NAG Library Routine Document

F08PEF (DHSEQR)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

Warning. The specification of the parameter LWORK changed at Mark 20: LWORK is no longer redundant.

1 Purpose

F08PEF (DHSEQR) computes all the eigenvalues and, optionally, the Schur factorization of a real Hessenberg matrix or a real general matrix which has been reduced to Hessenberg form.

2 Specification

```
SUBROUTINE F08PEF (JOB, COMPZ, N, ILO, IHI, H, LDH, WR, WI, Z, LDZ, WORK,       &
                   LWORK, INFO)
INTEGER            N, ILO, IHI, LDH, LDZ, LWORK, INFO
REAL (KIND=nag_wp) H(LDH,*), WR(*), WI(*), Z(LDZ,*), WORK(max(1,LWORK))
CHARACTER(1)       JOB, COMPZ
```

The routine may be called by its LAPACK name ***dhseqr***.

3 Description

F08PEF (DHSEQR) computes all the eigenvalues and, optionally, the Schur factorization of a real upper Hessenberg matrix H:

$$H = ZTZ^{\mathrm{T}},$$

where T is an upper quasi-triangular matrix (the Schur form of H), and Z is the orthogonal matrix whose columns are the Schur vectors z_i. See Section 8 for details of the structure of T.

The routine may also be used to compute the Schur factorization of a real general matrix A which has been reduced to upper Hessenberg form H:

$$\begin{aligned} A &= QHQ^{\mathrm{T}}, \text{ where } Q \text{ is orthogonal,} \\ &= (QZ)T(QZ)^{\mathrm{T}}. \end{aligned}$$

In this case, after F08NEF (DGEHRD) has been called to reduce A to Hessenberg form, F08NFF (DORGHR) must be called to form Q explicitly; Q is then passed to F08PEF (DHSEQR), which must be called with COMPZ = 'V'.

The routine can also take advantage of a previous call to F08NHF (DGEBAL) which may have balanced the original matrix before reducing it to Hessenberg form, so that the Hessenberg matrix H has the structure:

$$\begin{pmatrix} H_{11} & H_{12} & H_{13} \\ & H_{22} & H_{23} \\ & & H_{33} \end{pmatrix}$$

where H_{11} and H_{33} are upper triangular. If so, only the central diagonal block H_{22} (in rows and columns i_{lo} to i_{hi}) needs to be further reduced to Schur form (the blocks H_{12} and H_{23} are also affected). Therefore the values of i_{lo} and i_{hi} can be supplied to F08PEF (DHSEQR) directly. Also, F08NJF (DGEBAK) must be called after this routine to permute the Schur vectors of the balanced matrix to those of the original matrix. If F08NHF (DGEBAL) has not been called however, then i_{lo} must be set to 1 and i_{hi} to n. Note that if the Schur factorization of A is required, F08NHF (DGEBAL) must **not** be called with JOB = 'S' or 'B', because the balancing transformation is not orthogonal.

F08PEF (DHSEQR) uses a multishift form of the upper Hessenberg QR algorithm, due to Bai and Demmel (1989). The Schur vectors are normalized so that $\|z_i\|_2 = 1$, but are determined only to within a factor ± 1.

4 References

Bai Z and Demmel J W (1989) On a block implementation of Hessenberg multishift QR iteration *Internat. J. High Speed Comput.* **1** 97–112

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

1: JOB – CHARACTER(1) *Input*

On entry: indicates whether eigenvalues only or the Schur form T is required.

JOB = 'E'
Eigenvalues only are required.

JOB = 'S'
The Schur form T is required.

Constraint: JOB = 'E' or 'S'.

2: COMPZ – CHARACTER(1) *Input*

On entry: indicates whether the Schur vectors are to be computed.

COMPZ = 'N'
No Schur vectors are computed (and the array Z is not referenced).

COMPZ = 'I'
The Schur vectors of H are computed (and the array Z is initialized by the routine).

COMPZ = 'V'
The Schur vectors of A are computed (and the array Z must contain the matrix Q on entry).

Constraint: COMPZ = 'N', 'V' or 'I'.

3: N – INTEGER *Input*

On entry: n, the order of the matrix H.

Constraint: $\mathrm{N} \geq 0$.

4: ILO – INTEGER *Input*
5: IHI – INTEGER *Input*

On entry: if the matrix A has been balanced by F08NHF (DGEBAL), then ILO and IHI must contain the values returned by that routine. Otherwise, ILO must be set to 1 and IHI to N.

Constraint: $\mathrm{ILO} \geq 1$ and $\min(\mathrm{ILO}, \mathrm{N}) \leq \mathrm{IHI} \leq \mathrm{N}$.

6: H(LDH,$*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array H must be at least $\max(1, \mathrm{N})$.

On entry: the n by n upper Hessenberg matrix H, as returned by F08NEF (DGEHRD).

On exit: if JOB = 'E', the array contains no useful information.

