

# NAG Library Routine Document

## F08PSF (ZHSEQR)

**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

F08PSF (ZHSEQR) computes all the eigenvalues and, optionally, the Schur factorization of a complex Hessenberg matrix or a complex general matrix which has been reduced to Hessenberg form.

### 2 Specification

```
SUBROUTINE F08PSF (JOB, COMPZ, N, ILO, IHI, H, LDH, W, Z, LDZ, WORK, &
                  LWORK, INFO)
```

```
INTEGER          N, ILO, IHI, LDH, LDZ, LWORK, INFO
COMPLEX (KIND=nag_wp) H(LDH,*), W(*), Z(LDZ,*), WORK(max(1,LWORK))
CHARACTER(1)     JOB, COMPZ
```

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

### 3 Description

F08PSF (ZHSEQR) computes all the eigenvalues and, optionally, the Schur factorization of a complex upper Hessenberg matrix  $H$ :

$$H = ZTZ^H,$$

where  $T$  is an upper triangular matrix (the Schur form of  $H$ ), and  $Z$  is the unitary matrix whose columns are the Schur vectors  $z_i$ . The diagonal elements of  $T$  are the eigenvalues of  $H$ .

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

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

In this case, after F08NSF (ZGEBRD) has been called to reduce  $A$  to Hessenberg form, F08NTF (ZUNGHR) must be called to form  $Q$  explicitly;  $Q$  is then passed to F08PSF (ZHSEQR), which must be called with  $\text{COMPZ} = 'V'$ .

The routine can also take advantage of a previous call to F08NVF (ZGEBAL) 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  $i_{10}$  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  $i_{10}$  and  $i_{hi}$  can be supplied to F08PSF (ZHSEQR) directly. Also, F08NWF (ZGEBAK) must be called after this routine to permute the Schur vectors of the balanced matrix to those of the original matrix. If F08NVF (ZGEBAL) has not been called however, then  $i_{10}$  must be set to 1 and  $i_{hi}$  to  $n$ . Note that if the Schur factorization of  $A$  is required, F08NVF (ZGEBAL) must **not** be called with  $\text{JOB} = 'S'$  or  $'B'$ , because the balancing transformation is not unitary.

F08PSF (ZHSEQR) 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 complex factor of absolute value 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 = 'V'  
     The Schur vectors of  $A$  are computed (and the array  $Z$  must contain the matrix  $Q$  on entry).  
 COMPZ = 'I'  
     The Schur vectors of  $H$  are computed (and the array  $Z$  is initialized by the routine).  
*Constraint:* COMPZ = 'N', 'V' or 'I'.
- 3: N – INTEGER *Input*  
*On entry:*  $n$ , the order of the matrix  $H$ .  
*Constraint:*  $N \geq 0$ .
- 4: ILO – INTEGER *Input*  
 5: IHI – INTEGER *Input*  
*On entry:* if the matrix  $A$  has been balanced by F08NVF (ZGEBAL), then ILO and IHI must contain the values returned by that routine. Otherwise, ILO must be set to 1 and IHI to  $N$ .  
*Constraint:*  $ILO \geq 1$  and  $\min(ILO, N) \leq IHI \leq N$ .
- 6: H(LDH,\*) – COMPLEX (KIND=nag\_wp) array *Input/Output*  
**Note:** the second dimension of the array  $H$  must be at least  $\max(1, N)$ .  
*On entry:* the  $n$  by  $n$  upper Hessenberg matrix  $H$ , as returned by F08NSF (ZGEHRD).  
*On exit:* if JOB = 'E', the array contains no useful information.  
 If JOB = 'S',  $H$  is overwritten by the upper triangular matrix  $T$  from the Schur decomposition (the Schur form) unless INFO > 0.

