Time Series Regression 8

In Chapter 7 we discussed the nonstationary ARIMA and seasonal models. These models are nonstationary because the AR-characteristic equation has one or more roots on the unit circle. In the current chapter we focus on the nonstationary process for which the mean changes with time. One such time series model is

$$X_t = b_0 + b_1 t + Z_t, \tag{8.1}$$

where Z_t is a zero-mean stationary process. If Z_t were white noise, then (8.1) would be a simple linear regression (SLR) model with independent variable time (t) However, the assumption regarding the Z_t process is that it is stationary, which allows for correlated errors. Model (8.1) is sometimes called a *regression with correlated errors.* Model (8.1) is a special case of the model

$$X_t = s_t + Z_t, \tag{8.2}$$

where s_t is a deterministic (non-random) *signal* and Z_t is a zero-mean stationary process. Model (8.2) is often referred to as a signal+noise model. Model (8.1) is the special case in which $s_t = b_0 + b_1 t$, but other possible signals are $s_t = b_0 + b_1 t + b_2 t^2$, $s_t = \log(t), X_t = \cos(2\pi ft) + Z_t$, and so forth. For the signal+noise model in (8.2), $\mu_t = E[X_t] = E[s_t] + E[Z_t] = s_t$, because s_t is nonrandom and $E[Z_t] = 0$. Unless s_t is a constant function, then the mean, μ_t, changes with time t, and consequently X_t in (8.2) is *nonstationary*. In this chapter we will direct our attention to cases in which s_t is linear or a sinusoidal curve.

Key Point:

1. We will model the stationary error process Z_t using an AR(p) model.
2. Z_t could be modeled as an ARMA(p,q) process, but it is common to use an AR(p) model in this case.

8.1 LINE+NOISE MODELS

In this section we consider the line+noise model in (8.1). While this appears to be an easy model to fit to data and to interpret, there are issues that need to be considered. In Section 5.1.1.6 and 7.1.1.5, it was noted that data from models, $(1-\phi_1 B)Z_t = a_t$, can produce realizations with trending behavior when ϕ_1 is slightly

DOI: 10.1201/9781003089070-8

less than or equal to one (see Figure 7.3(a)). This "random trending" is amplified in the presence of one or more unit roots, but in this chapter we will focus on the situation in which Z_t is stationary.

Question: Given a realization with trending behavior, how can we tell whether the trending is due to random trending or due to an underlying regression line?

Model (8.1) seems to be the ideal model for helping answer this question. That is, using (8.1), it seems that we can let the "model decide" the source of the trending behavior. For example, the model has the option of attributing any trending (a) to a deterministic signal ($b_1 \neq 0$) or (b) to a random trend in Z_t ($b_1 = 0$). In this section we will discuss techniques for answering the above question.

Before addressing the problem of deciding whether there is an underlying "linear signal" in the data, it is important to understand the following Key Points.

Key Points:

1. The finding of a significant slope implies that the process is nonstationary because the mean of the process, μ_t, is non-constant and depends on t.
   - If $\hat{b}_0$ and $\hat{b}_1$ denote the estimates of b_0 and b_1 (and it is determined that $\hat{b}_1$ is significantly different from zero), then μ_t is estimated by $\hat{\mu}_t = \hat{b}_0 + \hat{b}_1 t$.
2. Given that conditions do not change, the finding of a significant slope suggests that forecasts would predict the existing trend line to continue.
3. Forecasting a functional form (such as a line) to continue into the future is an example of ***extrapolation.***
   - Good statisticians/data analysts know *to be careful* when extrapolating beyond the range of the observed independent variables (in this case, time).

Before dealing with the issue of forecasting, we address the problems related to the testing for the presence of a significant linear signal.

8.1.1 Testing for Linear Trend

In the following we discuss possible techniques the analyst might use to decide whether an apparent linear trend in a set of data implies that a (deterministic) linear trend term should be included in the model.

