F01LEF – NAG Fortran Library Routine Document

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

F01LEF computes an LU factorization of a real tridiagonal matrix, using Gaussian elimination with partial pivoting.

2 Specification

```
SUBROUTINE F01LEF(N, A, LAMBDA, B, C, TOL, D, IN, IFAIL)
INTEGER            N, IN(N), IFAIL
real               A(N), LAMBDA, B(N), C(N), TOL, D(N)
```

3 Description

The matrix $T - \lambda I$, where T is a real n by n tridiagonal matrix, is factorized as

$$T - \lambda I = PLU,$$

where P is a permutation matrix, L is a unit lower triangular matrix with at most one non-zero sub-diagonal element per column and U is an upper triangular matrix with at most two non-zero super-diagonal elements per column.

The factorization is obtained by Gaussian elimination with partial pivoting and implicit row scaling.

An indication of whether or not the matrix $T - \lambda I$ is nearly singular is returned in the nth element of the array IN. If it is important that $T - \lambda I$ is non-singular, as is usually the case when solving a system of tridiagonal equations, then it is strongly recommended that IN(n) is inspected on return from F01LEF. (See the parameter IN and Section 8 for further details.)

The parameter λ is included in the routine so that F01LEF may be used, in conjunction with F04LEF, to obtain eigenvectors of T by inverse iteration.

4 References

[1] Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation II, Linear Algebra* Springer-Verlag

[2] Wilkinson J H (1965) *The Algebraic Eigenvalue Problem* Oxford University Press, London

5 Parameters

1: N — INTEGER *Input*

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

Constraint: N ≥ 1.

2: A(N) — ***real*** array *Input/Output*

On entry: the diagonal elements of T.

On exit: the diagonal elements of the upper triangular matrix U.

3: LAMBDA — ***real*** *Input*

On entry: the scalar λ. The routine factorizes $T - \lambda I$.

4: B(N) — ***real*** array *Input/Output*

On entry: the super-diagonal elements of T, stored in B(2) to B(n); B(1) is not used.

On exit: the elements of the first super-diagonal of U, stored in B(2) to B(n).

5: C(N) — ***real*** array *Input/Output*

On entry: the sub-diagonal elements of T, stored in C(2) to C(n); C(1) is not used.

On exit: the sub-diagonal elements of L, stored in C(2) to C(n).

6: TOL — ***real*** *Input*

On entry: a relative tolerance used to indicate whether or not the matrix $(T-\lambda I)$ is nearly singular. TOL should normally be chosen as approximately the largest relative error in the elements of T. For example, if the elements of T are correct to about 4 significant figures, then TOL should be set to about 5×10^{-4}. See Section 8 for further details on how TOL is used. If TOL is supplied as less than ϵ, where ϵ is the ***machine precision***, then the value ϵ is used in place of TOL.

7: D(N) — ***real*** array *Output*

On exit: the elements of the second super-diagonal of U, stored in D(3) to D(n); D(1) and D(2) are not used.

8: IN(N) — INTEGER array *Output*

On exit: details of the permutation matrix P. If an interchange occurred at the kth step of the elimination, then IN(k) = 1, otherwise IN(k) = 0. If a diagonal element of U is small, indicating that $(T-\lambda I)$ is nearly singular, then the element IN(n) is returned as positive. Otherwise IN(n) is returned as 0. See Section 8 for further details. If the application is such that it is important that $(T-\lambda I)$ is not nearly singular, then it is strongly recommended that IN(n) is inspected on return.

9: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, N < 1.

7 Accuracy

The computed factorization will satisfy the equation

$$PLU = (T-\lambda I) + E,$$

where

$$\|E\|_1 \leq 9 \times \max_{i \geq j}(\|l_{ij}\|, \|l_{ij}\|^2)\epsilon\|T-\lambda I\|_1$$

where ϵ is the ***machine precision***.

8 Further Comments

The time taken by the routine is approximately proportional to n.

The factorization of a tridiagonal matrix proceeds in $(n-1)$ steps, each step eliminating one sub-diagonal element of the tridiagonal matrix. In order to avoid small pivot elements and to prevent growth in the size of the elements of L, rows k and $(k+1)$ will, if necessary, be interchanged at the kth step prior to the elimination.

The element IN(n) returns the smallest integer, j, for which

$$|u_{jj}| \leq \|(T-\lambda I)_j\|_1 \times \mathrm{TOL},$$

where $\|(T-\lambda I)_j\|_1$ denotes the sum of the absolute values of the jth row of the matrix $(T-\lambda I)$. If no such j exists, then IN(n) is returned as zero. If such a j exists, then $|u_{jj}|$ is small and hence $(T-\lambda I)$ is singular or nearly singular.

