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)

\[
(1.1) \quad z_t - \phi_1 z_{t-1} - \cdots - \phi_p z_{t-p} = \mu + \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q}
\]

where \( a_t \)'s are iid normal random variables each having a normal distribution with mean zero and variance \( \sigma_a^2 \). A finite segment of this process is observed: \( z = (z_1, \ldots, z_N)^T \) which then has an \( N \)-dimensional multivariate normal distribution with mean vector \( \mu_1^N \) 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 \( \sigma \) characterizes this stationary time series process. By taking variances of both sides of (1.1), the covariance function \( \sigma \) can be determined

\[
\text{Cov} (z_t - \phi_1 z_{t-1} - \cdots - \phi_p z_{t-p}, z_{t+s} - \phi_1 z_{t+s-1} - \cdots - \phi_q z_{t+s-q})
\]

\[
(1.3) \quad = \sum_{i=0}^{p} \sum_{j=0}^{q} \phi_i \phi_j \sigma(|s+i-j|) = \sum_{j=0}^{q} \theta_j \theta_{j+s} \quad \forall s \geq 0
\]

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 \( z_F = (z_{N+1}, \ldots, z_{N+n})^T \), conditional on \( z_N \), has a multivariate normal distribution with mean \( \mu_{1n} + A_{21}A^{-1}_N(z_N - \mu_{1n}) \) and covariance matrix \( \sigma^2_{a \times nN} \) where

\[
A_{N+n} = \begin{bmatrix}
A_N & A_{12} \\
A_{21} & A_{22}
\end{bmatrix}_n
\]

and \( A_{nN}^{-1} = A_{22} - A_{21}A^{-1}_N A_{12} \). In the notation to be used here,

\[
(z_F | z, \mu, \sigma^2_a)^{-N}(\mu_{1n} - A_{21}A^{-1}_N(z_N - \mu_{1n}), \sigma^2_{a \times N}) .
\]

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

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

\[
p(z | \mu, \sigma^2_a, \psi) = (2\pi)^{-N/2} |A_N|^{-1/2}(\sigma^2_a)^{-N/2} \exp \left\{ -\frac{1}{2\sigma^2_a} (z-\mu_{1n})^T A^{-1}_N (z-\mu_{1n}) \right\}
\]

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

\[
\ell(\mu, \sigma^2_a, \psi) = c_1 - N/2 \ln \sigma^2_a - \frac{1}{2} \ln |A_N| - \frac{1}{2\sigma^2_a} (z-\mu_{1n})^T A^{-1}_N (z-\mu_{1n})
\]
where \( C_1 \)'s are constants. To concentrate on \( \mu \), setting \( \partial \lambda(\mu, \sigma^2, \psi)/\partial \mu = 0 \) yields:

\[
\hat{\mu} = z A_N^{-1} \frac{T}{N^2} A_N^{-1} N
\]

Concentrating now on \( \sigma^2_a \), setting \( \partial \lambda(\mu, \sigma^2_a, \psi)/\partial \sigma^2_a \) to zero

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

which yields

\[
\sigma^2_a = (z - \hat{\mu} \frac{T}{N^2} A_N^{-1} (z - \hat{\mu})) / N
\]

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

\[
\ell_3(\psi) = C_2 - N/2 \ln Q(\psi) - \frac{1}{2} \ln |A_N|
\]

The MLE for the remaining parameters, \( \hat{\psi} \), is that vector \( \psi \) which maximizes \( \ell_3(\psi) \), which must be found numerically. Note that for identifiability purposes, the values of the parameter vector \( \psi \) 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 \( (z_F | \mu, \sigma^2_a, z, \psi) \):

\[
z_F = \hat{\mu} \frac{T}{N^2} A_N^{-1} (\psi)(z - \hat{\mu} \frac{T}{N^2} A_N^{-1} (z - \hat{\mu}))
\]
Likewise, the desired covariance matrix for the forecasts treats $\psi$ as given:

\[
\text{Cov}(z_{\hat{r}}) = \left[ A_{22}(\hat{\psi}) - A_{21}(\hat{\psi})A_{12}^{-1}(\hat{\psi})A_{12}(\hat{\psi}) \right] \sigma_a^2.
\]

The objective of this paper is to describe the algorithm ARMAML to compute the concentrated likelihood function $\lambda_3(\psi)$, 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 $\psi$.

