NAG 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

```c
#include <nag.h>
#include <nagf08.h>

void nag_dhsein (Nag_OrderType order, Nag_SideType side,
        Nag_EigValsSourceType eig_source, Nag_InitVeenumtype initv,
        Nag_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:

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

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, \( \max(|\text{Re}(x_i)| + |\text{Im}(x_i)|) = 1 \).

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

4 References


5 Arguments

1:  order – Nag_OrderType

   Input

   On entry: the order argument 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 3.2.1.3 in the Essential Introduction for a more detailed explanation of the use of this argument.

   Constraint: order = Nag_RowMajor or Nag_ColMajor.
2: side – Nag.SideType  
   \textit{Input}  
   \textit{On entry:} indicates whether left and/or right eigenvectors are to be computed. 
   \text{side} = \text{Nag\_RightSide} \quad \text{Only right eigenvectors are computed.} 
   \text{side} = \text{Nag\_LeftSide} \quad \text{Only left eigenvectors are computed.} 
   \text{side} = \text{Nag\_BothSides} \quad \text{Both left and right eigenvectors are computed.} 
   \textit{Constraint: side} = \text{Nag\_RightSide, Nag\_LeftSide or Nag\_BothSides.} 

3: eig_source – Nag.EigValsSourceType  
   \textit{Input}  
   \textit{On entry:} indicates whether the eigenvalues of \( H \) (stored in \( \text{wr} \) and \( \text{wi} \)) were found using \text{nag\_dhseqr (f08pec)}. 
   \text{eig\_source} = \text{Nag\_HSEQRSource} \quad \text{The eigenvalues of \( H \) were found using nag\_dhseqr (f08pec); thus if \( H \) has any zero subdiagonal 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.} 
   \text{eig\_source} = \text{Nag\_NotKnown} \quad \text{No such assumption is made and the function performs inverse iteration using the whole matrix.} 
   \textit{Constraint: eig\_source} = \text{Nag\_HSEQRSource or Nag\_NotKnown.} 

4: initv – Nag.InitVeenumtype  
   \textit{Input}  
   \textit{On entry:} indicates whether you are supplying initial estimates for the selected eigenvectors. 
   \text{initv} = \text{Nag\_NoVec} \quad \text{No initial estimates are supplied.} 
   \text{initv} = \text{Nag\_UserVec} \quad \text{Initial estimates are supplied in \( \text{vl} \) and/or \( \text{vr} \).} 
   \textit{Constraint: initv} = \text{Nag\_NoVec or Nag\_UserVec.} 

5: select\([\text{dim}]\) – Nag.Boolean  
   \textit{Input/Output}  
   \textit{Note:} the dimension, \( \text{dim} \), of the array \( \text{select} \) must be at least \( \text{max}(1, n) \). 
   \textit{On entry:} specifies which eigenvectors are to be computed. To obtain the real eigenvector corresponding to the real eigenvalue \( \text{wr}[j - 1] \), \( \text{select}[j - 1] \) must be set \text{Nag\_TRUE}. To select the complex eigenvector corresponding to the complex eigenvalue \( (\text{wr}[j - 1], \text{wi}[j - 1]) \) with complex conjugate \((\text{wr}[j], \text{wi}[j])\), \( \text{select}[j - 1] \) and/or \( \text{select}[j] \) must be set \text{Nag\_TRUE}; the eigenvector corresponding to the \text{first} eigenvalue in the pair is computed. 
   \textit{On exit:} if a complex eigenvector was selected as specified above, then \( \text{select}[j - 1] \) is set to \text{Nag\_TRUE} and \( \text{select}[j] \) to \text{Nag\_FALSE}. 

6: n – Integer  
   \textit{Input}  
   \textit{On entry:} \( n \), the order of the matrix \( H \). 
   \textit{Constraint:} \( n \geq 0 \). 

7: h\([\text{dim}]\) – const double  
   \textit{Input}  
   \textit{Note:} the dimension, \( \text{dim} \), of the array \( h \) must be at least \( \text{max}(1, \text{pdh} \times n) \).
