NAG Toolbox

nag_matop_complex_gen_matrix_frcht_log (f01kk)

1 Purpose

nag_matop_complex_gen_matrix_frcht_log (f01kk) computes the Fréchet derivative $L(A,E)$ of the matrix logarithm of the complex n by n matrix A applied to the complex n by n matrix E. The principal matrix logarithm $\log(A)$ is also returned.

2 Syntax

```
[a, e, ifail] = nag_matop_complex_gen_matrix_frcht_log(a, e, 'n', n)
[a, e, ifail] = f01kk(a, e, 'n', n)
```

3 Description

For a matrix with no eigenvalues on the closed negative real line, the principal matrix logarithm $\log(A)$ is the unique logarithm whose spectrum lies in the strip $\{z : -\pi < \mathrm{Im}(z) < \pi\}$.

The Fréchet derivative of the matrix logarithm of A is the unique linear mapping $E \mapsto L(A,E)$ such that for any matrix E

$$\log(A+E) - \log(A) - L(A,E) = o(\|E\|).$$

The derivative describes the first order effect of perturbations in A on the logarithm $\log(A)$.

nag_matop_complex_gen_matrix_frcht_log (f01kk) uses the algorithm of Al–Mohy *et al.* (2012) to compute $\log(A)$ and $L(A,E)$. The principal matrix logarithm $\log(A)$ is computed using a Schur decomposition, a Padé approximant and the inverse scaling and squaring method. The Padé approximant is then differentiated in order to obtain the Fréchet derivative $L(A,E)$. If A is nonsingular but has negative real eigenvalues, the principal logarithm is not defined, but nag_matop_complex_gen_matrix_frcht_log (f01kk) will return a non-principal logarithm and Fréchet derivative.

4 References

Al–Mohy A H and Higham N J (2011) Improved inverse scaling and squaring algorithms for the matrix logarithm *SIAM J. Sci. Comput.* **34(4)** C152–C169

Al–Mohy A H, Higham N J and Relton S D (2012) Computing the Fréchet derivative of the matrix logarithm and estimating the condition number *MIMS EPrint* **2012.72**

Higham N J (2008) *Functions of Matrices: Theory and Computation* SIAM, Philadelphia, PA, USA

5 Parameters

5.1 Compulsory Input Parameters

1: **a**(lda, :) – COMPLEX (KIND=nag_wp) array

The first dimension of the array **a** must be at least **n**.

The second dimension of the array **a** must be at least **n**.

The n by n matrix A.

2: **e**(lde, :) – COMPLEX (KIND=nag_wp) array

The first dimension of the array **e** must be at least **n**.

The second dimension of the array **e** must be at least **n**.

The n by n matrix E

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the first dimension of the arrays **a**, **e** and the second dimension of the arrays **a**, **e**. (An error is raised if these dimensions are not equal.)

n, the order of the matrix A.

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

5.3 Output Parameters

1: $\mathbf{a}(lda,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **a** will be **n**.

The second dimension of the array **a** will be **n**.

The n by n principal matrix logarithm, $\log(A)$. Alterntively, if $\mathbf{ifail} = 2$, a non-principal logarithm is returned.

2: $\mathbf{e}(lde,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **e** will be **n**.

The second dimension of the array **e** will be **n**.

With $\mathbf{ifail} = 0$, 2 or 3, the Fréchet derivative $L(A, E)$

3: **ifail** – INTEGER

$\mathbf{ifail} = 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

$\mathbf{ifail} = 1$

A is singular so the logarithm cannot be computed.

$\mathbf{ifail} = 2$

A has eigenvalues on the negative real line. The principal logarithm is not defined in this case, so a non-principal logarithm was returned.

$\mathbf{ifail} = 3$

$\log(A)$ has been computed using an IEEE double precision Padé approximant, although the arithmetic precision is higher than IEEE double precision.

$\mathbf{ifail} = 4$

An unexpected internal error occurred. This failure should not occur and suggests that the function has been called incorrectly.

