NAG Library Routine Document

F01ECF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F01ECF computes the matrix exponential, e^A, of a real n by n matrix A.

2 Specification

```
SUBROUTINE F01ECF (N, A, LDA, IFAIL)
INTEGER            N, LDA, IFAIL
REAL (KIND=nag_wp) A(LDA,*)
```

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 Arguments

1: N – INTEGER *Input*

On entry: n, the order of the matrix A.

Constraint: N $\geq$ 0.

2: A(LDA, $*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array A must be at least N.

On entry: the n by n matrix A.

On exit: the n by n matrix exponential e^A.

3: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01ECF is called.

Constraint: LDA $\geq$ N.

4: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

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

IFAIL $= 3$

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 $= -1$

On entry, N $= \langle value \rangle$.
Constraint: N ≥ 0.

IFAIL $= -3$

On entry, LDA $= \langle value \rangle$ and N $= \langle value \rangle$.
Constraint: LDA $\geq$ N.

IFAIL $= -99$

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

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

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

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

For a normal matrix A (for which $A^{\mathrm{T}}A = AA^{\mathrm{T}}$) 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 F01JGF should be used.

8 Parallelism and Performance

F01ECF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F01ECF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The integer allocatable memory required is N, and the real allocatable memory required is approximately $6 \times \mathrm{N}^2$.

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

If the Fréchet derivative of the matrix exponential is required then F01JHF 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).

10 Example

This example finds the matrix exponential of the matrix

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

10.1 Program Text

```
    Program f01ecfe

!     F01ECF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f01ecf, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
```

```
!     .. Local Scalars ..
      Integer                          :: i, ifail, lda, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'F01ECF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n
      lda = n
      Allocate (a(lda,n))
!     Read A from data file
      Read (nin,*)(a(i,1:n),i=1,n)

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
!     Find exp( A )
      Call f01ecf(n,a,lda,ifail)

!     Print solution
      Call x04caf('General',' ',n,n,a,lda,'Exp(A)',ifail)

    End Program f01ecfe
```

10.2 Program Data

```
F01ECF Example Program Data

  4                       : n

  1.0   2.0   2.0   2.0
  3.0   1.0   1.0   2.0
  3.0   2.0   1.0   2.0
  3.0   3.0   3.0   1.0   : a
```

10.3 Program Results

```
 F01ECF Example Program Results

 Exp(A)
              1           2           3           4
 1    740.7038    610.8500    542.2743    549.1753
 2    731.2510    603.5524    535.0884    542.2743
 3    823.7630    679.4257    603.5524    610.8500
 4    998.4355    823.7630    731.2510    740.7038
```