

# NAG Fortran Library Routine Document

## F08CTF (ZUNGQL)

**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

F08CTF (ZUNGQL) generates all or part of the complex  $m$  by  $m$  unitary matrix  $Q$  from a  $QL$  factorization computed by F08CSF (ZGEQLF).

### 2 Specification

```
SUBROUTINE F08CTF (M, N, K, A, LDA, TAU, WORK, LWORK, INFO)
INTEGER          M, N, K, LDA, LWORK, INFO
complex*16     A(LDA,*), TAU(*), WORK(*)
```

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

### 3 Description

F08CTF (ZUNGQL) is intended to be used after a call to F08CSF (ZGEQLF), which performs a  $QL$  factorization of a complex matrix  $A$ . The unitary matrix  $Q$  is represented as a product of elementary reflectors.

This routine may be used to generate  $Q$  explicitly as a square matrix, or to form only its trailing columns.

Usually  $Q$  is determined from the  $QL$  factorization of an  $m$  by  $p$  matrix  $A$  with  $m \geq p$ . The whole of  $Q$  may be computed by:

```
CALL ZUNGQL (M,M,P,A,LDA,TAU,WORK,LWORK,INFO)
```

(note that the array  $A$  must have at least  $m$  columns) or its trailing  $p$  columns by:

```
CALL ZUNGQL (M,P,P,A,LDA,TAU,WORK,LWORK,INFO)
```

The columns of  $Q$  returned by the last call form an orthonormal basis for the space spanned by the columns of  $A$ ; thus F08CSF (ZGEQLF) followed by F08CTF (ZUNGQL) can be used to orthogonalise the columns of  $A$ .

The information returned by F08CSF (ZGEQLF) also yields the  $QL$  factorization of the trailing  $k$  columns of  $A$ , where  $k < p$ . The unitary matrix arising from this factorization can be computed by:

```
CALL ZUNGQL (M,M,K,A,LDA,TAU,WORK,LWORK,INFO)
```

or its trailing  $k$  columns by:

```
CALL ZUNGQL (M,K,K,A,LDA,TAU,WORK,LWORK,INFO)
```

### 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>

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

## 5 Parameters

- 1: M – INTEGER *Input*  
*On entry:*  $m$ , the number of rows of the matrix  $Q$ .  
*Constraint:*  $M \geq 0$ .
- 2: N – INTEGER *Input*  
*On entry:*  $n$ , the number of columns of the matrix  $Q$ .  
*Constraint:*  $M \geq N \geq 0$ .
- 3: K – INTEGER *Input*  
*On entry:*  $k$ , the number of elementary reflectors whose product defines the matrix  $Q$ .  
*Constraint:*  $N \geq K \geq 0$ .
- 4: A(LDA,\*) – **complex\*16** array *Input/Output*  
**Note:** the second dimension of the array A must be at least  $\max(1, N)$ .  
*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  $i = 1, 2, \dots, k$ .  
*On exit:* the  $m$  by  $n$  matrix  $Q$ .
- 5: LDA – INTEGER *Input*  
*On entry:* the first dimension of the array A as declared in the (sub)program from which F08CTF (ZUNGQL) is called.  
*Constraint:*  $LDA \geq \max(1, M)$ .
- 6: 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).
- 7: WORK(\*) – **complex\*16** array *Workspace*  
**Note:** the dimension of the array WORK must be at least  $\max(1, LWORK)$ .  
*On exit:* if  $INFO = 0$ , the real part of  $WORK(1)$  contains the minimum value of LWORK required for optimal performance.
- 8: LWORK – INTEGER *Input*  
*On entry:* the dimension of the array WORK as declared in the (sub)program from which F08CTF (ZUNGQL) 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,  $LWORK \geq N \times nb$ , where  $nb$  is the optimal **block size**.  
*Constraint:*  $LWORK \geq \max(1, N)$ .
- 9: 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  $i$ th parameter had an illegal value. An explanatory message is output, and execution of the program is terminated.

## 7 Accuracy

The computed matrix  $Q$  differs from an exactly unitary matrix by a matrix  $E$  such that

$$\|E\|_2 = O(\epsilon),$$

where  $\epsilon$  is the *machine precision*.

## 8 Further Comments

The total number of real floating-point operations is approximately  $16mnk - 8(m+n)k^2 + \frac{16}{3}k^3$ ; when  $n = k$ , the number is approximately  $\frac{8}{3}n^2(3m - n)$ .

The real analogue of this routine is F08CFF (DORGQL).

