NAG Library Routine Document

F01BUF

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

F01BUF performs a $ULDL^{\mathrm{T}}U^{\mathrm{T}}$ decomposition of a real symmetric positive definite band matrix.

2 Specification

```
SUBROUTINE F01BUF (N, M1, K, A, LDA, W, IFAIL)
INTEGER            N, M1, K, LDA, IFAIL
REAL (KIND=nag_wp) A(LDA,N), W(M1)
```

3 Description

The symmetric positive definite matrix A, of order n and bandwidth $2m+1$, is divided into the leading principal sub-matrix of order k and its complement, where $m \leq k \leq n$. A UDU^{T} decomposition of the latter and an LDL^{T} decomposition of the former are obtained by means of a sequence of elementary transformations, where U is unit upper triangular, L is unit lower triangular and D is diagonal. Thus if $k = n$, an LDL^{T} decomposition of A is obtained.

This routine is specifically designed to precede F01BVF for the transformation of the symmetric-definite eigenproblem $Ax = \lambda Bx$ by the method of Crawford where A and B are of band form. In this context, k is chosen to be close to $n/2$ and the decomposition is applied to the matrix B.

4 References

Wilkinson J H (1965) *The Algebraic Eigenvalue Problem* Oxford University Press, Oxford

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation II, Linear Algebra* Springer–Verlag

5 Parameters

1: N – INTEGER *Input*

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

2: M1 – INTEGER *Input*

On entry: $m+1$, where m is the number of nonzero superdiagonals in A. Normally M1 $\ll$ N.

3: K – INTEGER *Input*

On entry: k, the change-over point in the decomposition.

Constraint: M1 $-1 \leq$ K $\leq$ N.

4: A(LDA, N) – REAL (KIND=nag_wp) array *Input/Output*

On entry: the upper triangle of the n by n symmetric band matrix A, with the diagonal of the matrix stored in the $(m+1)$th row of the array, and the m superdiagonals within the band stored in the first m rows of the array. Each column of the matrix is stored in the corresponding column of the array. For example, if $n = 6$ and $m = 2$, the storage scheme is

$$\begin{array}{cccccc} * & * & a_{13} & a_{24} & a_{35} & a_{46} \\ * & a_{12} & a_{23} & a_{34} & a_{45} & a_{56} \\ a_{11} & a_{22} & a_{33} & a_{44} & a_{55} & a_{66} \end{array}$$

Elements in the top left corner of the array are not used. The following code assigns the matrix elements within the band to the correct elements of the array:

```
      DO 20 J = 1, N
         DO 10 I = MAX(1,J-M1+1), J
            A(I-J+M1,J) = matrix(I,J)
10       CONTINUE
20    CONTINUE
```

On exit: A is overwritten by the corresponding elements of L, D and U.

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01BUF is called.

Constraint: LDA $\geq$ M1.

6: W(M1) – REAL (KIND=nag_wp) array *Workspace*

7: 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, K $<$ M1 or K $>$ N.

IFAIL $= 2$
IFAIL $= 3$

The matrix A is not positive definite, perhaps as a result of rounding errors, giving an element of D which is zero or negative. IFAIL $= 3$ when the failure occurs in the leading principal sub-matrix of order K and IFAIL $= 2$ when it occurs in the complement.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

The Cholesky decomposition of a positive definite matrix is known for its remarkable numerical stability (see Wilkinson (1965)). The computed U, L and D satisfy the relation $ULDL^{\mathrm{T}}U^{\mathrm{T}} = A + E$ where the 2-norms of A and E are related by $\|E\| \leq c(m+1)^2\epsilon\|A\|$ where c is a constant of order unity and ϵ is the ***machine precision***. In practice, the error is usually appreciably smaller than this.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by F01BUF is approximately proportional to $nm^2 + 3nm$.

This routine is specifically designed for use as the first stage in the solution of the generalized symmetric eigenproblem $Ax = \lambda Bx$ by Crawford's method which preserves band form in the transformation to a similar standard problem. In this context, for maximum efficiency, k should be chosen as the multiple of m nearest to $n/2$.

The matrix U is such that $U^{-1}AU^{-\mathrm{T}}$ is diagonal in its last $n-k$ rows and columns, L is such that $L^{-1}U^{-1}AU^{-\mathrm{T}}L^{-\mathrm{T}} = D$ and D is diagonal. To find U, L and D where $A = ULDL^{\mathrm{T}}U^{\mathrm{T}}$ requires $nm(m+3)/2 - m(m+1)(m+2)/3$ multiplications and divisions which, is independent of k.

10 Example

This example finds a $ULDL^{\mathrm{T}}U^{\mathrm{T}}$ decomposition of the real symmetric positive definite matrix

$$\begin{pmatrix} 3 & -9 & 6 & & & & \\ -9 & 31 & -2 & -4 & & & \\ 6 & -2 & 123 & -66 & 15 & & \\ & -4 & -66 & 145 & -24 & 4 & \\ & & 15 & -24 & 61 & -74 & -18 \\ & & & 4 & -74 & 98 & 24 \\ & & & & -18 & 24 & 6 \end{pmatrix}.$$

10.1 Program Text

```
    Program f01bufe

!     F01BUF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: f01buf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
!     .. Local Scalars ..
```

```
      Integer                          :: i, ifail, j, k, lda, m, m1, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), w(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max
!     .. Executable Statements ..
      Write (nout,*) 'F01BUF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, m1
      lda = m1
      Allocate (a(lda,n),w(m1))
      Read (nin,*)((a(j,i),j=max(1,m1+1-i),m1),i=1,n)
      m = m1 - 1
      k = ((n+m)/(2*m))*m

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f01buf(n,m1,k,a,lda,w,ifail)

      Write (nout,*)
      Write (nout,*) 'Computed upper triangular matrix'
      Write (nout,*)
      Do i = 1, n
        Write (nout,99999)(a(j,i),j=max(1,m1+1-i),m1)
      End Do

99999 Format (1X,8F9.4)
    End Program f01bufe
```

10.2 Program Data

```
F01BUF Example Program Data
  7  3                 : n, m1
     3
    -9    31
     6    -2   123
    -4   -66   145
    15   -24    61
     4   -74    98
   -18    24     6     : a
```

10.3 Program Results

```
 F01BUF Example Program Results

 Computed upper triangular matrix

    3.0000
   -3.0000   4.0000
    2.0000   4.0000   2.0000
   -1.0000   5.0000   3.0000
    3.0000  -4.0000   5.0000
    2.0000  -1.0000   2.0000
   -3.0000   4.0000   6.0000
```