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 $(\phi, \theta)$. Given the model and the ARMA parameters, $\psi$, the prior distribution on the precision $\tau$, the reciprocal of the disturbance variance $\sigma_a^2$, is gamma with shape parameter $\alpha$ and scale parameter $\beta$, i.e.,

\[
\pi(\tau | p, q, \phi, \theta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \tau^{\alpha-1} e^{-\beta \tau}, \tau > 0.
\]

where, again, $\tau = 1/\sigma_a^2$. Conditional on $\tau, p, q, \phi$ and $\theta$, the prior on the process mean is normal with mean $\gamma$ and precision $\tau \tau$.

Two distributions are now needed. One is the unconditional distribution of the observations $z$ (still, however, conditional on the model and $\psi$, the ARMA parameters), which is multivariate Student's $t$, $N$ dimensional (naturally), $2\alpha$ degrees of freedom, location vector $\gamma_1^N$ and precision matrix

\[
\frac{\alpha}{\beta} (A_N + \tau^{-1} T_N N_N^{-1})^{-1}
\]

(see DeGroot (1970), p. 59 for the definition of multivariate $t$). The density function for $z$ is given by
\[
\frac{\Gamma(N/2+\alpha)|A_N|^{-1}1_N^T}{\Gamma(\alpha)(2\pi)^{N/2}} \left[ 1 + \frac{1}{2\beta} (z-\gamma 1_N)^T (A_N+\tau^{-1}1_N^T)^{-1}(z-\gamma 1_N) \right]^{(N/2+\alpha)}
\]

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

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

where

\[(3.4) \quad a = 1_N - A_{21} A_N^{-1} 1_N \]
\[b = A_{21} A_N^{-1} z \]
\[c = E u | z = (\tau + z^T A_N^{-1} 1_N)/(\tau + 1_N^T A_N^{-1} 1_N) \]

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: \( |A_N|, 1_N^T A_N^{-1} 1_N, z^T A_N^{-1} z, z^T A_N^{-1} 1_N \). Additionally, two others are desired: the forecasts, i.e., the mean of \( z_F | z \), which is \( \tilde{u} + A_{21} A_N^{-1} (z_N - \tilde{u}_N) \).
and their covariance matrix: $A_{22} - A_{21}A_N^{-1}A_{12}$. Note that

$$\hat{\mu} = T_{1}^{-1}A_{1}^{-1}A_{1}^{-T}T_{1}^{-1}N_A^{-1}.$$  

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

\[(4.1)\]

$$B_{N+n}^{m} = \begin{pmatrix}
B_{m,N} & 0 \\
B_{l} & B_{n}
\end{pmatrix}$$

where $B_{m,N}$ is $N$ by $N$ and each row of $(B_{l} \ B_{n})$ includes

$$(-\phi_{p}, \phi_{p-1}, \ldots, \phi_{1}, 1),$$

with zeroes elsewhere, staggered such that $B_{n}$ is unit lower triangular. Now $B_{m,N}$ is also partitioned (with $m = \max(p,q)$)

\[(4.2)\]

$$B_{m,N} = \begin{pmatrix}
I_{m} & 0 \\
B_{3} & B_{N-m}
\end{pmatrix}$$

so that $(B_{3} \ B_{N-m})$ has the same structure as $(B_{l} \ B_{n})$. Note $B_{n}$ and $B_{N-m}$ have the same structure: unit lower triangular and banded.

Now note the following product:

\[(4.4)\]

$$B_{N+n}^{m}(A_{N+n})^{T} = \begin{pmatrix}
B_{m,N}A_N T_{N,m,N} & B_{m,N}A_N T_{1} + B_{m,N}A_{12} B_{n} \\
B_{1} A_{N} T_{1} + B_{1} A_{12} B_{n} & B_{1} A_{N} T_{1} + B_{1} A_{12} B_{n} + B_{1} A_{12} B_{n}
\end{pmatrix}$$
But this product can be partitioned:

\[
B_m^{N+m}A_{N+n}(B_m^{N+n})^T = \begin{pmatrix}
A_m & D_m^T & 0 \\
   & D_1 & C_1 \\
   & C_2 & E
\end{pmatrix}
\]

\[
\begin{array}{c c c}
m & N-m & n
\end{array}
\]

