15 Finding Models and Estimating Their Parameters

We have said several times that finding a model that imitates the properties of a data set makes it easy to simulate data like that we have observed, easy to predict the future of the data, as well as to get good estimates of the spectral density of the process generating the data.

In this section we investigate methods for finding a good model, but first we discuss estimating the parameters of an $ARMA$ model if we know the orders p and q. The methods we discuss will provide insight into how to determine the orders.

15.1 General Estimation Methods

If we have a realization $X(1), \ldots, X(n)$ from an $ARMA(p, q)$ model

$$X(t) + \alpha_1 X(t-1) + \ldots + \alpha_p X(t-p) = \epsilon(t) + \beta_1 \epsilon(t-1) + \ldots + \beta_q \epsilon(t-q),$$

where $\epsilon \sim WN(\sigma^2)$, the methods for estimating the α's, the β's, and σ^2 fall into three general classes.

 Copyright ©1999 by H.J. Newton

15.1.1 Method of Moments Estimators

We have seen many examples of relationships between one set of quantities for a time series model (such as autocovariances R) and another set (such as the α's and σ^2 for an $AR(p)$ model). Method of moments estimators are obtained by substituting a set of estimates (such as $\hat{R}$'s) into these relationships and then solving the relationship to get estimators of the other set of quantities. Thus the Yule Walker estimates of α and σ^2 for an $AR(p)$ model are obtained by solving

$$\text{Toepl}(\hat{R}(0), \hat{R}(1), \ldots, \hat{R}(p-1))\hat{\alpha} = -\hat{r},$$

and

$$\hat{\sigma}^2 = \hat{R}(0) + \hat{\alpha}_1\hat{R}(1) + \cdots, \hat{\alpha}_p\hat{R}(p),$$

where $\hat{r} = (\hat{R}(1) \ldots, \hat{R}(p))^T$. (The matrix system comes from the Yule Walker equations for $v = 1, \ldots, p$, while the formula for $\hat{\sigma}^2$ is the equation for $v = 0$.)

For an $MA(q)$ and $ARMA(p, q)$ process, one takes the formulas for the R's in terms of the parameters and solves for the parameters using $\hat{R}$'s instead of R's. Unlike the AR case, solutions need not exist. For example, for an $MA(1)$ with β close to one, it is easy for $\hat{\rho}(1)$ to exceed 0.5 and thus no value of $\hat{\beta}$ will satisfy the equation $\hat{\rho}(1) = \hat{\beta}/(1 + \hat{\beta}^2)$.

(See the CORRAR, CORRMA, and CORRARMA commands.)

 Copyright ©1999 by H.J. Newton

15.1.2 Regresssion Type Estimators

The form of an $ARMA$ model suggests regressing $X(t)$ on the previous p X's as well as on $\epsilon(t-1), \ldots, \epsilon(t-q)$. For an $AR(p)$ model, this idea actually works quite well as we can regress the vector $y = (X(p+1), \ldots, X(n))^T$ on the matrix

$$-\begin{bmatrix} X(p) & X(p-1) & \cdots & X(1) \\ X(p+1) & X(p) & \cdots & X(2) \\ \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ X(n-1) & X(n-2) & \cdots & X(n-p) \end{bmatrix}$$

which will give estimates of α.

If we have an MA or $ARMA$ model, we can't do the regression directly as we don't observe the ϵ's. However, if the model is invertible, then we can pretend that our data come from a "high order" AR model, estimate the parameters of that model, and then use the simple formulas for one step ahead predictors for AR's to get prediction errors $e(1), \ldots, e(n)$, and then use them as substitutes for the ϵ's. Thus, for example, if $p > q$, we can regress $y = (X(p+1), \ldots, X(n))^T$ on

 Copyright ©1999 by H.J. Newton

the matrix

$$\begin{bmatrix} -X(p) & \cdots & -X(1) & e(p) & \cdots & e(p-q) \\ -X(p+1) & \cdots & -X(2) & e(p+1) & \cdots & e(p-q+1) \\ \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\ -X(n-1) & \cdots & -X(n-p) & e(n-1) & \cdots & e(n-q) \end{bmatrix}.$$

15.1.3 Maximum Likelihood Methods

The most commonly used method for finding estimators in statistics is the maximum likelihood method (MLE). In time series, we assume that our realization is Gaussian and covariance stationary with zero mean and so the probability density function of data x is given by

$$f_X(x;\theta) = (2\pi)^{-n/2}|\Gamma_n|^{-1/2}e^{-x^T\Gamma_n^{-1}x},$$

where $\Gamma_n = \text{Toepl}(R(0), R(1), \ldots, R(n-1))$, the notation $|\Gamma_n|$ denotes the determinant of Γ_n, and the vector θ contains the parameters of the model to be estimated. This function is a function of both the data x and the parameters in θ. When doing estimation we interpret it as the "likelihood" of a particular value of θ for the observed set of data and use the notation $L(\theta; x)$ to make it clear we think of it as a function of θ for fixed data x. The MLE $\hat{\theta}$ is the value of θ that maximizes L.

