NAG Fortran Library Routine Document

F08AVF (CGELQF/ZGELQF)

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

F08AVF (CGELQF/ZGELQF) computes the LQ factorization of a complex m by n matrix.

2 Specification

```
SUBROUTINE F08AVF(M, N, A, LDA, TAU, WORK, LWORK, INFO)
ENTRY      cgelqf (M, N, A, LDA, TAU, WORK, LWORK, INFO)
INTEGER           M, N, LDA, LWORK, INFO
complex           A(LDA,*), TAU(*), WORK(*)
```

The ENTRY statement enables the routine to be called by its LAPACK name.

3 Description

This routine forms the LQ factorization of an arbitrary rectangular complex m by n matrix. No pivoting is performed.

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

$$A = (\,L \quad 0\,)Q$$

where L is an m by m lower triangular matrix (with real diagonal elements) and Q is an n by n unitary matrix. It is sometimes more convenient to write the factorization as

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

which reduces to

$$A = LQ_1,$$

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

If $m > n$, L is trapezoidal, and the factorization can be written

$$A = \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} Q$$

where L_1 is lower triangular and L_2 is rectangular.

The LQ factorization of A is essentially the same as the QR factorization of A^H, since

$$A = (\,L \quad 0\,)Q \Leftrightarrow A^H = Q^H \begin{pmatrix} L^H \\ 0 \end{pmatrix}.$$

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

Note also that for any $k < m$, the information returned in the first k rows of the array A represents an LQ factorization of the first k rows of the original matrix A.

4 References

None.

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*** 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 $m \leq n$, the elements above the diagonal are overwritten by details of the unitary matrix Q and the lower triangle is overwritten by the corresponding elements of the m by m lower triangular matrix L.

If $m > n$, the strictly upper triangular part is overwritten by details of the unitary matrix Q and the remaining elements are overwritten by the corresponding elements of the m by n lower trapezoidal matrix L.

The diagonal elements of L are real.

4: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08AVF (CGELQF/ZGELQF) is called.

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

5: TAU($*$) – ***complex*** array *Output*

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

On exit: further details of the unitary matrix Q.

6: WORK($*$) – ***complex*** 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 optimum performance.

7: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08AVF (CGELQF/ZGELQF) is called, unless LWORK $= -1$, in which case a workspace query is assumed and the routine only calculates the optimal dimension of WORK (using the formula given below).

Suggested value: for optimum performance LWORK should be at least $M \times nb$, where nb is the ***blocksize***.

Constraint: $LWORK \geq \max(1, M)$ or LWORK $= -1$.

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 ith 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 $\frac{8}{3}m^2(3n - m)$ if $m \le n$ or $\frac{8}{3}n^2(3m - n)$ if $m > n$.

To form the unitary matrix Q this routine may be followed by a call to F08AWF (CUNGLQ/ZUNGLQ):

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

but note that the first dimension of the array A, specified by the parameter LDA, must be at least N, which may be larger than was required by F08AVF (CGELQF/ZGELQF).

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

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

To apply Q to an arbitrary complex rectangular matrix C, this routine may be followed by a call to F08AXF (CUNMLQ/ZUNMLQ). For example,

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

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

The real analogue of this routine is F08AHF (SGELQF/DGELQF).

9 Example

To find the minimum-norm solutions of the under-determined systems of linear equations

$$Ax_1 = b_1 \quad \text{and} \quad Ax_2 = b_2$$

where b_1 and b_2 are the columns of the matrix B,

$$A = \begin{pmatrix} 0.28 - 0.36i & 0.50 - 0.86i & -0.77 - 0.48i & 1.58 + 0.66i \\ -0.50 - 1.10i & -1.21 + 0.76i & -0.32 - 0.24i & -0.27 - 1.15i \\ 0.36 - 0.51i & -0.07 + 1.33i & -0.75 + 0.47i & -0.08 + 1.01i \end{pmatrix}$$

and

$$B = \begin{pmatrix} -1.35 + 0.19i & 4.83 - 2.67i \\ 9.41 - 3.56i & -7.28 + 3.34i \\ -7.57 + 6.93i & 0.62 + 4.53i \end{pmatrix}.$$

9.1 Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F08AVF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, NMAX, LDA, LDB, NRHMAX, LWORK
      PARAMETER        (MMAX=8,NMAX=8,LDA=MMAX,LDB=NMAX,NRHMAX=NMAX,
     +                 LWORK=64*NMAX)
      complex          ZERO, ONE
      PARAMETER        (ZERO=(0.0e0,0.0e0),ONE=(1.0e0,0.0e0))
*     .. Local Scalars ..
      INTEGER          I, IFAIL, INFO, J, M, N, NRHS
*     .. Local Arrays ..
      complex          A(LDA,NMAX), B(LDB,NRHMAX), TAU(NMAX),
     +                 WORK(LWORK)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F06THF, X04DBF, cgelqf, ctrsm, cunmlq
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F08AVF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) M, N, NRHS
      IF (M.LE.MMAX .AND. N.LE.NMAX .AND. M.LE.N .AND. NRHS.LE.NRHMAX)
     +    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 LQ factorization of A
*
         CALL cgelqf(M,N,A,LDA,TAU,WORK,LWORK,INFO)
*
*        Solve L*Y = B, storing the result in B
*
         CALL ctrsm('Left','Lower','No transpose','Non-Unit',M,NRHS,ONE,
     +              A,LDA,B,LDB)
*
*        Set rows (M+1) to N of B to zero
*
         IF (M.LT.N) CALL F06THF('General',N-M,NRHS,ZERO,ZERO,B(M+1,1),
     +                           LDB)
*
*        Compute minimum-norm solution X = (Q**H)*B in B
*
         CALL cunmlq('Left','Conjugate transpose',N,NRHS,M,A,LDA,TAU,B,
     +               LDB,WORK,LWORK,INFO)
*
*        Print minimum-norm solution(s)
*
         WRITE (NOUT,*)
         IFAIL = 0
*
         CALL X04DBF('General',' ',N,NRHS,B,LDB,'Bracketed','F7.4',
     +               'Minimum-norm solution(s)','Integer',RLABS,
     +               'Integer',CLABS,80,0,IFAIL)
*
      END IF
      STOP
      END
```

9.2 Program Data

```
F08AVF Example Program Data
  3  4  2                                                      :Values of M, N and NRHS
 ( 0.28,-0.36) ( 0.50,-0.86) (-0.77,-0.48) ( 1.58, 0.66)
 (-0.50,-1.10) (-1.21, 0.76) (-0.32,-0.24) (-0.27,-1.15)
 ( 0.36,-0.51) (-0.07, 1.33) (-0.75, 0.47) (-0.08, 1.01)   :End of matrix A
 (-1.35, 0.19) ( 4.83,-2.67)
 ( 9.41,-3.56) (-7.28, 3.34)
 (-7.57, 6.93) ( 0.62, 4.53)                               :End of matrix B
```

9.3 Program Results

```
 F08AVF Example Program Results

 Minimum-norm solution(s)
                   1                  2
 1  (-2.8501, 6.4683) (-1.1682,-1.8886)
 2  ( 1.6264,-0.7799) ( 2.8377, 0.7654)
 3  ( 6.9290, 4.6481) (-1.7610,-0.7041)
 4  ( 1.4048, 3.2400) ( 1.0518,-1.6365)
```