8.1.1.1 Testing for Trend Using Simple Linear Regression

We begin by using "what we already know". That is, for a simple linear regression (SLR), the standard procedure is to test for significance of the slope. In that setting, the hypothesis $H_0 : b_1 = 0$, is tested using the statistic $\hat{b}_1 / SE\left(\hat{b}_1\right)$, which has a t-distribution when the errors are uncorrelated and normally distributed. See for example Moore, McCabe, and Craig (2017). In our setting, the independent variable is time t, and the realization is a sample from the dependent variable X_t.[1] In this section we examine the validity of using the standard simple linear regression t-test to test the significance of the slope in (8.1).

1 Note the possibility of notational confusion because in usual simple linear regression, the dependent variable is typically denoted by Y and the independent variable is called X. Here, X_t denotes the dependent variable.

Key Point: We use the simple linear regression approach here although we acknowledge the fact that the *assumption of uncorrelated errors is violated* (unless the AR(p) distributed errors, Z_t, are white noise).

The ***tswge*** function **`slr.wge`** performs this t-test and outputs the least squares estimates of b_0 and b_1. Using the SLR testing approach, we would conclude that there is a "deterministic" linear trend if H_0 is rejected. As an example, consider the model

$$X_t = 10 + .08t + Z_t, \tag{8.3}$$

where Z_t is a realization from the AR(1) model $(1 - .95B)Z_t = a_t$ with $\sigma_a^2 = 1$. The realization in Figure 8.1(a) was obtained using the R code

```
z=gen.arma.wge(n=100,phi=.95,sn=447,plot=FALSE)
t=1:100
line=10+.08*t
x=line+z
plotts.wge(x)
```

Figure 8.1(b) shows the realization in Figure 8.1(a) along with the least squares regression line whose coefficients are found using the simple linear regression function **`slr.wge`** in ***tswge***. Using the following command:

```
reg=slr.wge(x)
```

we obtain from the output

```
reg$b0hat (Intercept) 11.94136
t_test$b1hat Time 0.09125542
```

The p-value for the test, $H_0 : b_1 = 0$, is given by

```
t_test$pvalue [1] 7.434276e-23
```

In this case obviously $p < .001$. Figure 8.1(b) is obtained using the commands

```
plotts.wge(x)
fit=reg$b0hat+t*reg$b1hat
points(fit,type='l')
```

Consequently, the regression line has a highly significant slope based on the usual SLR test statistic. Figure 8.1(c) shows the residuals $x(t) - 11.941 - .091t$. These residuals should cause concern because they are clearly not uncorrelated, which, as noted, is a key assumption in the simple linear regression t-test for the significance of $\hat{b}_1$. Figure 8.1(c) can be produced using the commands

```
resid=x-fit
plot(resid,type='l')
abline(h=0)
```

Key Points:

1. The p-value output from **`slr.wge`** is based on an assumption of uncorrelated residuals which is often violated in time series data.
2. For this reason, p-values obtained using SLR methods should be viewed skeptically.

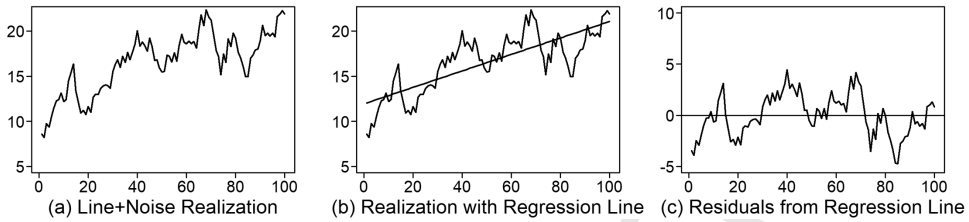

FIGURE 8.1 (a) Realization from line+noise model in (8.3), (b) realization in (a) with fitted regression line, and (c) residuals from fitted regression line.

It is reasonable to ask the following question.

Question: "Does the lack of uncorrelated residuals really make a difference or is that assumption just something that statisticians worry about?[2]

Example 8.1 Analysis of realizations from a line+noise model and an AR(1) model
To assist in examining the impact of correlated residuals, we consider the two models:

Model (a): $X_t = 10 + .1t + Z_{1t}$
Model (b): $X_t = 10 + Z_{2t}$

where Z_{1t} and Z_{2t} are the AR(1) processes $(1-.9B)Z_{1t} = a_t$ and $(1-.95B)Z_{2t} = a_t$, respectively, both with $\sigma_a^2 = 1$. Clearly, Model (a) has a deterministic linear signal while Model (b) does not. Models (a) and (b) and the ***tswge*** code to generate the realizations in Figures 8.2(a) and (b), are given below.

