NAG C Library Function Document

nag_dhsein (f08pkc)

1 Purpose

nag_dhsein (f08pkc) computes selected left and/or right eigenvectors of a real upper Hessenberg matrix corresponding to specified eigenvalues, by inverse iteration.

2 Specification

```
void nag_dhsein (Nag_OrderType order, Nag_SideType side,
     Nag_EigValsSourceType eig_source, Nag_InitVeenumtype initv, Boolean select[],
     Integer n, const double h[], Integer pdh, double wr[], const double wi[],
     double vl[], Integer pdvl, double vr[], Integer pdvr, Integer mm, Integer *m,
     Integer ifaill[], Integer ifailr[], NagError *fail)
```

3 Description

nag_dhsein (f08pkc) computes left and/or right eigenvectors of a real upper Hessenberg matrix H, corresponding to selected eigenvalues.

The right eigenvector x, and the left eigenvector y, corresponding to an eigenvalue λ, are defined by:

$$Hx = \lambda x \text{ and } y^H H = \lambda y^H \text{ (or } H^T y = \bar{\lambda} y).$$

Note that even though H is real, λ, x and y may be complex. If x is an eigenvector corresponding to a complex eigenvalue λ, then the complex conjugate vector $\bar{x}$ is the eigenvector corresponding to the complex conjugate eigenvalue $\bar{\lambda}$.

The eigenvectors are computed by inverse iteration. They are scaled so that, for a real eigenvector x, $\max(|x_i|) = 1$, and for a complex eigenvector, $\max(|\mathrm{Re}(x_i)| + |\mathrm{Im}(x_i)|) = 1$.

If H has been formed by reduction of a real general matrix A to upper Hessenberg form, then eigenvectors of H may be transformed to eigenvectors of A by a call to nag_dormhr (f08ngc).

4 References

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

5 Parameters

1: **order** – Nag_OrderType *Input*

On entry: the **order** parameter specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by **order** = **Nag_RowMajor**. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

Constraint: **order** = **Nag_RowMajor** or **Nag_ColMajor**.

2: **side** – Nag_SideType *Input*

On entry: indicates whether left and/or right eigenvectors are to be computed as follows:

if **side** = **Nag_RightSide**, then only right eigenvectors are computed;

if **side** = **Nag_LeftSide**, then only left eigenvectors are computed;

if **side** = **Nag_BothSides**, then both left and right eigenvectors are computed.

Constraint: **side** = **Nag_RightSide**, **Nag_LeftSide** or **Nag_BothSides**.

3: **eig_source** – Nag_EigValsSourceType *Input*

On entry: indicates whether the eigenvalues of H (stored in **wr** and **wi**) were found using nag_dhseqr (f08pec) as follows:

if **eig_source** = **Nag_HSEQRSource**, then the eigenvalues of H were found using nag_dhseqr (f08pec); thus if H has any zero sub-diagonal elements (and so is block triangular), then the jth eigenvalue can be assumed to be an eigenvalue of the block containing the jth row/column. This property allows the function to perform inverse iteration on just one diagonal block;

if **eig_source** = **Nag_NotKnown**, then no such assumption is made and the function performs inverse iteration using the whole matrix.

Constraint: **eig_source** = **Nag_HSEQRSource** or **Nag_NotKnown**.

4: **initv** – Nag_InitVeenumtype *Input*

On entry: indicates whether the user is supplying initial estimates for the selected eigenvectors as follows:

if **initv** = **Nag_NoVec**, no initial estimates are supplied;

if **initv** = **Nag_UserVec**, initial estimates are supplied in **vl** and/or **vr**.

Constraint: **initv** = **Nag_NoVec** or **Nag_UserVec**.

5: **select**[dim] – Boolean *Input/Output*

Note: the dimension, dim, of the array **select** must be at least $\max(1, \mathbf{n})$.

On entry: **select** specifies which eigenvectors are to be computed. To obtain the real eigenvector corresponding to the real eigenvalue **wr**[j], **select**[j] must be set **TRUE**. To select the complex eigenvector corresponding to the complex eigenvalue (**wr**[j], **wi**[j]) with complex conjugate (**wr**[$j+1$], **wi**[$j+1$]), **select**[j] and/or **select**[$j+1$] must be set **TRUE**; the eigenvector corresponding to the **first** eigenvalue in the pair is computed.

