NAG Library Function Document

nag_zheevx (f08fpc)

1 Purpose

nag_zheevx (f08fpc) computes selected eigenvalues and, optionally, eigenvectors of a complex n by n Hermitian matrix A. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

2 Specification

#include <nag.h>
#include <nagf08.h>
void nag_zheevx (Nag_OrderType order, Nag_JobType job, Nag_RangeType range, Nag_UploType uplo, Integer n, Complex a[], Integer pda, double vl, double vu, Integer il, Integer iu, double abstol, Integer *m, double w[], Complex z[], Integer pdz, Integer jfail[], NagError *fail)

3 Description

The Hermitian matrix A is first reduced to real tridiagonal form, using unitary similarity transformations. The required eigenvalues and eigenvectors are then computed from the tridiagonal matrix; the method used depends upon whether all, or selected, eigenvalues and eigenvectors are required.

4 References


5 Arguments

1: order – Nag_OrderType

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: job – Nag_JobType

On entry: indicates whether eigenvectors are computed.

job = Nag_EigVals

Only eigenvalues are computed.

job = Nag_DoBoth

Eigenvalues and eigenvectors are computed.

Constraint: job = Nag_EigVals or Nag_DoBoth.
3:  range – Nag_RangeType

  On entry: if range = Nag_AllValues, all eigenvalues will be found.
  If range = Nag_Interval, all eigenvalues in the half-open interval \([vl, vu]\) will be found.
  If range = Nag_Indices, the \(i_l\)th to \(i_u\)th eigenvalues will be found.

  Constraint: range = Nag_AllValues, Nag_Interval or Nag_Indices.

4:  uplo – Nag_UploType

  On entry: if uplo = Nag_Upper, the upper triangular part of \(A\) is stored.
  If uplo = Nag_Lower, the lower triangular part of \(A\) is stored.

  Constraint: uplo = Nag_Upper or Nag_Lower.

5:  n – Integer

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

  Constraint: \(n \geq 0\).

6:  a[\text{dim}] – Complex

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

  On entry: the \(n\) by \(n\) Hermitian matrix \(A\).

  If order = Nag_ColMajor, \(A_{ij}\) is stored in \(a[(j-1) \times \text{pda} + i - 1]\).

  If order = Nag_RowMajor, \(A_{ij}\) is stored in \(a[(i-1) \times \text{pda} + j - 1]\).

  If uplo = Nag_Upper, the upper triangular part of \(A\) must be stored and the elements of the array below the diagonal are not referenced.

  If uplo = Nag_Lower, the lower triangular part of \(A\) must be stored and the elements of the array above the diagonal are not referenced.

  On exit: the lower triangle (if uplo = Nag_Lower) or the upper triangle (if uplo = Nag_Upper) of \(a\), including the diagonal, is overwritten.

7:  pda – Integer

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

  Constraint: \(pda \geq \max(1, n)\).

8:  vl – double

9:  vu – double

  On entry: if range = Nag_Interval, the lower and upper bounds of the interval to be searched for eigenvalues.

  If range = Nag_AllValues or Nag_Indices, \(vl\) and \(vu\) are not referenced.

  Constraint: if range = Nag_Interval, \(vl < vu\).

10:  il – Integer

11:  iu – Integer

  On entry: if range = Nag_Indices, the indices (in ascending order) of the smallest and largest eigenvalues to be returned.

  If range = Nag_AllValues or Nag_Interval, \(il\) and \(iu\) are not referenced.
Constraints:

if \( \text{range} = \text{Nag}_\text{Indices} \) and \( n = 0, il = 1 \) and \( iu = 0 \);
if \( \text{range} = \text{Nag}_\text{Indices} \) and \( n > 0, 1 \leq il \leq iu \leq n \).

12: \( \text{abstol} \) – double  

\text{Input}

On entry: the absolute error tolerance for the eigenvalues. An approximate eigenvalue is accepted as converged when it is determined to lie in an interval \([a, b]\) of width less than or equal to

\[
\text{abstol} + \epsilon \max(|a|, |b|),
\]

where \( \epsilon \) is the \textit{machine precision}. If \( \text{abstol} \) is less than or equal to zero, then \( \epsilon \|T\|_1 \) will be used in its place, where \( T \) is the tridiagonal matrix obtained by reducing \( A \) to tridiagonal form. Eigenvalues will be computed most accurately when \( \text{abstol} \) is set to twice the underflow threshold \( 2 \times \text{Nag}_\text{real}_\text{safe}_\text{small}_\text{number} \), not zero. If this function returns with \( \text{fail} = \text{NE}_\text{CONVERGENCE} \), indicating that some eigenvectors did not converge, try setting \( \text{abstol} \) to \( 2 \times \text{Nag}_\text{real}_\text{safe}_\text{small}_\text{number} \). See Demmel and Kahan (1990).

