NAG Library Routine Document

F06WQF (ZHFRK)

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

F06WQF (ZHFRK) performs one of the Hermitian rank-k update operations

$$C \leftarrow \alpha A A^{\mathrm{H}} + \beta C \qquad \text{or} \qquad C \leftarrow \alpha A^{\mathrm{H}} A + \beta C,$$

where A is a real matrix, C is an n by n complex Hermitian matrix stored in Rectangular Full Packed (RFP) format, and α and β are real scalars. The RFP storage format is described in Section 3.3.3 in the F07 Chapter Introduction.

2 Specification

```
SUBROUTINE F06WQF (TRANSR, UPLO, TRANS, N, K, ALPHA, A, LDA, BETA, C)

INTEGER                 N, K, LDA
REAL (KIND=nag_wp)      ALPHA, BETA
COMPLEX (KIND=nag_wp) A(LDA,*), C(N*(N+1)/2)
CHARACTER(1)            TRANSR, UPLO, TRANS
```

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

3 Description

None.

4 References

None.

5 Parameters

1: TRANSR – CHARACTER(1) *Input*

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

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

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

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

2: UPLO – CHARACTER(1) *Input*

On entry: specifies whether the upper or lower triangular part of C is stored in RFP format.

UPLO = 'U'
The upper triangular part of C is stored in RFP format.

UPLO = 'L'
The lower triangular part of C is stored in RFP format.

Constraint: UPLO = 'U' or 'L'.

3: TRANS – CHARACTER(1) *Input*

On entry: specifies the operation to be performed.

TRANS = 'N'
$C \leftarrow \alpha A A^{\mathrm{H}} + \beta C$.

TRANS = 'C'
$C \leftarrow \alpha A^{\mathrm{H}} A + \beta C$.

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

4: N – INTEGER *Input*

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

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

5: K – INTEGER *Input*

On entry: k, the number of columns of A if TRANS = 'N', or the number of rows of A if TRANS = 'C'.

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

6: ALPHA – REAL (KIND=nag_wp) *Input*

On entry: the scalar α.

7: A(LDA,$*$) – COMPLEX (KIND=nag_wp) array *Input*

Note: the second dimension of the array A must be at least $\max(1, \mathrm{K})$ if TRANS = 'N' and at least $\max(1, \mathrm{N})$ if TRANS = 'C'.

On entry: the matrix A; A is n by k if TRANS = 'N', or k by n if TRANS = 'C'.

8: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F06WQF (ZHFRK) is called.

Constraints:

if TRANS = 'N', $\mathrm{LDA} \geq \max(1, \mathrm{N})$;
if TRANS = 'C', $\mathrm{LDA} \geq \max(1, \mathrm{K})$.

9: BETA – REAL (KIND=nag_wp) *Input*

On entry: the scalar β.

10: C($\mathrm{N} \times (\mathrm{N}+1)/2$) – COMPLEX (KIND=nag_wp) array *Input/Output*

On entry: the upper or lower triangular part (as specified by UPLO) of the n by n symmetric matrix C, stored in RFP format, as described in Section 3.3.3 in the F07 Chapter Introduction.

On exit: the updated matrix C, that is its upper or lower triangular part stored in RFP format.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

This example reads in the lower triangular part of a symmetric matrix C which it converts to RFP format. It also reads in α, β and a 4 by 3 matrix A and then performs the Hermitian rank-3 update $C \leftarrow \alpha AA^{\mathrm{H}} + \beta C$.

9.1 Program Text

```
    Program f06wqfe

!     F06WQF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04daf, zhfrk, ztfttr, ztrttf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: alpha, beta
      Integer                          :: i, ifail, info, k, lda, n
      Character (1)                    :: trans, transr, uplo
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), c(:,:), cf(:)
!     .. Executable Statements ..
      Write (nout,*) 'F06WQF Example Program Results'

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

      Read (nin,*) n, k, uplo, transr, alpha, beta, trans

      lda = n
      Allocate (c(lda,n),cf((n*(n+1))/2),a(lda,k))

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

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

!     Read matrix A from data file

      Read (nin,*)(a(i,1:k),i=1,n)

!     Convert C to rectangular full packed storage in CF

!     The NAG name equivalent of ztrttf is f01vef
      Call ztrttf(transr,uplo,n,c,lda,cf,info)

      Write (nout,*)
      Flush (nout)

!     Perform the rank-k update

!     The NAG name equivalent of zhfrk is f06wqf
```

```
      Call zhfrk(transr,uplo,trans,n,k,alpha,a,lda,beta,cf)

!     Convert CF back from rectangular full packed to standard format in C

!     The NAG name equivalent of ztfttr is f01vhf
      Call ztfttr(transr,uplo,n,cf,c,lda,info)

!     Print out the result, stored in the lower triangle of matrix C

      ifail = 0
      Call x04daf('Lower','N',n,n,c,lda,'The Solution',ifail)

    End Program f06wqfe
```

9.2 Program Data

```
F06WQF Example Program Data
 4 3 'L' 'N' 2.21 2.89 'N'                : N,K, UPLO,TRANSR, ALPHA,BETA, TRANS
 (1.0,3.0)
 (2.0,2.0) (3.0,3.0)
 (4.0,4.0) (4.0,4.0) (5.0,5.0)
 (5.0,5.0) (5.0,6.0) (6.0,6.0) (6.0,6.0)   : Unpacked matrix C
 ( 3.21, 1.32) ( 2.31, 0.25) ( 1.65, 1.87)
 ( 0.32,-1.55) ( 1.80, 1.88) ( 2.05,-0.89)
 ( 5.25,-2.95) (-1.95,-3.80) ( 1.58,-2.69)
 (-2.90,-3.04) (-1.11,-0.66) (-0.59, 0.80) : End of matrix A
```

9.3 Program Results

```
 F06WQF Example Program Results

 The Solution
             1          2          3          4
 1     55.1885
        0.0000

 2     17.5536    40.2153
       -9.2637     0.0000

 3     22.7883    14.2818   156.4204
      -59.3437    11.3638    -0.0000

 4    -19.8678    11.4084     7.0222    62.2194
        3.9432     9.7064   -44.0297    -0.0000
```