Note that the vector θ does not appear explicitly in the expression for L but rather the R's do. But $R(v)$ is a function of θ (recall for example the formulas for $R(v)$ in terms of β and σ^2 for an $MA(1)$).

 Copyright ©1999 by H.J. Newton

Actually calculating an MLE is very difficult numerically as L is not a linear function of θ. Thus we use iterative nonlinear optimization methods. In fact we use methods that require no derivatives. Such methods require starting values for θ and we usually use method of moments estimators (if they exist) as the starting values. In many situations, we can just use a vector of zeros as the starting values.

(See the DTARMA command.)

15.2 Standard Errors, Confidence Intervals and Tests of Hypotheses

Anytime one produces estimates of a parameter, it is important to also report the standard error of the estimate, that is, the square root of its ensemble variance, that is,

$$\mathrm{se}(\hat{\theta}) = \sqrt{\mathrm{Var}(\hat{\theta})},$$

because this allows us to attach a "margin for error" to the estimate, as well as to find confidence intervals for the estimate

$$\hat{\theta} \pm Z_{\alpha/2}\mathrm{se}(\hat{\theta}).$$

Further, the test statistic for testing the hypothesis

$$\mathrm{H}_0 : \theta = \theta_0$$

 Copyright ©1999 by H.J. Newton

is given by

$$Z = \frac{\hat{\theta} - \theta_0}{\text{se}(\hat{\theta})},$$

(which is rejected if $|Z| > Z_{\alpha/2}$).

The formulas for standard errors for AR and MA estimators are simple, while those for $ARMA$ processes are quite complex. The main estimation commands in timeslab each produce standard errors for the estimates they produce (the command COEFFSD is a general purpose command for finding standard errors for $ARMA$ estimates).

15.3 Determining The Order of an ARMA Model

In this section we consider two methods for choosing the form of an $ARMA$ model.

15.3.1 The AIC Criterion

To judge the goodness of an $ARMA(p, q)$ model, we first get estimates $\hat{\alpha}$, $\hat{\beta}$, and $\hat{\sigma}^2_{p,q}$ of the parameters of the model and then calculate the Akaike Information Criterion

$$\text{AIC}(p, q) = n \log \hat{\sigma}^2_{p,q} + 2(p + q).$$

The first part of this model is a measure of how well the model fits the data (it is like a sum of squares of residuals and will decrease as p and q

 Copyright ©1999 by H.J. Newton

increase) while the second is a "penalty" term for adding more parameters to the model.

We choose the model that has the smallest AIC.

15.3.2 Stepwise ARMA Selection

For many time series, an $ARMA$ model with some of its coefficients equal to zero fits better than the "full" model. For example an $ARMA(3,2)$ with AR lags 1 and 3 and MA lag 2 is

$$X(t) + \alpha_1 X(t-1) + \alpha_3 X(t-3) = \epsilon(t) + \beta_2 \epsilon(t-2),$$

and we could write $X \sim SARMA(1,3,-2)$, where $SARMA$ stands for Subset $ARMA$ and in the parentheses we list the indices of the AR and MA terms in the model with the MA indices having minus signs.

If we have maximum values for the AR and MA lags allowed in a model (for example, 20 and 20), we could try all possible subsets of these lags and calculate AIC for each one. Then we could choose the $SARMA$ model having smallest AIC. Unfortunately, there are 2^M possible subsets of M lags which is prohibitively large in most circumstances. Thus we borrow a "stepwise" procedure from regression analysis to find what often turns out to be a good $SARMA$ model. The process works as follows.

 Copyright ©1999 by H.J. Newton

1. We start with no lags in the model (that is, with a white noise model). At the first step, we pick a single lag as a contender to be added to the model (the one that would give the smallest AIC). We then use a test of hypotheses (a χ^2 test) to determine if the contender should be added. If so, we add the lag and go to the next step. If not, then we stop and conclude that white noise is the best model.

2. Once we have a lag in the model, we can do a series of identical steps. At each step, we pick a contender for deletion and a contender for addition to the model. We then see if the deletion contender should be deleted. If so, we delete it and go to the next step. If not, we see if the addition contender should be added. If so, we add it and go to the next step. If not, we stop and coclude the current model is the best fitting model.

3. We also stop if we conclude that we should delete the deletion contender and it was just added, or vice-versa. We also must set a maximum number of steps to run or we could get into an infinite loop of adding and deleting the same set of lags.

The ARMASEL **Command**

This command does the stepwise procedure described above. Its arguments are as follows:

```
ARMASEL(x,n,mr,ma,k1,k2,kopt,pval,p,q,
        alpha,beta,rvar,ier)
```

 Copyright ©1999 by H.J. Newton

Before using ARMASEL it is important to subtract the mean of the data set. The first two arguments are the data set and sample size, while `mr` and `ma` are used to determine maximum lags that can be chosen in a model (`mr-ma` is the maximum lag, and `ma` is the order of a long AR model fit to the data before doing a regression type analysis; values of 60 and 20 are usually satisfactory.) The arguments `p`, `q`, `alpha`, `beta`, and `rvar` on output are the orders, coefficients, and noise variance of the "full" $ARMA$ version of the model fit by the command (zeros are placed in `alpha` and `beta` for lags not in the model). The argument `pval` is only used for the case of `kopt=0` (see case 3 below). The other arguments depend on how the command is used.

