

TWO ALGORITHMS FOR ANALYSIS OF ARMA TIME SERIES MODELS

Abstract

Ansley (1978) derived an algorithm to compute the likelihood function of data from an ARMA time series model. The algorithm reported here follows Ansley's basic idea but has some different features. While it cannot accommodate seasonal models, it estimates the mean, gives forecasts and their covariance matrix, and avoids computing square roots. A second algorithm is described which computes the necessarily information for a structured Bayesian analysis.

I. Notation and Definitions

Consider the autoregressive-moving average process $\{z_t\}$, of order (p, q) , described by (1.1)

$$(1.1) \quad z_t - \phi_1 z_{t-1} - \dots - \phi_p z_{t-p} = \mu + a_t - \theta_1 a_{t-1} - \dots - \theta_q a_{t-q}$$

where a_t 's are iid normal random variables each having a normal distribution with mean zero and variance σ_a^2 . A finite segment of this process is observed: $z = (z_1, \dots, z_N)^T$ which then has an N-dimensional multivariate normal distribution with mean vector $\mu \mathbf{1}_N$ and covariance matrix $\sigma_a^2 A_N$. The matrix A_N is described by

$$(1.2) \quad (A_N)_{ij} = \sigma(|i-j|)$$

where the covariance function σ characterizes this stationary time series process. By taking variances of both sides of (1.1), the covariance function σ can be determined

$$(1.3) \quad \begin{aligned} & \text{Cov} (z_t - \phi_1 z_{t-1} - \dots - \phi_p z_{t-p}, z_{t+s} - \phi_1 z_{t+s-1} - \dots - \phi_q z_{t+s-q}) \\ &= \sum_{i=0}^p \sum_{j=0}^p \phi_i \phi_j \sigma(|s+i-j|) = \sum_{j=0}^q \theta_j \theta_{j+s} \quad \forall s \geq 0 \end{aligned}$$

where $\phi_0 = \theta_0 = -1$ and $\phi_i = \theta_j = 0$ for $i > p$ or $j > q$.

(Cf. Anderson (1971, p. 237) and McLeod (1975)).

Since forecasting is of primary interest, the distribution of the future n observations, that is $z_F = (z_{N+1}, \dots, z_{N+n})^T$, conditional on z_N , has a multivariate normal distribution with mean $\mu_{1n} + A_{21} A_N^{-1} (z_N - \mu_{1N})$ and covariance matrix $\sigma_a^2 A_{nN}^*$ where

$$A_{N+n} = \begin{bmatrix} A_N & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{matrix} N \\ n \end{matrix}$$

and $A_{nN}^* = A_{22} - A_{21} A_N^{-1} A_{12}$. In the notation to be used here,

$$(z_F | z, \mu, \sigma_a^2) \sim N(\mu_{1n} + A_{21} A_N^{-1} (z_N - \mu_{1N}), \sigma_a^2 A_{nN}^*).$$

Note that A_{N+n} is a function of $\psi = (\phi_1, \dots, \phi_p, \theta_1, \dots, \theta_q)$.

II. Maximum Likelihood

The analysis for applying the maximum likelihood approach to this problem is rather straightforward, but only when the model $M = (p, q)$ is assumed known. The density of the observations is given by

$$(2.1) \quad p(z | \mu, \sigma_a^2, \psi) = (2\pi)^{-N/2} |A_N|^{-1/2} (\sigma_a^2)^{-N/2} \exp \left\{ -\frac{1}{2\sigma_a^2} (z - \mu_{1N})^T A_N^{-1} (z - \mu_{1N}) \right\}$$

By taking the natural logarithm, the log-likelihood function is given by

$$(2.2) \quad \ell(\mu, \sigma_a^2, \psi) = C_1 - N/2 \ln \sigma_a^2 - 1/2 \ln |A_N| - \frac{1}{2\sigma_a^2} (z - \mu_{1N})^T A_N^{-1} (z - \mu_{1N})$$

where C_i 's are constants. To concentrate on μ , setting $\partial \ell(\mu, \sigma_a^2, \psi) / \partial \mu = 0$ yields:

$$(2.3) \quad \hat{\mu} = z^T A_N^{-1} 1_N / 1_N^T A_N^{-1} 1_N$$

Concentrating now on σ_a^2 , setting $\partial \ell(\hat{\mu}, \sigma_a^2, \psi) / \partial \sigma_a^2$ to zero

$$-\frac{N}{2} \hat{\sigma}_a^{-2} + \frac{1}{2} \hat{\sigma}_a^{-4} (z - \hat{\mu} 1_N)^T A_N^{-1} (z - \hat{\mu} 1_N) = 0$$

which yields

$$\hat{\sigma}_a^2 = (z - \hat{\mu} 1_N)^T A_N^{-1} (z - \hat{\mu} 1_N) / N$$

which involves the quadratic form $Q(\psi) = N \hat{\sigma}_a^2$. The concentrated log-likelihood function is not

$$(2.4) \quad \ell_3(\psi) = C_2 - N/2 \ln Q(\psi) - \frac{1}{2} \ln |A_N|$$

The MLE for the remaining parameters, $\hat{\psi}$, is that vector ψ which maximizes $\ell_3(\psi)$, which must be found numerically. Note that for identifiability purposes, the values of the parameter vector ψ must be restricted to that region which insures the process $\{z_t\}$ to be stationary and invertible.

