NAG Library Function Document

nag_zhpev (f08gnc)

1 Purpose

nag_zhpev (f08gnc) computes all the eigenvalues and, optionally, all the eigenvectors of a complex n by n Hermitian matrix A in packed storage.

2 Specification

```
#include <nag.h>
#include <nagf08.h>
void nag_zhpev (Nag_OrderType order, Nag_JobType job, Nag_UploType uplo,
     Integer n, Complex ap[], double w[], Complex z[], Integer pdz,
     NagError *fail)
```

3 Description

The Hermitian matrix A is first reduced to real tridiagonal form, using unitary similarity transformations, and then the QR algorithm is applied to the tridiagonal matrix to compute the eigenvalues and (optionally) the eigenvectors.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

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

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: **job** – Nag_JobType *Input*

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: **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.

4: **n** – Integer *Input*

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

Constraint: **n** ≥ 0.

5: **ap**[dim] – Complex *Input/Output*

Note: the dimension, *dim*, of the array **ap** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$.

On entry: the upper or lower triangle of the n by n Hermitian matrix A, packed by rows or columns.

The storage of elements A_{ij} depends on the **order** and **uplo** arguments as follows:

if **order** = 'Nag_ColMajor' and **uplo** = 'Nag_Upper',
A_{ij} is stored in **ap**$[(j-1) \times j/2 + i - 1]$, for $i \leq j$;
if **order** = 'Nag_ColMajor' and **uplo** = 'Nag_Lower',
A_{ij} is stored in **ap**$[(2n-j) \times (j-1)/2 + i - 1]$, for $i \geq j$;
if **order** = 'Nag_RowMajor' and **uplo** = 'Nag_Upper',
A_{ij} is stored in **ap**$[(2n-i) \times (i-1)/2 + j - 1]$, for $i \leq j$;
if **order** = 'Nag_RowMajor' and **uplo** = 'Nag_Lower',
A_{ij} is stored in **ap**$[(i-1) \times i/2 + j - 1]$, for $i \geq j$.

On exit: **ap** is overwritten by the values generated during the reduction to tridiagonal form. The elements of the diagonal and the off-diagonal of the tridiagonal matrix overwrite the corresponding elements of A.

6: **w**[**n**] – double *Output*

On exit: the eigenvalues in ascending order.

7: **z**[dim] – Complex *Output*

Note: the dimension, *dim*, of the array **z** must be at least

$\max(1, \mathbf{pdz} \times \mathbf{n})$ when **job** = Nag_DoBoth;
1 otherwise.

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

z$[(j-1) \times \mathbf{pdz} + i - 1]$ when **order** = Nag_ColMajor;
z$[(i-1) \times \mathbf{pdz} + j - 1]$ when **order** = Nag_RowMajor.

On exit: if **job** = Nag_DoBoth, **z** contains the orthonormal eigenvectors of the matrix A, with the ith column of Z holding the eigenvector associated with **w**$[i-1]$.

If **job** = Nag_EigVals, **z** is not referenced.

8: **pdz** – Integer *Input*

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

Constraints:

if **job** = Nag_DoBoth, **pdz** $\geq \max(1, \mathbf{n})$;
otherwise **pdz** ≥ 1.

9: **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.

NE_BAD_PARAM

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

NE_CONVERGENCE

The algorithm failed to converge; ⟨*value*⟩ off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

NE_ENUM_INT_2

On entry, **job** = ⟨*value*⟩, **pdz** = ⟨*value*⟩ and **n** = ⟨*value*⟩.
Constraint: if **job** = Nag_DoBoth, **pdz** $\geq \max(1, \mathbf{n})$;
otherwise **pdz** ≥ 1.

NE_INT

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

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

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.

7 Accuracy

The computed eigenvalues and eigenvectors are exact for a nearby matrix $(A + E)$, where

$$\|E\|_2 = O(\epsilon)\|A\|_2,$$

and ϵ is the ***machine precision***. See Section 4.7 of Anderson *et al.* (1999) for further details.

8 Parallelism and Performance

nag_zhpev (f08gnc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_zhpev (f08gnc) 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 Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

Each eigenvector is normalized so that the element of largest absolute value is real and positive.

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

The real analogue of this function is nag_dspev (f08gac).