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

The crux of the algorithm is that the matrix \( B_{m,N}^N A_{N,m,N}^T \) 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 \( L_{m,N}^{-1}N \), \( z \), and \( A_{N}^{-1} \), say, compute

\[
L_{m,N}^{-1}L_{m}^{-1}B_{m,N}^{-1}N \quad \text{and} \quad L_{m,N}^{-1}L_{m}^{-1}z .
\]

\[
(L_{m,N}^{-1}B_{m,N}^{-1})^{-1}D^{-1}(L_{m,N}^{-1}B_{m,N}^{-1}) = z_{B_{m,N}^{-1}}(L_{m,N}^{-1}D_{m,N}^{-1})^{-1}B_{m,N}^{-1}N
\]

\[
= z_{B_{m,N}^{-1}}(B_{m,N}^{-1}A_{N}^{-1}B_{m,N}^{-1})^{-1}B_{m,N}^{-1}N
\]

\[
= z_{A_{N}^{-1}}N
\]

(4.5)

and similarly obtain \( L_{N}^{-1}A_{N}^{-1}N \) and \( z_{A_{N}^{-1}}z . \)
To compute the forecasts, note that for an arbitrary vector \( a \),

\[
(0 \ E^T) L^{-T} D^{-1} (L^{-1} B, N a^T) = (0 \ E^T) (L D L^T)^{-1} B, N a^T
\]

\[
= (B_1 A^T_{1 m, N} + B_2 A^T_{2 m, N}) (B, N A^T_{m, N} - 1 B, N a^T)
\]

\[
= B, a + B, A_{21} A_{12}^{-1} a.
\]

(4.7) \[
A_{21} A_{12}^{-1} a = B_n^{-1} [ (0 \ E^T) L^{-T} D^{-1} (L^{-1} B, N a) - B, a ]
\]

Since the necessary expression is \( 1_n + A_{21} A_{12}^{-1} z - \hat{A} A_{21} A_{12}^{-1} N \),
substituting \( 1_N \) and \( z \) for \( a \) in (4.7) leaves the \( B_n^{-1} \) and \( B, a \) as
the only extra computation (see below for \( (0 \ E^T) L^{-T} \)).

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

\[
C_2 = (0 \ E^T) (L D L^T)^{-1} [0] = B_1 A_{1} B^T_{1} + B_2 A_{2} B^T_{2} + B, A_{12} B^T_{2} + B, A_{22} B^T_{2}
\]

\[
- (B_1 A_{1} B^T_{1 m, N} + B_2 A_{2} B^T_{2 m, N}) (B, N A_{1} B^T_{m, N} - 1 (B, N A_{1} B^T_{m, N} + B, N A_{12} B^T_{n})
\]

(4.8) \[
= B_n^{-1} A_{22} B^T_{n} - B_n^{-1} A_{21} A_{12}^{-1} A_{12} B^T_{n}
\]

Hence the needed computation is merely

(4.9) \[
B_n^{-1} \left[ C_2 - \left[ L^{-1} \begin{bmatrix} 0 \\ E^T \end{bmatrix} \right] D^{-1} \left[ L^{-1} \begin{bmatrix} 0 \\ E^T \end{bmatrix} \right] \right] 3_n^{-T}
\]

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 \( \sigma \) of (1.2). The method follows precisely the method of McLeod (1975).

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

\[
\begin{align*}
A_m & \text{ is found in AMZ} \\
D_1 & \text{ is found in DL} \\
C_1, E, C_2 & \text{ are found in CC}
\end{align*}
\]

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 \( \text{DET} \cdot 2^{\text{IID}} \).

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


APPENDIX

Listing of Programs

ARMAML
BARMAD
CFARMA
GRETA
ADJUST
GETSET
CRSEPP
SUBROUTINE ARMAML(Z,P,Q,N,PHI,THETA,ZHM,DET,IID,QF,F,COVF,NS)

