Maximum Likelihood Estimates of Non-Gaussian ARMA Models

Mark E. Lehr and Keh-Shin Lii
University of California Riverside, USA

We consider an approximate maximum likelihood algorithm for estimating parameters of possibly non-causal and non-invertible autoregressive moving average processes driven by independent identically distributed non-Gaussian noise. The normalized approximate maximum likelihood estimate has a global maximum which is consistent and efficient. The estimates and their associated asymptotic covariance matrix are calculated with a subroutine implemented in FORTRAN 77.
Keywords: Autoregressive moving average, maximum likelihood estimate, non-causal, non-invertible, non-Gaussian, stationary, white noise, genetic algorithm, simulated annealing.

1 INTRODUCTION

Time series data occur in a variety of disciplines including engineering, science, sociology and economics among others. There are many techniques that have been derived to infer the characteristics of such series. This is done by first hypothesizing a mathematical model to represent the data. Having chosen a model it then becomes possible to estimate parameters which will enhance our understanding of the mechanisms which generate this sequence. Once a satisfactory model has been developed then it maybe used to help separate the noise from the signal and even predict future values.

There is extensive literature on finite parameter time series models such as Brockwell and Davis [1991], Greene [1993], and Hamilton [1994] to name a few. Among these the autoregressive moving average (ARMA) models are extensively studied with wide applications. Existing literature on these models are mainly based upon

This research has been supported by ONR N00014-92-J-1086 and ONR N00014-93-1-0892.
Address for correspondence: Keh-Shin Lii, University of California Riverside, Statistics Department, Riverside, California 92521-0138. Email: ksl@gauss.ucr.edu

the Gaussian, causal and invertible assumptions. In recent years there has been considerable interest in the non-Gaussian, non-causal, and non-invertible ARMA models[Lii and Rosenblatt 1982] and [Rosenblatt 1985]. Applications of these models have been well documented [Nikias and Petropulu 1993]. Inferences on these models are based on higher order moment or cumulant information. This paper presents a maximum likelihood (MLE) based algorithm which gives efficient consistent estimates of the parameters of a non-Gaussian, possibly non-causal and non-invertible stationary ARMA process. Examples are given to illustrate the usefulness of the algorithm.

2 THEORY

We begin by considering ARMA sequences driven by independent identically distributed non-Gaussian noise [Lii and Rosenblatt 1996]. The MLE so developed is a function of the coefficients of the ARMA model and the probability density function of the driving noise process. The number of local maxima encountered are directly related to the roots of the ARMA polynomials. The surface of the MLE is non-linear in nature resulting in the need to implement methodical sequential search techniques if the parameter space is small or stochastic techniques if the parameter space is large.

With this in mind, we define a zero mean process $\{X_t, t = \ldots, -1, 0, 1, \ldots\}$ which is said to be an ARMA(p,q) process if $\{X_t\}$ is stationary and satisfies

$$X_t + \phi_1 X_{t-1} + \ldots + \phi_p X_{t-p} = Z_t + \theta_1 Z_{t-1} + \ldots + \theta_q Z_{t-q} \tag{1}$$

where Z_t are independent and identically distributed random variables with mean zero and variance σ^2. The parameters ϕ and θ are polynomial coefficients represented by

$$\phi_p(B) X_t = \theta_q(B) Z_t \tag{2}$$

with

$$\phi_p(B) = 1 + \phi_1 B + \phi_2 B^2 + \ldots + \phi_p B^p \tag{3}$$

and

$$\theta_q(B) = 1 + \theta_1 B + \theta_2 B^2 + \ldots + \theta_q B^q. \tag{4}$$

B is the backshift operator having the property

$$B^k Z_t = Z_{t-k}. \tag{5}$$

We assume that $\phi_p(z)$ and $\theta_q(z) \neq 0$ for all $|z| = 1$. Causality and invertibility are defined with respect to the roots of the polynomials. If all the roots of

$$\phi_p(B) = \prod_{i=1}^{p}(1 - \zeta_i B) \tag{6}$$

are greater than 1 then the autoregressive polynomial is said to be causal. If all the roots of

$$\theta_q(B) = \prod_{i=1}^{q}(1 - \zeta_i B) \tag{7}$$

are greater than 1 then the moving average polynomial is said to be invertible.

