NAG Toolbox for MATLAB

f02fh

1 Purpose

f02fh finds the eigenvalues of the generalized band symmetric eigenvalue problem $Ax = \lambda Bx$, where A and B are symmetric band matrices and B is positive-definite.

2 Syntax

```
[a, b, d, ifail] = f02fh(ma, a, mb, b, 'n', n)
```

3 Description

The generalized band symmetric eigenvalue problem $Ax = \lambda Bx$, where A is a symmetric band matrix of band width $2m_A + 1$ and B is a positive-definite symmetric band matrix of band width $2m_B + 1$, is solved by a variant of the method of Crawford.

The function first transforms the problem $Ax = \lambda Bx$ to a standard band symmetric eigenvalue problem $Cy = \lambda y$, where C is a band symmetric matrix of band width $2m_A + 1$, using f01bu and f01bv. This step involves the implicit inversion of the matrix B and so this function should be used with caution if B is ill-conditioned with respect to inversion.

The eigenvalues of the standard problem $Cy = \lambda y$ are then obtained by reducing C to tridiagonal form and then applying the QL variant of the QR algorithm to the tridiagonal form, using f08he and f08jf. The above-mentioned functions should be consulted for further information on the methods used.

Once the eigenvalues have been found by this function, selected eigenvectors may be obtained by repeated calls to f02sd with the original matrices A and B as data.

The function assumes that $m_A \geq m_B$ and hence if the band width of A is actually smaller than that of B, then A must be filled out with additional zero diagonals.

4 References

Crawford C R 1973 Reduction of a band-symmetric generalized eigenvalue problem *Comm. ACM* **16** 41–44

Wilkinson J H 1977 Some recent advances in numerical linear algebra *The State of the Art in Numerical Analysis* (ed D A H Jacobs) Academic Press

5 Parameters

5.1 Compulsory Input Parameters

1: **ma – int32 scalar**

m_A, the number of superdiagonals within the band of A. Normally $m_A \ll n$.

Constraint: $0 \leq \mathbf{ma} \leq \mathbf{n} - 1$.

2: **a(lda,n) – double array**

lda, the first dimension of the array, must be at least $\mathbf{ma} + 1$.

The upper triangle of the n by n symmetric band matrix A, with the diagonal of the matrix stored in the $(m_A + 1)$th row of the array, and the m_A superdiagonals within the band stored in the first m_A rows of the array. Each column of the matrix is stored in the corresponding column of the array. For example, if $n = 6$ and $m = 2$, the storage space is

$$\begin{array}{cccccc} * & * & a_{13} & a_{24} & a_{35} & a_{46} \\ * & a_{12} & a_{23} & a_{34} & a_{45} & a_{56} \\ a_{11} & a_{22} & a_{33} & a_{44} & a_{55} & a_{66} \end{array}$$

Elements in the top left corner of the array need not be set. The following code assigns the matrix elements within the band to the correct elements of the array:

```
for j=1:n
  for i=max(1,j-ma+1+1):j
    a(i-j+ma+1,j) = matrix(i,j);
  end
end
```

3: **mb – int32 scalar**

m_B, the number of superdiagonals within the band of B.

Constraint: $0 \leq \mathbf{mb} \leq \mathbf{ma}$.

4: **b(ldb,n) – double array**

ldb, the first dimension of the array, must be at least $\mathbf{mb} + 1$.

The upper triangle of the n by n symmetric positive-definite band matrix B, with the diagonal of the matrix stored in the $(m_B + 1)$th row of the array, and the m_B superdiagonals within the band stored in the first m_B rows of the array. Each column of the matrix is stored in the corresponding column of the array.

5.2 Optional Input Parameters

1: **n – int32 scalar**

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

n, the order of the matrices A and B.

Constraint: $\mathbf{n} \geq 1$.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldb, work, lwork

5.4 Output Parameters

1: **a(lda,n) – double array**

A contains the corresponding elements of C.

2: **b(ldb,n) – double array**

B is overwritten.

3: **d(n) – double array**

The eigenvalues in descending order of magnitude.

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, **n** < 1,
or **ma** < 0,
or **ma** $\geq$ **n**,
or **mb** < 0,
or **mb** $>$ **ma**,
or **lda** $\leq$ **ma**,
or **ldb** $\leq$ **mb**,
or **lwork** $< \max(\mathbf{n}, (3 \times \mathbf{ma} + \mathbf{mb}) \times (\mathbf{ma} + \mathbf{mb} + 1))$.

ifail $= 2$

The matrix B is either not positive-definite or is nearly singular.

ifail $= 3$

This failure is very unlikely to occur, but indicates that more than $30 \times \mathbf{n}$ iterations are required by the QR part of the algorithm. The input parameters should be carefully checked to ensure that the error is not due to an incorrect parameter.

7 Accuracy

The computed eigenvalues will be the exact eigenvalues of a neighbouring problem $(A + E)x = \lambda(B + F)x$, where $\|E\|$ and $\|F\|$ are of the order of $\epsilon c(B)\|A\|$ and $\epsilon c(B)\|B\|$ respectively, where $c(B)$ is the condition number of B with respect to inversion and ϵ is the ***machine precision***.

Thus if B is ill-conditioned with respect to inversion there may be a severe loss of accuracy in well-conditioned eigenvalues.

8 Further Comments

The time taken by f02fh is very approximately proportional to $n^2\left(\frac{m_A + m_B + 2}{m_A} + \frac{m_B^2}{8}\right)$, provided $m_A > 0$.

9 Example

```
ma = int32(2);
a = [0, 0, -1, -1, -1, -1, -1, -1, -1;
     0, 1, 2, 3, 4, 4, 3, 2, 1;
     5, 6, 7, 8, 9, 8, 7, 6, 5];
mb = int32(2);
b = [0, 0, -2, -2, -2, -2, -2, -2, -2;
     0, 2, 1, 1, 1, 1, 1, 1, 1;
     4, 5, 6, 6, 6, 6, 6, 6, 6];
[aOut, bOut, d, ifail] = f02fh(ma, a, mb, b)
```

```
aOut =
  Columns 1 through 7
        0         0         0         0   -0.0000         0         0
        0   -0.0508   -0.4358    1.1129    0.8164   -0.5164   -0.5285
   0.9468    0.9176    0.6206    4.2843    1.2251    2.1952    1.7569
  Columns 8 through 9
        0         0
  -0.1586    0.0520
   0.8251    0.8965
bOut =
  Columns 1 through 7
         0         0   -0.4425   -0.4343   -0.4260   -0.4088   -0.3977
         0    0.8891    0.3847    0.3694    0.3506    0.3229    0.2898
    3.1150    1.0003    4.5196    4.6054    4.6945    4.8920    5.0286
  Columns 8 through 9
   -0.3429   -0.3333
    0.2286    0.1667
    5.8333    6.0000
d =
    0.0544
    0.7578
    0.8277
    0.9188
    0.9429
    1.1667
    1.5582
    2.6623
    4.7791
ifail =
           0
```