NAG Toolbox for MATLAB

f02fd

1 Purpose

f02fd computes all the eigenvalues, and optionally all the eigenvectors, of a real symmetric-definite generalized eigenproblem.

2 Syntax

```
[a, b, w, ifail] = f02fd(itype, job, uplo, a, b, 'n', n)
```

3 Description

f02fd computes all the eigenvalues, and optionally all the eigenvectors, of a real symmetric-definite generalized eigenproblem of one of the following types:

1. $Az = \lambda Bz$
2. $ABz = \lambda z$
3. $BAz = \lambda z$

Here A and B are symmetric, and B must be positive-definite.

The method involves implicitly inverting B; hence if B is ill-conditioned with respect to inversion, the results may be inaccurate (see Section 7).

Note that the matrix Z of eigenvectors is not orthogonal, but satisfies the following relationships for the three types of problem above:

1. $Z^{\mathrm{T}}BZ = I$
2. $Z^{\mathrm{T}}BZ = I$
3. $Z^{\mathrm{T}}B^{-1}Z = I$

4 References

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

Parlett B N 1998 *The Symmetric Eigenvalue Problem* SIAM, Philadelphia

5 Parameters

5.1 Compulsory Input Parameters

1: **itype – int32 scalar**

Indicates the type of problem.

itype $= 1$

The problem is $Az = \lambda Bz$;

itype $= 2$

The problem is $ABz = \lambda z$;

itype $= 3$

The problem is $BAz = \lambda z$.

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

2: **job – string**

Indicates whether eigenvectors are to be computed.

job = 'N'

Only eigenvalues are computed.

job = 'V'

Eigenvalues and eigenvectors are computed.

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

3: **uplo – string**

Indicates whether the upper or lower triangular parts of A and B are stored.

uplo = 'U'

The upper triangular parts of A and B are stored.

uplo = 'L'

The lower triangular parts of A and B are stored.

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

4: **a(lda,$*$) – double array**

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

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

The n by n symmetric matrix A.

If **uplo** = 'U', the upper triangle of A must be stored and the elements of the array below the diagonal need not be set.

If **uplo** = 'L', the lower triangle of A must be stored and the elements of the array above the diagonal need not be set.

5: **b(ldb,$*$) – double array**

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

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

The n by n symmetric positive-definite matrix B.

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

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

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The second dimension of the array **a** The second dimension of the array **b**.

n, the order of the matrices A and B.

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

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldb, work, lwork

5.4 Output Parameters

1: **a(lda,$*$) – double array**

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

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

If **job** = 'V', **a** contains the matrix Z of eigenvectors, with the ith column holding the eigenvector z_i associated with the eigenvalue λ_i (stored in $\mathbf{w}(i)$).

If **uplo** = 'U', the upper triangular part of **a** is overwritten.

If **uplo** = 'L', the lower triangular part of **a** if overwritten.

2: **b(ldb,$*$) – double array**

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

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

The upper or lower triangle of B (as specified by **uplo**) contains the triangular factor U or L from the Cholesky factorization of B as $U^{\mathrm{T}}U$ or LL^{T}.

3: **w($*$) – double array**

Note: the dimension of the array **w** must be at least $\max(1, \mathbf{n})$.

The eigenvalues in ascending order.

4: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail = 1

On entry, **itype** $\neq$ 1, 2 or 3,
or **job** $\neq$ 'N' or 'V',
or **uplo** $\neq$ 'U' or 'L',
or $\mathbf{n} < 0$,
or $\mathbf{lda} < \max(1, \mathbf{n})$,
or $\mathbf{ldb} < \max(1, \mathbf{n})$,
or $\mathbf{lwork} < \max(1, 3 \times \mathbf{n})$.

ifail = 2

The QR algorithm failed to compute all the eigenvalues.

ifail $= 3$

The matrix B is not positive-definite.

