N + 2 - II
          L = I - 1
          H = 0.0
          SCALE = 0.0
          IF  (L .LT. 2)  GO  TO  130
          DO  120  K = 1, L
  120     SCALE = SCALE + ABS(Z(I,K))
          IF  (SCALE .NE. 0.0)  GO  TO  140
  130     E(I) = Z(I,L)
          GO  TO  290
  140     DO  150  K = 1, L
              Z(I,K) = Z(I,K) / SCALE
              H = H + Z(I,K) * Z(I,K)
  150     CONTINUE
          F = Z(I,L)
          G = -SIGN(SQRT(H),F)
          E(I) = SCALE * G
          H = H - F * G
          Z(I,L) = F - G
          F = 0.0
          DO  240  J = 1, L
              Z(J,I) = Z(I,J) / H
              G = 0.0
              DO  180  K = 1, J
  180         G = G + Z(J,K) * Z(I,K)
              JP1 = J + 1
              IF  (L .LT. JP1)  GO  TO  220
              DO  200  K = JP1, L
  200         G = G + Z(K,J) * Z(I,K)
  220         E(J) = G / H
              F = F + E(J) * Z(I,J)
  240     CONTINUE
          HH = F / (H + H)
          DO  260  J = 1, L
              F = Z(I,J)
              G = E(J) - HH * F
              E(J) = G
              DO  260  K = 1, J
                  Z(J,K) = Z(J,K) - F * E(K) - G * Z(I,K)
  260     CONTINUE
  290     D(I) = H
  300 CONTINUE
  320 D(1) = 0.0
      E(1) = 0.0
      DO  500  I = 1, N
          L = I - 1
          IF  (D(I) .EQ. 0.0)  GO  TO  380
          DO  360  J = 1, L
              G = 0.0
              DO  340  K = 1, L
  340         G = G + Z(I,K) * Z(K,J)
              DO  360  K = 1, L
                  Z(K,J) = Z(K,J) - G * Z(K,I)
  360     CONTINUE
  380     D(I) = Z(I,I)
          Z(I,I) = 1.0
          IF  (L .LT. 1)  GO  TO  500
          DO  400  J = 1, L
              Z(I,J) = 0.0
              Z(J,I) = 0.0
  400     CONTINUE
  500 CONTINUE
      RETURN
      END
      SUBROUTINE  TQL2(NM,N,D,E,Z,IERR)
      INTEGER  I,J,K,L,M,N,II,L1,NM,MML,IERR
      REAL  D(N),E(N),Z(NM,N)
      REAL  B,C,F,G,H,P,R,S,MACHEP
      REAL  SQRT,ABS,SIGN
      MACHEP = 2.**(-26)
      IERR = 0
      IF  (N .EQ. 1)  GO  TO  1001
      DO  100  I = 2, N
  100 E(I-1) = E(I)
```

```
      F = 0.0
      B = 0.0
      E(N) = 0.0
      DO  240  L = 1, N
          J = 0
          H = MACHEP * (ABS(D(L)) + ABS(E(L)))
          IF  (B .LT. H)  B = H
          DO  110  M = L, N
              IF  (ABS(E(M)) .LE. B)  GO  TO  120
  110     CONTINUE
  120     IF  (M .EQ. L)  GO  TO  220
  130     IF  (J .EQ. 30)  GO  TO  1000
          J = J + 1
          L1 = L + 1
          G = D(L)
          P = (D(L1) - G) / (2.0 * E(L))
          R = SQRT(P*P+1.0)
          D(L) = E(L) / (P + SIGN(R,P))
          H = G - D(L)
          DO  140  I = L1, N
  140     D(I) = D(I) - H
          F = F + H
          P = D(M)
          C = 1.0
          S = 0.0
          MML = M - L
          DO  200  II = 1, MML
              I = M - II
              G = C * E(I)
              H = C * P
              IF  (ABS(P) .LT. ABS(E(I)))  GO  TO  150
              C = E(I) / P
              R = SQRT(C*C+1.0)
              E(I+1) = S * P * R
              S = C / R
              C = 1.0 / R
              GO  TO  160
  150         C = P / E(I)
              R = SQRT(C*C+1.0)
              E(I+1) = S * E(I) * R
              S = 1.0 / R
              C = C * S
  160         P = C * D(I) - S * G
              D(I+1) = H + S * (C * G + S * D(I))
              DO  180  K = 1, N
                  H = Z(K,I+1)
                  Z(K,I+1) = S * Z(K,I) + C * H
                  Z(K,I) = C * Z(K,I) - S * H
  180         CONTINUE
  200     CONTINUE
          E(L) = S * P
