NAG Toolbox

nag_lapack_dsygvd (f08sc)

1 Purpose

nag_lapack_dsygvd (f08sc) computes all the eigenvalues and, optionally, the eigenvectors of a real generalized symmetric-definite eigenproblem, of the form

$$Az = \lambda Bz, \quad ABz = \lambda z \quad \text{or} \quad BAz = \lambda z,$$

where A and B are symmetric and B is also positive definite. If eigenvectors are desired, it uses a divide-and-conquer algorithm.

2 Syntax

```
[a, b, w, info] = nag_lapack_dsygvd(itype, jobz, uplo, a, b, 'n', n)
[a, b, w, info] = f08sc(itype, jobz, uplo, a, b, 'n', n)
```

3 Description

nag_lapack_dsygvd (f08sc) first performs a Cholesky factorization of the matrix B as $B = U^{\mathrm{T}}U$, when **uplo** $=$ 'U' or $B = LL^{\mathrm{T}}$, when **uplo** $=$ 'L'. The generalized problem is then reduced to a standard symmetric eigenvalue problem

$$Cx = \lambda x,$$

which is solved for the eigenvalues and, optionally, the eigenvectors; the eigenvectors are then backtransformed to give the eigenvectors of the original problem.

For the problem $Az = \lambda Bz$, the eigenvectors are normalized so that the matrix of eigenvectors, z, satisfies

$$Z^{\mathrm{T}}AZ = \Lambda \quad \text{and} \quad Z^{\mathrm{T}}BZ = I,$$

where Λ is the diagonal matrix whose diagonal elements are the eigenvalues. For the problem $ABz = \lambda z$ we correspondingly have

$$Z^{-1}AZ^{-\mathrm{T}} = \Lambda \quad \text{and} \quad Z^{\mathrm{T}}BZ = I,$$

and for $BAz = \lambda z$ we have

$$Z^{\mathrm{T}}AZ = \Lambda \quad \text{and} \quad Z^{\mathrm{T}}B^{-1}Z = I.$$

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 Parameters

5.1 Compulsory Input Parameters

1: **itype** – INTEGER

Specifies the problem type to be solved.

itype $= 1$
$Az = \lambda Bz$.

itype $= 2$
$ABz = \lambda z$.

itype $= 3$
$BAz = \lambda z$.

Constraint: **itype** $= 1$, 2 or 3.

2: **jobz** – CHARACTER(1)

Indicates whether eigenvectors are computed.

jobz = 'N'
Only eigenvalues are computed.

jobz = 'V'
Eigenvalues and eigenvectors are computed.

Constraint: **jobz** = 'N' or 'V'.

3: **uplo** – CHARACTER(1)

If **uplo** = 'U', the upper triangles of A and B are stored.

If **uplo** = 'L', the lower triangles of A and B are stored.

Constraint: **uplo** = 'U' or 'L'.

4: $\mathbf{a}(lda,:)$ – REAL (KIND=nag_wp) array

The first dimension of the array **a** must be at least $\max(1,\mathbf{n})$.

The second dimension of the array **a** must be at least $\max(1,\mathbf{n})$.

The n by n symmetric matrix A.

If **uplo** = 'U', the upper triangular part of a must be stored and the elements of the array below the diagonal are not referenced.

If **uplo** = 'L', the lower triangular part of a must be stored and the elements of the array above the diagonal are not referenced.

5: $\mathbf{b}(ldb,:)$ – REAL (KIND=nag_wp) array

The first dimension of the array **b** must be at least $\max(1,\mathbf{n})$.

The second dimension of the array **b** must be at least $\max(1,\mathbf{n})$.

The n by n symmetric matrix B.

If **uplo** = 'U', the upper triangular part of b must be stored and the elements of the array below the diagonal are not referenced.

If **uplo** = 'L', the lower triangular part of b must be stored and the elements of the array above the diagonal are not referenced.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the first dimension of the arrays **a**, **b** and the second dimension of the arrays **a**, **b**. (An error is raised if these dimensions are not equal.)

n, the order of the matrices A and B.

Constraint: $\mathbf{n} \geq 0$.

5.3 Output Parameters

1: $\mathbf{a}(lda,:)$ – REAL (KIND=nag_wp) array

The first dimension of the array **a** will be $\max(1,\mathbf{n})$.

The second dimension of the array **a** will be $\max(1,\mathbf{n})$.

If $\mathbf{jobz} = \text{'V'}$, **a** contains the matrix Z of eigenvectors. The eigenvectors are normalized as follows:

if $\mathbf{itype} = 1$ or 2, $Z^{\mathrm{T}}BZ = I$;

if $\mathbf{itype} = 3$, $Z^{\mathrm{T}}B^{-1}Z = I$.

