NAG Toolbox

nag_lapack_dggevx (f08wb)

1 Purpose

nag_lapack_dggevx (f08wb) computes for a pair of n by n real nonsymmetric matrices (A, B) the generalized eigenvalues and, optionally, the left and/or right generalized eigenvectors using the QZ algorithm.

Optionally it also computes a balancing transformation to improve the conditioning of the eigenvalues and eigenvectors, reciprocal condition numbers for the eigenvalues, and reciprocal condition numbers for the right eigenvectors.

2 Syntax

```
[a, b, alphar, alphai, beta, vl, vr, ilo, ihi, lscale, rscale, abnrm, bbnrm,
rconde, rcondv, info] = nag_lapack_dggevx(balanc, jobvl, jobvr, sense, a, b,
'n', n)
```

```
[a, b, alphar, alphai, beta, vl, vr, ilo, ihi, lscale, rscale, abnrm, bbnrm,
rconde, rcondv, info] = f08wb(balanc, jobvl, jobvr, sense, a, b, 'n', n)
```

3 Description

A generalized eigenvalue for a pair of matrices (A, B) is a scalar λ or a ratio $\alpha/\beta = \lambda$, such that $A - \lambda B$ is singular. It is usually represented as the pair (α, β), as there is a reasonable interpretation for $\beta = 0$, and even for both being zero.

The right eigenvector v_j corresponding to the eigenvalue λ_j of (A, B) satisfies

$$Av_j = \lambda_j B v_j.$$

The left eigenvector u_j corresponding to the eigenvalue λ_j of (A, B) satisfies

$$u_j^{\mathrm{H}} A = \lambda_j u_j^{\mathrm{H}} B,$$

where u_j^{H} is the conjugate-transpose of u_j.

All the eigenvalues and, if required, all the eigenvectors of the generalized eigenproblem $Ax = \lambda Bx$, where A and B are real, square matrices, are determined using the QZ algorithm. The QZ algorithm consists of four stages:

1. A is reduced to upper Hessenberg form and at the same time B is reduced to upper triangular form.
2. A is further reduced to quasi-triangular form while the triangular form of B is maintained. This is the real generalized Schur form of the pair (A, B).
3. The quasi-triangular form of A is reduced to triangular form and the eigenvalues extracted. This function does not actually produce the eigenvalues λ_j, but instead returns α_j and β_j such that

   $$\lambda_j = \alpha_j/\beta_j, \quad j = 1, 2, \dots, n.$$

   The division by β_j becomes your responsibility, since β_j may be zero, indicating an infinite eigenvalue. Pairs of complex eigenvalues occur with α_j/β_j and α_{j+1}/β_{j+1} complex conjugates, even though α_j and α_{j+1} are not conjugate.
4. If the eigenvectors are required they are obtained from the triangular matrices and then transformed back into the original coordinate system.

For details of the balancing option, see Section 3 in nag_lapack_dggbal (f08wh).

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 http://www.netlib.org/lapack/lug

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

Wilkinson J H (1979) Kronecker's canonical form and the QZ algorithm *Linear Algebra Appl.* **28** 285–303

5 Parameters

5.1 Compulsory Input Parameters

1: **balanc** – CHARACTER(1)

Specifies the balance option to be performed.

balanc = 'N'
Do not diagonally scale or permute.

balanc = 'P'
Permute only.

balanc = 'S'
Scale only.

balanc = 'B'
Both permute and scale.

Computed reciprocal condition numbers will be for the matrices after permuting and/or balancing. Permuting does not change condition numbers (in exact arithmetic), but balancing does. In the absence of other information, **balanc** = 'B' is recommended.

Constraint: **balanc** = 'N', 'P', 'S' or 'B'.

2: **jobvl** – CHARACTER(1)

If **jobvl** = 'N', do not compute the left generalized eigenvectors.

If **jobvl** = 'V', compute the left generalized eigenvectors.