          D(L) = C * P
          IF  (ABS(E(L)) .GT. B)  GO  TO  130
  220     D(L) = D(L) + F
  240 CONTINUE
      DO  300  II = 2, N
          I = II - 1
          K = I
          P = D(I)
          DO  260  J = II, N
              IF  (D(J) .GE. P)  GO  TO  260
              K = J
              P = D(J)
  260     CONTINUE
          IF  (K .EQ. I)  GO  TO  300
          D(K) = D(I)
          D(I) = P
          DO  280  J = 1, N
              P = Z(J,I)
              Z(J,I) = Z(J,K)
              Z(J,K) = P
  280     CONTINUE
  300 CONTINUE
      GO  TO  1001
 1000 IERR = L
 1001 RETURN
      END
      SUBROUTINE  RG(NM,N,A,WR,WI,MATZ,Z,IV1,FV1,IERR)
      INTEGER  N,NM,IS1,IS2,IERR,MATZ
      REAL  A(NM,N),WR(N),WI(N),Z(NM,N),FV1(N)
      INTEGER  IV1(N)
      IF  (N .LE. NM) GO TO 10
      IERR = 10 * N
      GO  TO  50
   10 IS1=1
      IS2=N
      CALL  ELMHES(NM,N,IS1,IS2,A,IV1)
      CALL  ELTRAN(NM,N,IS1,IS2,A,IV1,Z)
      CALL  HQR2(NM,N,IS1,IS2,A,WR,WI,Z,IERR)
   50 RETURN
      END
      SUBROUTINE  ELMHES(NM,N,LOW,IGH,A,INT)
      INTEGER  I,J,M,N,LA,NM,IGH,KP1,LOW,MM1,MP1
      REAL  A(NM,N)
      REAL  X,Y
      REAL  ABS
      INTEGER  INT(IGH)
      LA = IGH - 1
      KP1 = LOW + 1
      IF (LA .LT. KP1) GO  TO  200
```

```
      X = 0.0
      I = M
      DO  100  J = M, IGH
          IF (ABS(A(J,MM1)) .LE. ABS(X)) GO  TO  100
          X = A(J,MM1)
          I = J
100   CONTINUE
      INT(M) = I
      IF (I .EQ. M) GO  TO  130
      DO  110  J = MM1, N
          Y = A(I,J)
          A(I,J) = A(M,J)
          A(M,J) = Y
110   CONTINUE
      DO  120  J = 1, IGH
          Y = A(J,I)
          A(J,I) = A(J,M)
          A(J,M) = Y
120   CONTINUE
130   IF (X .EQ. 0.0) GO  TO  180
      MP1 = M + 1
      DO  160  I = MP1, IGH
          Y = A(I,MM1)
          IF (Y .EQ. 0.0) GO  TO  160
          Y = Y / X
          A(I,MM1) = Y
          DO  140  J = M, N
140       A(I,J) = A(I,J) - Y * A(M,J)
          DO  150  J = 1, IGH
150       A(J,M) = A(J,M) + Y * A(J,I)
160   CONTINUE
180 CONTINUE
200 RETURN
    END
    SUBROUTINE  ELTRAN(NM,N,LOW,IGH,A,INT,Z)
    INTEGER  I,J,N,KL,MM,MP,NM,IGH,LOW,MP1
    REAL  A(NM,IGH),Z(NM,N)
    INTEGER  INT(IGH)
    DO  80  I = 1, N
        DO  60  J = 1, N
60          Z(I,J) = 0.0
        Z(I,I) = 1.0
80  CONTINUE
    KL = IGH - LOW - 1
    IF (KL .LT. 1) GO  TO  200
    DO  140  MM = 1, KL
        MP = IGH - MM
        MP1 = MP + 1
        DO  100  I = MP1, IGH
100         Z(I,MP) = A(I,MP-1)
        I = INT(MP)
        IF (I .EQ. MP) GO  TO  140
        DO  130  J = MP, IGH
            Z(MP,J) = Z(I,J)
            Z(I,J) = 0.0
130     CONTINUE
        Z(I,MP) = 1.0
140 CONTINUE
200 RETURN
    END
    SUBROUTINE  HQR2(NM,N,LOW,IGH,H,WR,WI,Z,IERR)
    INTEGER  I,J,K,L,M,N,EN,II,JJ,LL,MM,NA,NM,NN,
   X         IGH,ITS,LOW,MP2,ENM2,IERR
    REAL  H(NM,N),WR(N),WI(N),Z(NM,N)
    REAL  P,Q,R,S,T,W,X,Y,RA,SA,VI,VR,ZZ,NORM,MACHEP
    REAL  SQRT,ABS,SIGN
    INTEGER MIN0
    LOGICAL  NOTLAS
    COMPLEX Z3
    COMPLEX CMPLX
    REAL  REAL,AIMAG
    MACHEP = 2.**(-26)
    IERR = 0
    NORM = 0.0
    K = 1
    DO 50 I = 1, N
        DO 40 J = K, N
40          NORM = NORM + ABS(H(I,J))
        K = I
        IF (I .GE. LOW .AND. I .LE. IGH) GO TO 50
        WR(I) = H(I,I)
        WI(I) = 0.0
50  CONTINUE
    EN = IGH
    T = 0.0
60  IF (EN .LT. LOW) GO TO 340
    ITS = 0
    NA = EN - 1
    ENM2 = NA - 1
70  DO 80 LL = LOW, EN
        L = EN + LOW - LL
        IF (L .EQ. LOW) GO TO 100
