NAG Library Routine Document

F01JJF

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

F01JJF computes an estimate of the relative condition number $\kappa_{\log}(A)$ of the logarithm of a real n by n matrix A, in the 1-norm. The principal matrix logarithm $\log(A)$ is also returned.

2 Specification

```
SUBROUTINE F01JJF (N, A, LDA, CONDLA, IFAIL)
INTEGER            N, LDA, IFAIL
REAL (KIND=nag_wp) A(LDA,*), CONDLA
```

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 $\log(A)$.

The relative condition number of the matrix logarithm can be defined by

$$\kappa_{\log}(A) = \frac{\|L(A)\|\|A\|}{\|\log(A)\|},$$

where $\|L(A)\|$ is the norm of the Fréchet derivative of the matrix logarithm at A.

To obtain the estimate of $\kappa_{\log}(A)$, F01JJF first estimates $\|L(A)\|$ by computing an estimate γ of a quantity $K \in \left[n^{-1}\|L(A)\|_1, n\|L(A)\|_1\right]$, such that $\gamma \leq K$.

The algorithms used to compute $\kappa_{\log}(A)$ and $\log(A)$ are based on a Schur decomposition, the inverse scaling and squaring method and Padé approximants. Further details can be found in Al–Mohy and Higham (2011) and Al–Mohy *et al.* (2012).

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

1: N – INTEGER *Input*

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

Constraint: N ≥ 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 F01JJF is called.

Constraint: LDA $\geq$ N.

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

On exit: with IFAIL $= 0$ or 3, an estimate of the relative condition number of the matrix logarithm, $\kappa_{\log}(A)$. Alternatively, if IFAIL $= 4$, contains the absolute condition number of the matrix logarithm.

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 has eigenvalues on the negative real line. The principal logarithm is not defined in this case; F01KJF can be used to return a complex, non-principal log.

IFAIL $= 3$

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

IFAIL $= 4$

The relative condition number is infinite. The absolute condition number was returned instead.

IFAIL $= 5$

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

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 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

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

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

F01JJF uses the norm estimation routine F04YDF to produce an estimate γ of a quantity $K \in \left[n^{-1}\|L(A)\|_1, n\|L(A)\|_1\right]$, such that $\gamma \leq K$. For further details on the accuracy of norm estimation, see the documentation for F04YDF.

For a normal matrix A (for which $A^{\mathrm{T}}A = AA^{\mathrm{T}}$), 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 $\log(A)$ 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) and Section 11.2 of Higham (2008) for details and further discussion.

8 Parallelism and Performance

F01JJF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F01JJF 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

F01JAF uses a similar algorithm to F01JJF to compute an estimate of the *absolute* condition number (which is related to the relative condition number by a factor of $\|A\|/\|\log(A)\|$). However, the required Fréchet derivatives are computed in a more efficient and stable manner by F01JJF and so its use is recommended over F01JAF.

The amount of real allocatable memory required by the algorithm is typically of the order $10n^2$.

The cost of the algorithm is $O(n^3)$ floating-point operations; see Al–Mohy *et al.* (2012).

If the matrix logarithm alone is required, without an estimate of the condition number, then F01EJF should be used. If the Fréchet derivative of the matrix logarithm is required then F01JKF should be used. If A has negative real eigenvalues then F01KJF can be used to return a complex, non-principal matrix logarithm and its condition number.

10 Example

This example estimates the relative condition number of the matrix logarithm $\log(A)$, where

$$A = \begin{pmatrix} 4 & -1 & 0 & 1 \\ 2 & 5 & -2 & 2 \\ 1 & 1 & 3 & -1 \\ 2 & 0 & 2 & 8 \end{pmatrix}.$$

10.1 Program Text

```
    Program f01jjfe

!     F01JJF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: f01jjf, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: condla
      Integer                          :: i, ifail, lda, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'F01JJF 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)

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0

!     Find log( A )
      Call f01jjf(n,a,lda,condla,ifail)

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

      Write (nout,*)
      Write (nout,99999) 'Estimated condition number is: ', condla
99999 Format (1X,A,F6.2)
    End Program f01jjfe
```

10.2 Program Data

```
F01JJF Example Program Data

4                           :Value of N

4.0    -1.0     0.0     1.0
2.0     5.0    -2.0     2.0
1.0     1.0     3.0    -1.0
2.0     0.0     2.0     8.0  :End of matrix A
```

10.3 Program Results

```
 F01JJF Example Program Results

 log(A)
             1          2          3          4
 1      1.4081    -0.2051    -0.1071     0.1904
 2      0.4396     1.7096    -0.5147     0.2226
 3      0.2560     0.2613     1.2485    -0.2413
 4      0.3030    -0.0107     0.3834     2.0891

 Estimated condition number is:   5.50
```