NAG Library Routine Document

F04MEF

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

F04MEF updates the solution to the Yule–Walker equations for a real symmetric positive definite Toeplitz system.

2 Specification

```
SUBROUTINE F04MEF (N, T, X, V, WORK, IFAIL)
INTEGER             N, IFAIL
REAL (KIND=nag_wp) T(0:N), X(*), V, WORK(N-1)
```

3 Description

F04MEF solves the equations

$$T_n x_n = -t_n,$$

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 t_n is the vector

$$t_n^{\mathrm{T}} = (\tau_1 \tau_2 \dots \tau_n),$$

given the solution of the equations

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

The routine will normally be used to successively solve the equations

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

If it is desired to solve the equations for a single value of n, then routine F04FEF may be called. This routine uses the method of Durbin (see Durbin (1960) and Golub and Van Loan (1996)).

4 References

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

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

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

Durbin J (1960) The fitting of time series models *Rev. Inst. Internat. Stat.* **28** 233

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

5 Arguments

1: N – INTEGER *Input*

On entry: the order of the Toeplitz matrix T.

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

2: T(0 : N) – REAL (KIND=nag_wp) array *Input*

On entry: T(0) must contain the value τ_0 of the diagonal elements of T, and the remaining N elements of T must contain the elements of the vector t_n.

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

3: X($*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the dimension of the array X must be at least $\max(1, \mathrm{N})$.

On entry: with $\mathrm{N} > 1$ the $(n-1)$ elements of the solution vector x_{n-1} as returned by a previous call to F04MEF. The element X(N) need not be specified.

Constraint: $|\mathrm{X}(\mathrm{N}-1)| < 1.0$. Note that this is the partial (auto)correlation coefficient, or reflection coefficient, for the $(n-1)$th step. If the constraint does not hold, then T_n cannot be positive definite.

On exit: the solution vector x_n. The element X(N) returns the partial (auto)correlation coefficient, or reflection coefficient, for the nth step. If $|\mathrm{X}(\mathrm{N})| \geq 1.0$, then the matrix T_{n+1} will not be positive definite to working accuracy.

4: V – REAL (KIND=nag_wp) *Input/Output*

On entry: with $\mathrm{N} > 1$ the mean square prediction error for the $(n-1)$th step, as returned by a previous call to F04MEF.

On exit: the mean square prediction error, or predictor error variance ratio, ν_n, for the nth step. (See Section 9 and the Introduction to Chapter G13.)

5: WORK(N − 1) – REAL (KIND=nag_wp) array *Workspace*

6: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 argument, 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: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL} = 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = -1$

On entry, $\mathrm{N} < 0$,
or $\mathrm{T}(0) \leq 0.0$,

or $\quad N > 1$ and $|X(N-1)| \geq 1.0$.

IFAIL $= 1$

The Toeplitz matrix T_{n+1} is not positive definite to working accuracy. If, on exit, X(N) is close to unity, then the principal minor was probably close to being singular, and the sequence $\tau_0, \tau_1, \ldots, \tau_N$ may be a valid sequence nevertheless. X returns the solution of the equations

$$T_n x_n = -t_n,$$

and V returns v_n, but it may not be positive.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The computed solution of the equations certainly satisfies

$$r = T_n x_n + t_n,$$

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

$$\|r\|_1 \leq c\epsilon\left(\prod_{i=1}^{n}(1+|p_i|)-1\right),$$

c being a modest function of n, ϵ being the ***machine precision*** and p_k being the kth element of x_k. This bound is almost certainly pessimistic, but it has not yet been established whether or not the method of Durbin is backward stable. For further information on stability issues see Bunch (1985), Bunch (1987), Cybenko (1980) and Golub and Van Loan (1996). The following bounds on $\left\|T_n^{-1}\right\|_1$ hold:

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

where v_n is the mean square prediction error for the nth step. (See Cybenko (1980).) Note that $v_n < v_{n-1}$. The norm of T_n^{-1} may also be estimated using routine F04YDF.

8 Parallelism and Performance

F04MEF 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 number of floating-point operations used by this routine is approximately $4n$.

The mean square prediction errors, v_i, is defined as

$$v_i = \left(\tau_0 + t_{i-1}^{\mathrm{T}} x_{i-1}\right) / \tau_0.$$

Note that $v_i = \left(1 - p_i^2\right) v_{i-1}$.

10 Example

This example finds the solution of the Yule–Walker equations $T_k x_k = -t_k$, $k = 1, 2, 3, 4$ where

$$T_4 = \begin{pmatrix} 4 & 3 & 2 & 1 \\ 3 & 4 & 3 & 2 \\ 2 & 3 & 4 & 3 \\ 1 & 2 & 3 & 4 \end{pmatrix} \quad \text{and} \quad t_4 = \begin{pmatrix} 3 \\ 2 \\ 1 \\ 0 \end{pmatrix}.$$

10.1 Program Text

```
    Program f04mefe

!     F04MEF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f04mef, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: v
      Integer                          :: ifail, k, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: t(:), work(:), x(:)
!     .. Executable Statements ..
      Write (nout,*) 'F04MEF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n
      Allocate (t(0:n),work(n-1),x(n))
      Read (nin,*) t(0:n)

      Do k = 1, n

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call f04mef(k,t,x,v,work,ifail)

        Write (nout,*)
        Write (nout,99999) 'Solution for system of order', k
        Write (nout,99998) x(1:k)
        Write (nout,*) 'Mean square prediction error'
        Write (nout,99998) v
      End Do

99999 Format (1X,A,I5)
99998 Format (1X,5F9.4)
    End Program f04mefe
```

10.2 Program Data

```
F04MEF Example Program Data

   4                         : n
   4.0  3.0  2.0  1.0  0.0     : vector T
```

10.3 Program Results

```
 F04MEF Example Program Results


 Solution for system of order    1
   -0.7500
 Mean square prediction error
    0.4375

 Solution for system of order    2
   -0.8571   0.1429
 Mean square prediction error
    0.4286

 Solution for system of order    3
   -0.8333   0.0000   0.1667
 Mean square prediction error
    0.4167

 Solution for system of order    4
   -0.8000   0.0000  -0.0000   0.2000
 Mean square prediction error
    0.4000
```