NAG Toolbox for MATLAB

f02bj

1 Purpose

f02bj calculates all the eigenvalues and, if required, all the eigenvectors of the generalized eigenproblem $Ax = \lambda Bx$, where A and B are real, square matrices, using the QZ algorithm.

2 Syntax

```
[a, b, alfr, alfi, beta, v, iter, ifail] = f02bj(a, b, eps1, matv, 'n',
n)
```

3 Description

All the eigenvalues and, if required, all the eigenvectors of the generalized eigenproblem $Ax = \lambda Bx$, where A and B are real, square matrices, are determined using the QZ algorithm. The QZ algorithm consists of four stages:

(i) A is reduced to upper Hessenberg form and at the same time B is reduced to upper triangular form.

(ii) A is further reduced to quasi-triangular form while the triangular form of B is maintained.

(iii) The quasi-triangular form of A is reduced to triangular form and the eigenvalues extracted. This function does not actually produce the eigenvalues λ_j, but instead returns α_j and β_j such that

$$\lambda_j = \alpha_j/\beta_j, \qquad j = 1, 2, \ldots, n.$$

The division by β_j becomes the responsibility of your program, since β_j may be zero, indicating an infinite eigenvalue. Pairs of complex eigenvalues occur with α_j/β_j and α_{j+1}/β_{j+1} complex conjugates, even though α_j and α_{j+1} are not conjugate.

(iv) If the eigenvectors are required (**matv** = **true**), they are obtained from the triangular matrices and then transformed back into the original co-ordinate system.

4 References

Moler C B and Stewart G W 1973 An algorithm for generalized matrix eigenproblems *SIAM J. Numer. Anal.* **10** 241–256

Ward R C 1975 The combination shift QZ algorithm *SIAM J. Numer. Anal.* **12** 835–853

Wilkinson J H 1979 Kronecker's canonical form and the QZ algorithm *Linear Algebra Appl.* **28** 285–303

5 Parameters

5.1 Compulsory Input Parameters

1: **a**(**lda**,**n**) – **double array**

lda, the first dimension of the array, must be at least **n**.

The n by n matrix A.

2: **b**(**ldb**,**n**) – **double array**

ldb, the first dimension of the array, must be at least **n**.

The n by n matrix B.

3: **eps1 – double scalar**

The tolerance used to determine negligible elements.

eps1 > 0.0

An element will be considered negligible if it is less than **eps1** times the norm of its matrix.

eps1 ≤ 0.0

machine precision is used in place of **eps1**.

A positive value of **eps1** may result in faster execution but less accurate results.

4: **matv – logical scalar**

Must be set **true** if the eigenvectors are required, otherwise **false**.

5.2 Optional Input Parameters

1: **n – int32 scalar**

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

n, the order of the matrices A and B.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldb, ldv

5.4 Output Parameters

1: **a(lda,n) – double array**

The array is overwritten.

2: **b(ldb,n) – double array**

The array is overwritten.

3: **alfr(n) – double array**

4: **alfi(n) – double array**

The real and imaginary parts of α_j, for $j = 1, 2, \ldots, n$.

5: **beta(n) – double array**

β_j, for $j = 1, 2, \ldots, n$.

6: **v(ldv,n) – double array**

If **matv** $=$ **true**,

if the jth eigenvalue is real, the jth column of **v** contains its eigenvector;
if the jth and $(j+1)$th eigenvalues form a complex pair, the jth and $(j+1)$th columns of **v** contain the real and imaginary parts of the eigenvector associated with the first eigenvalue of the pair. The conjugate of this vector is the eigenvector for the conjugate eigenvalue.

Each eigenvector is normalized so that the component of largest modulus is real and the sum of squares of the moduli equal one.

If **matv** $=$ **false**, **v** is not used.

7: **iter(n) – int32 array**

iter(j) contains the number of iterations needed to obtain the jth eigenvalue. Note that the eigenvalues are obtained in reverse order, starting with the nth.

8: **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 $= i$

More than $30 \times$ **n** iterations are required to determine all the diagonal 1 by 1 or 2 by 2 blocks of the quasi-triangular form in the second step of the QZ algorithm. **ifail** is set to the index i of the eigenvalue at which this failure occurs. If the soft failure option is used, α_j and β_j are correct for $j = i+1, i+2, \ldots, n$, but **v** does not contain any correct eigenvectors.

7 Accuracy

The computed eigenvalues are always exact for a problem $(A+E)x = \lambda(B+F)x$, where $\|E\|/\|A\|$ and $\|F\|/\|B\|$ are both of the order of $\max(\textbf{eps1}, \epsilon)$, ϵ being the ***machine precision***.

Note: interpretation of results obtained with the QZ algorithm often requires a clear understanding of the effects of small changes in the original data. These effects are reviewed in Wilkinson 1979, in relation to the significance of small values of α_j and β_j. It should be noted that if α_j and β_j are **both** small for any j, it may be that no reliance can be placed on **any** of the computed eigenvalues $\lambda_i = \alpha_i/\beta_i$. You are recommended to study Wilkinson 1979 and, if in difficulty, to seek expert advice on determining the sensitivity of the eigenvalues to perturbations in the data.

8 Further Comments

The time taken by f02bj is approximately proportional to n^3 and also depends on the value chosen for parameter **eps1**.

9 Example

```
a = [3.9, 12.5, -34.5, -0.5;
     4.3, 21.5, -47.5, 7.5;
     4.3, 21.5, -43.5, 3.5;
     4.4, 26, -46, 6];
b = [1, 2, -3, 1;
     1, 3, -5, 4;
     1, 3, -4, 3;
     1, 3, -4, 4];
eps1 = 1.111307226797642e-16;
matv = true;
[aOut, bOut, alfr, alfi, beta, v, iter, ifail] = f02bj(a, b, eps1, matv)
```

```
aOut =
    3.8009   -6.8540  -68.6636   66.8359
         0    5.5419   -5.5723    7.2621
         0    1.4267    0.9392   -3.3499
         0         0         0    4.0000
bOut =
    1.0000   -5.1771   -2.0907    7.5153
         0   -1.3317    0.3405    0.0175
         0         0    1.0000   -0.0408
    0.0000         0         0    1.0000
```

```
alfr =
    3.8009
    1.5629
    3.0301
    4.0000
alfi =
         0
    2.0839
   -4.0401
         0
beta =
    1.9005
    0.5210
    1.0100
    1.0000
v =
    0.9961    0.9449         0    0.9875
    0.0057    0.1890    0.0000    0.0110
    0.0626    0.1134   -0.1512   -0.0329
    0.0626    0.1134   -0.1512    0.1536
iter =
           0
           0
           5
           0
ifail =
           0
```