## 9 Example

This example generates the first four columns of the matrix  $Q$  of the  $QL$  factorization of  $A$  as returned by F08CSF (ZGEQLF), where

$$A = \begin{pmatrix} 0.96 - 0.81i & -0.03 + 0.96i & -0.91 + 2.06i & -0.05 + 0.41i \\ -0.98 + 1.98i & -1.20 + 0.19i & -0.66 + 0.42i & -0.81 + 0.56i \\ 0.62 - 0.46i & 1.01 + 0.02i & 0.63 - 0.17i & -1.11 + 0.60i \\ -0.37 + 0.38i & 0.19 - 0.54i & -0.98 - 0.36i & 0.22 - 0.20i \\ 0.83 + 0.51i & 0.20 + 0.01i & -0.17 - 0.46i & 1.47 + 1.59i \\ 1.08 - 0.28i & 0.20 - 0.12i & -0.07 + 1.23i & 0.26 + 0.26i \end{pmatrix}.$$

Note that the block size (NB) of 64 assumed in this example is not realistic for such a small problem, but should be suitable for large problems.

### 9.1 Program Text

```
*      F08CTF Example Program Text
*      Mark 21 Release. NAG Copyright 2004.
*      .. Parameters ..
INTEGER          NIN, NOUT
PARAMETER       (NIN=5,NOUT=6)
INTEGER          MMAX, NB, NMAX
PARAMETER       (MMAX=8,NB=64,NMAX=8)
INTEGER          LDA, LWORK
PARAMETER       (LDA=MMAX,LWORK=NB*NMAX)
*      .. Local Scalars ..
INTEGER          I, IFAIL, INFO, J, M, N
CHARACTER*30     TITLE
*      .. Local Arrays ..
COMPLEX *16      A(LDA,NMAX), TAU(NMAX), WORK(LWORK)
CHARACTER        CLABS(1), RLABS(1)
*      .. External Subroutines ..
EXTERNAL         X04DBF, ZGEQLF, ZUNGQL
*      .. Executable Statements ..
WRITE (NOUT,*) 'F08CTF Example Program Results'
WRITE (NOUT,*)
*      Skip heading in data file
READ (NIN,*)
READ (NIN,*) M, N
```

```

      IF (M.LE.MMAX .AND. N.LE.NMAX .AND. M.GE.N) THEN
*
*   Read A from data file
*
      READ (NIN,*) ((A(I,J),J=1,N),I=1,M)
*
*   Compute the QL factorization of A
*
      CALL ZGEQLF(M,N,A,LDA,TAU,WORK,LWORK,INFO)
*
*   Form the leading N columns of Q explicitly
*
      CALL ZUNGQL(M,N,N,A,LDA,TAU,WORK,LWORK,INFO)
*
*   Form the heading for X04DBF
*
      WRITE (TITLE,99999) N
*
*   Print the leading N columns of Q
*
      IFAIL = 0
      CALL X04DBF('General',' ',M,N,A,LDA,'Bracketed','F7.4',TITLE,
+              'Integer',RLABS,'Integer',CLABS,80,0,IFAIL)
*
      ELSE
        WRITE (NOUT,*) 'MMAX and/or NMAX is too small, and/or M.LT.N'
      END IF
      STOP
*
99999 FORMAT ('The leading ',I4,' columns of Q')
      END

```

## 9.2 Program Data

F08CTF Example Program Data

```

      6              4                      :Values of M and N

      ( 0.96,-0.81) (-0.03, 0.96) (-0.91, 2.06) (-0.05, 0.41)
      (-0.98, 1.98) (-1.20, 0.19) (-0.66, 0.42) (-0.81, 0.56)
      ( 0.62,-0.46) ( 1.01, 0.02) ( 0.63,-0.17) (-1.11, 0.60)
      (-0.37, 0.38) ( 0.19,-0.54) (-0.98,-0.36) ( 0.22,-0.20)
      ( 0.83, 0.51) ( 0.20, 0.01) (-0.17,-0.46) ( 1.47, 1.59)
      ( 1.08,-0.28) ( 0.20,-0.12) (-0.07, 1.23) ( 0.26, 0.26) :End of matrix A

```

## 9.3 Program Results

F08CTF Example Program Results

```

The leading      4 columns of Q
                1          2          3          4
1 ( 0.2810, 0.5020) (-0.2051,-0.1092) ( 0.3083,-0.6874) ( 0.0181,-0.1483)
2 ( 0.2707,-0.3296) ( 0.5711, 0.0432) ( 0.2251,-0.1313) ( 0.2930,-0.2025)
3 (-0.2864,-0.0094) (-0.5416, 0.0454) (-0.2062, 0.0691) ( 0.4015,-0.2170)
4 ( 0.2262,-0.3854) (-0.3387, 0.2228) ( 0.3259, 0.1178) (-0.0796, 0.0723)
5 ( 0.0341,-0.0760) ( 0.0098,-0.0712) ( 0.0753, 0.1412) (-0.5317,-0.5751)
6 (-0.3936,-0.2083) (-0.1296, 0.3691) ( 0.0264,-0.4134) (-0.0940,-0.0940)

```