NAG Toolbox

nag_lapack_dorgbr (f08kf)

1 Purpose

nag_lapack_dorgbr (f08kf) generates one of the real orthogonal matrices Q or P^{T} which were determined by nag_lapack_dgebrd (f08ke) when reducing a real matrix to bidiagonal form.

2 Syntax

```
[a, info] = nag_lapack_dorgbr(vect, k, a, tau, 'm', m, 'n', n)
[a, info] = f08kf(vect, k, a, tau, 'm', m, 'n', n)
```

3 Description

nag_lapack_dorgbr (f08kf) is intended to be used after a call to nag_lapack_dgebrd (f08ke), which reduces a real rectangular matrix A to bidiagonal form B by an orthogonal transformation: $A = QBP^{\mathrm{T}}$. nag_lapack_dgebrd (f08ke) represents the matrices Q and P^{T} as products of elementary reflectors.

This function 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 arguments **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:

   ```
   [a, info] = f08kf('Q', n, a, tau);
   ```

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

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

   ```
   [a, info] = f08kf('Q', n, a, tau);
   ```

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

   ```
   [a, info] = f08kf('P', m, a, tau);
   ```

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

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

   ```
   [a, info] = f08kf('P', m, a, tau);
   ```

4 References

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

5 Parameters

5.1 Compulsory Input Parameters

1: **vect** – CHARACTER(1)

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: **k** – INTEGER

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: $\mathbf{k} \geq 0$.

3: $\mathbf{a}(lda,:)$ – REAL (KIND=nag_wp) array

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

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

Details of the vectors which define the elementary reflectors, as returned by nag_lapack_dgebrd (f08ke).

4: $\mathbf{tau}(:)$ – REAL (KIND=nag_wp) array

The dimension of the array **tau** must be at least $\max(1, \min(\mathbf{m}, \mathbf{k}))$ if **vect** = 'Q' and at least $\max(1, \min(\mathbf{n}, \mathbf{k}))$ if **vect** = 'P'

Further details of the elementary reflectors, as returned by nag_lapack_dgebrd (f08ke) in its argument **tauq** if **vect** = 'Q', or in its argument **taup** if **vect** = 'P'.

5.2 Optional Input Parameters

1: **m** – INTEGER

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

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

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

2: **n** – INTEGER

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

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

Constraints:

$\mathbf{n} \geq 0$;
if **vect** = 'Q' and $\mathbf{m} > \mathbf{k}$, $\mathbf{m} \geq \mathbf{n} \geq \mathbf{k}$;
if **vect** = 'Q' and $\mathbf{m} \leq \mathbf{k}$, $\mathbf{m} = \mathbf{n}$;
if **vect** = 'P' and $\mathbf{n} > \mathbf{k}$, $\mathbf{n} \geq \mathbf{m} \geq \mathbf{k}$;
if **vect** = 'P' and $\mathbf{n} \leq \mathbf{k}$, $\mathbf{n} = \mathbf{m}$.

5.3 Output Parameters

1: $\mathbf{a}(lda,:)$ – REAL (KIND=nag_wp) array

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

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

The orthogonal matrix Q or P^{T}, or the leading rows or columns thereof, as specified by **vect**, **m** and **n**.

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: **vect**, 2: **m**, 3: **n**, 4: **k**, 5: **a**, 6: **lda**, 7: **tau**, 8: **work**, 9: **lwork**, 10: **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 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 function is nag_lapack_zungbr (f08kt).

9 Example

For this function 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 nag_lapack_dgebrd (f08ke). The program then calls nag_lapack_dorgbr (f08kf) twice to form Q and P^{T}, and passes these matrices to nag_lapack_dbdsqr (f08me), which computes the singular value decomposition of A.

9.1 Program Text

```
     function f08kf_example

fprintf('f08kf example results\n\n');

% Two cases of performing Singular Value Decomposition
% Case 1: m > n
ex1;
% Case 2: m < n
ex2;

function ex1

  m = nag_int(6);
  n = nag_int(4);
  a = [-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];

  % Reduce A to bidiagonal form
  [B, d, e, tauq, taup, info] =  ...
  f08ke(a);

  % Form P^T explicitly (n-by-n, k = m)
  vect = 'P';
  [PT, info] = f08kf( ...
                      vect, m, B, taup, 'm', n, 'n', n);
  % Form Q explicitly (m-by-n, k = n)
  vect = 'Q';
  [Q, info] = f08kf( ...
                     vect, n, B, tauq, 'm', m, 'n', n);

  % Compute SVD of A using its bidiagonal form.
  c = [];
  uplo = 'Upper';
  [s, ~, VT, U, c, info] = f08me( ...
                                  uplo, d, e, PT, Q, c);

  disp('Example 1: singular values');
  disp(s(1:n)');
  disp('Example 1: right singular vectors, by row');
  disp(VT(1:n,:));
  disp('Example 1: left singular vectors, by column');
  disp(U);

function ex2
  m = nag_int(4);
  n = nag_int(6);
  a = [-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];

  % Reduce A to bidiagonal form
  [B, d, e, tauq, taup, info] =  ...
  f08ke(a);

  % Form P^T explicitly (m-by-n, k = m)
  vect = 'P';
  [PT, info] = f08kf( ...
                      vect, m, B, taup, 'm', m, 'n', n);
```

```
  % Form Q explicitly (m-by-m, k = n)
  vect = 'Q';
  [Q, info] = f08kf( ...
                     vect, n, B, tauq, 'm', m, 'n', m);

  % Compute SVD of A using its bidiagonal form.
  c = [];
  uplo = 'Lower';
  [s, ~, VT, U, c, info] = f08me( ...
                                  uplo, d, e, PT, Q, c, 'n', m);

  disp('Example 2: singular values');
  disp(s(1:m)');
  disp('Example 2: right singular vectors, by row');
  disp(VT);
  disp('Example 2: left singular vectors, by column');
  disp(U(:,1:m));
```

9.2 Program Results

```
     f08kf example results

Example 1: singular values
    3.9987    3.0005    1.9967    0.9999

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

Example 1: left singular vectors, by column
   -0.0203    0.2794    0.4690    0.7692
   -0.7284   -0.3464   -0.0169   -0.0383
    0.4393   -0.4955   -0.2868    0.0822
   -0.4678    0.3258   -0.1536   -0.1636
   -0.2200   -0.6428    0.1125    0.3572
   -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
   -0.7933    0.3163   -0.3342   -0.1514    0.2142    0.3001
    0.1002    0.6442    0.4371    0.4890    0.3771    0.0501
    0.0111    0.1724   -0.6367    0.4354   -0.0430   -0.6111
    0.2361    0.0216   -0.1025   -0.5286    0.7460   -0.3120

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