C IN Z,P,Q,PHI,THETA
C N=LENGTH OF SERIES; NS=NO. OF FORECASTS
C OUT ZHM=MU*HAT ESTIMATE OF MEAN
C OUT DET*2**IID = DETERMINANT OF A_N
C OUT QF = Q
C OUT F = VECTOR OF FORECASTS
C OUT COVF = COVARIANCE MATRIX OF FORECASTS
C REAL Z(500),PHI(10),THETA(10),F(10),COVF(10,10)
C REAL BZ(500),BO(500),E(10,10)
C INTEGER P,Q
C REAL L(100)
C LOC(I,J)=MP1*MOD(I-1,MP1)+J-I+MP1
C D LIIES ON DIAGONAL OF L
C L STORED SLIDING, AND ONLY RECENT PART
C M=MAX0(P,Q)
C MP1=M+1
C CALL GETSET(PHI,THETA,P,Q)
C GETSET PREPARES VALUES OF A,D_1,C_1,C_2 AND E
C USES COMMON AREA NAMED XARMA** ***************
C P,Q < 10 < N NS <= 10
C ODO=0.
C ODZ=0.
C ZDZ=0.
C CREATE BZ AND B*ONE
DO 2I=1,M
BZ(I)=Z(I)
2 I=1.
S=1.
C IF (P,EQ.0) GO TO 6
DO 4I=1,P
4 S=S-PHI(I)
6 DO 8I=MP1,N
BZ(I)=Z(I)
8 I=1,P
9 BZ(I)=BZ(I)-PHI(J)*Z(I-J)
C BZ HOLDS B_M,N * Z
C BO HOLDS B_M,N * ONE
DET=1.
IID=0
DO 12 I=1,N
W=GRETA(I,I)
C GRETA GETS VALUES OF B,M,N * A,N * (B,M,N)-TRANSPOSE
C AKA A,M,C1,C2,D,T,AND E-
IF (I.EQ.1) GO TO 14
IMQ=1
IF(I.LE.M) GO TO 11
IF(Q.EQ.0) GO TO 14
IMQ=I-Q
11 IM1=I-1
DO 15 J=IMQ,IM1
S=GRETA(I,J)
IF(IMQ.GE.J) GO TO 15
JM1=J-1
DO 16 K=IMQ,JM1
S=S-L(LOC(J,K))*L(LOC(I,K))
16 L(LOC(I,J))=S
DO 17 J=IMQ,IM1
S=L(LOC(I,J))
L(LOC(I,J))=S/L(LOC(J,J))
W=W-S*L(LOC(I,J))
17 BO(I)=BO(I)-L(LOC(I,J))*BO(J)
CONTINUE
L(LOC(I,I))=W
C COMPUTE BILINEAR FORMS
ODO=ODO+BO(I)*BO(I)/W
ZDZ=ZDZ+BZ(I)*BZ(I)/W
ODZ=ODZ+BO(I)*BZ(I)/W
DE=DE*W
CALL ADJUST(DET,IID)
12 CONTINUE
C GET ESTIMATE OF MEAN
ZHM=ODZ/ODO
C GET QUADRATIC FORM FOR LIKELIHOOD
QF=ZDZ-ODZ*ZHM
C NOW START ON FORECASTS
DO 30 I=1,NS
F(I)=0.
30 I=I+1
DO 33 J=1,I
C INITIALIZE COV MX FOR FORECASTS
COVF(I,J)=GRETA(N+I,N+J)
IF(Q.EQ.0) GO TO 39
30 DO 32 K=1,Q
32 DO 33 I=K,Q
E(I,K)=GRETA(N+K,N-Q+I)
33 IF(I.EQ.K) GO TO 33
IM1=I-1
DO 34 J=K,IM1
E(I,K)=E(I,K)-L(LOC(N-Q+I,N-Q+J))*E(J,K)
34 IM1=I-1
CONTINUE
DO 36 I=K,Q
S=E(I,K)
K1=K-1
IF(K1.EQ.0) GO TO 38
DO 37 J=1,KM1
C0V(F(K,J)=C0V(F(K,J)-S*E(I,J)
E(I,K)=E(I,K)/L(LOC(N-Q+I,N-Q+I))
C0V(F(K,K)=C0V(F(K,K)-S*E(I,K)
F(K)=F(K)+E(I,K)*((BZ(N-Q+I)-ZHM*BO(N-Q+I))
CONTINUE
CONTINUE
SYMMETRIZE
DO 51 I=1,NS
DO 51 J=1,I
C0V(F(J,I)=C0V(F(I,J)
IF(P.EQ.0) GO TO 48
DO 41 I=1,P
DO 41 J=1,P
F(I)=F(I)+PHI(J)*(Z(N+I-J)-ZHM)
CONTINUE
FINISH FORECASTS WITH B_NS_INV
DO 42 I=1,NS
IF(I.EQ.1) GO TO 42
K=MIN0(I-1,P)
DO 43 J=1,K
F(I)=F(I)+PHI(J)*F(I-J)
CONTINUE
FORECASTS NEAR DONE NOW GET COV MX
B_NS_INV IN FRONT
DO 52 J=1,NS
DO 52 I=1,NS
LL=MIN0(I-1,P)
IF(I.EQ.1) GO TO 52
DO 53 K=1,LL
DO 53 K=1,LL
C0V(F(I,J)=C0V(F(I,J)+PHI(K)*C0V(F(I-K,J)
CONTINUE
B_NS_INV TRANSP IN BACK
DO 56 J=1,NS
DO 56 I=1,NS
IF(I.EQ.1) GO TO 56
LL=MIN0(I-1,P)
DO 55 K=1,LL
DO 55 K=1,LL
C0V(F(J,I)=C0V(F(J,I)+PHI(K)*C0V(F(J,I-K)
CONTINUE
CONTINUE
CONTINUE
DO 49 I=1,NS
F(I)=F(I)+ZHM
RETURN
END
SUBROUTINE BARMAD(Z,P,Q,N,PHI,THETA,EMGZ,DB,IID,QF,F,COVF,NS,GAM,TAU)

