NAG Library Routine Document

F08JUF (ZPTEQR)

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 argument WORK changed at Mark 20: the length of WORK needs to be increased.

1   Purpose

F08JUF (ZPTEQR) computes all the eigenvalues and, optionally, all the eigenvectors of a complex Hermitian positive definite matrix which has been reduced to tridiagonal form.

2   Specification

```
SUBROUTINE F08JUF (COMPZ, N, D, E, Z, LDZ, WORK, INFO)

INTEGER              N, LDZ, INFO
REAL (KIND=nag_wp)   D(*), E(*), WORK(4*N)
COMPLEX (KIND=nag_wp) Z(LDZ,*)
CHARACTER(1)         COMPZ
```

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

3   Description

F08JUF (ZPTEQR) computes all the eigenvalues and, optionally, all the eigenvectors of a real symmetric positive definite tridiagonal matrix T. In other words, it can compute the spectral factorization of T as

$$T = Z \Lambda Z^\mathrm{T},$$

where Λ is a diagonal matrix whose diagonal elements are the eigenvalues λ_i, and Z is the orthogonal matrix whose columns are the eigenvectors z_i. Thus

$$T z_i = \lambda_i z_i, \quad i = 1, 2, \ldots, n.$$

The routine stores the real orthogonal matrix Z in a complex array, so that it may be used to compute all the eigenvalues and eigenvectors of a complex Hermitian positive definite matrix A which has been reduced to tridiagonal form T:

$$
\begin{aligned}
A &= QTQ^\mathrm{H}, \text{ where } Q \text{ is unitary} \\
&= (QZ)\Lambda(QZ)^\mathrm{H}.
\end{aligned}
$$

In this case, the matrix Q must be formed explicitly and passed to F08JUF (ZPTEQR), which must be called with COMPZ = 'V'. The routines which must be called to perform the reduction to tridiagonal form and form Q are:

full matrix	F08FSF (ZHETRD) and F08FTF (ZUNGTR)
full matrix, packed storage	F08GSF (ZHPTRD) and F08GTF (ZUPGTR)
band matrix	F08HSF (ZHBTRD) with VECT = 'V'.

F08JUF (ZPTEQR) first factorizes T as LDL^H where L is unit lower bidiagonal and D is diagonal. It forms the bidiagonal matrix $B = LD^{\frac{1}{2}}$, and then calls F08MSF (ZBDSQR) to compute the singular values of B which are the same as the eigenvalues of T. The method used by the routine allows high relative accuracy to be achieved in the small eigenvalues of T. The eigenvectors are normalized so that $\|z_i\|_2 = 1$, but are determined only to within a complex factor of absolute value 1.

4 References

Barlow J and Demmel J W (1990) Computing accurate eigensystems of scaled diagonally dominant matrices *SIAM J. Numer. Anal.* **27** 762–791

5 Arguments

1:      COMPZ – CHARACTER(1)   *Input*

On entry: indicates whether the eigenvectors are to be computed.

COMPZ = 'N'
       Only the eigenvalues are computed (and the array Z is not referenced).

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

COMPZ = 'I'
       The eigenvalues and eigenvectors of T are computed (and the array Z is initialized by the routine).

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

2:      N – INTEGER   *Input*

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

Constraint: N $\geq$ 0.

3:      D($*$) – REAL (KIND=nag_wp) array   *Input/Output*

Note: the dimension of the array D must be at least $\max(1, N)$.

On entry: the diagonal elements of the tridiagonal matrix T.

On exit: the n eigenvalues in descending order, unless INFO > 0, in which case D is overwritten.

4:      E($*$) – REAL (KIND=nag_wp) array   *Input/Output*

Note: the dimension of the array E must be at least $\max(1, N - 1)$.

On entry: the off-diagonal elements of the tridiagonal matrix T.

On exit: E is overwritten.

5:      Z(LDZ, $*$) – COMPLEX (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 unitary matrix Q from the reduction to tridiagonal form.

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

On exit: if COMPZ = 'V' or 'I', the n required orthonormal eigenvectors stored as columns of Z; the ith column corresponds to the ith eigenvalue, where $i = 1, 2, \ldots, n$, unless INFO > 0.

