NAG Library Routine Document

F08BSF (ZGEQPF)

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

F08BSF (ZGEQPF) computes the QR factorization, with column pivoting, of a complex m by n matrix.

2 Specification

```
SUBROUTINE F08BSF(M, N, A, LDA, JPVT, TAU, WORK, RWORK, INFO)
INTEGER            M, N, LDA, JPVT(*), INFO
double precision   RWORK(2*N)
complex*16         A(LDA,*), TAU(min(M,N)), WORK(N)
```

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

3 Description

F08BSF (ZGEQPF) forms the QR factorization, with column pivoting, of an arbitrary rectangular complex m by n matrix.

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

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

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

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

which reduces to

$$AP = Q_1R,$$

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

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

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 permuted matrix AP.

The routine allows specified columns of A to be moved to the leading columns of AP at the start of the factorization and fixed there. The remaining columns are free to be interchanged so that at the ith stage the pivot column is chosen to be the column which maximizes the 2-norm of elements i to m over columns i to n.

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

2: N – INTEGER *Input*

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

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

3: A(LDA,$*$) – ***complex*16*** array *Input/Output*

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

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 F08BSF (ZGEQPF) is called.

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

5: JPVT($*$) – INTEGER array *Input/Output*

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

On entry: if $\mathrm{JPVT}(i) \neq 0$, then the ith column of A is moved to the beginning of AP before the decomposition is computed and is fixed in place during the computation. Otherwise, the ith column of A is a free column (i.e., one which may be interchanged during the computation with any other free column).

On exit: details of the permutation matrix P. More precisely, if $\mathrm{JPVT}(i) = k$, then the kth column of A is moved to become the ith column of AP; in other words, the columns of AP are the columns of A in the order $\mathrm{JPVT}(1), \mathrm{JPVT}(2), \ldots, \mathrm{JPVT}(n)$.

6: TAU($\min(\mathrm{M}, \mathrm{N})$) – ***complex*16*** array *Output*

On exit: further details of the unitary matrix Q.

7: WORK(N) – ***complex*16*** array *Workspace*

8: RWORK($2 \times \mathrm{N}$) – ***double precision*** array *Workspace*

9: INFO – INTEGER *Output*

On exit: $\mathrm{INFO} = 0$ 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 F08BSF (ZGEQPF) 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 F08BSF (ZGEQPF).

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, F08BSF (ZGEQPF) 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 without column pivoting, use F08ASF (ZGEQRF).

The real analogue of this routine is F08BEF (DGEQPF).

9 Example

This example solves the linear least-squares problems

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

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

$$A = \begin{pmatrix} 0.47 - 0.34i & -0.40 + 0.54i & 0.60 + 0.01i & 0.80 - 1.02i \\ -0.32 - 0.23i & -0.05 + 0.20i & -0.26 - 0.44i & -0.43 + 0.17i \\ 0.35 - 0.60i & -0.52 - 0.34i & 0.87 - 0.11i & -0.34 - 0.09i \\ 0.89 + 0.71i & -0.45 - 0.45i & -0.02 - 0.57i & 1.14 - 0.78i \\ -0.19 + 0.06i & 0.11 - 0.85i & 1.44 + 0.80i & 0.07 + 1.14i \end{pmatrix}$$

and

$$B = \begin{pmatrix} -0.85 - 1.63i & 2.49 + 4.01i \\ -2.16 + 3.52i & -0.14 + 7.98i \\ 4.57 - 5.71i & 8.36 - 0.28i \\ 6.38 - 7.40i & -3.55 + 1.29i \\ 8.41 + 9.39i & -6.72 + 5.03i \end{pmatrix}.$$

Here A is approximately rank-deficient, and hence it is preferable to use F08BSF (ZGEQPF) rather than F08ASF (ZGEQRF).

