NAG Toolbox

nag_lapack_zgebal (f08nv)

1 Purpose

nag_lapack_zgebal (f08nv) balances a complex general matrix in order to improve the accuracy of computed eigenvalues and/or eigenvectors.

2 Syntax

```
[a, ilo, ihi, scale, info] = nag_lapack_zgebal(job, a, 'n', n)
[a, ilo, ihi, scale, info] = f08nv(job, a, 'n', n)
```

3 Description

nag_lapack_zgebal (f08nv) balances a complex general matrix A. The term 'balancing' covers two steps, each of which involves a similarity transformation of A. The function can perform either or both of these steps.

1. The function first attempts to permute A to block upper triangular form by a similarity transformation:

$$PAP^{\mathrm{T}} = A' = \begin{pmatrix} A'_{11} & A'_{12} & A'_{13} \\ 0 & A'_{22} & A'_{23} \\ 0 & 0 & A'_{33} \end{pmatrix}$$

   where P is a permutation matrix, and A'_{11} and A'_{33} are upper triangular. Then the diagonal elements of A'_{11} and A'_{33} are eigenvalues of A. The rest of the eigenvalues of A are the eigenvalues of the central diagonal block A'_{22}, in rows and columns i_{lo} to i_{hi}. Subsequent operations to compute the eigenvalues of A (or its Schur factorization) need only be applied to these rows and columns; this can save a significant amount of work if $i_{\mathrm{lo}} > 1$ and $i_{\mathrm{hi}} < n$. If no suitable permutation exists (as is often the case), the function sets $i_{\mathrm{lo}} = 1$ and $i_{\mathrm{hi}} = n$, and A'_{22} is the whole of A.

2. The function applies a diagonal similarity transformation to A', to make the rows and columns of A'_{22} as close in norm as possible:

$$A'' = DA'D^{-1} = \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} A'_{11} & A'_{12} & A'_{13} \\ 0 & A'_{22} & A'_{23} \\ 0 & 0 & A'_{33} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22}^{-1} & 0 \\ 0 & 0 & I \end{pmatrix}.$$

   This scaling can reduce the norm of the matrix (i.e., $\left\|A''_{22}\right\| < \left\|A'_{22}\right\|$) and hence reduce the effect of rounding errors on the accuracy of computed eigenvalues and eigenvectors.

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: **job** – CHARACTER(1)

Indicates whether A is to be permuted and/or scaled (or neither).

job = 'N'
A is neither permuted nor scaled (but values are assigned to **ilo**, **ihi** and **scale**).

job = 'P'
A is permuted but not scaled.

job = 'S'
A is scaled but not permuted.

job = 'B'
A is both permuted and scaled.

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

2: **a**$(lda,:)$ – COMPLEX (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 matrix A.

5.2 Optional Input Parameters

1: **n** – INTEGER

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

n, the order of the matrix A.

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

5.3 Output Parameters

1: **a**$(lda,:)$ – COMPLEX (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 $\max(1,\mathbf{n})$.

a stores the balanced matrix. If **job** = 'N', **a** is not referenced.

2: **ilo** – INTEGER

3: **ihi** – INTEGER

The values i_{lo} and i_{hi} such that on exit $\mathbf{a}(i,j)$ is zero if $i > j$ and $1 \leq j < i_{\mathrm{lo}}$ or $i_{\mathrm{hi}} < i \leq n$.

If **job** = 'N' or 'S', $i_{\mathrm{lo}} = 1$ and $i_{\mathrm{hi}} = n$.

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

Details of the permutations and scaling factors applied to A. More precisely, if p_j is the index of the row and column interchanged with row and column j and d_j is the scaling factor used to balance row and column j then

$$\mathbf{scale}(j) = \begin{cases} p_j, & j = 1,2,\ldots,i_{\mathrm{lo}}-1 \\ d_j, & j = i_{\mathrm{lo}}, i_{\mathrm{lo}}+1,\ldots,i_{\mathrm{hi}} \\ p_j, & j = i_{\mathrm{hi}}+1, i_{\mathrm{hi}}+2,\ldots,n. \end{cases} \quad \text{and}$$

The order in which the interchanges are made is n to $i_{\mathrm{hi}}+1$ then 1 to $i_{\mathrm{lo}}-1$.

5: **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: **n**, 3: **a**, 4: **lda**, 5: **ilo**, 6: **ihi**, 7: **scale**, 8: **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 errors are negligible, compared with those in subsequent computations.

