NAG Library Routine Document

F08KFF (DORGBR)

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

F08KFF (DORGBR) generates one of the real orthogonal matrices Q or P^{T} which were determined by F08KEF (DGEBRD) when reducing a real matrix to bidiagonal form.

2 Specification

```
SUBROUTINE F08KFF (VECT, M, N, K, A, LDA, TAU, WORK, LWORK, INFO)

INTEGER            M, N, K, LDA, LWORK, INFO
REAL (KIND=nag_wp) A(LDA,*), TAU(*), WORK(max(1,LWORK))
CHARACTER(1)       VECT
```

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

3 Description

F08KFF (DORGBR) 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: $A = QBP^{\mathrm{T}}$. F08KEF (DGEBRD) represents the matrices Q and P^{T} as products of elementary reflectors.

This routine may be used to generate Q or P^{T} explicitly as square matrices, or in some cases just the leading columns of Q or the leading rows of P^{T}.

The various possibilities are specified by the parameters VECT, M, N and K. The appropriate values to cover the most likely cases are as follows (assuming that A was an m by n matrix):

1. To form the full m by m matrix Q:

   ```
   CALL DORGBR('Q',m,m,n,...)
   ```

   (note that the array A must have at least m columns).

2. If $m > n$, to form the n leading columns of Q:

   ```
   CALL DORGBR('Q',m,n,n,...)
   ```

3. To form the full n by n matrix P^{T}:

   ```
   CALL DORGBR('P',n,n,m,...)
   ```

   (note that the array A must have at least n rows).

4. If $m < n$, to form the m leading rows of P^{T}:

   ```
   CALL DORGBR('P',m,n,m,...)
   ```

4 References

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

5 Parameters

1: VECT – CHARACTER(1) *Input*

On entry: indicates whether the orthogonal matrix Q or P^{T} is generated.

VECT = 'Q'
Q is generated.

VECT = 'P'
P^{T} is generated.

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

2: M – INTEGER *Input*

On entry: m, the number of rows of the orthogonal matrix Q or P^{T} to be returned.

Constraint: M ≥ 0.

3: N – INTEGER *Input*

On entry: n, the number of columns of the orthogonal matrix Q or P^{T} to be returned.

Constraints:

N ≥ 0;
if VECT = 'Q' and M $>$ K, M $\geq$ N $\geq$ K;
if VECT = 'Q' and M $\leq$ K, M = N;
if VECT = 'P' and N $>$ K, N $\geq$ M $\geq$ K;
if VECT = 'P' and N $\leq$ K, N = M.

4: 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.

5: A(LDA,$*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array A must be at least max(1, N).

On entry: details of the vectors which define the elementary reflectors, as returned by F08KEF (DGEBRD).

On exit: the orthogonal matrix Q or P^{T}, or the leading rows or columns thereof, as specified by VECT, M and N.

6: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08KFF (DORGBR) is called.

Constraint: LDA $\geq$ max(1, M).

7: TAU($*$) – REAL (KIND=nag_wp) array *Input*

Note: the dimension of the array TAU must be at least max(1, min(M, K)) if VECT = 'Q' and at least max(1, min(N, K)) if VECT = 'P'.

On entry: further details of the elementary reflectors, as returned by F08KEF (DGEBRD) in its parameter TAUQ if VECT = 'Q', or in its parameter TAUP if VECT = 'P'.

8: 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.

9: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08KFF (DORGBR) 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 ≥ min(M, N) × *nb*, where *nb* is the optimal ***block size***.

Constraint: LWORK ≥ max(1, min(M, N)) or LWORK = −1.

10: 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 matrix Q differs from an exactly orthogonal matrix by a matrix E such that

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

where ϵ is the ***machine precision***. A similar statement holds for the computed matrix P^{T}.

8 Further Comments

The total number of floating point operations for the cases listed in Section 3 are approximately as follows:

1. To form the whole of Q:

   $\frac{4}{3}n\left(3m^2 - 3mn + n^2\right)$ if $m > n$,

   $\frac{4}{3}m^3$ if $m \leq n$;

2. To form the n leading columns of Q when $m > n$:

   $\frac{2}{3}n^2(3m - n)$;

3. To form the whole of P^{T}:

   $\frac{4}{3}n^3$ if $m \geq n$,

   $\frac{4}{3}m\left(3n^2 - 3mn + m^2\right)$ if $m < n$;

4. To form the m leading rows of P^{T} when $m < n$:

   $\frac{2}{3}m^2(3n - m)$.

The complex analogue of this routine is F08KTF (ZUNGBR).

9 Example

For this routine two examples are presented, both of which involve computing the singular value decomposition of a matrix A, where

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

in the first example and

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

in the second. A must first be reduced to tridiagonal form by F08KEF (DGEBRD). The program then calls F08KFF (DORGBR) twice to form Q and P^{T}, and passes these matrices to F08MEF (DBDSQR), which computes the singular value decomposition of A.

