NAG Toolbox

nag_lapack_dhgeqz (f08xe)

1 Purpose

nag_lapack_dhgeqz (f08xe) implements the QZ method for finding generalized eigenvalues of the real matrix pair (A, B) of order n, which is in the generalized upper Hessenberg form.

2 Syntax

```
[a, b, alphar, alphai, beta, q, z, info] = nag_lapack_dhgeqz(job, compq, compz,
ilo, ihi, a, b, q, z, 'n', n)
```

```
[a, b, alphar, alphai, beta, q, z, info] = f08xe(job, compq, compz, ilo, ihi, a,
b, q, z, 'n', n)
```

3 Description

nag_lapack_dhgeqz (f08xe) implements a single-double-shift version of the QZ method for finding the generalized eigenvalues of the real matrix pair (A, B) which is in the generalized upper Hessenberg form. If the matrix pair (A, B) is not in the generalized upper Hessenberg form, then the function nag_lapack_dgghrd (f08we) should be called before invoking nag_lapack_dhgeqz (f08xe).

This problem is mathematically equivalent to solving the equation

$$\det(A - \lambda B) = 0.$$

Note that, to avoid underflow, overflow and other arithmetic problems, the generalized eigenvalues λ_j are never computed explicitly by this function but defined as ratios between two computed values, α_j and β_j:

$$\lambda_j = \alpha_j / \beta_j.$$

The arguments α_j, in general, are finite complex values and β_j are finite real non-negative values.

If desired, the matrix pair (A, B) may be reduced to generalized Schur form. That is, the transformed matrix B is upper triangular and the transformed matrix A is block upper triangular, where the diagonal blocks are either 1 by 1 or 2 by 2. The 1 by 1 blocks provide generalized eigenvalues which are real and the 2 by 2 blocks give complex generalized eigenvalues.

The argument **job** specifies two options. If **job** $=$ 'S' then the matrix pair (A, B) is simultaneously reduced to Schur form by applying one orthogonal transformation (usually called Q) on the left and another (usually called Z) on the right. That is,

$$\begin{array}{c} A \leftarrow Q^{\mathrm{T}} A Z \\ B \leftarrow Q^{\mathrm{T}} B Z \end{array}$$

The 2 by 2 upper-triangular diagonal blocks of B corresponding to 2 by 2 blocks of **a** will be reduced to non-negative diagonal matrices. That is, if $\mathbf{a}(j+1, j)$ is nonzero, then $\mathbf{b}(j+1, j) = \mathbf{b}(j, j+1) = 0$ and $\mathbf{b}(j, j)$ and $\mathbf{b}(j+1, j+1)$ will be non-negative.

If **job** $=$ 'E', then at each iteration the same transformations are computed but they are only applied to those parts of A and B which are needed to compute α and β. This option could be used if generalized eigenvalues are required but not generalized eigenvectors.

If **job** $=$ 'S' and **compq** $=$ 'V' or 'I', and **compz** $=$ 'V' or 'I', then the orthogonal transformations used to reduce the pair (A, B) are accumulated into the input arrays **q** and **z**. If generalized eigenvectors are required then **job** must be set to **job** $=$ 'S' and if left (right) generalized eigenvectors are to be computed then **compq** (**compz**) must be set to **compq** $=$ 'V' or 'I' and not **compq** $\neq$ 'N'.

If **compq** = 'I', then eigenvectors are accumulated on the identity matrix and on exit the array **q** contains the left eigenvector matrix Q. However, if **compq** = 'V' then the transformations are accumulated on the user-supplied matrix Q_0 in array **q** on entry and thus on exit **q** contains the matrix product QQ_0. A similar convention is used for **compz**.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia

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

Moler C B and Stewart G W (1973) An algorithm for generalized matrix eigenproblems *SIAM J. Numer. Anal.* **10** 241–256

Stewart G W and Sun J-G (1990) *Matrix Perturbation Theory* Academic Press, London

5 Parameters

5.1 Compulsory Input Parameters

1: **job** – CHARACTER(1)

Specifies the operations to be performed on (A, B).

job = 'E'
The matrix pair (A, B) on exit might not be in the generalized Schur form.

job = 'S'
The matrix pair (A, B) on exit will be in the generalized Schur form.

Constraint: **job** = 'E' or 'S'.

2: **compq** – CHARACTER(1)

Specifies the operations to be performed on Q:

compq = 'N'
The array **q** is unchanged.

compq = 'V'
The left transformation Q is accumulated on the array **q**.

compq = 'I'
The array **q** is initialized to the identity matrix before the left transformation Q is accumulated in **q**.

Constraint: **compq** = 'N', 'V' or 'I'.

3: **compz** – CHARACTER(1)

Specifies the operations to be performed on Z.

compz = 'N'
The array **z** is unchanged.

compz = 'V'
The right transformation Z is accumulated on the array **z**.

compz = 'I'
The array **z** is initialized to the identity matrix before the right transformation Z is accumulated in **z**.

