NAG Library Routine Document

F02XUF

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

F02XUF returns all, or part, of the singular value decomposition of a complex upper triangular matrix.

2 Specification

```
SUBROUTINE F02XUF (N, A, LDA, NCOLB, B, LDB, WANTQ, Q, LDQ, SV, WANTP,         &
                   RWORK, CWORK, IFAIL)
INTEGER               N, LDA, NCOLB, LDB, LDQ, IFAIL
REAL (KIND=nag_wp)    SV(N), RWORK(*)
COMPLEX (KIND=nag_wp) A(LDA,*), B(LDB,*), Q(LDQ,*), CWORK(max(1,N-1))
LOGICAL               WANTQ, WANTP
```

3 Description

The n by n upper triangular matrix R is factorized as

$$R = QSP^{\mathrm{H}},$$

where Q and P are n by n unitary matrices and S is an n by n diagonal matrix with real non-negative diagonal elements, $sv_1, sv_2, \ldots, sv_n$, ordered such that

$$sv_1 \geq sv_2 \geq \cdots \geq sv_n \geq 0.$$

The columns of Q are the left-hand singular vectors of R, the diagonal elements of S are the singular values of R and the columns of P are the right-hand singular vectors of R.

Either or both of Q and P^{H} may be requested and the matrix C given by

$$C = Q^{\mathrm{H}}B,$$

where B is an n by *ncolb* given matrix, may also be requested.

F02XUF obtains the singular value decomposition by first reducing R to bidiagonal form by means of Givens plane rotations and then using the QR algorithm to obtain the singular value decomposition of the bidiagonal form.

Good background descriptions to the singular value decomposition are given in Dongarra *et al.* (1979), Hammarling (1985) and Wilkinson (1978).

Note that if K is any unitary diagonal matrix so that

$$KK^{\mathrm{H}} = I,$$

then

$$A = (QK)S(PK)^H$$

is also a singular value decomposition of A.

4 References

Dongarra J J, Moler C B, Bunch J R and Stewart G W (1979) *LINPACK Users' Guide* SIAM, Philadelphia

Hammarling S (1985) The singular value decomposition in multivariate statistics *SIGNUM Newsl.* **20(3)** 2–25

Wilkinson J H (1978) Singular Value Decomposition – Basic Aspects *Numerical Software – Needs and Availability* (ed D A H Jacobs) Academic Press

5 Parameters

1: N – INTEGER *Input*

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

If $\mathrm{N} = 0$, an immediate return is effected.

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

2: A(LDA,$*$) – COMPLEX (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 leading n by n upper triangular part of the array A must contain the upper triangular matrix R.

On exit: if WANTP = .TRUE., the n by n part of A will contain the n by n unitary matrix P^{H}, otherwise the n by n upper triangular part of A is used as internal workspace, but the strictly lower triangular part of A is not referenced.

3: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F02XUF is called.

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

4: NCOLB – INTEGER *Input*

On entry: *ncolb*, the number of columns of the matrix B.

If $\mathrm{NCOLB} = 0$, the array B is not referenced.

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

5: B(LDB,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: if $\mathrm{NCOLB} > 0$, the leading n by *ncolb* part of the array B must contain the matrix to be transformed.

On exit: is overwritten by the n by *ncolb* matrix $Q^{\mathrm{H}}B$.

6: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F02XUF is called.

Constraints:

if $\mathrm{NCOLB} > 0$, $\mathrm{LDB} \geq \max(1, \mathrm{N})$;
otherwise $\mathrm{LDB} \geq 1$.

7: WANTQ – LOGICAL *Input*

On entry: must be .TRUE. if the matrix Q is required.

If WANTQ = .FALSE. then the array Q is not referenced.

8: Q(LDQ,$*$) – COMPLEX (KIND=nag_wp) array *Output*

Note: the second dimension of the array Q must be at least $\max(1, N)$ if WANTQ = .TRUE., and at least 1 otherwise.

On exit: if WANTQ = .TRUE., the leading n by n part of the array Q will contain the unitary matrix Q. Otherwise the array Q is not referenced.

9: LDQ – INTEGER *Input*

On entry: the first dimension of the array Q as declared in the (sub)program from which F02XUF is called.

Constraints:

if WANTQ = .TRUE., LDQ $\geq \max(1, N)$;
otherwise LDQ ≥ 1.

10: SV(N) – REAL (KIND=nag_wp) array *Output*

On exit: the n diagonal elements of the matrix S.

11: WANTP – LOGICAL *Input*

On entry: must be .TRUE. if the matrix P^{H} is required, in which case P^{H} is returned in the array A, otherwise WANTP must be .FALSE..

12: RWORK($*$) – REAL (KIND=nag_wp) array *Output*

Note: the dimension of the array RWORK must be at least $\max(1, 2 \times (N - 1))$ if NCOLB $= 0$ and WANTQ = .FALSE. and WANTP = .FALSE., $\max(1, 3 \times (N - 1))$ if NCOLB $= 0$ and WANTQ = .FALSE. and WANTP = .TRUE. or NCOLB > 0 and WANTP = .FALSE. or WANTQ = .TRUE. and WANTP = .FALSE., and at least $\max(1, 5 \times (N - 1))$ otherwise.

On exit: RWORK(N) contains the total number of iterations taken by the QR algorithm.

The rest of the array is used as workspace.

13: CWORK($\max(1, N - 1)$) – COMPLEX (KIND=nag_wp) array *Workspace*

14: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (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 or warnings detected by the routine:

IFAIL $= -1$

On entry, N < 0,
or LDA $<$ N,
or NCOLB < 0,
or LDB $<$ N and NCOLB > 0,
or LDQ $<$ N and WANTQ $=$ .TRUE.