- 7: LDH – INTEGER *Input*  
*On entry:* the first dimension of the array H as declared in the (sub)program from which F08PSF (ZHSEQR) is called.  
*Constraint:*  $LDH \geq \max(1, N)$ .
- 8: W(\*) – COMPLEX (KIND=nag\_wp) array *Output*  
**Note:** the dimension of the array W must be at least  $\max(1, N)$ .  
*On exit:* the computed eigenvalues, unless  $INFO > 0$  (in which case see Section 6). The eigenvalues are stored in the same order as on the diagonal of the Schur form  $T$  (if computed).
- 9: 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 Hessenberg form.  
 If  $COMPZ = 'I'$ , Z need not be set.  
*On exit:* if  $COMPZ = 'V'$  or  $'I'$ , Z contains the unitary matrix of the required Schur vectors, unless  $INFO > 0$ .  
 If  $COMPZ = 'N'$ , Z is not referenced.
- 10: LDZ – INTEGER *Input*  
*On entry:* the first dimension of the array Z as declared in the (sub)program from which F08PSF (ZHSEQR) is called.  
*Constraints:*  
     if  $COMPZ = 'V'$  or  $'I'$ ,  $LDZ \geq \max(1, N)$ ;  
     if  $COMPZ = 'N'$ ,  $LDZ \geq 1$ .
- 11: WORK(max(1,LWORK)) – COMPLEX (KIND=nag\_wp) array *Workspace*  
*On exit:* if  $INFO = 0$ , the real part of WORK(1) contains the minimum value of LWORK required for optimal performance.
- 12: LWORK – INTEGER *Input*  
*On entry:* the dimension of the array WORK as declared in the (sub)program from which F08PSF (ZHSEQR) 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  $LWORK = -1$ .
- 13: 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

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

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

If  $\text{JOB} = \text{'S'}$ , then on exit

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

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

If  $\text{COMPZ} = \text{'V'}$ , then on exit

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

where  $U$  is defined in (\*) (regardless of the value of  $\text{JOB}$ ).

If  $\text{COMPZ} = \text{'I'}$ , then on exit

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

where  $U$  is defined in (\*) (regardless of the value of  $\text{JOB}$ ).

If  $\text{INFO} > 0$  and  $\text{COMPZ} = \text{'N'}$ , 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  $\epsilon$  is the *machine precision*.

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

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

where  $c(n)$  is a modestly increasing function of  $n$ , and  $s_i$  is the reciprocal condition number of  $\lambda_i$ . The condition numbers  $s_i$  may be computed by calling F08QYF (ZTRSNA).

## 8 Parallelism and Performance

F08PSF (ZHSEQR) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F08PSF (ZHSEQR) 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 depends on how rapidly the algorithm converges, but is typically about:

$25n^3$  if only eigenvalues are computed;

$35n^3$  if the Schur form is computed;

$70n^3$  if the full Schur factorization is computed.

The real analogue of this routine is F08PEF (DHSEQR).

## 10 Example

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

$$H = \begin{pmatrix} -3.9700 - 5.0400i & -1.1318 - 2.5693i & -4.6027 - 0.1426i & -1.4249 + 1.7330i \\ -5.4797 + 0.0000i & 1.8585 - 1.5502i & 4.4145 - 0.7638i & -0.4805 - 1.1976i \\ 0.0000 + 0.0000i & 6.2673 + 0.0000i & -0.4504 - 0.0290i & -1.3467 + 1.6579i \\ 0.0000 + 0.0000i & 0.0000 + 0.0000i & -3.5000 + 0.0000i & 2.5619 - 3.3708i \end{pmatrix}.$$

See also Section 10 in F08NTF (ZUNGHR), which illustrates the use of this routine to compute the Schur factorization of a general matrix.