The forecast function for the maximum likelihood approach is quite simple, again, given M . That is, the forecast is just the mean of $(z_F | \mu, \sigma_a^2, z, \psi)$:

$$(2.5) \quad z_F = \hat{\mu} 1_n + A_{21}(\hat{\psi}) A_N^{-1}(\hat{\psi}) (z - \hat{\mu} 1_N)$$

Likewise, the desired covariance matrix for the forecasts treats ψ as given:

$$(2.6) \quad \text{Cov}(z_F) = \left[A_{22}(\hat{\psi}) - A_{21}(\hat{\psi})A_N^{-1}(\hat{\psi})A_{12}(\hat{\psi}) \right] \hat{\sigma}_a^2 .$$

The objective of this paper is to describe the algorithm ARMAML to compute the concentrated likelihood function $\ell_3(\psi)$, the forecasts and their covariance matrix. The expressions for these are given by (2.4), (2.5) and (2.6). All of these are, again, given a set of ARMA parameters ψ .

III. Bayesian Analysis

For the purposes of this paper, the description of the Bayesian approach for which the algorithm presented here is intended begins conditional on the model (p,q) and the parameters (ϕ,θ) . Given the model and the ARMA parameters, ψ , the prior distribution on the precision r , the reciprocal of the disturbance variance σ_a^2 , is gamma with shape parameter α and scale parameter β , i.e.,

$$(3.1) \quad \pi(r|p,q,\phi,\theta) = \frac{\beta^\alpha}{\Gamma(\alpha)} r^{\alpha-1} e^{-\beta r}, \quad r > 0 .$$

where, again, $r \equiv 1/\sigma_a^2$. Conditional on r , p , q , ϕ and θ , the prior on the process mean is normal with mean γ and precision τr .

Two distributions are now needed. One is the unconditional distribution of the observations z (still, however, conditional on the model and ψ , the ARMA parameters), which is multivariate Student's t , N dimensional (naturally), 2α degrees of freedom, location vector $\gamma \mathbf{1}_N$ and precision matrix $\frac{\alpha}{\beta} (A_N + \tau^{-1} \mathbf{1}_N \mathbf{1}_N^T)^{-1}$ (see DeGroot (1970), p. 59 for the definition of multivariate t). The density function for z is given by

$$(3.2) \quad \frac{\Gamma(N/2 + \alpha) |A_N + \tau^{-1} l_N l_N^T|^{-1/2} (\alpha/\beta)^{N/2}}{\Gamma(\alpha) (2\pi\alpha)^{N/2} \left[1 + \frac{1}{2\beta} (z - \gamma l_N)^T (A_N + \tau^{-1} l_N l_N^T)^{-1} (z - \gamma l_N) \right]^{(N/2 + \alpha)}}$$

The other important distribution is the posterior distribution of the forecasts, which is multivariate Student's t with n dimensions, $2\alpha + N$ degrees of freedom, location vector $b + ca$ and precision matrix

$$(3.3) \quad \left(\frac{2\alpha + N}{2\beta} \right) \left(A_{22} - A_{21} A_N^{-1} A_{12} + (\tau + l_N^T A_N^{-1} l_N) a a^T \right)^{-1}$$

where

$$(3.4) \quad \begin{aligned} a &= l_n - A_{21} A_N^{-1} l_N \\ b &= A_{21} A_N^{-1} z \\ c &= E u | z = (\tau \gamma + z^T A_N^{-1} l_N) / (\tau + l_N^T A_N^{-1} l_N) \end{aligned}$$

Notice that the main difference in the computation of the forecasts and the covariance (precision, in this case) matrix between the maximum likelihood and the Bayesian methods is the computation of a and b , which is not a great hardship. The quadratic form in the distribution of z is slightly more difficult.

IV. The Algorithm ARMAML

The concentrated likelihood function given by 2.4 requires four scalars:

$|A_N|$, $l_N^T A_N^{-1} l_N$, $z^T A_N^{-1} z$, $z^T A_N^{-1} l_N$. Additionally, two others are desired: the forecasts, i.e., the mean of $z_F | z$, which is $\hat{\mu}_N + A_{21} A_N^{-1} (z_N - \hat{\mu}_N)$.

and their covariance matrix: $A_{22} - A_{21}A_N^{-1}A_{12}$. Note that

$$\hat{\mu} = z^T A_N^{-1} 1_N / 1_N^T A_N^{-1} 1_N.$$

The algorithm to be described below is basically a modification of Ansley's (1978, 1979).

