NAG Toolbox

nag_lapack_zunmql (f08cu)

1 Purpose

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

2 Syntax

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

3 Description

nag_lapack_zunmql (f08cu) is intended to be used following a call to nag_lapack_zgeqlf (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 10 in nag_lapack_zgeqlf (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 http://www.netlib.org/lapack/lug

5 Parameters

5.1 Compulsory Input Parameters

1: **side** – CHARACTER(1)

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** – CHARACTER(1)

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 (KIND=nag_wp) 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 **a** must be at least $\max(1, \mathbf{k})$.

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

4: **tau**(:) – COMPLEX (KIND=nag_wp) array

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

Further details of the elementary reflectors, as returned by nag_lapack_zgeqlf (f08cs).

5: **c**(ldc, :) – COMPLEX (KIND=nag_wp) array

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

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

The m by n matrix C.

5.2 Optional Input Parameters

1: **m** – INTEGER

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

m, the number of rows of the matrix C.

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

2: **n** – INTEGER

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

n, the number of columns of the matrix C.

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

3: **k** – INTEGER

Default: the second dimension of the arrays **a**, **tau**.

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

Constraints:

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

5.3 Output Parameters

1: **c**(ldc, :) – COMPLEX (KIND=nag_wp) array

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

The second dimension of the array **c** will be $\max(1, \mathbf{n})$.

c stores QC or Q^HC or CQ or CQ^H as specified by **side** and **trans**.

2: **info** – INTEGER

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

6 Error Indicators and Warnings

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 nag_lapack_dormql (f08cg).

9 Example

See Section 10 in nag_lapack_zgeqlf (f08cs).

9.1 Program Text

```
function f08cu_example

fprintf('f08cu example results\n\n');

m = 6;
n = 4;
a = [ 0.96 - 0.81i, -0.03 + 0.96i, -0.91 + 2.06i, -0.05 + 0.41i;
     -0.98 + 1.98i, -1.20 + 0.19i, -0.66 + 0.42i, -0.81 + 0.56i;
      0.62 - 0.46i,  1.01 + 0.02i,  0.63 - 0.17i, -1.11 + 0.60i;
     -0.37 + 0.38i,  0.19 - 0.54i, -0.98 - 0.36i,  0.22 - 0.20i;
      0.83 + 0.51i,  0.20 + 0.01i, -0.17 - 0.46i,  1.47 + 1.59i;
      1.08 - 0.28i,  0.20 - 0.12i, -0.07 + 1.23i,  0.26 + 0.26i];

b = [-2.09 + 1.93i,  3.26 - 2.70i;
      3.34 - 3.53i, -6.22 + 1.16i;
     -4.94 - 2.04i,  7.94 - 3.13i;
      0.17 + 4.23i,  1.04 - 4.26i;
     -5.19 + 3.63i, -2.31 - 2.12i;
      0.98 + 2.53i, -1.39 - 4.05i];

% Compute the QL factorization of A
[ql, tau, info] = f08cs(a);

% LX = Q^H B = C; compute C = (Q^H)*B
[c, info] = f08cu( ...
    'Left', 'ConjTrans', ql, tau, b);
```

```
% Least-squares solution X = L^-1 C (lower n part)
il = m-n+1;
[x, info] = f07ts( ...
    'Lower', 'Notrans','Non-Unit', ql(il:m,:), c(il:m,:));

% Print least-squares solutions
ncols  = nag_int(80);
indent = nag_int(0);
[ifail] = x04db( ...
  'General', ' ', x, 'Bracketed', 'F7.4', ...
  'Least-squares solution(s)', 'Integer', 'Integer', ...
  ncols, indent);

% Compute estimates of the square roots of the residual sums of squares.
rnorm = zeros(2,1);
for j=1:2
  rnorm(j) = norm(c(1:m-n,j));
end
fprintf('\nSquare root(s) of the residual sum(s) of squares\n');
fprintf('\t%11.2e    %11.2e\n', rnorm(1), rnorm(2));
```

9.2 Program Results

```
     f08cu example results

 Least-squares solution(s)
                    1                 2
 1  (-0.5044,-1.2179) ( 0.7629, 1.4529)
 2  (-2.4281, 2.8574) ( 5.1570,-3.6089)
 3  ( 1.4872,-2.1955) (-2.6518, 2.1203)
 4  ( 0.4537, 2.6904) (-2.7606, 0.3318)

Square root(s) of the residual sum(s) of squares
    6.88e-02       1.87e-01
```