        S = ABS(H(L-1,L-1)) + ABS(H(L,L))
        IF (S .EQ. 0.0) S = NORM
        IF (ABS(H(L,L-1)) .LE. MACHEP * S) GO TO 100
80  CONTINUE
100 X = H(EN,EN)
    IF (L .EQ. EN) GO TO 270
    Y = H(NA,NA)
```

```
      W = H(EN,NA) * H(NA,EN)
      IF (L .EQ. NA) GO TO 280
      IF (ITS .EQ. 30) GO TO 1000
      IF (ITS .NE. 10 .AND. ITS .NE. 20) GO TO 130
      T = T + X
      DO 120 I = LOW, EN
  120    H(I,I) = H(I,I) - X
      S = ABS(H(EN,NA)) + ABS(H(NA,ENM2))
      X = 0.75 * S
      Y = X
      W = -0.4375 * S * S
  130 ITS = ITS + 1
      DO 140 MM = L, ENM2
         M = ENM2 + L - MM
         ZZ = H(M,M)
         R = X - ZZ
         S = Y - ZZ
         P = (R * S - W) / H(M+1,M) + H(M,M+1)
         Q = H(M+1,M+1) - ZZ -R - S
         R = H(M+2,M+1)
         S = ABS(P) + ABS(Q) + ABS(R)
         P = P / S
         Q = Q / S
         R = R / S
         IF (M .EQ. L) GO TO 150
         IF (ABS(H(M,M-1)) * (ABS(Q) + ABS(R)) .LE. MACHEP * ABS(P)
     X    * (ABS(H(M-1,M-1)) + ABS(ZZ) + ABS(H(M+1,M+1)))) GO TO 150
  140 CONTINUE
  150 MP2 = M + 2
      DO 160 I = MP2, EN
         H(I,I-2) = 0.0
         IF (I .EQ. MP2) GO TO 160
         H(I,I-3) = 0.0
  160 CONTINUE
      DO 260 K = M, NA
         NOTLAS = K .NE. NA
         IF (K .EQ. M) GO TO 170
         P = H(K,K-1)
         Q = H(K+1,K-1)
         R = 0.0
         IF (NOTLAS) R = H(K+2,K-1)
         X = ABS(P) + ABS(Q) + ABS(R)
         IF (X .EQ. 0.0) GO TO 260
         P = P / X
         Q = Q / X
         R = R / X
  170    S = SIGN(SQRT(P*P+Q*Q+R*R),P)
         IF (K .EQ. M) GO TO 180
         H(K,K-1) = -S * X
         GO TO 190
  180    IF (L .NE. M) H(K,K-1) = -H(K,K-1)
  190    P = P + S
         X = P / S
         Y = Q / S
         ZZ = R / S
         Q = Q / P
         R = R / P
         DO 210 J = K, N
            P = H(K,J) + Q * H(K+1,J)
            IF (.NOT. NOTLAS) GO TO 200
            P = P + R * H(K+2,J)
            H(K+2,J) = H(K+2,J) - P * ZZ
  200       H(K+1,J) = H(K+1,J) - P * Y
            H(K,J) = H(K,J) - P * X
  210    CONTINUE
         J = MIN0(EN,K+3)
         DO 230 I = 1, J
            P = X * H(I,K) + Y * H(I,K+1)
            IF (.NOT. NOTLAS) GO TO 220
            P = P + ZZ * H(I,K+2)
            H(I,K+2) = H(I,K+2) - P * R
  220       H(I,K+1) = H(I,K+1) - P * Q
            H(I,K) = H(I,K) - P
  230    CONTINUE
         DO 250 I = LOW, IGH
            P = X * Z(I,K) + Y * Z(I,K+1)
            IF (.NOT. NOTLAS) GO TO 240
            P = P + ZZ * Z(I,K+2)
            Z(I,K+2) = Z(I,K+2) - P * R
  240       Z(I,K+1) = Z(I,K+1) - P * Q
            Z(I,K) = Z(I,K) - P
  250    CONTINUE
  260 CONTINUE
      GO TO 70
  270 H(EN,EN) = X + T
      WR(EN) = H(EN,EN)
      WI(EN) = 0.0
      EN = NA
      GO TO 60
  280 P = (Y - X) / 2.0
      Q = P * P + W
      ZZ = SQRT(ABS(Q))
      H(EN,EN) = X + T
      X = H(EN,EN)
      H(NA,NA) = Y + T
      IF (Q .LT. 0.0) GO TO 320
      ZZ = P + SIGN(ZZ,P)
      WR(NA) = X + ZZ
      WR(EN) = WR(NA)
      IF (ZZ .NE. 0.0) WR(EN) = X - W / ZZ
```

```
      WI(NA) = 0.0
      WI(EN) = 0.0
      X = H(EN,NA)
      S = ABS(X) + ABS(ZZ)
      P = X / S
      Q = ZZ / S
      R = SQRT(P*P+Q*Q)
      P = P / R
      Q = Q / R
      DO 290 J = NA, N
         ZZ = H(NA,J)
         H(NA,J) = Q * ZZ + P * H(EN,J)
         H(EN,J) = Q * H(EN,J) - P * ZZ
  290 CONTINUE
      DO 300 I = 1, EN
         ZZ = H(I,NA)
         H(I,NA) = Q * ZZ + P * H(I,EN)
         H(I,EN) = Q * H(I,EN) - P * ZZ
  300 CONTINUE
      DO 310 I = LOW, IGH
         ZZ = Z(I,NA)
