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 \( \lambda \), are defined by:

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

Note that even though \( H \) is real, \( \lambda, x \) and \( y \) may be complex. If \( x \) is an eigenvector corresponding to a complex eigenvalue \( \lambda \), 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, \( \max(|\Re(x_i)| + |\Im(x_i)|) = 1 \).

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


5 Parameters

1: \( \text{order} \) – Nag_OrderType

\( \text{Input} \)

\( \text{On entry:} \) 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.

\( \text{Constraint:} \) order = Nag_RowMajor or Nag_ColMajor.

2: \( \text{side} \) – Nag_SideType

\( \text{Input} \)

\( \text{On entry:} \) indicates whether left and/or right eigenvectors are to be computed as follows:

\( \text{if} \ \text{side} = \text{Nag_RightSide}, \ \text{then only right eigenvectors are computed;} \)

\( \text{if} \ \text{side} = \text{Nag_LeftSide}, \ \text{then only left eigenvectors are computed;} \)
if \( \text{side} = \text{Nag\_BothSides} \), then both left and right eigenvectors are computed.

**Constraint:** \( \text{side} = \text{Nag\_RightSide}, \text{Nag\_LeftSide} \) or \( \text{Nag\_BothSides} \).

3: \( \text{eig\_source} \) – \text{Nag\_EigValsSourceType}

*Input*

*On entry:* indicates whether the eigenvalues of \( H \) (stored in \( \text{wr} \) and \( \text{wi} \)) were found using \text{nag\_dhseqr} (f08pec) as follows:

- if \( \text{eig\_source} = \text{Nag\_HSEQRSource} \), then the eigenvalues of \( H \) were found using \text{nag\_dhseqr} (f08pec); thus if \( H \) has any zero sub-diagonal elements (and so is block triangular), then the \( j \)th eigenvalue can be assumed to be an eigenvalue of the block containing the \( j \)th row/column. This property allows the function to perform inverse iteration on just one diagonal block;
- if \( \text{eig\_source} = \text{Nag\_NotKnown} \), then no such assumption is made and the function performs inverse iteration using the whole matrix.

**Constraint:** \( \text{eig\_source} = \text{Nag\_HSEQRSource} \) or \( \text{Nag\_NotKnown} \).

4: \( \text{initv} \) – \text{Nag\_InitVeenumtype}

*Input*

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

- if \( \text{initv} = \text{Nag\_NoVec} \), no initial estimates are supplied;
- if \( \text{initv} = \text{Nag\_UserVec} \), initial estimates are supplied in \( \text{vl} \) and/or \( \text{vr} \).

**Constraint:** \( \text{initv} = \text{Nag\_NoVec} \) or \( \text{Nag\_UserVec} \).

5: \( \text{select}[\text{dim}] \) – \text{Boolean}

*Input/Output*

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

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

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

6: \( n \) – \text{Integer}

*Input*

*On entry:* \( n \), the order of the matrix \( H \).

**Constraint:** \( n \geq 0 \).

7: \( h[\text{dim}] \) – \text{const double}

*Input*

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

If \( \text{order} = \text{Nag\_ColMajor} \), the \( (i,j) \)th element of the matrix \( H \) is stored in \( h[(j-1) \times \text{pdh} + i - 1] \) and if \( \text{order} = \text{Nag\_RowMajor} \), the \( (i,j) \)th element of the matrix \( H \) is stored in \( h[(i-1) \times \text{pdh} + j - 1] \).

*On entry:* the \( n \) by \( n \) upper Hessenberg matrix \( H \).

8: \( \text{pdh} \) – \text{Integer}

*Input*

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

**Constraint:** \( \text{pdh} \geq \max(1, n) \).
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.

vl[\text{dim}] – double  \hspace{1cm} \text{Input/Output}

Note: the dimension, \text{dim}, of the array vl must be at least
\[
\max(1, \text{pdvl} \times \text{mm}) \quad \text{when} \quad \text{side} = \text{Nag\_LeftSide} \quad \text{or} \quad \text{Nag\_BothSides} \quad \text{and} \quad \text{order} = \text{Nag\_ColMajor};
\]
\[
\max(1, \text{pdvl} \times \text{n}) \quad \text{when} \quad \text{side} = \text{Nag\_LeftSide} \quad \text{or} \quad \text{Nag\_BothSides} \quad \text{and} \quad \text{order} = \text{Nag\_RowMajor};
\]
\[\text{1 when} \quad \text{side} = \text{Nag\_RightSide}.\]

If \text{order} = \text{Nag\_ColMajor}, the \((i, j)\)th element of the matrix is stored in \(\text{vl}[(j - 1) \times \text{pdvl} + i - 1]\) and if \text{order} = \text{Nag\_RowMajor}, the \((i, j)\)th element of the matrix is stored in \(\text{vl}[(i - 1) \times \text{pdvl} + j - 1]\).

