# NAG Library Routine Document

# F08CVF (ZGERQF)

**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

F08CVF (ZGERQF) computes an RQ factorization of a complex $m$ by $n$ matrix $A$.

## 2 Specification

```
SUBROUTINE F08CVF (M, N, A, LDA, TAU, WORK, LWORK, INFO)
INTEGER               M, N, LDA, LWORK, INFO
COMPLEX (KIND=nag_wp) A(LDA,*), TAU(*), WORK(max(1,LWORK))
```

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

## 3 Description

F08CVF (ZGERQF) forms the $RQ$ factorization of an arbitrary rectangular real $m$ by $n$ matrix. If $m \leq n$, the factorization is given by

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

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

$$A = RQ,$$

where $R$ is an $m$ by $n$ upper trapezoidal matrix and $Q$ is again an $n$ by $n$ unitary matrix. In the case where $m < n$ the factorization can be expressed as

$$A = \begin{pmatrix} 0 & R \end{pmatrix} \begin{pmatrix} Q_1 \\ Q_2 \end{pmatrix} = RQ_2,$$

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

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 9).

## 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 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 Arguments

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*: $\mathrm{N} \geq 0$.

3: $\mathrm{A}(\mathrm{LDA}, *)$ – COMPLEX (KIND=nag_wp) array *Input/Output*

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

*On entry*: the $m$ by $n$ matrix $A$.

*On exit*: if $m \leq n$, the upper triangle of the subarray $\mathrm{A}(1:m, n-m+1:n)$ contains the $m$ by $m$ upper triangular matrix $R$.

If $m \geq n$, the elements on and above the $(m-n)$th subdiagonal contain the $m$ by $n$ upper trapezoidal matrix $R$; the remaining elements, with the array TAU, represent the unitary matrix $Q$ as a product of $\min(m, n)$ 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 F08CVF (ZGERQF) is called.

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

5: $\mathrm{TAU}(*)$ – COMPLEX (KIND=nag_wp) array *Output*

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

*On exit*: the scalar factors of the elementary reflectors.

6: $\mathrm{WORK}(\max(1, \mathrm{LWORK}))$ – COMPLEX (KIND=nag_wp) array *Workspace*

*On exit*: if $\mathrm{INFO} = 0$, the real part of $\mathrm{WORK}(1)$ 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 F08CVF (ZGERQF) is called.

If $\mathrm{LWORK} = -1$, 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, $\mathrm{LWORK} \geq \mathrm{M} \times nb$, where $nb$ is the optimal ***block size***.

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

8: INFO – INTEGER *Output*

*On exit*: $\mathrm{INFO} = 0$ unless the routine detects an error (see Section 6).

## 6 Error Indicators and Warnings

$\mathrm{INFO} < 0$

If $\mathrm{INFO} = -i$, argument $i$ 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 $\epsilon$ is the ***machine precision***.

## 8 Parallelism and Performance

F08CVF (ZGERQF) 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 floating-point operations is approximately $\frac{2}{3}m^2(3n - m)$ if $m \leq n$, or $\frac{2}{3}n^2(3m - n)$ if $m > n$.

To form the unitary matrix $Q$ F08CVF (ZGERQF) may be followed by a call to F08CWF (ZUNGRQ):

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

but note that the first dimension of the array A must be at least N, which may be larger than was required by F08CVF (ZGERQF). When $m \leq n$, it is often only the first $m$ rows of $Q$ that are required and they may be formed by the call:

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

To apply $Q$ to an arbitrary real rectangular matrix $C$, F08CVF (ZGERQF) may be followed by a call to F08CXF (ZUNMRQ). For example:

```
CALL ZUNMRQ('Left','C',N,P,MIN(M,N),A,LDA,TAU,C,LDC, &
               WORK,LWORK,INFO)
```

forms $C = Q^{\mathrm{H}}C$, where $C$ is $n$ by $p$.

The real analogue of this routine is F08CHF (DGERQF).

## 10 Example

This example finds the minimum norm solution to the underdetermined equations

$$Ax = b$$

where

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

and

$$b = \begin{pmatrix} -1.35 + 0.19i \\ 9.41 - 3.56i \\ -7.57 + 6.93i \end{pmatrix}.$$

The solution is obtained by first obtaining an $RQ$ 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.

### 10.1 Program Text

```
    Program f08cvfe

!     F08CVF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, zgerqf, ztrtrs, zunmrq
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Complex (Kind=nag_wp), Parameter :: zero = (0.0_nag_wp,0.0_nag_wp)
      Integer, Parameter               :: nb = 64, nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, info, lda, lwork, m, n
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:), tau(:), work(:),     &
                                          x(:)
!     .. Executable Statements ..
      Write (nout,*) 'F08CVF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n
      lda = m
      lwork = nb*m
      Allocate (a(lda,n),b(m),tau(m),work(lwork),x(n))

!     Read the matrix A and the vector b from data file

      Read (nin,*)(a(i,1:n),i=1,m)
      Read (nin,*) b(1:m)

!     Compute the RQ factorization of A
!     The NAG name equivalent of zgerqf is f08cvf
      Call zgerqf(m,n,a,lda,tau,work,lwork,info)

!     Copy the m-element vector b into elements x(n-m+1), ..., x(n) of x

      x(n-m+1:n) = b(1:m)

!     Solve R*y2 = b, storing the result in x2
!     The NAG name equivalent of ztrtrs is f07tsf
      Call ztrtrs('Upper','No transpose','Non-Unit',m,1,a(1,n-m+1),lda,        &
        x(n-m+1),m,info)

      If (info>0) Then
        Write (nout,*) 'The upper triangular factor, R, of A is singular, '
        Write (nout,*) 'the least squares solution could not be computed'
      Else

!       Set y1 to zero (stored in x(1:n-m))

        x(1:n-m) = zero

!       Compute minimum-norm solution x = (Q**H)*y
!       The NAG name equivalent of zunmrq is f08cxf
        Call zunmrq('Left','Conjugate transpose',n,1,m,a,lda,tau,x,n,work,     &
          lwork,info)

!       Print minimum-norm solution

        Write (nout,*) 'Minimum-norm solution'
        Write (nout,99999) x(1:n)
      End If

99999 Format (4(' (',F8.4,',',F8.4,')',:))
    End Program f08cvfe
```

### 10.2 Program Data

```
F08CVF Example Program Data

   3            4                              :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)
 ( 9.41,-3.56)
 (-7.57, 6.93)                                           :End of vector b
```

### 10.3 Program Results

```
 F08CVF Example Program Results

 Minimum-norm solution
 ( -2.8501,  6.4683) (  1.6264, -0.7799) (  6.9290,  4.6481) (  1.4048,  3.2400)
```