         Z(I,NA) = Q * ZZ + P * Z(I,EN)
         Z(I,EN) = Q * Z(I,EN) - P * ZZ
  310 CONTINUE
      GO TO 330
  320 WR(NA) = X + P
      WR(EN) = X + P
      WI(NA) = ZZ
      WI(EN) = -ZZ
  330 EN = ENM2
      GO TO 60
  340 IF (NORM .EQ. 0.0) GO TO 1001
      DO 800 NN = 1, N
         EN = N + 1 - NN
         P = WR(EN)
         Q = WI(EN)
         NA = EN - 1
         IF (Q) 710, 600, 800
  600    M = EN
         H(EN,EN) = 1.0
         IF (NA .EQ. 0) GO TO 800
         DO 700 II = 1, NA
            I = EN - II
            W = H(I,I) - P
            R = H(I,EN)
            IF (M .GT. NA) GO TO 620
            DO 610 J = M, NA
  610          R = R + H(I,J) * H(J,EN)
  620       IF (WI(I) .GE. 0.0) GO TO 630
            ZZ = W
            S = R
            GO TO 700
  630       M = I
            IF (WI(I) .NE. 0.0) GO TO 640
            T = W
            IF (W .EQ. 0.0) T = MACHEP * NORM
            H(I,EN) = -R / T
            GO TO 700
  640       X = H(I,I+1)
            Y = H(I+1,I)
            Q = (WR(I) - P) * (WR(I) - P) + WI(I) * WI(I)
            T = (X * S - ZZ * R) / Q
            H(I,EN) = T
            IF (ABS(X) .LE. ABS(ZZ)) GO TO 650
            H(I+1,EN) = (-R - W * T) / X
            GO TO 700
  650       H(I+1,EN) = (-S - Y * T) / ZZ
  700    CONTINUE
         GO TO 800
  710    M = NA
         IF (ABS(H(EN,NA)) .LE. ABS(H(NA,EN))) GO TO 720
         H(NA,NA) = Q / H(EN,NA)
         H(NA,EN) = -(H(EN,EN) - P) / H(EN,NA)
         GO TO 730
  720    Z3 = CMPLX(0.0,-H(NA,EN)) / CMPLX(H(NA,NA)-P,Q)
         H(NA,NA) = REAL(Z3)
         H(NA,EN) = AIMAG(Z3)
  730    H(EN,NA) = 0.0
         H(EN,EN) = 1.0
         ENM2 = NA - 1
         IF (ENM2 .EQ. 0) GO TO 800
         DO 790 II = 1, ENM2
            I = NA - II
            W = H(I,I) - P
            RA = 0.0
            SA = H(I,EN)
            DO 760 J = M, NA
               RA = RA + H(I,J) * H(J,NA)
               SA = SA + H(I,J) * H(J,EN)
  760       CONTINUE
            IF (WI(I) .GE. 0.0) GO TO 770
            ZZ = W
            R = RA
            S = SA
            GO TO 790
  770       M = I
            IF (WI(I) .NE. 0.0) GO TO 780
            Z3 = CMPLX(-RA,-SA) / CMPLX(W,Q)
            H(I,NA) = REAL(Z3)
            H(I,EN) = AIMAG(Z3)
```

```
      GO TO 790
780   X = H(I,I+1)
      Y = H(I+1,I)
      VR = (WR(I) - P) * (WR(I) - P) + WI(I) * WI(I) - Q * Q
      VI = (WR(I) - P) * 2.0 * Q
      IF (VR .EQ. 0.0 .AND. VI .EQ. 0.0) VR = MACHEP * NORM
     X  * (ABS(W) + ABS(Q) + ABS(X) + ABS(Y) + ABS(ZZ))
      Z3 = CMPLX(X*R-ZZ*RA+Q*SA,X*S-ZZ*SA-Q*RA) / CMPLX(VR,VI)
      H(I,NA) = REAL(Z3)
      H(I,EN) = AIMAG(Z3)
      IF (ABS(X) .LE. ABS(ZZ) + ABS(Q)) GO TO 785
      H(I+1,NA) = (-RA - W * H(I,NA) + Q * H(I,EN)) / X
      H(I+1,EN) = (-SA - W * H(I,EN) - Q * H(I,NA)) / X
      GO TO 790
785   Z3 = CMPLX(-R-Y*H(I,NA),-S-Y*H(I,EN)) / CMPLX(ZZ,Q)
      H(I+1,NA) = REAL(Z3)
      H(I+1,EN) = AIMAG(Z3)
790   CONTINUE
800 CONTINUE
    DO 840 I = 1, N
       IF (I .GE. LOW .AND. I .LE. IGH) GO TO 840
       DO 820 J = I, N
820       Z(I,J) = H(I,J)
840 CONTINUE
    DO 880 JJ = LOW, N
       J = N + LOW - JJ
       M = MIN0(J,IGH)
       DO 880 I = LOW, IGH
          ZZ = 0.0
          DO 860 K = LOW, M
860          ZZ = ZZ + Z(I,K) * H(K,J)
          Z(I,J) = ZZ
880 CONTINUE
    GO TO 1001
1000 IERR = EN
1001 RETURN
    END
     SUBROUTINE  TRED2(NM,N,A,D,E,Z)
     INTEGER  I,J,K,L,N,II,NM,JP1
     REAL  A(NM,N),D(N),E(N),Z(NM,N)
     REAL  F,G,H,HH,SCALE
