**Lecture 9 Model Specification and Forecasts**

# 9.1 Model specification and parameter estimation

In Section 7.2 we used the sample *acf* to detect the autocorrelation of a time series. The sample *acf* is an estimator of the theoretical *acf* that was studied in Lecture 8 for various stationary time series models.

The sample *acf* is a good indicator of the order of an MA(q) process since the *acf* = 0 for k > q. It is not so for an AR(p) process, because the *acf*  ≠ 0 for all k. Thus, for an MA process, it is possible to identify the pattern in the sample *acf* and fit a model with the same pattern in the *acf*. For an AR process, we need a particular tool to determine the order, the ***partial autocorrelation function (pacf)***,denoted by , which is, for a given time series {}, the correlation coefficient of  and , conditional on , , …, . In general, for an AR(p) process,  = 0 for k > p. However, for an MA(q) process  is non-zero for all k but decays exponentially to zero. The *pacf*  is estimated by the sample *pacf* .

For a given nonstationary time series, the nonstationarity will be apparent in the time series plot. The sample *acf* is also a good indicator of nonstationarity. To identify an ARIMA model, we shall inspect the first difference of the series. If the differenced series do not appear to support a stationary ARMA model, we then take another difference and again check the differenced series, and so on.

Suppose that, for a given time series of length n, the model has been specified as an ARIMA(p, d, q). Then the parameters in the model, such as the φ’s, θ’s, and  , are to be estimated. The commonly used estimation method is the method of maximum likelihood.

**Example 9.1** Re Example 7.6, let {} be the residuals.

|  |
| --- |
| z<-co2.14.fit$residuals  plot.ts(z)  acf(z)  pacf(z) |
|  |
| z.fit<-arima(z,order=c(2,0,0))  z.fit |
| Call:  arima(x = z, order = c(2, 0, 0))  Coefficients:  ar1 ar2 intercept  0.6679 0.2869 0.3001  s.e. 0.0735 0.0746 0.4303  sigma^2 estimated as 0.07506: log likelihood = -22, aic = 52 |
| coeftest(z.fit) # in {lmtest} |
| z test of coefficients:  Estimate Std. Error z value Pr(>|z|)  ar1 0.667948 0.073487 9.0893 < 2.2e-16 \*\*\*  ar2 0.286938 0.074561 3.8483 0.0001189 \*\*\*  intercept 0.300098 0.430312 0.6974 0.4855552  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1 |
| z.fit<-arima(z,order=c(2,0,0),include.mean=FALSE)  z.fit |
| Call:  arima(x = z, order = c(2, 0, 0), include.mean = FALSE)  Coefficients:  ar1 ar2  0.6704 0.2851  s.e. 0.0735 0.0748  sigma^2 estimated as 0.0753: log likelihood = -22.28, aic = 50.55 |
| coeftest(z.fit) |
| z test of coefficients:  Estimate Std. Error z value Pr(>|z|)  ar1 0.670368 0.073528 9.1172 < 2.2e-16 \*\*\*  ar2 0.285077 0.074802 3.8111 0.0001384 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1 |

Thus the model is an AR(2) process and the estimated model is given by

 = 0.67 + 0.29 . █

# 9.2 Model diagnostics and over-fitting method

As always, the residual is defined as *Residual = Observed – Fitted*. The residual analysis is based on the following argument: If the fitted model is correct and the parameter estimates are close to the true values, then the residuals, as the counterpart of the white noise, should be i.i.d. normal random variables.

# Over-fitting method: Suppose that an AR(2) model is suggested with estimates and . To confirm the model’s appropriateness, we “overfit” with an AR(3) model with estimates , and . The AR(2) model would be finally adopted if

1.  is not significantly different from zero;
2.  ≈  and  ≈ .

**Example 9.2** Re Example 9.1, the sample *acf* shows no autocorrelation in the residuals and the Q-Q plot indicates that the distribution of the residuals is approximately normal. Notice also that the residual plot exhibits a rectangular shape, implying a constant variance over time. Moreover, the over-fitting method confirms that the AR(2) is appropriate.

|  |
| --- |
| plot.ts(z.fit$residuals)  acf(z.fit$residuals)  qqnorm(z.fit$residuals) |
|  |
| z.fit1<-arima(z,order=c(3,0,0),include.mean=FALSE)  coeftest(z.fit1) |
| z test of coefficients:  Estimate Std. Error z value Pr(>|z|)  ar1 0.633755 0.076211 8.3158 < 2e-16 \*\*\*  ar2 0.202158 0.089440 2.2603 0.02380 \*  ar3 0.128402 0.077303 1.6610 0.09671 .  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1 |

█

# 9.3 Forecasting and forecast errors

# Given a observed time series {, , …, }, the *minimum mean square error forecast* is defined by

 = E(|, , …, )

where *t* is called the ***origin*** of the forecast and *L* is the ***lead time*** for the forecast.

We use the stationary AR(1) process {} as an example to show how the forecasts are derived. Suppose that

 - μ = φ( - μ) + .

Consider *L* = 1. Since

 - μ = φ( - μ) + ,

by taking conditional expectations for both sides, we have

E( - μ|, , …, ) = E[φ( - μ) + |, , …, ],

Then

E(|, , …, ) - μ

= φE(|, , …, ) - φμ + E(|, , …, )

= φ - φμ .

