NAG Library Routine Document

F08ASF (ZGEQRF)

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

F08ASF (ZGEQRF) computes the QR factorization of a complex m by n matrix.

2 Specification

```
SUBROUTINE F08ASF (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 ***zgeqrf***.

3 Description

F08ASF (ZGEQRF) forms the QR factorization of an arbitrary rectangular complex m by n matrix. No pivoting is performed.

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

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

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

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

which reduces to

$$A = Q_1 R,$$

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

If $m < n$, R is trapezoidal, and the factorization can be written

$$A = Q\begin{pmatrix} R_1 & R_2 \end{pmatrix},$$

where R_1 is upper triangular and R_2 is rectangular.

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 $k < n$, the information returned in the first k columns of the array A represents a QR factorization of the first k columns of the original matrix A.

4 References

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

2: N – INTEGER *Input*

On entry: n, the number of columns of the matrix A.

Constraint: $\text{N} \geq 0$.

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

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

On entry: the m by n matrix A.

On exit: if $m \geq n$, the elements below the diagonal are overwritten by details of the unitary matrix Q and the upper triangle is overwritten by the corresponding elements of the n by n upper triangular matrix R.

If $m < n$, the strictly lower 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 upper trapezoidal matrix R.

The diagonal elements of R are real.

4: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08ASF (ZGEQRF) is called.

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

5: TAU($*$) – COMPLEX (KIND=nag_wp) array *Output*

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

On exit: further details of the unitary matrix Q.

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

On exit: if $\text{INFO} = 0$, 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 F08ASF (ZGEQRF) is called.

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

Constraint: $\text{LWORK} \geq \max(1, \text{N})$ or $\text{LWORK} = -1$.

8: INFO – INTEGER *Output*

On exit: $\text{INFO} = 0$ 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 $= -i$, 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 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 F08ASF (ZGEQRF) may be followed by a call to F08ATF (ZUNGQR):

```
CALL ZUNGQR(M,M,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 F08ASF (ZGEQRF).

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

To apply Q to an arbitrary complex rectangular matrix C, F08ASF (ZGEQRF) may be followed by a call to F08AUF (ZUNMQR). For example,

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

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

To compute a QR factorization with column pivoting, use F08BSF (ZGEQPF).

The real analogue of this routine is F08AEF (DGEQRF).

9 Example

This example solves the linear least squares problems

$$\text{minimize} \|Ax_i - b_i\|_2^2, \qquad i = 1, 2$$

where b_1 and b_2 are the columns 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

$$B = \begin{pmatrix} -1.54 + 0.76i & 3.17 - 2.09i \\ 0.12 - 1.92i & -6.53 + 4.18i \\ -9.08 - 4.31i & 7.28 + 0.73i \\ 7.49 + 3.65i & 0.91 - 3.97i \\ -5.63 - 2.12i & -5.46 - 1.64i \\ 2.37 + 8.03i & -2.84 - 5.86i \end{pmatrix}.$$

9.1 Program Text

```
    Program f08asfe

!     F08ASF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dznrm2, nag_wp, x04dbf, zgeqrf, ztrtrs, zunmqr
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nb = 64, nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, j, lda, ldb, lwork,  &
                                          m, n, nrhs
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:), tau(:), work(:)
      Real (Kind=nag_wp), Allocatable  :: rnorm(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F08ASF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n, nrhs
      lda = m
      ldb = m
      lwork = nb*n
      Allocate (a(lda,n),b(ldb,nrhs),tau(n),work(lwork),rnorm(nrhs))

!     Read A and B from data file

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

!     Compute the QR factorization of A
!     The NAG name equivalent of zgeqrf is f08asf
      Call zgeqrf(m,n,a,lda,tau,work,lwork,info)

!     Compute C = (C1) = (Q**H)*B, storing the result in B
!                 (C2)
!     The NAG name equivalent of zunmqr is f08auf
      Call zunmqr('Left','Conjugate transpose',m,nrhs,n,a,lda,tau,b,ldb,work, &
        lwork,info)

!     Compute least-squares solutions by backsubstitution in
!     R*X = C1
!     The NAG name equivalent of ztrtrs is f07tsf
      Call ztrtrs('Upper','No transpose','Non-Unit',n,nrhs,a,lda,b,ldb,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

!       Print least-squares solutions
```

```
!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04dbf('General',' ',n,nrhs,b,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
!       The NAG name equivalent of dznrm2 is f06jjf
        Do j = 1, nrhs
          rnorm(j) = dznrm2(m-n,b(n+1,j),1)
        End Do

        Write (nout,*)
        Write (nout,*) 'Square root(s) of the residual sum(s) of squares'
        Write (nout,99999) rnorm(1:nrhs)
      End If

99999 Format (3X,1P,7E11.2)
    End Program f08asfe
```

9.2 Program Data

```
F08ASF 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

```
 F08ASF 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
```