The input process is assumed to have a probability density function $f(z)$ which must be normalized by the standard deviation σ as the scale factor (i.e. $f_\sigma(z) = \sigma^{-1}f(z\sigma^{-1})$). This noise process can be represented by a Laurent series expansion of $\theta^{-1}(z)\phi(z)$,

$$\begin{aligned} Z_t &= \theta(B)^{-1}\phi(B)X_t && (8)\\ &= \sum_{j=-\infty}^{\infty} \pi_j X_{t-j}. && (9) \end{aligned}$$

To facilitate the computation of π_j, the maximum likelihood function, and the covariance matrix it is convenient to factor the polynomials associated with the ARMA process into

$$\begin{aligned} \phi(B) &\equiv \phi^+(B)\phi^-(B) && (10)\\ &\equiv (1 + \phi_1^+ B + \ldots + \phi_r^+ B^r)(1 + \phi_{r+1}^- B + \ldots + \phi_p^- B^s), && (11)\\ \theta(B) &\equiv \theta^+(B)\theta^-(B) && (12)\\ &\equiv (1 + \theta_1^+ B + \ldots + \theta_{r'}^+ B^{r'})(1 + \theta_{r'+1}^- B + \ldots + \theta_q^- B^{s'}), && (13) \end{aligned}$$

where ϕ^+ and θ^+ have no roots on the closed unit disc and ϕ^- and θ^- have all roots in the interior of the unit disc. The inverses are

$$\begin{aligned} \phi^{-1}(B) &= (\phi^+(B))^{-1}(\phi^-(B))^{-1} && (14)\\ &\equiv (\alpha(B))(\beta(B)) && (15)\\ &\equiv (\sum_{j=0}^{\infty}\alpha_j B^j)(\sum_{j=s}^{\infty}\beta_j B^{-j}), && (16)\\ \theta^{-1}(B) &= (\theta^+(B))^{-1}(\theta^-(B))^{-1} && (17)\\ &\equiv (\alpha'(B))(\beta'(B)) && (18)\\ &\equiv (\sum_{j=0}^{\infty}\alpha'_j B^j)(\sum_{j=s'}^{\infty}\beta'_j B^{-j}). && (19) \end{aligned}$$

Then the driving noise process Z_t is

$$\begin{aligned} Z_t &= (\alpha'(B)(\phi(B)))\beta'(B)X_t & (20)\\ &\equiv \alpha''(B)\beta'(B)X_t & (21)\\ &\equiv (\sum_{j=0}^{\infty} \alpha_j'' B^j)(\sum_{j=s'}^{\infty} \beta_j' B^{-j})X_t. & (22) \end{aligned}$$

For numerical purposes, the computation of the noise process is approximated by

$$\hat{Z}_t = \sum_{j=0}^{p+L_{\alpha'}} \sum_{k=s'}^{s'+L_{\beta'}} \alpha_j'' \beta_k' X_{t-j+k} \tag{23}$$

where $L_{\alpha'}$, $L_{\beta'}$ represent the truncated length of the α' and β' polynomials. The procedure for estimating an ARMA process without the Gaussian, causal, and invertible assumptions has been developed by Lii and Rosenblatt [1996]. The MLE is a function of the input sequence and the parameter space which has the following form:

$$\overline{L}(\underline{\eta}) = \sum_{t=k+1}^{n-k} \frac{log(f_\sigma(\hat{Z}_t))}{n-2k} + log(|\phi_p^-|) - log(|\theta_q^-|) \tag{24}$$

where k represents a truncation at the boundaries of the data by a factor of $O(n^{0.5})$ implemented by $max(10, \sqrt{n})$ and the vector $\underline{\eta}$ represents the $p+q+1$ unknown parameters

$$\underline{\eta} = (\phi_1^+, \ldots, \phi_r^+, \phi_{r+1}^-, \ldots, \phi_p^-, \theta_1^+, \ldots, \theta_{r'}^+, \theta_{r'+1}^-, \ldots, \theta_q^-, \sigma)' \tag{25}$$

which are to be estimated. Parameterization of the estimates as defined in equation (1) where $\underline{\eta}_1 = (\phi_1, \ldots, \phi_p, \theta_1, \ldots, \theta_q, \sigma)'$ uses equations (10) and (12). The covariance matrix $\sum^{-1}$ is obtained from equation (1.5) and Table (1) of Lii and Rosenblatt [1996]. The Jacobian $R = [\partial\underline{\eta}_1/\partial\underline{\eta}]$ is required to calculate the covariance matrix $(R\sum^{-1} R')$ for the estimates of $\underline{\eta}_1$. We now give a few examples to illustrate the algorithm.

3 EXAMPLES

3.1 ARMA's with Recipricol Roots

Consider the case of an ARMA(1,1) where the roots of the AR and MA process are reciprocals.

$$(1 - \zeta B)X_t = (1 - \zeta^{-1} B)Z_t \tag{26}$$

This model is unidentifiable using second order statistics such as the auto-covariance function (ACF) and the partial auto-covariance function (PACF). None of the non-zero lagged results are significantly different from zero. The power spectrum is thus a constant across all frequencies which is the characteristic of white noise.

However, the procedure outlined above is quite capable of detecting this condition should the underlying process have a non-Gaussian noise distribution. Simulating a Student t with 4 degrees of freedom as the density of Z_t and a sample size of 800 with $\zeta = 0.5$ in (26) produces the log-likelihood surface depicted in Figure 1. (Refer to equations 24 and 26 for the definition of the surface and the ARMA model.) With 2 roots there are 4 local maximum since the root and its reciprocal produce areas maximizing the MLE. One of the root combinations gives the Global maximum. The surface features have been truncated at the bottom in this Figure to accentuate the areas where the maxima occur. Table 1 reveals that the parameters calculated using Splus are essentially zero. This corresponds to a white noise process as anticipated. The reason is that Splus assumes causality, invertibility, and a Gaussian distribution for the noise process.

