NAG Library Routine Document

F01JDF

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

F01JDF computes an estimate of the relative condition number $\kappa_{A^{1/2}}$ and a bound on the relative residual, in the Frobenius norm, for the square root of a real n by n matrix A. The principal square root, $A^{1/2}$, of A is also returned.

2 Specification

```
SUBROUTINE F01JDF (N, A, LDA, ALPHA, CONDSA, IFAIL)
INTEGER            N, LDA, IFAIL
REAL (KIND=nag_wp) A(LDA,*), ALPHA, CONDSA
```

3 Description

For a matrix with no eigenvalues on the closed negative real line, the principal matrix square root, $A^{1/2}$, of A is the unique square root with eigenvalues in the right half-plane.

The Fréchet derivative of a matrix function $A^{1/2}$ in the direction of the matrix E is the linear function mapping E to $L(A,E)$ such that

$$(A+E)^{1/2} - A^{1/2} - L(A,E) = o(\|A\|).$$

The absolute condition number is given by the norm of the Fréchet derivative which is defined by

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

The Fréchet derivative is linear in E and can therefore be written as

$$\mathrm{vec}(L(A,E)) = K(A)\mathrm{vec}(E),$$

where the vec operator stacks the columns of a matrix into one vector, so that $K(A)$ is $n^2 \times n^2$.

F01JDF uses Algorithm 3.20 from Higham (2008) to compute an estimate γ such that $\gamma \leq \|K(X)\|_F$. The quantity of γ provides a good approximation to $\|L(A)\|_F$. The relative condition number, $\kappa_{A^{1/2}}$, is then computed via

$$\kappa_{A^{1/2}} = \frac{\|L(A)\|_F \|A\|_F}{\|A^{1/2}\|_F}.$$

$\kappa_{A^{1/2}}$ is returned in the argument CONDSA.

$A^{1/2}$ is computed using the algorithm described in Higham (1987). This is a real arithmetic version of the algorithm of Björck and Hammarling (1983). In addition, a blocking scheme described in Deadman *et al.* (2013) is used.

The computed quantity α is a measure of the stability of the relative residual (see Section 7). It is computed via

$$\alpha = \frac{\left\|A^{1/2}\right\|_F^2}{\|A\|_F}.$$

4 References

Björck Å and Hammarling S (1983) A Schur method for the square root of a matrix *Linear Algebra Appl.* **52/53** 127–140

Deadman E, Higham N J and Ralha R (2013) Blocked Schur Algorithms for Computing the Matrix Square Root *Applied Parallel and Scientific Computing: 11th International Conference, (PARA 2012, Helsinki, Finland)* P. Manninen and P. Öster, Eds *Lecture Notes in Computer Science* **7782** 171–181 Springer–Verlag

Higham N J (1987) Computing real square roots of a real matrix *Linear Algebra Appl.* **88/89** 405–430

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: contains, if IFAIL $= 0$, the n by n principal matrix square root, $A^{1/2}$. Alternatively, if IFAIL $= 1$, contains an n by n non-principal square root of A.

3: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01JDF is called.

Constraint: LDA $\geq$ N.

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

On exit: an estimate of the stability of the relative residual for the computed principal (if IFAIL $= 0$) or non-principal (if IFAIL $= 1$) matrix square root, α.

5: CONDSA – REAL (KIND=nag_wp) *Output*

On exit: an estimate of the relative condition number, in the Frobenius norm, of the principal (if IFAIL $= 0$) or non-principal (if IFAIL $= 1$) matrix square root at A, $\kappa_{A^{1/2}}$.

6: 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 a semisimple vanishing eigenvalue. A non-principal square root was returned.

IFAIL $= 2$

A has a defective vanishing eigenvalue. The square root and condition number cannot be found in this case.

IFAIL $= 3$

A has a negative real eigenvalue. The principal square root is not defined. F01KDF can be used to return a complex, non-principal square root.

IFAIL $= 4$

An error occurred when computing the matrix square root. Consequently, ALPHA and CONDSA could not be computed. It is likely that the routine was called incorrectly.

IFAIL $= 5$

An error occurred when computing the condition number. The matrix square root was still returned but you should use F01ENF to check if it is the principal matrix square root.

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

If the computed square root is $\tilde{X}$, then the relative residual

$$\frac{\left\|A - \tilde{X}^2\right\|_F}{\|A\|_F},$$

is bounded approximately by $n\alpha\epsilon$, where ϵ is ***machine precision***. The relative error in $\tilde{X}$ is bounded approximately by $n\alpha\kappa_{A^{1/2}}\epsilon$.

8 Parallelism and Performance

F01JDF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F01JDF 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

Approximately $3 \times n^2$ of real allocatable memory is required by the routine.

The cost of computing the matrix square root is $85n^3/3$ floating-point operations. The cost of computing the condition number depends on how fast the algorithm converges. It typically takes over twice as long as computing the matrix square root.

If condition estimates are not required then it is more efficient to use F01ENF to obtain the matrix square root alone. Condition estimates for the square root of a complex matrix can be obtained via F01KDF.

10 Example

This example estimates the matrix square root and condition number of the matrix

$$A = \begin{pmatrix} -5 & 2 & -1 & 1 \\ -2 & -3 & 19 & 27 \\ -9 & 0 & 15 & 24 \\ 7 & 8 & 11 & 16 \end{pmatrix}.$$

10.1 Program Text

```
    Program f01jdfe

!     F01JDF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: f01jdf, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: alpha, condsa
      Integer                          :: i, ifail, lda, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'F01JDF 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 sqrt(A)
      Call f01jdf(n,a,lda,alpha,condsa,ifail)

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

      Write (nout,*)
      Write (nout,99999) 'Estimated relative condition number is: ', condsa
      Write (nout,99999) 'Condition number for the relative residual is: ', &
        alpha

99999 Format (1X,A,F6.2)

    End Program f01jdfe
```

10.2 Program Data

```
F01JDF Example Program Data

4                               :Value of N

-5.0     2.0     -1.0     1.0
-2.0    -3.0     19.0    27.0
-9.0     0.0     15.0    24.0
 7.0     8.0     11.0    16.0  :End of matrix A
```

10.3 Program Results

```
 F01JDF Example Program Results

 sqrt(A)
             1          2          3          4
 1      1.0000     2.0000    -1.0000    -1.0000
 2     -3.0000     1.0000     2.0000     4.0000
 3     -2.0000     3.0000     1.0000     2.0000
 4      2.0000    -1.0000     3.0000     4.0000

 Estimated relative condition number is:  77.10
 Condition number for the relative residual is:   1.70
```