Model (a): $X_{1t} = 10 + .1t + Z_{1t}$ where z_{1t} is the AR(1) realization generated by the command

```
z1=gen.arma.wge(n=100,phi=.90,sn=852)
time=1:100
x1=10+.1*time+z1
```

Alternatively, the ***tswge*** command

```
x1=gen.sigplusnoise.wge(n=100,b0=10,b1=.1,phi=.9,sn=65987)
```

generates the same realization, **x1**.

Model (b): $X_{2t} = 10 + Z_{2t}$ where z_{2t} is the AR(1) realization generated by the ***tswge*** command

```
z2=gen.arma.wge(n=100,phi=.95,sn=6587)
time=1:100
x2=10+z2
```

2 Consider the one-sample *t*-test. The mathematical validity of the *t*-test requires the data to be from a normal distribution. However, because of the central limit theorem, the normality assumption is not critical for validity in the case of "large" sample sizes (often taken to be $n > 30$). The normality assumption is still required for the test statistic to *truly* follow a *t* distribution, but the *t*-test performs well for large sample sizes in the case of non-normal data. The question here is whether the assumption of uncorrelated residuals is similarly a condition that only has underlying mathematical implications.

Figures 8.2(a.1) and (a.2) show a decomposition of x_{1t}. That is, Figure 8.2(a) is the sum of the deterministic linear signal, $s_t = 10 + .1t$, plotted in Figure 8.2(a.1) and the AR(1) realization, z_{1t}, in Figure 8.2(a.2). The AR(1) realization z_{1t} has short-term, random wandering behavior and is positively correlated data. Because the mean, $\mu_t = s_t = 10 + .1t$ depends on time, Model (a) is nonstationary.

Model (b) has a horizontal signal, $s_t = 10$, plotted in Figure 8.2(b.1). The realization in Figure 8.2(b) is obtained by adding 10 to each value in the zero-mean error series in Figure 8.2(b.2). For Model (b), $\mu_t = 10$ does not depend on time, and thus x_{2t} is a realization from a *stationary* process (and $b_1 = 0$).

Applying the commands

```
slr.wge(x1)
slr.wge(x2)
```

we obtain $p < .001$ in each case. The decision to reject $H_0 : b_1 = 0$ is correct for realization `x1`. However, in the case of realization `x2` there is no deterministic trend, and the random upward trending behavior has "tricked" the *t*-test into rejecting reject $H_0 : b_1 = 0$, even though it is true.

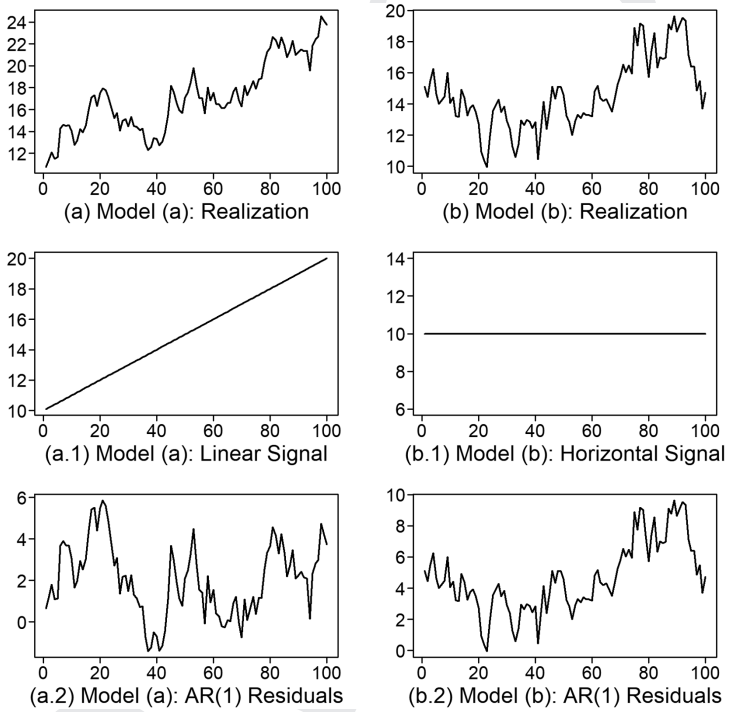

FIGURE 8.2 (a) Realization from x_{1t} from Model (a), (b) realization x_{2t} from Model (b), (a.1) the line $s_t = 10 + .1t$ and (b.1) the horizontal line $s_t = 10$; (a.2) AR(1) realization z_{1t} and (b.2) AR(1) realization z_{2t}.