Table 1: ARMA(1,1) Parameter Estimates given Distribution Assumptions. *Note: Results given as mean/std.*

Simulated Parameters	**Student t(4) Estimation**	**Gaussian Estimation**
ϕ_1 = -0.5	-0.518/0.037	-0.046/0.076
θ_1 = -2.0	-2.017/0.152	-0.066/0.074
σ = 1.0	0.951/0.079	1.83

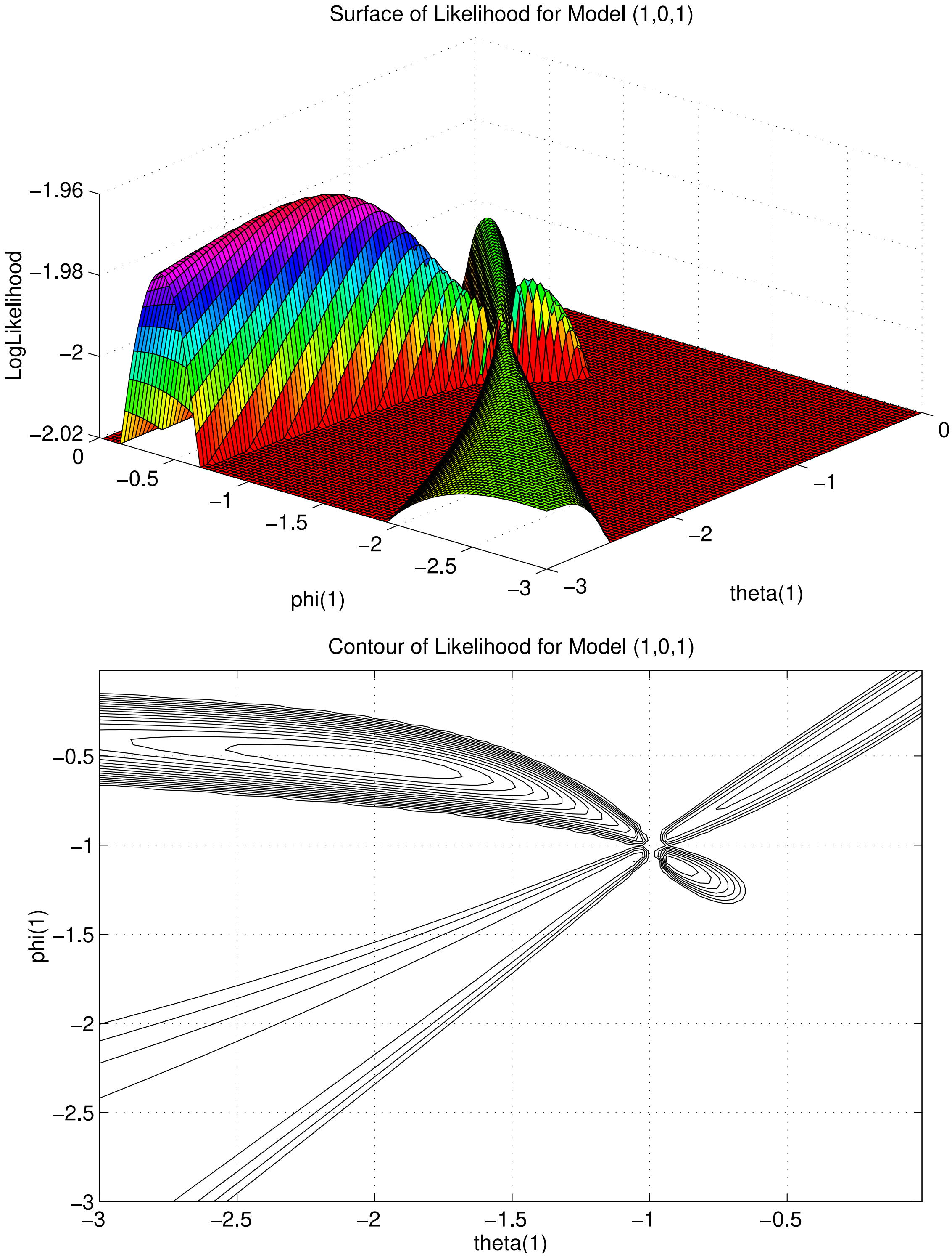

Figure 1: The maximum likelihood surface for the simulated data

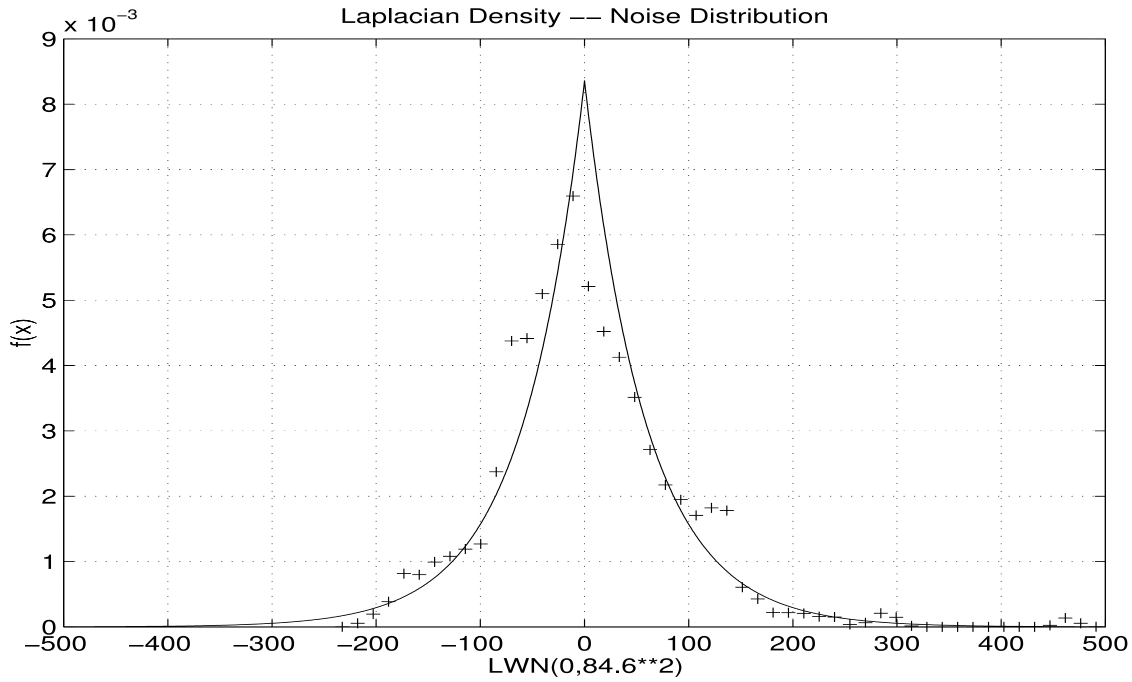

Figure 2: Residuals compared to a Laplacian Distribution. (The + represent the histogram of the residuals while the solid line is a fitted Laplacian density function)

3.2 Economic Data

This example uses unemployment data for the United States from 1948 to 1977. The data consist of monthly statistics over the course of 30 years resulting in 360 samples which have not been adjusted for seasonality [Herzberg 1985]. The initial identification process using the ACF and PACF points to the possibility of unit roots at lag 1 and at lag 12. The Dickey-Fuller procedure [Hamilton 1994] and the Durbin-Watson procedure [Greene 1993] confirm that single unit roots at lag 1 and 12 are significant possibilities based upon the hypothesized model of a random walk with drift. After differencing at the lags indicated, the result is a zero mean stationary time series. The Gaussian, causal, and invertible assumption using Splus leads to the following ARIMA model $(2,1,0)_1(0,1,1)_{12}$ represented by

$$(1-B)(1-B^{12})(1-\phi_2 B^2)X_t = (1-\theta_{12}B^{12})Z_t. \tag{27}$$

The above model was determined based upon the Akaike's AIC criterion. At this point, we examine the residuals and determine if they exhibit a Gaussian structure (Figure 2). It appears that they do not and a first approximation of their density might be the Laplacian distribution. Using this assumption and the non-Gaussian,