This routine may be followed by F04LEF to solve systems of tridiagonal equations. Users wishing to solve single systems of tridiagonal equations may wish to be aware of F04EAF, which solves tridiagonal systems with a single call. F04EAF requires less storage and will generally be faster than the combination of F01LEF and F04LEF, but no test for near singularity is included in F04EAF and so it should only be used when the equations are known to be non-singular.

9 Example

To factorize the tridiagonal matrix T where

$$T = \begin{pmatrix} 3.0 & 2.1 & 0 & 0 & 0 \\ 3.4 & 2.3 & -1.0 & 0 & 0 \\ 0 & 3.6 & -5.0 & 1.9 & 0 \\ 0 & 0 & 7.0 & -0.9 & 8.0 \\ 0 & 0 & 0 & -6.0 & 7.1 \end{pmatrix}$$

and then to solve the equations $Tx = y$, where

$$y = \begin{pmatrix} 2.7 \\ -0.5 \\ 2.6 \\ 0.6 \\ 2.7 \end{pmatrix}$$

by a call to F04LEF. The example program sets TOL $= 5 \times 10^{-5}$ and, of course, sets LAMBDA $= 0$.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F01LEF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NMAX
      PARAMETER        (NMAX=50)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             LAMBDA, TOL
      INTEGER          I, IFAIL, JOB, N
*     .. Local Arrays ..
      real             A(NMAX), B(NMAX), C(NMAX), D(NMAX), Y(NMAX)
      INTEGER          IN(NMAX)
```

```
*     .. External Subroutines ..
      EXTERNAL         F01LEF, F04LEF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F01LEF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      WRITE (NOUT,*)
      IF (N.LT.1 .OR. N.GT.NMAX) THEN
         WRITE (NOUT,99999) 'N is out of range. N = ', N
      ELSE
         READ (NIN,*) (A(I),I=1,N)
         READ (NIN,*) (B(I),I=2,N)
         READ (NIN,*) (C(I),I=2,N)
         TOL = 0.00005e0
         LAMBDA = 0.0e0
         IFAIL = 1
*
         CALL F01LEF(N,A,LAMBDA,B,C,TOL,D,IN,IFAIL)
*
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99999) 'F01LEF fails. IFAIL =', IFAIL
         ELSE
            IF (IN(N).NE.0) THEN
               WRITE (NOUT,*) 'Matrix is singular or nearly singular'
               WRITE (NOUT,99998) 'Diagonal element', IN(N), 'is small'
            ELSE
               WRITE (NOUT,*) 'Details of factorization'
               WRITE (NOUT,*)
               WRITE (NOUT,*) ' Main diagonal of U'
               WRITE (NOUT,99997) (A(I),I=1,N)
               WRITE (NOUT,*)
               WRITE (NOUT,*) ' First super-diagonal of U'
               WRITE (NOUT,99997) (B(I),I=2,N)
               WRITE (NOUT,*)
               WRITE (NOUT,*) ' Second super-diagonal of U'
               WRITE (NOUT,99997) (D(I),I=3,N)
               WRITE (NOUT,*)
               WRITE (NOUT,*) ' Multipliers'
               WRITE (NOUT,99997) (C(I),I=2,N)
               WRITE (NOUT,*)
               WRITE (NOUT,*) ' Vector of interchanges'
               WRITE (NOUT,99996) (IN(I-1),I=2,N)
*
               READ (NIN,*) (Y(I),I=1,N)
               JOB = 1
               IFAIL = 1
*
               CALL F04LEF(JOB,N,A,B,C,D,IN,Y,TOL,IFAIL)
*
               WRITE (NOUT,*)
               IF (IFAIL.NE.0) THEN
                  WRITE (NOUT,99999) 'F04LEF fails. IFAIL =', IFAIL
               ELSE
                  WRITE (NOUT,*) ' Solution vector'
                  WRITE (NOUT,99997) (Y(I),I=1,N)
               END IF
            END IF
```

```
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,A,I4,A)
99997 FORMAT (1X,8F9.4)
99996 FORMAT (1X,5I9)
      END
```

9.2 Program Data

```
F01LEF Example Program Data
   5
  3.0   2.3  -5.0  -0.9   7.1
  2.1  -1.0   1.9   8.0
  3.4   3.6   7.0  -6.0
  2.7  -0.5   2.6   0.6   2.7
```

9.3 Program Results

```
 F01LEF Example Program Results

 Details of factorization

  Main diagonal of U
    3.0000   3.6000   7.0000  -6.0000   1.1508

  First super-diagonal of U
    2.1000  -5.0000  -0.9000   7.1000

  Second super-diagonal of U
    0.0000   1.9000   8.0000

  Multipliers
    1.1333  -0.0222  -0.1587   0.0168

  Vector of interchanges
         0        1        1        1

  Solution vector
   -4.0000   7.0000   3.0000  -4.0000  -3.0000
```