Key Points:

1. Because positively correlated stationary time series can show trending behavior in the absence of an actual deterministic trend (as was the situation in Model (b)), it is difficult to correctly detect trend in the presence of positively correlated residuals.

2. The model selected has a major impact on forecasts!

   - Forecasts from a stationary model tend toward the sample mean (see Section 6.2.5)
   - We will see that forecasts from a line+noise model such as (8.1) will tend toward the fitted regression line and will tend to move upward or downward depending on the estimated slope.

8.1.1.2 A t-test Simulation

The situation in Model (b) brings up the following questions:

Questions:

(1) How often does the standard *t*-test detect a trend in stationary AR(1) realizations (which do not have trend components)?
(2) How important is the "uncorrelated" assumption when using the standard SLR *t*-test on time series data?

A simulation study investigated these questions. Realizations were generated from the model $X_t = 10 + Z_t$ where Z_t is a zero mean AR(1) model with $\sigma_a^2 = 1$ and where ϕ_1 takes on the values .8, .9, .95, and .99. These AR(1) models, with positive ϕ_1, have realizations that are positively autocorrelated. In the simulation study, realization lengths $n = 100$, 200, and 500 were used, and 1,000 realizations were generated from each of the twelve "ϕ_1 and realization length" combinations. For each realization the hypothesis $H_0 : b_1 = 0$ was tested at the $\alpha = .05$ level using SLR methods. Notice that the models are all stationary with mean 10 (that is, *the null hypothesis is true in all cases*), and whenever a trend was detected by obtaining a significant slope (rejecting $H_0 : b_1 = 0$), a Type I error was made. That is, the null hypothesis has been rejected when it is true.

For a given "ϕ_1 /realization length" combination, the value in Table 8.1 is the percentage of realizations out of 1,000 for which the standard *t*-test rejected the null and found a trend (significant slope). Because these are designed to be $\alpha = .05$ level tests and the null hypothesis is true, a trend should be detected about 5% of the time. If positive correlation in the residuals is not a "big deal", then we would expect each of the tabled values to be about 5%. A quick glance at the table says that something is very wrong!

TABLE 8.1 Percentage of times out of 1,000 replications that the *t*-test incorrectly found a significant slope for the given stationary AR(1) models and tabled realization lengths. The tests were all conducted at the $\alpha = .05$ level

		ϕ_1			
		.8	.9	.95	.99
n	100	50.6%	66.1%	74.7%	84.3%
	200	50.8%	63.5%	76.3%	87.7%
	500	53.1%	65.9%	75.9%	90.7%

Note that the SE of tabled values is at most 1.58%.[a]

[a] A conservative SE of a tabled value is $\sqrt{.5 * .5 / 1000} \times 100\% = 1.58\%$. See Moore et al. (2017).

The realizations in the simulation were generated from the model $X_{2t} = 10 + Z_{2t}$. Recall that Figure 8.2(b) shows a particular realization for the case in which Z_{2t} is an AR(1) model with $\phi_1 = .95$. That

is, it is a stationary realization (the slope is zero), but for which correlation structure produced a "random trending behavior" that "tricked" the *t*-test into finding a significant slope. The questions above had to do with how often this might happen. For example, the tabled value for $\phi_1 = .95$ and $n = 100$ is 66.1% (Wow!) That is, about two-thirds of realizations of length $n = 100$ from the stationary model, $(1 - .95B)X_t = a_t$, will have enough "trending behavior" for the *t*-test to (incorrectly) find a significant slope. Thus, we see that incorrectly finding a significant slope occurs *much* more often than the nominal 5%, and consequently, the usual *t*-test is failing to perform properly in this case.