Table 2: Parameter Estimates given Distribution Assumptions for ARIMA model $(2,1,0)_1(0,1,1)_{12}$. *Note: Results given as mean/std.*

Parameters	Laplacian Estimation	Gaussian Estimation
$\phi_2 =$	-0.041/0.0401	-0.121/0.0535
$\theta_{12} =$	-0.706/0.0285	-0.762/0.0358
$\sigma =$	89.0/5.049	83.7

non-causal, and non-invertible MLE procedure we obtain the parameter estimates in Table 2. The AR coefficient is not distingushiable from zero and the MA coefficient is slightly smaller and still significant. The ARIMA model $(0,1,0)_1(0,1,1)_{12}$ represented by

$$(1-B)(1-B^{12})X_t = (1-\theta_{12}B^{12})Z_t. \tag{28}$$

is the result of using distributional assumptions consistent with the residuals. The model has been reduced to its most parsimonious form exhibiting yearly correlations which would be expected of employment data. No attempt was made to examine other possible models. The procedure outlined above involves finding the best standard model using existing methods, examining the residuals, choosing an appropriate distribution, and then reexamine the results to determine an improvement in the model to gain additional insights into the process.

3.3 Other Distributional Considerations

The exponential power distribution is a sufficiently rich family to explore the general behavior of symmetrical, unimodal distributions. The Normal, Laplacian, and even Uniform like distributions can be generated from this family, so it might offer a reasonably rich parametric distribution to examine the effects of different assumptions. This of course does not limit the procedure since any parametric or non-parametric assumption can be investigated by the algorithm.