On entry: if \text{initv} = \text{Nag\_UserVec} and \text{side} = \text{Nag\_LeftSide} or \text{Nag\_BothSides}, \text{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 \text{initv} = \text{Nag\_NoVec}, \text{vl} need not be set.

On exit: if \text{side} = \text{Nag\_LeftSide} or \text{Nag\_BothSides}, \text{vl} contains the computed left eigenvectors (as specified by \text{select}). The eigenvectors are stored consecutively in the rows or columns of the array (depending on the value of \text{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.

\text{vl} is not referenced if \text{side} = \text{Nag\_RightSide}.

pdvl – Integer  \hspace{1cm} \text{Input}

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

Constraints:
\[
\text{if} \quad \text{order} = \text{Nag\_ColMajor},
\]
\[
\text{if} \quad \text{side} = \text{Nag\_LeftSide} \text{ or } \text{Nag\_BothSides}, \quad \text{pdvl} \geq \max(1, \text{n});
\]
\[
\text{if} \quad \text{side} = \text{Nag\_RightSide}, \quad \text{pdvl} \geq 1;
\]
\[
\text{if} \quad \text{order} = \text{Nag\_RowMajor},
\]
\[
\text{if} \quad \text{side} = \text{Nag\_LeftSide} \text{ or } \text{Nag\_BothSides}, \quad \text{pdvl} \geq \max(1, \text{mm});
\]
\[
\text{if} \quad \text{side} = \text{Nag\_RightSide}, \quad \text{pdvl} \geq 1.
\]

vr[\text{dim}] – double  \hspace{1cm} \text{Input/Output}

Note: the dimension, \text{dim}, of the array vr must be at least
\[
\max(1, \text{pdvr} \times \text{mm}) \quad \text{when} \quad \text{side} = \text{Nag\_RightSide} \quad \text{or} \quad \text{Nag\_BothSides} \quad \text{and} \quad \text{order} = \text{Nag\_ColMajor};
\]
\[
\max(1, \text{pdvr} \times \text{n}) \quad \text{when} \quad \text{side} = \text{Nag\_RightSide} \quad \text{or} \quad \text{Nag\_BothSides} \quad \text{and} \quad \text{order} = \text{Nag\_RowMajor};
\]
\[\text{1 when} \quad \text{side} = \text{Nag\_LeftSide}.
\]

If \text{order} = \text{Nag\_ColMajor}, the \((i, j)\)th element of the matrix is stored in \(\text{vr}[(j - 1) \times \text{pdvr} + i - 1]\) and if \text{order} = \text{Nag\_RowMajor}, the \((i, j)\)th element of the matrix is stored in \(\text{vr}[(i - 1) \times \text{pdvr} + j - 1]\).
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

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 ≥ max(1, n);
    if side = Nag_LeftSide, pdvr ≥ 1;

if order = Nag_RowMajor,
    if side = Nag_RightSide or Nag_BothSides, pdvr ≥ max(1, mm);
    if side = Nag_LeftSide, pdvr ≥ 1.

15: mm – Integer

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 ≤ required_rowcol ≤ n.

Constraint: mm ≥ required_rowcol.

16: m – Integer

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

Note: the dimension, dim, of the array ifaill must be at least max(1, 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 ≥ 0 if the eigenvector stored in the ith row or column of vl
(corresponding to the jth eigenvalue as held in (wr[j], 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

Note: the dimension, dim, of the array ifailr must be at least max(1, 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 ≥ 0 if the eigenvector stored in the ith row or column of vr
(corresponding to the jth eigenvalue as held in (wr[j], 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 ≥ 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 ≥ max(1, n).

NE_ENUM_INT_2
On entry, side = ⟨value⟩, n = ⟨value⟩, pdvl = ⟨value⟩.
Constraint: if side = Nag_LeftSide or Nag_BothSides, pdvl ≥ max(1, n);
if side = Nag_RightSide, pdvl ≥ 1.

On entry, side = ⟨value⟩, n = ⟨value⟩, pdvr = ⟨value⟩.
Constraint: if side = Nag_RightSide or Nag_BothSides, pdvr ≥ max(1, n);
if side = Nag_LeftSide, pdvr ≥ 1.

On entry, side = ⟨value⟩, mm = ⟨value⟩, pdvl = ⟨value⟩.
Constraint: if side = Nag_LeftSide or Nag_BothSides, pdvl ≥ max(1, mm);
if side = Nag_RightSide, pdvl ≥ 1.

On entry, side = ⟨value⟩, mm = ⟨value⟩, pdvr = ⟨value⟩.
Constraint: if side = Nag_RightSide or Nag_BothSides, pdvr ≥ max(1, mm);
if side = Nag_LeftSide, pdvr ≥ 1.

NE_CONVERGENCE
⟨value⟩ eigenvectors (as indicated by arguments ifaill and/or ifailr) failed to converge. The
respective 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.
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).