NAG Toolbox for MATLAB

f08yu

1 Purpose

f08yu reorders the generalized Schur factorization of a complex matrix pair in generalized Schur form, so that a selected cluster of eigenvalues appears in the leading elements on the diagonal of the generalized Schur form. The function also, optionally, computes the reciprocal condition numbers of the cluster of eigenvalues and/or corresponding deflating subspaces.

2 Syntax

```
[a, b, alpha, beta, q, z, m, pr, pl, dif, info] = f08yu(ijob, wantq,
wantz, select, a, b, q, z, 'n', n)
```

3 Description

f08yu factorizes the generalized complex n by n matrix pair (S,T) in generalized Schur form, using a unitary equivalence transformation as

$$S = \hat{Q}\hat{S}\hat{Z}^{\mathrm{H}}, \qquad T = \hat{Q}\hat{T}\hat{Z}^{\mathrm{H}},$$

where $\left(\hat{S},\hat{T}\right)$ are also in generalized Schur form and have the selected eigenvalues as the leading diagonal elements. The leading columns of Q and Z are the generalized Schur vectors corresponding to the selected eigenvalues and form orthonormal subspaces for the left and right eigenspaces (deflating subspaces) of the pair (S,T).

The pair (S,T) are in generalized Schur form if S and T are upper triangular as returned, for example, by f08xn, or f08xs with **job** = 'S'. The diagonal elements define the generalized eigenvalues (α_i,β_i), for $i = 1,2,\ldots,n$ of the pair (S,T). The eigenvalues are given by

$$\lambda_i = \alpha_i/\beta_i,$$

but are returned as the pair (α_i,β_i) in order to avoid possible overflow in computing λ_i. Optionally, the function returns reciprocals of condition number estimates for the selected eigenvalue cluster, p and q, the right and left projection norms, and of deflating subspaces, Dif_u and Dif_l. For more information see Sections 2.4.8 and 4.11 of Anderson *et al.* 1999.

If S and T are the result of a generalized Schur factorization of a matrix pair (A,B)

$$A = QSZ^{\mathrm{H}}, \qquad B = QTZ^{\mathrm{H}}$$

then, optionally, the matrices Q and Z can be updated as $Q\hat{Q}$ and $Z\hat{Z}$. Note that the condition numbers of the pair (S,T) are the same as those of the pair (A,B).

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 URL: http://www.netlib.org/lapack/lug

5 Parameters

5.1 Compulsory Input Parameters

1: **ijob – int32 scalar**

Specifies whether condition numbers are required for the cluster of eigenvalues (p and q) or the deflating subspaces (Dif_u and Dif_l).

ijob $= 0$

Only reorder with respect to **select**. No extras.

ijob $= 1$

Reciprocal of norms of 'projections' onto left and right eigenspaces with respect to the selected cluster (p and q).

ijob $= 2$

The upper bounds on Dif_u and Dif_l. F-norm-based estimate (**dif**$(1:2)$).

ijob $= 3$

Estimate of Dif_u and Dif_l. 1-norm-based estimate (**dif**$(1:2)$). About five times as expensive as **ijob** $= 2$.

ijob $= 4$

Compute **pl**, **pr** and **dif** as in **ijob** $= 0$, 1 and 2. Economic version to get it all.

ijob $= 5$

Compute **pl**, **pr** and **dif** as in **ijob** $= 0$, 1 and 3.

2: **wantq – logical scalar**

If **wantq** $=$ **true**, update the left transformation matrix Q.

If **wantq** $=$ **false**, do not update Q.

3: **wantz – logical scalar**

If **wantz** $=$ **true**, update the right transformation matrix Z.

If **wantz** $=$ **false**, do not update Z.

4: **select**$(*)$ **– logical array**

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

Specifies the eigenvalues in the selected cluster. To select an eigenvalue λ_j, **select**(j) must be set to **true**.

5: **a**(**lda,***$*$) **– complex 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 matrix S in the pair (S,T).

6: **b**(**ldb,***$*$) **– complex 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 matrix T, in the pair (S,T).

7: **q**(**ldq,***$*$) **– complex array**

The first dimension, **ldq**, of the array **q** must satisfy

if **wantq** $=$ **true**, **ldq** $\geq \max(1,\mathbf{n})$;
ldq ≥ 1 otherwise.