Consider the $(N+n)$ by $(N+n)$ matrix B_{N+n}^m partitioned as

$$(4.1) \quad B_{N+n}^m = \begin{pmatrix} B_{m,N} & 0 \\ B_1 & B_n \end{pmatrix}$$

where $B_{m,N}$ is N by N and each row of $(B_1 \ B_n)$ includes $(-\phi_p - \phi_{p-1} \dots - \phi_1 \ 1)$, with zeroes elsewhere, staggered such that B_n is unit lower triangular. Now $B_{m,N}$ is also partitioned (with $m = \max(p,q)$)

$$(4.2) \quad B_{m,N} = \begin{pmatrix} I_m & 0 \\ B_3 & B_{N-m} \end{pmatrix}$$

so that $(B_3 \ B_{N-m})$ has the same structure as $(B_1 \ B_n)$. Note B_n and B_{N-m} have the same structure: unit lower triangular and banded.

Now note the following product:

$$(4.4) \quad B_{N+n}^m A_{N+n} (B_{N+n}^m)^T = \begin{pmatrix} B_{m,N} A_{N,N} B_{m,N}^T & B_{m,N} A_{N,1} B_1^T + B_{m,N} A_{N,2} B_n^T \\ B_1 A_{1,m,N} B_{m,N}^T + B_n A_{21} B_{m,N}^T & B_1 A_{1,1} B_1^T + B_n A_{21} B_1^T + B_1 A_{12} B_n^T + B_n A_{22} B_n^T \end{pmatrix}$$

But this product can be partitioned:

$$(4.4) \quad \begin{matrix} B_{N+m}^m & A_{N+n} & (B_{N+n}^m)^T \\ & & \end{matrix} = \begin{pmatrix} A_m & D_1^T & 0 \\ D_1 & C_1 & E^T \\ 0 & E & C_2 \end{pmatrix} \begin{matrix} m \\ N-m \\ n \end{matrix}$$

where A_m = cov mx of ARMA process and C_1 (N-m by N-m), C_2 , and the last (N+n-m) rows and columns, are the covariance matrices of a MA process of appropriate length with the same parameters (q and θ) as always. Hence all of the matrices are banded and can be mostly zeroes.

The crux of the algorithm is that the matrix $B_{m,N} A_N B_{m,N}^T$ is a symmetric, positive definite banded matrix, with bandwidth not exceeding m , and, hence, has a factorization of the form LDL^T where L is unit lower triangular and banded and D is diagonal. This factorization is slightly different than that used by Ansley (1979) which can be found (in Algol) in Martin and Wilkinson (1971). The LDL^T factorization avoids computing square roots. Note also, since $|B_{m,N}| = 1$, that $|A_N| = |D|$, which is easily computed.

To compute the bilinear forms in 1_N , z , and A_N^{-1} , say, compute $L^{-1} B_{m,N} 1_N$, $L^{-1} B_{m,N} z$, then

$$(4.5) \quad \begin{aligned} (L^{-1} B_{m,N} z)^T D^{-1} (L^{-1} B_{m,N} 1_N) &= z^T B_{m,N} (LDL^T)^{-1} B_{m,N} 1_N \\ &= z^T B_{m,N}^T (B_{m,N} A_N B_{m,N})^{-1} B_{m,N} 1_N \\ &= z^T A_N^{-1} 1_N \end{aligned}$$

and similarly obtain $1_n^T A_n^{-1} 1_n$ and $z^T A_n^{-1} z$.

To compute the forecasts, note that for an arbitrary vector a ,

$$\begin{aligned}
 (4.6) \quad (O \ E^T) L^{-T} D^{-1} (L^{-1} B_{m,N} a) &= (O \ E^T) (LDL^T)^{-1} B_{m,N} a \\
 &= (B_{1N} A_{1N} B_{1N}^T + B_{2N} A_{2N} B_{2N}^T) (B_{m,N} A_{m,N} B_{m,N}^T)^{-1} B_{m,N} a \\
 &= B_{1N} a + B_{2N} A_{2N}^{-1} A_{1N}^{-1} a .
 \end{aligned}$$

$$(4.7) \quad A_{21} A_N^{-1} a = B_n^{-1} [(O \ E^T) L^{-T} D^{-1} (L^{-1} B_{m,N} a) - B_{1N} a]$$

Since the necessary expression is $l_n + A_{21} A_n^{-1} z - \hat{u} A_{21} A_N^{-1} l_N$, substituting l_N and z for a in (4.7) leaves the B_n^{-1} and $B_{1N} a$ as the only extra computation (see below for $(O \ E^T) L^T$).