Constraint: **jobvl** = 'N' or 'V'.

3: **jobvr** – CHARACTER(1)

If **jobvr** = 'N', do not compute the right generalized eigenvectors.

If **jobvr** = 'V', compute the right generalized eigenvectors.

Constraint: **jobvr** = 'N' or 'V'.

4: **sense** – CHARACTER(1)

Determines which reciprocal condition numbers are computed.

sense = 'N'
None are computed.

sense = 'E'
Computed for eigenvalues only.

sense = 'V'
Computed for eigenvectors only.

sense = 'B'
Computed for eigenvalues and eigenvectors.

Constraint: **sense** = 'N', 'E', 'V' or 'B'.

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

The first dimension of the array **a** must be at least max(1, **n**).

The second dimension of the array **a** must be at least max(1, **n**).

The matrix A in the pair (A, B).

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

The first dimension of the array **b** must be at least max(1, **n**).

The second dimension of the array **b** must be at least max(1, **n**).

The matrix B in the pair (A, B).

5.2 Optional Input Parameters

1: **n** – INTEGER

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

n, the order of the matrices A and B.

Constraint: **n** ≥ 0.

5.3 Output Parameters

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

The first dimension of the array **a** will be max(1, **n**).

The second dimension of the array **a** will be max(1, **n**).

a has been overwritten. If **jobvl** = 'V' or **jobvr** = 'V' or both, then A contains the first part of the real Schur form of the 'balanced' versions of the input A and B.

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

The first dimension of the array **b** will be max(1, **n**).

The second dimension of the array **b** will be max(1, **n**).

b has been overwritten.

3: **alphar**(**n**) – REAL (KIND=nag_wp) array

The element **alphar**(j) contains the real part of α_j.

4: **alphai**(**n**) – REAL (KIND=nag_wp) array

The element **alphai**(j) contains the imaginary part of α_j.

5: **beta**(**n**) – REAL (KIND=nag_wp) array

(**alphar**(j) + **alphai**(j) $\times i$)/**beta**(j), for $j = 1, 2, \ldots,$ **n**, will be the generalized eigenvalues.

If **alphai**(j) is zero, then the jth eigenvalue is real; if positive, then the jth and $(j+1)$st eigenvalues are a complex conjugate pair, with **alphai**($j+1$) negative.

Note: the quotients **alphar**(j)/**beta**(j) and **alphai**(j)/**beta**(j) may easily overflow or underflow, and **beta**(j) may even be zero. Thus, you should avoid naively computing the ratio α_j/β_j.

However, $\max|\alpha_j|$ will always be less than and usually comparable with $\|\mathbf{a}\|_2$ in magnitude, and $\max|\beta_j|$ will always be less than and usually comparable with $\|\mathbf{b}\|_2$.

6: **vl**$(ldvl, :)$ – REAL (KIND=nag_wp) array

The first dimension, $ldvl$, of the array **vl** will be

if **jobvl** $=$ 'V', $ldvl = \max(1, \mathbf{n})$;
otherwise $ldvl = 1$.

The second dimension of the array **vl** will be $\max(1, \mathbf{n})$ if **jobvl** $=$ 'V' and 1 otherwise.

If **jobvl** $=$ 'V', the left generalized eigenvectors u_j are stored one after another in the columns of **vl**, in the same order as the corresponding eigenvalues. Each eigenvector will be scaled so the largest component will have |real part| + |imag. part| = 1.

If **jobvl** $=$ 'N', **vl** is not referenced.

7: **vr**$(ldvr, :)$ – REAL (KIND=nag_wp) array

The first dimension, $ldvr$, of the array **vr** will be

if **jobvr** $=$ 'V', $ldvr = \max(1, \mathbf{n})$;
otherwise $ldvr = 1$.

The second dimension of the array **vr** will be $\max(1, \mathbf{n})$ if **jobvr** $=$ 'V' and 1 otherwise.

