NAG Library Routine Document

F01EJF

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

F01EJF computes the principal matrix logarithm, $\log(A)$, of a real n by n matrix A, with no eigenvalues on the closed negative real line.

2 Specification

```
SUBROUTINE F01EJF (N, A, LDA, IMNORM, IFAIL)

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

3 Description

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

$\log(A)$ is computed using the Schur–Parlett algorithm for the matrix logarithm described in Higham (2008) and Davies and Higham (2003).

4 References

Davies P I and Higham N J (2003) A Schur–Parlett algorithm for computing matrix functions. *SIAM J. Matrix Anal. Appl.* **25(2)** 464–485

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

5 Parameters

1: N – INTEGER *Input*

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

Constraint: $\mathrm{N} \geq 0$.

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 principal matrix logarithm, $\log(A)$.

3: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01EJF is called.

Constraint: $\mathrm{LDA} \geq \max(1, \mathrm{N})$.

4: IMNORM – REAL (KIND=nag_wp) *Output*

On exit: if A has complex eigenvalues, F01EJF will use complex arithmetic to compute $\log(A)$. The imaginary part is discarded at the end of the computation, because it will theoretically vanish. IMNORM contains the 1-norm of the imaginary part, which should be used to check that the routine has given a reliable answer.

If A has real eigenvalues, F01EJF uses real arithmetic and IMNORM $= 0$.

5: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction 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 parameter, 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 $= 1$

A is singular so the logarithm cannot be computed.

IFAIL $= 2$

A was found to have eigenvalues on the negative real line.
The principal logarithm is not defined in this case, F01FJF can be used to find a complex non-principal logarithm.

IFAIL $= 3$

The arithmetic precision is higher than that used for the Padé approximant computed matrix logarithm.

IFAIL $= 4$

An unexpected internal error 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 $= -999$

Allocation of memory failed.
The real allocatable memory required is approximately $3\mathrm{N}^2$.

7 Accuracy

For a normal matrix A (for which $A^{\mathrm{T}}A = AA^{\mathrm{T}}$), the Schur decomposition is diagonal and the algorithm 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. See Section 9.4 of Higham (2008) for details and further discussion.

For discussion of the condition of the matrix logarithm see Section 11.2 of Higham (2008). In particular, the condition number of the matrix logarithm at A, $\kappa_{\log}(A)$, which is a measure of the sensitivity of the computed logarithm to perturbations in the matrix A, satisfies

$$\kappa_{\log}(A) \geq \frac{\kappa(A)}{\|\log(A)\|},$$

where $\kappa(A)$ is the condition number of A. Further, the sensitivity of the computation of $\log(A)$ 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.

8 Further Comments

If A has real eigenvalues then up to $4n^2$ of real allocatable memory may be required. Otherwise up to $4n^2$ of complex allocatable memory may be required.

The cost of the algorithm is $O(n^3)$ floating point operations. The exact cost depends on the eigenvalue distribution of A; see Algorithm 11.11 of Higham (2008).

If estimates of the condition number of the matrix logarithm are required then F01JAF should be used.

F01FJF can be used to find the principal logarithm of a complex matrix. It can also be used to return a complex, non-principal logarithm if a real matrix has no principal logarithm due to the presence of negative eigenvalues.

9 Example

This example finds the principal matrix logarithm of the matrix

$$A = \begin{pmatrix} 3 & -3 & 1 & 1 \\ 2 & 1 & -2 & 1 \\ 1 & 1 & 3 & -1 \\ 2 & 0 & 2 & 0 \end{pmatrix}.$$

9.1 Program Text

```
    Program f01ejfe

!     F01EJF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f01ejf, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)              :: imnorm
      Integer                         :: i, ifail, lda, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'F01EJF Example Program Results'
      Write (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)

!     Find log( A )
      ifail = 0
      Call f01ejf(n,a,lda,imnorm,ifail)

!     Print solution
      ifail = 0
      Call x04caf('G','N',n,n,a,lda,'log(A)',ifail)

!     Print the norm of the imaginary part to check it is small
      Write (nout,*)
      Write (nout,Fmt='(1X,A,F6.2)') 'Imnorm =', imnorm

    End Program f01ejfe
```

9.2 Program Data

```
F01EJF Example Program Data

4                                :Value of N

3.0    -3.0     1.0     1.0
2.0     1.0    -2.0     1.0
1.0     1.0     3.0    -1.0
2.0     0.0     2.0     0.0  :End of matrix A
```

9.3 Program Results

```
 F01EJF Example Program Results

 log(A)
             1          2          3          4
 1      1.1957    -1.2076    -0.5802     1.0872
 2      0.8464     1.0133    -0.5985    -0.1641
 3      0.4389     0.6701     1.8449    -1.2111
 4      1.2792     0.6177     2.1448    -1.9743

 Imnorm =  0.00
```