Constraint: **compz** = 'N', 'V' or 'I'.

4: **ilo** – INTEGER

5: **ihi** – INTEGER

The indices i_{lo} and i_{hi}, respectively which define the upper triangular parts of A. The submatrices $A(1 : i_{\mathrm{lo}} - 1, 1 : i_{\mathrm{lo}} - 1)$ and $A(i_{\mathrm{hi}} + 1 : n, i_{\mathrm{hi}} + 1 : n)$ are then upper triangular. These arguments are provided by nag_lapack_dggbal (f08wh) if the matrix pair was previously balanced; otherwise, **ilo** $= 1$ and **ihi** $=$ **n**.

Constraints:

if **n** > 0, $1 \leq$ **ilo** $\leq$ **ihi** $\leq$ **n**;
if **n** $= 0$, **ilo** $= 1$ and **ihi** $= 0$.

6: **a**$(lda, :)$ – REAL (KIND=nag_wp) array

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

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

The n by n upper Hessenberg matrix A. The elements below the first subdiagonal must be set to zero.

7: **b**$(ldb, :)$ – REAL (KIND=nag_wp) array

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

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

The n by n upper triangular matrix B. The elements below the diagonal must be zero.

8: **q**$(ldq, :)$ – REAL (KIND=nag_wp) array

The first dimension, ldq, of the array **q** must satisfy

if **compq** = 'V' or 'I', $ldq \geq \mathbf{n}$;
if **compq** = 'N', $ldq \geq 1$.

The second dimension of the array **q** must be at least $\max(1, \mathbf{n})$ if **compq** = 'V' or 'I' and at least 1 if **compq** = 'N'.

If **compq** = 'V', the matrix Q_0. The matrix Q_0 is usually the matrix Q returned by nag_lapack_dgghrd (f08we).

If **compq** = 'N', **q** is not referenced.

9: **z**$(ldz, :)$ – REAL (KIND=nag_wp) array

The first dimension, ldz, of the array **z** must satisfy

if **compz** = 'V' or 'I', $ldz \geq \mathbf{n}$;
if **compz** = 'N', $ldz \geq 1$.

The second dimension of the array **z** must be at least $\max(1, \mathbf{n})$ if **compz** = 'V' or 'I' and at least 1 if **compz** = 'N'.

If **compz** = 'V', the matrix Z_0. The matrix Z_0 is usually the matrix Z returned by nag_lapack_dgghrd (f08we).

If **compz** = 'N', **z** is not referenced.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the second dimension of the arrays **a**, **b**, **q**, **z** and the first dimension of the arrays **a**, **b**, **q**, **z**. (An error is raised if these dimensions are not equal.)

n, the order of the matrices A, B, Q and Z.

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

5.3 Output Parameters

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

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

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

If **job** = 'S', the matrix pair (A,B) will be simultaneously reduced to generalized Schur form.

If **job** = 'E', the 1 by 1 and 2 by 2 diagonal blocks of the matrix pair (A,B) will give generalized eigenvalues but the remaining elements will be irrelevant.

2: $\mathbf{b}(ldb,:)$ – REAL (KIND=nag_wp) array

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

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

If **job** = 'S', the matrix pair (A,B) will be simultaneously reduced to generalized Schur form.

If **job** = 'E', the 1 by 1 and 2 by 2 diagonal blocks of the matrix pair (A,B) will give generalized eigenvalues but the remaining elements will be irrelevant.

3: $\mathbf{alphar}(\mathbf{n})$ – REAL (KIND=nag_wp) array

The real parts of α_j, for $j = 1,2,\ldots,n$.

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

The imaginary parts of α_j, for $j = 1,2,\ldots,n$.

5: $\mathbf{beta}(\mathbf{n})$ – REAL (KIND=nag_wp) array

β_j, for $j = 1,2,\ldots,n$.

6: $\mathbf{q}(ldq,:)$ – REAL (KIND=nag_wp) array

The first dimension, ldq, of the array **q** will be

if **compq** = 'V' or 'I', $ldq = \mathbf{n}$;
if **compq** = 'N', $ldq = 1$.

The second dimension of the array **q** will be $\max(1,\mathbf{n})$ if **compq** = 'V' or 'I' and at least 1 if **compq** = 'N'.

If **compq** = 'V', **q** contains the matrix product QQ_0.

If **compq** = 'I', **q** contains the transformation matrix Q.

7: $\mathbf{z}(ldz,:)$ – REAL (KIND=nag_wp) array

The first dimension, ldz, of the array **z** will be

if **compz** = 'V' or 'I', $ldz = \mathbf{n}$;
if **compz** = 'N', $ldz = 1$.

The second dimension of the array **z** will be $\max(1,\mathbf{n})$ if **compz** = 'V' or 'I' and at least 1 if **compz** = 'N'.

If **compz** = 'V', **z** contains the matrix product ZZ_0.

If **compz** = 'I', **z** contains the transformation matrix Z.