Therefore

 = μ + φ( - μ).

In words, a proportion φ of the current deviation from the process mean is added to the process mean to forecast the next process value.

Consider L = 2. Since

 - μ = φ( - μ) + 

and,

E(|, , …, ) - μ

= φE(|, , …, ) - φμ + E(|, , …, )

= φE(|, , …, ) - φμ .

Therefore

 = μ + φ[ - μ].

In general, for L ≥ 1,

 = μ + φ[ - μ].

By iterating backwards on L, we have

 = μ + φ( - μ).

In words, the current deviation from the mean is *discounted* by a factor φ, whose magnitude decreases with increasing lead time L. The discounted deviation is then added to the process mean to produce the lead L forecast. If μ = 0, then  = μ + φ.

Let  =  -  be the L-step-ahead forecast error. The main results are:

1. for stationary ARMA(p, q) models,

*  ≈ μ, for large L
* E[] = 0, i.e., the forecasts are unbiased
* Var[] ≈ Var(), for large L .

1. For nonstationary ARIMA(p, d, q) models (d > 0), the forecasts are unbiased and the variances of the forecast errors are unbounded.

The results are illustrated in the following two figures.





**Example 9.3** Re Examples 9.1-2, forecasts can be produced by the package {forecast}.

|  |
| --- |
| forecast(z.fit,24) # in {forecast} |
| Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  169 1.091412 0.7397413 1.443083 0.55357809 1.629246  170 1.050307 0.6269277 1.473686 0.40280448 1.697809  ……  191 0.5039820 -0.44487901 1.452843 -0.94717572 1.955140  192 0.4866457 -0.46982355 1.443115 -0.97614779 1.949439 |
| plot(forecast(z.fit,24)) |
|  |
| plot(forecast(z.fit,250)) |
|  |

By combining the results in Examples 7.3 and 9.1, we have the complete model for the CO2 concentrations data

 = + 

where *t* is the time,  the seasonal trend (detail in Example 7.3) and  the AR(2) process (detail in Example 9.1). The following are the forecasts of CO2 concentrations for the next 2 years with time origin 168 and lead time *L* = 1, 2, …, 24.

|  |
| --- |
| co2.14.fit<-lm(co2.14~0+time+factor(month.14))  dp<-predict(co2.14.fit, data.frame(time=c(169:192), month.14=factor(c(1:12,1:12))),interval='confidence')[,1]  sf<- forecast(z.fit,24)  m<-dp+ sf$mean  l<- dp+ sf$lower[,2]  u<- dp+ sf$upper[,2]  plot(c(169:192),dp,type="b",ylim=c(372,383),col='red',ylab='dp=red, sf=green')  lines(c(169:192),m,type="b",col='green')  lines(c(169:192),l)  lines(c(169:192),u) |
|  |

█

**9.4 Further discussion**

We may compare the two models

=  (Example 7.3)

= +  (Example 9.1)

using their respective forecasts. The idea is to choose a forecast origin so that some actual values can be compared with forecast values. We would expect that the actual values are within the prediction limits, if the model is “sound”.

Let *obs* be the last 12 CO2 observations:

374.68 375.63 376.11 377.65 378.35 378.13

376.62 374.50 372.99 373.00 374.35 375.70

Forecasts are made based on each of the two models with time origin 157 and lead time *L* = 1, 2, …, 12.

|  |
| --- |
| = |
| y<-MaunaLoaCO2[c(1:156),]  y.fit<-lm(co2.14~0+time+factor(month.14))  yf<-predict(y.fit, data.frame(time=c(157:168), month.14=factor(c(1:12))),interval='confidence')  plot(c(157:168),yf[,1],type='b',ylim=c(368,380),ylab='obs=red', xlab='regression only')  lines(c(157:168),obs,type='b',col='red')  lines(c(157:168),yf[,2])  lines(c(157:168),yf[,3]) |
|  |

|  |
| --- |
| = + |
| z<-y.fit$residuals  z.fit<-arima(z,order=c(2,0,0), include.mean=FALSE)  z.fit  zf<- forecast(z.fit,12)  m<-yf[,1]+ zf$mean  l<- yf[,1]+ zf$lower[,2]  u<- yf[,1]+ zf$upper[,2]  plot(c(157:168),m,type="b",ylim=c(368,380), ylab='obs=red',xlab='regression with AR(2)')  lines(c(157:168),obs,type="b",col='red')  lines(c(157:168),l)  lines(c(157:168),u) |
|  |

**Exercises**

* 1. Re-do the examples in this talk.
  2. We have seen that an AR(2) model in Example 9.1 is adequate. To illustrate model misspecification, attempt to fit an AR(1) model to the series. Check the residuals for “whiteness” by looking at their time series plot, *acf* and the normality plot. Comment on the results.
  3. Re data file *ts.series.csv*, for each of the time series,

1. Fit an ARIMA model and estimate the parameters;
2. Perform diagnostic checking on the model;
3. Use the over-fitting method to confirm the model’s appropriateness.

**References**

* Chatfield, C. (2004), *The Analysis of Time Series: An Introduction*, CHAPMAN & HALL.
* Cryer, J. D. (1986), *Time Series Analysis*, PWS-KENT.
* Millard, S.P. and Neerchal, N. K. (2000), *Environmental Statistics with S-PLUS*, Chapman & Hall.