The second dimension of the array must be at least $\max(1,\mathbf{n})$ if **wantq** $=$ **true**, and at least 1 otherwise

If **wantq** = **true**, the n by n matrix Q.

8: **z(ldz,∗) – complex array**

The first dimension, **ldz**, of the array **z** must satisfy

if **wantz** = **true**, **ldz** $\geq \max(1, \mathbf{n})$;
ldz ≥ 1 otherwise.

The second dimension of the array must be at least $\max(1, \mathbf{n})$ if **wantz** = **true**, and at least 1 otherwise

If **wantz** = **true**, the n by n matrix Z.

5.2 Optional Input Parameters

1: **n – int32 scalar**

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 S and T.

Constraint: **n** ≥ 0.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldb, ldq, ldz, work, lwork, iwork, liwork

5.4 Output Parameters

1: **a(lda,∗) – complex 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 updated matrix $\hat{S}$.

2: **b(ldb,∗) – complex 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 updated matrix $\hat{T}$

3: **alpha(∗) – complex array**

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

See the description of **beta**.

4: **beta(∗) – complex array**

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

alpha and **beta** contain diagonal elements of $\hat{S}$ and $\hat{T}$, respectively, when the pair (S, T) has been reduced to generalized Schur form. **alpha**(i)/**beta**(i), for $i = 1, \ldots, \mathbf{n}$, are the eigenvalues.

5: **q(ldq,∗) – complex array**

The first dimension, **ldq**, of the array **q** must satisfy

if **wantq** = **true**, **ldq** $\geq \max(1, \mathbf{n})$;
ldq ≥ 1 otherwise.

The second dimension of the array must be at least $\max(1, \mathbf{n})$ if **wantq** $=$ **true**, and at least 1 otherwise

If **wantq** $=$ **true**, the updated matrix $Q\hat{Q}$.

If **wantq** $=$ **false**, **q** is not referenced.

6: **z**(**ldz**,$*$) – **complex array**

The first dimension, **ldz**, of the array **z** must satisfy

if **wantz** $=$ **true**, **ldz** $\geq \max(1, \mathbf{n})$;
ldz ≥ 1 otherwise.

The second dimension of the array must be at least $\max(1, \mathbf{n})$ if **wantz** $=$ **true**, and at least 1 otherwise

If **wantz** $=$ **true**, the updated matrix $Z\hat{Z}$.

If **wantz** $=$ **false**, **z** is not referenced.

7: **m** – **int32 scalar**

The dimension of the specified pair of left and right eigenspaces (deflating subspaces).

8: **pr** – **double scalar**

9: **pl** – **double scalar**

If **ijob** $= 1$, 4 or 5, **pl** and **pr** are lower bounds on the reciprocal of the norm of 'projections' p and q onto left and right eigenspace with respect to the selected cluster. $0 <$ **pl**, **pr** ≤ 1.

If **m** $= 0$ or **m** $=$ **n**, **pl** $=$ **pr** $= 1$.

If **ijob** $= 0$, 2 or 3, **pl** and **pr** are not referenced.

10: **dif**($*$) – **double array**

Note: the dimension of the array **dif** must be at least 2.

If **ijob** ≥ 2, **dif**$(1:2)$ store the estimates of Dif_u and Dif_l.

If **ijob** $= 2$ or 4, **dif**$(1:2)$ are F-norm-based upper bounds on Dif_u and Dif_l.

If **ijob** $= 3$ or 5, **dif**$(1:2)$ are 1-norm-based estimates of Dif_u and Dif_l.

If **m** $= 0$ or **n**, **dif**$(1:2) = \|(A, B)\|_F$.

If **ijob** $= 0$ or 1, **dif** is not referenced.

11: **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: **ijob**, 2: **wantq**, 3: **wantz**, 4: **select**, 5: **n**, 6: **a**, 7: **lda**, 8: **b**, 9: **ldb**, 10: **alpha**, 11: **beta**, 12: **q**, 13: **ldq**, 14: **z**, 15: **ldz**, 16: **m**, 17: **pr**, 18: **pl**, 19: **dif**, 20: **work**, 21: **lwork**, 22: **iwork**, 23: **liwork**, 24: **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.

info $= 1$

Reordering of (S, T) failed because the transformed matrix pair $\left(\hat{S}, \hat{T}\right)$ would be too far from generalized Schur form; the problem is very ill-conditioned. (S, T) may have been partially reordered. If requested, 0 is returned in **dif**$(1:2)$, **pl** and **pr**.

