NAG Toolbox for MATLAB

f08cu

1 Purpose

f08cu multiplies a general complex m by n matrix C by the complex unitary matrix Q from a QL factorization computed by f08cs.

2 Syntax

```
[c, info] = f08cu(side, trans, a, tau, c, 'm', m, 'n', n, 'k', k)
```

3 Description

f08cu is intended to be used following a call to f08cs, which performs a QL factorization of a complex matrix A and represents the unitary matrix Q as a product of elementary reflectors.

This function may be used to form one of the matrix products

$$QC, \quad Q^{\mathrm{H}}C, \quad CQ, \quad CQ^{\mathrm{H}},$$

overwriting the result on C, which may be any complex rectangular m by n matrix.

A common application of this function is in solving linear least squares problems, as described in the F08 Chapter Introduction, and illustrated in Section 9 of the document for f08cs.

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 URL: http://www.netlib.org/lapack/lug

5 Parameters

5.1 Compulsory Input Parameters

1: **side – string**

Indicates how Q or Q^{H} is to be applied to C.

side = 'L'

Q or Q^{H} is applied to C from the left.

side = 'R'

Q or Q^{H} is applied to C from the right.

Constraint: **side** = 'L' or 'R'.

2: **trans – string**

Indicates whether Q or Q^{H} is to be applied to C.

trans = 'N'

Q is applied to C.

trans = 'C'

Q^{H} is applied to C.

Constraint: **trans** = 'N' or 'C'.

3: **a**(**lda**,∗) – **complex array**

The first dimension, **lda**, of the array **a** must satisfy

if **side** = 'L', **lda** $\geq \max(1, \mathbf{m})$;
if **side** = 'R', **lda** $\geq \max(1, \mathbf{n})$.

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

Details of the vectors which define the elementary reflectors, as returned by f08cs.

4: **tau**(∗) – **complex array**

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

Further details of the elementary reflectors as returned by f08cs.

5: **c**(**ldc**,∗) – **complex array**

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

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

The m by n matrix C.

5.2 Optional Input Parameters

1: **m** – **int32 scalar**

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

m, the number of rows of the matrix C.

Constraint: **m** ≥ 0.

2: **n** – **int32 scalar**

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

n, the number of columns of the matrix C.

Constraint: **n** ≥ 0.

3: **k** – **int32 scalar**

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

k, the number of elementary reflectors whose product defines the matrix Q.

Constraints:

if **side** = 'L', **m** $\geq$ **k** ≥ 0;
if **side** = 'R', **n** $\geq$ **k** ≥ 0.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldc, work, lwork

5.4 Output Parameters

1: **c**(**ldc**,∗) – **complex array**

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

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

c contains QC or $Q^{\mathrm{H}}C$ or CQ or CQ^{H} as specified by **side** and **trans**.

2: **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: **side**, 2: **trans**, 3: **m**, 4: **n**, 5: **k**, 6: **a**, 7: **lda**, 8: **tau**, 9: **c**, 10: **ldc**, 11: **work**, 12: **lwork**, 13: **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 result differs from the exact result by a matrix E such that

$$\|E\|_2 = O\epsilon\|C\|_2$$

where ϵ is the ***machine precision***.

8 Further Comments

The total number of floating point operations is approximately $8nk(2m-k)$ if **side** = 'L' and $8mk(2n-k)$ if **side** = 'R'.

The real analogue of this function is f08cg.

9 Example

```
side = 'Left';
trans = 'Conjugate Transpose';
a       =       [complex(-0.1732849729587456,       -0.353512300439208),
complex(0.09114631819652766, ...
                  +0.2475043484243412),     complex(-0.3728300824744412,
+0.4958059927232391), ...
     complex(-0.004843574322500398, +0.1359553259506703);
           complex(-0.2462313258920653,  +0.2360567043400598),  complex(-
0.4802181323902914, ...
                  -0.1420168005324323),     complex(-0.1765491709076989,
+0.1573286421732319), ...
     complex(-0.2500132710791688, +0.2066154899636792);
         complex(-1.835343413987657,  +0),  complex(0.4821546631632405,  -
0.09601786537269767), ...
           complex(0.21466822640392,   +0.1362481009678133),   complex(-
0.3473326302453916, +0.2282035599626739);
            complex(-1.000908232487384,   +1.32039822375845),   complex(-
1.575493318887248, ...
              +0),    complex(-0.2486033881873816,    -0.20517929204053),
complex(0.06655374884663712, -0.0718369909305264);
           complex(0.4156798554945925,  +1.306842647249796),  complex(-
1.101370732269467, ...
            -0.05966135311459431),   complex(-2.991745620440206,   +0),
complex(0.5272403281091034, +0.480309771000454);
          complex(-0.9547699266414058,  +0.7897042065311224),  complex(-
0.3055741168299267, ...
                 +0.2217392484654138),     complex(-0.01276691481133434,
+0.08423270423682098), complex(-2.764959312539698, +0)];
tau = [complex(1.56428134034193, -0.1176469112365368);
```

```
     complex(1.251296240416963, -0.1907664156695332);
     complex(1.137210468449182, -0.3464883871917274);
     complex(1.094033933454587, +0.09403393345458752)];
c = [complex(-2.09, +1.93), complex(3.26, -2.7);
     complex(3.34, -3.53), complex(-6.22, +1.16);
     complex(-4.94, -2.04), complex(7.94, -3.13);
     complex(0.17, +4.23), complex(1.04, -4.26);
     complex(-5.19, +3.63), complex(-2.31, -2.12);
     complex(0.98, +2.53), complex(-1.39, -4.05)];
[cOut, info] = f08cu(side, trans, a, tau, c)
```

```
cOut =
  -0.0185 - 0.0462i  -0.1172 + 0.0995i
   0.0289 + 0.0378i   0.0100 + 0.1053i
   0.9257 + 2.2352i  -1.4002 - 2.6665i
   5.9384 - 3.9488i -10.8068 + 5.2389i
  -0.2225 + 2.4007i   0.4569 - 1.0754i
   0.4632 - 7.9327i   4.8369 + 0.2937i
info =
           0
```