If **jobvr** $=$ 'V', the right generalized eigenvectors v_j are stored one after another in the columns of **vr**, in the same order as the corresponding eigenvalues. Each eigenvector will be scaled so the largest component will have |real part| + |imag. part| = 1.

If **jobvr** $=$ 'N', **vr** is not referenced.

8: **ilo** – INTEGER
9: **ihi** – INTEGER

ilo and **ihi** are integer values such that $\mathbf{a}(i,j) = 0$ and $\mathbf{b}(i,j) = 0$ if $i > j$ and $j = 1, 2, \ldots, \mathbf{ilo} - 1$ or $i = \mathbf{ihi} + 1, \ldots, \mathbf{n}$.

If **balanc** $=$ 'N' or 'S', **ilo** $= 1$ and **ihi** $=$ **n**.

10: **lscale**$(\mathbf{n})$ – REAL (KIND=nag_wp) array

Details of the permutations and scaling factors applied to the left side of A and B.

If pl_j is the index of the row interchanged with row j, and dl_j is the scaling factor applied to row j, then:

lscale$(j) = pl_j$, for $j = 1, 2, \ldots, \mathbf{ilo} - 1$;

lscale $= dl_j$, for $j = \mathbf{ilo}, \ldots, \mathbf{ihi}$;

lscale $= pl_j$, for $j = \mathbf{ihi} + 1, \ldots, \mathbf{n}$.

The order in which the interchanges are made is **n** to **ihi** $+ 1$, then 1 to **ilo** $- 1$.

11: **rscale**$(\mathbf{n})$ – REAL (KIND=nag_wp) array

Details of the permutations and scaling factors applied to the right side of A and B.

If pr_j is the index of the column interchanged with column j, and dr_j is the scaling factor applied to column j, then:

rscale$(j) = pr_j$, for $j = 1, 2, \ldots, \mathbf{ilo} - 1$;

if **rscale** $= dr_j$, for $j = \mathbf{ilo}, \ldots, \mathbf{ihi}$;

if **rscale** $= pr_j$, for $j = \mathbf{ihi} + 1, \ldots, \mathbf{n}$.

The order in which the interchanges are made is **n** to **ihi** $+ 1$, then 1 to **ilo** $- 1$.

12: **abnrm** – REAL (KIND=nag_wp)

The 1-norm of the balanced matrix A.

13: **bbnrm** – REAL (KIND=nag_wp)

The 1-norm of the balanced matrix B.

14: **rconde**(:) – REAL (KIND=nag_wp) array

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

If **sense** $=$ 'E' or 'B', the reciprocal condition numbers of the eigenvalues, stored in consecutive elements of the array. For a complex conjugate pair of eigenvalues two consecutive elements of **rconde** are set to the same value. Thus **rconde**(j), **rcondv**(j), and the jth columns of **vl** and **vr** all correspond to the jth eigenpair.

If **sense** $=$ 'V', **rconde** is not referenced.

15: **rcondv**(:) – REAL (KIND=nag_wp) array

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

If **sense** $=$ 'V' or 'B', the estimated reciprocal condition numbers of the eigenvectors, stored in consecutive elements of the array. For a complex eigenvector two consecutive elements of **rcondv** are set to the same value.

If **sense** $=$ 'E', **rcondv** is not referenced.

16: **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: **balanc**, 2: **jobvl**, 3: **jobvr**, 4: **sense**, 5: **n**, 6: **a**, 7: **lda**, 8: **b**, 9: **ldb**, 10: **alphar**, 11: **alphai**, 12: **beta**, 13: **vl**, 14: **ldvl**, 15: **vr**, 16: **ldvr**, 17: **ilo**, 18: **ihi**, 19: **lscale**, 20: **rscale**, 21: **abnrm**, 22: **bbnrm**, 23: **rconde**, 24: **rcondv**, 25: **work**, 26: **lwork**, 27: **iwork**, 28: **bwork**, 29: **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 $= 1$ to **n** (*warning*)

The QZ iteration failed. No eigenvectors have been calculated, but **alphar**(j), **alphai**(j), and **beta**(j) should be correct for $j =$ **info** $+ 1, \ldots,$ **n**.

info $=$ **n** $+ 1$

Unexpected error returned from nag_lapack_dhgeqz (f08xe).

info $=$ **n** $+ 2$

Error returned from nag_lapack_dtgevc (f08yk).

