

NAG Library Routine Document

F08BEF (DGEQPF)

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

F08BEF (DGEQPF) computes the QR factorization, with column pivoting, of a real m by n matrix.

2 Specification

```
SUBROUTINE F08BEF (M, N, A, LDA, JPVT, TAU, WORK, INFO)
  INTEGER          M, N, LDA, JPVT(*), INFO
  REAL (KIND=nag_wp) A(LDA,*), TAU(min(M,N)), WORK(3*N)
```

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

3 Description

F08BEF (DGEQPF) forms the QR factorization, with column pivoting, of an arbitrary rectangular real 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, Q is an m by m orthogonal matrix and P is an n by n permutation matrix. It is sometimes more convenient to write the factorization as

$$AP = (Q_1 \quad Q_2) \begin{pmatrix} R \\ 0 \end{pmatrix},$$

which reduces to

$$AP = Q_1 R,$$

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

$$AP = Q (R_1 \quad R_2),$$

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 9).

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 i th 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 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: $N \geq 0$.
- 3: A(LDA,*) – REAL (KIND=nag_wp) 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 elements below the diagonal are overwritten by details of the orthogonal 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 orthogonal matrix Q and the remaining elements are overwritten by the corresponding elements of the m by n upper trapezoidal matrix R .
- 4: LDA – INTEGER *Input*
On entry: the first dimension of the array A as declared in the (sub)program from which F08BEF (DGEQPF) is called.
Constraint: $LDA \geq \max(1, M)$.
- 5: JPVT(*) – INTEGER array *Input/Output*
Note: the dimension of the array JPVT must be at least $\max(1, N)$.
On entry: if $JPVT(i) \neq 0$, then the i th 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 i th 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 $JPVT(i) = k$, then the k th column of A is moved to become the i th column of AP ; in other words, the columns of AP are the columns of A in the order $JPVT(1), JPVT(2), \dots, JPVT(n)$.
- 6: TAU(min(M,N)) – REAL (KIND=nag_wp) array *Output*
On exit: further details of the orthogonal matrix Q .
- 7: WORK(3 × N) – REAL (KIND=nag_wp) array *Workspace*
- 8: INFO – INTEGER *Output*
On exit: $INFO = 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

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 Parallelism and Performance

F08BEF (DGEQPF) 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 $\frac{2}{3}n^2(3m - n)$ if $m \geq n$ or $\frac{2}{3}m^2(3n - m)$ if $m < n$.

To form the orthogonal matrix Q F08BEF (DGEQPF) may be followed by a call to F08AFF (DORGQR):

```
CALL DORGQR(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 F08BEF (DGEQPF).

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

To apply Q to an arbitrary real rectangular matrix C , F08BEF (DGEQPF) may be followed by a call to F08AGF (DORMQR). For example,

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

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

To compute a QR factorization without column pivoting, use F08AEF (DGEQRF).

The complex analogue of this routine is F08BSF (ZGEQPF).

10 Example

This example finds the basic solutions for the linear least squares problems

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

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

$$A = \begin{pmatrix} -0.09 & 0.14 & -0.46 & 0.68 & 1.29 \\ -1.56 & 0.20 & 0.29 & 1.09 & 0.51 \\ -1.48 & -0.43 & 0.89 & -0.71 & -0.96 \\ -1.09 & 0.84 & 0.77 & 2.11 & -1.27 \\ 0.08 & 0.55 & -1.13 & 0.14 & 1.74 \\ -1.59 & -0.72 & 1.06 & 1.24 & 0.34 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} -0.01 & -0.04 \\ 0.04 & -0.03 \\ 0.05 & 0.01 \\ -0.03 & -0.02 \\ 0.02 & 0.05 \\ -0.06 & 0.07 \end{pmatrix}.$$

Here A is approximately rank-deficient, and hence it is preferable to use F08BEF (DGEQPF) rather than F08AEF (DGEQRF).

10.1 Program Text

```

Program f08befe

!      F08BEF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
Use nag_library, Only: dgeqpf, dormqr, dtrsv, nag_wp, x04caf
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Real (Kind=nag_wp), Parameter      :: zero = 0.0E0_nag_wp
Integer, Parameter                 :: nin = 5, nout = 6
!      .. Local Scalars ..
Real (Kind=nag_wp)                 :: tol
Integer                             :: i, ifail, info, k, lda, ldb, ldx,      &
                                     lwork, m, n, nrhs
!      .. Local Arrays ..
Real (Kind=nag_wp), Allocatable    :: a(:,,:), b(:,,:), tau(:), work(:),    &
                                     x(:,,:)
Integer, Allocatable                :: jpvt(:)
!      .. Intrinsic Procedures ..
Intrinsic                           :: abs
!      .. Executable Statements ..
Write (nout,*) 'F08BEF Example Program Results'
!      Skip heading in data file
Read (nin,*)
Read (nin,*) m, n, nrhs
lda = m
ldb = m
ldx = m
lwork = 64*n
Allocate (a(lda,n),b(ldb,nrhs),tau(n),work(lwork),x(ldx,nrhs),jpvt(n))

!      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)

!      Initialize JPVT to be zero so that all columns are free

jpvt(1:n) = 0

!      Compute the QR factorization of A

!      The NAG name equivalent of dgeqpf is f08bef
Call dgeqpf(m,n,a,lda,jpvt,tau,work,info)

!      Choose TOL to reflect the relative accuracy of the input data

tol = 0.01E0_nag_wp

!      Determine which columns of R to use

loop: Do k = 1, n

```

```

        If (abs(a(k,k))<=tol*abs(a(1,1))) Then
            Exit loop
        End If
    End Do loop

!   Compute C = (Q**T)*B, storing the result in B
    k = k - 1

!   The NAG name equivalent of dormqr is f08agf
    Call dormqr('Left','Transpose',m,nrhs,n,a,lda,tau,b,ldb,work,lwork,info)

!   Compute least squares solution by back-substitution in R*B = C
    Do i = 1, nrhs

!       The NAG name equivalent of dtrsv is f06pjf
        Call dtrsv('Upper','No transpose','Non-Unit',k,a,lda,b(1,i),1)

!       Set the unused elements of the I-th solution vector to zero
        b(k+1:n,i) = zero

    End Do

!   Unscramble the least squares solution stored in B
    Do i = 1, n
        x(jpvt(i),1:nrhs) = b(i,1:nrhs)
    End Do

!   Print least squares solution
    Write (nout,*)
    Flush (nout)

!   ifail: behaviour on error exit
!           =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
    ifail = 0
    Call x04caf('General',' ',n,nrhs,x,ldx,'Least squares solution',ifail)

End Program f08befe

```

10.2 Program Data

```

F08BEF Example Program Data
  6  5  2                               :Values of M, N and NRHS
-0.09  0.14 -0.46  0.68  1.29
-1.56  0.20  0.29  1.09  0.51
-1.48 -0.43  0.89 -0.71 -0.96
-1.09  0.84  0.77  2.11 -1.27
  0.08  0.55 -1.13  0.14  1.74
-1.59 -0.72  1.06  1.24  0.34 :End of matrix A
-0.01 -0.04
  0.04 -0.03
  0.05  0.01
-0.03 -0.02
  0.02  0.05
-0.06  0.07                               :End of matrix B

```

10.3 Program Results

F08BEF Example Program Results

Least squares solution

	1	2
1	-0.0370	-0.0044
2	0.0647	-0.0335
3	0.0000	0.0000
4	-0.0515	0.0018
5	0.0066	0.0102