     REAL  SQRT,ABS,SIGN

     DO  100  I = 1, N
        DO  100  J = 1, I
           Z(I,J) = A(I,J)
 100 CONTINUE
     IF  (N .EQ. 1)  GO  TO  320
     DO  300  II = 2, N
        I = N + 2 - II
        L = I - 1
        H = 0.0
        SCALE = 0.0
        IF  (L .LT. 2)  GO  TO  130
        DO  120  K = 1, L
120     SCALE = SCALE + ABS(Z(I,K))
        IF  (SCALE .NE. 0.0)  GO  TO  140
130     E(I) = Z(I,L)
        GO  TO  290
140     DO  150  K = 1, L
           Z(I,K) = Z(I,K) / SCALE
           H = H + Z(I,K) * Z(I,K)
150     CONTINUE
        F = Z(I,L)
        G = -SIGN(SQRT(H),F)
        E(I) = SCALE * G
        H = H - F * G
        Z(I,L) = F - G
        F = 0.0
        DO  240  J = 1, L
           Z(J,I) = Z(I,J) / H
           G = 0.0
           DO  180  K = 1, J
180        G = G + Z(J,K) * Z(I,K)
           JP1 = J + 1
           IF  (L .LT. JP1)  GO  TO  220
           DO  200  K = JP1, L
200        G = G + Z(K,J) * Z(I,K)
220        E(J) = G / H
           F = F + E(J) * Z(I,J)
240     CONTINUE
        HH = F / (H + H)
        DO  260  J = 1, L
           F = Z(I,J)
           G = E(J) - HH * F
           E(J) = G
           DO  260  K = 1, J
              Z(J,K) = Z(J,K) - F * E(K) - G * Z(I,K)
260     CONTINUE
290     D(I) = H
300  CONTINUE
320  D(1) = 0.0
     E(1) = 0.0
     DO  500  I = 1, N
        L = I - 1
        IF  (D(I) .EQ. 0.0)  GO  TO  380
        DO  360  J = 1, L
           G = 0.0
           DO  340  K = 1, L
```

```
340         G = G + Z(I,K) * Z(K,J)
            DO  360  K = 1, L
               Z(K,J) = Z(K,J) - G * Z(K,I)
360      CONTINUE
380      D(I) = Z(I,I)
         Z(I,I) = 1.0
         IF  (L .LT. 1)  GO  TO  500
         DO  400  J = 1, L
            Z(I,J) = 0.0
            Z(J,I) = 0.0
400      CONTINUE
500 CONTINUE
    RETURN
    END
    SUBROUTINE  TQL2(NM,N,D,E,Z,IERR)
    INTEGER  I,J,K,L,M,N,II,L1,NM,MML,IERR
    REAL  D(N),E(N),Z(NM,N)
    REAL  B,C,F,G,H,P,R,S,MACHEP
    REAL  SQRT,ABS,SIGN
    MACHEP = 2.**(-26)
    IERR = 0
    IF  (N .EQ. 1)  GO  TO  1001
    DO  100  I = 2, N
100 E(I-1) = E(I)
    F = 0.0
    B = 0.0
    E(N) = 0.0
    DO  240  L = 1, N
        J = 0
        H = MACHEP * (ABS(D(L)) + ABS(E(L)))
        IF  (B .LT. H)  B = H
        DO  110  M = L, N
           IF  (ABS(E(M)) .LE. B)  GO  TO  120
110     CONTINUE
120     IF  (M .EQ. L)  GO  TO  220
130     IF  (J .EQ. 30)  GO  TO  1000
        J = J + 1
        L1 = L + 1
        G = D(L)
        P = (D(L1) - G) / (2.0 * E(L))
        R = SQRT(P*P+1.0)
        D(L) = E(L) / (P + SIGN(R,P))
        H = G - D(L)
        DO  140  I = L1, N
140     D(I) = D(I) - H
        F = F + H
        P = D(M)
        C = 1.0
        S = 0.0
        MML = M - L
        DO  200  II = 1, MML
           I = M - II
           G = C * E(I)
           H = C * P
           IF  (ABS(P) .LT. ABS(E(I)))  GO  TO  150
           C = E(I) / P
           R = SQRT(C*C+1.0)
           E(I+1) = S * P * R
           S = C / R
           C = 1.0 / R
           GO  TO  160
150        C = P / E(I)
           R = SQRT(C*C+1.0)
           E(I+1) = S * E(I) * R
           S = 1.0 / R
           C = C * S
160        P = C * D(I) - S * G
           D(I+1) = H + S * (C * G + S * D(I))
           DO  180  K = 1, N
              H = Z(K,I+1)
              Z(K,I+1) = S * Z(K,I) + C * H
              Z(K,I) = C * Z(K,I) - S * H
180        CONTINUE
200     CONTINUE
        E(L) = S * P
        D(L) = C * P
        IF  (ABS(E(L)) .GT. B)  GO  TO  130
220     D(L) = D(L) + F
240 CONTINUE