Now to compute the covariance matrix for the forecasts: $A_{22} - A_{21} A_N^{-1} A_{12}$, note the following expressions:

$$\begin{aligned}
 C_2 - (O \ E^T) (LDL^T)^{-1} \begin{Bmatrix} O \\ E \end{Bmatrix} &= B_{1N} A_{1N} B_{1N}^T + B_{2N} A_{2N} B_{2N}^T + B_{1N} A_{12} B_{2N}^T + B_{2N} A_{22} B_{2N}^T \\
 &\quad - (B_{1N} A_{1N} B_{1N}^T + B_{2N} A_{2N} B_{2N}^T) (B_{m,N} A_{m,N} B_{m,N}^T)^{-1} (B_{m,N} A_{m,N} B_{1N}^T + B_{m,N} A_{m,N} B_{2N}^T) \\
 (4.8) \quad &= B_{1N} A_{22} B_{1N}^T - B_{1N} A_{21} A_N^{-1} A_{12} B_{1N}^T
 \end{aligned}$$

Hence the needed computation is merely

$$(4.9) \quad B_n^{-1} \left[C_2 - \left\{ L^{-1} \begin{Bmatrix} O \\ E^T \end{Bmatrix} \right\}^T D^{-1} \left\{ L^{-1} \begin{Bmatrix} O \\ E^T \end{Bmatrix} \right\} \right] B_n^{-T}$$

Note that the bracketed expression above is the one needed in (4.7).

V. Other Algorithms

The algorithm BARMAD follows ARMAML but computes the expressions needed for (3.2), (3.3) and (3.4). The essential differences are that both vectors a and b of (3.4) must be computed and that a scaled outer product of a must be added onto the covariance matrix calculated in ARMAML.

Algorithm CFARMA yields the covariance function σ of (1.2). The method follows precisely the method of McLeod (1975).

Algorithm GETSET computes the entries of the covariance matrix in (4.4):

A_m is found in AMZ
 D_1 is found in D1
 C_1, E, C_2 are found in CC

Algorithm GETSET is a set-up algorithm (it calls CFARMA); requests for entries of (4.4) are supplied through GRETA.

Algorithm ADJUST implements a device for avoiding overflow in the computation of the determinant by storing it in the form $DET \cdot 2^{IID}$.

Algorithm CRSEPP solves linear equations by the Crout factorization using partial pivoting and is called by CFARMA.

References

- Anderson, T. W. (1970). *The Statistical Analysis of Time Series*, Wiley, New York.
- Ansley, Craig F. (1979). "An Algorithm for the Exact Likelihood of a Mixed Autoregressive-Moving Average Process," *Biometrika* 66, 1, pp. 59-65.
- Ansley, Craig F. (1978). Subroutine ARMA -- Exact Likelihood for Univariate ARMA Processes. Unpublished program documentation.
- DeGroot, Morris H. (1970). *Optimal Statistical Decisions*. McGraw-Hill, New York.
- Martin, R. S. and J. H. Wilkinson. (1971). Symmetric Decomposition of Positive Definite Band Matrices, *Numerische Mathematik* 1, 355-61.
- McLeod, Ian. (1975). Derivation of the Theoretical Autocovariance Function of Autoregressive-Moving Average Time Series, *Applied Statistics* 24, 2, pp. 255-6. Correction: 26, p. 194.

APPENDIX

Listing of Programs

ARMAML
BARMAD
CFARMA
GRETA
ADJUST
GETSET
CRSEPP

```

00010      SUBROUTINE ARMAML(Z,P,Q,N,PHI,THETA,ZHM,DET,IID,QF,F,COVF,NS)
00020 C    IN  Z,P,Q,PHI,THETA
00030 C          N=LENGTH OF SERIES;  NS=NO. OF FORECASTS
00040 C    OUT ZHM=MU*HAT  ESTIMATE OF MEAN
00050 C    OUT  DET*2**IID = DETERMINANT OF A_N
00060 C    OUT  QF = Q
00070 C    OUT  F = VECTOR OF FORECASTS
00080 C    OUT  COVF = COVARIANCE MATRIX OF FORECASTS
00090      REAL Z(500),PHI(10),THETA(10),F(10),COVF(10,10)
00100      REAL BZ(500),BO(500),E(10,10)
00110      INTEGER P,Q
00120      REAL L(100)
00130      LOC(I,J)=MP1*MOD(I-1,MP1)+J-I+MP1
00140 C    D LIES ON DIAGONAL OF L
00150 C    L STORED SLIDING, AND ONLY RECENT PART
00160      M=MAX0(P,Q)
00170      MP1=M+1
00180      CALL GETSET(PHI,THETA,P,Q)
00190 C    GETSET PREPARES VALUES OF A,D_1,C_1,C_2 AND E
00200 C    USES COMMON AREA NAMED XARMAX *****
00210 C    P,Q < 10 < N      NS <= 10
00220      ODO=0.
00230      ODZ=0.
00240      ZDZ=0.
00250 C    CREATE BZ  AND  B*ONE
00260      DO 2I=1,M
00270      BZ(I)=Z(I)
00280 2    BO(I)=1.
00290      S=1.
00300      IF (P.EQ.0) GO TO 6
00310      DO 4I=1,P
00320 4    S=S-PHI(I)
00330 6    DO 8I=MP1,N
00340      BZ(I)=Z(I)
00350      IF(P.EQ.0) GO TO 8
00360      DO 9J=1,P
00370 9    BZ(I)=BZ(I)-PHI(J)*Z(I-J)
00380 8    BO(I)=S
00390 C    BZ HOLDS B_M,N * Z
00400 C    BO HOLDS B_M,N * ONE

