NAG Library Function Document
nag_zhbevx (f08hpc)

1 Purpose
nag_zhbevx (f08hpc) computes selected eigenvalues and, optionally, eigenvectors of a complex $n$ by $n$ Hermitian band matrix $A$ of bandwidth $(2kd + 1)$. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

2 Specification

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

void nag_zhbevx (Nag_OrderType order, Nag_JobType job, Nag_RangeType range,
                 Nag_UploType uplo, Integer n, Integer kd, Complex ab[], Integer pdab,
                 Complex q[], Integer pdq, 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 band 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:  
```c
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:  
```c
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
    Input
    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 ith to jth eigenvalues will be found.

Constraint: range = Nag_AllValues, Nag_Interval or Nag_Indices.

4: uplo – Nag_UploType
    Input
    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
    Input
    On entry: n, the order of the matrix A.
    Constraint: n ≥ 0.

6: kd – Integer
    Input
    On entry: if uplo = Nag_Upper, the number of superdiagonals, k_d, of the matrix A.
    If uplo = Nag_Lower, the number of subdiagonals, k_d, of the matrix A.

Constraint: kd ≥ 0.

7: ab[dim] – Complex
    Input/Output
    Note: the dimension, dim, of the array ab must be at least max(1, pdab × n).
    On entry: the upper or lower triangle of the n by n Hermitian band matrix A.
    This is stored as a notional two-dimensional array with row elements or column elements stored contiguously. The storage of elements of A_{ij}, depends on the order and uplo arguments as follows:

    if order = Nag_ColMajor and uplo = Nag_Upper,
    A_{ij} is stored in ab[k_d + i - j + (j - 1) × pdab], for j = 1, ..., n and
    i = max(1, j - k_d), ..., j;
    if order = Nag_ColMajor and uplo = Nag_Lower,
    A_{ij} is stored in ab[i - j + (j - 1) × pdab], for j = 1, ..., n and
    i = j, ..., min(n, j + k_d);
    if order = Nag_RowMajor and uplo = Nag_Upper,
    A_{ij} is stored in ab[j - i + (i - 1) × pdab], for i = 1, ..., n and
    j = i, ..., min(n, i + k_d);
    if order = Nag_RowMajor and uplo = Nag_Lower,
    A_{ij} is stored in ab[k_d + j - i + (i - 1) × pdab], for i = 1, ..., n and
    j = max(1, i - k_d), ..., i.

On exit: ab is overwritten by values generated during the reduction to tridiagonal form.

The first superdiagonal or subdiagonal and the diagonal of the tridiagonal matrix T are returned in ab using the same storage format as described above.
8: \textbf{pdab} – Integer

\textit{Input}

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

\textit{Constraint}: \(\texttt{pdab} \geq \texttt{kd} + 1\).

9: \textbf{q}[\textit{dim}] – Complex

\textit{Output}

\textit{Note}: the dimension, \textit{dim}, of the array \texttt{q} must be at least

\[
\max(1, \texttt{pdq} \times \texttt{n}) \text{ when } \texttt{job} = \text{Nag\_DoBoth};
\]

\[ 1 \text{ otherwise.}
\]

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

\[
\texttt{q}[(j - 1) \times \texttt{pdq} + i - 1] \text{ when } \texttt{order} = \text{Nag\_ColMajor};
\]

\[
\texttt{q}[(i - 1) \times \texttt{pdq} + j - 1] \text{ when } \texttt{order} = \text{Nag\_RowMajor}.
\]

\textit{On exit}: if \(\texttt{job} = \text{Nag\_DoBoth}\), the \(n\) by \(n\) unitary matrix used in the reduction to tridiagonal form.

If \(\texttt{job} = \text{Nag\_EigVals}\), \texttt{q} is not referenced.

10: \textbf{pdq} – Integer

\textit{Input}

\textit{On entry}: the stride separating row or column elements (depending on the value of \texttt{order}) in the array \texttt{q}.

\textit{Constraints}:

\[
\text{if } \texttt{job} = \text{Nag\_DoBoth}, \texttt{pdq} \geq \max(1, \texttt{n});
\]

\[ \text{otherwise } \texttt{pdq} \geq 1.\]

11: \textbf{vl} – double

\textit{Input}

\textit{On entry}: if \texttt{range} = \text{Nag\_Interval}, the lower and upper bounds of the interval to be searched for eigenvalues.

If \texttt{range} = \text{Nag\_AllValues} or \text{Nag\_Indices}, \texttt{vl} and \texttt{vu} are not referenced.