If JOB = 'S', H is overwritten by the upper quasi-triangular matrix T from the Schur decomposition (the Schur form) unless $\mathrm{INFO} > 0$.

7: LDH – INTEGER *Input*

On entry: the first dimension of the array H as declared in the (sub)program from which F08PEF (DHSEQR) is called.

Constraint: LDH $\geq$ max$(1, N)$.

8: WR$(*)$ – REAL (KIND=nag_wp) array *Output*
9: WI$(*)$ – REAL (KIND=nag_wp) array *Output*

Note: the dimension of the arrays WR and WI must be at least max$(1, N)$.

On exit: the real and imaginary parts, respectively, of the computed eigenvalues, unless INFO > 0 (in which case see Section 6). Complex conjugate pairs of eigenvalues appear consecutively with the eigenvalue having positive imaginary part first. The eigenvalues are stored in the same order as on the diagonal of the Schur form T (if computed); see Section 8 for details.

10: Z(LDZ,$*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array Z must be at least max$(1, N)$ if COMPZ = 'V' or 'I' and at least 1 if COMPZ = 'N'.

On entry: if COMPZ = 'V', Z must contain the orthogonal matrix Q from the reduction to Hessenberg form.

If COMPZ = 'I', Z need not be set.

On exit: if COMPZ = 'V' or 'I', Z contains the orthogonal matrix of the required Schur vectors, unless INFO > 0.

If COMPZ = 'N', Z is not referenced.

11: LDZ – INTEGER *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which F08PEF (DHSEQR) is called.

Constraints:

if COMPZ = 'I' or 'V', LDZ $\geq$ max$(1, N)$;
if COMPZ = 'N', LDZ ≥ 1.

12: WORK(max$(1, $LWORK$)$) – REAL (KIND=nag_wp) array *Workspace*

On exit: if INFO $= 0$, WORK(1) contains the minimum value of LWORK required for optimal performance.

13: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08PEF (DHSEQR) is called, unless LWORK $= -1$, in which case a workspace query is assumed and the routine only calculates the minimum dimension of WORK.

Constraint: LWORK $\geq$ max$(1, N)$ or LWORK $= -1$.

14: INFO – INTEGER *Output*

On exit: INFO $= 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO $= -i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO > 0

The algorithm has failed to find all the eigenvalues after a total of $30 \times (\mathrm{IHI} - \mathrm{ILO} + 1)$ iterations. If INFO $= i$, elements $1, 2, \ldots, \mathrm{ILO} - 1$ and $i + 1, i + 2, \ldots, n$ of WR and WI contain the real and imaginary parts of contain the eigenvalues which have been found.

If JOB = 'E', then on exit, the remaining unconverged eigenvalues are the eigenvalues of the upper Hessenberg matrix $\hat{H}$, formed from $\mathrm{H(ILO : INFO, ILO : INFO)}$, i.e., the ILO through INFO rows and columns of the final output matrix H.

If JOB = 'S', then on exit

$$(*) \qquad H_i U = U \tilde{H}$$

for some matrix U, where H_i is the input upper Hessenberg matrix and $\tilde{H}$ is an upper Hessenberg matrix formed from $\mathrm{H(INFO + 1 : IHI, INFO + 1 : IHI)}$.

If COMPZ = 'V', then on exit

$$Z_{\mathrm{out}} = Z_{\mathrm{in}} U$$

where U is defined in $(*)$ (regardless of the value of JOB).

If COMPZ = 'I', then on exit

$$Z_{\mathrm{out}} = U$$

where U is defined in $(*)$ (regardless of the value of JOB).

If INFO > 0 and COMPZ = 'N', then Z is not accessed.

7 Accuracy

The computed Schur factorization is the exact factorization of a nearby matrix $(H + E)$, where

$$\|E\|_2 = O(\epsilon)\|H\|_2,$$

and ϵ is the ***machine precision***.

If λ_i is an exact eigenvalue, and $\tilde{\lambda}_i$ is the corresponding computed value, then

$$\left|\tilde{\lambda}_i - \lambda_i\right| \leq \frac{c(n)\epsilon\|H\|_2}{s_i},$$

where $c(n)$ is a modestly increasing function of n, and s_i is the reciprocal condition number of λ_i. The condition numbers s_i may be computed by calling F08QLF (DTRSNA).

8 Further Comments

The total number of floating point operations depends on how rapidly the algorithm converges, but is typically about:

$7n^3$ if only eigenvalues are computed;

$10n^3$ if the Schur form is computed;

$20n^3$ if the full Schur factorization is computed.

The Schur form T has the following structure (referred to as **canonical** Schur form).

If all the computed eigenvalues are real, T is upper triangular, and the diagonal elements of T are the eigenvalues; $\mathrm{WR}(i) = t_{ii}$, for $i = 1, 2, \ldots, n$, and $\mathrm{WI}(i) = 0.0$.