ARMASEL can be used in many ways including:

1. **Estimating parameters of a specified subset model:** In this case, `k1` and `k2` are the number of AR and MA lags, respectively, while `kopt=-(k1+k2)` and `alpha` and `beta` are vectors of length `k1` and `k2` containing the AR and MA lags, respectively.

2. **Finding the best model having a specified number of lags:** The "best" `k1` AR lags and `k2` MA lags are contenders to be in the model while `kopt` is the number of lags wanted in the model, that is, the command will continue until exactly `kopt` lags are in the model. For example, a subset AR model having exactly three lags can be specified by `k2=0` and `kopt=3`.

 Copyright ©1999 by H.J. Newton

3. **Finding the best model with specified maximum number of possible lags:** This is the same as the previous case except `kopt=0`. This is the method to be used if you don't want to specify anything about the model to be chosen except `k1` and `k2`. The argument `pval` is a probability and is thus between 0 and 1. It specifies how easy it is for a lag to enter the model during the stepwise procedure. A value close to 0 (1) makes it easy (hard) for a lag to enter.

15.4 The SEASEST and SEASPRED Commands

Once a model has been suggested by AIC and/or the stepwise $ARMA$ procedure, we need to estimate its parameters and find the residuals (prediction errors in predicting $X(1)$, $X(2)$ from $X(1)$, $X(3)$ from $X(1)$ and $X(2)$, and so on). If these residuals are judged to be white noise, then the chosen model is judged to be adequate.

The SEASEST command uses an approximate MLE procedure to find estimates of a very wide class of models. It can fit full $ARMA$, subset $ARMA$ (including AR or MA versions of these), or combinations of full and subset models.

An example of this combination of full and subset model is the airline data, where a number of investiagtors have found that the $\hat{\rho}$'s for the

 Copyright ©1999 by H.J. Newton

series

$$W(t) = (1 - L)(1 - L^{12}) \log(X(t)),$$

that is, the first and twelfth difference of the logarithm of the data are

```
  1 | -0.34  0.11 -0.20  0.02  0.06  0.03
  7 | -0.06  0.00  0.18 -0.08  0.06 -0.39
 13 |  0.15 -0.06  0.15 -0.14  0.07  0.02
 19 | -0.01 -0.12  0.04 -0.09  0.22 -0.02
 25 | -0.10  0.05 -0.03  0.05 -0.02 -0.05
```

which suggests a "multiplicative" model of the form

$$W(t) = (1 + \beta L)(1 + \gamma L^{12})\epsilon(t),$$

that is, a combination of an $MA(1)$ and a $SARMA(-12)$ model (it suggests this because the $\hat{\rho}$'s for $v = 1$ and $v = 12$ are "large" and the one for $v = 13$ seems to be the product of the ones for $v = 1$ and $v = 12$ and this would be what you would get for the true ρ's for the model for W).

Once the parameters of a model have been estimated and the residuals judged to be white noise, the SEASPRED command can be used to produce forecasts of the future of the series. The input to the command is the original data and the specification of the model together with the estimates produced by SEASEST.

 Copyright ©1999 by H.J. Newton

15.4.1 Specifying a Model in SEASEST and SEASPRED

The SEASEST command takes as input transformed data and vectors `ords` and `lags` specifying the model to be used and the lags in the model (if any), and a vector `coeffs` of starting values for the parameters (which get replaced by the final values on output).

The SEASPRED command takes as input the original data as well as the vectors `ords` with three new elements appended, `lags`, and `coeffs` containing the output estimates from SEASEST.

Thus the difficult part of using the commands is the specification of `ords`. The general class of models is of the form of a multiplicative $ARMA$ model which has full and subset AR parts and full and subset MA parts. The vector `ords` contains p, P, q, and Q where p and q are the full $ARMA$ orders and P and Q are the number of AR and MA lags in the $SARMA$ part (usually these are not all nonzero). For SEASEST there is a fifth element; namely a 1 if differencing was not done and 0 if it was.

The vector `coeffs` has starting values for the `p+P+q+Q` coefficients in the model (as well as another 0 if differencing was not done).

For SEASPRED there are three more elements of `ords`; namely the number of times first differencing was done, the number of times Sth differencing was done, and the value of S. For SEASPRED the vector

 Copyright ©1999 by H.J. Newton

coeffs has the output coefficients from SEASEST followed by a constant called m and the value λ for the power transform ($\lambda = 0$ means logarithm, $\lambda = 1$ means no power transform). The number m is the constant added to the data when doing the power transform in order to make the data positive before doing the transform (usually $m = 0$).

15.5 The IDNEW and PREDS Macros

These macros make it easy to perform the complicated analysis described above. IDNEW guides you through the transformation and model selection portions of the analysis, while the PREDS macro does the estimation, residual checking, and prediction portions of the analysis.

 Copyright ©1999 by H.J. Newton