\textit{Constraint}: if \texttt{range} = \text{Nag\_Interval}, \(\texttt{vl} < \texttt{vu}\).

12: \textbf{vu} – double

\textit{Input}

\textit{On entry}: if \texttt{range} = \text{Nag\_Interval}, the lower and upper bounds of the interval to be searched for eigenvalues.

If \texttt{range} = \text{Nag\_AllValues} or \text{Nag\_Interval}, \texttt{vl} and \texttt{vu} are not referenced.

\textit{Constraint}: if \texttt{range} = \text{Nag\_Interval}, \(\texttt{vl} < \texttt{vu}\).

13: \textbf{il} – Integer

\textit{Input}

\textit{On entry}: if \texttt{range} = \text{Nag\_Indices}, the indices (in ascending order) of the smallest and largest eigenvalues to be returned.

If \texttt{range} = \text{Nag\_AllValues} or \text{Nag\_Interval}, \texttt{il} and \texttt{iu} are not referenced.

\textit{Constraints}:

\[
\text{if } \texttt{range} = \text{Nag\_Indices} \text{ and } \texttt{n} = 0, \texttt{il} = 1 \text{ and } \texttt{iu} = 0;
\]

\[
\text{if } \texttt{range} = \text{Nag\_Indices} \text{ and } \texttt{n} > 0, 1 \leq \texttt{il} \leq \texttt{iu} \leq \texttt{n}.
\]

14: \textbf{iu} – Integer

\textit{Input}

\textit{On entry}: if \texttt{range} = \text{Nag\_Indices}, the indices (in ascending order) of the smallest and largest eigenvalues to be returned.

If \texttt{range} = \text{Nag\_AllValues} or \text{Nag\_Interval}, \texttt{il} and \texttt{iu} are not referenced.

\textit{Constraints}:

\[
\text{if } \texttt{range} = \text{Nag\_Indices} \text{ and } \texttt{n} = 0, \texttt{il} = 1 \text{ and } \texttt{iu} = 0;
\]

\[
\text{if } \texttt{range} = \text{Nag\_Indices} \text{ and } \texttt{n} > 0, 1 \leq \texttt{il} \leq \texttt{iu} \leq \texttt{n}.
\]

15: \textbf{abstol} – double

\textit{Input}

\textit{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 \texttt{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 \texttt{abstol} is set to twice the underflow threshold \(2 \times \text{nag\_real\_safe\_small\_number}\), not zero. If this function returns with \texttt{fail\_code} =
NE_CONVERGENCE, indicating that some eigenvectors did not converge, try setting abstol to $2 \times \text{nag_real_safe_small_number}$. See Demmel and Kahan (1990).

16:  \textbf{m} – Integer * \\
\textit{Output} \\
\textit{On exit:} the total number of eigenvalues found. $0 \leq \text{m} \leq \text{n}$. \\
If \textbf{range} = \text{Nag\_AllValues}, $\text{m} = \text{n}$. \\
If \textbf{range} = \text{Nag\_Indices}, $\text{m} = \text{iu} - \text{il} + 1$.

17:  \textbf{w}[	ext{n}] – double \\
\textit{Output} \\
\textit{On exit:} the first \text{m} elements contain the selected eigenvalues in ascending order.

18:  \textbf{z}[	ext{dim}] – Complex \\
\textit{Output} \\
\textit{Note:} the dimension, \text{dim}, of the array \textbf{z} must be at least \[ \max(1, \text{pdz} \times \text{n}) \] when \job = \text{Nag\_DoBoth}; \\
1 otherwise. \\
The $(i,j)$th element of the matrix $Z$ is stored in \\
\[
\text{z}[(j - 1) \times \text{pdz} + i - 1] \quad \text{when order} = \text{Nag\_ColMajor}; \\
\text{z}[(i - 1) \times \text{pdz} + j - 1] \quad \text{when order} = \text{Nag\_RowMajor}.
\]
\textit{On exit:} if \job = \text{Nag\_DoBoth}, then \\
if \fail.code = \text{NE\_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 (\fail.code = \text{NE\_CONVERGENCE}), then that column of \\
$Z$ contains the latest approximation to the eigenvector, and the index of the eigenvector is \\
returned in \textbf{jfail}.

If \job = \text{Nag\_EigVals}, \textbf{z} is not referenced.

19:  \textbf{pdz} – Integer \\
\textit{Input} \\
\textit{On entry:} the stride separating row or column elements (depending on the value of \textbf{order}) in the \\
array \textbf{z}. \\
\textit{Constraints:} \\
\quad if \job = \text{Nag\_DoBoth}, \text{pdz} \geq \max(1, \text{n}); \\
\quad otherwise \text{pdz} \geq 1.

