NAG Library Routine Document

F06WPF (ZTFSM)

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

F06WPF (ZTFSM) performs one of the matrix-matrix operations

$$B \leftarrow \alpha A^{-1}B, \qquad B \leftarrow \alpha A^{-\mathrm{H}}B,$$
$$B \leftarrow \alpha BA^{-1} \quad \text{or} \quad B \leftarrow \alpha BA^{-\mathrm{H}},$$

where A is a complex triangular matrix stored in Rectangular Full Packed (RFP) format, B is an m by n complex matrix, and α is a complex scalar. $A^{-\mathrm{H}}$ denotes $\left(A^{\mathrm{H}}\right)^{-1}$ or equivalently $\left(A^{-1}\right)^{\mathrm{H}}$. The RFP storage format is described in Section 3.3.3 in the F07 Chapter Introduction.

No test for singularity or near-singularity of A is included in this routine. Such tests must be performed before calling this routine.

2 Specification

```
SUBROUTINE F06WPF (TRANSR, SIDE, UPLO, TRANS, DIAG, M, N, ALPHA, A, B, LDB)

INTEGER                 M, N, LDB
COMPLEX (KIND=nag_wp) ALPHA, A(*), B(LDB,*)
CHARACTER(1)            TRANSR, SIDE, UPLO, TRANS, DIAG
```

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

3 Description

F06WPF (ZTFSM) solves (for X) a triangular linear system of one of the forms

$$AX = \alpha B, \qquad A^{\mathrm{H}}X = \alpha B,$$
$$XA = \alpha B \quad \text{or} \quad XA^{\mathrm{H}} = \alpha B,$$

where A is a complex triangular matrix stored in RFP format, B, X are m by n complex matrices, and α is a complex scalar.

4 References

None.

5 Parameters

1: TRANSR – CHARACTER(1) *Input*

On entry: specifies whether the normal RFP representation of A or its conjugate transpose is stored.

TRANSR = 'N'
The matrix A is stored in normal RFP format.

TRANSR = 'C'
The conjugate transpose of the RFP representation of the matrix A is stored.

Constraint: TRANSR = 'N' or 'C'.

2: SIDE – CHARACTER(1) *Input*

On entry: specifies whether B is operated on from the left or the right, or similarly whether A (or its transpose) appears to the left or right of the solution matrix in the linear system to be solved.

SIDE = 'L'
B is pre-multiplied from the left. The system to be solved has the form $AX = \alpha B$ or $A^{\mathrm{H}}X = \alpha B$.

SIDE = 'R'
B is post-multiplied from the right. The system to be solved has the form $XA = \alpha B$ or $XA^{\mathrm{H}} = \alpha B$.

Constraint: SIDE = 'L' or 'R'.

3: 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'.

4: TRANS – CHARACTER(1) *Input*

On entry: specifies whether the operation involves A^{-1} or $A^{-\mathrm{H}}$, i.e., whether or not A is transpose conjugated in the linear system to be solved.

TRANS = 'N'
The operation involves A^{-1}, i.e., A is not transpose conjugated.

TRANS = 'C'
The operation involves $A^{-\mathrm{H}}$, i.e., A is transpose conjugated.

Constraint: TRANS = 'N' or 'C'.

5: DIAG – CHARACTER(1) *Input*

On entry: specifies whether A has nonunit or unit diagonal elements.

DIAG = 'N'
The diagonal elements of A are stored explicitly.

DIAG = 'U'
The diagonal elements of A are assumed to be 1, the corresponding elements of A are not referenced.

Constraint: DIAG = 'N' or 'U'.

6: M – INTEGER *Input*

On entry: m, the number of rows of the matrix B.

Constraint: M ≥ 0.

7: N – INTEGER *Input*

On entry: n, the number of columns of the matrix B.

Constraint: N ≥ 0.

8: ALPHA – COMPLEX (KIND=nag_wp) *Input*

On entry: the scalar α.