C IN Z,P,Q,PHI,THETA
N=LENGTH OF SERIES; NS=NO. OF FORECASTS
IN PRIOR ON MEAN IS NORMAL(GAM,TAU)
OUT EMGZ=MEAN OF POSTERIOR OF MU
OUT DB*2**IID = DETERMINANT FOR DENSITY OF Z
OUT QF = QUADRATIC FORM FOR DENSITY OF Z
OUT F = VECTOR OF FORECASTS
OUT COVF = COVARIANCE MATRIX OF FORECASTS
REAL Z(500),PHI(10),THETA(10),F(10),COVF(10,10)
REAL BZ(500),BO(500),E(10,10),AS(10),BS(10)
INTEGER P,Q
REAL L(100)
LOC(I,J)=MP1*MOD(I-1,MP1)+J-1+MP1
D LIES ON DIAGONAL OF L
L STORED SLIDING, AND ONLY RECENT PART
M=MAX0(P,Q)
MP1=M+1
CALL GETSET(PHI,THETA,P,Q)
GETSET PREPARES VALUES OF A,D_1,C_1,C_2 AND E
USES COMMON AREA NAMED XARMAX
P,Q < 10 < N NS <= 10
ODO=0.
ODZ=0.
ZDZ=0.
CREATE BZ AND B*ONE
DO 2I=1,M
BZ(I)=Z(I)
2. BO(I)=1.
S=1.
IF (P.EQ.0) GO TO 6
DO 4I=1,P
S=S-PHI(I)
6 DO 8I=MP1,N
BZ(I)=Z(I)
IF (P.EQ.0) GO TO 8
DO 9J=1,P
9 BZ(I)=BZ(I)-PHI(J)*Z(I-J)
BO(I)=S
BZ HOLDS B M,N * Z
BO HOLDS _B M,N * ONE
DB=1.
IID=0
DO 12I=1,N
W=GRETA(I,I)
GRETA GETS VALUES OF B M,N * A N * (B M,N)-TRANSPOSE
AKA A M, C_1, C_2,D_I, AND E-
IF (I.EQ.1) GO TO 14
IMQ=1
IF (I.LE.M) GO TO 11
IF (Q.EQ.0) GO TO 14
IMQ=I-Q
11 IM1=I-1
DO 15J=IMQ,IM1
S=GREAT(A(I,J))
IF(IMQ,GE.J) GO TO 15
JM1=J-1
DO 16K=IMQ,JM1
16 S=S-L(LOC(J,K))*L(LOC(I,K))
L(LOC(I,J))=S
DO 17J=IMQ,IM1
S=L(LOC(I,J))
L(LOC(I,J))=S/L(LOC(J,J))
W=W-S*L(LOC(I,J))
BZ(I)=BZ(I)-L(LOC(I,J))*BZ(J)
BO(I)=BO(I)-L(LOC(I,J))*BO(J)
CONTINUE
C COMPUTE BILINEAR FORMS
ODO=ODO+BO(I)*BO(I)/W
ZDZ=ZDZ+BZ(I)*BZ(I)/W
ODZ=ODZ+BO(I)*BZ(I)/W
DB=DB*W
CALL ADJUST(DB,IID)
CONTINUE
C GET MEAN OF POSTERIOR OF MU**** E(MU GIVEN Z)
EMGZ=(GAM*TAU+ODZ)/(TAU+ODO)
C GET QUADRATIC FORM FOR DENSITY
QF=ZDZ+GAM*GAM*TAU-EMGZ*(GAM*TAU+ODZ)
C ADJUST DETERMINANT FOR BAYESIAN ANALYSIS
DB=DB*(1.+ODO/TAU)
C NOW START ON FORECASTS
DO 30 I=1,NS
F(I)=0.
AS(I)=0.
BS(I)=0.
DO 30 J=1,I
C INITIALIZE COV MX FOR FORECASTS
COVF(I,J)=GREAT(N+I,N+J)
IF (Q.EQ.0) GO TO 39
C GET D-INV L-INV ( Ø E )
DO 32K=1,Q
DO 33I=K,Q
E(I,K)=GREAT(N+K,N-Q+I)
IF(I.EQ.K) GO TO 33
IM1=I-1
DO 34J=K,IM1
