NAG Fortran Library Routine Document

F08CUF (ZUNMQL)

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

F08CUF (ZUNMQL) multiplies a general complex m by n matrix C by the complex unitary matrix Q from a QL factorization computed by F08CSF (ZGEQLF).

2 Specification

```
      SUBROUTINE F08CUF (SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC, WORK,
     1                   LWORK, INFO)
      INTEGER            M, N, K, LDA, LDC, LWORK, INFO
      complex*16         A(LDA,*), TAU(*), C(LDC,*), WORK(*)
      CHARACTER*1        SIDE, TRANS
```

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

3 Description

F08CUF (ZUNMQL) is intended to be used following a call to F08CSF (ZGEQLF), which performs a QL 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 linear least squares problems, as described in the F08 Chapter Introduction, and illustrated in Section 9 of the document for F08CSF (ZGEQLF).

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 URL: http://www.netlib.org/lapack/lug

5 Parameters

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: $M \geq 0$.

4: N – INTEGER *Input*

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

Constraint: $N \geq 0$.

5: K – INTEGER *Input*

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

Constraints:

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

6: A(LDA,$*$) – ***complex*16*** array *Input*

Note: the second dimension of the array A must be at least $\max(1, K)$.

On entry: details of the vectors which define the elementary reflectors as returned by F08CSF (ZGEQLF) and stored in the $(n - k + i)$th columns of A, for $i = 1, 2, \ldots, k$.

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08CUF (ZUNMQL) is called.

Constraints:

if SIDE = 'L', $LDA \geq \max(1, M)$;
if SIDE = 'R', $LDA \geq \max(1, N)$.

8: TAU($*$) – ***complex*16*** array *Input*

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

On entry: further details of the elementary reflectors as returned by F08CSF (ZGEQLF).

9: C(LDC,$*$) – ***complex*16*** array *Input/Output*

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

On entry: the m by n matrix C.

On exit: is overwritten by QC or $Q^H C$ 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 F08CUF (ZUNMQL) is called.

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

11: WORK($*$) – ***complex*16*** array *Workspace*

Note: the dimension of the array WORK must be at least $\max(1, \text{LWORK})$.

On exit: if 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 F08CUF (ZUNMQL) is called.

If 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, LWORK $\geq$ N $\times$ nb if SIDE = 'L' and at least M $\times$ nb if SIDE = 'R', where nb is the optimal ***block size***.

Constraints:

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

13: INFO – INTEGER *Output*

On exit: INFO = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO $= -i$, the ith parameter had an illegal value. 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 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 F08CGF (DORMQL).

9 Example

See Section 9 of the document for F08CSF (ZGEQLF).