

NAG Library Function Document

nag_real_symm_eigensystem (f02abc)

1 Purpose

nag_real_symm_eigensystem (f02abc) calculates all the eigenvalues and eigenvectors of a real symmetric matrix.

2 Specification

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

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

3 Description

nag_real_symm_eigensystem (f02abc) reduces the real symmetric matrix A to a real symmetric tridiagonal matrix by Householder's method. The eigenvalues and eigenvectors are calculated using the QL 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>On entry:</i> n , the order of the matrix A .
<i>Constraint:</i> $n \geq 1$. | Input |
| 2: | a [$n \times tda$] – const double
Note: the (i, j) th element of the matrix A is stored in a [($i - 1$) \times tda + $j - 1$].
<i>On entry:</i> the lower triangle of the n by n symmetric matrix A . The elements of the array above the diagonal need not be set. See also Section 9 | Input |
| 3: | tda – Integer
<i>On entry:</i> the stride separating matrix column elements in the array a .
<i>Constraint:</i> tda $\geq n$. | Input |
| 4: | r [n] – double
<i>On exit:</i> the eigenvalues in ascending order. | Output |
| 5: | v [$n \times tdv$] – double
Note: the (i, j) th element of the matrix V is stored in v [($i - 1$) \times tdv + $j - 1$].
<i>On exit:</i> the normalized eigenvectors, stored by columns; the i th column corresponds to the i th eigenvalue. The eigenvectors are normalized so that the sum of squares of the elements is equal to 1. | Output |

- 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_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 eigenvectors is dependent on their inherent sensitivity to changes in the original matrix. For a detailed error analysis see pages 222 and 235 of Wilkinson and Reinsch (1971).

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by nag_real_symm_eigensystem (f02abc) is approximately proportional to n^3 .

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

10 Example

To calculate all the eigenvalues and eigenvectors of the real symmetric matrix

$$\begin{pmatrix} 0.5 & 0.0 & 2.3 & -2.6 \\ 0.0 & 0.5 & -1.4 & -0.7 \\ 2.3 & -1.4 & 0.5 & 0.0 \\ -2.6 & -0.7 & 0.0 & 0.5 \end{pmatrix}.$$

10.1 Program Text

```

/* nag_real_symm_eigensystem (f02abc) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 2 revised, 1992.
 * 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)
{
  Integer  exit_status = 0, i, j, n, tda, tdv;
  NagError fail;
  double   *a = 0, *r = 0, *v = 0;

  INIT_FAIL(fail);

  printf(
    "nag_real_symm_eigensystem (f02abc) Example Program Results\n");
  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[\n]");
#else
  scanf("%*[\n]");
#endif
#ifdef _WIN32
  scanf_s("%"NAG_IFMT"", &n);
#else
  scanf("%"NAG_IFMT"", &n);
#endif
  if (n >= 1)
  {
    if (!(a = NAG_ALLOC(n*n, double)) ||
        !(r = NAG_ALLOC(n, double)) ||
        !(v = NAG_ALLOC(n*n, double)))
    {
      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++)
#ifdef _WIN32
    scanf_s("%lf", &A(i, j));
#else
    scanf("%lf", &A(i, j));
#endif
  /* nag_real_symm_eigensystem (f02abc).
   * All eigenvalues and eigenvectors of real symmetric matrix
   */
  nag_real_symm_eigensystem(n, a, tda, r, v, tdv,

```

```

                                &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_real_symm_eigensystem (f02abc).\n%s\n",
           fail.message);
    exit_status = 1;
    goto END;
}

printf("Eigenvalues\n");
for (i = 0; i < n; i++)
    printf("%9.4f%s", r[i], (i%8 == 7 || i == n-1)?"\n":" ");
printf("Eigenvectors\n");
for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
        printf("%9.4f%s", V(i, j), (j%8 == 7 || j == n-1)?"\n":" ");
END:
NAG_FREE(a);
NAG_FREE(r);
NAG_FREE(v);
return exit_status;
}

```

10.2 Program Data

nag_real_symm_eigensystem (f02abc) Example Program Data

```

4
0.5  0.0  2.3 -2.6
0.0  0.5 -1.4 -0.7
2.3 -1.4  0.5  0.0
-2.6 -0.7  0.0  0.5

```

10.3 Program Results

nag_real_symm_eigensystem (f02abc) Example Program Results

```

Eigenvalues
-3.0000  -1.0000   2.0000   4.0000
Eigenvectors
 0.7000   0.1000   0.1000  -0.7000
-0.1000   0.7000   0.7000   0.1000
-0.5000   0.5000  -0.5000  -0.5000
 0.5000   0.5000  -0.5000   0.5000

```