13: \( \text{m} \) – Integer *

\text{Output}

On exit: the total number of eigenvalues found. \( 0 \leq \text{m} \leq n \).

If \( \text{range} = \text{Nag}_\text{AllValues} \), \( \text{m} = n \).
If \( \text{range} = \text{Nag}_\text{Indices} \), \( \text{m} = iu - il + 1 \).

14: \( \text{w}[\text{dim}] \) – double  

\text{Output}

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

On exit: the first \( \text{m} \) elements contain the selected eigenvalues in ascending order.

15: \( \text{z}[\text{dim}] \) – Complex  

\text{Output}

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

\[
\max(1, \text{pdz} \times n) \text{ when } \text{job} = \text{Nag}_\text{DoBoth};
\]
1 otherwise.

The \( i \)th element of the \( j \)th vector \( Z \) is stored in

\[
\text{z}[(j-1) \times \text{pdz} + i - 1] \text{ when } \text{order} = \text{Nag}_\text{ColMajor};
\]
\[
\text{z}[(i-1) \times \text{pdz} + j - 1] \text{ when } \text{order} = \text{Nag}_\text{RowMajor}.
\]

On exit: if \( \text{job} = \text{Nag}_\text{DoBoth} \), then

if \( \text{fail} = \text{NE}_\text{NOERROR} \), the first \( \text{m} \) columns of \( Z \) contain the orthonormal eigenvectors of the matrix \( A \) corresponding to the selected eigenvalues, with the \( i \)th column of \( Z \) holding the eigenvector associated with \( \text{w}[(i-1)] \);

if an eigenvector fails to converge (\( \text{fail} = \text{NE}_\text{CONVERGENCE} \)), then that column of \( Z \) contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in \( \text{jfail} \).

If \( \text{job} = \text{Nag}_\text{EigVals} \), \( \text{z} \) is not referenced.

16: \( \text{pdz} \) – Integer  

\text{Input}

On entry: the stride used in the array \( \text{z} \).

Constraints:

if \( \text{job} = \text{Nag}_\text{DoBoth} \), \( \text{pdz} \geq \max(1, n) \);
otherwise \( \text{pdz} \geq 1 \).
17: $\text{jfail}[\dim]$ – Integer

**Output**

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

*On exit:* if $\text{job} = \text{Nag}_{\text{DoBoth}}$, then

- if $\text{fail.code} = \text{NE\_NOERROR}$, the first $m$ elements of $\text{jfail}$ are zero;
- if $\text{fail.code} = \text{NE\_CONVERGENCE}$, $\text{jfail}$ contains the indices of the eigenvectors that failed to converge.

If $\text{job} = \text{Nag\_EigVals}$, $\text{jfail}$ is not referenced.

18: $\text{fail}$ – NagError *

**Input/Output**

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

6 Error Indicators and Warnings

**NE\_ALLOC\_FAIL**

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

**NE\_BAD\_PARAM**

On entry, argument `<value>` had an illegal value.

**NE\_CONVERGENCE**

The algorithm failed to converge; `<value>` eigenvectors did not converge. Their indices are stored in array $\text{jfail}$.

**NE\_ENUM\_INT\_2**

On entry, $\text{job} = \langle\text{value}\rangle$, $\text{pdz} = \langle\text{value}\rangle$ and $\text{n} = \langle\text{value}\rangle$.

Constraint: if $\text{job} = \text{Nag\_DoBoth}$, $\text{pdz} \geq \max(1, n)$; otherwise $\text{pdz} \geq 1$.

**NE\_ENUM\_INT\_3**

On entry, $\text{range} = \langle\text{value}\rangle$, $\text{il} = \langle\text{value}\rangle$, $\text{iu} = \langle\text{value}\rangle$ and $\text{n} = \langle\text{value}\rangle$.