9.1 Program Text

```
*     F08BSF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, NMAX, LDA, LDB, LDX, NRHMAX, LWORK
      PARAMETER        (MMAX=8,NMAX=8,LDA=MMAX,LDB=MMAX,LDX=MMAX,
     +                 NRHMAX=NMAX,LWORK=64*NMAX)
      COMPLEX *16      ZERO
      PARAMETER        (ZERO=(0.0D0,0.0D0))
*     .. Local Scalars ..
      DOUBLE PRECISION TOL
      INTEGER          I, IFAIL, INFO, J, K, M, N, NRHS
*     .. Local Arrays ..
      COMPLEX *16      A(LDA,NMAX), B(LDB,NRHMAX), TAU(NMAX),
     +                 WORK(LWORK), X(LDX,NRHMAX)
      DOUBLE PRECISION RWORK(2*NMAX)
      INTEGER          JPVT(NMAX)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F06DBF, F06HBF, X04DBF, ZGEQPF, ZTRSV, ZUNMQR
*     .. Intrinsic Functions ..
      INTRINSIC        ABS
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F08BSF 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.GE.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)
*
*        Initialize JPVT to be zero so that all columns are free
*
         CALL F06DBF(N,0,JPVT,1)
*
*        Compute the QR factorization of A
*
         CALL ZGEQPF(M,N,A,LDA,JPVT,TAU,WORK,RWORK,INFO)
*
*        Choose TOL to reflect the relative accuracy of the input data
*
         TOL = 0.01D0
*
*        Determine which columns of R to use
*
         DO 20 K = 1, N
            IF (ABS(A(K,K)).LE.TOL*ABS(A(1,1))) GO TO 40
   20    CONTINUE
*
*        Compute C = (Q**H)*B, storing the result in B
*
```

```
   40    K = K - 1
*
         CALL ZUNMQR('Left','Conjugate Transpose',M,NRHS,K,A,LDA,TAU,B,
     +               LDB,WORK,LWORK,INFO)
*
*        Compute least-squares solution by backsubstitution in R*B = C
*
         DO 60 I = 1, NRHS
*
            CALL ZTRSV('Upper','No transpose','Non-Unit',K,A,LDA,B(1,I),
     +                 1)
*
*           Set the unused elements of the I-th solution vector to zero
*
            CALL F06HBF(N-K,ZERO,B(K+1,I),1)
*
   60    CONTINUE
*
*        Unscramble the least-squares solution stored in B
*
         DO 100 I = 1, N
            DO 80 J = 1, NRHS
               X(JPVT(I),J) = B(I,J)
   80       CONTINUE
  100    CONTINUE
*
*        Print least-squares solution
*
         WRITE (NOUT,*)
         IFAIL = 0
*
         CALL X04DBF('General',' ',N,NRHS,X,LDX,'Bracketed','F7.4',
     +               'Least-squares solution','Integer',RLABS,'Integer',
     +               CLABS,80,0,IFAIL)
*
      END IF
      END
```

9.2 Program Data

```
F08BSF Example Program Data
  5  4  2                                             :Values of M, N and NRHS
 ( 0.47,-0.34) (-0.40, 0.54) ( 0.60, 0.01) ( 0.80,-1.02)
 (-0.32,-0.23) (-0.05, 0.20) (-0.26,-0.44) (-0.43, 0.17)
 ( 0.35,-0.60) (-0.52,-0.34) ( 0.87,-0.11) (-0.34,-0.09)
 ( 0.89, 0.71) (-0.45,-0.45) (-0.02,-0.57) ( 1.14,-0.78)
 (-0.19, 0.06) ( 0.11,-0.85) ( 1.44, 0.80) ( 0.07, 1.14)   :End of matrix A
 (-0.85,-1.63) ( 2.49, 4.01)
 (-2.16, 3.52) (-0.14, 7.98)
 ( 4.57,-5.71) ( 8.36,-0.28)
 ( 6.38,-7.40) (-3.55, 1.29)
 ( 8.41, 9.39) (-6.72, 5.03)                               :End of matrix B
```

9.3 Program Results

```
 F08BSF Example Program Results

 Least-squares solution
                   1                  2
 1  ( 0.0000, 0.0000) ( 0.0000, 0.0000)
 2  ( 2.6925, 8.0446) (-2.0563,-2.9759)
 3  ( 2.7602, 2.5455) ( 1.0588, 1.4635)
 4  ( 2.7383, 0.5123) (-1.4150, 0.2982)
```