NAG Library Routine Document

F08KUF (ZUNMBR)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F08KUF (ZUNMBR) multiplies an arbitrary complex m by n matrix C by one of the complex unitary matrices Q or P which were determined by F08KSF (ZGEBRD) when reducing a complex matrix to bidiagonal form.

2 Specification

```
SUBROUTINE F08KUF (VECT, SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC, WORK,       &
                   LWORK, INFO)

INTEGER                 M, N, K, LDA, LDC, LWORK, INFO
COMPLEX (KIND=nag_wp) A(LDA,*), TAU(*), C(LDC,*), WORK(max(1,LWORK))
CHARACTER(1)            VECT, SIDE, TRANS
```

The routine may be called by its LAPACK name ***zunmbr***.

3 Description

F08KUF (ZUNMBR) is intended to be used after a call to F08KSF (ZGEBRD), which reduces a complex rectangular matrix A to real bidiagonal form B by a unitary transformation: $A = QBP^{\mathrm{H}}$. F08KSF (ZGEBRD) represents the matrices Q and P^{H} as products of elementary reflectors.

This routine 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.

1: VECT – CHARACTER(1) *Input*

On entry: 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 – CHARACTER(1) *Input*

On entry: 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 – CHARACTER(1) *Input*

On entry: 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: M – INTEGER *Input*

On entry: m, the number of rows of the matrix C.

Constraint: M ≥ 0.

5: N – INTEGER *Input*

On entry: n, the number of columns of the matrix C.

Constraint: N ≥ 0.

6: K – INTEGER *Input*

On entry: 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: K ≥ 0.

7: A(LDA,$*$) – COMPLEX (KIND=nag_wp) array *Input*

Note: the second dimension of the array A must be at least $\max(1, \min(r, \mathrm{K}))$ if VECT = 'Q' and at least $\max(1, r)$ if VECT = 'P'.

On entry: details of the vectors which define the elementary reflectors, as returned by F08KSF (ZGEBRD).

8: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08KUF (ZUNMBR) is called.

Constraints:

if VECT = 'Q', LDA $\geq \max(1, r)$;
if VECT = 'P', LDA $\geq \max(1, \min(r, \mathrm{K}))$.

9: TAU($*$) – COMPLEX (KIND=nag_wp) array *Input*

Note: the dimension of the array TAU must be at least $\max(1, \min(r, \mathrm{K}))$.

On entry: further details of the elementary reflectors, as returned by F08KSF (ZGEBRD) in its parameter TAUQ if VECT = 'Q', or in its parameter TAUP if VECT = 'P'.

10: C(LDC,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array C must be at least $\max(1, \mathrm{N})$.

On entry: the matrix C.

On exit: C is overwritten by 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.

11: LDC – INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which F08KUF (ZUNMBR) is called.

Constraint: LDC $\geq \max(1, \mathrm{M})$.

12: WORK($\max(1, \mathrm{LWORK})$) – COMPLEX (KIND=nag_wp) array *Workspace*

On exit: if INFO $= 0$, the real part of WORK(1) contains the minimum value of LWORK required for optimal performance.

13: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08KUF (ZUNMBR) is called.

If LWORK $= -1$, a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued.

Suggested value: for optimal performance, LWORK $\geq \mathrm{N} \times nb$ if SIDE $=$ 'L' and at least $\mathrm{M} \times nb$ if SIDE $=$ 'R', where nb is the optimal ***block size***.

Constraints:

if SIDE $=$ 'L', LWORK $\geq \max(1, \mathrm{N})$ or LWORK $= -1$;
if SIDE $=$ 'R', LWORK $\geq \max(1, \mathrm{M})$ or LWORK $= -1$.

14: INFO – INTEGER *Output*

On exit: INFO $= 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO $= -i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

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 routine is F08KGF (DORMBR).

9 Example

For this routine two examples are presented. Both illustrate how the reduction to bidiagonal form of a matrix A may be preceded by a QR or LQ factorization of A.

In the first example, $m > n$, and

$$A = \begin{pmatrix} 0.96 - 0.81i & -0.03 + 0.96i & -0.91 + 2.06i & -0.05 + 0.41i \\ -0.98 + 1.98i & -1.20 + 0.19i & -0.66 + 0.42i & -0.81 + 0.56i \\ 0.62 - 0.46i & 1.01 + 0.02i & 0.63 - 0.17i & -1.11 + 0.60i \\ -0.37 + 0.38i & 0.19 - 0.54i & -0.98 - 0.36i & 0.22 - 0.20i \\ 0.83 + 0.51i & 0.20 + 0.01i & -0.17 - 0.46i & 1.47 + 1.59i \\ 1.08 - 0.28i & 0.20 - 0.12i & -0.07 + 1.23i & 0.26 + 0.26i \end{pmatrix}.$$

The routine first performs a QR factorization of A as $A = Q_a R$ and then reduces the factor R to bidiagonal form B: $R = Q_b B P^{\mathrm{H}}$. Finally it forms Q_a and calls F08KUF (ZUNMBR) to form $Q = Q_a Q_b$.

