

NAG Fortran Library Routine Document

F08CSF (ZGEQLF)

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

F08CSF (ZGEQLF) computes a *QL* factorization of a complex m by n matrix A .

2 Specification

```
SUBROUTINE F08CSF (M, N, A, LDA, TAU, WORK, LWORK, INFO)
INTEGER          M, N, LDA, LWORK, INFO
complex*16     A(LDA,*), TAU(*), WORK(*)
```

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

3 Description

F08CSF (ZGEQLF) forms the *QL* factorization of an arbitrary rectangular real m by n matrix.

If $m \geq n$, the factorization is given by:

$$A = Q \begin{pmatrix} 0 \\ L \end{pmatrix},$$

where L is an n by n lower triangular matrix and Q is an m by m unitary matrix. If $m < n$ the factorization is given by

$$A = QL,$$

where L is an m by n lower trapezoidal matrix and Q is again an m by m unitary matrix. In the case where $m > n$ the factorization can be expressed as

$$A = (Q_1 \quad Q_2) \begin{pmatrix} 0 \\ L \end{pmatrix} = Q_2 L,$$

where Q_1 consists of the first $m - n$ columns of Q , and Q_2 the remaining n columns.

The matrix Q is not formed explicitly but is represented as a product of $\min(m, n)$ elementary reflectors (see Section 3.3.6 in the F08 Chapter Introduction for details). Routines are provided to work with Q in this representation (see Section 8).

Note also that for any $k < n$, the information returned in the last k columns of the array A represents a *QL* factorization of the last k columns of the original matrix A .

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 A .
Constraint: $M \geq 0$.
- 2: N – INTEGER *Input*
On entry: n , the number of columns of the matrix A .
Constraint: $N \geq 0$.
- 3: A(LDA,*) – **complex*16** array *Input/Output*
Note: the second dimension of the array A must be at least $\max(1, N)$.
On entry: the m by n matrix A .
On exit: if $m \geq n$, the lower triangle of the subarray $A(m - n + 1 : m, 1 : n)$ contains the n by n lower triangular matrix L .
 If $m \leq n$, the elements on and below the $(n - m)$ th superdiagonal contain the m by n lower trapezoidal matrix L . The remaining elements, with the array TAU, represent the unitary matrix Q as a product of elementary reflectors (see Section 3.3.6 in the F08 Chapter Introduction).
- 4: LDA – INTEGER *Input*
On entry: the first dimension of the array A as declared in the (sub)program from which F08CSF (ZGEQLF) is called.
Constraint: $LDA \geq \max(1, M)$.
- 5: TAU(*) – **complex*16** array *Output*
Note: the dimension of the array TAU must be at least $\max(1, \min(M, N))$.
On exit: the scalar factors of the elementary reflectors (see Section 8).
- 6: 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.
- 7: LWORK – INTEGER *Input*
On entry: the dimension of the array WORK as declared in the (sub)program from which F08CSF (ZGEQLF) 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: $LWORK \geq \max(1, N)$.
- 8: 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 factorization is the exact factorization of a nearby matrix $(A + E)$, where

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

and ϵ is the *machine precision*.

8 Further Comments

The total number of real floating-point operations is approximately $\frac{8}{3}n^2(3m - n)$ if $m \geq n$ or $\frac{8}{3}m^2(3n - m)$ if $m < n$.

To form the unitary matrix Q F08CSF (ZGEQLF) may be followed by a call to F08CTF (ZUNGQL):

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

but note that the second dimension of the array A must be at least M, which may be larger than was required by F08CSF (ZGEQLF).

When $m \geq n$, it is often only the first n columns of Q that are required, and they may be formed by the call:

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

To apply Q to an arbitrary complex rectangular matrix C , F08CSF (ZGEQLF) may be followed by a call to F08CUF (ZUNMQL). For example,

```
CALL ZUNMQL ('Left', 'Conjugate Transpose', M,P,MIN(M,N),A,LDA,TAU,
+ C,LDC,WORK,LWORK,INFO)
```

forms $C = Q^H C$, where C is m by p .

The real analogue of this routine is F08CEF (DGEQLF).

9 Example

This example solves the linear least-squares problems

$$\min_x \|b_j - Ax_j\|_2, j = 1, 2$$

for x_1 and x_2 , where b_j is the j th column of the matrix B ,

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

and

$$B = \begin{pmatrix} -2.09 + 1.93i & 3.26 - 2.70i \\ 3.34 - 3.53i & -6.22 + 1.16i \\ -4.94 - 2.04i & 7.94 - 3.13i \\ 0.17 + 4.23i & 1.04 - 4.26i \\ -5.19 + 3.63i & -2.31 - 2.12i \\ 0.98 + 2.53i & -1.39 - 4.05i \end{pmatrix}.$$

The solution is obtained by first obtaining a *QL* factorization of the matrix *A*.

