

NAG Library Routine Document

F01JCF

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

F01JCF computes an estimate of the absolute condition number of a matrix function f at a real n by n matrix A in the 1-norm, using analytical derivatives of f you have supplied.

2 Specification

```
SUBROUTINE F01JCF (N, A, LDA, F, IUSER, RUSER, IFLAG, CONDA, NORMA, NORMFA,      &
                  IFAIL)
INTEGER          N, LDA, IUSER(*), IFLAG, IFAIL
REAL (KIND=nag_wp) A(LDA,*), RUSER(*), CONDA, NORMA, NORMFA
EXTERNAL        F
```

3 Description

The absolute condition number of f at A , $\text{cond}_{\text{abs}}(f, A)$ is given by the norm of the Fréchet derivative of f , $L(A, E)$, which is defined by

$$\|L(X)\| := \max_{E \neq 0} \frac{\|L(X, E)\|}{\|E\|}.$$

The Fréchet derivative in the direction E , $L(X, E)$ is linear in E and can therefore be written as

$$\text{vec}(L(X, E)) = K(X)\text{vec}(E),$$

where the vec operator stacks the columns of a matrix into one vector, so that $K(X)$ is $n^2 \times n^2$. F01JCF computes an estimate γ such that $\gamma \leq \|K(X)\|_1$, where $\|K(X)\|_1 \in [n^{-1}\|L(X)\|_1, n\|L(X)\|_1]$. The relative condition number can then be computed via

$$\text{cond}_{\text{rel}}(f, A) = \frac{\text{cond}_{\text{abs}}(f, A)\|A\|_1}{\|f(A)\|_1}.$$

The algorithm used to find γ is detailed in Section 3.4 of Higham (2008).

The function f , and the derivatives of f , are returned by subroutine F which, given an integer m , evaluates $f^{(m)}(z_i)$ at a number of (generally complex) points z_i , for $i = 1, 2, \dots, n_z$. For any z on the real line, $f(z)$ must also be real. F01JCF is therefore appropriate for routines that can be evaluated on the complex plane and whose derivatives, of arbitrary order, can also be evaluated on the complex plane.

4 References

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: $N \geq 0$.

- 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, $f(A)$.
- 3: LDA – INTEGER Input
On entry: the first dimension of the array A as declared in the (sub)program from which F01JCF is called.
Constraint: $LDA \geq \max(1, N)$.
- 4: F – SUBROUTINE, supplied by the user. External Procedure
 Given an integer m , the subroutine F evaluates $f^{(m)}(z_i)$ at a number of points z_i .

The specification of F is:

```
SUBROUTINE F (M, IFLAG, NZ, Z, FZ, IUSER, RUSER)
```

```
INTEGER M, IFLAG, NZ, IUSER(*)
```

```
REAL (KIND=nag_wp) RUSER(*)
```

```
COMPLEX (KIND=nag_wp) Z(NZ), FZ(NZ)
```

- 1: M – INTEGER Input
On entry: the order, m , of the derivative required.
 If $M = 0$, $f(z_i)$ should be returned. For $M > 0$, $f^{(m)}(z_i)$ should be returned.
- 2: IFLAG – INTEGER Input/Output
On entry: IFLAG will be zero.
On exit: IFLAG should either be unchanged from its entry value of zero, or may be set nonzero to indicate that there is a problem in evaluating the function $f(z)$; for instance $f(z)$ may not be defined. If IFLAG is returned as nonzero then F01JCF will terminate the computation, with IFAIL = 3.
- 3: NZ – INTEGER Input
On entry: n_z , the number of function or derivative values required.
- 4: Z(NZ) – COMPLEX (KIND=nag_wp) array Input
On entry: the n_z points z_1, z_2, \dots, z_{n_z} at which the function f is to be evaluated.
- 5: FZ(NZ) – COMPLEX (KIND=nag_wp) array Output
On exit: the n_z function or derivative values. $FZ(i)$ should return the value $f^{(m)}(z_i)$, for $i = 1, 2, \dots, n_z$. If z_i lies on the real line, then so must $f^{(m)}(z_i)$.
- 6: IUSER(*) – INTEGER array User Workspace
 7: RUSER(*) – REAL (KIND=nag_wp) array User Workspace
 F is called with the parameters IUSER and RUSER as supplied to F01JCF. You are free to use the arrays IUSER and RUSER to supply information to F as an alternative to using COMMON global variables.

F must either be a module subprogram USED by, or declared as EXTERNAL in, the (sub)program from which F01JCF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

- 5: IUSER(*) – INTEGER array *User Workspace*
 6: RUSER(*) – REAL (KIND=nag_wp) array *User Workspace*

IUSER and RUSER are not used by F01JCF, but are passed directly to F and may be used to pass information to this routine as an alternative to using COMMON global variables.

- 7: IFLAG – INTEGER *Output*

On exit: IFLAG = 0, unless IFLAG has been set nonzero inside F, in which case IFLAG will be the value set and IFAIL will be set to IFAIL = 3.

- 8: CONDA – REAL (KIND=nag_wp) *Output*

On exit: an estimate of the absolute condition number of f at A .

- 9: NORMA – REAL (KIND=nag_wp) *Output*

On exit: the 1-norm of A .