9.1 Program Text

```
    Program f08kffe

!     F08KFF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dbdsqr, dgebrd, dorgbr, f06qff, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                           :: i, ic, ifail, info, lda, ldc, ldu,   &
                                           ldvt, lwork, m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable   :: a(:,:), c(:,:), d(:), e(:), taup(:), &
                                           tauq(:), u(:,:), vt(:,:), work(:)
!     .. Executable Statements ..
      Write (nout,*) 'F08KFF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Do ic = 1, 2
        Read (nin,*) m, n
        lda = m
        ldc = n
        ldu = m
        ldvt = n
        lwork = 64*(m+n)
        Allocate (a(lda,n),c(ldc,n),d(n),e(n-1),taup(n),tauq(n),u(ldu,n), &
          vt(ldvt,n),work(lwork))

!       Read A from data file

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

!       Reduce A to bidiagonal form
!       The NAG name equivalent of dgebrd is f08kef
        Call dgebrd(m,n,a,lda,d,e,tauq,taup,work,lwork,info)

        If (m>=n) Then
```

```
!         Copy A to VT and U

          Call f06qff('Upper',n,n,a,lda,vt,ldvt)
          Call f06qff('Lower',m,n,a,lda,u,ldu)

!         Form P**T explicitly, storing the result in VT
!         The NAG name equivalent of dorgbr is f08kff
          Call dorgbr('P',n,n,m,vt,ldvt,taup,work,lwork,info)

!         Form Q explicitly, storing the result in U
!         The NAG name equivalent of dorgbr is f08kff
          Call dorgbr('Q',m,n,n,u,ldu,tauq,work,lwork,info)

!         Compute the SVD of A
!         The NAG name equivalent of dbdsqr is f08mef
          Call dbdsqr('Upper',n,n,m,0,d,e,vt,ldvt,u,ldu,c,ldc,work,info)

!         Print singular values, left & right singular vectors

          Write (nout,*)
          Write (nout,*) 'Example 1: singular values'
          Write (nout,99999) d(1:n)
          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',' ',n,n,vt,ldvt, &
            'Example 1: right singular vectors, by row',ifail)

          Write (nout,*)
          Flush (nout)

          Call x04caf('General',' ',m,n,u,ldu, &
            'Example 1: left singular vectors, by column',ifail)

        Else

!         Copy A to VT and U

          Call f06qff('Upper',m,n,a,lda,vt,ldvt)
          Call f06qff('Lower',m,m,a,lda,u,ldu)

!         Form P**T explicitly, storing the result in VT
!         The NAG name equivalent of dorgbr is f08kff
          Call dorgbr('P',m,n,m,vt,ldvt,taup,work,lwork,info)

!         Form Q explicitly, storing the result in U
!         The NAG name equivalent of dorgbr is f08kff
          Call dorgbr('Q',m,m,n,u,ldu,tauq,work,lwork,info)

!         Compute the SVD of A
!         The NAG name equivalent of dbdsqr is f08mef
          Call dbdsqr('Lower',m,n,m,0,d,e,vt,ldvt,u,ldu,c,ldc,work,info)

!         Print singular values, left & right singular vectors

          Write (nout,*)
          Write (nout,*) 'Example 2: singular values'
          Write (nout,99999) d(1:m)
          Write (nout,*)
          Flush (nout)

          ifail = 0
          Call x04caf('General',' ',m,n,vt,ldvt, &
            'Example 2: right singular vectors, by row',ifail)

          Write (nout,*)
          Flush (nout)
```

```
          Call x04caf('General',' ',m,m,u,ldu, &
            'Example 2: left singular vectors, by column',ifail)

        End If
        Deallocate (a,c,d,e,taup,tauq,u,vt,work)
      End Do

99999 Format (3X,(8F8.4))
    End Program f08kffe
```

9.2 Program Data

```
F08KFF 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

```
 F08KFF Example Program Results

 Example 1: singular values
     3.9987  3.0005  1.9967  0.9999

 Example 1: right singular vectors, by row
          1       2       3       4
 1   0.8251 -0.2794  0.2048  0.4463
 2  -0.4530 -0.2121 -0.2622  0.8252
 3  -0.2829 -0.7961  0.4952 -0.2026
 4   0.1841 -0.4931 -0.8026 -0.2807

 Example 1: left singular vectors, by column
          1       2       3       4
 1  -0.0203  0.2794  0.4690  0.7692
 2  -0.7284 -0.3464 -0.0169 -0.0383
 3   0.4393 -0.4955 -0.2868  0.0822
 4  -0.4678  0.3258 -0.1536 -0.1636
 5  -0.2200 -0.6428  0.1125  0.3572
 6  -0.0935  0.1927 -0.8132  0.4957

 Example 2: singular values
     7.9987  7.0059  5.9952  4.9989

 Example 2: right singular vectors, by row
          1       2       3       4       5       6
 1  -0.7933  0.3163 -0.3342 -0.1514  0.2142  0.3001
 2   0.1002  0.6442  0.4371  0.4890  0.3771  0.0501
 3   0.0111  0.1724 -0.6367  0.4354 -0.0430 -0.6111
 4   0.2361  0.0216 -0.1025 -0.5286  0.7460 -0.3120

 Example 2: left singular vectors, by column
          1       2       3       4
 1   0.8884  0.1275  0.4331  0.0838
 2   0.0733 -0.8264  0.1943 -0.5234
 3  -0.0361  0.5435  0.0756 -0.8352
 4   0.4518 -0.0733 -0.8769 -0.1466
```