E(I,K)=E(I,K)-L(LOC(N-Q+I,N-Q+J))*E(J,K)
CONTINUE
DO 361=K,Q
S=E(I,K)
KM1=K-1
IF(KM1.EQ.0) GO TO 38
DO 37J=1,KM1
37 COVF(K,J)=COVF(K,J)-S*E(I,J)
38 E(I,K)=E(I,K)/L(LOC(N-Q+I,N-Q+I))
COVF(K,K)=COVF(K,K)-S*E(I,K)
AS(K)=AS(K)+E(I,K)*BO(N-Q+I)
36 BS(K)=BS(K)+E(I,K)*BZ(N-Q+I)
CONTINUE
32 CONTINUE
39 CONTINUE
C SYMMETRIZE
DO 51I=1,NS
DO 51J=1,I
51 COVF(J,I)=COVF(I,J)
IF(P.EQ.0) GO TO 48
DO 41I=1,P
DO 41J=I,P
41 AS(I)=AS(I)+PHI(J)
BS(I)=BS(I)+PHI(J)*Z(N+I-J)
C FINISH FORECASTS WITH B_NS-INV
DO 42I=1,NS
IF(I.EQ.1) GO TO 42
K=MIN0(I-1,P)
DO 43 J=1,K
43 AS(I)=AS(I)+PHI(J)*AS(I-J)
42 CONTINUE
C FORECASTS NEAR DONE NOW GET COV MX
C B_NS-INV IN FRONT
DO 52J=1,NS
DO 52I=1,NS
52 L=MIN0(I-1,P)
IF(I.EQ.1) GO TO 52
DO 53K=1,L
53 COVF(I,J)=COVF(I,J)+PHI(K)*COVF(I-K,J)
52 CONTINUE
C B_NS-INV TRANSP IN BACK
DO 56J=1,NS
DO 56I=1,NS
56 L=MIN0(I-1,P)
DO 55K=1,L
55 COVF(J,I)=COVF(J,I)+PHI(K)*COVF(J,I-K)
56 CONTINUE
48 CONTINUE
49 F(I)=EMGZ*(1.-AS(I))+BS(I)
49 W=TAU+ODO
DO 58 I=1,NS
DO 58 J=1,NS
58 COVF(I,J)=COVF(I,J)+AS(I)*AS(J)/W
CONTINUE
RETURN
END
SUBROUTINE CFARMA(PHI,THETA,P,Q,X)
COMPUTES COVARIANCE FUNCTION, SIGMA(0), ..., SIGMA(M)
FOR ARMA PROCESS USING MACLEOD'S ALGORITHM
CRSEPP IS LINEAR EQUATIONS SOLVER
REAL PHI(10), THETA(10), A(10,10), X(10), C(10)
INTEGER P,Q,R,RP1,QP1
R=MAX0(P,Q)
RP1=R+1
C(1)=1.
IF(Q.EQ.0) GO TO 4
DO 2K=1,Q
J=K+1
C(J)=-THETA(K)
IF(P.EQ.0) GO TO 2
L=MIN0(P,K)
DO 3I=1,L
3  C(J)=C(J)+PHI(I)*C(J-I)
2 CONTINUE
3 CONTINUE
X(1)=C(1)
DO 9I=2,10
X(I)=0.
IF(Q.EQ.0) GO TO 6
DO 7I=1,Q
X(I)=X(I)-THETA(I)*C(I+1)
DO 8K=1,Q
J=K+1
DO 8I=K,Q
X(J)=X(J)-THETA(I)*C(I-K+1)
IF(P.EQ.0) RETURN
DO 12I=1,RP1
DO 12J=1,RP1
A(I,J)=0.
DO 14I=1,RP1
A(I,I)=1.
DO 14J=1,P
L=ABS(I-J-1)+1
A(I,L)=A(I,L)-PHI(J)
CALL CRSEPP(A,RP1,S,X)
RETURN
END
REAL FUNCTION GRETA(I,J)
GRETA GETS QUICKLY VALUES OF B_M,N * A_N * (B_M,N)-TRANSPOSE
COMMON /XARMAX/M,IQ1,AM,CC,D1
K=I-J+1
GRETA=0.
IF(K.LE.IQ1) GRETA=CC(K)
IF(J.GT.M) RETURN
IF(I.GT.M) GO TO 4
GRETA=AM(K)
RETURN
4 GRETA=D1(K-1)
RETURN
END
SUBROUTINE ADJUST(D,I)