IFAIL > 0

The QR algorithm has failed to converge in $50 \times$ N iterations. In this case $\mathrm{SV}(1), \mathrm{SV}(2), \ldots, \mathrm{SV}(\mathrm{IFAIL})$ may not have been found correctly and the remaining singular values may not be the smallest. The matrix R will nevertheless have been factorized as $R = QEP^{\mathrm{H}}$, where E is a bidiagonal matrix with $\mathrm{SV}(1), \mathrm{SV}(2), \ldots, \mathrm{SV}(n)$ as the diagonal elements and $\mathrm{RWORK}(1), \mathrm{RWORK}(2), \ldots, \mathrm{RWORK}(n-1)$ as the superdiagonal elements.

This failure is not likely to occur.

7 Accuracy

The computed factors Q, S and P satisfy the relation

$$QSP^{\mathrm{H}} = A + E,$$

where

$$\|E\| \leq c\epsilon\|A\|,$$

ϵ is the ***machine precision***, c is a modest function of n and $\|.\|$ denotes the spectral (two) norm. Note that $\|A\| = sv_1$.

8 Further Comments

For given values of NCOLB, WANTQ and WANTP, the number of floating point operations required is approximately proportional to n^3.

Following the use of this routine the rank of R may be estimated by a call to the INTEGER FUNCTION F06KLF. The statement

```
IRANK = F06KLF(N,SV,1,TOL)
```

returns the value $(k-1)$ in *IRANK*, where k is the smallest integer for which $sv(k) < tol \times sv(1)$, where *tol* is the tolerance supplied in TOL, so that *IRANK* is an estimate of the rank of S and thus also of R. If TOL is supplied as negative then the ***machine precision*** is used in place of TOL.

9 Example

This example finds the singular value decomposition of the 3 by 3 upper triangular matrix

$$A = \begin{pmatrix} 1 & 1+i & 1+i \\ 0 & -2 & -1-i \\ 0 & 0 & -3 \end{pmatrix}$$

together with the vector $Q^{\mathrm{H}}b$ for the vector

$$b = \begin{pmatrix} 1+1i \\ -1 \\ -1+1i \end{pmatrix}.$$

9.1 Program Text

```
    Program f02xufe

!     F02XUF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f02xuf, nag_wp, x04dbf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                           :: i, ifail, lcwork, lda, ldb, ldq,     &
                                           lrwork, n, ncolb
      Logical                           :: wantp, wantq
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:), cwork(:), q(:,:)
      Real (Kind=nag_wp), Allocatable  :: rwork(:), sv(:)
      Character (1)                     :: clabs(1), rlabs(1)
!     .. Intrinsic Procedures ..
      Intrinsic                         :: conjg
!     .. Executable Statements ..
      Write (nout,*) 'F02XUF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, ncolb
      lcwork = n - 1
      lda = n
      ldb = n
      ldq = n
      lrwork = 5*(n-1)
      Allocate (a(lda,n),b(ldb),cwork(lcwork),q(ldq,n),rwork(lrwork),sv(n))
      Read (nin,*)(a(i,i:n),i=1,n)
      Read (nin,*) b(1:n)
!     Find the SVD of A.
      wantq = .True.
      wantp = .True.

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f02xuf(n,a,lda,ncolb,b,ldb,wantq,q,ldq,sv,wantp,rwork,cwork,ifail)

      Write (nout,*) 'Singular value decomposition of A'
      Write (nout,*)
      Write (nout,*) 'Singular values'
      Write (nout,99999) sv(1:n)
      Write (nout,*)
      Flush (nout)

      ifail = 0
      Call x04dbf('General',' ',n,n,q,ldq,'Bracketed','F7.4', &
        'Left-hand singular vectors, by column','N',rlabs,'N',clabs,80,0, &
        ifail)

      Write (nout,*)
      Write (nout,*) 'Right-hand singular vectors, by column'
      Do i = 1, n
        Write (nout,99998) conjg(a(1:n,i))
```

```
      End Do
      Write (nout,*)
      Write (nout,*) 'Vector conjg( Q'' )*B'
      Write (nout,99998) b(1:n)

99999 Format (1X,3F9.4)
99998 Format (3X,3('(',F7.4,',',F8.4,') ':))
    End Program f02xufe
```

9.2 Program Data

```
F02XUF Example Program Data

   3     1                                    : n, ncolb

  (1.0,  0.0)   ( 1.0,  1.0)   ( 1.0,  1.0)
                (-2.0,  0.0)   (-1.0, -1.0)
                               (-3.0,  0.0)    : matrix A

  (1.0,  1.0)   (-1.0,  0.0)   (-1.0,  1.0)    : vector B
```

9.3 Program Results

```
 F02XUF Example Program Results

 Singular value decomposition of A

 Singular values
    3.9263   2.0000   0.7641

 Left-hand singular vectors, by column
   (-0.5005,-0.0000) (-0.4529, 0.0000) ( 0.7378, 0.0000)
   ( 0.5152,-0.1514) ( 0.1132,-0.5661) ( 0.4190,-0.4502)
   ( 0.4041,-0.5457) ( 0.0000, 0.6794) ( 0.2741, 0.0468)

 Right-hand singular vectors, by column
   (-0.1275, -0.0000) (-0.2265, -0.0000) ( 0.9656, -0.0000)
   (-0.3899,  0.2046) (-0.3397,  0.7926) (-0.1311,  0.2129)
   (-0.5289,  0.7142) (-0.0000, -0.4529) (-0.0698, -0.0119)

 Vector conjg( Q' )*B
   (-1.9656, -0.7935) ( 0.1132, -0.3397) ( 0.0915,  0.6086)
```