NAG Toolbox for MATLAB

f04mf

1 Purpose

f04mf updates the solution of the equations $Tx = b$, where T is a real symmetric positive-definite Toeplitz matrix.

2 Syntax

```
[x, p, work, ifail] = f04mf(t, b, x, work, 'n', n)
```

3 Description

f04mf 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 $b_n = (\beta_1 \beta_2 \dots \beta_n)^{\mathrm{T}}$, given the solution of the equations

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

This function will normally be used to successively solve the equations

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

If it is desired to solve the equations for a single value of n, then function f04ff may be called. This function uses the method of Levinson (see Levinson 1947 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

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

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

5 Parameters

5.1 Compulsory Input Parameters

1: **t(0 : $*$) – double array**

Note: the dimension of the array **t** must be at least $\max(1, \mathbf{n})$.

t(i) must contain the values τ_i, $i = 0, 1, \ldots, \mathbf{n} - 1$.

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

2: **b($*$) – double array**

Note: the dimension of the array **b** must be at least $\max(1, \mathbf{n})$.

The right-hand side vector b_n.

3: **x($*$) – double array**

Note: the dimension of the array **x** must be at least $\max(1, \mathbf{n})$.

With $\mathbf{n} > 1$ the $(n-1)$ elements of the solution vector x_{n-1} as returned by a previous call to f04mf. The element $\mathbf{x}(\mathbf{n})$ need not be specified.

4: **work($*$) – double array**

Note: the dimension of the array **work** must be at least $\max(1, 2 \times \mathbf{n} - 1)$.

With $\mathbf{n} > 2$ the elements of **work** should be as returned from a previous call to f04mf with $(\mathbf{n} - 1)$ as the parameter **n**.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the array **t** The dimension of the array **b** The dimension of the array **x**.

The order of the Toeplitz matrix T.

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

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **x($*$) – double array**

Note: the dimension of the array **x** must be at least $\max(1, \mathbf{n})$.

The solution vector x_n.

2: **p – double scalar**

The reflection coefficient p_{n-1}. (See Section 8.)

3: **work($*$) – double array**

Note: the dimension of the array **work** must be at least $\max(1, 2 \times \mathbf{n} - 1)$.

The first $(\mathbf{n} - 1)$ elements of **work** contain the solution to the Yule–Walker equations

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

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

4: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

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

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

c being a modest function of n, ϵ being the ***machine precision*** and $C(T)$ being 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:

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

(See Golub and Van Loan 1996.) The norm of T_n^{-1} may also be estimated using function f04yc. For further information on stability issues see Bunch 1985, Bunch 1987, Cybenko 1980 and Golub and Van Loan 1996.

8 Further Comments

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

If y_i is the solution of the equations

$$T_i y_i = -(\tau_1 \tau_2 \dots \tau_i)^{\mathrm{T}},$$

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

9 Example

```
t = [4; 3; 2; 1];
b = [1; 1; 1; 1];
x = [0];
work = zeros(9,1);
fprintf('\n');
for k=1:4
  [x, p, work, ifail] = f04mf(t(1:k), b(1:k), x, work);
```

```
   fprintf('Solution for system of order %d\n', k);
  disp(transpose(x));
  if k > 1
    fprintf('Reflection coefficient\n    %6.4g\n\n', p);
  end
  if k < 4
    x = [x; 0]; % Extend x by one element
  end
end
```

```
Solution for system of order 1
    0.2500
Solution for system of order 2
    0.1429    0.1429
Reflection coefficient
     -0.75

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
```