NAG Toolbox

nag_matop_complex_gen_matrix_exp (f01fc)

1 Purpose

nag_matop_complex_gen_matrix_exp (f01fc) computes the matrix exponential, e^A, of a complex n by n matrix A.

2 Syntax

```
[a, ifail] = nag_matop_complex_gen_matrix_exp(a, 'n', n)
[a, ifail] = f01fc(a, 'n', n)
```

3 Description

e^A is computed using a Padé approximant and the scaling and squaring method described in Al–Mohy and Higham (2009).

4 References

Al–Mohy A H and Higham N J (2009) A new scaling and squaring algorithm for the matrix exponential *SIAM J. Matrix Anal.* **31(3)** 970–989

Higham N J (2005) The scaling and squaring method for the matrix exponential revisited *SIAM J. Matrix Anal. Appl.* **26(4)** 1179–1193

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

Moler C B and Van Loan C F (2003) Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later *SIAM Rev.* **45** 3–49

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.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the first dimension of the array **a**.

n, the order of the matrix A.

Constraint: **n** ≥ 0.

5.3 Output Parameters

1: **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**.

With **ifail** $= 0$, the n by n matrix exponential e^A.

2: **ifail** – INTEGER

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

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

The linear equations to be solved for the Padé approximant are singular; it is likely that this function has been called incorrectly.

ifail $= 2$

The linear equations to be solved are nearly singular and the Padé approximant probably has no correct figures; it is likely that this function has been called incorrectly.

ifail $= 3$ (*warning*)

e^A has been computed using an IEEE double precision Padé approximant, although the arithmetic precision is higher than IEEE double precision.

ifail $= 4$

An unexpected internal error has occurred. Please contact NAG.

ifail $= -1$

Constraint: **n** ≥ 0.

ifail $= -3$

Constraint: $lda \geq$ **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 computed matrix, e^A, is guaranteed to be close to the exact matrix, that is, the method is forward stable. No such guarantee can be given for non-normal matrices. See Al–Mohy and Higham (2009) and Section 10.3 of Higham (2008) for details and further discussion.

If estimates of the condition number of the matrix exponential are required then nag_matop_complex_gen_matrix_cond_exp (f01kg) should be used.

8 Further Comments

The integer allocatable memory required is **n**, and the complex allocatable memory required is approximately $6 \times \mathbf{n}^2$.

The cost of the algorithm is $O(n^3)$; see Section 5 of Al–Mohy and Higham (2009). The complex allocatable memory required is approximately $6 \times n^2$.

If the Fréchet derivative of the matrix exponential is required then nag_matop_complex_gen_matrix_frcht_exp (f01kh) should be used.

As well as the excellent book cited above, the classic reference for the computation of the matrix exponential is Moler and Van Loan (2003).

9 Example

This example finds the matrix exponential of the matrix

$$A = \begin{pmatrix} 1+i & 2+i & 2+i & 2+i \\ 3+2i & 1 & 1 & 2+i \\ 3+2i & 2+i & 1 & 2+i \\ 3+2i & 3+2i & 3+2i & 1+i \end{pmatrix}.$$

9.1 Program Text

```
     function f01fc_example

fprintf('f01fc example results\n\n');

a = [ 1 + 1i,  2 + 1i,  2 + 1i,  2 + 1i;
      3 + 2i,  1 + 0i,  1 + 0i,  2 + 1i;
      3 + 2i,  2 + 1i,  1 + 0i,  2 + 1i;
      3 + 2i,  3 + 2i,  3 + 2i,  1 + 1i];

% Compute exp(a)
[expa, ifail] = f01fc(a);

% Display results
[ifail] = x04da('g', ' ', expa, 'Exp(a)');
```

9.2 Program Results

```
     f01fc example results

 Exp(a)
                1           2           3           4
 1   -157.9003  -194.6526  -186.5627  -155.7669
     -754.3717  -555.0507  -475.4533  -520.1876

 2   -206.8899  -225.4985  -212.4414  -186.5627
     -694.7443  -505.3938  -431.0611  -475.4533

 3   -208.7476  -238.4962  -225.4985  -194.6526
     -808.2090  -590.8045  -505.3938  -555.0507

 4   -133.3958  -208.7476  -206.8899  -157.9003
    -1085.5496  -808.2090  -694.7443  -754.3717
```