- 10: NORMFA – REAL (KIND=nag_wp) *Output*

On exit: the 1-norm of $f(A)$.

- 11: 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

An internal error occurred when estimating the norm of the Fréchet derivative of f at A . Please contact NAG.

IFAIL = 2

An internal error occurred when evaluating the matrix function $f(A)$. You can investigate further by calling F01EMF with the matrix A and the function f .

IFAIL = 3

IFLAG has been set nonzero by the user-supplied subroutine.

IFAIL = -1

On entry, $N < 0$.

Input argument number $\langle value \rangle$ is invalid.

IFAIL = -3

On entry, parameter LDA is invalid.
Constraint: $LDA \geq N$.

IFAIL = -999

Allocation of memory failed.

7 Accuracy

F01JCF uses the norm estimation routine F04YDF to estimate a quantity γ , where $\gamma \leq \|K(X)\|_1$ and $\|K(X)\|_1 \in [n^{-1}\|L(X)\|_1, n\|L(X)\|_1]$. For further details on the accuracy of norm estimation, see the documentation for F04YDF.

8 Further Comments

Approximately $6n^2$ of real allocatable memory is required by the routine, in addition to the memory used by the underlying matrix function routine F01EMF.

F01JCF returns the matrix function $f(A)$. This is computed using F01EMF. If only $f(A)$ is required, without an estimate of the condition number, then it is far more efficient to use F01EMF directly.

The complex analogue of this routine is F01KCF.

9 Example

This example estimates the absolute and relative condition numbers of the matrix function e^{2A} where

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

9.1 Program Text

```
! Mark 24 Release. NAG Copyright 2012.

Module f01jcf_mod

! .. Use Statements ..
Use nag_library, Only: nag_wp
! .. Implicit None Statement ..
Implicit None
Contains
Subroutine fexp2(m,iflag,nz,z,fz,iuser,ruser)

! .. Use Statements ..
Use nag_library, Only: nag_wp
! .. Implicit None Statement ..
Implicit None
! .. Scalar Arguments ..
Integer, Intent (Inout)      :: iflag
Integer, Intent (In)        :: m, nz
! .. Array Arguments ..
Complex (Kind=nag_wp), Intent (Out) :: fz(nz)
Complex (Kind=nag_wp), Intent (In)  :: z(nz)
Real (Kind=nag_wp), Intent (Inout)  :: ruser(*)
Integer, Intent (Inout)           :: iuser(*)
! .. Intrinsic Procedures ..
Intrinsic                      :: cmplx, exp
! .. Executable Statements ..
Continue
fz(1:nz) = (cmplx(2.0E0_nag_wp,0.0_nag_wp,kind=nag_wp)**m)* &
```

```

        exp((2.0E0_nag_wp,0.0E0_nag_wp)*z(1:nz))
    Return
End Subroutine fexp2

End Module f01jcfe_mod

Program f01jcfe

!     F01JCF Example Main Program

!     .. Use Statements ..
Use nag_library, Only: f01jcf, nag_wp, x02ajf, x04caf
Use f01jcfe_mod, Only: fexp2
!     .. Implicit None Statement ..
Implicit None
!     .. Parameters ..
Integer, Parameter                :: nin = 5, nout = 6
!     .. Local Scalars ..
Real (Kind=nag_wp)                :: conda, cond_rel, eps, norma,      &
                                   normfa
Integer                             :: i, ifail, iflag, lda, n
!     .. Local Arrays ..
Real (Kind=nag_wp), Allocatable    :: a(:, :)
Real (Kind=nag_wp)                 :: ruser(1)
Integer                             :: iuser(1)
!     .. Executable Statements ..
Write (nout,*) 'F01JCF 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)

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

!     Find absolute condition number estimate
ifail = 0
Call f01jcf(n,a,lda,fexp2,iuser,ruser,iflag,conda,norma,normfa,ifail)

If (ifail==0) Then
!     Print solution
Write (nout,*)
Write (nout,*) 'F(A) = exp(2A)'
Write (nout,99999) 'Estimated absolute condition number is: ', conda

!     Find relative condition number estimate
eps = x02ajf()
If (normfa>eps) Then
    cond_rel = conda*norma/normfa
    Write (nout,99999) 'Estimated relative condition number is: ', &
        cond_rel
Else
    Write (nout,99998) 'The estimated norm of f(A) is effectively zero', &
        'and so the relative condition number is undefined.'
End If
End If

99999 Format (1X,A,F7.2)
99998 Format (/1X,A/1X,A)

End Program f01jcfe

```

9.2 Program Data

F01JCF Example Program Data

```
4                               :Value of N
0.0  -1.0  -1.0   1.0
-2.0   0.0   1.0  -1.0
 2.0  -1.0   2.0  -2.0
-1.0  -2.0   0.0  -1.0 :End of matrix A
```

9.3 Program Results

F01JCF Example Program Results

```
A
      1      2      3      4
1      0.0000  -1.0000  -1.0000   1.0000
2     -2.0000   0.0000   1.0000  -1.0000
3      2.0000  -1.0000   2.0000  -2.0000
4     -1.0000  -2.0000   0.0000  -1.0000
```

$F(A) = \exp(2A)$

Estimated absolute condition number is: 183.90

Estimated relative condition number is: 13.90