If $\mathbf{jobz} = \text{'N'}$, the upper triangle (if $\mathbf{uplo} = \text{'U'}$) or the lower triangle (if $\mathbf{uplo} = \text{'L'}$) of **a**, including the diagonal, is overwritten.

2: $\mathbf{b}(ldb,:)$ – REAL (KIND=nag_wp) array

The first dimension of the array **b** will be $\max(1,\mathbf{n})$.

The second dimension of the array **b** will be $\max(1,\mathbf{n})$.

The triangular factor U or L from the Cholesky factorization $B = U^{\mathrm{T}}U$ or $B = LL^{\mathrm{T}}$.

3: $\mathbf{w}(\mathbf{n})$ – REAL (KIND=nag_wp) array

The eigenvalues in ascending order.

4: **info** – INTEGER

$\mathbf{info} = 0$ unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

$\mathbf{info} = -i$

If $\mathbf{info} = -i$, parameter i had an illegal value on entry. The parameters are numbered as follows:

1: **itype**, 2: **jobz**, 3: **uplo**, 4: **n**, 5: **a**, 6: **lda**, 7: **b**, 8: **ldb**, 9: **w**, 10: **work**, 11: **lwork**, 12: **iwork**, 13: **liwork**, 14: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

$\mathbf{info} = 1 \text{ to } \mathbf{n}$

If $\mathbf{info} = i$, nag_lapack_dsyevd (f08fc) failed to converge; i i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

info $>$ **n**

nag_lapack_dpotrf (f07fd) returned an error code; i.e., if **info** $=$ **n** $+ i$, for $1 \leq i \leq$ **n**, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

7 Accuracy

If B is ill-conditioned with respect to inversion, then the error bounds for the computed eigenvalues and vectors may be large, although when the diagonal elements of B differ widely in magnitude the eigenvalues and eigenvectors may be less sensitive than the condition of B would suggest. See Section 4.10 of Anderson *et al.* (1999) for details of the error bounds.

The example program below illustrates the computation of approximate error bounds.

8 Further Comments

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

The complex analogue of this function is nag_lapack_zhegvd (f08sq).

9 Example

This example finds all the eigenvalues and eigenvectors of the generalized symmetric eigenproblem $ABz = \lambda z$, where

$$A = \begin{pmatrix} 0.24 & 0.39 & 0.42 & -0.16 \\ 0.39 & -0.11 & 0.79 & 0.63 \\ 0.42 & 0.79 & -0.25 & 0.48 \\ -0.16 & 0.63 & 0.48 & -0.03 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 4.16 & -3.12 & 0.56 & -0.10 \\ -3.12 & 5.03 & -0.83 & 1.09 \\ 0.56 & -0.83 & 0.76 & 0.34 \\ -0.10 & 1.09 & 0.34 & 1.18 \end{pmatrix},$$

together with an estimate of the condition number of B, and approximate error bounds for the computed eigenvalues and eigenvectors.

The example program for nag_lapack_dsygv (f08sa) illustrates solving a generalized symmetric eigenproblem of the form $Az = \lambda Bz$.

9.1 Program Text

```
     function f08sc_example

fprintf('f08sc example results\n\n');

% Upper triangular parts of symmetric matrix A and symmetric definite matrix B
uplo = 'Upper';
n = 4;
a = [0.24,  0.39,  0.42, -0.16;
     0,    -0.11,  0.79,  0.63;
     0,     0,    -0.25,  0.48;
     0,     0,     0,    -0.03];
b = [4.16, -3.12,  0.56, -0.10;
     0,     5.03, -0.83,  1.09;
     0,     0,     0.76,  0.34;
     0,     0,     0,     1.18];

% Generalized eigenvalues and eigenvectors for problem ABz = lambda z
itype = nag_int(2);
jobz = 'Vectors';
[Z, U, w, info] = f08sc( ...
  itype, jobz, uplo, a, b);

% Normalize eigenvectors: largest element positive (with z'Bz = I)
for j = 1:n
  [~,k] = max(abs(Z(:,j)));
  if Z(k,j) < 0
```

```
    Z(:,j) = -Z(:,j);
  end
end

disp('Eigenvalues');
disp(w');
disp('Eigenvectors');
disp(Z);
```

9.2 Program Results

```
     f08sc example results

Eigenvalues
   -3.5411   -0.3347    0.2983    2.2544

Eigenvectors
   -0.0356   -0.1039   -0.7459    0.1909
    0.3809    0.4322   -0.7845    0.3540
   -0.2943    1.5644   -0.7144    0.5665
   -0.3186   -1.0647    1.1184    0.3859
```