```

```

00410      DET=1.
00420      IID=0
00430      DO 12I=1,N
00440      W=GRETA(I,I)
00450 C    GRETA GETS VALUES OF B M,N * A N * (B M,N)-TRANSPOSE
00460 C    AKA  A M, C 1, C 2,D I, AND E
00470      IF (I.EQ.1) GO TO 14
00480      IMQ=1
00490      IF(I.LE.M) GO TO 11
00500      IF(Q.EQ.0) GO TO 14
00510      IMQ=I-Q
00520 11    IM1=I-1
00530      DO 15J=IMQ,IM1
00540      S=GRETA(I,J)
00550      IF(IMQ.GE.J) GO TO 15
00560      JM1=J-1
00570      DO 16K=IMQ,JM1
00580 16    S=S-L(LOC(J,K))*L(LOC(I,K))
00590 15    L(LOC(I,J))=S
00600      DO 17J=IMQ,IM1
00610      S=L(LOC(I,J))
00620      L(LOC(I,J))=S/L(LOC(J,J))
00630      W=W-S*L(LOC(I,J))
00640      BZ(I)=BZ(I)-L(LOC(I,J))*BZ(J)
00650 17    BO(I)=BO(I)-L(LOC(I,J))*BO(J)
00660 14    CONTINUE
00670      L(LOC(I,I))=W
00680 C    COMPUTE BILINEAR FORMS
00690      ODO=ODO+BO(I)*BO(I)/W
00700      ZDZ=ZDZ+BZ(I)*BZ(I)/W
00710      ODZ=ODZ+BO(I)*BZ(I)/W
00720      DET=DET*W
00730      CALL ADJUST(DET,IID)
00740 12    CONTINUE
00750 C    GET ESTIMATE OF MEAN
00760      ZHM=ODZ/ODO
00770 C    GET QUADRATIC FORM FOR LIKELIHOOD
00780      QF=ZDZ-ODZ*ZHM
00790 C    NOW START ON FORECASTS
00800      DO 30 I=1,NS
00810      F(I)=0.
00820      DO 30 J=1,I
00830 C    INITIALIZE COV MX FOR FORECASTS
00840 30    COVF(I,J)=GRETA(N+I,N+J)
00850      IF (Q.EQ.0) GO TO 39
00860 C    GET D-INV L-INV ( 0 E )
00870      DO 32K=1,Q
00880      DO 33I=K,Q
00890      E(I,K)=GRETA(N+K,N-Q+I)
00900      IF(I.EQ.K) GO TO 33
00910      IM1=I-1
00920      DO 34J=K,IM1
00930 34    E(I,K)=E(I,K)-L(LOC(N-Q+I,N-Q+J))*E(J,K)
00940 33    CONTINUE

```

```

00950      DO 36I=K,Q
00960      S=E(I,K)
00970      KM1=K-1
00980      IF(KM1.EQ.0) GO TO 38
00990      DO 37J=1,KM1
01000      37 COVF(K,J)=COVF(K,J)-S*E(I,J)
01010      38 E(I,K)=E(I,K)/L(LOC(N-Q+I,N-Q+I))
01020      COVF(K,K)=COVF(K,K)-S*E(I,K)
01030      36 F(K)=F(K)+E(I,K)*(BZ(N-Q+I)-ZHM*BO(N-Q+I))
01040      32 CONTINUE
01050      39 CONTINUE
01060 C     SYMMETRIZE
01070      DO 51I=1,NS
01080      DO 51J=1,I
01090      51 COVF(J,I)=COVF(I,J)
01100      IF(P.EQ.0) GO TO 48
01110      DO 41I=1,P
01120      DO 41J=I,P
01130      41 F(I)=F(I)+PHI(J)*(Z(N+I-J)-ZHM)
01140 C     FINISH FORECASTS WITH B_NS-INV
01150      DO 42I=1,NS
01160      IF(I.EQ.1) GO TO 42
01170      K=MIN0(I-1,P)
01180      DO 43 J=1,K
01190      43 F(I)=F(I)+PHI(J)*F(I-J)
01200      42 CONTINUE
01210 C     FORECASTS NEAR DONE      NOW GET COV MX
01220 C     B_NS-INV IN FRONT
01230      DO 52J=1,NS
01240      DO 52I=1,NS
01250      LL=MIN0(I-1,P)
01260      IF(I.EQ.1) GO TO 52
01270      DO 53K=1,LL
01280      53 COVF(I,J)=COVF(I,J)+PHI(K)*COVF(I-K,J)
01290      52 CONTINUE
01300 C     B_NS-INV TRANSP IN BACK
01310      DO 56J=1,NS
01320      DO 56I=1,NS
01330      IF(I.EQ.1) GO TO 56
01340      LL=MIN0(I-1,P)
01350      DO 55K=1,LL
01360      55 COVF(J,I)=COVF(J,I)+PHI(K)*COVF(J,I-K)
01370      56 CONTINUE
01380      48 CONTINUE
01390      DO 49I=1,NS
01400      49 F(I)=F(I)+ZHM
01410      RETURN
01420      END