7 Accuracy

The computed generalized Schur form is nearly the exact generalized Schur form for nearby matrices $(S + E)$ and $(T + F)$, where

$$\|E\|_2 = O\epsilon\|S\|_2 \qquad \text{and} \qquad \|F\|_2 = O\epsilon\|T\|_2,$$

and ϵ is the ***machine precision***. See Section 4.11 of Anderson *et al.* 1999 for further details of error bounds for the generalized nonsymmetric eigenproblem, and for information on the condition numbers returned.

8 Further Comments

The real analogue of this function is f08yg.

9 Example

```
ijob = int32(4);
wantq = true;
wantz = true;
select = [false;
     true;
     true;
     false];
a = [complex(4, +4), complex(1, +1), complex(1, +1), complex(2, -1);
     complex(0, +0), complex(2, +1), complex(1, +1), complex(1, +1);
     complex(0, +0), complex(0, +0), complex(2, -1), complex(1, +1);
     complex(0, +0), complex(0, +0), complex(0, +0), complex(6, -2)];
b = [complex(2, +0), complex(1, +1), complex(1, +1), complex(3, -1);
     complex(0, +0), complex(1, +0), complex(2, +1), complex(1, +1);
     complex(0, +0), complex(0, +0), complex(1, +0), complex(1, +1);
     complex(0, +0), complex(0, +0), complex(0, +0), complex(2, +0)];
q = [complex(1, +0), complex(0, +0), complex(0, +0), complex(0, +0);
     complex(0, +0), complex(1, +0), complex(0, +0), complex(0, +0);
     complex(0, +0), complex(0, +0), complex(1, +0), complex(0, +0);
     complex(0, +0), complex(0, +0), complex(0, +0), complex(1, +0)];
z = [complex(1, +0), complex(0, +0), complex(0, +0), complex(0, +0);
     complex(0, +0), complex(1, +0), complex(0, +0), complex(0, +0);
     complex(0, +0), complex(0, +0), complex(1, +0), complex(0, +0);
     complex(0, +0), complex(0, +0), complex(0, +0), complex(1, +0)];
[aOut, bOut, alpha, beta, qOut, zOut, m, pr, pl, dif, info] = ...
    f08yu(ijob, wantq, wantz, select, a, b, q, z)
```

```
aOut =
    4.6904 + 2.3452i   -2.1563 + 0.1192i    1.9599 - 0.5174i    1.8091 -
1.2060i
         0               2.0084 - 1.0042i    0.9161 - 0.2762i    1.8574 -
0.5326i
         0                    0              1.6985 + 1.6985i    0.1270 +
0.7231i
         0                    0                   0              6.0000 -
2.0000i
bOut =
    2.3452              -0.6181 + 1.8237i    0.9290 + 0.5409i    2.7136 -
1.5076i
         0               1.0042              1.2251 - 1.1857i    1.8541 -
0.2929i
         0                    0              0.8492              0.1435 +
```

```
0.9053i
        0                   0                   0             2.0000
alpha =
   4.6904 + 2.3452i
   2.0084 - 1.0042i
   1.6985 + 1.6985i
   6.0000 - 2.0000i
beta =
    2.3452
    1.0042
    0.8492
    2.0000
qOut =
   0.9045 + 0.3015i  -0.0033 - 0.2397i   0.0166 - 0.1822i        0
   0.3015             0.2497 + 0.7157i   0.1325 + 0.5630i        0
        0             0.0549 + 0.6042i   0.0110 - 0.7949i        0
        0                   0                   0             1.0000
zOut =
   0.7071            -0.5607             0.0390 - 0.4290i        0
   0.7071             0.5607            -0.0390 + 0.4290i        0
        0             0.0552 + 0.6067i   0.7930                  0
        0                   0                   0             1.0000
m =
           2
pr =
    0.1123
pl =
    0.1425
dif =
    0.2175
    0.2623
info =
           0
```