

# NAG Library Routine Document

## F08KGF (DORMBR)

**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

F08KGF (DORMBR) multiplies an arbitrary real  $m$  by  $n$  matrix  $C$  by one of the real orthogonal matrices  $Q$  or  $P$  which were determined by F08KEF (DGEBRD) when reducing a real matrix to bidiagonal form.

### 2 Specification

```
SUBROUTINE F08KGF (VECT, SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC, WORK,      &
                  LWORK, INFO)
INTEGER          M, N, K, LDA, LDC, LWORK, INFO
REAL (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 *dormbr*.

### 3 Description

F08KGF (DORMBR) is intended to be used after a call to F08KEF (DGEBRD), which reduces a real rectangular matrix  $A$  to bidiagonal form  $B$  by an orthogonal transformation:  $A = QB P^T$ . F08KEF (DGEBRD) represents the matrices  $Q$  and  $P^T$  as products of elementary reflectors.

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

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

overwriting the result on  $C$  (which may be any real 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  $P^T$ : if  $SIDE = 'L'$ ,  $r = M$  and if  $SIDE = 'R'$ ,  $r = N$ .

1: VECT – CHARACTER(1) *Input*

*On entry:* indicates whether  $Q$  or  $Q^T$  or  $P$  or  $P^T$  is to be applied to  $C$ .

VECT = 'Q'

$Q$  or  $Q^T$  is applied to  $C$ .

VECT = 'P'

$P$  or  $P^T$  is applied to  $C$ .

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

- 2: SIDE – CHARACTER(1) *Input*  
*On entry:* indicates how  $Q$  or  $Q^T$  or  $P$  or  $P^T$  is to be applied to  $C$ .  
 SIDE = 'L'  
 $Q$  or  $Q^T$  or  $P$  or  $P^T$  is applied to  $C$  from the left.  
 SIDE = 'R'  
 $Q$  or  $Q^T$  or  $P$  or  $P^T$  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  $Q^T$  or  $P^T$  is to be applied to  $C$ .  
 TRANS = 'N'  
 $Q$  or  $P$  is applied to  $C$ .  
 TRANS = 'T'  
 $Q^T$  or  $P^T$  is applied to  $C$ .  
*Constraint:* TRANS = 'N' or 'T'.
- 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:* 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:*  $K \geq 0$ .
- 7: A(LDA,\*) – REAL (KIND=nag\_wp) array *Input*  
**Note:** the second dimension of the array  $A$  must be at least  $\max(1, \min(r, K))$  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 F08KEF (DGEHRD).
- 8: LDA – INTEGER *Input*  
*On entry:* the first dimension of the array  $A$  as declared in the (sub)program from which F08KGF (DORMBR) is called.  
*Constraints:*  
 if VECT = 'Q',  $LDA \geq \max(1, r)$ ;  
 if VECT = 'P',  $LDA \geq \max(1, \min(r, K))$ .
- 9: TAU(\*) – REAL (KIND=nag\_wp) array *Input*  
**Note:** the dimension of the array TAU must be at least  $\max(1, \min(r, K))$ .  
*On entry:* further details of the elementary reflectors, as returned by F08KEF (DGEHRD) in its parameter TAUQ if VECT = 'Q', or in its parameter TAUP if VECT = 'P'.

- 10: C(LDC,\*) – REAL (KIND=nag\_wp) array Input/Output  
**Note:** the second dimension of the array C must be at least  $\max(1, N)$ .  
*On entry:* the matrix C.  
*On exit:* C is overwritten by  $QC$  or  $Q^T C$  or  $CQ$  or  $C^T Q$  or  $PC$  or  $P^T C$  or  $CP$  or  $C^T P$  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 F08KGF (DORMBR) is called.  
*Constraint:*  $LDC \geq \max(1, M)$ .
- 12: WORK(max(1, LWORK)) – REAL (KIND=nag\_wp) array Workspace  
*On exit:* if INFO = 0, 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 F08KGF (DORMBR) 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,  $LWORK \geq N \times nb$  if SIDE = 'L' and at least  $M \times nb$  if SIDE = 'R', where *nb* is the optimal **block size**.  
*Constraints:*  
     if SIDE = 'L',  $LWORK \geq \max(1, N)$  or LWORK = -1;  
     if SIDE = 'R',  $LWORK \geq \max(1, M)$  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  $\epsilon$  is the **machine precision**.

## 8 Further Comments

The total number of floating point operations is approximately

if SIDE = 'L' and  $m \geq k$ ,  $2nk(2m - k)$ ;

if SIDE = 'R' and  $n \geq k$ ,  $2mk(2n - k)$ ;

if SIDE = 'L' and  $m < k$ ,  $2m^2n$ ;

if SIDE = 'R' and  $n < k$ ,  $2mn^2$ ,

where  $k$  is the value of the parameter K.

The complex analogue of this routine is F08KUF (ZUNMBR).

## 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

$$A = \begin{pmatrix} -0.57 & -1.28 & -0.39 & 0.25 \\ -1.93 & 1.08 & -0.31 & -2.14 \\ 2.30 & 0.24 & 0.40 & -0.35 \\ -1.93 & 0.64 & -0.66 & 0.08 \\ 0.15 & 0.30 & 0.15 & -2.13 \\ -0.02 & 1.03 & -1.43 & 0.50 \end{pmatrix}.$$

The routine first performs a  $QR$  factorization of  $A$  as  $A = Q_a R$  and then reduces the factor  $R$  to bidiagonal form  $B$ :  $R = Q_b B P^T$ . Finally it forms  $Q_a$  and calls F08KGF (DORMBR) to form  $Q = Q_a Q_b$ .

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

$$A = \begin{pmatrix} -5.42 & 3.28 & -3.68 & 0.27 & 2.06 & 0.46 \\ -1.65 & -3.40 & -3.20 & -1.03 & -4.06 & -0.01 \\ -0.37 & 2.35 & 1.90 & 4.31 & -1.76 & 1.13 \\ -3.15 & -0.11 & 1.99 & -2.70 & 0.26 & 4.50 \end{pmatrix}.$$

The routine first performs an  $LQ$  factorization of  $A$  as  $A = L P_a^T$  and then reduces the factor  $L$  to bidiagonal form  $B$ :  $L = Q_b B P_b^T$ . Finally it forms  $P_b^T$  and calls F08KGF (DORMBR) to form  $P^T = P_b^T P_a^T$ .

### 9.1 Program Text

Program f08kgfe