```

```

01430      SUBROUTINE BARMAD(Z,P,Q,N,PHI,THETA,EMGZ,DB,IID,QF,F,COVF,NS,GAM,
01440      ITAU)
01450 C    IN   Z,P,Q,PHI,THETA
01460 C          N=LENGTH OF SERIES; NS=NO. OF FORECASTS
01470 C    IN   PRIOR ON MEAN IS NORMAL(GAM,TAU)
01480 C    OUT  EMGZ=MEAN OF POSTEIOR OF MU
01490 C    OUT  DB*2**IID = DETERMINANT FOR DENSITY OF Z
01500 C    OUT  QF = QUADRATIC FORM FOR DENSITY OF Z
01510 C    OUT  F = VECTOR OF FORECASTS
01520 C    OUT  COVF = COVARIANCE MATRIX OF FORECASTS
01530      REAL Z(500),PHI(10),THETA(10),F(10),COVF(10,10)
01540      REAL BZ(500),BO(500),E(10,10),AS(10),BS(10)
01550      INTEGER P,Q
01560      REAL L(100)
01570      LOC(I,J)=MP1*MOD(I-1,MP1)+J-I+MP1
01580 C    D LIES ON DIAGONAL OF L
01590 C    L STORED SLIDING, AND ONLY RECENT PART
01600      M=MAX0(P,Q)
01610      MP1=M+1
01620      CALL GETSET(PHI,THETA,P,Q)
01630 C    GETSET PREPARES VALUES OF A,D 1,C 1,C 2 AND E
01640 C    USES COMMON AREA NAMED XARMAX *****
01650 C    P,Q < 10 < N      NS <= 10
01660      ODO=0.
01670      ODZ=0.
01680      ZDZ=0.
01690 C    CREATE BZ AND B*ONE
01700      DO 2I=1,M
01710      BZ(I)=Z(I)
01720 2    BO(I)=1.
01730      S=1.
01740      IF (P.EQ.0) GO TO 6
01750      DO 4I=1,P
01760 4    S=S-PHI(I)
01770 6    DO 8I=MP1,N
01780      BZ(I)=Z(I)
01790      IF(P.EQ.0) GO TO 8
01800      DO 9J=1,P
01810 9    BZ(I)=BZ(I)-PHI(J)*Z(I-J)
01820 8    BO(I)=S
01830 C    BZ HOLDS B_M,N * Z
01840 C    BO HOLDS B_M,N * ONE
01850      DB=1.
01860      IID=0
01870      DO 12I=1,N
01880      W=GRETA(I,I)
01890 C    GRETA GETS VALUES OF B M,N * A N * (B_M,N)-TRANSPOSE
01900 C    AKA  A M, C 1, C 2,D I, AND E
01910      IF (I.EQ.1) GO TO 14
01920      IMQ=1
01930      IF(I.LE.M) GO TO 11
01940      IF(Q.EQ.0) GO TO 14
01950      IMQ=I-Q
01960 11    IM1=I-1

```

```

01970      DO 15J=IMQ,IM1
01980      S=GRETA(I,J)
01990      IF(IMQ.GE.J) GO TO 15
02000      JM1=J-1
02010      DO 16K=IMQ,JM1
02020  16  S=S-L(LOC(J,K))*L(LOC(I,K))
02030  15  L(LOC(I,J))=S
02040      DO 17J=IMQ,IM1
02050      S=L(LOC(I,J))
02060      L(LOC(I,J))=S/L(LOC(J,J))
02070      W=W-S*L(LOC(I,J))
02080      BZ(I)=BZ(I)-L(LOC(I,J))*BZ(J)
02090  17  BO(I)=BO(I)-L(LOC(I,J))*BO(J)
02100  14  CONTINUE
02110      L(LOC(I,I))=W
02120 C    COMPUTE BILINEAR FORMS
02130      ODO=ODO+BO(I)*BO(I)/W
02140      ZDZ=ZDZ+BZ(I)*BZ(I)/W
02150      ODZ=ODZ+BO(I)*BZ(I)/W
02160      DB=DB*W
02170      CALL ADJUST(DB,IID)
02180  12  CONTINUE
02190 C    GET MEAN OF POSTERIOR OF MU**** E (MU GIVEN Z)
02200      EMGZ=(GAM*TAU+ODZ)/(TAU+ODO)
02210 C    GET QUADRATIC FORM FOR DENSITY
02220      QF=ZDZ+GAM*GAM*TAU-EMGZ*(GAM*TAU+ODZ)
02230 C    ADJUST DETERMINANT FOR BAYESIAN ANALYSIS
02240      DB=DB*(1.+ODO/TAU)
02250 C    NOW START ON FORECASTS
02260      DO 30 I=1,NS
02270      F(I)=0.
02280      AS(I)=0.
02290      BS(I)=0.
02300      DO 30 J=1,I
02310 C    INITIALIZE COV MX FOR FORECASTS
02320  30  COVF(I,J)=GRETA(N+I,N+J)
02330      IF(Q.EQ.0) GO TO 39
02340 C    GET D-INV L-INV ( 0 E )
02350      DO 32K=1,Q
02360      DO 33I=K,Q
02370      E(I,K)=GRETA(N+K,N-Q+I)
02380      IF(I.EQ.K) GO TO 33
02390      IM1=I-1
02400      DO 34J=K,IM1
02410  34  E(I,K)=E(I,K)-L(LOC(N-Q+I,N-Q+J))*E(J,K)
02420  33  CONTINUE