9: A(∗) – COMPLEX (KIND=nag_wp) array *Input*

Note: the dimension of the array A must be at least $\max(1, \mathrm{M} \times (\mathrm{M}+1)/2)$ if SIDE = 'L' and at least $\max(1, \mathrm{N} \times (\mathrm{N}+1)/2)$ if SIDE = 'R'.

On entry: A, the m by m triangular matrix A if SIDE = 'L' or the n by n triangular matrix A if SIDE = 'R', stored in RFP format, as described in Section 3.3.3 in the F07 Chapter Introduction.

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

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

On entry: the m by n matrix B.

If ALPHA = 0, B need not be set.

On exit: the updated matrix B, or similarly the solution matrix X.

11: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F06WPF (ZTFSM) is called.

Constraint: LDB $\geq \max(1, \mathrm{M})$.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

This example reads in the upper triangular part of a symmetric matrix A which it converts to RFP format. It also reads in α and a 4 by 3 matrix B and then performs the matrix-matrix operation $B \leftarrow \alpha A^{-1}B$.

9.1 Program Text

```
    Program f06wpfe

!     F06WPF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04daf, ztfsm, ztrttf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Complex (Kind=nag_wp)            :: alpha
      Integer                          :: i, ifail, info, lda, ldb, m, n
      Character (1)                    :: side, trans, transr, uplo
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), af(:), b(:,:), work(:)
!     .. Executable Statements ..
      Write (nout,*) 'F06WPF Example Program Results'
```

```
!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) m, n, uplo, transr, side, alpha, trans

      lda = m
      ldb = m
      Allocate (a(lda,m),af((m*(m+1))/2),work(m),b(ldb,n))

!     Read upper or lower triangle of matrix A from data file

      If (uplo=='L' .Or. uplo=='l') Then
        Do i = 1, m
          Read (nin,*) a(i,1:i)
        End Do
      Else
        Do i = 1, m
          Read (nin,*) a(i,i:m)
        End Do
      End If

!     Read matrix B from data file

      Read (nin,*)(b(i,1:n),i=1,m)

!     Convert A to rectangular full packed storage in AF

!     The NAG name equivalent of ztrttf is f01vef
      Call ztrttf(transr,uplo,m,a,lda,af,info)

      Write (nout,*)
      Flush (nout)

!     The NAG name equivalent of ztfsm is f06wpf
      Call ztfsm(transr,side,uplo,trans,'N',m,n,alpha,af,b,ldb)
      Call x04daf('General',' ',m,n,b,ldb,'The Solution',ifail)

    End Program f06wpfe
```

9.2 Program Data

```
F06WPF Example Program Data
 4 3 'U' 'N' 'L' (4.21,1.28) 'N'          : M, N, UPLO,TRANSR,SIDE, ALPHA, TRANS
 (1.1,1.1) (1.2,1.2) (1.3,1.3) (1.4,1.4)
 (2.2,2.2) (2.3,2.3) (2.4,2.4)
 (3.3,3.3) (3.4,3.4)
 (4.4,4.4)                                : Unpacked Matrix A
 ( 1.80,0.59) ( 2.88, 1.23) (2.05, 0.78)
 ( 5.25,0.12) ( 1.76,-2.95) (2.20,-0.95)
 ( 1.58,2.01) (-2.69, 3.18) (0.11,-2.90)
 (-1.11,1.11) (-0.66, 1.66) (1.59,-0.59) : End of matrix B
```

9.3 Program Results

```
 F06WPF Example Program Results

 The Solution
              1          2          3
 1     -2.0339     8.6009     3.8676
        2.6429     4.3188     2.2452

 2      4.3280     1.0930     3.3517
       -4.3756    -8.8840    -0.0650

 3      2.5393    -0.9711    -2.0155
       -0.1237     2.5460    -1.5364

 4     -0.3229     0.1410     0.7955
        1.0621     1.2554    -0.8975
```