Key Points:

1. The percentage of cases in which a hypothesis test actually rejects the null hypothesis when it is true is called the "observed significance level".
2. The values in Table 8.1 are observed significance levels of the *t*-test under each "ϕ_1 and realization length" scenario.

Examination of all tabled values in Table 8.1 shows that the results of the *t*-test are uniformly ***awful***. In the terminology of the Key Point above, the "observed significance levels" are *unacceptably high*. At the extreme in the table, for realizations of length $n = 500$ and $\phi_1 = .99$, the SLR test (incorrectly) detected a significant trend about 90% of the time (instead of the hoped for 5%)!

Key Point: The finding of a significant slope when using the *t*-test on time series data "cannot be trusted."

QR 8.1 A t-test Simulation

Important Note:
Before continuing, it is critically important to understand the implications of the information in Table 8.1. Hypothesis testing is designed so that if the null hypothesis is rejected, say at the $\alpha = .05$ level, then the analyst can feel "confident" that the null hypothesis is not true. In the current case, rejection of the null hypothesis should provide confidence that there is a real deterministic linear trend signal in the data. In each simulation scenario, even though the null hypothesis, $H_0 : b_1 = 0$, is true, there is *at least a 50% chance of rejecting the null hypothesis*. That is, if the standard SLR *t*-test for slope is used in this "correlated errors" setting and the result of the standard *t*-test "says" to reject the null hypothesis, the analyst should have *no confidence* that there really is a deterministic linear signal driving the data.

Key Points:

1. The requirement of uncorrelated residuals associated with the usual *t*-test for slope is a Big Deal!

2. ***Don't use the usual t-test for detecting a trend in time series data unless the residuals from the line appear to be uncorrelated.***
3. What is needed are tests of $H_0 : b_1 = 0$ that take into account or adjust for the correlation in the residuals.

8.1.1.3 Cochrane-Orcutt Tests for Trend

The simulation study in Section 8.1.1.2 shows that the standard *t*-test for slope is not appropriate for the time series setting of (8.1). This is a well-known fact, but we used it here because using the SLR is the first thing someone with statistical (but not time series) training would think of doing. Obviously, there is a need for better tests that take into consideration the correlation in the residuals. The most common such test is the Cochrane-Orcutt (1949) test. We consider again the line+noise model, $X_t = b_0 + b_1 t + Z_t$, given previously in (8.1). The outline below is for the case in which Z_t satisfies the AR(1) model $\left(1 - \phi_1 B\right) Z_t = a_t$. The AR(1) is used in the outline for simplicity, but the restriction to AR(1) errors is not imposed in the analyses we will discuss. The version of the Cochrane-Orcutt (CO) procedure using an AR(1) to model the residuals is as follows:

(1) Fit a linear regression line to the original data by calculating the least squares estimates $\hat{b}_0$ and $\hat{b}_1$.
(2) Find $\hat{Z}_t = X_t - \hat{b}_0 - \hat{b}_1 t$ (which we will assume here to be approximately distributed as AR(1)).
(3) Calculate $\hat{\phi}_1$ by fitting an AR(1) model to $\hat{Z}_t$.
(4) Calculate $\hat{Y}_t = \left(1 - \hat{\phi}_1 B\right) X_t$, $t = 2, \ldots, n$.

Note that $\hat{Y}_t = \left(1 - \hat{\phi}_1 B\right) X_t = \left(1 - \hat{\phi}_1 B\right)\left(b_0 + b_1 t + Z_t\right)$

$$= \left(1 - \hat{\phi}_1 B\right)\left(b_0 + b_1 t\right) + \left(1 - \hat{\phi}_1 B\right) Z_t$$

$$= b_0 + b_1 t - \hat{\phi}_1 b_0 - \hat{\phi}_1 b_1 \left(t - 1\right) + \left(1 - \hat{\phi}_1 B\right) Z_t,$$

using the facts that $Bb_0 = b_0$ and $Bt = t - 1$. Rearranging terms

$$\begin{aligned} \hat{Y}_t &= b_0 \left(1 - \hat{\phi}_1\right) + b_1 \left(t - \hat{\phi}_1 \left(t - 1\right)\right) + \left(1 - \hat{\phi}_1 B\right) Z_t \\ &= c + b_1 t_{\hat{\phi}_1} + g_t, \end{aligned} \tag{8.4}$$

where $c = b_0 \left(1 - \hat{\phi}_1\right)$, $t_{\hat{\phi}_1} = t - \hat{\phi}_1 \left(t - 1\right)$, and $g_t = \left(1 - \hat{\phi}_1 B\right) Z_t$.[3]