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

```

*      F08CSF Example Program Text
*      Mark 21 Release. NAG Copyright 2004.
*      .. Parameters ..
INTEGER          NIN, NOUT
PARAMETER       (NIN=5,NOUT=6)
INTEGER          MMAX, NB, NMAX, NRHSMX
PARAMETER       (MMAX=8,NB=64,NMAX=8,NRHSMX=2)
INTEGER          LDA, LDB, LWORK
PARAMETER       (LDA=MMAX,LDB=MMAX,LWORK=NB*NMAX)
*      .. Local Scalars ..
INTEGER          I, IFAIL, INFO, J, M, N, NRHS
*      .. Local Arrays ..
COMPLEX *16     A(LDA,NMAX), B(LDB,NRHSMX), TAU(NMAX),
+              WORK(LWORK)
DOUBLE PRECISION RNORM(NRHSMX)
CHARACTER       CLABS(1), RLABS(1)
*      .. External Functions ..
DOUBLE PRECISION DZNRM2
EXTERNAL        DZNRM2
*      .. External Subroutines ..
EXTERNAL        X04DBF, ZGEQLF, ZTRTRS, ZUNMQL
*      .. Executable Statements ..
WRITE (NOUT,*) 'F08CSF Example Program Results'
WRITE (NOUT,*)
*      Skip heading in data file
READ (NIN,*)
READ (NIN,*) M, N, NRHS
IF (M.LE.MMAX .AND. N.LE.NMAX .AND. M.GE.N .AND. NRHS.LE.NRHSMX)
+  THEN
*
*      Read A and B from data file
*
READ (NIN,*) ((A(I,J),J=1,N),I=1,M)
READ (NIN,*) ((B(I,J),J=1,NRHS),I=1,M)
*
*      Compute the QL factorization of A
*
CALL ZGEQLF(M,N,A,LDA,TAU,WORK,LWORK,INFO)
*
*      Compute C = (C1) = (Q**H)*B, storing the result in B
*              (C2)
*
CALL ZUNMQL('Left','Conjugate Transpose',M,NRHS,N,A,LDA,TAU,B,
+          LDB,WORK,LWORK,INFO)
*
*      Compute least-squares solutions by backsubstitution in
*      L*X = C2
*
CALL ZTRTRS('Lower','No transpose','Non-Unit',N,NRHS,A(M-N+1,1)
+          ,LDA,B(M-N+1,1),LDB,INFO)
*
IF (INFO.GT.0) THEN
WRITE (NOUT,*)

```

```

+       'The lower triangular factor, L, of A is singular, '
      WRITE (NOUT,*)
+       'the least squares solution could not be computed'
      GO TO 40
      END IF
*
*       Print least-squares solution(s)
*
      IFAIL = 0
      CALL X04DBF('General', ' ', N, NRHS, B(M-N+1,1), LDB, 'Bracketed',
+              'F7.4', 'Least-squares solution(s)', 'Integer', RLABS,
+              'Integer', CLABS, 80, 0, IFAIL)
*
*       Compute and print estimates of the square roots of the residual
*       sums of squares
*
      DO 20 J = 1, NRHS
        RNORM(J) = DZNRM2(M-N, B(1,J), 1)
20     CONTINUE
*
      WRITE (NOUT,*)
      WRITE (NOUT,*)
+     'Square root(s) of the residual sum(s) of squares'
      WRITE (NOUT,99999) (RNORM(J), J=1, NRHS)
      ELSE
        WRITE (NOUT,*)
+       'One or more of MMAX, NMAX or NRHSMX is too small, ',
+       'and/or M.LT.N'
      END IF
40    CONTINUE
      STOP
*
99999 FORMAT (3X,1P,7E11.2)
      END

```

9.2 Program Data

F08CSF Example Program Data

```

6           4           2           :Values of M, N and NRHS

( 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

(-2.09, 1.93) ( 3.26,-2.70)
( 3.34,-3.53) (-6.22, 1.16)
(-4.94,-2.04) ( 7.94,-3.13)
( 0.17, 4.23) ( 1.04,-4.26)
(-5.19, 3.63) (-2.31,-2.12)
( 0.98, 2.53) (-1.39,-4.05) :End of matrix B

```

9.3 Program Results

F08CSF Example Program Results

Least-squares solution(s)

```

           1           2
1  (-0.5044,-1.2179) ( 0.7629, 1.4529)
2  (-2.4281, 2.8574) ( 5.1570,-3.6089)
3  ( 1.4872,-2.1955) (-2.6518, 2.1203)
4  ( 0.4537, 2.6904) (-2.7606, 0.3318)

```

Square root(s) of the residual sum(s) of squares

```

6.88E-02  1.87E-01

```