20:  \textbf{jfail}[	ext{dim}] – Integer \\
\textit{Output} \\
\textit{Note:} the dimension, \text{dim}, of the array \textbf{jfail} must be at least $\max(1, \text{n})$. \\
\textit{On exit:} if \job = \text{Nag\_DoBoth}, then \\
if \fail.code = \text{NE\_NOERROR}, the first \text{m} elements of \textbf{jfail} are zero; \\
if \fail.code = \text{NE\_CONVERGENCE}, \textbf{jfail} contains the indices of the eigenvectors that \\
failed to converge.

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

21:  \textbf{fail} – NagError * \\
\textit{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 jfail.

NE_ENUM_INT_2
On entry, job = <value>, pdq = <value> and n = <value>.
Constraint: if job = Nag_DoBoth, pdq ≥ max(1, n);
otherwise pdq ≥ 1.
On entry, job = <value>, pdz = <value> and n = <value>.
Constraint: if job = Nag_DoBoth, pdz ≥ max(1, n);
otherwise pdz ≥ 1.

NE_ENUM_INT_3
On entry, range = <value>, il = <value>, iu = <value> and n = <value>.
Constraint: if range = Nag_Indices and n = 0, il = 1 and iu = 0;
if range = Nag_Indices and n > 0, 1 ≤ il ≤ iu ≤ n.

NE_ENUM_REAL_2
On entry, range = <value>, vl = <value> and vu = <value>.
Constraint: if range = Nag_Interval, vl < vu.

NE_INT
On entry, kd = <value>.
Constraint: kd ≥ 0.
On entry, n = <value>.
Constraint: n ≥ 0.
On entry, pdab = <value>.
Constraint: pdab > 0.
On entry, pdq = <value>.
Constraint: pdq > 0.
On entry, pdz = <value>.
Constraint: pdz > 0.

NE_INT_2
On entry, pdab = <value> and kd = <value>.
Constraint: pdab ≥ kd + 1.

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.
7 Accuracy

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.

8 Parallelism and Performance

\texttt{nag_zhbevx (f08hpc)} is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

\texttt{nag_zhbevx (f08hpc)} 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 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 total number of floating-point operations is proportional to \( k_d n^2 \) if \( \text{job} = \text{Nag} \_\text{EigVals} \), and is proportional to \( n^3 \) if \( \text{job} = \text{Nag} \_\text{DoBoth} \) and \( \text{range} = \text{Nag} \_\text{AllValues} \), otherwise the number of floating-point operations will depend upon the number of computed eigenvectors.

The real analogue of this function is \texttt{nag_dsbevx (f08hbc)}.

10 Example

This example finds the eigenvalues in the half-open interval \((-2, 2] \), and the corresponding eigenvectors, of the Hermitian band matrix
\[
A = \begin{pmatrix}
1 & 2 - i & 3 - i & 0 & 0 \\
2 + i & 2 & 3 - 2i & 4 - 2i & 0 \\
3 + i & 3 + 2i & 3 & 4 - 3i & 5 - 3i \\
0 & 4 + 2i & 4 + 3i & 4 & 5 - 4i \\
0 & 0 & 5 + 3i & 5 + 4i & 5
\end{pmatrix}.
\]

10.1 Program Text

/* \texttt{nag_zhbevx (f08hpc)} Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 23, 2011. */
*/
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.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, kd, m, n, pdab, pdq, pdz;
/* Arrays */
char nag_enum_arg[40];
Complex *ab = 0, *q = 0, *z = 0;
double *w = 0;
Integer *index = 0;
/* Nag Types */
Nag_OrderType order;
Nag_UploType uplo;
NagError fail, fail_print;

#ifdef NAG_COLUMN_MAJOR
#define AB_UPPER(I, J) ab[(J - 1) * pdab + kd + I - J]
#define AB_LOWER(I, J) ab[(J - 1) * pdab + I - J]
#else
#define AB_UPPER(I, J) ab[(I - 1) * pdab + J - I]
#define AB_LOWER(I, J) ab[(I - 1) * pdab + kd + J - I]
#endif
INIT_FAIL(fail);
printf("nag_zhbevx (f08hpc) Example Program Results\n\n");
/* Skip heading in data file */
#ifdef _WIN32
scanf_s("%*[\n"]);
#else
scanf("%*[\n"]);
#endif
#ifdef _WIN32
scanf_s("%"NAG_IFMT"%"NAG_IFMT"%*[\n"]", &n, &kd);
#else
scanf("%"NAG_IFMT"%"NAG_IFMT"%*[\n"]", &n, &kd);
#endif