7 Accuracy

The computed eigenvalues and eigenvectors are exact for a nearby matrices $(A + E)$ and $(B + F)$, where

$$\|(E, F)\|_F = O(\epsilon)\|(A, B)\|_F,$$

and ϵ is the ***machine precision***.

An approximate error bound on the chordal distance between the ith computed generalized eigenvalue w and the corresponding exact eigenvalue λ is

$$\epsilon \times \|\mathbf{abnrm}, \mathbf{bbnrm}\|_2/\mathbf{rconde}(i).$$

An approximate error bound for the angle between the ith computed eigenvector u_j or v_j is given by

$$\epsilon \times \|\mathbf{abnrm}, \mathbf{bbnrm}\|_2/\mathbf{rcondv}(i).$$

For further explanation of the reciprocal condition numbers **rconde** and **rcondv**, see Section 4.11 of Anderson *et al.* (1999).

Note: interpretation of results obtained with the QZ algorithm often requires a clear understanding of the effects of small changes in the original data. These effects are reviewed in Wilkinson (1979), in relation to the significance of small values of α_j and β_j. It should be noted that if α_j and β_j are **both** small for any j, it may be that no reliance can be placed on **any** of the computed eigenvalues $\lambda_i = \alpha_i/\beta_i$. You are recommended to study Wilkinson (1979) and, if in difficulty, to seek expert advice on determining the sensitivity of the eigenvalues to perturbations in the data.

8 Further Comments

The total number of floating-point operations is proportional to n^3.

The complex analogue of this function is nag_lapack_zggevx (f08wp).

9 Example

This example finds all the eigenvalues and right eigenvectors of the matrix pair (A, B), where

$$A = \begin{pmatrix} 3.9 & 12.5 & -34.5 & -0.5 \\ 4.3 & 21.5 & -47.5 & 7.5 \\ 4.3 & 21.5 & -43.5 & 3.5 \\ 4.4 & 26.0 & -46.0 & 6.0 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 1.0 & 2.0 & -3.0 & 1.0 \\ 1.0 & 3.0 & -5.0 & 4.0 \\ 1.0 & 3.0 & -4.0 & 3.0 \\ 1.0 & 3.0 & -4.0 & 4.0 \end{pmatrix},$$

together with estimates of the condition number and forward error bounds for each eigenvalue and eigenvector. The option to balance the matrix pair is used.

Note that the block size (NB) of 64 assumed in this example is not realistic for such a small problem, but should be suitable for large problems.