    DO  300  II = 2, N
        I = II - 1
        K = I
        P = D(I)
        DO  260  J = II, N
           IF  (D(J) .GE. P)  GO  TO  260
           K = J
           P = D(J)
260     CONTINUE
        IF  (K .EQ. I)  GO  TO  300
        D(K) = D(I)
        D(I) = P
        DO  280  J = 1, N
           P = Z(J,I)
           Z(J,I) = Z(J,K)
           Z(J,K) = P
280     CONTINUE
300 CONTINUE
    GO  TO  1001
```

```
 1000 IERR = L
 1001 RETURN
      END
      SUBROUTINE  RG(NM,N,A,WR,WI,MATZ,Z,IV1,FV1,IERR)
      INTEGER  N,NM,IS1,IS2,IERR,MATZ
      REAL  A(NM,N),WR(N),WI(N),Z(NM,N),FV1(N)
      INTEGER  IV1(N)
      IF  (N .LE. NM) GO TO 10
      IERR = 10 * N
      GO  TO  50
   10 IS1=1
      IS2=N
      CALL  ELMHES(NM,N,IS1,IS2,A,IV1)
      CALL  ELTRAN(NM,N,IS1,IS2,A,IV1,Z)
      CALL  HQR2(NM,N,IS1,IS2,A,WR,WI,Z,IERR)
   50 RETURN
      END
      SUBROUTINE  ELMHES(NM,N,LOW,IGH,A,INT)
      INTEGER  I,J,M,N,LA,NM,IGH,KP1,LOW,MM1,MP1
      REAL  A(NM,N)
      REAL  X,Y
      REAL  ABS
      INTEGER  INT(IGH)
      LA = IGH - 1
      KP1 = LOW + 1
      IF (LA .LT. KP1) GO  TO  200
      DO  180  M = KP1, LA
          MM1 = M - 1
          X = 0.0
          I = M
          DO  100  J = M, IGH
              IF (ABS(A(J,MM1)) .LE. ABS(X)) GO  TO  100
              X = A(J,MM1)
              I = J
  100     CONTINUE
          INT(M) = I
          IF (I .EQ. M) GO  TO  130
          DO  110  J = MM1, N
              Y = A(I,J)
              A(I,J) = A(M,J)
              A(M,J) = Y
  110     CONTINUE
          DO  120  J = 1, IGH
              Y = A(J,I)
              A(J,I) = A(J,M)
              A(J,M) = Y
  120     CONTINUE
  130     IF (X .EQ. 0.0) GO  TO  180
          MP1 = M + 1
          DO  160  I = MP1, IGH
              Y = A(I,MM1)
              IF (Y .EQ. 0.0) GO  TO  160
              Y = Y / X
              A(I,MM1) = Y
              DO  140  J = M, N
  140         A(I,J) = A(I,J) - Y * A(M,J)
              DO  150  J = 1, IGH
  150         A(J,M) = A(J,M) + Y * A(J,I)
  160     CONTINUE
  180 CONTINUE
  200 RETURN
      END
      SUBROUTINE  ELTRAN(NM,N,LOW,IGH,A,INT,Z)
      INTEGER  I,J,N,KL,MM,MP,NM,IGH,LOW,MP1
      REAL  A(NM,IGH),Z(NM,N)
      INTEGER  INT(IGH)
      DO  80  I = 1, N
          DO  60  J = 1, N
   60         Z(I,J) = 0.0
          Z(I,I) = 1.0
   80 CONTINUE
      KL = IGH - LOW - 1
      IF (KL .LT. 1) GO  TO  200
      DO  140  MM = 1, KL
          MP = IGH - MM
          MP1 = MP + 1
          DO  100  I = MP1, IGH
  100         Z(I,MP) = A(I,MP-1)
          I = INT(MP)
          IF (I .EQ. MP) GO  TO  140
          DO  130  J = MP, IGH
              Z(MP,J) = Z(I,J)
              Z(I,J) = 0.0
  130     CONTINUE
          Z(I,MP) = 1.0
  140 CONTINUE
  200 RETURN
      END
      SUBROUTINE  HQR2(NM,N,LOW,IGH,H,WR,WI,Z,IERR)
      INTEGER  I,J,K,L,M,N,EN,II,JJ,LL,MM,NA,NM,NN,
     X         IGH,ITS,LOW,MP2,ENM2,IERR
      REAL  H(NM,N),WR(N),WI(N),Z(NM,N)
      REAL  P,Q,R,S,T,W,X,Y,RA,SA,VI,VR,ZZ,NORM,MACHEP
      REAL  SQRT,ABS,SIGN
      INTEGER MIN0
      LOGICAL  NOTLAS
      COMPLEX Z3
      COMPLEX CMPLX
      REAL  REAL,AIMAG
      MACHEP = 2.**(-26)
```

```
      IERR = 0
      NORM = 0.0
      K = 1
      DO 50 I = 1, N
         DO 40 J = K, N
40          NORM = NORM + ABS(H(I,J))
         K = I
         IF (I .GE. LOW .AND. I .LE. IGH) GO TO 50
         WR(I) = H(I,I)
         WI(I) = 0.0
50    CONTINUE
      EN = IGH
      T = 0.0
60    IF (EN .LT. LOW) GO TO 340
      ITS = 0
      NA = EN - 1