8 Further Comments

If the matrix A is balanced by nag_lapack_zgebal (f08nv), then any eigenvectors computed subsequently are eigenvectors of the matrix A'' (see Section 3) and hence nag_lapack_zgebak (f08nw) **must** then be called to transform them back to eigenvectors of A.

If the Schur vectors of A are required, then this function must **not** be called with **job** = 'S' or 'B', because then the balancing transformation is not unitary. If this function is called with **job** = 'P', then any Schur vectors computed subsequently are Schur vectors of the matrix A'', and nag_lapack_zgebak (f08nw) **must** be called (with **side** = 'R') to transform them back to Schur vectors of A.

The total number of real floating-point operations is approximately proportional to n^2.

The real analogue of this function is nag_lapack_dgebal (f08nh).

9 Example

This example computes all the eigenvalues and right eigenvectors of the matrix A, where

$$A = \begin{pmatrix} 1.50 - 2.75i & 0.00 + 0.00i & 0.00 + 0.00i & 0.00 + 0.00i \\ -8.06 - 1.24i & -2.50 - 0.50i & 0.00 + 0.00i & -0.75 + 0.50i \\ -2.09 + 7.56i & 1.39 + 3.97i & -1.25 + 0.75i & -4.82 - 5.67i \\ 6.18 + 9.79i & -0.92 - 0.62i & 0.00 + 0.00i & -2.50 - 0.50i \end{pmatrix}.$$

The program first calls nag_lapack_zgebal (f08nv) to balance the matrix; it then computes the Schur factorization of the balanced matrix, by reduction to Hessenberg form and the QR algorithm. Then it calls nag_lapack_ztrevc (f08qx) to compute the right eigenvectors of the balanced matrix, and finally calls nag_lapack_zgebak (f08nw) to transform the eigenvectors back to eigenvectors of the original matrix A.

9.1 Program Text

```
function f08nv_example

fprintf('f08nv example results\n\n');

n = nag_int(4);
a = [  1.50 - 2.75i,  0.00 + 0.00i,  0.00 + 0.00i,  0.00 + 0.00i;
      -8.06 - 1.24i, -2.50 - 0.50i,  0.00 + 0.00i, -0.75 + 0.50i;
      -2.09 + 7.56i,  1.39 + 3.97i, -1.25 + 0.75i, -4.82 - 5.67i;
       6.18 + 9.79i, -0.92 - 0.62i,  0.00 + 0.00i, -2.50 - 0.50i];

% Balance A
job = 'Both';
```

```
[a, ilo, ihi, scale, info] = f08nv(job, a);

% Reduce a to upper Hessenberg form
[H, tau, info] = f08ns(ilo, ihi, a);

% Form Q
[Q, info] = f08nt(ilo, ihi, H, tau);

% Calculate the eigenvalues and Schur factorisation of A
[H, w, Z, info] = f08ps( ...
                         'Schur Form', 'Vectors', ilo, ihi, H, Q);

disp('Eigenvalues:');
disp(w);

% Calculate the eigenvectors of A
[select, ~, VR, m, info] = ...
f08qx( ...
       'Right', 'Backtransform', false, H, complex(zeros(1)), Z, n);
% Rescale
[VR, info] = f08nw( ...
                    'Both', 'Right', ilo, ihi, scale, VR);

% Normalize: largest elements are real
for i = 1:n
  [~,k] = max(abs(real(VR(:,i)))+abs(imag(VR(:,i))));
  VR(:,i) = VR(:,i)*conj(VR(k,i))/abs(VR(k,i))/norm(VR(:,i));
end

disp('Eigenvectors:');
disp(VR);
```

9.2 Program Results

```
     f08nv example results

Eigenvalues:
  -1.2500 + 0.7500i
  -1.5000 - 0.4975i
  -3.5000 - 0.5025i
   1.5000 - 2.7500i

Eigenvectors:
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.1418 - 0.0407i
   0.0000 + 0.0000i  -0.1015 + 0.0009i   0.1756 - 0.4131i  -0.2711 - 0.1812i
   1.0000 + 0.0000i   0.9884 + 0.0000i   0.7420 + 0.0000i   0.8213 + 0.0000i
   0.0000 + 0.0000i   0.0941 + 0.0619i   0.4170 - 0.2722i   0.1110 + 0.4303i
```