```
!      F08KGF Example Program Text
!
!      Mark 24 Release. NAG Copyright 2012.
!
!      .. Use Statements ..
!      Use nag_library, Only: dgebrd, dgelqf, dgeqrf, dorglq, dorgqr, dormbr, &
!                               f06qff, f06qhf, nag_wp, x04caf
!      .. Implicit None Statement ..
!      Implicit None
!      .. Parameters ..
!      Real (Kind=nag_wp), Parameter      :: zero = 0.0E0_nag_wp
!      Integer, Parameter                 :: nin = 5, nout = 6
!      .. Local Scalars ..
!      Integer                             :: i, ic, ifail, info, lda, ldpt, ldu, &
!                                           lwork, m, n
!      .. Local Arrays ..
!      Real (Kind=nag_wp), Allocatable    :: a(:,,:), d(:), e(:), pt(:,,:), tau(:), &
!                                           taup(:), tauq(:), u(:,,:), work(:)
!      .. Executable Statements ..
!      Write (nout,*) 'F08KGF Example Program Results'
!      Skip heading in data file
!      Read (nin,*)
!      Do ic = 1, 2
!         Read (nin,*) m, n
!         lda = m
!         ldpt = n
!         ldu = m
```

```

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

!
! 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 dgeqrf is f08aef
!   Call dgeqrf(m,n,a,lda,tau,work,lwork,info)

!
!   Copy A to U
!   Call f06qff('Lower',m,n,a,lda,u,ldu)

!
!   Form Q explicitly, storing the result in U
!   The NAG name equivalent of dorgqr is f08aff
!   Call dorgqr(m,n,n,u,ldu,tau,work,lwork,info)

!
!   Copy R to PT (used as workspace)
!   Call f06qff('Upper',n,n,a,lda,pt,ldpt)