QR 8.2 Cochrane-Orcutt Method

[3] The AR(1) results above easily extend to the general case of an AR(p) fit to Z_t. If $\hat{\phi}(B) = 1 - \hat{\phi}_1 B - \cdots - \hat{\phi}_p B^p$, then $\hat{Y}_t = c + b_1 t_{\hat{\phi}} + g_t$, where $c = b_0 \left(1 - \hat{\phi}_1 - \cdots - \hat{\phi}_p\right)$, $t_{\hat{\phi}1} = t - \hat{\phi}_1 \left(t - 1\right) - \cdots - \hat{\phi}_p = 1 - \hat{\phi}_1 B - \cdots - {}_1 \left(t - p\right)$, $g_t = \hat{\phi}(B) Z_t$.

Some things to notice regarding (8.4) are the following:

(a) This is a new line+noise model with new constant, c, but with *the same slope* b_1
(b) The new "time" variable is $t_{\hat{\phi}_1}$
(c) The new residuals are denoted as g_t : Because $(1-\phi_1 B)Z_t = a_t$, can be inverted to obtain $Z_t = (1-\phi_1 B)^{-1} a_t$, then g_t can be expressed as

$$g_t = (1-\hat{\phi}_1 B)Z_t = (1-\hat{\phi}_1 B)(1-\phi_1 B)^{-1} a_t.$$

Because the operators $(1-\hat{\phi}_1 B)$ and $(1-\phi_1 B)^{-1}$ *nearly cancel* each other, it follows that g_t is similar to a_t and thus g_t should be "nearly uncorrelated".

In summary, the CO method uses a transformation that results in a new line+noise model with the same slope, b_1, and nearly uncorrelated residuals.Thus, to test $H_0 : b_1 = 0$, the CO method uses the usual regression t-test on this "new" data because the assumption of uncorrelated errors is reasonable. Using this transformed data, $H_0 : b_1 = 0$ is tested using the test statistic

$$t_{CO} = \frac{\hat{b}_1^{(CO)}}{SE(\hat{b}_1^{(CO)})}, \tag{8.5}$$

where $\hat{b}_1^{(CO)}$ is the least squares (regression) estimate of b_1 in the "new" model, and $SE\left(\hat{b}_1^{(CO)}\right)$ is its usual standard error.

Key Points:

1. The CO method employs a transformation to produce almost uncorrelated residuals.
2. ***tswge*** command **`co.wge`** implements the CO procedure by
   - finding the least squares regression line
   - computing the residuals, $\hat{Z}_t$, from the regression line
   - fitting the residuals with an AR(p) model with operator $\hat{\phi}(B)$ (using AIC-type measures to identify p). (Burg estimates are used because they always obtain a stationary solution.)
   - transforming the data to obtain $\hat{Y}_t = \hat{\phi}(B)X_t = c + b_1 t_{\hat{\phi}_1} + g_t$.
   - using (8.5) to test for the significance of the slope in this transformed model under the assumption that g_t is "nearly" uncorrelated data.

Table 8.2 shows simulation results analogous to those in Table 8.1. That is, 1,000 replications were generated from each "ϕ_1 and realization length" combination, and the value in the table is the percentage of realizations out of 1,000 for which the CO test found a significant slope while testing at the $\alpha = .05$ level. Note again that for each "ϕ_1 and realization length" combination, the null hypothesis is true. That is, if the CO test appropriately corrected for the correlation in the residuals, the tabled values should each be close to 5%.

Table 8.2 results provide some "good news" and some "bad news". First the good news. The values in Table 8.2 are much closer to 5% than are those in Table 8.1. However, the bad news is that, although the CO method clearly made an adjustment for the correlation structure, the adjustment was not sufficient to produce a valid 5% level test. The only value in the table that is within two SE's of 5% is 7.5% for

"$n = 500 / \phi_1 = .8$". Most of the other values are unacceptably high. For example, given a "stationary" realization of length $n = 100$ with $\phi_1 = .95$, there is a 32.7% chance that the CO test will (incorrectly) "detect" a significant trend. The implication is that if the CO test detects a significant linear trend, there is still a lack of "confidence" that the trend is not just a random trend in a realization from a stationary model (and should not be predicted to continue).

