NAG Library Routine Document

F01JFF

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

F01JFF computes the Fréchet derivative $L(A,E)$ of the pth power (where p is real) of the real n by n matrix A applied to the real n by n matrix E. The principal matrix power A^p is also returned.

2 Specification

```
SUBROUTINE F01JFF (N, A, LDA, E, LDE, P, IFAIL)
INTEGER            N, LDA, LDE, IFAIL
REAL (KIND=nag_wp) A(LDA,*), E(LDE,*), P
```

3 Description

For a matrix A with no eigenvalues on the closed negative real line, A^p ($p \in \mathbb{R}$) can be defined as

$$A^p = \exp(p\log(A))$$

where $\log(A)$ is the principal logarithm of A (the unique logarithm whose spectrum lies in the strip $\{z : -\pi < \mathrm{Im}(z) < \pi\}$).

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

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

The derivative describes the first-order effect of perturbations in A on the matrix power A^p.

F01JFF uses the algorithms of Higham and Lin (2011) and Higham and Lin (2013) to compute A^p and $L(A,E)$. The real number p is expressed as $p = q + r$ where $q \in (-1,1)$ and $r \in \mathbb{Z}$. Then $A^p = A^q A^r$. The integer power A^r is found using a combination of binary powering and, if necessary, matrix inversion. The fractional power A^q is computed using a Schur decomposition, a Padé approximant and the scaling and squaring method. The Padé approximant is differentiated in order to obtain the Fréchet derivative of A^q and $L(A,E)$ is then computed using a combination of the chain rule and the product rule for Fréchet derivatives.

4 References

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

Higham N J and Lin L (2011) A Schur–Padé algorithm for fractional powers of a matrix *SIAM J. Matrix Anal. Appl.* **32(3)** 1056–1078

Higham N J and Lin L (2013) An improved Schur–Padé algorithm for fractional powers of a matrix and their Fréchet derivatives *MIMS Eprint 2013.1* Manchester Institute for Mathematical Sciences, School of Mathematics, University of Manchester http://eprints.ma.man.ac.uk/

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 pth power, A^p.

3: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01JFF is called.

Constraint: LDA $\geq$ N.

4: E(LDE, $*$) – REAL (KIND=nag_wp) array *Input/Output*

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

On entry: the n by n matrix E.

On exit: the Fréchet derivative $L(A, E)$.

5: LDE – INTEGER *Input*

On entry: the first dimension of the array E as declared in the (sub)program from which F01JFF is called.

Constraint: LDE $\geq$ N.

6: P – REAL (KIND=nag_wp) *Input*

On entry: the required power of A.

7: 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 has eigenvalues on the negative real line. The principal pth power is not defined in this case; F01KFF can be used to find a complex, non-principal pth power.

IFAIL $= 2$

A is singular so the pth power cannot be computed.

IFAIL $= 3$

A^p 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 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 $= -5$

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

For a normal matrix A (for which $A^{\mathrm{T}}A = AA^{\mathrm{T}}$), the Schur decomposition is diagonal and the computation of the fractional part of the matrix power reduces to evaluating powers of the eigenvalues of A and then constructing A^p using the Schur vectors. This should give a very accurate result. In general, however, no error bounds are available for the algorithm. See Higham and Lin (2011) and Higham and Lin (2013) for details and further discussion.

If the condition number of the matrix power is required then F01JEF should be used.

8 Parallelism and Performance

F01JFF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F01JFF 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 real allocatable memory required by the algorithm is approximately $6 \times n^2$.

The cost of the algorithm is $O(n^3)$ floating-point operations; see Higham and Lin (2011) and Higham and Lin (2013).

If the matrix pth power alone is required, without the Fréchet derivative, then F01EQF should be used. If the condition number of the matrix power is required then F01JEF should be used. If A has negative real eigenvalues then F01KFF can be used to return a complex, non-principal pth power and its Fréchet derivative $L(A, E)$.

10 Example

This example finds A^p and the Fréchet derivative of the matrix power $L(A, E)$, where $p = 0.2$,

$$A = \begin{pmatrix} 3 & 3 & 2 & 1 \\ 3 & 1 & 0 & 2 \\ 1 & 1 & 4 & 3 \\ 3 & 0 & 3 & 1 \end{pmatrix} \quad \text{and} \quad E = \begin{pmatrix} 1 & 0 & 2 & 1 \\ 0 & 4 & 5 & 2 \\ 1 & 0 & 0 & 0 \\ 2 & 3 & 3 & 0 \end{pmatrix}.$$

10.1 Program Text

```
    Program f01jffe

!     F01JFF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: f01jff, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: p
      Integer                          :: i, ifail, lda, lde, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), e(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'F01JFF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, p
      lda = n
      lde = n
      Allocate (a(lda,n))
      Allocate (e(lde,n))
!     Read A from data file
      Read (nin,*)(a(i,1:n),i=1,n)
!     Read E from data file
      Read (nin,*)(e(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 A^p and L_p(A,E)
      Call f01jff(n,a,lda,e,lde,p,ifail)

!     Print solution
```

```
      Call x04caf('General',' ',n,n,a,lda,'A^p',ifail)
      Write (nout,*)
      Call x04caf('General',' ',n,n,e,lde,'L_p(A,E)',ifail)

    End Program f01jffe
```

10.2 Program Data

```
F01JFF Example Program Data

4     0.2              :Values of N and P

3.0   3.0   2.0   1.0
3.0   1.0   0.0   2.0
1.0   1.0   4.0   3.0
3.0   0.0   3.0   1.0  :End of matrix A

1.0   0.0   2.0   1.0
0.0   4.0   5.0   2.0
1.0   0.0   0.0   0.0
2.0   3.0   3.0   0.0  :End of matrix E
```

10.3 Program Results

```
 F01JFF Example Program Results

 A^p
              1          2          3          4
 1       1.2446     0.2375     0.2172    -0.1359
 2       0.0925     1.1239    -0.1453     0.3731
 3      -0.0769     0.1972     1.3131     0.1837
 4       0.3985    -0.2902     0.1085     1.1560

 L_p(A,E)
           1       2       3       4
 1    0.2189 -0.2004  0.0509  0.0290
 2   -0.3177  0.4143  0.3044  0.0760
 3   -0.0033 -0.1335 -0.2789  0.2699
 4    0.1972  0.3333  0.5379 -0.5228
```