F04MFF – 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

F04MFF updates the solution of the equations $Tx = b$, where T is a real symmetric positive-definite Toeplitz matrix.

2 Specification

```
SUBROUTINE F04MFF(N, T, B, X, P, WORK, IFAIL)
INTEGER           N, IFAIL
real              T(0:*), B(*), X(*), P, WORK(*)
```

3 Description

This routine solves the equations

$$T_n x_n = b_n,$$

where T_n is the n by n symmetric positive-definite Toeplitz matrix

$$T_n = \begin{pmatrix} \tau_0 & \tau_1 & \tau_2 & \dots & \tau_{n-1} \\ \tau_1 & \tau_0 & \tau_1 & \dots & \tau_{n-2} \\ \tau_2 & \tau_1 & \tau_0 & \dots & \tau_{n-3} \\ . & . & . & & . \\ \tau_{n-1} & \tau_{n-2} & \tau_{n-3} & \dots & \tau_0 \end{pmatrix}$$

and b_n is the n element vector $b_n = (\beta_1 \beta_2 \dots \beta_n)^T$, given the solution of the equations

$$T_{n-1} x_{n-1} = b_{n-1}.$$

This routine will normally be used to successively solve the equations

$$T_k x_k = b_k, \quad k = 1, 2, \dots, n.$$

If it is desired to solve the equations for a single value of n, then routine F04FFF may be called. This routine uses the method of Levinson [4], [5].

4 References

[1] Bunch J R (1985) Stability of methods for solving Toeplitz systems of equations *SIAM J. Sci. Statist. Comput.* **6** 349–364

[2] Bunch J R (1987) The weak and strong stability of algorithms in numerical linear algebra *Linear Algebra Appl.* **88/89** 49–66

[3] Cybenko G (1980) The numerical stability of the Levinson–Durbin algorithm for Toeplitz systems of equations *SIAM J. Sci. Statist. Comput.* **1** 303–319

[4] Golub G H and van Loan C F (1996) *Matrix Computations* Johns Hopkins University Press (3rd Edition), Baltimore

[5] Levinson N (1947) The Weiner RMS error criterion in filter design and prediction *J. Math. Phys.* **25** 261–278

5 Parameters

1: N — INTEGER *Input*

On entry: the order of the Toeplitz matrix T.

Constraint: $N \geq 0$. When $N = 0$, then an immediate return is effected.

2: T(0:∗) — ***real*** array *Input*

Note: the dimension of the array T must be at least max(1,N).

On entry: T(i) must contain the values τ_i, $i = 0, 1, \ldots, N-1$.

Constraint: $T(0) > 0.0$. Note that if this is not true, then the Toeplitz matrix cannot be positive-definite.

3: B(∗) — ***real*** array *Input*

Note: the dimension of the array B must be at least max(1,N).

On entry: the right-hand side vector b_n.

4: X(∗) — ***real*** array *Input/Output*

Note: the dimension of the array X must be at least max(1,N).

On entry: with $N > 1$ the $(n-1)$ elements of the solution vector x_{n-1} as returned by a previous call to this routine. The element X(N) need not be specified.

On exit: the solution vector x_n.

5: P — ***real*** *Output*

On exit: the reflection coefficient p_{n-1}. (See Section 8.)

6: WORK(∗) — ***real*** array *Input/Output*

Note: the dimension of the array WORK must be at least max(1,2∗N−1).

On entry: with $N > 2$ the elements of WORK should be as returned from a previous call to F04MFF with (N−1) as the argument N.

On exit: the first (N−1) elements of WORK contain the solution to the Yule–Walker equations

$$T_{n-1}y_{n-1} = -t_{n-1},$$

where $t_{n-1} = (\tau_1\tau_2 \ldots \tau_{n-1})^T$.

7: 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

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = −1

On entry, $N < 0$,
or $T(0) \leq 0.0$.

IFAIL = 1

The Toeplitz matrix T_n is not positive-definite to working accuracy. If, on exit, P is close to unity, then T_n was probably close to being singular.

7 Accuracy

The computed solution of the equations certainly satisfies

$$r = T_n x_n - b_n,$$

where $\|r\|_1$ is approximately bounded by

$$\|r\|_1 \le c\epsilon C(T_n),$$

c being a modest function of n, ϵ being the ***machine precision*** and $C(T)$ is the condition number of T with respect to inversion. This bound is almost certainly pessimistic, but it seems unlikely that the method of Levinson is backward stable, so caution should be exercised when T_n is ill-conditioned. The following bound on T_n^{-1} holds,

$$\max\left(\frac{1}{\prod_{i=1}^{n-1}(1-p_i^2)}, \frac{1}{\prod_{i=1}^{n-1}(1-p_i)}\right) \le \|T_n^{-1}\|_1 \le \prod_{i=1}^{n-1}\left(\frac{1+|p_i|}{1-|p_i|}\right).$$

(See Golub and Van Loan [4].) The norm of T_n^{-1} may also be estimated using routine F04YCF. For further information on stability issues see Bunch [1] and [2], Cybenko [3] Golub and Van Loan [4].

8 Further Comments

The number of floating-point operations used by this routine is approximately $8n$.

If y_i is the solution of the equations

$$T_i y_i = -(\tau_1 \tau_2 \ldots \tau_i)^T,$$

then the reflection coefficient p_i is defined as the ith element of y_i.

9 Example

To find the solution of the equations $T_k x_k = b_k$, $k = 1, 2, 3, 4$, where

$$T_4 = \begin{pmatrix} 4 & 3 & 2 & 1 \\ 3 & 4 & 3 & 2 \\ 2 & 3 & 4 & 3 \\ 1 & 2 & 3 & 4 \end{pmatrix} \text{ and } b_4 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

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.

```
*     F04MFF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX
      PARAMETER        (NMAX=100)
*     .. Local Scalars ..
      real             P
      INTEGER          I, IFAIL, K, N
*     .. Local Arrays ..
      real             B(NMAX), T(0:NMAX-1), WORK(2*NMAX-1), X(NMAX)
```

```
*     .. External Subroutines ..
      EXTERNAL         F04MFF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F04MFF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      WRITE (NOUT,*)
      IF ((N.LT.0) .OR. (N.GT.NMAX)) THEN
         WRITE (NOUT,99999) 'N is out of range. N = ', N
      ELSE
         READ (NIN,*) (T(I),I=0,N-1)
         READ (NIN,*) (B(I),I=1,N)
*
         DO 20 K = 1, N
*
            IFAIL = 0
*
            CALL F04MFF(K,T,B,X,P,WORK,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Solution for system of order', K
            WRITE (NOUT,99998) (X(I),I=1,K)
            IF (K.GT.1) THEN
               WRITE (NOUT,*) 'Reflection coefficient'
               WRITE (NOUT,99998) P
            END IF
   20    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,5F9.4)
      END
```

9.2 Program Data

```
F04MFF Example Program Data

   4                    :Value of N
   4.0  3.0  2.0  1.0    :End of vector T
   1.0  1.0  1.0  1.0    :End of vector B
```

9.3 Program Results

```
 F04MFF Example Program Results


 Solution for system of order    1
    0.2500

 Solution for system of order    2
    0.1429   0.1429
 Reflection coefficient
   -0.7500
```

```
Solution for system of order    3
   0.1667   0.0000   0.1667
Reflection coefficient
   0.1429

Solution for system of order    4
   0.2000   0.0000   0.0000   0.2000
Reflection coefficient
   0.1667
```