```

```

02430      DO 36I=K,Q
02440      S=E(I,K)
02450      KM1=K-1
02460      IF(KM1.EQ.0) GO TO 38
02470      DO 37J=1,KM1
02480      37 COVF(K,J)=COVF(K,J)-S*E(I,J)
02490      38 E(I,K)=E(I,K)/L(LOC(N-Q+I,N-Q+I))
02500      COVF(K,K)=COVF(K,K)-S*E(I,K)
02510      AS(K)=AS(K)+E(I,K)*BO(N-Q+I)
02520      36 BS(K)=BS(K)+E(I,K)*BZ(N-Q+I)
02530      32 CONTINUE
02540      39 CONTINUE
02550 C      SYMMETRIZE
02560      DO 51I=1,NS
02570      DO 51J=1,I
02580      51 COVF(J,I)=COVF(I,J)
02590      IF(P.EQ.0) GO TO 48
02600      DO 41I=1,P
02610      DO 41J=I,P
02620      AS(I)=AS(I)+PHI(J)
02630      41 BS(I)=BS(I)+PHI(J)*Z(N+I-J)
02640 C      FINISH FORECASTS WITH B_NS-INV
02650      DO 42I=1,NS
02660      IF(I.EQ.1) GO TO 42
02670      K=MIN0(I-1,P)
02680      DO 43 J=1,K
02690      AS(I)=AS(I)+PHI(J)*AS(I-J)
02700      43 BS(I)=BS(I)+PHI(J)*BS(I-J)
02710      42 CONTINUE
02720 C      FORECASTS NEAR DONE      NOW GET COV MX
02730 C      B_NS-INV IN FRONT
02740      DO 52J=1,NS
02750      DO 52I=1,NS
02760      LL=MIN0(I-1,P)
02770      IF(I.EQ.1) GO TO 52
02780      DO 53K=1,LL
02790      53 COVF(I,J)=COVF(I,J)+PHI(K)*COVF(I-K,J)
02800      52 CONTINUE
02810 C      B_NS-INV TRANSP IN BACK
02820      DO 56J=1,NS
02830      DO 56I=1,NS
02840      IF(I.EQ.1) GO TO 56
02850      LL=MIN0(I-1,P)
02860      DO 55K=1,LL
02870      55 COVF(J,I)=COVF(J,I)+PHI(K)*COVF(J,I-K)
02880      56 CONTINUE
02890      48 CONTINUE
02900      DO 49I=1,NS
02910      49 F(I)=EMGZ*(1.-AS(I))+BS(I)
02920      W=TAU+ODO
02930      DO 58 I=1,NS
02940      DO 58 J=1,NS
02950      58 COVF(I,J)=COVF(I,J)+AS(I)*AS(J)/W
02960      RETURN
02970      END

```

```

02980      SUBROUTINE CFARMA(PHI,THETA,P,Q,X)
02990 C      COMPUTES COVARIANCE FUNCTION, SIGMA(0),...,SIGMA(M)
03000 C      FOR ARMA PROCESS USING MCLEOD'S ALGORITHM
03010 C      CRSEPP IS LINEAR EQUATIONS SOLVER
03020      REAL PHI(10),THETA(10),A(10,10),X(10),C(10)
03030      INTEGER P,Q,R,RP1,QP1
03040      R=MAX0(P,Q)
03050      RP1=R+1
03060      C(1)=1.
03070      IF(Q.EQ.0) GO TO 4
03080      DO 2K=1,Q
03090      J=K+1
03100      C(J)=-THETA(K)
03110      IF(P.EQ.0) GO TO 2
03120      L=MIN0(P,K)
03130      DO 3I=1,L
03140      3 C(J)=C(J)+PHI(I)*C(J-I)
03150      2 CONTINUE
03160      4 CONTINUE
03170      X(1)=C(1)
03180      DO 9I=2,10
03190      9 X(I)=0.
03200      IF(Q.EQ.0) GO TO 6
03210      DO 7I=1,Q
03220      7 X(1)=X(1)-THETA(I)*C(I+1)
03230      DO 8K=1,Q
03240      J=K+1
03250      DO 8I=K,Q
03260      8 X(J)=X(J)-THETA(I)*C(I-K+1)
03270      6 IF(P.EQ.0) RETURN
03280      DO 12I=1,RP1
03290      DO 12J=1,RP1
03300      12 A(I,J)=0.
03310 C      NB THIS IS NEG OF MACLEOD'S A
03320      DO 14I=1,RP1
03330      A(I,I)=1.
03340      DO 14J=1,P
03350      L=IABS(I-J-1)+1
03360      14 A(I,L)=A(I,L)-PHI(J)
03370      CALL CRSEPP(A,RP1,S,X)
03380      RETURN
03390      END
03400      REAL FUNCTION GRETA(I,J)
03410 C      GRETA GETS (QUICKLY) VALUES OF B_M,N * A_N * (B_M,N)-TRANSPOSE
03420      REAL AM(10),D1(10),CC(10)
03430      COMMON /XARMAX/M,IQ1,AM,CC,D1
03440      K=I-J+1
03450      GRETA=0.
03460      IF(K.LE.IQ1) GRETA=CC(K)
03470      IF(J.GT.M) RETURN
03480      IF(I.GT.M) GO TO 4
03490      GRETA=AM(K)
03500      RETURN
03510      4 GRETA=D1(K-1)
03520      RETURN
03530      END