Constraint: if $\text{range} = \text{Nag\_Indices}$ and $\text{n} = 0$, $\text{il} = 1$ and $\text{iu} = 0$; if $\text{range} = \text{Nag\_Indices}$ and $\text{n} > 0$, $1 \leq \text{il} \leq \text{iu} \leq \text{n}$.

**NE\_ENUM\_REAL\_2**

On entry, $\text{range} = \langle\text{value}\rangle$, $\text{vl} = \langle\text{value}\rangle$ and $\text{vu} = \langle\text{value}\rangle$.

Constraint: if $\text{range} = \text{Nag\_Interval}$, $\text{vl} < \text{vu}$.

**NE\_INT**

On entry, $\text{n} = \langle\text{value}\rangle$.

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

On entry, $\text{pda} = \langle\text{value}\rangle$.

Constraint: $\text{pda} > 0$.

On entry, $\text{pdz} = \langle\text{value}\rangle$.

Constraint: $\text{pdz} > 0$.

**NE\_INT\_2**

On entry, $\text{pda} = \langle\text{value}\rangle$ and $\text{n} = \langle\text{value}\rangle$.

Constraint: $\text{pda} \geq \max(1, n)$. 
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.

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.

The computed eigenvalues and eigenvectors are exact for a nearby matrix \(A + E\), where
\[
\|E\|_2 = O(\epsilon)\|A\|_2,
\]
and \(\epsilon\) is the machine precision. See Section 4.7 of Anderson et al. (1999) for further details.

nag_zheevx (f08fpc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_zheevx (f08fpc) 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.

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

The real analogue of this function is nag_dsyevx (f08fbc).

This example finds the eigenvalues in the half-open interval \((-2, 2]\), and the corresponding eigenvectors, of the Hermitian matrix
\[
A = \begin{pmatrix}
1 & 2 - i & 3 - i & 4 - i \\
2 + i & 2 & 3 - 2i & 4 - 2i \\
3 + i & 3 + 2i & 3 & 4 - 3i \\
4 + i & 4 + 2i & 4 + 3i & 4
\end{pmatrix}.
\]

/* nag_zheevx (f08fpc) Example Program. */
* Copyright 2014 Numerical Algorithms Group.
* Mark 23, 2011.
*/

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <naga02.h>
#include <nagf08.h>
#include <nagx04.h>