8: **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: **job**, 2: **compq**, 3: **compz**, 4: **n**, 5: **ilo**, 6: **ihi**, 7: **a**, 8: **lda**, 9: **b**, 10: **ldb**, 11: **alphar**, 12: **alphai**, 13: **beta**, 14: **q**, 15: **ldq**, 16: **z**, 17: **ldz**, 18: **work**, 19: **lwork**, 20: **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.

info > 0 (*warning*)

If $1 \leq$ **info** $\leq$ **n**, the QZ iteration did not converge and the matrix pair (A, B) is not in the generalized Schur form at exit. However, if **info** $<$ **n**, then the computed α_i and β_i should be correct for $i =$ **info** $+ 1, \ldots,$ **n**.

If **n** $+ 1 \leq$ **info** $\leq 2 \times$ **n**, the computation of shifts failed and the matrix pair (A, B) is not in the generalized Schur form at exit. However, if **info** $< 2 \times$ **n**, then the computed α_i and β_i should be correct for $i =$ **info** $-$ **n** $+ 1, \ldots,$ **n**.

If **info** $> 2 \times$ **n**, then an unexpected Library error has occurred. Please contact NAG with details of your program.

7 Accuracy

Please consult Section 4.11 of the LAPACK Users' Guide (see Anderson *et al.* (1999)) and Chapter 6 of Stewart and Sun (1990), for more information.

8 Further Comments

nag_lapack_dhgeqz (f08xe) is the fifth step in the solution of the real generalized eigenvalue problem and is called after nag_lapack_dgghrd (f08we).

The complex analogue of this function is nag_lapack_zhgeqz (f08xs).

9 Example

This example computes the α and β arguments, which defines the generalized eigenvalues, of the matrix pair (A, B) given by

$$A = \begin{pmatrix} 1.0 & 1.0 & 1.0 & 1.0 & 1.0 \\ 2.0 & 4.0 & 8.0 & 16.0 & 32.0 \\ 3.0 & 9.0 & 27.0 & 81.0 & 243.0 \\ 4.0 & 16.0 & 64.0 & 256.0 & 1024.0 \\ 5.0 & 25.0 & 125.0 & 625.0 & 3125.0 \end{pmatrix}$$

$$B = \begin{pmatrix} 1.0 & 2.0 & 3.0 & 4.0 & 5.0 \\ 1.0 & 4.0 & 9.0 & 16.0 & 25.0 \\ 1.0 & 8.0 & 27.0 & 64.0 & 125.0 \\ 1.0 & 16.0 & 81.0 & 256.0 & 625.0 \\ 1.0 & 32.0 & 243.0 & 1024.0 & 3125.0 \end{pmatrix}.$$

This requires calls to five functions: nag_lapack_dggbal (f08wh) to balance the matrix, nag_lapack_dgeqrf (f08ae) to perform the QR factorization of B, nag_lapack_dormqr (f08ag) to apply Q to A, nag_lapack_dgghrd (f08we) to reduce the matrix pair to the generalized Hessenberg form and nag_lapack_dhgeqz (f08xe) to compute the eigenvalues using the QZ algorithm.

9.1 Program Text

```
function f08xe_example

fprintf('f08xe example results\n\n');

a = [ 1.0   1.0    1.0    1.0     1.0;
      2.0   4.0    8.0   16.0    32.0;
      3.0   9.0   27.0   81.0   243.0;
      4.0  16.0   64.0  256.0  1024.0;
      5.0  25.0  125.0  625.0  3125.0];
b = a';

%' Balance A and B
job = 'B';
[a, b, ilo, ihi, lscale, rscale, info] = ...
  f08wh(job, a, b);

bbal = b(ilo:ihi,ilo:ihi);
abal = a(ilo:ihi,ilo:ihi);

% QR factorize balanced B
[QR, tau, info] = f08ae(bbal);

% Perform C = Q^T*A
side = 'Left';
trans = 'Transpose';
[c, info] = f08ag(...
                  side, trans, QR, tau, abal);

% Generalized Hessenberg form (C,R) -> (H,T)
compq = 'No Q';
compz = 'No Z';
z = eye(4);
q = eye(4);
jlo = nag_int(1);
jhi = nag_int(ihi-ilo+1);
[H, T, ~, ~, info] = ...
  f08we(...
        compq, compz, jlo, jhi, c, QR, q, z);

% Find eigenvalues of generalized Hessenberg form
%    = eigenvalues of (A,B).
job = 'Eigenvalues';
[~, ~, alphar, alphai, beta, ~, ~, info] = ...
  f08xe(...
        job, compq, compz, jlo, jhi, H, T, q, z);

disp('Generalized eigenvalues of (A,B):');
w = complex(alphar+i*alphai);
disp(w./beta);
```

9.2 Program Results

```
f08xe example results

Generalized eigenvalues of (A,B):
  -2.4367 + 0.0000i
   0.6069 + 0.7948i
   0.6069 - 0.7948i
   1.0000 + 0.0000i
  -0.4104 + 0.0000i
```