The zero mean exponential power distribution has the following form [Box and Tiao 1973]:

$$f(y) = \frac{e^{-0.5|y|^{\frac{2}{1+\beta}}}}{\Gamma(1+\frac{1+\beta}{2})2^{1+\frac{1+\beta}{2}}} \tag{29}$$

Table 3: ARMA(2,3) Parameter Estimates given Distribution Assumptions. *Note: Results given as mean/std.*

Simulated Parameters	**Exp-Power Estimation**	**Gaussian Estimation**
$\boldsymbol{\phi_1} = 1.333...$	1.320/0.023	0.646/0.023
$\boldsymbol{\phi_2} = 0.444...$	0.440/0.022	0.002/0.018
$\theta_1 = 0.5$	0.524/0.036	-1.011/0.028
$\theta_2 =$ -1.25	-1.246/0.031	0.265/0.027
$\theta_3 = 0.375$	0.363/0.035	0.007/0.024
$\sigma = 1$	0.998/0.019	1.53

where $\beta = (-1, 1]$. An ARMA(2,3) using this distribution will be generated having multiple roots and one in particular that will be reciprocal such as

$$(1 + \zeta_1 B)^2 X_t = (1 + \zeta_1^{-1} B)(1 - \zeta_2 B)^2 Z_t. \tag{30}$$

In this case, second order statistics produces a model which is an ARMA(1,2) having the following form

$$(1 + \zeta_1 B) X_t = (1 - \zeta_2 B)^2 Z_t. \tag{31}$$

As before, the reciprocal root reduces the number of parameters under the normal assumption. The reason for this can be found in the roots of the polynomial and the ability to reparameterize using the auto-covariance generating function [Brockwell and Davis 1991]. To illustrate this, a simulated sequence with a sample size of 2000 and $\beta = 0.75$, $\zeta_1 = 2/3$, and $\zeta_2 = 1/2$ produces the specific estimates shown by Table 3. This confirms the anticipated results exhibited by equation 30 and 31 which matches the simulated parameter values under the two different assumptions. Table 4 provides the covariance matrix which indicates the interrelationships of the parameters under the power distribution assumption.

Table 4: ARMA(2,3) Covariance Matrix using the Non-Gaussian Parameter Estimation Process

	$\boldsymbol{\phi_1}$	$\boldsymbol{\phi_2}$	θ_1	θ_2	θ_3	σ
$\boldsymbol{\phi_1}$	0.000517	0.000476	0.000312	0.000373	-0.000622	0.000079
$\boldsymbol{\phi_2}$	0.000476	0.000476	0.000331	0.000321	-0.000574	0.000050
θ_1	0.000312	0.000331	0.001281	-0.000335	-0.000580	-0.000469
θ_2	0.000373	0.000321	-0.000335	0.000987	-0.000529	0.000459
θ_3	-0.000622	-0.000574	-0.000580	-0.000529	0.001191	-0.000110
σ	0.000079	0.000050	-0.000469	0.000459	-0.000110	0.000376

4 PROGRAM STRUCTURE AND I/O

4.1 Using NGMLE

The algorithm described previously has been implemented in FORTRAN 77. Though a single call to a subroutine is enough to obtain the estimates required, the program is composed of building blocks which are broken into the following libraries. The user will be unaware of this except at the time of compilation where the libraries must be located in the same directory or path. The following is a list of libraries utilized by the non-Gaussian MLE subroutine (ngmle).

POPFNC.for	Develops the MLE [Lii and Rosenblatt 1996] and stands for the *pop* ulation *func* tion associated with a popular stochastic maximization technique. Includes polynomial multiplication and inversion subroutines.
COVAR.for	Calculates the covariance matrix of the unknown parameters and includes matrix utilities.
SEARCH.for	Performs the search of the MLE surface. Decides which search method to use and either implements the uniform grid search or the Simulated Annealing/Genetic Algorithm. [Holland, 1992],[Vanhaarhoven and Aarts 1987]
PUBLIC.for	Contains the public domain software which includes a random number generator, polynomial root finder and matrix inverter.

The driver program must call the non-Gaussian MLE subroutine as described below. The user must compile the libraries and supply the scaled density function. Here is a rough outline of how to structure the driver program.

```
c       Dimension all variables including the work
c       space variables.
        . . .
c       Read in data
        . . .
c       Assign input values for the subroutine call
        . . .
        call ngmle(nx,x,p,q,etalim,c1,c2,ndiv,nrep,supsag,rspc,ispc
                  z,nz,xmle,eta,covvec,stdev,nnzc)
        . . .
c       Output Results
c               Be sure to resolve the covariance vector
c               into the covariance matrix properly.
c               Refer to the example test drivers.
        . . .
```

The input-output table follows which defines the parameters for the subroutine. In addition, sample calculations are provided for normalizing a distribution and defining the constants required by the program.