!
!   Set the strictly lower triangular part of R to zero
!   Call f06qhf('Lower',n-1,n-1,zero,zero,pt(2,1),ldpt)

!
!   Bidiagonalize R
!   The NAG name equivalent of dgebrd is f08kef
!   Call dgebrd(n,n,pt,ldpt,d,e,tauq,taup,work,lwork,info)

!
!   Update Q, storing the result in U
!   The NAG name equivalent of dormbr is f08kgf
!   Call dormbr('Q','Right','No transpose',m,n,n,pt,ldpt,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 x04caf('General',' ',m,n,u,ldu,'Example 1: matrix Q',ifail)

Else

!
!   Compute the LQ factorization of A
!   The NAG name equivalent of dgelqf is f08ahf
!   Call dgelqf(m,n,a,lda,tau,work,lwork,info)

!
!   Copy A to PT
!   Call f06qff('Upper',m,n,a,lda,pt,ldpt)

!
!   Form Q explicitly, storing the result in PT
!   The NAG name equivalent of dorglq is f08ajf
!   Call dorglq(n,n,m,pt,ldpt,tau,work,lwork,info)

!
!   Copy L to U (used as workspace)
!   Call f06qff('Lower',m,m,a,lda,u,ldu)

!
!   Set the strictly upper triangular part of L to zero
!   Call f06qhf('Upper',m-1,m-1,zero,zero,u(1,2),ldu)

```

```

!      Bidiagonalize L
!      The NAG name equivalent of dgebrd is f08kef
!      Call dgebrd(m,m,u,ldu,d,e,tauq,taup,work,lwork,info)

!      Update P**T, storing the result in PT
!      The NAG name equivalent of dormbr is f08kgf
!      Call dormbr('P','Left','Transpose',m,n,m,u,ldu,taup,pt,ldpt,work, &
!      lwork,info)

!      Print bidiagonal form and matrix P**T

      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 x04caf('General',' ',m,n,pt,ldpt,'Example 2: matrix P**T', &
      ifail)

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

99999 Format (3X,(8F8.4))
      End Program f08kgfe

```

## 9.2 Program Data

```

F08KGF Example Program Data
  6 4                               :Values of M and N, Example 1
-0.57 -1.28 -0.39  0.25
-1.93  1.08 -0.31 -2.14
 2.30  0.24  0.40 -0.35
-1.93  0.64 -0.66  0.08
 0.15  0.30  0.15 -2.13
-0.02  1.03 -1.43  0.50           :End of matrix A
 4 6                               :Values of M and N, Example 2
-5.42  3.28 -3.68  0.27  2.06  0.46
-1.65 -3.40 -3.20 -1.03 -4.06 -0.01
-0.37  2.35  1.90  4.31 -1.76  1.13
-3.15 -0.11  1.99 -2.70  0.26  4.50 :End of matrix A

```

## 9.3 Program Results

F08KGF Example Program Results

Example 1: bidiagonal matrix B

Diagonal

3.6177 -2.4161 1.9213 -1.4265

Super-diagonal

1.2587 -1.5262 1.1895

Example 1: matrix Q

	1	2	3	4
1	-0.1576	-0.2690	0.2612	0.8513
2	-0.5335	0.5311	-0.2922	0.0184
3	0.6358	0.3495	-0.0250	-0.0210
4	-0.5335	0.0035	0.1537	-0.2592
5	0.0415	0.5572	-0.2917	0.4523
6	-0.0055	0.4614	0.8585	-0.0532

Example 2: bidiagonal matrix B

Diagonal

-7.7724 6.1573 -6.0576 5.7933

Super-diagonal

1.1926 0.5734 -1.9143

Example 2: matrix P\*\*T

	1	2	3	4	5	6
1	-0.7104	0.4299	-0.4824	0.0354	0.2700	0.0603
2	0.3583	0.1382	-0.4110	0.4044	0.0951	-0.7148
3	-0.0507	0.4244	0.3795	0.7402	-0.2773	0.2203
4	0.2442	0.4016	0.4158	-0.1354	0.7666	-0.0137

---