10 Example

This example finds all the eigenvalues of the Hermitian matrix

$$A = \begin{pmatrix} 1 & 2-i & 3-i & 4-i \\ 2+i & 2 & 3-2i & 4-2i \\ 3+i & 3+2i & 3 & 4-3i \\ 4+i & 4+2i & 4+3i & 4 \end{pmatrix},$$

together with approximate error bounds for the computed eigenvalues.

10.1 Program Text

```
/* nag_zhpev (f08gnc) Example Program.
 *
 * Copyright 2011 Numerical Algorithms Group.
 *
 * Mark 23, 2011.
 */

#include <math.h>
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf08.h>
#include <nagx02.h>
#include <nagx04.h>

int main(void)
{
  /* Scalars */
  double        eerrbd, eps;
  Integer       exit_status = 0, i, j, n;
  /* Arrays */
  char          nag_enum_arg[40];
  Complex       *ap = 0, *dummy = 0;
  double        *w = 0;
  /* Nag Types */
  Nag_OrderType order;
  Nag_UploType  uplo;
  NagError      fail;

#ifdef NAG_COLUMN_MAJOR
#define AP_UPPER(I, J) ap[J * (J - 1) / 2 + I - 1]
#define AP_LOWER(I, J) ap[(2 * n - J) * (J - 1) / 2 + I - 1]
  order = Nag_ColMajor;
#else
#define AP_LOWER(I, J) ap[I * (I - 1) / 2 + J - 1]
#define AP_UPPER(I, J) ap[(2 * n - I) * (I - 1) / 2 + J - 1]
  order = Nag_RowMajor;
#endif

  INIT_FAIL(fail);

  printf("nag_zhpev (f08gnc) Example Program Results\n\n");

  /* Skip heading in data file */
  scanf("%*[^\n]");
  scanf("%ld%*[^\n]", &n);

  /* Read uplo */
  scanf("%39s%*[^\n]", nag_enum_arg);
  /* 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 (!(ap = NAG_ALLOC(n*(n+1)/2, Complex)) ||
```

```
      !(dummy = NAG_ALLOC(1, Complex)) ||
      !(w = NAG_ALLOC(n, double)))
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  /* Read the upper or lower triangular part of the matrix A from data file */
  if (uplo == Nag_Upper) {
    for (i = 1; i <= n; ++i)
      for (j = i; j <= n; ++j)
        scanf(" ( %lf , %lf )", &AP_UPPER(i, j).re, &AP_UPPER(i, j).im);
    scanf("%*[^\n]");
  }
  else if (uplo == Nag_Lower) {
    for (i = 1; i <= n; ++i)
      for (j = 1; j <= i; ++j)
        scanf(" ( %lf , %lf )", &AP_LOWER(i, j).re, &AP_LOWER(i, j).im);
    scanf("%*[^\n]");
  }

  /* nag_zhpev (f08gnc).
   * Solve the Hermitian eigenvalue problem.
   */
  nag_zhpev(order, Nag_EigVals, uplo, n, ap, w, dummy, 1, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_zhpev (f08gnc).\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }

  /* Print solution */
  printf("Eigenvalues\n");
  for (j = 0; j < n; ++j)
    printf("%8.4f%s", w[j], (j+1)%8 == 0?"\n":" ");
  printf("\n");

  /* Get the machine precision, eps, using nag_machine_precision (X02AJC)
   * and compute the approximate error bound for the computed eigenvalues.
   * Note that for the 2-norm, ||A|| = max{|w[i]|,i=0..n-1}), and since
   * the eigenvalues are in ascending order: ||A|| = max(|w[0]|,|w[n-1]|).
   */
  eps = nag_machine_precision;
  eerrbd = eps * MAX(fabs(w[0]), fabs(w[n - 1]));

  /* Print the approximate error bound for the eigenvalues */
  printf("\nError estimate for the eigenvalues\n");
  printf("%11.1e\n", eerrbd);

 END:
  NAG_FREE(ap);
  NAG_FREE(dummy);
  NAG_FREE(w);

  return exit_status;
}

#undef AP_UPPER
#undef AP_LOWER
```

10.2 Program Data

```
nag_zhpev (f08gnc) Example Program Data

  4                                                    :Value of n
  Nag_Upper                                            :Value of uplo

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

10.3 Program Results

```
nag_zhpev (f08gnc) Example Program Results

Eigenvalues
 -4.2443  -0.6886   1.1412  13.7916

Error estimate for the eigenvalues
    1.5e-15
```