

NAG Library Routine Document

F01VHF (ZTFTTR)

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

F01VHF (ZTFTTR) unpacks a complex triangular matrix, stored in a Rectangular Full Packed (RFP) format array, to a full format array.

2 Specification

```
SUBROUTINE F01VHF (TRANSR, UPLO, N, AR, A, LDA, INFO)
  INTEGER          N, LDA, INFO
  COMPLEX (KIND=nag_wp) AR(N*(N+1)/2), A(LDA,*)
  CHARACTER(1)     TRANSR, UPLO
```

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

3 Description

F01VHF (ZTFTTR) unpacks a complex n by n triangular matrix A , stored in RFP format to conventional storage in a full format array. This routine is intended for possible use in conjunction with routines from Chapters F06, F07 and F16 where some routines that use triangular matrices store them in RFP format. The RFP storage format is described in Section 3.3.3 in the F07 Chapter Introduction.

4 References

Gustavson F G, Waśniewski J, Dongarra J J and Langou J (2010) Rectangular full packed format for Cholesky's algorithm: factorization, solution, and inversion *ACM Trans. Math. Software* **37**, 2

5 Arguments

- 1: TRANSR – CHARACTER(1) *Input*
On entry: specifies whether the normal RFP representation of A or its conjugate transpose is stored.
 TRANSR = 'N'
 The RFP representation of the matrix A is stored.
 TRANSR = 'C'
 The conjugate transpose of the RFP representation of the matrix A is stored.
Constraint: TRANSR = 'N' or 'C'.
- 2: UPLO – CHARACTER(1) *Input*
On entry: specifies whether A is upper or lower triangular.
 UPLO = 'U'
 A is upper triangular.
 UPLO = 'L'
 A is lower triangular.
Constraint: UPLO = 'U' or 'L'.

- 3: N – INTEGER *Input*
On entry: n , the order of the matrix A .
Constraint: $N \geq 0$.
- 4: AR($N \times (N + 1)/2$) – COMPLEX (KIND=nag_wp) array *Input*
On entry: the upper or lower n by n triangular matrix A (as specified by UPLO) in either normal or transposed RFP format (as specified by TRANSR). The storage format is described in Section 3.3.3 in the F07 Chapter Introduction.
- 5: A(LDA,*) – COMPLEX (KIND=nag_wp) array *Output*
Note: the second dimension of the array A must be at least N .
On exit: the triangular matrix A .
 If UPLO = 'U', A is upper triangular and the elements of the array below the diagonal are not referenced.
 If UPLO = 'L', A is lower triangular and the elements of the array above the diagonal are not referenced.
- 6: LDA – INTEGER *Input*
On entry: the first dimension of the array A as declared in the (sub)program from which F01VHF (ZTFTTR) is called.
Constraint: $LDA \geq \max(1, N)$.
- 7: INFO – INTEGER *Output*
On exit: INFO = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

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

Not applicable.

8 Parallelism and Performance

F01VHF (ZTFTTR) is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example reads in a triangular matrix in RFP format and unpacks it to full format.

10.1 Program Text

Program f01vhfe

```

!      F01VHF Example Program Text
!
!      Mark 26 Release. NAG Copyright 2016.
!
!      .. Use Statements ..
!      Use nag_library, Only: nag_wp, x04dbf, ztfttr
!      .. Implicit None Statement ..
!      Implicit None
!      .. Parameters ..
!      Integer, Parameter
!      Integer, Parameter          :: incl = 1, indent = 0, ncols = 80,      &
!                                  nin = 5, nout = 6
!      Character (1), Parameter
!      Character (1), Parameter    :: brac = 'B', diag = 'N',                &
!                                  intlabel = 'I', matrix = 'G',                &
!                                  nolabel = 'N'
!      Character (4), Parameter
!      Character (4), Parameter    :: form = 'F5.2'
!      .. Local Scalars ..
!      Integer
!      Integer                     :: i, ifail, info, k, lar1, lda, lenar, &
!                                  n, q
!      Character (21)
!      Character (1)               :: title
!      Character (1)               :: transr, uplo
!      .. Local Arrays ..
!      Complex (Kind=nag_wp), Allocatable :: a(:, :), ar(:)
!      Character (1)                 :: clabs(1), rlabs(1)
!      .. Executable Statements ..
!      Write (nout,*) 'F01VHF Example Program Results'
!      Skip heading in data file
!      Read (nin,*)
!      Write (nout,*)
!      Flush (nout)
!      Read (nin,*) n, uplo, transr
!      lda = n
!      lenar = n*(n+1)/2
!      Allocate (a(lda,n),ar(lenar))
!
!      Setup notional dimensions of RFP matrix AR
!      k = n/2
!      q = n - k
!      If (transr=='N' .Or. transr=='n') Then
!          lar1 = 2*k + 1
!      Else
!          lar1 = q
!      End If
!
!      Read an RFP matrix into array AR
!      Do i = 1, lar1
!          Read (nin,*) ar(i:lenar:lar1)
!      End Do
!
!      Print the Rectangular Full Packed array
!      title = 'RFP Packed Array AR:'
!      ifail = 0
!      Call x04dbf(matrix,diag,lenar,incl,ar,lenar,brac,form,title,intlabel,      &
!                  rlabs,nolabel,clabs,ncols,indent,ifail)
!
!      Write (nout,*)
!      Flush (nout)
!
!      Convert to triangular form
!      The NAG name equivalent of ztfttr is f01vhf
!      Call ztfttr(transr,uplo,n,ar,a,lda,info)
!
!      Print the unpacked array
!      title = 'Unpacked Matrix A:'

```

```

        ifail = 0
        Call x04dbf(uplo,diag,n,n,a,lda,brac,form,title,intlabel,rlabs,intlabel, &
            clabs,ncols,indent,ifail)

    End Program f01vhfe

```

10.2 Program Data

F01VHF Example Program Data

```

    4      'U'      'N'      : n, uplo, transr
    ( 1.30, 1.30)  ( 1.40, 1.40)
    ( 2.30, 2.30)  ( 2.40, 2.40)
    ( 3.30, 3.30)  ( 3.40, 3.40)
    ( 1.10,-1.10)  ( 4.40, 4.40)
    ( 1.20,-1.20)  ( 2.20,-2.20) : RFP Matrix AR

```

10.3 Program Results

F01VHF Example Program Results

RFP Packed Array AR:

```

 1 ( 1.30, 1.30)
 2 ( 2.30, 2.30)
 3 ( 3.30, 3.30)
 4 ( 1.10,-1.10)
 5 ( 1.20,-1.20)
 6 ( 1.40, 1.40)
 7 ( 2.40, 2.40)
 8 ( 3.40, 3.40)
 9 ( 4.40, 4.40)
10 ( 2.20,-2.20)

```

Unpacked Matrix A:

```

      1      2      3      4
1 ( 1.10, 1.10) ( 1.20, 1.20) ( 1.30, 1.30) ( 1.40, 1.40)
2      ( 2.20, 2.20) ( 2.30, 2.30) ( 2.40, 2.40)
3      ( 3.30, 3.30) ( 3.40, 3.40)
4      ( 4.40, 4.40)

```