      ENM2 = NA - 1
70    DO 80 LL = LOW, EN
         L = EN + LOW - LL
         IF (L .EQ. LOW) GO TO 100
         S = ABS(H(L-1,L-1)) + ABS(H(L,L))
         IF (S .EQ. 0.0) S = NORM
         IF (ABS(H(L,L-1)) .LE. MACHEP * S) GO TO 100
80    CONTINUE
100   X = H(EN,EN)
      IF (L .EQ. EN) GO TO 270
      Y = H(NA,NA)
      W = H(EN,NA) * H(NA,EN)
      IF (L .EQ. NA) GO TO 280
      IF (ITS .EQ. 30) GO TO 1000
      IF (ITS .NE. 10 .AND. ITS .NE. 20) GO TO 130
      T = T + X
      DO 120 I = LOW, EN
120      H(I,I) = H(I,I) - X
      S = ABS(H(EN,NA)) + ABS(H(NA,ENM2))
      X = 0.75 * S
      Y = X
      W = -0.4375 *.S * S
130   ITS = ITS + 1
      DO 140 MM = L, ENM2
         M = ENM2 + L - MM
         ZZ = H(M,M)
         R = X - ZZ
         S = Y - ZZ
         P = (R * S - W) / H(M+1,M) + H(M,M+1)
         Q = H(M+1,M+1) - ZZ -R - S
         R = H(M+2,M+1)
         S = ABS(P) + ABS(Q) + ABS(R)
         P = P / S
         Q = Q / S
         R = R / S
         IF (M .EQ. L) GO TO 150
         IF (ABS(H(M,M-1)) * (ABS(Q) + ABS(R)) .LE. MACHEP * ABS(P)
     X    * (ABS(H(M-1,M-1)) + ABS(ZZ) + ABS(H(M+1,M+1)))) GO TO 150
140   CONTINUE
150   MP2 = M + 2
      DO 160 I = MP2, EN
         H(I,I-2) = 0.0
         IF (I .EQ. MP2) GO TO 160
         H(I,I-3) = 0.0
160   CONTINUE
      DO 260 K = M, NA
         NOTLAS = K .NE. NA
         IF (K .EQ. M) GO TO 170
         P = H(K,K-1)
         Q = H(K+1,K-1)
         R = 0.0
         IF (NOTLAS) R = H(K+2,K-1)
         X = ABS(P) + ABS(Q) + ABS(R)
         IF (X .EQ. 0.0) GO TO 260
         P = P / X
         Q = Q / X
         R = R / X
170      S = SIGN(SQRT(P*P+Q*Q+R*R),P)
         IF (K .EQ. M) GO TO 180
         H(K,K-1) = -S * X
         GO TO 190
180      IF (L .NE. M) H(K,K-1) = -H(K,K-1)
190      P = P + S
         X = P / S
         Y = Q / S
         ZZ = R / S
         Q = Q / P
         R = R / P
         DO 210 J = K, N
            P = H(K,J) + Q * H(K+1,J)
            IF (.NOT. NOTLAS) GO TO 200
            P = P + R * H(K+2,J)
            H(K+2,J) = H(K+2,J) - P * ZZ
200         H(K+1,J) = H(K+1,J) - P * Y
            H(K,J) = H(K,J) - P * X
210      CONTINUE
         J = MIN0(EN,K+3)
         DO 230 I = 1, J
            P = X * H(I,K) + Y * H(I,K+1)
            IF (.NOT. NOTLAS) GO TO 220
            P = P + ZZ * H(I,K+2)
            H(I,K+2) = H(I,K+2) - P * R
220         H(I,K+1) = H(I,K+1) - P * Q
            H(I,K) = H(I,K) - P
```

```
  230    CONTINUE
         DO 250 I = LOW, IGH
            P = X * Z(I,K) + Y * Z(I,K+1)
            IF (.NOT. NOTLAS) GO TO 240
            P = P + ZZ * Z(I,K+2)
            Z(I,K+2) = Z(I,K+2) - P * R
  240       Z(I,K+1) = Z(I,K+1) - P * Q
            Z(I,K) = Z(I,K) - P
  250    CONTINUE
  260 CONTINUE
      GO TO 70
  270 H(EN,EN) = X + T
      WR(EN) = H(EN,EN)
      WI(EN) = 0.0
      EN = NA
      GO TO 60
  280 P = (Y - X) / 2.0
      Q = P * P + W
      ZZ = SQRT(ABS(Q))
      H(EN,EN) = X + T
      X = H(EN,EN)
      H(NA,NA) = Y + T
      IF (Q .LT. 0.0) GO TO 320
      ZZ = P + SIGN(ZZ,P)
      WR(NA) = X + ZZ
      WR(EN) = WR(NA)
      IF (ZZ .NE. 0.0) WR(EN) = X - W / ZZ
      WI(NA) = 0.0
      WI(EN) = 0.0
      X = H(EN,NA)
      S = ABS(X) + ABS(ZZ)
      P = X / S
      Q = ZZ / S
      R = SQRT(P*P+Q*Q)
      P = P / R
      Q = Q / R
      DO 290 J = NA, N
         ZZ = H(NA,J)
         H(NA,J) = Q * ZZ + P * H(EN,J)
         H(EN,J) = Q * H(EN,J) - P * ZZ
  290 CONTINUE
      DO 300 I = 1, EN
         ZZ = H(I,NA)