The \((i, j)\)th element of the matrix \(H\) is stored in

\[
\begin{align*}
h[(j - 1) \times \text{pdh} + i - 1] & \quad \text{when order} = \text{Nag\_ColMajor;} \\
h[(i - 1) \times \text{pdh} + j - 1] & \quad \text{when order} = \text{Nag\_RowMajor.}
\end{align*}
\]

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

8: \(\text{pdh} - \text{Integer}\) \hspace{1cm} \text{Input}

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

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

9: \(\text{wr}[\text{dim}] - \text{double}\) \hspace{1cm} \text{Input/Output}

10: \(\text{wi}[\text{dim}] - \text{const double}\) \hspace{1cm} \text{Input}

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

*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 \(\text{eig\_source} = \text{Nag\_HSEQRSource}\), the arrays must be exactly as returned by \(\text{nag\_dheqdr (f08pec)}\).

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

11: \(\text{vl}[\text{dim}] - \text{double}\) \hspace{1cm} \text{Input/Output}

*Note:* the dimension, \(\text{dim}\), of the array \(\text{vl}\) must be at least

\[
\max(1, \text{pdvl} \times \text{mm}) \quad \text{when} \quad \text{side} = \text{Nag\_LeftSide or Nag\_BothSides and} \\
\text{order} = \text{Nag\_ColMajor;} \\
\max(1, n \times \text{pdvl}) \quad \text{when} \quad \text{side} = \text{Nag\_LeftSide or Nag\_BothSides and} \\
\text{order} = \text{Nag\_RowMajor;} \\
1 \quad \text{when} \quad \text{side} = \text{Nag\_RightSide.}
\]

The \((i, j)\)th element of the matrix is stored in

\[
\begin{align*}
\text{vl}[(j - 1) \times \text{pdvl} + i - 1] & \quad \text{when order} = \text{Nag\_ColMajor;} \\
\text{vl}[(i - 1) \times \text{pdvl} + j - 1] & \quad \text{when order} = \text{Nag\_RowMajor.}
\end{align*}
\]

*On entry:* if \(\text{initv} = \text{Nag\_UserVec}\) and \(\text{side} = \text{Nag\_LeftSide or Nag\_BothSides}\), \(\text{vl}\) 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 \(\text{side} = \text{Nag\_LeftSide or Nag\_BothSides}\), \(\text{vl}\) 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.

If \(\text{side} = \text{Nag\_RightSide}\), \(\text{vl}\) is not referenced.

12: \(\text{pdvl} - \text{Integer}\) \hspace{1cm} \text{Input}

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

*Constraints:*

if \(\text{order} = \text{Nag\_ColMajor},\)

if \(\text{side} = \text{Nag\_LeftSide or Nag\_BothSides}\), \(\text{pdvl} \geq n;\)

if \(\text{side} = \text{Nag\_RightSide}\), \(\text{pdvl} \geq 1;\)
if order = Nag_RowMajor,
    if side = Nag_LeftSide or Nag_BothSides, pdvl ≥ max(1, mm);
    if side = Nag_RightSide, pdvl ≥ 1..

13: vr[dim] – double

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

    The \((i, j)\)th element of the matrix is stored in
    \[
    \begin{align*}
    vr[(j - 1) \times pdvr + i - 1] & \quad \text{when order = Nag_ColMajor;} \\
    vr[(i - 1) \times pdvr + j - 1] & \quad \text{when order = Nag_RowMajor.}
    \end{align*}
    \]

    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 argument), 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.

    If side = Nag_LeftSide, vr is not referenced.

14: pdvr – Integer

    On entry: the stride separating 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 ≥ 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.
ifail[\textit{dim}] – Integer

\textbf{Note:} the dimension, \textit{dim}, of the array \textit{ifail} must be at least
\[
\max(1, \text{mm}) \quad \text{when} \; \textit{side} = \text{Nag\_LeftSide or Nag\_BothSides};
\]
\[
1 \quad \text{when} \; \textit{side} = \text{Nag\_RightSide}.
\]

\textit{On exit:} if \textit{side} = \text{Nag\_LeftSide} or \text{Nag\_BothSides}, then \textit{ifail} \[i-1\] = 0 if the selected left eigenvector converged and \textit{ifail} \[i-1\] = j \geq 0 if the eigenvector stored in the \textit{i}th row or column of \textit{vl} (corresponding to the \textit{j}th eigenvalue as held in \textit{(wr}\[j-1\],\textit{wi}\[j-1\]) failed to converge. If the \textit{i}th and \((i+1)\)th rows or columns of \textit{vl} contain a selected complex eigenvector, then \textit{ifail} \[i-1\] and \textit{ifail} \[i\] are set to the same value.

If \textit{side} = \text{Nag\_RightSide}, \textit{ifail} is not referenced.

