

# NAG C Library Function Document

## nag\_zunmlq (f08axc)

### 1 Purpose

nag\_zunmlq (f08axc) multiplies an arbitrary complex matrix  $C$  by the complex unitary matrix  $Q$  from an  $LQ$  factorization computed by nag\_zgelqf (f08avc).

### 2 Specification

```
void nag_zunmlq (Nag_OrderType order, Nag_SideType side, Nag_TransType trans,
                Integer m, Integer n, Integer k, const Complex a[], Integer pda,
                const Complex tau[], Complex c[], Integer pdc, NagError *fail)
```

### 3 Description

nag\_zunmlq (f08axc) is intended to be used after a call to nag\_zgelqf (f08avc), which performs an  $LQ$  factorization of a complex matrix  $A$ . The unitary matrix  $Q$  is represented as a product of elementary reflectors.

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

$$QC, Q^H C, CQ \text{ or } CQ^H,$$

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

- 1: **order** – Nag\_OrderType *Input*  
*On entry:* the **order** parameter specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by **order** = **Nag\_RowMajor**. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.  
*Constraint:* **order** = **Nag\_RowMajor** or **Nag\_ColMajor**.
- 2: **side** – Nag\_SideType *Input*  
*On entry:* indicates how  $Q$  or  $Q^H$  is to be applied to  $C$  as follows:  
     if **side** = **Nag\_LeftSide**,  $Q$  or  $Q^H$  is applied to  $C$  from the left;  
     if **side** = **Nag\_RightSide**,  $Q$  or  $Q^H$  is applied to  $C$  from the right.  
*Constraint:* **side** = **Nag\_LeftSide** or **Nag\_RightSide**.
- 3: **trans** – Nag\_TransType *Input*  
*On entry:* indicates whether  $Q$  or  $Q^H$  is to be applied to  $C$  as follows:  
     if **trans** = **Nag\_NoTrans**,  $Q$  is applied to  $C$ ;  
     if **trans** = **Nag\_ConjTrans**,  $Q^H$  is applied to  $C$ .  
*Constraint:* **trans** = **Nag\_NoTrans** or **Nag\_ConjTrans**.

- 4: **m** – Integer *Input*  
*On entry:*  $m$ , the number of rows of the matrix  $C$ .  
*Constraint:*  $m \geq 0$ .
- 5: **n** – Integer *Input*  
*On entry:*  $n$ , the number of columns of the matrix  $C$ .  
*Constraint:*  $n \geq 0$ .
- 6: **k** – Integer *Input*  
*On entry:*  $k$ , the number of elementary reflectors whose product defines the matrix  $Q$ .  
*Constraints:*  
     if **side** = **Nag\_LeftSide**,  $m \geq k \geq 0$ ;  
     if **side** = **Nag\_RightSide**,  $n \geq k \geq 0$ .
- 7: **a**[*dim*] – Complex *Input/Output*  
**Note:** the dimension, *dim*, of the array **a** must be at least  
      $\max(1, \mathbf{pda} \times \mathbf{m})$  when **side** = **Nag\_LeftSide** and **order** = **Nag\_ColMajor**;  
      $\max(1, \mathbf{pda} \times \mathbf{k})$  when **side** = **Nag\_LeftSide** and **order** = **Nag\_RowMajor**;  
      $\max(1, \mathbf{pda} \times \mathbf{n})$  when **side** = **Nag\_RightSide** and **order** = **Nag\_ColMajor**;  
      $\max(1, \mathbf{pda} \times \mathbf{k})$  when **side** = **Nag\_RightSide** and **order** = **Nag\_RowMajor**.  
 If **order** = **Nag\_ColMajor**, the  $(i, j)$ th element of the matrix  $A$  is stored in **a**[( $j - 1$ )  $\times$  **pda** +  $i - 1$ ] and  
 if **order** = **Nag\_RowMajor**, the  $(i, j)$ th element of the matrix  $A$  is stored in **a**[( $i - 1$ )  $\times$  **pda** +  $j - 1$ ].  
*On entry:* details of the vectors which define the elementary reflectors, as returned by nag\_zgelqf (f08avc).  
*On exit:* used as internal workspace prior to being restored and hence is unchanged.
- 8: **pda** – Integer *Input*  
*On entry:* the stride separating matrix row or column elements (depending on the value of **order**) in the array **a**.  
*Constraints:*  
     if **order** = **Nag\_ColMajor**,  $\mathbf{pda} \geq \max(1, \mathbf{k})$ ;  
     if **order** = **Nag\_RowMajor**,  
         if **side** = **Nag\_LeftSide**,  $\mathbf{pda} \geq \max(1, \mathbf{m})$ ;  
         if **side** = **Nag\_RightSide**,  $\mathbf{pda} \geq \max(1, \mathbf{n})$ .
- 9: **tau**[*dim*] – const Complex *Input*  
**Note:** the dimension, *dim*, of the array **tau** must be at least  $\max(1, \mathbf{k})$ .  
*On entry:* further details of the elementary reflectors, as returned by nag\_zgelqf (f08avc).
- 10: **c**[*dim*] – Complex *Input/Output*  
**Note:** the dimension, *dim*, of the array **c** must be at least  $\max(1, \mathbf{pdc} \times \mathbf{n})$  when **order** = **Nag\_ColMajor** and at least  $\max(1, \mathbf{pdc} \times \mathbf{m})$  when **order** = **Nag\_RowMajor**.  
 If **order** = **Nag\_ColMajor**, the  $(i, j)$ th element of the matrix  $C$  is stored in **c**[( $j - 1$ )  $\times$  **pdc** +  $i - 1$ ] and  
 if **order** = **Nag\_RowMajor**, the  $(i, j)$ th element of the matrix  $C$  is stored in **c**[( $i - 1$ )  $\times$  **pdc** +  $j - 1$ ].  
*On entry:* the  $m$  by  $n$  matrix  $C$ .  
*On exit:* **c** is overwritten by  $QC$  or  $Q^H C$  or  $CQ$  or  $CQ^H$  as specified by **side** and **trans**.

