

NAG Library Routine Document

F06VJF

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

F06VJF permutes the rows or columns of a complex rectangular matrix using an integer array of permutations.

2 Specification

```
SUBROUTINE F06VJF (SIDE, TRANS, N, PERM, K, B, LDB)
  INTEGER          N, PERM(*), K, LDB
  COMPLEX (KIND=nag_wp) B(LDB,*)
  CHARACTER(1)    SIDE, TRANS
```

3 Description

F06VJF performs one of the permutation operations

$$\begin{array}{l} B \leftarrow P^T B, \quad B \leftarrow P B, \\ B \leftarrow B P^T \quad \text{or} \quad B \leftarrow B P, \end{array}$$

where B is a complex matrix, and P is a permutation matrix.

P is represented in the form

$$P = P_{1,p_1} P_{2,p_2} \cdots P_{n,p_n},$$

where $P_{i,j}$ is the permutation matrix that interchanges items i and j ; that is, $P_{i,j}$ is the unit matrix with rows and columns i and j interchanged. If $i = j$, $P_{i,j} = I$.

Let m denote the number of rows of B if $\text{SIDE} = \text{'L'}$, or the number of columns of B if $\text{SIDE} = \text{'R'}$: the routine does not require m to be passed as an argument, but assumes that $m \geq p_i$, for $i = 1, 2, \dots, n$.

This routine requires the indices p_i to be supplied in an integer array; F06VKF performs the same operation with the indices supplied in a real array.

4 References

None.

5 Arguments

1: SIDE – CHARACTER(1) Input
 2: TRANS – CHARACTER(1) Input

On entry: specifies the operation to be performed.

SIDE = 'L' and TRANS = 'T'
 $B \leftarrow P^T B$.

SIDE = 'L' and TRANS = 'N'
 $B \leftarrow P B$.

SIDE = 'R' and TRANS = 'T'
 $B \leftarrow B P^T$.

SIDE = 'R' and TRANS = 'N'
 $B \leftarrow BP.$

Constraints:

SIDE = 'L' or 'R';
 TRANS = 'N' or 'T'.

- 3: N – INTEGER *Input*
On entry: n , the number of interchanges in the representation of P .
Constraint: $N \geq 0$.
- 4: PERM(*) – INTEGER array *Input*
Note: the dimension of the array PERM must be at least $\max(1, N)$.
On entry: the n indices p_i which define the interchanges in the representation of P . It is usual to have $p_i \geq i$, but this is not necessary.
Constraint: $1 \leq \text{PERM}(i) \leq m$.
- 5: K – INTEGER *Input*
On entry: k , the number of columns of B if SIDE = 'L', or the number of rows of B if SIDE = 'R'.
Constraint: $K \geq 0$.
- 6: B(LDB, *) – COMPLEX (KIND=nag_wp) array *Input/Output*
Note: the second dimension of the array B must be at least $\max(1, K)$ if SIDE = 'L' and at least $\max\left(1, \max_k \{\text{PERM}(k)\}\right)$ if SIDE = 'R'.
On entry: the original matrix B ; B is m by k if SIDE = 'L', or k by m if SIDE = 'R'.
On exit: the permuted matrix B .
- 7: LDB – INTEGER *Input*
On entry: the first dimension of the array B as declared in the (sub)program from which F06VJF is called.
Constraints:
 if SIDE = 'L', $\text{LDB} \geq \max(1, m)$;
 if SIDE = 'R', $\text{LDB} \geq \max(1, K)$.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Parallelism and Performance

F06VJF is not threaded in any implementation.

9 Further Comments

None.

10 Example

None.
