NAG Library Routine Document

F06TPF

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

F06TPF performs a QR factorization (as a sequence of plane rotations) of a complex upper triangular matrix that has been modified by a rank-1 update.

2 Specification

```
SUBROUTINE F06TPF (N, ALPHA, X, INCX, Y, INCY, A, LDA, C, S)

INTEGER                 N, INCX, INCY, LDA
REAL (KIND=nag_wp)      C(N-1)
COMPLEX (KIND=nag_wp) ALPHA, X(*), Y(*), A(LDA,*), S(N)
```

3 Description

F06TPF performs a QR factorization of an upper triangular matrix which has been modified by a rank-1 update:

$$\alpha x y^{\mathrm{T}} + U = QR$$

where U and R are n by n complex upper triangular matrices with real diagonal elements, x and y are n-element complex vectors, α is a complex scalar, and Q is an n by n complex unitary matrix.

Q is formed as the product of two sequences of plane rotations and a unitary diagonal matrix D:

$$Q^{\mathrm{H}} = DQ_{n-1} \cdots Q_2 Q_1 P_1 P_2 \cdots P_{n-1}$$

where

P_k is a rotation in the (k, n) plane, chosen to annihilate x_k: thus $Px = \beta e_n$, where $P = P_1 P_2 \cdots P_{n-1}$ and e_n is the last column of the unit matrix;

Q_k is a rotation in the (k, n) plane, chosen to annihilate the (n, k) element of $\left(\alpha\beta e_n y^{\mathrm{T}} + PU\right)$, and thus restore it to upper triangular form;

$D = \mathrm{diag}(1, \ldots, 1, d_n)$, with d_n chosen to make r_{nn} real; $|d_n| = 1$.

The 2 by 2 plane 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. The tangents of the rotations P_k are returned in the array X; the cosines and sines of these rotations can be recovered by calling F06BCF. The cosines and sines of the rotations Q_k are returned directly in the arrays C and S.

4 References

None.

5 Parameters

1: N – INTEGER *Input*

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

Constraint: N ≥ 0.

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

On entry: the scalar α.

3: X$(*)$ – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the dimension of the array X must be at least $\max(1, 1 + (\text{N} - 1) \times \text{INCX})$.

On entry: the n-element vector x. x_i must be stored in X$(1 + (i - 1) \times \text{INCX})$, for $i = 1, 2, \ldots, \text{N}$.

Intermediate elements of X are not referenced.

On exit: the referenced elements are overwritten by details of the sequence of plane rotations.

4: INCX – INTEGER *Input*

On entry: the increment in the subscripts of X between successive elements of x.

Constraint: INCX > 0.

5: Y$(*)$ – COMPLEX (KIND=nag_wp) array *Input*

Note: the dimension of the array Y must be at least $\max(1, 1 + (\text{N} - 1) \times \text{INCY})$.

On entry: the n-element vector y. y_i must be stored in Y$(1 + (i - 1) \times \text{INCY})$, for $i = 1, 2, \ldots, \text{N}$.

Intermediate elements of Y are not referenced.

6: INCY – INTEGER *Input*

On entry: the increment in the subscripts of Y between successive elements of y.

Constraint: INCY > 0.

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. The imaginary parts of the diagonal elements must be zero.

On exit: the upper triangular matrix R. The imaginary parts of the diagonal elements must be zero.

8: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F06TPF is called.

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

9: C(N $- 1$) – REAL (KIND=nag_wp) array *Output*

On exit: the cosines of the rotations Q_k, for $k = 1, 2, \ldots, n - 1$.

10: S(N) – COMPLEX (KIND=nag_wp) array *Output*

On exit: the sines of the rotations Q_k, for $k = 1, 2, \ldots, n - 1$; S(n) holds d_n, the nth diagonal element of D.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

None.