On exit: if a complex eigenvector was selected as specified above, then **select**[j] is set to **TRUE** and **select**[$j+1$] to **FALSE**.

6: **n** – Integer *Input*

On entry: n, the order of the matrix H.

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

7: **h**[dim] – const double *Input*

Note: the dimension, dim, of the array **h** must be at least $\max(1, \mathbf{pdh} \times \mathbf{n})$.

If **order** = **Nag_ColMajor**, the (i,j)th element of the matrix H is stored in $\mathbf{h}[(j-1) \times \mathbf{pdh} + i - 1]$ and if **order** = **Nag_RowMajor**, the (i,j)th element of the matrix H is stored in $\mathbf{h}[(i-1) \times \mathbf{pdh} + j - 1]$.

On entry: the n by n upper Hessenberg matrix H.

8: **pdh** – Integer *Input*

On entry: the stride separating matrix row or column elements (depending on the value of **order**) in the array **h**.

Constraint: $\mathbf{pdh} \geq \max(1, \mathbf{n})$.

9: **wr**[dim] – double *Input/Output*
10: **wi**[dim] – const double *Input*

Note: the dimensions, dim, of the arrays **wr** and **wi** must each be at least $\max(1, \mathbf{n})$.

On entry: the real and imaginary parts, respectively, of the eigenvalues of the matrix H. Complex conjugate pairs of values must be stored in consecutive elements of the arrays. If **eig_source** = **Nag_HSEQRSource**, the arrays **must** be exactly as returned by nag_dhseqr (f08pec).

On exit: some elements of **wr** may be modified, as close eigenvalues are perturbed slightly in searching for independent eigenvectors.

11: **vl**[dim] – double *Input/Output*

Note: the dimension, dim, of the array **vl** must be at least
- $\max(1, \mathbf{pdvl} \times \mathbf{mm})$ when **side** = **Nag_LeftSide** or **Nag_BothSides** and **order** = **Nag_ColMajor**;
- $\max(1, \mathbf{pdvl} \times \mathbf{n})$ when **side** = **Nag_LeftSide** or **Nag_BothSides** and **order** = **Nag_RowMajor**;
- 1 when **side** = **Nag_RightSide**.

If **order** = **Nag_ColMajor**, the (i, j)th element of the matrix is stored in $\mathbf{vl}[(j-1) \times \mathbf{pdvl} + i - 1]$ and if **order** = **Nag_RowMajor**, the (i, j)th element of the matrix is stored in $\mathbf{vl}[(i-1) \times \mathbf{pdvl} + j - 1]$.

On entry: if **initv** = **Nag_UserVec** and **side** = **Nag_LeftSide** or **Nag_BothSides**, **vl** must contain starting vectors for inverse iteration for the left eigenvectors. Each starting vector must be stored in the same rows or columns as will be used to store the corresponding eigenvector (see below). If **initv** = **Nag_NoVec**, **vl** need not be set.

On exit: if **side** = **Nag_LeftSide** or **Nag_BothSides**, **vl** contains the computed left eigenvectors (as specified by **select**). The eigenvectors are stored consecutively in the rows or columns of the array (depending on the value of **order**), in the same order as their eigenvalues. Corresponding to each selected real eigenvalue is a real eigenvector, occupying one row or column. Corresponding to each selected complex eigenvalue is a complex eigenvector, occupying two rows or columns: the first row or column holds the real part and the second row or column holds the imaginary part.

vl is not referenced if **side** = **Nag_RightSide**.

12: **pdvl** – Integer *Input*

On entry: the stride separating matrix row or column elements (depending on the value of **order**) in the array **vl**.

Constraints:

if **order** = **Nag_ColMajor**,
  if **side** = **Nag_LeftSide** or **Nag_BothSides**, $\mathbf{pdvl} \geq \max(1, \mathbf{n})$;
  if **side** = **Nag_RightSide**, $\mathbf{pdvl} \geq 1$;

if **order** = **Nag_RowMajor**,
  if **side** = **Nag_LeftSide** or **Nag_BothSides**, $\mathbf{pdvl} \geq \max(1, \mathbf{mm})$;
  if **side** = **Nag_RightSide**, $\mathbf{pdvl} \geq 1$.