In the second example, $m < n$, and

$$A = \begin{pmatrix} 0.28 - 0.36i & 0.50 - 0.86i & -0.77 - 0.48i & 1.58 + 0.66i \\ -0.50 - 1.10i & -1.21 + 0.76i & -0.32 - 0.24i & -0.27 - 1.15i \\ 0.36 - 0.51i & -0.07 + 1.33i & -0.75 + 0.47i & -0.08 + 1.01i \end{pmatrix}.$$

The routine first performs an LQ factorization of A as $A = LP_a^{\mathrm{H}}$ and then reduces the factor L to bidiagonal form B: $L = QBP_b^{\mathrm{H}}$. Finally it forms P_b^{H} and calls F08KUF (ZUNMBR) to form $P^{\mathrm{H}} = P_b^{\mathrm{H}} P_a^{\mathrm{H}}$.

9.1 Program Text

```
    Program f08kufe

!     F08KUF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f06tff, f06thf, nag_wp, x04dbf, zgebrd, zgelqf,   &
                             zgeqrf, zunglq, zungqr, zunmbr
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Complex (Kind=nag_wp), Parameter :: zero = (0.0E0_nag_wp,0.0E0_nag_wp)
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ic, ifail, info, lda, ldph, ldu,  &
                                          lwork, m, n
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), ph(:,:), tau(:), taup(:),  &
                                            tauq(:), u(:,:), work(:)
      Real (Kind=nag_wp), Allocatable  :: d(:), e(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F08KUF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Do ic = 1, 2
        Read (nin,*) m, n
        lda = m
```

```
        ldph = n
        ldu = m
        lwork = 64*(m+n)
        Allocate (a(lda,n),ph(ldph,n),tau(n),taup(n),tauq(n),u(ldu,n), &
          work(lwork),d(n),e(n-1))

!       Read A from data file

        Read (nin,*)(a(i,1:n),i=1,m)

        If (m>=n) Then

!         Compute the QR factorization of A
!         The NAG name equivalent of zgeqrf is f08asf
          Call zgeqrf(m,n,a,lda,tau,work,lwork,info)

!         Copy A to U
          Call f06tff('Lower',m,n,a,lda,u,ldu)

!         Form Q explicitly, storing the result in U
!         The NAG name equivalent of zungqr is f08atf
          Call zungqr(m,n,n,u,ldu,tau,work,lwork,info)

!         Copy R to PH (used as workspace)
          Call f06tff('Upper',n,n,a,lda,ph,ldph)

!         Set the strictly lower triangular part of R to zero
          Call f06thf('Lower',n-1,n-1,zero,zero,ph(2,1),ldph)

!         Bidiagonalize R
!         The NAG name equivalent of zgebrd is f08ksf
          Call zgebrd(n,n,ph,ldph,d,e,tauq,taup,work,lwork,info)

!         Update Q, storing the result in U
!         The NAG name equivalent of zunmbr is f08kuf
          Call zunmbr('Q','Right','No transpose',m,n,n,ph,ldph,tauq,u,ldu, &
            work,lwork,info)

!         Print bidiagonal form and matrix Q

          Write (nout,*)
          Write (nout,*) 'Example 1: bidiagonal matrix B'
          Write (nout,*) 'Diagonal'
          Write (nout,99999) d(1:n)
          Write (nout,*) 'Super-diagonal'
          Write (nout,99999) e(1:n-1)
          Write (nout,*)
          Flush (nout)

!         ifail: behaviour on error exit
!                =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
          ifail = 0
          Call x04dbf('General',' ',m,n,u,ldu,'Bracketed','F7.4', &
            'Example 1: matrix Q','Integer',rlabs,'Integer',clabs,80,0,ifail)

        Else

!         Compute the LQ factorization of A
!         The NAG name equivalent of zgelqf is f08avf
          Call zgelqf(m,n,a,lda,tau,work,lwork,info)

!         Copy A to PH
          Call f06tff('Upper',m,n,a,lda,ph,ldph)

!         Form Q explicitly, storing the result in PH
!         The NAG name equivalent of zunglq is f08awf
          Call zunglq(n,n,m,ph,ldph,tau,work,lwork,info)

!         Copy L to U (used as workspace)
          Call f06tff('Lower',m,m,a,lda,u,ldu)
```

```
!         Set the strictly upper triangular part of L to zero
          Call f06thf('Upper',m-1,m-1,zero,zero,u(1,2),ldu)

!         Bidiagonalize L
!         The NAG name equivalent of zgebrd is f08ksf
          Call zgebrd(m,m,u,ldu,d,e,tauq,taup,work,lwork,info)

!         Update P**H, storing the result in PH
!         The NAG name equivalent of zunmbr is f08kuf
          Call zunmbr('P','Left','Conjugate transpose',m,n,m,u,ldu,taup,ph, &
            ldph,work,lwork,info)

!         Print bidiagonal form and matrix P**H

          Write (nout,*)
          Write (nout,*) 'Example 2: bidiagonal matrix B'
          Write (nout,*) 'Diagonal'
          Write (nout,99999) d(1:m)
          Write (nout,*) 'Super-diagonal'
          Write (nout,99999) e(1:m-1)
          Write (nout,*)
          Flush (nout)

          ifail = 0
          Call x04dbf('General',' ',m,n,ph,ldph,'Bracketed','F7.4', &
            'Example 2: matrix P**H','Integer',rlabs,'Integer',clabs,80,0, &
            ifail)

        End If
        Deallocate (a,ph,tau,taup,tauq,u,work,d,e)
      End Do

99999 Format (3X,(8F8.4))
    End Program f08kufe
```

