NAG Toolbox

nag_lapack_zungbr (f08kt)

1 Purpose

nag_lapack_zungbr (f08kt) generates one of the complex unitary matrices Q or P^{H} which were determined by nag_lapack_zgebrd (f08ks) when reducing a complex matrix to bidiagonal form.

2 Syntax

```
[a, info] = nag_lapack_zungbr(vect, k, a, tau, 'm', m, 'n', n)
[a, info] = f08kt(vect, k, a, tau, 'm', m, 'n', n)
```

3 Description

nag_lapack_zungbr (f08kt) is intended to be used after a call to nag_lapack_zgebrd (f08ks), which reduces a complex rectangular matrix A to real bidiagonal form B by a unitary transformation: $A = QBP^{\mathrm{H}}$. nag_lapack_zgebrd (f08ks) represents the matrices Q and P^{H} as products of elementary reflectors.

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

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] = f08kt('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] = f08kt('Q', n, a(1:m,1:n), tau);
   ```

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

   ```
   [a, info] = f08kt('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^{H}:

   ```
   [a, info] = f08kt('P', m, a(1:m,1:n), 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 unitary matrix Q or P^{H} is generated.

vect = 'Q'
Q is generated.

vect = 'P'
P^{H} 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: **a**$(lda,:)$ – COMPLEX (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_zgebrd (f08ks).

4: **tau**(:) – COMPLEX (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_zgebrd (f08ks) 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 unitary matrix Q or P^{H} 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 unitary matrix Q or P^{H} 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: **a**$(lda,:)$ – COMPLEX (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 unitary matrix Q or P^{H}, 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 unitary 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^{H}.

8 Further Comments

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

1. To form the whole of Q:

   $\frac{16}{3}n\left(3m^2 - 3mn + n^2\right)$ if $m > n$,

   $\frac{16}{3}m^3$ if $m \leq n$;

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

   $\frac{8}{3}n^2(3m - n)$;

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

   $\frac{16}{3}n^3$ if $m \geq n$,

   $\frac{16}{3}m^3\left(3n^2 - 3mn + m^2\right)$ if $m < n$;

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

   $\frac{8}{3}m^2(3n - m)$.

The real analogue of this function is nag_lapack_dorgbr (f08kf).

9 Example

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

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

in the first example and

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

in the second. A must first be reduced to tridiagonal form by nag_lapack_zgebrd (f08ks). The program

then calls nag_lapack_zungbr (f08kt) twice to form Q and P^{H}, and passes these matrices to nag_lapack_zbdsqr (f08ms), which computes the singular value decomposition of A.

9.1 Program Text

```
     function f08kt_example

fprintf('f08kt 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 = [ 0.96 - 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];

  % Reduce A to bidiagonal form
  [B, d, e, tauq, taup, info] =  ...
  f08ks(a);

  % Form PT explicitly (n-by-n, k = m)
  vect = 'P';
  [PT, info] = f08kt( ...
                      vect, m, B, taup, 'm', n, 'n', n);

  % Form Q explicitly (m-by-n, k = n)
  vect = 'Q';
  [Q, info] = f08kt( ...
                     vect, n, B, tauq, 'm', m, 'n', n);

  % Compute SVD of A using its bidiagonal form.
  c = [];
  uplo = 'Upper';
  [s, ~, VT, U, c, info] = f08ms( ...
                                  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(3);
  n = nag_int(4);
  a = [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];

  % Reduce A to bidiagonal form
  [B, d, e, tauq, taup, info] =  ...
  f08ks(a);

  % Form P^T explicitly (m-by-n, k = m)
  vect = 'P';
  [PT, info] = f08kt( ...
                      vect, m, B, taup, 'm', m, 'n', n);
  % Form Q explicitly (m-by-m, k = n)
  vect = 'Q';
```

```
  [Q, info] = f08kt( ...
                     vect, n, B, tauq, 'm', m, 'n', m);

  % Compute SVD of A using its bidiagonal form.
  c = [];
  uplo = 'Lower';
  [s, ~, VT, U, c, info] = f08ms( ...
                                  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

```
     f08kt example results

Example 1: singular values
    3.9994    3.0003    1.9944    0.9995

Example 1: right singular vectors, by row
  -0.6971 + 0.0000i  -0.0867 - 0.3548i   0.0560 - 0.5400i  -0.1878 - 0.2253i
   0.2403 + 0.0000i   0.0725 - 0.2336i  -0.2477 - 0.5291i   0.7026 + 0.2177i
  -0.5123 + 0.0000i  -0.3030 - 0.1735i   0.0678 + 0.5162i   0.4418 + 0.3864i
  -0.4403 + 0.0000i   0.5294 + 0.6361i  -0.3027 - 0.0346i   0.1667 + 0.0258i

Example 1: left singular vectors, by column
  -0.5634 + 0.0016i  -0.2687 - 0.2749i   0.2451 + 0.4657i   0.3787 + 0.2987i
   0.1205 - 0.6108i  -0.2909 + 0.1085i   0.4329 - 0.1758i  -0.0182 - 0.0437i
  -0.0816 + 0.1613i  -0.1660 + 0.3885i  -0.4667 + 0.3821i  -0.0800 - 0.2276i
   0.1441 - 0.1532i   0.1984 - 0.1737i  -0.0034 + 0.1555i   0.2608 - 0.5382i
  -0.2487 - 0.0926i   0.6253 + 0.3304i   0.2643 - 0.0194i   0.1002 + 0.0140i
  -0.3758 + 0.0793i  -0.0307 - 0.0816i   0.1266 + 0.1747i  -0.4175 - 0.4058i

Example 2: singular values
    3.0004    1.9967    0.9973

Example 2: right singular vectors, by row
   0.2454 - 0.0001i   0.2942 - 0.5843i   0.0162 - 0.0810i   0.6794 + 0.2083i
  -0.1692 + 0.5194i   0.1915 - 0.4374i   0.5205 - 0.0244i  -0.3149 - 0.3208i
  -0.5553 + 0.1403i   0.1438 - 0.1507i  -0.5684 - 0.5505i  -0.0318 - 0.0378i

Example 2: left singular vectors, by column
   0.6518 + 0.0000i  -0.4312 + 0.0000i   0.6239 + 0.0000i
  -0.4437 - 0.5027i  -0.3794 + 0.1026i   0.2014 + 0.5961i
  -0.2012 + 0.2916i  -0.8122 + 0.0030i  -0.3511 - 0.3026i
```