9.1 Program Text

```
   function f08wb_example

fprintf('f08wb example results\n\n');

% Generalized eigenvalues and right eigenvectors of (A, B):
n = 4;
a = [3.9, 12.5, -34.5, -0.5;
     4.3, 21.5, -47.5,  7.5;
     4.3, 21.5, -43.5,  3.5;
     4.4, 26,   -46,    6];
b = [1,    2,    -3,    1;
     1,    3,    -5,    4;
     1,    3,    -4,    3;
     1,    3,    -4,    4];

balanc = 'Balance';
```

```
jobvl = 'No vectors (left)';
jobvr = 'Vectors (right)';
sense = 'Both reciprocal condition numbers';
[~, ~, alphar, alphai, beta, ~, VR, ~, ~, ~, ~, abnrm, bbnrm, ...
      rconde, rcondv, info] = ...
  f08wb( ...
  balanc, jobvl, jobvr, sense, a, b);

epsilon = x02aj;
small   = x02am;
absnrm  = sqrt(abnrm^2+bbnrm^2);
tol     = epsilon*absnrm;

for j=1:n

  % display information on the jth eigenvalue
  if (abs(alphar(j)) + abs(alphai(j)))*small >= abs(beta(j))
    fprintf('\n%4d: Eigenvalue is numerically infinite or undetermined\n',j);
    fprintf('%4d: alphar = %11.4e, alphai = %11.4e, beta = %11.4e\n', ...
     j, alphar(j), alphai(j), beta(j));
  else
    fprintf('\n%4d: Eigenvalue =  ', j);
    if alphai(j)==0
      fprintf('%10.4e\n',alphar(j)/beta(j));
    elseif alphai(j)>=0
      fprintf('%10.4e + %10.4ei\n',alphar(j)/beta(j), alphai(j)/beta(j));
    else
      fprintf('%10.4e - %10.4ei\n',alphar(j)/beta(j), -alphai(j)/beta(j));
    end
  end

  if rconde(j) > 0
    fprintf('\n      Condition number            = %8.1e\n', 1/rconde(j));
    fprintf('      Error bound                 = %8.1e\n', tol/rconde(j));
  else
    fprintf('\n      Reciprocal condition number = %8.1e\n', rconde(j));
    fprintf('      Error bound is infinite\n');
  end

  fprintf('\n      Eigenvector:\n');
  if alphai(j) == 0
    fprintf('%30.4e\n',VR(:, j));
  else
    if alphai(j) > 0
      k = j;
    else
      k = j-1;
    end
    for l = 1:n
      if VR(l,k+1)>=0
        fprintf('%30.4e + %10.4ei\n',VR(l, k), VR(l, k+1));
      else
        fprintf('%30.4e - %10.4ei\n',VR(l, k), -VR(l, k+1));
      end
    end
  end

  if rcondv(j) > 0
    fprintf('\n      Condition number            = %8.1e\n', 1/rcondv(j));
    fprintf('      Error bound                 = %8.1e\n', tol/rcondv(j));
  else
    fprintf('\n      Reciprocal condition number = %8.1e\n', rcondv(j));
    fprintf('      Error bound is infinite\n');
  end
end
```

9.2 Program Results

```
      f08wb example results

    1: Eigenvalue =  2.0000e+00

       Condition number            =  1.1e+01
       Error bound                 =  2.5e-14

       Eigenvector:
                    -1.0000e+00
                    -5.7143e-03
                    -6.2857e-02
                    -6.2857e-02

       Condition number            =  8.0e+00
       Error bound                 =  1.9e-14

    2: Eigenvalue =  3.0000e+00 + 4.0000e+00i

       Condition number            =  6.1e+00
       Error bound                 =  1.4e-14

       Eigenvector:
                    -4.2550e-01 - 5.7450e-01i
                    -8.5099e-02 - 1.1490e-01i
                    -1.4298e-01 - 8.6125e-04i
                    -1.4298e-01 - 8.6125e-04i

       Condition number            =  2.6e+01
       Error bound                 =  6.2e-14

    3: Eigenvalue =  3.0000e+00 - 4.0000e+00i

       Condition number            =  6.1e+00
       Error bound                 =  1.4e-14

       Eigenvector:
                    -4.2550e-01 - 5.7450e-01i
                    -8.5099e-02 - 1.1490e-01i
                    -1.4298e-01 - 8.6125e-04i
                    -1.4298e-01 - 8.6125e-04i

       Condition number            =  2.6e+01
       Error bound                 =  6.2e-14

    4: Eigenvalue =  4.0000e+00

       Condition number            =  1.9e+00
       Error bound                 =  4.6e-15

       Eigenvector:
                    -1.0000e+00
                    -1.1111e-02
                     3.3333e-02
                    -1.5556e-01

       Condition number            =  1.4e+01
       Error bound                 =  3.3e-14
```