13: **vr**[dim] – double *Input/Output*

Note: the dimension, dim, of the array **vr** must be at least
- $\max(1, \mathbf{pdvr} \times \mathbf{mm})$ when **side** = **Nag_RightSide** or **Nag_BothSides** and **order** = **Nag_ColMajor**;
- $\max(1, \mathbf{pdvr} \times \mathbf{n})$ when **side** = **Nag_RightSide** or **Nag_BothSides** and **order** = **Nag_RowMajor**;
- 1 when **side** = **Nag_LeftSide**.

If **order** = **Nag_ColMajor**, the (i, j)th element of the matrix is stored in $\mathbf{vr}[(j-1) \times \mathbf{pdvr} + i - 1]$ and if **order** = **Nag_RowMajor**, the (i, j)th element of the matrix is stored in $\mathbf{vr}[(i-1) \times \mathbf{pdvr} + j - 1]$.

On entry: if **initv** = **Nag_UserVec** and **side** = **Nag_RightSide** or **Nag_BothSides**, **vr** must contain starting vectors for inverse iteration for the right eigenvectors. Each starting vector must be stored in the same rows or columns as will be used to store the corresponding eigenvector (see below). If **initv** = **Nag_NoVec**, **vr** need not be set.

On exit: if **side** = **Nag_RightSide** or **Nag_BothSides**, **vr** contains the computed right eigenvectors (as specified by **select**). The eigenvectors are stored consecutively in the rows or columns of the array (depending on the **order** parameter), in the same order as their eigenvalues. Corresponding to each selected real eigenvalue is a real eigenvector, occupying one row or column. Corresponding to each selected complex eigenvalue is a complex eigenvector, occupying two rows or columns: the first row or column holds the real part and the second row or column holds the imaginary part.

vr is not referenced if **side** = **Nag_LeftSide**.

14: **pdvr** – Integer *Input*

On entry: the stride separating matrix row or column elements (depending on the value of **order**) in the array **vr**.

Constraints:

if **order** = **Nag_ColMajor**,
if **side** = **Nag_RightSide** or **Nag_BothSides**, **pdvr** $\geq \max(1, \mathbf{n})$;
if **side** = **Nag_LeftSide**, **pdvr** ≥ 1;

if **order** = **Nag_RowMajor**,
if **side** = **Nag_RightSide** or **Nag_BothSides**, **pdvr** $\geq \max(1, \mathbf{mm})$;
if **side** = **Nag_LeftSide**, **pdvr** ≥ 1.

15: **mm** – Integer *Input*

On entry: the number of columns in the arrays **vl** and/or **vr** if **order** = **Nag_ColMajor** or the number of rows in the arrays if **order** = **Nag_RowMajor**. The actual number of rows or columns required, $required\_rowcol$, is obtained by counting 1 for each selected real eigenvector and 2 for each selected complex eigenvector (see **select**); $0 \leq required\_rowcol \leq n$.

Constraint: **mm** $\geq required\_rowcol$.

16: **m** – Integer * *Output*

On exit: $required\_rowcol$, the number of rows or columns of **vl** and/or **vr** required to store the selected eigenvectors.

17: **ifaill**$[dim]$ – Integer *Output*

Note: the dimension, dim, of the array **ifaill** must be at least $\max(1, \mathbf{mm})$ when **side** = **Nag_LeftSide** or **Nag_BothSides** and at least 1 when **side** = **Nag_RightSide**.

On exit: if **side** = **Nag_LeftSide** or **Nag_BothSides**, then **ifaill**$[i]$ = 0 if the selected left eigenvector converged and **ifaill**$[i] = j \geq 0$ if the eigenvector stored in the ith row or column of **vl** (corresponding to the jth eigenvalue as held in $(\mathbf{wr}[j], \mathbf{wi}[j])$ failed to converge. If the ith and $(i+1)$th rows or columns of **vl** contain a selected complex eigenvector, then **ifaill**$[i]$ and **ifaill**$[i+1]$ are set to the same value.

ifaill is not referenced if **side** = **Nag_RightSide**.