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

6:      LDZ – INTEGER   *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which F08JUF (ZPTEQR) is called.

Constraints:

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

7:    WORK($4 \times$ N) – REAL (KIND=nag_wp) array   *Workspace*

8:    INFO – INTEGER   *Output*

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

6   Error Indicators and Warnings

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

> If INFO $= i$, the leading minor of order i is not positive definite and the Cholesky factorization of T could not be completed. Hence T itself is not positive definite.

> If INFO $= $ N $+ i$, the algorithm to compute the singular values of the Cholesky factor B failed to converge; i off-diagonal elements did not converge to zero.

7   Accuracy

The eigenvalues and eigenvectors of T are computed to high relative accuracy which means that if they vary widely in magnitude, then any small eigenvalues (and corresponding eigenvectors) will be computed more accurately than, for example, with the standard QR method. However, the reduction to tridiagonal form (prior to calling the routine) may exclude the possibility of obtaining high relative accuracy in the small eigenvalues of the original matrix if its eigenvalues vary widely in magnitude.

To be more precise, let H be the tridiagonal matrix defined by $H = DTD$, where D is diagonal with $d_{ii} = t_{ii}^{-\frac{1}{2}}$, and $h_{ii} = 1$ for all i. If λ_i is an exact eigenvalue of T and $\tilde{\lambda}_i$ is the corresponding computed value, then

$$\left| \tilde{\lambda}_i - \lambda_i \right| \leq c(n)\epsilon\kappa_2(H)\lambda_i$$

where $c(n)$ is a modestly increasing function of n, ϵ is the ***machine precision***, and $\kappa_2(H)$ is the condition number of H with respect to inversion defined by: $\kappa_2(H) = \|H\| \cdot \|H^{-1}\|$.

If z_i is the corresponding exact eigenvector of T, and $\tilde{z}_i$ is the corresponding computed eigenvector, then the angle $\theta(\tilde{z}_i, z_i)$ between them is bounded as follows:

$$\theta(\tilde{z}_i, z_i) \leq \frac{c(n)\epsilon\kappa_2(H)}{relgap_i}$$

where $relgap_i$ is the relative gap between λ_i and the other eigenvalues, defined by

$$relgap_i = \min_{i \neq j} \frac{\left| \lambda_i - \lambda_j \right|}{\left(\lambda_i + \lambda_j \right)}.$$

8   Parallelism and Performance

F08JUF (ZPTEQR) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F08JUF (ZPTEQR) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9   Further Comments

The total number of real floating-point operations is typically about $30n^2$ if COMPZ = 'N' and about $12n^3$ if COMPZ = 'V' or 'I', but depends on how rapidly the algorithm converges. When COMPZ = 'N', the operations are all performed in scalar mode; the additional operations to compute the eigenvectors when COMPZ = 'V' or 'I' can be vectorized and on some machines may be performed much faster.

The real analogue of this routine is F08JGF (DPTEQR).

10   Example

This example computes all the eigenvalues and eigenvectors of the complex Hermitian positive definite matrix A, where

$$A = \begin{pmatrix} 6.02 + 0.00i & -0.45 + 0.25i & -1.30 + 1.74i & 1.45 - 0.66i \\ -0.45 - 0.25i & 2.91 + 0.00i & 0.05 + 1.56i & -1.04 + 1.27i \\ -1.30 - 1.74i & 0.05 - 1.56i & 3.29 + 0.00i & 0.14 + 1.70i \\ 1.45 + 0.66i & -1.04 - 1.27i & 0.14 - 1.70i & 4.18 + 0.00i \end{pmatrix}.$$