- 11: **pdc** – Integer *Input*  
*On entry:* the stride separating matrix row or column elements (depending on the value of **order**) in the array **c**.  
*Constraints:*  
 if **order** = **Nag\_ColMajor**, **pdc**  $\geq \max(1, \mathbf{m})$ ;  
 if **order** = **Nag\_RowMajor**, **pdc**  $\geq \max(1, \mathbf{n})$ .
- 12: **fail** – NagError \* *Output*  
 The NAG error parameter (see the Essential Introduction).

## 6 Error Indicators and Warnings

### NE\_INT

On entry, **m** =  $\langle value \rangle$ .  
 Constraint: **m**  $\geq 0$ .

On entry, **n** =  $\langle value \rangle$ .  
 Constraint: **n**  $\geq 0$ .

On entry, **pda** =  $\langle value \rangle$ .  
 Constraint: **pda**  $> 0$ .

On entry, **pdc** =  $\langle value \rangle$ .  
 Constraint: **pdc**  $> 0$ .

### NE\_INT\_2

On entry, **pda** =  $\langle value \rangle$ , **k** =  $\langle value \rangle$ .  
 Constraint: **pda**  $\geq \max(1, \mathbf{k})$ .

On entry, **pdc** =  $\langle value \rangle$ , **m** =  $\langle value \rangle$ .  
 Constraint: **pdc**  $\geq \max(1, \mathbf{m})$ .

On entry, **pdc** =  $\langle value \rangle$ , **n** =  $\langle value \rangle$ .  
 Constraint: **pdc**  $\geq \max(1, \mathbf{n})$ .

### NE\_ENUM\_INT\_3

On entry, **side** =  $\langle value \rangle$ , **m** =  $\langle value \rangle$ , **n** =  $\langle value \rangle$ , **k** =  $\langle value \rangle$ .  
 Constraint: if **side** = **Nag\_LeftSide**, **m**  $\geq \mathbf{k} \geq 0$ ;  
 if **side** = **Nag\_RightSide**, **n**  $\geq \mathbf{k} \geq 0$ .

On entry, **side** =  $\langle value \rangle$ , **m** =  $\langle value \rangle$ , **n** =  $\langle value \rangle$ , **pda** =  $\langle value \rangle$ .  
 Constraint: if **side** = **Nag\_LeftSide**, **pda**  $\geq \max(1, \mathbf{m})$ ;  
 if **side** = **Nag\_RightSide**, **pda**  $\geq \max(1, \mathbf{n})$ .

### NE\_ALLOC\_FAIL

Memory allocation failed.

### NE\_BAD\_PARAM

On entry, parameter  $\langle value \rangle$  had an illegal value.

### NE\_INTERNAL\_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

## 7 Accuracy

The computed result differs from the exact result by a matrix  $E$  such that

$$\|E\|_2 = O(\epsilon)\|C\|_2,$$

where  $\epsilon$  is the *machine precision*.

## 8 Further Comments

The total number of real floating-point operations is approximately  $8nk(2m - k)$  if **side** = **Nag\_LeftSide** and  $8mk(2n - k)$  if **side** = **Nag\_RightSide**.

The real analogue of this function is nag\_dormlq (f08akc).

## 9 Example

See Section 9 of the document for nag\_zgelqf (f08avc).

---