NAG Library Routine Document

F01ZDF

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

F01ZDF copies a complex band matrix stored in a packed array into an unpacked array, or vice versa.

2 Specification

```
SUBROUTINE F01ZDF (JOB, M, N, KL, KU, A, LDA, B, LDB, IFAIL)
INTEGER                 M, N, KL, KU, LDA, LDB, IFAIL
COMPLEX (KIND=nag_wp) A(LDA,N), B(LDB,*)
CHARACTER(1)            JOB
```

3 Description

F01ZDF unpacks a band matrix that is stored in a packed array, or packs a band matrix that is stored in an unpacked array. The band matrix has m rows, n columns, k_l nonzero subdiagonals, and k_u nonzero superdiagonals. This routine is intended for possible use in conjunction with routines from Chapters F06, F07 and F08, where routines that use band matrices store them in the packed form described below.

4 References

None.

5 Arguments

1: JOB – CHARACTER(1) *Input*

On entry: specifies whether the band matrix is to be packed or unpacked.

JOB = 'P' (Pack)
The band matrix is to be packed into array B.

JOB = 'U' (Unpack)
The band matrix is to be unpacked into array A.

Constraint: JOB = 'P' or 'U'.

2: M – INTEGER *Input*

3: N – INTEGER *Input*

On entry: m and n, the number of rows and columns of the band matrix, respectively.

Constraints:

M > 0;
N > 0.

4: KL – INTEGER *Input*

On entry: k_l, the number of subdiagonals of the band matrix.

Constraint: KL ≥ 0.

5: KU – INTEGER *Input*

On entry: k_u, the number of superdiagonals of the band matrix.

Constraint: KU ≥ 0.

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

On entry: if JOB = 'P', then the leading m by n part of A must contain the band matrix stored in unpacked form. Elements of the array that lie outside the banded part of the matrix are not referenced and need not be assigned.

On exit: if JOB = 'U', then the leading m by n part of A contains the band matrix stored in unpacked form. Elements of the leading m by n part of A that are not within the banded part of the matrix are assigned the value zero.

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01ZDF is called.

Constraint: LDA $\geq$ M.

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

Note: the second dimension of the array B must be at least min(M + KU, N).

On entry: if JOB = 'U', then B must contain the band matrix in packed form, in the leading $(k_l + k_u + 1)$ by $\min(m + k_u, n)$ part of the array. The matrix is packed column by column, with the leading diagonal of the matrix in row $(k_u + 1)$ of B, the first superdiagonal starting at position 2 in row k_u, the first subdiagonal starting at position 1 in row $(k_u + 2)$, and so on. Elements of B that are not needed to store the band matrix, for instance the leading k_u by k_u triangle, are not referenced and need not be assigned.

On exit: if JOB = 'P', then B contains the band matrix stored in packed form. Elements of B that are not needed to store the band matrix are not referenced.

9: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F01ZDF is called.

Constraint: LDB $\geq$ (KL + KU + 1).

10: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 argument, 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, JOB $\neq$ 'P' or 'U'.

IFAIL $= 2$

On entry, KL < 0.

IFAIL $= 3$

On entry, KU < 0.

IFAIL $= 4$

On entry, LDA $<$ M.

IFAIL $= 5$

On entry, LDB $<$ KL $+$ KU $+ 1$.

IFAIL $= 6$

On entry, M < 1,
or N < 1.

IFAIL $= -99$

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

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

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

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

F01ZDF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example reads a matrix A in unpacked form, and copies it to the packed matrix B.

10.1 Program Text

```
    Program f01zdfe

!     F01ZDF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f01zdf, nag_wp, x04daf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, kl, ku, lda, ldb, m, n
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'F01ZDF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Write (nout,*)
      Flush (nout)
      Read (nin,*) m, n, kl, ku
      lda = n
      ldb = lda
      Allocate (a(lda,n),b(ldb,n))
!     Read a banded matrix of size m by n. kl is the number of
!     subdiagonals, ku the number of superdiagonals.
      Do i = 1, n
        Read (nin,*) a(i,1:n)
      End Do
!     Clear the packed matrix array B, so that no elements are
!     unassigned when we print B later.
      b(1:(kl+ku+1),1:n) = (0.0E+0_nag_wp,0.0E+0_nag_wp)

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
!     Print the unpacked matrix
      Call x04daf('G','X',n,n,a,lda,'Unpacked Matrix A:',ifail)

      Write (nout,*)
      Flush (nout)

!     Convert to packed matrix form
      ifail = 0
      Call f01zdf('Pack',m,n,kl,ku,a,lda,b,ldb,ifail)

!     Print the packed matrix
      ifail = 0
      Call x04daf('G','X',kl+ku+1,n,b,ldb,'Packed Matrix B:',ifail)

    End Program f01zdfe
```

10.2 Program Data

```
F01ZDF Example Program Data
 5  5  1  1                                                    : m, n, kl, ku
 (1.1,-1.1)  (1.2,-1.2)  (0.0, 0.0)  (0.0, 0.0)  (0.0, 0.0)  : Unpacked Matrix A
 (2.1,-2.1)  (2.2,-2.2)  (2.3,-2.3)  (0.0, 0.0)  (0.0, 0.0)
 (0.0, 0.0)  (3.2,-3.2)  (3.3,-3.3)  (3.4,-3.4)  (0.0, 0.0)
 (0.0, 0.0)  (0.0, 0.0)  (4.3,-4.3)  (4.4,-4.4)  (4.5,-4.5)
 (0.0, 0.0)  (0.0, 0.0)  (0.0, 0.0)  (5.4,-5.4)  (5.5,-5.5)
```

10.3 Program Results

```
 F01ZDF Example Program Results

 Unpacked Matrix A:
             1          2          3          4          5
 1      1.1000     1.2000     0.0000     0.0000     0.0000
       -1.1000    -1.2000     0.0000     0.0000     0.0000

 2      2.1000     2.2000     2.3000     0.0000     0.0000
       -2.1000    -2.2000    -2.3000     0.0000     0.0000

 3      0.0000     3.2000     3.3000     3.4000     0.0000
        0.0000    -3.2000    -3.3000    -3.4000     0.0000

 4      0.0000     0.0000     4.3000     4.4000     4.5000
        0.0000     0.0000    -4.3000    -4.4000    -4.5000

 5      0.0000     0.0000     0.0000     5.4000     5.5000
        0.0000     0.0000     0.0000    -5.4000    -5.5000

 Packed Matrix B:
             1          2          3          4          5
 1      0.0000     1.2000     2.3000     3.4000     4.5000
        0.0000    -1.2000    -2.3000    -3.4000    -4.5000

 2      1.1000     2.2000     3.3000     4.4000     5.5000
       -1.1000    -2.2000    -3.3000    -4.4000    -5.5000

 3      2.1000     3.2000     4.3000     5.4000     0.0000
       -2.1000    -3.2000    -4.3000    -5.4000     0.0000
```