NAG Library Routine Document

F06TSF

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

F06TSF performs a QR or RQ factorization (as a sequence of plane rotations) of a complex upper spiked matrix.

2 Specification

```
SUBROUTINE F06TSF (SIDE, N, K1, K2, C, S, A, LDA)

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

3 Description

F06TSF transforms an n by n complex upper spiked matrix H to upper triangular form R by applying a complex unitary matrix P from the left or the right. H is assumed to have real diagonal elements except where the spike joins the diagonal; R has real diagonal elements. P is formed as a sequence of plane rotations in planes k_1 to k_2.

If SIDE = 'L', H is assumed to have a row spike, with nonzero elements $h_{k_2,k}$, for $k = k_1, \ldots, k_2 - 1$. The rotations are applied from the left:

$$PH = R,$$

where $P = DP_{k_2-1} \cdots P_{k_1+1}P_{k_1}$, P_k is a rotation in the (k, k_2) plane and $D = \mathrm{diag}(1, \ldots, 1, d_{k_2}, 1, \ldots, 1)$ with $|d_{k_2}| = 1$.

If SIDE = 'R', H is assumed to have a column spike, with nonzero elements h_{k+1,k_1}, for $k = k_1, \ldots, k_2 - 1$. The rotations are applied from the right:

$$HP^{\mathrm{H}} = R,$$

where $P = DP_{k_1}P_{k_1+1} \cdots P_{k_2-1}$, P_k is a rotation in the $(k_1, k+1)$ plane and $D = \mathrm{diag}(1, \ldots, 1, d_{k_1}, 1, \ldots, 1)$ with $|d_{k_1}| = 1$.

The 2 by 2 plane rotation part of P_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 H is operated on from the left or the right.

SIDE = 'L'
: H is pre-multiplied from the left.

SIDE = 'R'
: H is post-multiplied from the right.

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

2: N – INTEGER *Input*

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

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

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

On entry: the values k_1 and k_2.

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

5: $\mathrm{C}(\mathrm{K2} - 1)$ – REAL (KIND=nag_wp) array *Output*

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

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

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

On entry: the nonzero elements of the spike of H: $\mathrm{S}(k)$ must hold $h_{k_2,k}$ if SIDE = 'L', and h_{k+1,k_1} if SIDE = 'R', for $k = k_1, \ldots, k_2 - 1$.

On exit: $\mathrm{S}(k)$ holds s_k, the sine of the rotation P_k, for $k = k_1, \ldots, k_2 - 1$; $\mathrm{S}(k_2)$ holds d_{k_2}, the k_2th diagonal element of D, if SIDE = 'L', or d_{k_1}, the k_1th diagonal element of D, if SIDE = 'R'.

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

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

On entry: the upper triangular part of the n by n upper spiked matrix H. The imaginary parts of the diagonal elements must be zero, except for the (k_2, k_2) element if SIDE = 'L', or the (k_1, k_1) element if SIDE = 'R'.

On exit: the upper triangular matrix R. The imaginary parts of the diagonal elements are set to zero.

8: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F06TSF is called.

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

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Parallelism and Performance

F06TSF 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.