         H(I,NA) = Q * ZZ + P * H(I,EN)
         H(I,EN) = Q * H(I,EN) - P * ZZ
  300 CONTINUE
      DO 310 I = LOW, IGH
         ZZ = Z(I,NA)
         Z(I,NA) = Q * ZZ + P * Z(I,EN)
         Z(I,EN) = Q * Z(I,EN) - P * ZZ
  310 CONTINUE
      GO TO 330
  320 WR(NA) = X + P
      WR(EN) = X + P
      WI(NA) = ZZ
      WI(EN) = -ZZ
  330 EN = ENM2
      GO TO 60
  340 IF (NORM .EQ. 0.0) GO TO 1001
      DO 800 NN = 1, N
         EN = N + 1 - NN
         P = WR(EN)
         Q = WI(EN)
         NA = EN - 1
         IF (Q) 710, 600, 800
  600    M = EN
         H(EN,EN) = 1.0
         IF (NA .EQ. 0) GO TO 800
         DO 700 II = 1, NA
            I = EN - II
            W = H(I,I) - P
            R = H(I,EN)
            IF (M .GT. NA) GO TO 620
            DO 610 J = M, NA
  610          R = R + H(I,J) * H(J,EN)
  620       IF (WI(I) .GE. 0.0) GO TO 630
            ZZ = W
            S = R
            GO TO 700
  630       M = I
            IF (WI(I) .NE. 0.0) GO TO 640
            T = W
            IF (W .EQ. 0.0) T = MACHEP * NORM
            H(I,EN) = -R / T
            GO TO 700
  640       X = H(I,I+1)
            Y = H(I+1,I)
            Q = (WR(I) - P) * (WR(I) - P) + WI(I) * WI(I)
            T = (X * S - ZZ * R) / Q
            H(I,EN) = T
            IF (ABS(X) .LE. ABS(ZZ)) GO TO 650
            H(I+1,EN) = (-R - W * T) / X
            GO TO 700
  650       H(I+1,EN) = (-S - Y * T) / ZZ
  700    CONTINUE
         GO TO 800
  710    M = NA
         IF (ABS(H(EN,NA)) .LE. ABS(H(NA,EN))) GO TO 720
         H(NA,NA) = Q / H(EN,NA)
         H(NA,EN) = -(H(EN,EN) - P) / H(EN,NA)
```

```
      GO TO 730
720   Z3 = CMPLX(0.0,-H(NA,EN)) / CMPLX(H(NA,NA)-P,Q)
      H(NA,NA) = REAL(Z3)
      H(NA,EN) = AIMAG(Z3)
730   H(EN,NA) = 0.0
      H(EN,EN) = 1.0
      ENM2 = NA - 1
      IF (ENM2 .EQ. 0) GO TO 800
      DO 790 II = 1, ENM2
         I = NA - II
         W = H(I,I) - P
         RA = 0.0
         SA = H(I,EN)
         DO 760 J = M, NA
            RA = RA + H(I,J) * H(J,NA)
            SA = SA + H(I,J) * H(J,EN)
760      CONTINUE
         IF (WI(I) .GE. 0.0) GO TO 770
         ZZ = W
         R = RA
         S = SA
         GO TO 790
770      M = I
         IF (WI(I) .NE. 0.0) GO TO 780
         Z3 = CMPLX(-RA,-SA) / CMPLX(W,Q)
         H(I,NA) = REAL(Z3)
         H(I,EN) = AIMAG(Z3)
         GO TO 790
780      X = H(I,I+1)
         Y = H(I+1,I)
         VR = (WR(I) - P) * (WR(I) - P) + WI(I) * WI(I) - Q * Q
         VI = (WR(I) - P) * 2.0 * Q
         IF (VR .EQ. 0.0 .AND. VI .EQ. 0.0) VR = MACHEP * NORM
     X    * (ABS(W) + ABS(Q) + ABS(X) + ABS(Y) + ABS(ZZ))
         Z3 = CMPLX(X*R-ZZ*RA+Q*SA,X*S-ZZ*SA-Q*RA) / CMPLX(VR,VI)
         H(I,NA) = REAL(Z3)
         H(I,EN) = AIMAG(Z3)
         IF (ABS(X) .LE. ABS(ZZ) + ABS(Q)) GO TO 785
         H(I+1,NA) = (-RA - W * H(I,NA) + Q * H(I,EN)) / X
         H(I+1,EN) = (-SA - W * H(I,EN) - Q * H(I,NA)) / X
         GO TO 790
785      Z3 = CMPLX(-R-Y*H(I,NA),-S-Y*H(I,EN)) / CMPLX(ZZ,Q)
         H(I+1,NA) = REAL(Z3)
         H(I+1,EN) = AIMAG(Z3)
790   CONTINUE
800   CONTINUE
      DO 840 I = 1, N
         IF (I .GE. LOW .AND. I .LE. IGH) GO TO 840
         DO 820 J = I, N
820         Z(I,J) = H(I,J)
840   CONTINUE
      DO 880 JJ = LOW, N
         J = N + LOW - JJ
         M = MIN0(J,IGH)
         DO 880 I = LOW, IGH
            ZZ = 0.0
            DO 860 K = LOW, M
860            ZZ = ZZ + Z(I,K) * H(K,J)
            Z(I,J) = ZZ
880   CONTINUE
      GO TO 1001
1000  IERR = EN
1001  RETURN
      END
```