NAG Toolbox for MATLAB

f08ts

1 Purpose

f08ts reduces a complex Hermitian-definite generalized eigenproblem $Az = \lambda Bz$, $ABz = \lambda z$ or $BAz = \lambda z$ to the standard form $Cy = \lambda y$, where A is a complex Hermitian matrix and B has been factorized by f07gr, using packed storage.

2 Syntax

```
[ap, info] = f08ts(itype, uplo, n, ap, bp)
```

3 Description

To reduce the complex Hermitian-definite generalized eigenproblem $Az = \lambda Bz$, $ABz = \lambda z$ or $BAz = \lambda z$ to the standard form $Cy = \lambda y$ using packed storage, f08ts must be preceded by a call to f07gr which computes the Cholesky factorization of B; B must be positive-definite.

The different problem types are specified by the parameter **itype**, as indicated in the table below. The table shows how C is computed by the function, and also how the eigenvectors z of the original problem can be recovered from the eigenvectors of the standard form.

itype	Problem	**uplo**	B	C	z
1	$Az = \lambda Bz$	'U' 'L'	$U^{\mathrm{H}}U$ LL^{H}	$U^{-\mathrm{H}}AU^{-1}$ $L^{-1}AL^{-\mathrm{H}}$	$U^{-1}y$ $L^{-\mathrm{H}}y$
2	$ABz = \lambda z$	'U' 'L'	$U^{\mathrm{H}}U$ LL^{H}	UAU^{H} $L^{\mathrm{H}}AL$	$U^{-1}y$ $L^{-\mathrm{H}}y$
3	$BAz = \lambda z$	'U' 'L'	$U^{\mathrm{H}}U$ LL^{H}	UAU^{H} $L^{\mathrm{H}}AL$	$U^{\mathrm{H}}y$ Ly

4 References

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 – int32 scalar**

Indicates how the standard form is computed.

itype $= 1$

if **uplo** $=$ 'U', $C = U^{-\mathrm{H}}AU^{-1}$;
if **uplo** $=$ 'L', $C = L^{-1}AL^{-\mathrm{H}}$.

itype $= 2$ or 3

if **uplo** $=$ 'U', $C = UAU^{\mathrm{H}}$;
if **uplo** $=$ 'L', $C = L^{\mathrm{H}}AL$.

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

2: **uplo – string**

Indicates whether the upper or lower triangular part of A is stored and how B has been factorized.

uplo $=$ 'U'

The upper triangular part of A is stored and $B = U^{\mathrm{H}}U$.

uplo $=$ 'L'

The lower triangular part of A is stored and $B = LL^{\mathrm{H}}$.

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

3: **n – int32 scalar**

n, the order of the matrices A and B.

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

4: **ap**$(*)$ **– complex array**

Note: the dimension of the array **ap** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$.

The n by n Hermitian matrix A, packed by columns.

More precisely,

if **uplo** $=$ 'U', the upper triangle of A must be stored with element A_{ij} in $\mathbf{ap}(i+j(j-1)/2)$ for $i \leq j$;
if **uplo** $=$ 'L', the lower triangle of A must be stored with element A_{ij} in $\mathbf{ap}(i+(2n-j)(j-1)/2)$ for $i \geq j$.

5: **bp**$(*)$ **– complex array**

Note: the dimension of the array **bp** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$.

The Cholesky factor of B as specified by **uplo** and returned by f07gr.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **ap**$(*)$ **– complex array**

Note: the dimension of the array **ap** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$.

The upper or lower triangle of **ap** contains the corresponding upper or lower triangle of C as specified by **itype** and **uplo**, using the same packed storage format as described above.

2: **info – int32 scalar**

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

6 Error Indicators and Warnings

Errors or warnings detected by the function:

info $= -i$

If **info** $= -i$, parameter i had an illegal value on entry. The parameters are numbered as follows: 1: **itype**, 2: **uplo**, 3: **n**, 4: **ap**, 5: **bp**, 6: **info**.

7 Accuracy

Forming the reduced matrix C is a stable procedure. However it involves implicit multiplication by B^{-1} if (**itype** $= 1$) or B (if **itype** $= 2$ or 3). When f08ts is used as a step in the computation of eigenvalues and eigenvectors of the original problem, there may be a significant loss of accuracy if B is ill-conditioned with respect to inversion. See the document for f08sn for further details.

8 Further Comments

The total number of real floating-point operations is approximately $4n^3$.

The real analogue of this function is f08te.

9 Example

```
itype = int32(1);
uplo = 'L';
n = int32(4);
ap = [complex(-7.36, +0);
     complex(0.77, +0.43);
     complex(-0.64, +0.92);
     complex(3.01, +6.97);
     complex(3.49, +0);
     complex(2.19, -4.45);
     complex(1.9, -3.73);
     complex(0.12, +0);
     complex(2.88, +3.17);
     complex(-2.54, +0)];
bp = [complex(1.797220075561143, +0);
     complex(0.8401864749527325, +1.068316577423342);
     complex(1.057188279741849, -0.467388502622712);
     complex(0.233694251311356, -1.391037210186643);
     complex(1.316353439509685, +0);
     complex(-0.4701749470106329, +0.3130658155999466);
     complex(0.08335250923944192, +0.03676071443037458);
     complex(1.560392977137124, +0);
     complex(0.9359617337923402, +0.9899692192815736);
     complex(0.6603332973655893, +0)];
[apOut, info] = f08ts(itype, uplo, n, ap, bp)
```

```
apOut =
  -2.2786
   1.7799 + 2.0310i
   2.2594 - 0.0996i
  -0.1206 - 2.5286i
  -1.1255
   0.0090 - 0.4261i
  -1.0602 - 0.8600i
  -0.3715
   2.3103 + 0.9198i
  -0.7133
info =
           0
```