9.2 Program Data

```
F08KUF Example Program Data
  6  4                                          :Values of M and N, Example 1
 ( 0.96,-0.81) (-0.03, 0.96) (-0.91, 2.06) (-0.05, 0.41)
 (-0.98, 1.98) (-1.20, 0.19) (-0.66, 0.42) (-0.81, 0.56)
 ( 0.62,-0.46) ( 1.01, 0.02) ( 0.63,-0.17) (-1.11, 0.60)
 (-0.37, 0.38) ( 0.19,-0.54) (-0.98,-0.36) ( 0.22,-0.20)
 ( 0.83, 0.51) ( 0.20, 0.01) (-0.17,-0.46) ( 1.47, 1.59)
 ( 1.08,-0.28) ( 0.20,-0.12) (-0.07, 1.23) ( 0.26, 0.26)   :End of matrix A
  3  4                                          :Values of M and N, Example 2
 ( 0.28,-0.36) ( 0.50,-0.86) (-0.77,-0.48) ( 1.58, 0.66)
 (-0.50,-1.10) (-1.21, 0.76) (-0.32,-0.24) (-0.27,-1.15)
 ( 0.36,-0.51) (-0.07, 1.33) (-0.75, 0.47) (-0.08, 1.01)   :End of matrix A
```

9.3 Program Results

```
 F08KUF Example Program Results

 Example 1: bidiagonal matrix B
 Diagonal
    -3.0870 -2.0660 -1.8731 -2.0022
 Super-diagonal
     2.1126 -1.2628  1.6126

 Example 1: matrix Q
                    1                  2                  3                  4
 1  (-0.3110, 0.2624) ( 0.6521, 0.5532) ( 0.0427, 0.0361) (-0.2634,-0.0741)
 2  ( 0.3175,-0.6414) ( 0.3488, 0.0721) ( 0.2287, 0.0069) ( 0.1101,-0.0326)
 3  (-0.2008, 0.1490) (-0.3103, 0.0230) ( 0.1855,-0.1817) (-0.2956, 0.5648)
 4  ( 0.1199,-0.1231) (-0.0046,-0.0005) (-0.3305, 0.4821) (-0.0675, 0.3464)
 5  (-0.2689,-0.1652) ( 0.1794,-0.0586) (-0.5235,-0.2580) ( 0.3927, 0.1450)
 6  (-0.3499, 0.0907) ( 0.0829,-0.0506) ( 0.3202, 0.3038) ( 0.3174, 0.3241)

 Example 2: bidiagonal matrix B
```

```
 Diagonal
     2.7615  1.6298 -1.3275
 Super-diagonal
    -0.9500 -1.0183

 Example 2: matrix P**H
                    1                   2                   3                   4
 1  (-0.1258, 0.1618) (-0.2247, 0.3864) ( 0.3460, 0.2157) (-0.7099,-0.2966)
 2  ( 0.4148, 0.1795) ( 0.1368,-0.3976) ( 0.6885, 0.3386) ( 0.1667,-0.0494)
 3  ( 0.4575,-0.4807) (-0.2733, 0.4981) (-0.0230, 0.3861) ( 0.1730, 0.2395)
```