$\mathbf{ifail} = -1$

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

ifail $= -3$

Constraint: $lda \geq$ **n**.

ifail $= -5$

Constraint: $lde \geq$ **n**.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

For a normal matrix A (for which $A^{\mathrm{H}}A = AA^{\mathrm{H}}$), the Schur decomposition is diagonal and the computation of the matrix logarithm reduces to evaluating the logarithm of the eigenvalues of A and then constructing $\log(A)$ using the Schur vectors. This should give a very accurate result. In general, however, no error bounds are available for the algorithm. The sensitivity of the computation of $\log(A)$ and $L(A, E)$ is worst when A has an eigenvalue of very small modulus or has a complex conjugate pair of eigenvalues lying close to the negative real axis. See Al–Mohy and Higham (2011), Al–Mohy *et al.* (2012) and Section 11.2 of Higham (2008) for details and further discussion.

8 Further Comments

The cost of the algorithm is $O(n^3)$ floating-point operations. The complex allocatable memory required is approximately $5n^2$; see Al–Mohy *et al.* (2012) for further details.

If the matrix logarithm alone is required, without the Fréchet derivative, then nag_matop_complex_gen_matrix_log (f01fj) should be used. If the condition number of the matrix logarithm is required then nag_matop_complex_gen_matrix_cond_log (f01kj) should be used. The real analogue of this function is nag_matop_real_gen_matrix_frcht_log (f01jk).

9 Example

This example finds the principal matrix logarithm $\log(A)$ and the Fréchet derivative $L(A, E)$, where

$$A = \begin{pmatrix} 1+4i & 3i & i & 2 \\ 2i & 3 & 1 & 1+i \\ i & 2+i & 2 & i \\ 1+2i & 3+2i & 1+2i & 3+i \end{pmatrix} \quad \text{and} \quad E = \begin{pmatrix} 1 & 1+2i & 2 & 2+i \\ 1+3i & i & 1 & 0 \\ 2i & 4+i & 1 & 1 \\ 1 & 2+2i & 3i & 1 \end{pmatrix}.$$

9.1 Program Text

```
     function f01kk_example

fprintf('f01kk example results\n\n');

% Logarithm of matrix A and Frechet derivative of log(A)E.

a = [ 1+4i    3i     i  2;
        2i  3     1     1+i;
      i     2+ i  2       i;
      1+2i  3+2i  1+2i  3+i];

e = [ 1     1+2i  2     2+i;
```

```
      1+3i      i  1     0;
        2i  4+ i  1     1;
      1     2+2i    3i  1];

[loga, lae, ifail] = f01kk(a,e);

[ifail] = x04da('General', ' ', loga, 'log(A):');
disp(' ');
[ifail] = x04da('General', ' ', lae, 'L_log(A,E):');
```

9.2 Program Results

```
     f01kk example results

 log(A):
             1          2          3          4
 1      1.4188     0.2758    -0.2240     0.4528
        1.2438     1.0040     0.0826    -0.5887

 2      0.2299     1.0702     0.5292     0.1976
        0.4825    -0.3306    -0.0422     0.1532

 3      0.1328     0.9235     0.6051    -0.1211
       -0.0462     0.3060    -0.0973     0.2966

 4      0.4704     1.0779     0.2724     0.9612
       -0.0891     0.0538     0.7627     0.2680

 L_log(A,E):
             1          2          3          4
 1      0.1620    -0.0593    -0.1543     0.5534
       -0.6532     0.8434    -1.3537     0.0869

 2      0.6673     0.0637     0.3421    -0.4639
        0.7351    -0.0911     0.1136    -0.3399

 3     -0.2500     1.4898    -0.1547     0.3319
       -0.0433     0.6186    -0.0495    -0.3078

 4     -0.4004     0.5834    -0.5153     0.4407
       -0.5893    -0.5926     1.4107     0.1236
```