NAG Library Routine Document

F08AUF (ZUNMQR)

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

F08AUF (ZUNMQR) multiplies an arbitrary complex matrix C by the complex unitary matrix Q from a QR factorization computed by F08ASF (ZGEQRF), F08BSF (ZGEQPF) or F08BTF (ZGEQP3).

2 Specification

```
SUBROUTINE F08AUF (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 ***zunmqr***.

3 Description

F08AUF (ZUNMQR) is intended to be used after a call to F08ASF (ZGEQRF), F08BSF (ZGEQPF) or F08BTF (ZGEQP3), which perform a QR factorization of a complex matrix A. The unitary matrix Q is represented as a product of elementary reflectors.

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

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

overwriting the result on C (which may be any complex rectangular 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 10 in F08ASF (ZGEQRF).

4 References

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

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

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*

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

On entry: details of the vectors which define the elementary reflectors, as returned by F08ASF (ZGEQRF), F08BSF (ZGEQPF) or F08BTF (ZGEQP3).

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08AUF (ZUNMQR) is called.

Constraints:

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

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: further details of the elementary reflectors, as returned by F08ASF (ZGEQRF), F08BSF (ZGEQPF) or F08BTF (ZGEQP3).

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 F08AUF (ZUNMQR) is called.

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

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

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 F08AUF (ZUNMQR) 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 $\geq \mathrm{N} \times nb$ if SIDE $=$ 'L' and at least M $\times nb$ if SIDE $=$ 'R', where nb is the optimal ***block size***.

Constraints:

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

INFO < 0

If INFO $= -i$, argument i 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 Parallelism and Performance

F08AUF (ZUNMQR) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F08AUF (ZUNMQR) 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 real 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 F08AGF (DORMQR).

10 Example

See Section 10 in F08ASF (ZGEQRF).