TABLE 8.2 Percentage of times out of 1,000 replications that the CO test found a significant slope for the given AR(1) models and tabled realization lengths. The tests were all run at the $\alpha = .05$ level

		.8	.9	.95	.99
	100	15.0%	21.4%	32.7%	48.0%
n	200	10.8%	12.9%	20.0%	42.3%
	500	7.5%	8.8%	11.4%	29.5%

Note tbat the SE of a tabled value is at most 158%.

Example 8.1 (revisited)
The ***tswge*** command **`co.wge`** can be used to apply the CO test to test for linear trend in realizations **`x1`** and **`x2`** in Example 8.1. Recall that **`x1`** is a realization from Model (a) (with trend) and **`x2`** is from Model (b) (without trend). Applying the commands

```
co.wge(x1)
co.wge(x2)
```

we obtain $p < .001$ for **`x1`** from Model (a). That is, CO correctly identifies the underlying trend. For realization **`x2`** from Model (b), the p-value for the CO test is .016, implying that CO detects a trend (but the model has no trend). Again, note that in Table 8.2 we see that incorrectly detecting a trend for the $n = 100$ and $\phi_1 = .95$ case tends to occur about 32.7% of the time.

8.1.1.4 Bootstrap-Based Test for Trend

Although the Cochrane-Orcutt method is widely used to test for trend in time series data, the results in Table 8.2 clearly show it can have excessively high "observed significance levels". Again, the problem with observed significance levels is well known (see Woodward and Gray (1993), Vogelsang (1998), and Sun and Pantula (1999), to name a few). Other tests, discussed in Woodward and Gray (1993) are available, but they all have a similar problem. Woodward, Bottone, and Gray (1997) addressed the problem of testing for trend using a bootstrap-based approach. This is denoted as the WBG test. The WBG testing approach, implemented in ***tswge*** function **`wbg.boot.wge`**, can be summarized as follows:

- The t_{CO} statistic is calculated as described previously on the original (observed) dataset.
- An autoregressive model is fit to the original data. Note that the data may or may not actually contain a deterministic trend. Fitting an AR model implicitly assumes that any trending behavior in the series is due to the correlation structure alone. That is, the assumption is being made that the null hypothesis ($H_0 : b_1 = 0$) is true.
- B realizations are generated from the autoregressive model fit to the original data. A parametric bootstrap procedure is used which assumes normal errors and generates the a_ts in the generated realizations as random normal deviates.
  - If normality of the errors is questionable, then a nonparametric bootstrap could be used.
- For the bth realization, $b = 1, \ldots, B$, we calculate $t_{CO}^{(b)}$ using the CO procedure. The procedure is as follows:
  - Find the least squares regression line
  - Compute the residuals, $\hat{Z}_t$, from the regression line

  - Fit the residuals with an AR(p) model with pth order operator $\hat{\phi}(B)$ (using AIC-type measures based on Burg estimates to identify p)
  - Transform the data to obtain $\hat{Y}_t = \hat{\phi}(B)X_t$
  - Calculate $t_{CO}^{(b)}$ as in (8.5) for each of the B bootstrap realizations, $b = 1,\ldots,B$.
- Recalling that t_{CO} is the test statistic calculated on the original dataset, then for a two-sided test, $H_0 : b_1 = 0$ is rejected at significance level α if $|t_{CO}| > t^*_{1-\alpha/2}$, where t^*_β is the βth empirical quantile of the collection $t_{CO}^{(b)}$, $b = 1,\ldots,B$ with standard adjustments made for one-sided tests.

Key Point: We use ***Burg estimates*** in the WBG method.

1. For each of the B bootstrap realizations, AIC-type methods are used for identifying p. Using the Burg estimates speeds up the process which will involve using an AIC-type procedure on the original data and all B realizations (where B is typically 199, 399, …).
2. The Burg estimates always find a stationary solution. ML estimates may sometimes fail to converge or may find an explosively nonstationary solution.
3. The computational speed with which Burg estimates can be computed makes them ideal for the WBG bootstrap approach because of the hundreds of realizations that are analyzed.

