NAG Library Routine Document

F06ZRF (ZHER2K)

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

F06ZRF (ZHER2K) performs one of the Hermitian rank-$2k$ update operations

$$C \leftarrow \alpha AB^{\mathrm{H}} + \bar{\alpha}BA^{\mathrm{H}} + \beta C \quad \text{or} \quad C \leftarrow \alpha A^{\mathrm{H}}B + \bar{\alpha}B^{\mathrm{H}}A + \beta C,$$

where A and B are complex matrices, C is an n by n complex Hermitian matrix, α is a complex scalar, and β is a real scalar.

2 Specification

```
SUBROUTINE F06ZRF (UPLO, TRANS, N, K, ALPHA, A, LDA, B, LDB, BETA, C,      &
                   LDC)

INTEGER                 N, K, LDA, LDB, LDC
REAL (KIND=nag_wp)      BETA
COMPLEX (KIND=nag_wp)   ALPHA, A(LDA,*), B(LDB,*), C(LDC,*)
CHARACTER(1)            UPLO, TRANS
```

The routine may be called by its BLAS name ***zher2k***.

3 Description

None.

4 References

None.

5 Arguments

1: UPLO – CHARACTER(1) *Input*

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

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

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

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

2: TRANS – CHARACTER(1) *Input*

On entry: specifies the operation to be performed.

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

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

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

3: N – INTEGER *Input*

On entry: n, the order of the matrix C; the number of rows of A if TRANS = 'N', or the number of columns of A if TRANS = 'C'.

Constraint: $N \geq 0$.

4: 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: $K \geq 0$.

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

On entry: the scalar α.

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

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

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

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F06ZRF (ZHER2K) is called.

Constraints:

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

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

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

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

9: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F06ZRF (ZHER2K) is called.

Constraints:

if TRANS = 'N', $LDB \geq \max(1, N)$;
if TRANS = 'C', $LDB \geq \max(1, K)$.

10: BETA – REAL (KIND=nag_wp) *Input*

On entry: the scalar β.

11: $C(LDC, *)$ – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: the n by n Hermitian matrix C.

If UPLO = 'U', the upper triangular part of C must be stored and the elements of the array below the diagonal are not referenced.

If UPLO = 'L', the lower triangular part of C must be stored and the elements of the array above the diagonal are not referenced.

On exit: the updated matrix C. The imaginary parts of the diagonal elements are set to zero.

12: LDC – INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which F06ZRF (ZHER2K) is called.

Constraint: LDC $\geq \max(1, \mathrm{N})$.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Parallelism and Performance

F06ZRF (ZHER2K) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F06ZRF (ZHER2K) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

None.

10 Example

None.