7 Accuracy

If λ_i is an exact eigenvalue, and $\tilde{\lambda}_i$ is the corresponding computed value, then

for problems of the form $Az = \lambda Bz$,

$$\left|\tilde{\lambda}_i - \lambda_i\right| \leq c(n)\epsilon\|A\|_2\left\|B^{-1}\right\|_2;$$

for problems of the form $ABz = \lambda z$ or $BAz = \lambda z$,

$$\left|\tilde{\lambda}_i - \lambda_i\right| \leq c(n)\epsilon\|A\|_2\|B\|_2.$$

Here $c(n)$ is a modestly increasing function of n, and ϵ is the ***machine precision***.

If z_i is the corresponding exact eigenvector, and $\tilde{z}_i$ is the corresponding computed eigenvector, then the angle $\theta(\tilde{z}_i, z_i)$ between them is bounded as follows:

for problems of the form $Az = \lambda Bz$,

$$\theta(\tilde{z}_i, z_i) \leq \frac{c(n)\epsilon\|A\|_2\left\|B^{-1}\right\|_2(\kappa_2(B))^{1/2}}{\min\limits_{i \neq j}\left|\lambda_i - \lambda_j\right|};$$

for problems of the form $ABz = \lambda z$ or $BAz = \lambda z$,

$$\theta(\tilde{z}_i, z_i) \leq \frac{c(n)\epsilon\|A\|_2\|B\|_2(\kappa_2(B))^{1/2}}{\min\limits_{i \neq j}\left|\lambda_i - \lambda_j\right|}.$$

Here $\kappa_2(B)$ is the condition number of B with respect to inversion defined by: $\kappa_2(B) = \|B\|.\left\|B^{-1}\right\|$. Thus the accuracy of a computed eigenvector depends on the gap between its eigenvalue and all the other eigenvalues, and also on the condition of B.

8 Further Comments

f02fd calls functions from LAPACK in Chapter F08. It first reduces the problem to an equivalent standard eigenproblem $Cy = \lambda y$. It then reduces C to tridiagonal form T, using an orthogonal similarity transformation: $C = QTQ^{\mathrm{T}}$. To compute eigenvalues only, the function uses a root-free variant of the symmetric tridiagonal QR algorithm to reduce T to a diagonal matrix Λ. If eigenvectors are required, the function first forms the orthogonal matrix Q that was used in the reduction to tridiagonal form; it then uses the symmetric tridiagonal QR algorithm to reduce T to Λ, using a further orthogonal transformation: $T = S\Lambda S^{\mathrm{T}}$; and at the same time accumulates the matrix $Y = QS$, which is the matrix of eigenvectors of C. Finally it transforms the eigenvectors of C back to those of the original generalized problem.

Each eigenvector z is normalized so that:

for problems of the form $Az = \lambda Bz$ or $ABz = \lambda z$, $z^{\mathrm{T}}Bz = 1$;

for problems of the form $BAz = \lambda z$, $z^{\mathrm{T}}B^{-1}z = 1$.

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

9 Example

```
itype = int32(1);
job = 'Vectors';
uplo = 'L';
a = [0.24, 0, 0, 0;
     0.39, -0.11, 0, 0;
```

```
     0.42, 0.79, -0.25, 0;
     -0.16, 0.63, 0.48, -0.03];
b = [4.16, 0, 0, 0;
     -3.12, 5.03, 0, 0;
     0.56, -0.83, 0.76, 0;
     -0.1, 1.09, 0.34, 1.18];
[aOut, bOut, w, ifail] = f02fd(itype, job, uplo, a, b)
```

```
aOut =
    0.0690   -0.3080    0.4469   -0.5528
    0.5740   -0.5329    0.0371   -0.6766
    1.5428    0.3496   -0.0505   -0.9276
   -1.4004    0.6211   -0.4743    0.2510
bOut =
    2.0396         0         0         0
   -1.5297    1.6401         0         0
    0.2746   -0.2500    0.7887         0
   -0.0490    0.6189    0.6443    0.6161
w =
   -2.2254
   -0.4548
    0.1001
    1.1270
ifail =
           0
```