10.1 Program Text

```
      Program f08jufe

!      F08JUF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
       Use nag_library, Only: dznrm2, f06tff, nag_wp, x04dbf, zhetrd, zpteqr,   &
                              zungtr
!      .. Implicit None Statement ..
       Implicit None
!      .. Parameters ..
       Integer, Parameter                :: nin = 5, nout = 6
!      .. Local Scalars ..
       Complex (Kind=nag_wp)             :: scal
       Integer                           :: i, ifail, info, k, lda, ldz, lwork,  &
                                            n
       Character (1)                     :: uplo
!      .. Local Arrays ..
       Complex (Kind=nag_wp), Allocatable :: a(:,:), tau(:), work(:), z(:,:)
       Real (Kind=nag_wp), Allocatable   :: d(:), e(:), rwork(:)
       Character (1)                     :: clabs(1), rlabs(1)
!      .. Intrinsic Procedures ..
       Intrinsic                         :: abs, conjg, maxloc
!      .. Executable Statements ..
       Write (nout,*) 'F08JUF Example Program Results'
!      Skip heading in data file
       Read (nin,*)
       Read (nin,*) n
       lda = n
       ldz = n
       lwork = 64*n
       Allocate (a(lda,n),tau(n),work(lwork),z(ldz,n),d(n),e(n),rwork(4*n))

!      Read A from data file

       Read (nin,*) uplo
       If (uplo=='U') Then
         Read (nin,*)(a(i,i:n),i=1,n)
       Else If (uplo=='L') Then
         Read (nin,*)(a(i,1:i),i=1,n)
```

```
        End If

!       Reduce A to tridiagonal form T = (Q**H)*A*Q

!       The NAG name equivalent of zhetrd is f08fsf
        Call zhetrd(uplo,n,a,lda,d,e,tau,work,lwork,info)

!       Copy A into Z

        Call f06tff(uplo,n,n,a,lda,z,ldz)

!       Form Q explicitly, storing the result in Z

!       The NAG name equivalent of zungtr is f08ftf
        Call zungtr(uplo,n,z,ldz,tau,work,lwork,info)

!       Calculate all the eigenvalues and eigenvectors of A

!       The NAG name equivalent of zpteqr is f08juf
        Call zpteqr('V',n,d,e,z,ldz,rwork,info)

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

!         Print eigenvalues and eigenvectors

          Write (nout,*) 'Eigenvalues'
          Write (nout,99999) d(1:n)
          Write (nout,*)
          Flush (nout)

!         Normalize the eigenvectors, largest element real
          Do i = 1, n
            rwork(1:n) = abs(z(1:n,i))
            k = maxloc(rwork(1:n),1)
            scal = conjg(z(k,i))/abs(z(k,i))/dznrm2(n,z(1,i),1)
            z(1:n,i) = z(1:n,i)*scal
          End Do

!         ifail: behaviour on error exit
!               =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
          ifail = 0
          Call x04dbf('General',' ',n,n,z,ldz,'Bracketed','F7.4','Eigenvectors', &
            'Integer',rlabs,'Integer',clabs,80,0,ifail)

        End If

99999 Format (8X,4(F7.4,11X,:))
      End Program f08jufe
```

10.2 Program Data

```
F08JUF Example Program Data
  4                                                          :Value of N
  'L'                                                        :Value of UPLO
 ( 6.02, 0.00)
 (-0.45,-0.25) ( 2.91, 0.00)
 (-1.30,-1.74) ( 0.05,-1.56) ( 3.29, 0.00)
 ( 1.45, 0.66) (-1.04,-1.27) ( 0.14,-1.70) ( 4.18, 0.00)   :End of matrix A
```

10.3 Program Results

```
 F08JUF Example Program Results

 Eigenvalues
         7.9995           5.9976           2.0003           0.4026

 Eigenvectors
```

```
                        1                   2                   3                   4
  1 ( 0.7289, 0.0000) ( 0.2001, 0.4724) (-0.2133, 0.1498) ( 0.0995,-0.3573)
  2 (-0.1651,-0.2067) (-0.2461, 0.3742) ( 0.7308, 0.0000) ( 0.2867,-0.3364)
  3 (-0.4170,-0.1413) ( 0.4476, 0.1455) (-0.3282, 0.0471) ( 0.6890, 0.0000)
  4 ( 0.1748, 0.4175) ( 0.5610, 0.0000) ( 0.5203, 0.1317) ( 0.0659, 0.4336)
```