18: **ifailr**$[dim]$ – Integer *Output*

Note: the dimension, dim, of the array **ifailr** must be at least $\max(1, \mathbf{mm})$ when **side** = **Nag_RightSide** or **Nag_BothSides** and at least 1 when **side** = **Nag_LeftSide**.

On exit: if **side** = **Nag_RightSide** or **Nag_BothSides**, then **ifailr**$[i]$ = 0 if the selected right eigenvector converged and **ifailr**$[i] = j \geq 0$ if the eigenvector stored in the ith row or column of **vr** (corresponding to the jth eigenvalue as held in $(\mathbf{wr}[j], \mathbf{wi}[j])$) failed to converge. If the ith and $(i+1)$th rows or columns of **vr** contain a selected complex eigenvector, then **ifailr**$[i]$ and **ifailr**$[i+1]$ are set to the same value.

ifailr is not referenced if **side** = **Nag_LeftSide**.

19: **fail** – NagError * *Output*

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **n** = ⟨*value*⟩.
Constraint: **n** ≥ 0.

On entry, **mm** = ⟨*value*⟩.
Constraint: **mm** $\geq$ *required_rowcol*, where *required_rowcol* is obtained by counting 1 for each selected real eigenvector and 2 for each selected complex eigenvector.

On entry, **pdh** = ⟨*value*⟩.
Constraint: **pdh** > 0.

On entry, **pdvl** = ⟨*value*⟩.
Constraint: **pdvl** > 0.

On entry, **pdvr** = ⟨*value*⟩.
Constraint: **pdvr** > 0.

NE_INT_2

On entry, **pdh** = ⟨*value*⟩, **n** = ⟨*value*⟩.
Constraint: **pdh** $\geq \max(1, \mathbf{n})$.

NE_ENUM_INT_2

On entry, **side** = ⟨*value*⟩, **n** = ⟨*value*⟩, **pdvl** = ⟨*value*⟩.
Constraint: if **side** = **Nag_LeftSide** or **Nag_BothSides**, **pdvl** $\geq \max(1, \mathbf{n})$;
if **side** = **Nag_RightSide**, **pdvl** ≥ 1.

On entry, **side** = ⟨*value*⟩, **n** = ⟨*value*⟩, **pdvr** = ⟨*value*⟩.
Constraint: if **side** = **Nag_RightSide** or **Nag_BothSides**, **pdvr** $\geq \max(1, \mathbf{n})$;
if **side** = **Nag_LeftSide**, **pdvr** ≥ 1.

On entry, **side** = ⟨*value*⟩, **mm** = ⟨*value*⟩, **pdvl** = ⟨*value*⟩.
Constraint: if **side** = **Nag_LeftSide** or **Nag_BothSides**, **pdvl** $\geq \max(1, \mathbf{mm})$;
if **side** = **Nag_RightSide**, **pdvl** ≥ 1.

On entry, **side** = ⟨*value*⟩, **mm** = ⟨*value*⟩, **pdvr** = ⟨*value*⟩.
Constraint: if **side** = **Nag_RightSide** or **Nag_BothSides**, **pdvr** $\geq \max(1, \mathbf{mm})$;
if **side** = **Nag_LeftSide**, **pdvr** ≥ 1.

NE_CONVERGENCE

⟨*value*⟩ eigenvectors (as indicated by arguments **ifaill** and/or **ifailr**) failed to converge. The corresponding columns of **vl** and/or **vr** contain no useful information.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter ⟨*value*⟩ had an illegal value.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

7 Accuracy

Each computed right eigenvector x_i is the exact eigenvector of a nearby matrix $A + E_i$, such that $\|E_i\| = O(\epsilon)\|A\|$. Hence the residual is small:

$$\|Ax_i - \lambda_i x_i\| = O(\epsilon)\|A\|.$$

However eigenvectors corresponding to close or coincident eigenvalues may not accurately span the relevant subspaces.

Similar remarks apply to computed left eigenvectors.

8 Further Comments

The complex analogue of this function is nag_zhsein (f08pxc).

9 Example

See Section 9 of the document for nag_dormhr (f08ngc).