nag_real_symm_general_eigensystem (f02aec)

1. Purpose

nag_real_symm_general_eigensystem (f02aec) calculates all the eigenvalues and eigenvectors of $Ax = \lambda Bx$, where A is a real symmetric matrix and B is a real symmetric positive-definite matrix.

2. Specification

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

void nag_real_symm_general_eigensystem(Integer n, double a[], Integer tda,
     double b[], Integer tdb, double r[], double v[], Integer tdv,
     NagError *fail)
```

3. Description

The problem is reduced to the standard symmetric eigenproblem using Cholesky's method to decompose B into triangular matrices $B = LL^T$, where L is lower triangular. Then $Ax = \lambda Bx$ implies $(L^{-1}AL^{-T})(L^Tx) = \lambda(L^Tx)$; hence the eigenvalues of $Ax = \lambda Bx$ are those of $Py = \lambda y$, where P is the symmetric matrix $L^{-1}AL^{-T}$. Householder's method is used to tridiagonalise the matrix P and the eigenvalues are found using the QL algorithm. An eigenvector z of the derived problem is related to an eigenvector x of the original problem by $z = L^Tx$. The eigenvectors z are determined using the QL algorithm and are normalised so that $z^Tz = 1$; the eigenvectors of the original problem are then determined by solving $L^Tx = z$, and are normalised so that $x^TBx = 1$.

4. Parameters

n

Input: n, the order of the matrices A and B.
Constraint: $\mathbf{n} \geq 1$.

a[n][tda]

Input: the upper triangle of the n by n symmetric matrix A. The elements of the array below the diagonal need not be set.
Output: the lower triangle of the array is overwritten. The rest of the array is unchanged. See also Section 6.

tda

Input: the second dimension of the array **a** as declared in the function from which nag_real_symm_general_eigensystem is called.
Constraint: $\mathbf{tda} \geq \mathbf{n}$.

b[n][tdb]

Input: the upper triangle of the n by n symmetric positive-definite matrix B. The elements of the array below the diagonal need not be set.
Output: the elements below the diagonal are overwritten. The rest of the array is unchanged.

tdb

Input: the second dimension of the array **b** as declared in the function from which nag_real_symm_general_eigensystem is called.
Constraint: $\mathbf{tdb} \geq \mathbf{n}$.

r[n]

Output: the eigenvalues in ascending order.

v[n][tdv]

Output: the normalised eigenvectors, stored by columns; the ith column corresponds to the ith eigenvalue. The eigenvectors x are normalised so that $x^TBx = 1$. See also Section 6.

tdv

Input: the second dimension of the array **v** as declared in the function from which nag_real_symm_general_eigensystem is called.
Constraint: $\mathbf{tdv} \geq \mathbf{n}$.

fail
The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_NOT_POS_DEF
The matrix B is not positive-definite, possibly due to rounding errors.

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

NE_INT_ARG_LT
On entry, **n** must not be less than 1: **n** $= \langle value \rangle$.

NE_2_INT_ARG_LT
On entry, **tda** $= \langle value \rangle$ while **n** $= \langle value \rangle$. These parameters must satisfy **tda** $\geq$ **n**.
On entry, **tdb** $= \langle value \rangle$ while **n** $= \langle value \rangle$. These parameters must satisfy **tdb** $\geq$ **n**.
On entry, **tdv** $= \langle value \rangle$ while **n** $= \langle value \rangle$. These parameters must satisfy **tdv** $\geq$ **n**.

NE_ALLOC_FAIL
Memory allocation failed.

6. Further Comments

The time taken by the function is approximately proportional to n^3.

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

6.1. Accuracy

In general this function is very accurate. However, if B is ill-conditioned with respect to inversion, the eigenvectors could be inaccurately determined. For a detailed error analysis see Wilkinson and Reinsch (1971) pp 310, 222 and 235.

6.2. References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation (Vol II, Linear Algebra)* Springer-Verlag pp 303–314, 212–226 and 227–240.

7. See Also

None.

8. Example

To calculate all the eigenvalues and eigenvectors of the general symmetric eigenproblem $Ax = \lambda Bx$ where A is the symmetric matrix

$$\begin{pmatrix} 0.5 & 1.5 & 6.6 & 4.8 \\ 1.5 & 6.5 & 16.2 & 8.6 \\ 6.6 & 16.2 & 37.6 & 9.8 \\ 4.8 & 8.6 & 9.8 & -17.1 \end{pmatrix}$$

and B is the symmetric positive-definite matrix

$$\begin{pmatrix} 1 & 3 & 4 & 1 \\ 3 & 13 & 16 & 11 \\ 4 & 16 & 24 & 18 \\ 1 & 11 & 18 & 27 \end{pmatrix}.$$

8.1. Program Text

```
/* nag_real_symm_general_eigensystem(f02aec) Example Program
 *
 * Copyright 1990 Numerical Algorithms Group.
 *
 * Mark 2 revised, 1992.
 */

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

#define NMAX 8
#define TDA NMAX
#define TDB NMAX
#define TDV NMAX

main()
{

  Integer  i, j, n;
  double a[NMAX][TDA], b[NMAX][TDB], r[NMAX], v[NMAX][TDV];

  Vprintf("f02aec Example Program Results\n");
  /* Skip heading in data file */
  Vscanf("%*[^\n]");
  Vscanf("%ld",&n);
  if (n<1 || n>NMAX)
    {
      Vfprintf(stderr, "N is out of range: N = %5ld\n", n);
      exit(EXIT_FAILURE);
    }
  for (i=0; i<n; i++)
    {
      for (j=0; j<n; j++)
        Vscanf("%lf",&a[i][j]);
      for (j=0; j<n; j++)
        Vscanf("%lf",&b[i][j]);
    }
  f02aec(n, (double *)a, (Integer)TDA, (double *)b, (Integer)TDB, r,
         (double *)v, (Integer)TDV, NAGERR_DEFAULT);
  Vprintf("Eigenvalues\n");
  for (i=0; i<n; i++)
    Vprintf("%9.4f%s",r[i],(i%8==7 || i==n-1) ? "\n" : " ");
  Vprintf("Eigenvectors\n");
  for (i=0; i<n; i++)
    for (j=0; j<n; j++)
      Vprintf("%9.4f%s",v[i][j],(j%8==7 || j==n-1) ? "\n" : " ");
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

```
f02aec Example Program Data
  4
   0.5   1.5   6.6   4.8     1.0   3.0   4.0   1.0
   1.5   6.5  16.2   8.6     3.0  13.0  16.0  11.0
   6.6  16.2  37.6   9.8     4.0  16.0  24.0  18.0
   4.8   8.6   9.8 -17.1     1.0  11.0  18.0  27.0
```

8.3. Program Results

```
f02aec Example Program Results
Eigenvalues
  -3.0000   -1.0000    2.0000    4.0000
Eigenvectors
  -4.3500   -2.0500   -3.9500    2.6500
   0.0500    0.1500    0.8500    0.0500
   1.0000    0.5000    0.5000   -1.0000
  -0.5000   -0.5000   -0.5000    0.5000
```