

NAG Library Function Document

nag_hermitian_eigensystem (f02axc)

1 Purpose

nag_hermitian_eigensystem (f02axc) calculates all the eigenvalues and eigenvectors of a complex Hermitian matrix.

2 Specification

```
#include <nag.h>
#include <nagf02.h>

void nag_hermitian_eigensystem (Integer n, const Complex a[], Integer tda,
    double r[], Complex v[], Integer tdv, NagError *fail)
```

3 Description

The complex Hermitian matrix A is first reduced to a real tridiagonal matrix by $n-2$ unitary transformations and a subsequent diagonal transformation. The eigenvalues and eigenvectors are then derived using the QL algorithm, an adaptation of the QR algorithm.

4 References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation II, Linear Algebra* Springer-Verlag

5 Arguments

- | | | |
|----|--|---------------|
| 1: | n – Integer | <i>Input</i> |
| | <i>On entry:</i> n , the order of the matrix A . | |
| | <i>Constraint:</i> $n \geq 1$. | |
| 2: | a [$n \times \mathbf{tda}$] – const Complex | <i>Input</i> |
| | <i>On entry:</i> the elements of the lower triangle of the n by n complex Hermitian matrix A . Elements of the array above the diagonal need not be set. See also Section 9. | |
| 3: | tda – Integer | <i>Input</i> |
| | <i>On entry:</i> the stride separating matrix column elements in the array a . | |
| | <i>Constraint:</i> $\mathbf{tda} \geq n$. | |
| 4: | r [n] – double | <i>Output</i> |
| | <i>On exit:</i> the eigenvalues in ascending order. | |
| 5: | v [$n \times \mathbf{tdv}$] – Complex | <i>Output</i> |
| | Note: the (i, j) th element of the matrix V is stored in $\mathbf{v}[(i-1) \times \mathbf{tdv} + j - 1]$. | |
| | <i>On exit:</i> the eigenvectors, stored by columns. The i th column corresponds to the i th eigenvector. The eigenvectors are normalized so that the sum of the squares of the moduli of the elements is equal to 1 and the element of largest modulus is real. See also Section 9. | |

- 6: **tdv** – Integer *Input*
On entry: the stride separating matrix column elements in the array **v**.
Constraint: **tdv** \geq **n**.
- 7: **fail** – NagError * *Input/Output*
The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_2_INT_ARG_LT

On entry, **tda** = $\langle value \rangle$ while **n** = $\langle value \rangle$. These arguments must satisfy **tda** \geq **n**.

On entry, **tdv** = $\langle value \rangle$ while **n** = $\langle value \rangle$. These arguments must satisfy **tdv** \geq **n**.

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_DIAG_IMAG_NON_ZERO

Matrix diagonal element $\mathbf{a}[(\langle value \rangle) \times \mathbf{tda} + \langle value \rangle]$ has nonzero imaginary part.

NE_INT_ARG_LT

On entry, **n** = $\langle value \rangle$.

Constraint: **n** \geq 1.

NE_TOO_MANY_ITERATIONS

More than $\langle value \rangle$ iterations are required to isolate all the eigenvalues.

7 Accuracy

The eigenvectors are always accurately orthogonal but the accuracy of the individual eigenvalues and eigenvectors is dependent on their inherent sensitivity to small changes in the original matrix. For a detailed error analysis see page 235 of Wilkinson and Reinsch (1971).

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by `nag_hermitian_eigensystem` (f02axc) is approximately proportional to n^3 .

The function may be called with the same actual array supplied for **a** and **v**, in which case the eigenvectors will overwrite the original matrix *A*.

10 Example

To calculate the eigenvalues and eigenvectors of the complex Hermitian matrix:

$$\begin{pmatrix} 0.50 & 0.00 & 1.84 + 1.38i & 2.08 - 1.56i \\ 0.00 & 0.50 & 1.12 + 0.84i & -0.56 + 0.42i \\ 1.84 - 1.38i & 1.12 - 0.84i & 0.50 & 0.00 \\ 2.08 + 1.56i & -0.56 - 0.42i & 0.00 & 0.50 \end{pmatrix}.$$

10.1 Program Text

```

/* nag_hermitian_eigensystem (f02axc) Example Program.
 *
 * Copyright 1991 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 * Mark 8 revised, 2004.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagf02.h>

#define A(I, J) a[(I) *tda + J]
#define V(I, J) v[(I) *tdv + J]
int main(void)
{
    Complex *a = 0, *v = 0;
    Integer exit_status = 0, i, j, n, tda, tdv;
    NagError fail;
    double *r = 0;

    INIT_FAIL(fail);

    printf(
        "nag_hermitian_eigensystem (f02axc) Example Program Results\n");
    scanf("%*[\n]"); /* Skip heading in data file */
    scanf("%ld", &n);
    if (n >= 1)
    {
        if (!(r = NAG_ALLOC(n, double)) ||
            !(a = NAG_ALLOC((n)*(n), Complex)) ||
            !(v = NAG_ALLOC((n)*(n), Complex)))
        {
            printf("Allocation failure\n");
            exit_status = -1;
            goto END;
        }
        tda = n;
        tdv = n;
    }
    else
    {
        printf("Invalid n.\n");
        exit_status = 1;
        return exit_status;
    }
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            scanf(" ( %lf, %lf ) ", &A(i, j).re, &A(i, j).im);
    /* nag_hermitian_eigensystem (f02axc).
     * All eigenvalues and eigenvectors of complex Hermitian
     * matrix
     */
    nag_hermitian_eigensystem(n, a, tda, r, v, tdv, &fail);
    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_hermitian_eigensystem (f02axc).\n%s\n",
            fail.message);
        exit_status = 1;
        goto END;
    }
    printf("Eigenvalues\n");
    for (i = 0; i < n; i++)
        printf("%9.4f", r[i]);
    printf("\nEigenvectors\n");
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            printf("(%7.3f %7.3f )%s", V(i, j).re, V(i, j).im,

```

```

                (j%4 == 3 || j == n-1)? "\n": " ");
END:
  NAG_FREE(r);
  NAG_FREE(a);
  NAG_FREE(v);
  return exit_status;
}

```

10.2 Program Data

```

nag_hermitian_eigensystem (f02axc) Example Program Data
4
(0.50, 0.00) ( 0.00, 0.00) (1.84, 1.38 ) ( 2.08,-1.56 )
(0.00, 0.00) ( 0.50, 0.00) (1.12, 0.84 ) (-0.56, 0.42 )
(1.84,-1.38) ( 1.12,-0.84) (0.50, 0.00 ) ( 0.00, 0.00 )
(2.08, 1.56) (-0.56,-0.42) (0.00, 0.00 ) ( 0.50, 0.00 )

```

10.3 Program Results

```

nag_hermitian_eigensystem (f02axc) Example Program Results
Eigenvalues
-3.0000 -1.0000  2.0000  4.0000
Eigenvectors
( 0.700  0.000 ) ( -0.100 -0.000 ) ( -0.100  0.000 ) ( 0.700  0.000 )
( 0.100 -0.000 ) ( 0.700  0.000 ) ( 0.700  0.000 ) ( 0.100  0.000 )
( -0.400  0.300 ) ( -0.400  0.300 ) ( 0.400 -0.300 ) ( 0.400 -0.300 )
( -0.400 -0.300 ) ( 0.400  0.300 ) ( -0.400 -0.300 ) ( 0.400  0.300 )

```
