NAG Fortran Library Routine Document

F08NVF (ZGEBAL)

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.

1 Purpose

F08NVF (ZGEBAL) balances a complex general matrix in order to improve the accuracy of computed eigenvalues and/or eigenvectors.

2 Specification

```
SUBROUTINE F08NVF (JOB, N, A, LDA, ILO, IHI, SCALE, INFO)
INTEGER            N, LDA, ILO, IHI, INFO
double precision   SCALE(*)
complex*16         A(LDA,*)
CHARACTER*1        JOB
```

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

3 Description

F08NVF (ZGEBAL) balances a complex general matrix A. The term 'balancing' covers two steps, each of which involves a similarity transformation of A. The routine can perform either or both of these steps.

1. The routine first attempts to permute A to block upper triangular form by a similarity transformation:

$$PAP^{\mathrm{T}} = A' = \begin{pmatrix} A'_{11} & A'_{12} & A'_{13} \\ 0 & A'_{22} & A'_{23} \\ 0 & 0 & A'_{33} \end{pmatrix}$$

   where P is a permutation matrix, and A'_{11} and A'_{33} are upper triangular. Then the diagonal elements of A'_{11} and A'_{33} are eigenvalues of A. The rest of the eigenvalues of A are the eigenvalues of the central diagonal block A'_{22}, in rows and columns i_{lo} to i_{hi}. Subsequent operations to compute the eigenvalues of A (or its Schur factorization) need only be applied to these rows and columns; this can save a significant amount of work if $i_{\mathrm{lo}} > 1$ and $i_{\mathrm{hi}} < n$. If no suitable permutation exists (as is often the case), the routine sets $i_{\mathrm{lo}} = 1$ and $i_{\mathrm{hi}} = n$, and A'_{22} is the whole of A.

2. The routine applies a diagonal similarity transformation to A', to make the rows and columns of A'_{22} as close in norm as possible:

$$A'' = DA'D^{-1} = \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} A'_{11} & A'_{12} & A'_{13} \\ 0 & A'_{22} & A'_{23} \\ 0 & 0 & A'_{33} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22}^{-1} & 0 \\ 0 & 0 & I \end{pmatrix}.$$

   This scaling can reduce the norm of the matrix (that is, $\|A''_{22}\| < \|A'_{22}\|$) and hence reduce the effect of rounding errors on the accuracy of computed eigenvalues and eigenvectors.

4 References

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 A is to be permuted and/or scaled (or neither).

JOB = 'N'

A is neither permuted nor scaled (but values are assigned to ILO, IHI and SCALE).

JOB = 'P'

A is permuted but not scaled.

JOB = 'S'

A is scaled but not permuted.

JOB = 'B'

A is both permuted and scaled.

Constraint: JOB = 'N', 'P', 'S' or 'B'.

2: N – INTEGER *Input*

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

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

3: A(LDA,$*$) – ***complex*16*** array *Input/Output*

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

On entry: the n by n matrix A.

On exit: is overwritten by the balanced matrix. If JOB = 'N', A is not referenced.

4: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08NVF (ZGEBAL) is called.

Constraint: $\mathrm{LDA} \geq \max(1, \mathrm{N})$.

5: ILO – INTEGER *Output*
6: IHI – INTEGER *Output*

On exit: the values i_{lo} and i_{hi} such that on exit $\mathrm{A}(i,j)$ is zero if $i > j$ and $1 \leq j < i_{\mathrm{lo}}$ or $i_{\mathrm{hi}} < i \leq n$.

If JOB = 'N' or 'S', $i_{\mathrm{lo}} = 1$ and $i_{\mathrm{hi}} = n$.

7: SCALE($*$) – ***double precision*** array *Output*

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

On exit: details of the permutations and scaling factors applied to A. More precisely, if p_j is the index of the row and column interchanged with row and column j and d_j is the scaling factor used to balance row and column j then

$$\mathrm{SCALE}(j) = \begin{cases} p_j, & j = 1, 2, \ldots, i_{\mathrm{lo}} - 1 \\ d_j, & j = i_{\mathrm{lo}}, i_{\mathrm{lo}} + 1, \ldots, i_{\mathrm{hi}} \quad \text{and} \\ p_j, & j = i_{\mathrm{hi}} + 1, i_{\mathrm{hi}} + 2, \ldots, n. \end{cases}$$

The order in which the interchanges are made is n to $i_{\mathrm{hi}} + 1$ then 1 to $i_{\mathrm{lo}} - 1$.

8: 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 $= -i$, the ith parameter had an illegal value. An explanatory message is output, and execution of the program is terminated.

7 Accuracy

The errors are negligible, compared with those in subsequent computations.

8 Further Comments

If the matrix A is balanced by F08NVF (ZGEBAL), then any eigenvectors computed subsequently are eigenvectors of the matrix A'' (see Section 3) and hence F08NWF (ZGEBAK) **must** then be called to transform them back to eigenvectors of A.

If the Schur vectors of A are required, then this routine must **not** be called with JOB = 'S' or 'B', because then the balancing transformation is not unitary. If this routine is called with JOB = 'P', then any Schur vectors computed subsequently are Schur vectors of the matrix A'', and F08NWF (ZGEBAK) **must** be called (with SIDE = 'R') to transform them back to Schur vectors of A.

The total number of real floating-point operations is approximately proportional to n^2.

The real analogue of this routine is F08NHF (DGEBAL).