INPUTS

TYPE	VARIABLE	COMMENTS
INTEGER	NX	The actual number in the sequence X.
REAL(array)	X	The ARMA sequence to be analyzed.
INTEGER	P	An integer in the ARMA(p,q) sequence. The AR degree.
INTEGER	Q	An integer in the ARMA(p,q) sequence. The MA degree.
REAL (2 dim array)	ETALIM	The lower and upper bounds within which to search for the coefficient values, ETALIM(1,j)=lower bound of the jth coefficient. ETALIM(2,j)=upper bound of the jth coefficient.
REAL	C1	A constant used in the covariance calculations.
REAL	C2	A constant used in the covariance calculations.
INTEGER	NDIV	The number of subdivisions per parameter to sequentially search the MLE surface.
INTEGER	NREP	The number of repetitions to search the surface as the search volume is collapsed about the global maximum. Each collapse improves the estimates accuracy by 1/NDIV.
INTEGER	SUPSAG	Suppression of the genetic algorithm and simulated annealing. SUPSAG = 1 suppresses the stochastic search in favor of the sequential.
REAL(array)	RSPC	A real work space vector. Its dimension is (11*101+2*NX+124*NP+10*NP*NP) NP=P+Q+1 where P,Q, and NX have been previously defined.
COMPLEX(array)	CSPC	A complex work space vector. Its dimension is (4*NP)
INTEGER(array)	ISPC	A integer work space vector. Its dimension is (200+4*NP) where NP has been previously defined.

OUTPUTS

REAL(array)	Z	The estimated driving sequence
INTEGER	NZ	The number of components in the estimated driving sequence vector.
REAL	XMLE	The non-Gaussian MLE. Refer to equation (24).
INTEGER(array)	ETA	The estimated parameter values. Refer to equation (1).
REAL(array)	COVVEC	The estimated covariance vector for the parameter estimates.
REAL(array)	STDEV	The standard deviation of the estimated parameters.
INTEGER	NNZC	The number of non-zero parameters. Refer to Driver program for an ARMA(2,12) example.

4.2 User Supplied Functions and Constants

The user must supply the probability density of the driving noise process $f_\sigma(z) = \sigma^{-1} f(z\sigma^{-1})$ in order to calculate the ARMA model parameters from equation (1). The two constants c_1 and c_2 are utilized for the calculation of the covariance matrix [equations 1.4 from Lii and Rosenblatt 1996]. The constants have the following form

$$c_1 = \sigma^2 E(\frac{f'_\sigma}{f_\sigma}(z))^2, \tag{32}$$

$$c_2 = E(z\frac{f'_\sigma}{f_\sigma}(z))^2. \tag{33}$$

Note: The two constants are fixed for the given probability distribution. These are calculated prior to the analysis if the density is of a suitable parametric form but may require a two step process if numerical integration is required to obtain the constants once the scale factor has been found.

To facilitate the discussion of scaling the density and calculating the constants,the ARMA(2,12) example depicted in the driver program will be fully evaluated. This should be sufficient to allow the user the ability to prescribe any parametric or non-parametric density function. If the input density function is non-parametric then (c_1 and c_2) will have to be calculated numerically.

4.2.1 Example to scale the Laplace Density

The Laplacian probability density is normally written as

$$f(z) = \frac{1}{2\lambda} e^{\frac{-|z|}{\lambda}}. \tag{34}$$

To scale this function we need to modify z by the appropriate quantity. The standard deviation of this function is $\sigma = \sqrt{2}\lambda$. Therefore, the proper scaling function is

$$f_\sigma(z) = \frac{\sqrt{2}\lambda}{\sigma} f(\frac{\sqrt{2}\lambda z}{\sigma}) \tag{35}$$

$$= \frac{1}{\sqrt{2}\sigma} e^{\frac{-\sqrt{2}|z|}{\sigma}}. \tag{36}$$

The density supplied by the user becomes

$$f(z) = \frac{1}{\sqrt{2}} e^{-\sqrt{2}|z|} \tag{37}$$

which in Fortran code would be the following function.

```
function dnsity(z)
        dnsity = exp(-sqrt(2.0)abs(z))/sqrt(2.0)
return
end
```

The actual calling sequence in the program would be fsigma=dnsity(z/sigma)/sigma. The probability density function must be specified properly or the results will not be as intended. The program will search for the best σ to maximize the log likelihood.

