NAG Library Routine Document

F08CXF (ZUNMRQ)

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

F08CXF (ZUNMRQ) multiplies a general complex m by n matrix C by the complex unitary matrix Q from an RQ factorization computed by F08CVF (ZGERQF).

2 Specification

```
SUBROUTINE F08CXF (SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC, WORK,        &
                   LWORK, INFO)
INTEGER                M, N, K, LDA, LDC, LWORK, INFO
COMPLEX (KIND=nag_wp) A(LDA,*), TAU(*), C(LDC,*), WORK(max(1,LWORK))
CHARACTER(1)           SIDE, TRANS
```

The routine may be called by its LAPACK name ***zunmrq***.

3 Description

F08CXF (ZUNMRQ) is intended to be used following a call to F08CVF (ZGERQF), which performs an RQ factorization of a complex matrix A and represents the unitary matrix Q as a product of elementary reflectors.

This routine may be used to form one of the matrix products

$$QC, \quad Q^{\mathrm{H}}C, \quad CQ, \quad CQ^{\mathrm{H}},$$

overwriting the result on C, which may be any complex rectangular m by n matrix.

A common application of this routine is in solving underdetermined linear least squares problems, as described in the F08 Chapter Introduction, and illustrated in Section 10 in F08CVF (ZGERQF).

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Arguments

1: SIDE – CHARACTER(1) *Input*

On entry: indicates how Q or Q^{H} is to be applied to C.

SIDE = 'L'
: Q or Q^{H} is applied to C from the left.

SIDE = 'R'
: Q or Q^{H} is applied to C from the right.

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

2: TRANS – CHARACTER(1) *Input*

On entry: indicates whether Q or Q^{H} is to be applied to C.

TRANS = 'N'
: Q is applied to C.

TRANS = 'C'
: Q^{H} is applied to C.

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

3: M – INTEGER *Input*

On entry: m, the number of rows of the matrix C.

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

4: N – INTEGER *Input*

On entry: n, the number of columns of the matrix C.

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

5: K – INTEGER *Input*

On entry: k, the number of elementary reflectors whose product defines the matrix Q.

Constraints:

if SIDE = 'L', $\mathrm{M} \geq \mathrm{K} \geq 0$;
if SIDE = 'R', $\mathrm{N} \geq \mathrm{K} \geq 0$.

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

Note: the second dimension of the array A must be at least $\max(1, \mathrm{M})$ if SIDE = 'L' and at least $\max(1, \mathrm{N})$ if SIDE = 'R'.

On entry: the ith row of A must contain the vector which defines the elementary reflector H_i, for $i = 1, 2, \ldots, k$, as returned by F08CVF (ZGERQF).

On exit: is modified by F08CXF (ZUNMRQ) but restored on exit.

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08CXF (ZUNMRQ) is called.

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

8: TAU($*$) – COMPLEX (KIND=nag_wp) array *Input*

Note: the dimension of the array TAU must be at least $\max(1, \mathrm{K})$.

On entry: TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by F08CVF (ZGERQF).

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

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

On entry: the m by n matrix C.

On exit: C is overwritten by QC or $Q^{\mathrm{H}}C$ or CQ or CQ^{H} as specified by SIDE and TRANS.

10: LDC – INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which F08CXF (ZUNMRQ) is called.

Constraint: $\text{LDC} \geq \max(1, \text{M})$.

11: WORK(max(1, LWORK)) – COMPLEX (KIND=nag_wp) array *Workspace*

On exit: if $\text{INFO} = 0$, the real part of WORK(1) contains the minimum value of LWORK required for optimal performance.

12: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08CXF (ZUNMRQ) is called.

If $\text{LWORK} = -1$, a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued.

Suggested value: for optimal performance, $\text{LWORK} \geq \text{N} \times nb$ if SIDE = 'L' and at least $\text{M} \times nb$ if SIDE = 'R', where nb is the optimal ***block size***.

Constraints:

if SIDE = 'L', $\text{LWORK} \geq \max(1, \text{N})$ or $\text{LWORK} = -1$;
if SIDE = 'R', $\text{LWORK} \geq \max(1, \text{M})$ or $\text{LWORK} = -1$.

13: INFO – INTEGER *Output*

On exit: $\text{INFO} = 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

$\text{INFO} < 0$

If $\text{INFO} = -i$, argument i had an illegal value.

If $\text{INFO} = -999$, dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

An explanatory message is output, and execution of the program is terminated.

7 Accuracy

The computed result differs from the exact result by a matrix E such that

$$\|E\|_2 = O\epsilon\|C\|_2$$

where ϵ is the ***machine precision***.

8 Parallelism and Performance

F08CXF (ZUNMRQ) 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

The total number of floating-point operations is approximately $8nk(2m - k)$ if SIDE = 'L' and $8mk(2n - k)$ if SIDE = 'R'.

The real analogue of this routine is F08CKF (DORMRQ).

10 Example

See Section 10 in F08CVF (ZGERQF).