```

```

03540      SUBROUTINE ADJUST(D,I)
03550 C    ADJUST KEEPS DET FROM EXPLODING
03560      IF(D.LE.0.) GO TO 6
03570      3  IF(D.GE.1.) GO TO 4
03580      D=D*16.
03590      I=I-4
03600      GO TO 3
03610      4  IF(D.LE.16.) RETURN
03620      D=D/16.
03630      I=I+4
03640      GO TO 4
03650      6  I=-2147483644
03660      RETURN
03670      END
03680      SUBROUTINE GETSET(PHI,THETA,P,Q)
03690      REAL PHI(10),THETA(10),AM(10),CC(10),D1(10)
03700      INTEGER P,Q
03710      COMMON /XARMAX/M,IQ1,AM,CC,D1
03720      IQ1=Q+1
03730      M=MAX0(P,Q)
03740      CALL CFARMA(PHI,THETA,P,Q,AM)
03750      CC(1)=1.
03760      IF(Q.EQ.0) GO TO 6
03770      DO 2I=1,Q
03780      2  CC(1)=CC(1)+THETA(I)**2
03790      DO 4I=1,Q
03800      IP1=I+1
03810      CC(I+1)=-THETA(I)
03820      IF(I.EQ.Q) GO TO 4
03830      DO 5J=IP1,Q
03840      5  CC(I+1)=CC(IP1)+THETA(J)*THETA(J-I)
03850      4  CONTINUE
03860      6  CONTINUE
03870      DO 7I=1,M
03880      D1(I)=AM(I+1)
03890      IF(P.EQ.0) GO TO 7
03900      DO 8K=1,P
03910      8  D1(I)=D1(I)-PHI(K)*AM(IABS(I-K)+1)
03920      7  CONTINUE
03930      RETURN
03940      END

```

```

03950      SUBROUTINE CRSEPP(A,N,D,V)
03960      REAL A(10,10),V(10)
03970      INTEGER R
03980      D=1.
03990      DO 2K=1,N
04000      DO 3I=K,N
04010      S=A(I,K)
04020      IF(K.EQ.1) GO TO 3
04030      KM1=K-1
04040      DO 4L=1,KM1
04050      4  S=S-A(I,L)*A(L,K)
04060      3  A(I,K)=S
04070      S=0.
04080      R=0
04090      DO 5I=K,N
04100      IF(ABS(A(I,K)).LE.S) GO TO 5
04110      R=I
04120      S=ABS(A(I,K))
04130      5  CONTINUE
04140      IF(S.EQ.0.) STOP
04150      D=D*A(R,K)
04160      IF(R.EQ.K) GO TO 1
04170      D=-D
04180      DO 6J=1,N
04190      T=A(R,J)
04200      A(R,J)=A(K,J)
04210      6  A(K,J)=T
04220      T=V(R)
04230      V(R)=V(K)
04240      V(K)=T
04250      1  IF(K.EQ.N) GO TO 2
04260      KP1=K+1
04270      DO 7J=KP1,N
04280      S=A(K,J)
04290      IF(K.EQ.1) GO TO 7
04300      DO 8L=1,KM1
04310      8  S=S-A(K,L)*A(L,J)
04320      7  A(K,J)=S/A(K,K)
04330      2  CONTINUE
04340      V(1)=V(1)/A(1,1)
04350      DO 9K=2,N
04360      KM1=K-1
04370      DO 10I=1,KM1
04380      10 V(K)=V(K)-A(K,I)*V(I)
04390      9  V(K)=V(K)/A(K,K)
04400      DO 11J=2,N
04410      K=N+1-J
04420      KP1=K+1
04430      DO 12I=KP1,N
04440      12 V(K)=V(K)-A(K,I)*V(I)
04450      11 CONTINUE
04460      RETURN
04470      END

```