

NAG Toolbox for MATLAB

f08be

1 Purpose

f08be computes the QR factorization, with column pivoting, of a real m by n matrix.

2 Syntax

```
[a, jpvvt, tau, info] = f08be(a, jpvvt, 'm', m, 'n', n)
```

3 Description

f08be 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). Functions 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 function 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 Parameters

5.1 Compulsory Input Parameters

1: **a(lda,*)** – double array

The first dimension of the array **a** must be at least $\max(1, m)$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

The m by n matrix A .

2: **jpvt(*) – int32 array**

Note: the dimension of the array **jpvt** must be at least $\max(1, \mathbf{n})$.

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).

5.2 Optional Input Parameters

1: **m – int32 scalar**

Default: The first dimension of the array **a**.

m , the number of rows of the matrix A .

Constraint: $\mathbf{m} \geq 0$.

2: **n – int32 scalar**

Default: The second dimension of the array **a**.

n , the number of columns of the matrix A .

Constraint: $\mathbf{n} \geq 0$.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, work

5.4 Output Parameters

1: **a(lda,*) – double array**

The first dimension of the array **a** must be at least $\max(1, \mathbf{m})$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

If $m \geq n$, the elements below the diagonal are overwritten by details of the orthogonal matrix Q and the upper triangle contains the corresponding elements of the n by n upper triangular matrix R .

If $m < n$, the strictly lower triangular part contains 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 .

2: **jpvt(*) – int32 array**

Note: the dimension of the array **jpvt** must be at least $\max(1, \mathbf{n})$.

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), ..., **jpvt**(n).

3: **tau(*) – double array**

Note: the dimension of the array **tau** must be at least $\max(1, \min(\mathbf{m}, \mathbf{n}))$.

further details of the orthogonal matrix Q .

4: **info – int32 scalar**

info = 0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

info = $-i$

If **info** = $-i$, parameter i had an illegal value on entry. The parameters are numbered as follows:

1: **m**, 2: **n**, 3: **a**, 4: **lda**, 5: **jpvt**, 6: **tau**, 7: **work**, 8: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

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 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 f08be may be followed by a call to f08af:

```
[a, info] = f08af(a(:,1:m), tau);
```

but note that the second dimension of the array **a** must be at least **m**, which may be larger than was required by f08be.

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:

```
[a, info] = f08af(a, tau);
```

To apply Q to an arbitrary real rectangular matrix C , f08be may be followed by a call to f08ag. For example,

```
[c, info] = f08ag('Left','Transpose', a(:,1:min(m,n)), tau, c);
```

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

To compute a QR factorization without column pivoting, use f08ae.

The complex analogue of this function is f08bs.

9 Example

```
a = [-0.09, 0.14, -0.46, 0.68, 1.29;
      -1.56, 0.2, 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];
jpvt = [int32(0);
         int32(0);
         int32(0);
         int32(0);
         int32(0)];
[aOut, jpvtOut, tau, info] = f08be(a, jpvt)

aOut =
```

```
      2.8904      0.5162     -1.7198      0.2024     -1.5026
      0.5234     -2.7084     -0.3648     -0.0873      1.1475
      0.4966     -0.4778      2.2523      0.8397     -0.0060
      0.3657     -0.5520     -0.3306     -1.0086      0.7116
     -0.0268      0.6259     -0.0197      0.1934     -0.0034
      0.5335     -0.0259      0.0087     -0.3961     -0.6352
jpvtOut =
          1
          5
          4
          2
          3
tau =
      1.0311
      1.0388
      1.8022
      1.6746
      1.4250
info =
          0
```
