NAG Library Routine Document

F08NHF (DGEBAL)

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

F08NHF (DGEBAL) balances a real general matrix in order to improve the accuracy of computed eigenvalues and/or eigenvectors.

2 Specification

```
SUBROUTINE F08NHF (JOB, N, A, LDA, ILO, IHI, SCALE, INFO)

INTEGER              N, LDA, ILO, IHI, INFO
REAL (KIND=nag_wp) A(LDA,*), SCALE(N)
CHARACTER(1)         JOB
```

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

3 Description

F08NHF (DGEBAL) balances a real 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 (i.e., $\|A''_{22}\| < \|A'_{22}\|$) 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,$*$) – REAL (KIND=nag_wp) 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: A 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 F08NHF (DGEBAL) 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(N) – REAL (KIND=nag_wp) array *Output*

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

7 Accuracy

The errors are negligible.

8 Further Comments

If the matrix A is balanced by F08NHF (DGEBAL), then any eigenvectors computed subsequently are eigenvectors of the matrix A'' (see Section 3) and hence F08NJF (DGEBAK) **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 orthogonal. If this routine is called with JOB = 'P', then any Schur vectors computed subsequently are Schur vectors of the matrix A'', and F08NJF (DGEBAK) **must** be called (with SIDE = 'R') to transform them back to Schur vectors of A.

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

The complex analogue of this routine is F08NVF (ZGEBAL).

9 Example

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

$$A = \begin{pmatrix} 5.14 & 0.91 & 0.00 & -32.80 \\ 0.91 & 0.20 & 0.00 & 34.50 \\ 1.90 & 0.80 & -0.40 & -3.00 \\ -0.33 & 0.35 & 0.00 & 0.66 \end{pmatrix}.$$

The program first calls F08NHF (DGEBAL) 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 F08QKF (DTREVC) to compute the right eigenvectors of the balanced matrix, and finally calls F08NJF (DGEBAK) to transform the eigenvectors back to eigenvectors of the original matrix A.

9.1 Program Text

```
    Program f08nhfe

!     F08NHF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dgebak, dgebal, dgehrd, dhseqr, dorghr, dtrevc,   &
                             nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: firstnz
      Integer                          :: i, ifail, ihi, ilo, info, j, lda,    &
                                          ldh, ldvl, ldvr, lwork, m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), h(:,:), scale(:), tau(:),    &
                                          vl(:,:), vr(:,:), wi(:), work(:),    &
```

```
                                     wr(:)
      Logical                     :: select(1)
!     .. Executable Statements ..
      Write (nout,*) 'F08NHF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n
      ldvl = 1
      lda = n
      ldh = n
      ldvr = n
      lwork = 64*n
      Allocate (a(lda,n),h(ldh,n),scale(n),tau(n),vl(ldvl,1),vr(ldvr,n),wi(n), &
        work(lwork),wr(n))

!     Read A from data file
      Read (nin,*)(a(i,1:n),i=1,n)

!     Balance A
!     The NAG name equivalent of dgebal is f08nhf
      Call dgebal('Both',n,a,lda,ilo,ihi,scale,info)

!     Reduce A to upper Hessenberg form H = (Q**T)*A*Q
!     The NAG name equivalent of dgehrd is f08nef
      Call dgehrd(n,ilo,ihi,a,lda,tau,work,lwork,info)

!     Copy A to H and VR
      h(1:n,1:n) = a(1:n,1:n)
      vr(1:n,1:n) = a(1:n,1:n)

!     Form Q explicitly, storing the result in VR
!     The NAG name equivalent of dorghr is f08nff
      Call dorghr(n,1,n,vr,ldvr,tau,work,lwork,info)

!     Calculate the eigenvalues and Schur factorization of A
!     The NAG name equivalent of dhseqr is f08pef
      Call dhseqr('Schur form','Vectors',n,ilo,ihi,h,ldh,wr,wi,vr,ldvr,work, &
        lwork,info)

      Write (nout,*)
      If (info>0) Then
        Write (nout,*) 'Failure to converge.'
      Else
        Write (nout,*) 'Eigenvalues'
        Write (nout,99999)(' (',wr(i),',',wi(i),')',i=1,n)

!       Calculate the eigenvectors of A, storing the result in VR

!       The NAG name equivalent of dtrevc is f08qkf
        Call dtrevc('Right','Backtransform',select,n,h,ldh,vl,ldvl,vr,ldvr,n, &
          m,work,info)

!       The NAG name equivalent of dgebak is f08njf
        Call dgebak('Both','Right',n,ilo,ihi,scale,m,vr,ldvr,info)

!       Print eigenvectors

        Write (nout,*)
        Flush (nout)

!       Normalize the eigenvectors
        Do i = 1, m
          Do j = n, 1, -1
            If (vr(j,i)/=(0._nag_wp,0._nag_wp)) Then
              firstnz = vr(j,i)
            End If
          End Do
          vr(1:n,i) = vr(1:n,i)/firstnz
        End Do

!       ifail: behaviour on error exit
```

```
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04caf('General',' ',n,m,vr,ldvr,'Contents of array VR',ifail)

      End If

99999 Format (1X,A,F8.4,A,F8.4,A)
    End Program f08nhfe
```

9.2 Program Data

```
F08NHF Example Program Data
  4                         :Value of N
 5.14  0.91   0.00 -32.80
 0.91  0.20   0.00  34.50
 1.90  0.80  -0.40  -3.00
-0.33  0.35   0.00   0.66   :End of matrix A
```

9.3 Program Results

```
 F08NHF Example Program Results

 Eigenvalues
  ( -0.4000,  0.0000)
  ( -4.0208,  0.0000)
  (  3.0136,  0.0000)
  (  7.0072,  0.0000)

 Contents of array VR
            1          2          3          4
 1      0.0000     1.0000     1.0000     1.0000
 2      0.0000    -2.0366     1.6950    -0.1802
 3      1.0000     0.1098     0.8555     0.2621
 4      0.0000     0.2228     0.1119    -0.0619
```