NAG Toolbox for MATLAB

f08ku

1 Purpose

f08ku multiplies an arbitrary complex m by n matrix C by one of the complex unitary matrices Q or P which were determined by f08ks when reducing a complex matrix to bidiagonal form.

2 Syntax

```
[c, info] = f08ku(vect, side, trans, k, a, tau, c, 'm', m, 'n', n)
```

3 Description

f08ku is intended to be used after a call to f08ks, which reduces a complex rectangular matrix A to real bidiagonal form B by a unitary transformation: $A = QBP^{\mathrm{H}}$. f08ks represents the matrices Q and P^{H} as products of elementary reflectors.

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

$$QC, Q^{\mathrm{H}}C, CQ, CQ^{\mathrm{H}}, PC, P^{\mathrm{H}}C, CP \text{ or } CP^{\mathrm{H}},$$

overwriting the result on C (which may be any complex rectangular matrix).

4 References

Golub G H and Van Loan C F 1996 *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

Note: in the descriptions below, r denotes the order of Q or P^{H}: if **side** = 'L', r = **m** and if **side** = 'R', r = **n**.

5.1 Compulsory Input Parameters

1: **vect – string**

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

vect = 'Q'

Q or Q^{H} is applied to C.

vect = 'P'

P or P^{H} is applied to C.

Constraint: **vect** = 'Q' or 'P'.

2: **side – string**

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

side = 'L'

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

side = 'R'

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

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

3: **trans – string**

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

trans = 'N'

Q or P is applied to C.

trans = 'C'

Q^{H} or P^{H} is applied to C.

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

4: **k – int32 scalar**

If **vect** = 'Q', the number of columns in the original matrix A.

If **vect** = 'P', the number of rows in the original matrix A.

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

5: **a(lda,∗) – complex array**

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

if **vect** = 'Q', $\mathbf{lda} \geq \max(1, r)$;
if **vect** = 'P', $\mathbf{lda} \geq \max(1, \min(r, \mathbf{k}))$.

The second dimension of the array must be at least $\max(1, \min(r, \mathbf{k}))$ if **vect** = 'Q' and at least $\max(1, r)$ if **vect** = 'P'

Details of the vectors which define the elementary reflectors, as returned by f08ks.

6: **tau(∗) – complex array**

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

Further details of the elementary reflectors, as returned by f08ks in its parameter **tauq** if **vect** = 'Q', or in its parameter **taup** if **vect** = 'P'.

7: **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 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: $\mathbf{m} \geq 0$.

2: **n – int32 scalar**

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

n, the number of columns of the matrix C.

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

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 $C^{\mathrm{H}}Q$ or PC or $P^{\mathrm{H}}C$ or CP or $C^{\mathrm{H}}P$ as specified by **vect**, **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: **vect**, 2: **side**, 3: **trans**, 4: **m**, 5: **n**, 6: **k**, 7: **a**, 8: **lda**, 9: **tau**, 10: **c**, 11: **ldc**, 12: **work**, 13: **lwork**, 14: **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 real floating-point operations is approximately

if **side** = 'L' and $m \geq k$, $8nk(2m - k)$;
if **side** = 'R' and $n \geq k$, $8mk(2n - k)$;
if **side** = 'L' and $m < k$, $8m^2n$;
if **side** = 'R' and $n < k$, $8mn^2$,

where k is the value of the parameter **k**.

The real analogue of this function is f08kg.

9 Example

```
vect = 'Q';
side = 'Right';
trans = 'No transpose';
k = int32(4);
a  =  [complex(-3.087005021051958,  +0),  complex(2.112571007455839,  +0),
complex(0.05433411079440312, ...
                    +0.4543118496773522),     complex(0.375743827925403,
+0.1070087304094524);
           complex(0,  +0),  complex(-2.066039276679068,  +0),  complex(-
1.262810106655224, ...
     +0), complex(0.02827717828732752, +0.1650056103049374);
       complex(0, +0), complex(-0.2804787991136917, -0.4124461074713915),
...
     complex(-1.873128891125712, +0), complex(1.612633872800391, +0);
       complex(0, +0), complex(0.2103472372638732, -0.4460760994276616),
complex(-0.5708419424841372, ...
     +0.06437446295221612), complex(-2.002182866206992, +0)];
tau = [complex(0, +0);
     complex(1.098198238126112, +0.5158162160396563);
     complex(1.455158088337053, -0.2659229774958434);
     complex(1.989879752802885, -0.1419086853962756)];
c   =   [complex(-0.3109810296560065,   +0.2623902437722555),   complex(-
0.3175341445023601, ...
                  +0.4834967063433271),    complex(0.4966143187964562,    -
0.2996834399081108), ...
     complex(-0.007195817944538043, -0.3717893210480535);
            complex(0.3174598011071733,  -0.6413983736655133),  complex(-
0.2061862718385913, ...
                +0.1576964755255177),    complex(-0.07925902434510126,    -
0.3093749483233558), ...
     complex(-0.02816166060051156, -0.1491467515264786);
                     complex(-0.2008419149861708,    +0.1490117433768365),
complex(0.4891881009599058, ...
                  -0.09002535062431442),    complex(0.035745707605966,    -
0.02190382125352534), ...
     complex(0.5624615849142621, -0.07099355406423355);
                      complex(0.1198572718465858,    -0.1230966575721692),
complex(0.2566010661168247, ...
                  -0.3055384784364648),    complex(0.4488646004434153,    -
0.2140825016792581), ...
     complex(-0.1651301691537539, +0.1799762380267428);
                     complex(-0.2688690152234222,    -0.1652086720047534),
complex(0.1696708265434812, ...
                    -0.2490702105456178),    complex(-0.04956098112958145,
+0.1157544723073297), ...
     complex(-0.48852018643744, -0.4540377976759434);
           complex(-0.3498536583630073,  +0.09070280031633525),  complex(-
0.049104334058638, ...
                 -0.3133356336974162),    complex(-0.1256478989223875,    -
0.5299605073526112), ...
     complex(0.1039065872268565, +0.04496306210314513)];
[cOut, info] = f08ku(vect, side, trans, k, a, tau, c)
```

```
cOut =
   -0.3110 + 0.2624i    0.6521 + 0.5532i    0.0427 + 0.0361i   -0.2634 -
0.0741i
    0.3175 - 0.6414i    0.3488 + 0.0721i    0.2287 + 0.0069i    0.1101 -
0.0326i
   -0.2008 + 0.1490i   -0.3103 + 0.0230i    0.1855 - 0.1817i   -0.2956 +
0.5648i
    0.1199 - 0.1231i   -0.0046 - 0.0005i   -0.3305 + 0.4821i   -0.0675 +
0.3464i
   -0.2689 - 0.1652i    0.1794 - 0.0586i   -0.5235 - 0.2580i    0.3927 +
0.1450i
   -0.3499 + 0.0907i    0.0829 - 0.0506i    0.3202 + 0.3038i    0.3174 +
0.3241i
```

```
info =
           0
```