### 10.1 Program Text

```

Program f08psfe

!      F08PSF Example Program Text

!      Mark 25 Release. NAG Copyright 2014.

!      .. Use Statements ..
Use nag_library, Only: nag_wp, x02ajf, x04dbf, zgemm, zhseqr,      &
                        zlange => f06uaf
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
Complex (Kind=nag_wp)      :: alpha, beta
Real (Kind=nag_wp)        :: norm
Integer                    :: i, ifail, info, ldc, ldd, ldh, ldz, &
                        lwork, n
!      .. Local Arrays ..
Complex (Kind=nag_wp), Allocatable :: c(:,,:), d(:,,:), h(:,,:), w(:),      &
                        work(:), z(:,,:)
Real (Kind=nag_wp)          :: rwork(1)
Character (1)               :: clabs(1), rlabs(1)
!      .. Intrinsic Procedures ..
Intrinsic                   :: cmplx
!      .. Executable Statements ..
Write (nout,*) 'F08PSF Example Program Results'
Write (nout,*)
Flush (nout)
!      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),w(n),work(lwork),z(ldz,n))

!      Read H from data file

Read (nin,*)(h(i,1:n),i=1,n)

!      Store H in D
d(1:ldd,1:n) = h(1:ldh,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 x04dbf('General',' ',n,n,h,ldh,'Bracketed','F7.4','Matrix H', &
!           'Integer',rlabs,'Integer',clabs,80,0,ifail)

!      Calculate the eigenvalues and Schur factorization of H

!      The NAG name equivalent of zhseqr is f08psf
!      Call zhseqr('Schur form','Initialize Z',n,1,n,h,ldh,w,z,ldz,work,lwork, &
!           info)

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

!           Compute A - Z*T*Z`H from Schur factorization of A, and store in matrix D
!           The NAG name equivalent of zgemm is f06zaf
!           alpha = cmplx(1,kind=nag_wp)
!           beta = cmplx(0,kind=nag_wp)
!           Call zgemm('N','N',n,n,n,alpha,z,ldz,h,ldh,beta,c,ldc)
!           alpha = cmplx(-1,kind=nag_wp)
!           beta = cmplx(1,kind=nag_wp)
!           Call zgemm('N','C',n,n,n,alpha,c,ldc,z,ldz,beta,d,ldd)

!           Find norm of matrix D and print warning if it is too large
!           f06uaf is the NAG name equivalent of the LAPACK auxiliary zlange
!           norm = zlange('O',ldd,n,d,ldd,rwork)

!           If (norm>x02ajf()*0.8_nag_wp) Then
!               Write (nout,*) 'Norm of A-(Z*T*Z`H) is much greater than 0.'
!               Write (nout,*) 'Schur factorization has failed.'
!           Else
!               Print eigenvalues
!               Write (nout,*) 'Eigenvalues'
!               Write (nout,99999)(w(i),i=1,n)
!           End If

!      End If

99999 Format ((3X,4(' (',F7.4,',',F7.4,')':)))
!      End Program f08psfe

```

## 10.2 Program Data

F08PSF Example Program Data

```

4                                     :Value of N
(-3.9700,-5.0400) (-1.1318,-2.5693) (-4.6027,-0.1426) (-1.4249, 1.7330)
(-5.4797, 0.0000) ( 1.8585,-1.5502) ( 4.4145,-0.7638) (-0.4805,-1.1976)
( 0.0000, 0.0000) ( 6.2673, 0.0000) (-0.4504,-0.0290) (-1.3467, 1.6579)
( 0.0000, 0.0000) ( 0.0000, 0.0000) (-3.5000, 0.0000) ( 2.5619,-3.3708)
                                     :End of matrix H

```

## 10.3 Program Results

F08PSF Example Program Results

Matrix H

```

           1           2           3           4
1  (-3.9700,-5.0400) (-1.1318,-2.5693) (-4.6027,-0.1426) (-1.4249, 1.7330)
2  (-5.4797, 0.0000) ( 1.8585,-1.5502) ( 4.4145,-0.7638) (-0.4805,-1.1976)

```

3 ( 0.0000, 0.0000) ( 6.2673, 0.0000) (-0.4504,-0.0290) (-1.3467, 1.6579)  
4 ( 0.0000, 0.0000) ( 0.0000, 0.0000) (-3.5000, 0.0000) ( 2.5619,-3.3708)

Eigenvalues

(-6.0004,-6.9998) (-5.0000, 2.0060) ( 7.9982,-0.9964) ( 3.0023,-3.9998)

---