9 Example

This example computes all the eigenvalues and right eigenvectors of the matrix A, where

$$A = \begin{pmatrix} 1.50-2.75i & 0.00+0.00i & 0.00+0.00i & 0.00+0.00i \\ -8.06-1.24i & -2.50-0.50i & 0.00+0.00i & -0.75+0.50i \\ -2.09+7.56i & 1.39+3.97i & -1.25+0.75i & -4.82-5.67i \\ 6.18+9.79i & -0.92-0.62i & 0.00+0.00i & -2.50-0.50i \end{pmatrix}.$$

The program first calls F08NVF (ZGEBAL) to balance the matrix; it then computes the Schur factorization of the balanced matrix, by reduction to Hessenberg form and the QR algorithm. Then it calls F08QXF (ZTREVC) to compute the right eigenvectors of the balanced matrix, and finally calls F08NWF (ZGEBAK) to transform the eigenvectors back to eigenvectors of the original matrix A.

9.1 Program Text

```
*     F08NVF Example Program Text
*     Mark 20 Revised. NAG Copyright 2001.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LDA, LDH, LWORK, LDVL, LDVR
      PARAMETER        (NMAX=8,LDA=NMAX,LDH=NMAX,LWORK=64*NMAX,LDVL=1,
     +                 LDVR=NMAX)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, IHI, ILO, INFO, J, M, N
*     .. Local Arrays ..
      COMPLEX *16      A(LDA,NMAX), H(LDH,NMAX), TAU(NMAX), VL(LDVL,1),
     +                 VR(LDVR,NMAX), W(NMAX), WORK(LWORK)
      DOUBLE PRECISION RWORK(NMAX), SCALE(NMAX)
      LOGICAL          SELECT(1)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F06TFF, X04DBF, ZGEBAK, ZGEBAL, ZGEHRD, ZHSEQR,
     +                 ZTREVC, ZUNGHR
*     .. Intrinsic Functions ..
```

```
      INTRINSIC        DBLE, AIMAG
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F08NVF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
*
*        Read A from data file
*
         READ (NIN,*) ((A(I,J),J=1,N),I=1,N)
*
*        Balance A
*
         CALL ZGEBAL('Both',N,A,LDA,ILO,IHI,SCALE,INFO)
*
*        Reduce A to upper Hessenberg form H = (Q**H)*A*Q
*
         CALL ZGEHRD(N,ILO,IHI,A,LDA,TAU,WORK,LWORK,INFO)
*
*        Copy A to H
*
         CALL F06TFF('General',N,N,A,LDA,H,LDH)
*
*        Copy A into VR
*
         CALL F06TFF('General',N,N,A,LDA,VR,LDVR)
*
*        Form Q explicitly, storing the result in VR
*
         CALL ZUNGHR(N,1,N,VR,LDVR,TAU,WORK,LWORK,INFO)
*
*        Calculate the eigenvalues and Schur factorization of A
*
         CALL ZHSEQR('Schur form','Vectors',N,ILO,IHI,H,LDH,W,VR,LDVR,
     +               WORK,LWORK,INFO)
*
         WRITE (NOUT,*)
         IF (INFO.GT.0) THEN
            WRITE (NOUT,*) 'Failure to converge.'
         ELSE
            WRITE (NOUT,*) 'Eigenvalues'
            WRITE (NOUT,99999) (' (',DBLE(W(I)),',',AIMAG(W(I)),')',I=1,
     +        N)
*
*           Calculate the eigenvectors of A, storing the result in VR
*
            CALL ZTREVC('Right','Backtransform',SELECT,N,H,LDH,VL,LDVL,
     +                  VR,LDVR,N,M,WORK,RWORK,INFO)
*
            CALL ZGEBAK('Both','Right',N,ILO,IHI,SCALE,M,VR,LDVR,INFO)
*
*           Print eigenvectors
*
            WRITE (NOUT,*)
            IFAIL = 0
*
            CALL X04DBF('General',' ',N,M,VR,LDVR,'Bracketed','F7.4',
     +                  'Contents of array VR','Integer',RLABS,
     +                  'Integer',CLABS,80,0,IFAIL)
*
         END IF
      END IF
      STOP
*
99999 FORMAT ((3X,4(A,F7.4,A,F7.4,A,:)))
      END
```

9.2 Program Data

```
F08NVF Example Program Data
  4                                                             :Value of N
 ( 1.50,-2.75) ( 0.00, 0.00) ( 0.00, 0.00) ( 0.00, 0.00)
 (-8.06,-1.24) (-2.50,-0.50) ( 0.00, 0.00) (-0.75, 0.50)
 (-2.09, 7.56) ( 1.39, 3.97) (-1.25, 0.75) (-4.82,-5.67)
 ( 6.18, 9.79) (-0.92,-0.62) ( 0.00, 0.00) (-2.50,-0.50)   :End of matrix A
```

9.3 Program Results

```
 F08NVF Example Program Results

 Eigenvalues
    (-1.2500, 0.7500) (-1.5000,-0.4975) (-3.5000,-0.5025) ( 1.5000,-2.7500)

 Contents of array VR
                   1                  2                  3                  4
 1  ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 0.1452, 0.0000)
 2  ( 0.0000, 0.0000) (-0.0616, 0.0413) ( 0.4613,-0.0000) (-0.2072,-0.2450)
 3  ( 1.0000, 0.0000) ( 0.6032,-0.3968) ( 0.2983, 0.7017) ( 0.7768, 0.2232)
 4  ( 0.0000, 0.0000) ( 0.0822, 0.0000) ( 0.4251, 0.2850) (-0.0119, 0.4372)
```