ifailr[\textit{dim}] – Integer

\textbf{Note:} the dimension, \textit{dim}, of the array \textit{ifailr} must be at least
\[
\max(1, \text{mm}) \quad \text{when} \; \textit{side} = \text{Nag\_RightSide or Nag\_BothSides};
\]
\[
1 \quad \text{when} \; \textit{side} = \text{Nag\_LeftSide}.
\]

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

If \textit{side} = \text{Nag\_LeftSide}, \textit{ifailr} is not referenced.

\textbf{fail} – NagError *

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

\textbf{NE\_ALLOC\_FAIL}

Dynamic memory allocation failed.

See Section 3.2.1.2 in the Essential Introduction for further information.

\textbf{NE\_BAD\_PARAM}

On entry, argument \textit{\textlangle value\textrangle} had an illegal value.

\textbf{NE\_CONVERGENCE}

\textit{\textlangle value\textrangle} eigenvectors (as indicated by arguments \textit{ifail} and/or \textit{ifailr}) failed to converge. The corresponding columns of \textit{vl} and/or \textit{vr} contain no useful information.

\textbf{NE\_ENUM\_INT\_2}

On entry, \textit{side} = \textit{\textlangle value\textrangle}, \textit{pdvl} = \textit{\textlangle value\textrangle}, \textit{mm} = \textit{\textlangle value\textrangle}.

Constraint: if \textit{side} = \text{Nag\_LeftSide or Nag\_BothSides}, \textit{pdvl} \geq \max(1, \text{mm});

if \textit{side} = \text{Nag\_RightSide}, \textit{pdvl} \geq 1.

On entry, \textit{side} = \textit{\textlangle value\textrangle}, \textit{pdvl} = \textit{\textlangle value\textrangle} and \textit{n} = \textit{\textlangle value\textrangle}.

Constraint: if \textit{side} = \text{Nag\_LeftSide or Nag\_BothSides}, \textit{pdvl} \geq \textit{n};

if \textit{side} = \text{Nag\_RightSide}, \textit{pdvl} \geq 1.

On entry, \textit{side} = \textit{\textlangle value\textrangle}, \textit{pdvr} = \textit{\textlangle value\textrangle}, \textit{mm} = \textit{\textlangle value\textrangle}.

Constraint: if \textit{side} = \text{Nag\_RightSide or Nag\_BothSides}, \textit{pdvr} \geq \max(1, \text{mm});

if \textit{side} = \text{Nag\_LeftSide}, \textit{pdvr} \geq 1.
On entry, \( \text{side} = (\text{value}) \), \( \text{pdvr} = (\text{value}) \) and \( n = (\text{value}) \).
Constraint: if \( \text{side} = \text{Nag\_RightSide} \) or \( \text{Nag\_BothSides} \), \( \text{pdvr} \geq n \); if \( \text{side} = \text{Nag\_LeftSide} \), \( \text{pdvr} \geq 1 \).

**NE\_INT**

On entry, \( \text{mm} = (\text{value}) \).
Constraint: \( \text{mm} \geq \text{required\_rowcol} \), where \( \text{required\_rowcol} \) is obtained by counting 1 for each selected real eigenvector and 2 for each selected complex eigenvector.

On entry, \( n = (\text{value}) \).
Constraint: \( n \geq 0 \).

On entry, \( \text{pdh} = (\text{value}) \).
Constraint: \( \text{pdh} > 0 \).

On entry, \( \text{pdvl} = (\text{value}) \).
Constraint: \( \text{pdvl} > 0 \).

On entry, \( \text{pdvr} = (\text{value}) \).
Constraint: \( \text{pdvr} > 0 \).

**NE\_INT\_2**

On entry, \( \text{pdh} = (\text{value}) \) and \( n = (\text{value}) \).
Constraint: \( \text{pdh} \geq \max(1, n) \).

**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 contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

**NE\_NO\_LICENCE**

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

### 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 Parallelism and Performance

\texttt{nag\_dhsein} (\texttt{f08pkc}) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

\texttt{nag\_dhsein} (\texttt{f08pkc}) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the \texttt{X06 Chapter Introduction} for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users’ Note for your implementation for any additional implementation-specific information.
9 Further Comments
The complex analogue of this function is nag_zhsein (f08pxc).

10 Example
See Section 10 in nag_dormhr (f08ngc).