int main(void)
{
/* Scalars */
double abstol, vl, vu;
Integer exit_status = 0, i, il = 0, iu = 0, j, m, n, pda, pdz;
/* Arrays */
Complex *a = 0, *z = 0;
double *w = 0;
Integer *index = 0;
/* Nag Types */
Nag_OrderType order;
NagError fail, fail_print;
#ifdef NAG_COLUMN_MAJOR
#define A(I, J) a[(J - 1) * pda + I - 1]
#define Z(I, J) z[(J - 1) * pdz + I - 1]
order = Nag_ColMajor;
#else
#define A(I, J) a[(I - 1) * pda + J - 1]
#define Z(I, J) z[(I - 1) * pdz + J - 1]
order = Nag_RowMajor;
#endif
INIT_FAIL(fail);
printf("nag_zheevx (f08fpc) Example Program Results\n\n");
/* Skip heading in data file */
#ifdef _WIN32
scanf_s("%*[\n"]);
#else
scanf("%*[\n"]);
#endif
#ifdef _WIN32
scanf("%"NAG_IFMT"%*[\n"]", &n);
#else
scanf("%"NAG_IFMT"%*[\n"]", &n);
#endif
m = n;
#ifdef NAG_COLUMN_MAJOR
pda = n;
pdz = n;
#else
pda = n;
pdz = m;
#endif
/* Allocate memory */
if (!((a = NAG_ALLOC(n*n, Complex)) ||
    (z = NAG_ALLOC(n*m, Complex)) ||
    (w = NAG_ALLOC(n, double)) ||
    (index = NAG_ALLOC(n, Integer)) ))
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}
pda = n;
#ifdef NAG_COLUMN_MAJOR
pdz = n;
#else
pdz = m;
#endif
/* Read the lower and upper bounds of the interval to be searched, *
* and read the upper triangular part of the matrix A from data file. */
```c
/*
 * #ifdef _WIN32
 * scanf_s("%lf%lf\n", &vl, &vu);
 * #else
 * scanf("%lf%lf\n", &vl, &vu);
 * #endif
 * for (i = 1; i <= n; ++i)
 *     for (j = i; j <= n; ++j)
 *     #ifdef _WIN32
 *         scanf_s(" (%lf , %lf )", &A(i, j).re, &A(i, j).im);
 *     #else
 *         scanf(" (%lf , %lf )", &A(i, j).re, &A(i, j).im);
 *     #endif
 * #ifdef _WIN32
 *     scanf_s("%\n");
 * #else
 *     scanf("%\n");
 * #endif
 * /* Set the absolute error tolerance for eigenvalues.
 * With abstol set to zero, the default value is used instead.
 */
 * abstol = 0.0;
 * /* nag_zheevx (f08fpc).
 * Solve the Hermitian eigenvalue problem.
 */
 * nag_zheevx(order, Nag_DoBoth, Nag_Interval, Nag_Upper, n, a, pda, vl, vu, il, iu, abstol, &m, w, z, pdz, index, &fail);
 * if (fail.code != NE_NOERROR && fail.code != NE_CONVERGENCE)
 * {
 *     printf("Error from nag_zheevx (f08fpc).\n", fail.message);
 *     exit_status = 1;
 *     goto END;
 * }
 * /* nag_complex_divide (a02cdc).
 * Normalize the eigenvectors.
 */
 * for(j=1; j<=m; j++)
 *     for(i=n; i>=1; i--)
 *         z(i, j) = nag_complex_divide(z(i, j),z(1, j));
 * /* Print solution */
 * printf("Number of eigenvalues found =%5\n", m);
 * printf("\nEigenvalues\n");
 * for (j = 0; j < m; ++j)
 *     printf("%8.4f\n", w[j], (j+1)%8 == 0?"\n": " ");
 * printf("\n\n");
 * /* nag_gen_complx_mat_print (x04dac).
 * Print selected eigenvectors.
 */
 * INIT_FAIL(fail_print);
 * fflush(stdout);
 * nag_gen_complx_mat_print(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, m, z, pdz, "Selected eigenvectors", 0, &fail_print);
 * if (fail_print.code != NE_NOERROR)
 * {
 *     printf("Error from nag_gen_complx_mat_print (x04dac).\n", fail_print.message);
 *     exit_status = 1;
 *     goto END;
 * }
 * if (fail.code == NE_CONVERGENCE)
 * {
 *     printf("eigenvectors failed to converge\n");
 *     printf("Indices of eigenvectors that did not converge\n");
 *     for (j = 0; j < m; ++j)
 *         printf("%8NAG_IFMT\n", index[j], (j+1)%8 == 0?"\n": " ");
 * 
 */
```
10.2 Program Data

nag_zheevx (f08fpc) Example Program Data

<table>
<thead>
<tr>
<th>Value of n</th>
<th>-2.0</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values of vl and vu</td>
<td>(1.0, 0.0)</td>
<td>(2.0, -1.0)</td>
</tr>
<tr>
<td></td>
<td>(2.0, 0.0)</td>
<td>(3.0, -2.0)</td>
</tr>
<tr>
<td></td>
<td>(3.0, 0.0)</td>
<td>(4.0, -3.0)</td>
</tr>
<tr>
<td></td>
<td>(4.0, 0.0)</td>
<td></td>
</tr>
<tr>
<td>End of matrix A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10.3 Program Results

nag_zheevx (f08fpc) Example Program Results

Number of eigenvalues found = 2

Eigenvalues
-0.6886  1.1412

Selected eigenvectors

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>0.0000</td>
<td>-0.0000</td>
</tr>
<tr>
<td>2</td>
<td>-0.7703</td>
<td>0.0516</td>
</tr>
<tr>
<td></td>
<td>-0.1746</td>
<td>1.2795</td>
</tr>
<tr>
<td>3</td>
<td>0.4559</td>
<td>-1.1962</td>
</tr>
<tr>
<td></td>
<td>0.4892</td>
<td>-0.2954</td>
</tr>
<tr>
<td>4</td>
<td>-0.3464</td>
<td>0.7876</td>
</tr>
<tr>
<td></td>
<td>-0.4448</td>
<td>-0.5075</td>
</tr>
</tbody>
</table>