F02HAF – NAG Fortran Library Routine Document

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

F02HAF computes all the eigenvalues, and optionally all the eigenvectors, of a complex Hermitian matrix.

2 Specification

```
    SUBROUTINE F02HAF(JOB, UPLO, N, A, LDA, W, RWORK, WORK, LWORK,
   1                  IFAIL)
    INTEGER           N, LDA, LWORK, IFAIL
    real              W(*), RWORK(*)
    complex           A(LDA,*), WORK(LWORK)
    CHARACTER*1       JOB, UPLO
```

3 Description

This routine computes all the eigenvalues, and optionally all the eigenvectors, of a complex Hermitian matrix A:

$$Az_i = \lambda_i z_i \text{ for } i = 1, 2, \ldots, n.$$

In other words, it computes the spectral factorization of A:

$$A = Z \Lambda Z^H,$$

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

4 References

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

[2] Parlett B N (1980) *The Symmetric Eigenvalue Problem* Prentice–Hall

5 Parameters

1: JOB — CHARACTER*1 *Input*

On entry: indicates whether eigenvectors are to be computed as follows:

if JOB = 'N', then only eigenvalues are computed;
if JOB = 'V', then eigenvalues and eigenvectors are computed.

Constraint: JOB = 'N' or 'V'.

2: UPLO — CHARACTER*1 *Input*

On entry: indicates whether the upper or lower triangular part of A is stored as follows:

if UPLO = 'U', then the upper triangular part of A is stored;
if UPLO = 'L', then the lower triangular part of A is stored.

Constraint: UPLO = 'U' or 'L'.

3: N — INTEGER *Input*

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

Constraint: N $\geq$ 0.

4: A(LDA,∗) — ***complex*** array *Input/Output*

Note: the second dimension of the array A must be at least max(1,N).

On entry: the n by n Hermitian matrix A. If UPLO = 'U', the upper triangle of A must be stored and the elements of the array below the diagonal need not be set; if UPLO = 'L', the lower triangle of A must be stored and the elements of the array above the diagonal need not be set.

On exit: If JOB = 'V', A contains the unitary matrix Z of eigenvectors, with the ith column holding the eigenvector z_i associated with the eigenvalue λ_i (stored in W(i)). If JOB = 'N', the original contents of A are overwritten.

5: LDA — INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F02HAF is called.

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

6: W(∗) — ***real*** array *Output*

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

On exit: the eigenvalues in ascending order.

7: RWORK(∗) — ***real*** array *Workspace*

Note: the dimension of the array RWORK must be at least max(1,3×N).

8: WORK(LWORK) — ***complex*** array *Workspace*

9: LWORK — INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F02HAF is called. On some high-performance computers, increasing the dimension of WORK will enable the routine to run faster; a value of 64 × N should allow near-optimal performance on almost all machines.

Constraint: LWORK $\geq$ max(1,2×N).

10: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

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

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, JOB ≠ 'N' or 'V',
or UPLO ≠ 'U' or 'L',
or N < 0,
or LDA < max(1,N),
or LWORK < max(1,2×N).

IFAIL = 2

The QR algorithm failed to compute all the eigenvalues.

IFAIL = 3

For some i, A(i,i) has a non-zero imaginary part (thus A is not Hermitian).

7 Accuracy

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

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

where $c(n)$ is a modestly increasing function of n, and ϵ is the ***machine precision***.

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

$$\theta(\tilde{z}_i, z_i) \leq \frac{c(n)\epsilon\|A\|_2}{\min_{i \neq j} |\lambda_i - \lambda_j|}.$$

Thus the accuracy of a computed eigenvector depends on the gap between its eigenvalue and all the other eigenvalues.

8 Further Comments

The routine calls routines from LAPACK in the F08 Chapter Introduction. It first reduces A to real tridiagonal form T, using a unitary similarity transformation: $A = QTQ^H$. If only eigenvalues are required, the routine uses a root-free variant of the symmetric tridiagonal QR algorithm. If eigenvectors are required, the routine first forms the unitary matrix Q that was used in the reduction to tridiagonal form; it then uses the symmetric tridiagonal QR algorithm to reduce T to Λ, using a real orthogonal transformation: $T = S\Lambda S^T$; and at the same time accumulates the matrix $Z = QS$.

Each eigenvector z is normalized so that $\|z\|_2 = 1$ and the element of largest absolute value is real and positive.

The time taken by the routine is approximately proportional to n^3.

9 Example

To compute all the eigenvalues and eigenvectors of the matrix A, where

$$A = \begin{pmatrix} -2.28+0.00i & 1.78-2.03i & 2.26+0.10i & -0.12+2.53i \\ 1.78+2.03i & -1.12+0.00i & 0.01+0.43i & -1.07+0.86i \\ 2.26-0.10i & 0.01-0.43i & -0.37+0.00i & 2.31-0.92i \\ -0.12-2.53i & -1.07-0.86i & 2.31+0.92i & -0.73+0.00i \end{pmatrix}.$$

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F02HAF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LDA, LWORK
      PARAMETER        (NMAX=8,LDA=NMAX,LWORK=64*NMAX)
*     .. Local Scalars ..
```

```
      INTEGER          I, IFAIL, J, N
      CHARACTER        UPLO
*     .. Local Arrays ..
      complex          A(LDA,NMAX), WORK(LWORK)
      real             RWORK(3*NMAX), W(NMAX)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F02HAF, X04DBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F02HAF 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,*) UPLO
         IF (UPLO.EQ.'U') THEN
            READ (NIN,*) ((A(I,J),J=I,N),I=1,N)
         ELSE IF (UPLO.EQ.'L') THEN
            READ (NIN,*) ((A(I,J),J=1,I),I=1,N)
         END IF
*
*        Compute eigenvalues and eigenvectors
*
         IFAIL = 0
*
         CALL F02HAF('Vectors',UPLO,N,A,LDA,W,RWORK,WORK,LWORK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Eigenvalues'
         WRITE (NOUT,99999) (W(I),I=1,N)
         WRITE (NOUT,*)
*
         CALL X04DBF('General',' ',N,N,A,LDA,'Bracketed','F7.4',
     +               'Eigenvectors','Integer',RLABS,'Integer',CLABS,80,
     +               0,IFAIL)
*
      END IF
      STOP
*
99999 FORMAT (3X,4(F12.4,6X))
      END
```

9.2 Program Data

```
F02HAF Example Program Data
  4                                                             :Value of N
  'L'                                                           :Value of UPLO
 (-2.28, 0.00)
 ( 1.78, 2.03) (-1.12, 0.00)
 ( 2.26,-0.10) ( 0.01,-0.43) (-0.37, 0.00)
 (-0.12,-2.53) (-1.07,-0.86) ( 2.31, 0.92) (-0.73, 0.00)   :End of matrix A
```

9.3 Program Results

```
F02HAF Example Program Results

Eigenvalues
       -6.0002           -3.0030            0.5036            3.9996

Eigenvectors
                   1                 2                 3                 4
1  ( 0.7299, 0.0000) (-0.2120, 0.1497) ( 0.1000,-0.3570) ( 0.1991, 0.4720)
2  (-0.1663,-0.2061) ( 0.7307, 0.0000) ( 0.2863,-0.3353) (-0.2467, 0.3751)
3  (-0.4165,-0.1417) (-0.3291, 0.0479) ( 0.6890, 0.0000) ( 0.4468, 0.1466)
4  ( 0.1743, 0.4162) ( 0.5200, 0.1329) ( 0.0662, 0.4347) ( 0.5612, 0.0000)
```