If some of the computed eigenvalues form complex conjugate pairs, then T has 2 by 2 diagonal blocks. Each diagonal block has the form

$$\begin{pmatrix} t_{ii} & t_{i,i+1} \\ t_{i+1,i} & t_{i+1,i+1} \end{pmatrix} = \begin{pmatrix} \alpha & \beta \\ \gamma & \alpha \end{pmatrix}$$

where $\beta\gamma < 0$. The corresponding eigenvalues are $\alpha \pm \sqrt{\beta\gamma}$; $\mathrm{WR}(i) = \mathrm{WR}(i+1) = \alpha$; $\mathrm{WI}(i) = +\sqrt{|\beta\gamma|}$; $\mathrm{WI}(i+1) = -\mathrm{WI}(i)$.

The complex analogue of this routine is F08PSF (ZHSEQR).

9 Example

This example computes all the eigenvalues and the Schur factorization of the upper Hessenberg matrix H, where

$$H = \begin{pmatrix} 0.3500 & -0.1160 & -0.3886 & -0.2942 \\ -0.5140 & 0.1225 & 0.1004 & 0.1126 \\ 0.0000 & 0.6443 & -0.1357 & -0.0977 \\ 0.0000 & 0.0000 & 0.4262 & 0.1632 \end{pmatrix}.$$

See also Section 9 in F08NFF (DORGHR), which illustrates the use of this routine to compute the Schur factorization of a general matrix.

9.1 Program Text

```
    Program f08pefe

!     F08PEF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dgemm, dhseqr, dlange => f06raf, nag_wp, x02ajf,  &
                             x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                 :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)                 :: alpha, beta, norm
      Integer                            :: i, ifail, info, ldc, ldd, ldh, ldz,  &
                                            lwork, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable    :: c(:,:), d(:,:), h(:,:), wi(:),       &
                                            work(:), wr(:), z(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'F08PEF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n
      ldc = n
      ldd = n
      ldh = n
      ldz = n
      lwork = n
      Allocate (c(ldc,n),d(ldd,n),h(ldh,n),wi(n),work(lwork),wr(n),z(ldz,n))

!     Read H from data file
      Read (nin,*)(h(i,1:n),i=1,n)

!     Copy H into D
      d(1:n,1:n) = h(1:n,1:n)

!     Print Matrix H
!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call x04caf('General',' ',n,n,h,ldh,'Matrix H',ifail)

!     Calculate the eigenvalues and Schur factorization of H

!     The NAG name equivalent of dhseqr is f08pef
```

```
      Call dhseqr('Schur form','Initialize Z',n,1,n,h,ldh,wr,wi,z,ldz,work, &
        lwork,info)

      Write (nout,*)
      If (info>0) Then
        Write (nout,*) 'Failure to converge.'
      Else

!       Compute H - Z*T*Z^T from the factorization of A and store in matrix D
!       The NAG name equivelent of dgemm is f06yaf
        alpha = 1.0_nag_wp
        beta = 0.0_nag_wp
        Call dgemm('N','N',n,n,n,alpha,z,ldz,h,ldh,beta,c,ldc)
        alpha = -1.0_nag_wp
        beta = 1.0_nag_wp
        Call dgemm('N','T',n,n,n,alpha,c,ldc,z,ldz,beta,d,ldd)

!       Find norm of matrix D and print warning if it is too large
!       f06raf is the NAG name equivalent of the LAPACK auxiliary dlange
        norm = dlange('O',ldd,n,d,ldd,work)
        If (norm>x02ajf()**0.8_nag_wp) Then
          Write (nout,*) 'Norm of H-(Z*T*Z^T) is much greater than 0.'
          Write (nout,*) 'Schur factorization has failed.'
        Else
!         Print eigenvalues
          Write (nout,*) 'Eigenvalues'
          Write (nout,99999)(' (',wr(i),',',wi(i),')',i=1,n)
        End If
      End If

99999 Format (1X,A,F8.4,A,F8.4,A)
    End Program f08pefe
```

9.2 Program Data

```
F08PEF Example Program Data
  4                                   :Value of N
  0.3500  -0.1160  -0.3886  -0.2942
 -0.5140   0.1225   0.1004   0.1126
  0.0000   0.6443  -0.1357  -0.0977
  0.0000   0.0000   0.4262   0.1632   :End of matrix H
```

9.3 Program Results

```
 F08PEF Example Program Results
 Matrix H
          1       2       3       4
 1   0.3500 -0.1160 -0.3886 -0.2942
 2  -0.5140  0.1225  0.1004  0.1126
 3   0.0000  0.6443 -0.1357 -0.0977
 4   0.0000  0.0000  0.4262  0.1632

 Eigenvalues
  (  0.7995,  0.0000)
  ( -0.0994,  0.4008)
  ( -0.0994, -0.4008)
  ( -0.1007,  0.0000)
```