C ADJUST KEEPS DET FROM EXPLODING

IF(D.LE.0.) GO TO 6

IF(D.GE.1.) GO TO 4

D=D*16.

I=I-4

GO TO 3

IF(D.LE.16.) RETURN

D=D/16.

I=I+4

GO TO 4

I=-2147483644

RETURN

END

SUBROUTINE GETSET(PHI,THETA,P,Q)

REAL PHI(10),THETA(10),AM(10),CC(10),DL(10)

INTEGER P,Q

COMMON /XARMAX/M,IQ1,AM,CC,DL

IQ1=Q+1

M=MAX0(P,Q)

CALL CFARMA(PHI,THETA,P,Q,AM)

CC(1)=1.

IF(Q.EQ.0) GO TO 6

DO 2I=1,Q

CC(1)=CC(1)+THETA(I)**2

DO 4I=1,Q

IP1=I+1

CC(IP1)=CC(IP1)+THETA(J)*THETA(J-1)

CONTINUE

CONTINUE

DO 7I=1,M

D1(I)=AM(I+1)

IF(P.EQ.0) GO TO 7

DO 8K=1,P

D1(I)=D1(I)-PHI(K)*AM(IABS(I-K)+1)

CONTINUE

RETURN

END
SUBROUTINE CRSEPP(A,N,D,V)
REAL A(10,10),V(10)
INTEGER R
D=1.
DO 2K=1,N
DO 3I=K,N
S=A(I,K)
IF(K.EQ.1) GO TO 3
KM1=K-1
DO 4L=1,KM1
4 S=S-A(I,L)*A(L,K)
A(I,K)=S
S=0.
R=0
DO 5I=K,N
IF(ABS(A(I,K)) LE S) GO TO 5
R=I
S=ABS(A(I,K))
CONTINUE
IF(S.EQ.0.) STOP
D=D*A(R,K)
IF(R.EQ.K) GO TO 1
D=-D
DO 6J=1,N
T=A(R,J)
A(R,J)=A(K,J)
A(K,J)=T
V(R)=V(K)
V(K)=T
IF(K.EQ.N) GO TO 2
KP1=K+1
DO 8J=KP1,N
S=A(K,J)
IF(K.EQ.1) GO TO 7
DO 9L=1,KM1
S=S-A(K,L)*A(L,J)
A(K,J)=S/A(K,K)
CONTINUE
V(1)=V(1)/A(1,1)
DO 9K=2,N
KM1=K-1
DO 10I=1,KM1
V(K)=V(K)-A(K,I)*V(I)
V(K)=V(K)/A(K,K)
DO 11J=2,N
K=N+1-J
KP1=K+1
DO 12I=KP1,N
V(K)=V(K)-A(K,I)*V(I)
CONTINUE
RETURN
END