4.2.2 Example Calculation and implementation of c_1 for the Laplacian Distribution

$$c_1 = \sigma^2 E(\frac{f_\sigma^{'}}{f_\sigma}(z))^2 \tag{38}$$

By using a simple technique of taking the derivative of a log, we can manipulate the above equation for the Laplace distribution by

$$\frac{f_\sigma^{'}}{f_\sigma}(z) = \frac{d}{dz} log(f_\sigma(z)) \tag{39}$$

which gives

$$(\frac{d}{dz} log(f_\sigma(z)))^2 = \frac{2}{\sigma^2}. \tag{40}$$

Finding the expectation of this gives the following constant

$$\begin{aligned} c_1 &= \sigma^2 E(\frac{2}{\sigma^2}) && (41) \\ &= 2. && (42) \end{aligned}$$

Table 5: Constants utilized in the asymptotic covariance calculations for familiar distributions

Name	$\mathbf{c_1}$	$\mathbf{c_2}$
Laplace	2	2
Gaussian	1	3
Student t(4)	1.43	2.14
Exp Power ($\beta = -0.75$)	2.55	9

4.2.3 Example Calculation and implementation of c_2 for the Laplacian Distribution

$$c_2 = E(z\frac{f_\sigma^{'}}{f_\sigma}(z))^2 \tag{43}$$

Similarly for this

$$\begin{aligned} c_2 &= E(z^2\frac{2}{\sigma^2}) && (44) \\ &= 2 && (45) \end{aligned}$$

since the scaled expectation squared is simply the variance of this distribution. Table 5 gives various common distribution values for the constants defined above.

5 PROGRAMMATIC CONSIDERATIONS

5.1 Restrictions

The root finding technique (refer to equations 10-13) requires that the polynomial under investigation be the order specified. If the range to search for the last coefficient (i.e. ϕ_p or θ_q) passes exactly through zero thus reducing the order of the polynomial, then an error will occur. This can be eliminated by making sure that the range and the corresponding search grid never pass through this point. It is also important to make sure that the density function is not exactly zero (refer to equation 24). This can be mitigated by providing a lower tolerance limit in the user supplied density function.

No limits have been placed on the number of parameters $\underline{\eta}$ or sequence size x_t. However, the procedure has only been utilized with 19 parameters and a sequence size of 5000 for testing ARMA processes. The Simulated Annealing and Genetic Algorithms have been tested indivdually for up to 80 parameters and the polynomial root finder has been tested up to a 100th degree polynomial. This has been adequate for all the problems we have encountered. If the need arises to go beyond this limit then the increase in sequence size and the parameter space will increase the computer time proportionately.

5.2 Precision

Single Precision is used throughout. The numerical procedures for inverting the MA polynomial $\theta(B)$ is a source of possible error for roots close to the unit circle. Currently a maximum of 100 coefficients for $L_{\alpha'}$, $L_{\beta'}$ from equation 23 are used in this procedure which automatically varies according to the position of the roots . The closer the roots are to the unit radius, the more terms are required in the inverted polynomial truncation. A tolerance of 10^{-6} is utilized for a run of 5 coefficients as a secondary truncation criteria. It is assumed that the sequence has been checked for unit roots and they are eliminated prior to analysis by this algorithm.

Computational results to 5 significant places have been obtained on the Sun sparc 2, 5, and 20 Workstations using different operating systems. The Dec Alpha A500MP-R also matched these results with the stated accuracy. These have been compared to Monte-Carlo simulations for low-order ARMA processes which agree with the numerical results of this algorithm [Breidt et al., 1991 and Lii and Rosenblatt 1992,1996].

5.3 Timing

Several techniques are utilized in the searching process due to the multiple local maxima in the Likelihood surface. If the number of parameters is small, the program will use a uniformly distributed grid search procedure to find the Global Maximum. If the number of parameters is large, then stochastic procedures (i.e. simulated annealing and genetic algorithms) will be simultaneously executed and compared to locate the Global Maximum.

The program will automatically choose the technique which will search the parameter domain in the shortest execution time unless the stochastic methods are intentionally suppressed. The Sun Sparc stations and the DEC Alpha were used to determine the optimum mix of deterministic and stochastic search procedures to speed the examination of the Likelihood surface. If the total MLE computations **NMLE** exceeds 120,000 then the procedure will opt for a stochastic search technique. This is an ad-hoc number derived from experience for a number of different problems. The equation which determines the number of computations called for is a product of the number of subdivisions **NDIV** of each non-zero parameter

$$NMLE = (NDIV)^{NNZC}. \tag{46}$$

The number of subdivisions **NDIV** is controlled by the user whereas the model being investigated sets the number of non-zero coefficients to **NNZC**. To force the stochastic search the user can artificially make the number of subdivisions large enough to exceed $(120,000)^{\frac{1}{NNZC}}$. A flag has also been included to suppress the stochastic search methods if only a deterministic search is desired.

In general, by increasing the sequence length, the execution time will be proportionately increased. However, increasing the number of parameters will exponentially increase the execution time. A tradeoff has been accomplished by using a stochastic search procedure which permits an efficient utilization of cpu time thereby making the increase in parameters a proportional increase in execution time.

5.4 Additional Comments

Some of the numerical support routines used in implementing this procedure were obtained from the internet. The three main functions found in the public domain include a pseudo-random number generator, polynomial root finder, and a matrix inverter. They are composed of subroutines and functions which are called by the program and found in PUBLIC.for identified by the following subroutine names:

SUNIF A psuedo-uniform random number generator used in the stochastic search procedures. i.e. genetic algorithm/simulated annealing
==================================
NIST Guide to Available Math Software.
Fullsource for module 599 from package TOMS.
Retrieved from NETLIB on Wed Jun 4 20:21:26 1997.
==================================
ALGORITHM 599, COLLECTED ALGORITHMS
FROM ACM ALGORITHM APPEARED IN
ACM-TRANS. MATH. SOFTWARE,
VOL.9, NO. 2, JUN., 1983, P. 255-257.