/* Read uplo */
#ifdef _WIN32
scanf_s("%39s%*[\n"]", nag_enum_arg, _countof(nag_enum_arg));
#else
scanf("%39s%*[\n"]", nag_enum_arg);
#endif
/* nag_enum_name_to_value (x04nac).
* Converts NAG enum member name to value
*/
uplo = (Nag_UploType) nag_enum_name_to_value(nag_enum_arg);

/* Allocate memory */
if (!((ab = NAG_ALLOC((kd+1)*n, Complex)) ||
    (q = NAG_ALLOC(n*n, Complex)) ||
    (z = NAG_ALLOC(n*n, Complex)) ||
    (w = NAG_ALLOC(n, double)) ||
    (index = NAG_ALLOC(n, Integer)))
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}

pdab = kd + 1;
pdq = n;
pdz = n;

/* Read the lower and upper bounds of the interval to be searched,
* and read the upper or lower triangular part of the matrix A
* from data file.
*/
#ifdef _WIN32
scanf_s("%lf%lf%*[\n"]", &vl, &vu);
#else

#endif

Mark 25 f08hpc.7
#else
    scanf("%lf%lf%*\n", &vl, &vu);
#endif
if (uplo == Nag_Upper) {
    for (i = 1; i <= n; ++i)
        for (j = i; j <= MIN(n, i + kd); ++j)
            #ifdef _WIN32
                scanf_s(" ( %lf , %lf )", &AB_UPPER(i, j).re, &AB_UPPER(i, j).im);
            #else
                scanf(" ( %lf , %lf )", &AB_UPPER(i, j).re, &AB_UPPER(i, j).im);
            #endif
        #ifdef _WIN32
            scanf_s("%*\n");
        #else
            scanf("%*\n");
        #endif
    }
else if (uplo == Nag_Lower) {
    for (i = 1; i <= n; ++i)
        for (j = MAX(1, i - kd); j <= i; ++j)
            #ifdef _WIN32
                scanf_s(" ( %lf , %lf )", &AB_LOWER(i, j).re, &AB_LOWER(i, j).im);
            #else
                scanf(" ( %lf , %lf )", &AB_LOWER(i, j).re, &AB_LOWER(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_zhbevx (f08hpc).  
 * Solve the band symmetric eigenvalue problem.  
 */
if (fail.code != NE_NOERROR && fail.code != NE_CONVERGENCE)
{
    printf("Error from nag_zhbevx (f08hpc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Print solution */
printf("Number of eigenvalues found =%5"NAG_IFMT"\n", m);
    printf("\nEigenvalues\n");
    for (j = 0; j < m; ++j)
        printf("%8.4f%s", w[j], (j+1)%8 == 0?"\n": "");
    printf("\n");
/* nag_gen_complx_mat_print (x04dac).  
 * Print selected eigenvectors.  
 */
if (fail_print.code != NE_NOERROR)
{
    printf("Error from nag_gen_complx_mat_print (x04dac).\n%s\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("%8"NAG_IFMT"%s", index[j], (j+1)%8 == 0?"\n":" ");
}

END:
NAG_FREE(ab);
NAG_FREE(w);
NAG_FREE(index);

    return exit_status;
}

#define AB_UPPER
#define AB_LOWER

10.2 Program Data

nag_zhbevx (f08hpc) Example Program Data

    5  2 :Values of n and kd
    Nag_Upper          :Value of uplo
    -2.0  2.0          :Values of vl and vu

(1.0, 0.0) (2.0,-1.0) (3.0,-1.0) (4.0,-2.0) (5.0,-3.0)
(2.0, 0.0) (3.0,-2.0) (4.0,-3.0) (5.0,-4.0)
(3.0, 0.0) (4.0,-2.0) (5.0,-4.0)
(4.0, 0.0) (5.0,-3.0)
(5.0, 0.0) :End of matrix A

10.3 Program Results

nag_zhbevx (f08hpc) Example Program Results

Number of eigenvalues found = 2

Eigenvalues
-1.4094  1.4421

Selected eigenvectors
  1  2
1  0.6367  0.4516
   -0.0000  -0.0000
2  -0.2578 -0.3029
    0.2413  -0.4402
3  -0.3039  0.3160
   -0.3481  0.2978
4   0.3450 -0.4088
    -0.0832  -0.3213
5  -0.2469  0.0204
    0.2634  0.2262