NAG Library Routine Document

F06TTF

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

F06TTF performs a QR or RQ factorization of the product of a complex upper triangular matrix and a complex matrix of plane rotations.

2 Specification

```
SUBROUTINE F06TTF (SIDE, N, K1, K2, C, S, A, LDA)

INTEGER                N, K1, K2, LDA
REAL (KIND=nag_wp)     C(*)
COMPLEX (KIND=nag_wp) S(*), A(LDA,*)
CHARACTER(1)           SIDE
```

3 Description

F06TTF performs one of the transformations

$$R \leftarrow PUQ^{\mathrm{H}} \quad \text{or} \quad R \leftarrow QUP^{\mathrm{H}},$$

where U is a given n by n complex upper triangular matrix, P is a given complex unitary matrix, and Q is a complex unitary matrix chosen to make R upper triangular. Both P and Q are represented as sequences of plane rotations in planes k_1 to k_2.

If SIDE $=$ 'L',

$$R \leftarrow PUQ^{\mathrm{H}},$$

where $P = P_{k_2-1} \dots P_{k_1+1} P_{k_1}$ and $Q = Q_{k_2-1} \dots Q_{k_1+1} Q_{k_1}$.

If SIDE $=$ 'R',

$$R \leftarrow QUP^{\mathrm{H}},$$

where $P = P_{k_1} P_{k_1+1} \dots P_{k_2-1}$ and $Q = Q_{k_1} Q_{k_1+1} \dots Q_{k_2-1}$.

In either case P_k and Q_k are rotations in the $(k, k+1)$ plane.

The 2 by 2 rotation part of P_k or Q_k has the form

$$\begin{pmatrix} c_k & \bar{s}_k \\ -s_k & c_k \end{pmatrix}$$

with c_k real.

4 References

None.

5 Arguments

1: SIDE – CHARACTER(1) *Input*

On entry: specifies whether P is applied from the left or the right in the transformation.

SIDE = 'L'
P is applied from the left.

SIDE = 'R'
P is applied from the right.

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

2: N – INTEGER *Input*

On entry: n, the order of the matrices U and R.

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

3: K1 – INTEGER *Input*
4: K2 – INTEGER *Input*

On entry: the values k_1 and k_2.

If $\text{K1} < 1$ or $\text{K2} \leq \text{K1}$ or $\text{K2} > \text{N}$, an immediate return is effected.

5: C($*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the dimension of the array C must be at least $\text{K2} - \text{K1}$.

On entry: $\text{C}(k)$ must hold the cosine of the rotation P_k, for $k = k_1, \ldots, k_2 - 1$.

On exit: $\text{C}(k)$ holds the cosine of the rotation Q_k, for $k = k_1, \ldots, k_2 - 1$.

6: S($*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the dimension of the array S must be at least $\text{K2} - \text{K1}$.

On entry: $\text{S}(k)$ must hold the sine of the rotation P_k, for $k = k_1, \ldots, k_2 - 1$.

On exit: $\text{S}(k)$ holds the sine of the rotation Q_k, for $k = k_1, \ldots, k_2 - 1$.

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

Note: the second dimension of the array A must be at least N.

On entry: the n by n upper triangular matrix U.

On exit: the upper triangular matrix R.

8: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F06TTF is called.

Constraint: $\text{LDA} \geq \max(1, \text{N})$.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Parallelism and Performance

F06TTF is not threaded in any implementation.

9 Further Comments

None.

10 Example

None.