RPQR79 Polynomial root finder utilized to seperate roots inside and outside the unit circle.

====================================

BEGIN: PROLOGUE RPQR79

DATE WRITTEN: 800601 (YYMMDD)

REVISION DATE: 820801 (YYMMDD)

CATEGORY NO. F1A1A

KEYWORDS: POLYNOMIAL ROOTS,REAL,ROOTS, ZEROES,ZEROS

====================================

AUTHOR: VANDEVENDER, W. H., (SNLA)

PURPOSE: To find the zeros of a polynomial with real coefficients.

DESCRIPTION: This routine is an interface to an eigenvalue routine in EISPACK.

This interface was written by Walter H. Vandevender.

ABSTRACT: This routine computes all roots of a polynomial with real coefficients by computing the eigenvalues of the companion matrix.

SPOSV Matrix inversion for the Covariance terms.

====================================

NIST GUIDE TO AVAILABLE MATH SOFTWARE. FULLSOURCE FOR MODULE SPOSV FROM PACKAGE LAPACK. RETRIEVED FROM NETLIB ON TUE JUN 17 13:13:58 1997.

====================================

LAPACK DRIVER ROUTINE (VERSION 2.0)

UNIV. OF TENNESSEE, UNIV. OF CALIFORNIA BERKELEY, NAG LTD.,

COURANT INSTITUTE, ARGONNE NATIONAL LAB, AND RICE UNIVERSITY

MARCH 31, 1993

PURPOSE

SPOSV COMPUTES THE SOLUTION TO A REAL SYSTEM OF LINEAR EQUATIONS

$A * X = B$,

WHERE A IS AN N-BY-N SYMMETRIC POSITIVE DEFINITE MATRIX AND X AND B ARE N-BY-NRHS MATRICES.

There is a typographical error in the paper defining the covariance matrix found in Table 1 of Lii and Rosenblatt, 1996. The 8th term in the table should be

$$\sigma_{u,v} = \sum_j \alpha'_{j-u} \beta'_{j+v-p-r'}. \tag{47}$$

In addition, our definition of all the ARMA polynomials (ϕ and θ) utilizes a positive sign convention which changes the sign of the 14th and 15th terms in the same Table (Table 1 of Lii and Rosenblatt, 1996). With these three modifications a direct comparison of the code and the equations in the table may be made since the equations and the code have the same lexicographical characteristics. A modification has also been made in the parameterization of the model results. The MLE parameterization utilized ϕ^-, ϕ^+, θ^- and θ^+. To obtain the ϕ and θ parameterization required the development of the Jacobian to produce the final results. This has been implemented by the covariance library functions. Only an interested user comparing the code with the referenced papers need be aware of these differences.

REFERENCES

Box, G. E. P. and Tiao, G. C. (1973) *Bayesian Inference in Statistical Analysis*, Addison-Wesley Publishing Company, 156-160.

Breidt F. J. et al. (1991) Maximum Likelihood Estimation for Noncausal Autoregressive Processes, *Journal of Multivariate Analysis*, **36**, 175-198.

Brockwell, P. J. and Davis, R. A. (1991) *Time Series Theory and Methods*, 2nd ed., Springer-Verlag.

Greene, W. H. (1993) *Econometric Analysis*, Macmillan Publishing Company.

Hamilton, J. D. (1994) *Time Series Analysis*, Princeton University Press.

Herzberg, A.M. (1985) *DATA: A Collection of Problems from Many fields for the Student and Research Worker*, Springer-Verlag, 391-395.

Holland, J. H. (1992) Genetic Algorithms, *Scientific American*, July, 66-72.

Lii, K. S. and Rosenblatt, M. (1982) Deconvolution and estimation of transfer function phase and coefficients for nonGaussian linear processes *Ann. Statist.*, **10**, 1195-1208.

Lii, K. S. and Rosenblatt, M. (1992) An Approximate Maximum Likelihood Estimation for NonGaussian Non-Minimum Phase Moving Average Processes, *Journal of Multivariate Analysis*, **43**, 272-299.

Lii, K. S. and Rosenblatt, M. (1996) Maximum Likelihood Estimation for NonGaussian Nonminimum Phase ARMA Sequences, *Statistica Sinica*, **6(1)**, 1-22, January.

Nikias, C. L. and Petropulu A. P. (1993) *Higher Order Spectral Analysis*, Prentice Hall.

Rosenblatt, M. (1985) *Stationary Sequences and Random Fields*, Birkhauser.

Vanhaarhoven, P.J.M. and Aarts, E.H.L. (1987) *Simulated Annealing Theory and Applications*, Kluwer Academic Publishers.