The simulation scenarios in Tables 8.1 and Table 8.2 were repeated using the WBG method. Table 8.3 shows the results of these simulations. In the table it can be seen that all observed significance levels are within two standard errors of 5% in all except two cases: for "$n = 100$ / $\phi_1 = .99$" and for "$n = 200$ / $\phi_1 = .8$". Note that while the observed significance levels using the WBG bootstrap were about 11% and certainly larger than the nominal 5% level, the corresponding observed significance levels in Table 8.2 using the CO test were about 45%.

QR 8.3 Bootstrap Test for Trend

TABLE 8.3 Percentage of times out of 1,000 replications and using $B = 199$ that the WBG test found a significant slope for the given AR(1) models and tabled realization lengths. The tests were all run at the $\alpha = .05$ level

		.8	.9	.95	.99
	100	4.7%	5.2%	7.4%	11.9%
n	200	4.2%	5.2%	5.5%	10.2%
	500	4.6%	5.3%	5.5%	8.0%

Note that the SE of a tabled value is at most 1.58%.

Example 8.1 (revisited again) The *tswge* command **`wbg.boot.wge`** can be used to apply the WBG bootstrap test for trend in realizations **`x1`** and **`x2`** in Example 8.1. We use the commands

```
wbg.boot.wge(x1)
wbg.boot.wge(x2)
```

That is, the WBG test correctly identifies the underlying trend. For realization, **x2**, from Model (b), $p = .120$, and the WBG test correctly fails to reject the null hypothesis of no trend. Recall that both the SLR test and the CO tests detected trend for Models (a) and (b) although Model (b) did not have a trend component. From Table 8.3 we see that we only reject the null hypothesis under this scenario ($n = 100$ and $\phi_1 = .95$) about 7.4% of the time (as compared to 32.7% for CO) .

It should be noted that, because of the fact that the bootstrap randomly generates 399 realizations from the fitted AR model, the results of subsequent **wbg.boot.wge** commands will differ somewhat. For example, we issued the above commands four more times. The *p*-values for the four issuances of each command are as follows:

x1($b_1 \neq 0$)	**x2**($b_1 = 0$)
.013	.148
.020	.113
.030	.143
.018	.092

In each case the results are consistent with those given above. In order to be able to replicate a particular bootstrap result you can set a value for seed **sn**. For example, issuing the commands

```
wbg.boot.wge(x1,sn=234)
wbg.boot.wge(x2,sn=183)
```

you will always obtain *p*-values .020 and .113, respectively.

(1) Don't Miss This Point about Power

Simulation results (not shown) based on generating 1,000 realizations from line+noise Model (a) show that the SLR, CO, and WBG test (correctly) found a significant trend in 99.8%, 91%, and 50.7% of the realizations, respectively.[4] In these simulations the SLR and CO tests have impressively high powers, but they come at a cost. Consider again Tables 8.1 and 8.2 for the case $b_1 = 0, n = 100$ and $\phi_1 = .9$, (for which there is no deterministic linear term). In this setting the SLR and CO tests (incorrectly) detected a significant deterministic linear trend 66.1% and 21.4% of the time, respectively. On the other hand, Table 8.3 shows that the WBG test (incorrectly) rejected $H_0 : b_1 = 0$, 5.2% of the time, which is consistent with what would be expected for an $\alpha = .05$ level test. That is, if $H_0 : b_1 = 0$ is rejected, the analyst can be "confident" that a deterministic linear trend component does exist. However, the SLR and CO tests do not provide similar protection. That is, they "cry wolf" (reject H_0) too many times when there is no wolf (H_0 is true).

Key Points:

1. The WBG bootstrap-based test controls the observed significance level much better than the SLR and CO tests in the presence of correlated residuals.
2. The SLR and CO tests have higher power than WBG, but the higher powers are artificial because they are not controlling the observed significance level.

4 The 99.8%, 91%, and 50.7% quoted here are measures of the power of the tests under the particular alternative $b_1 = .1$, n = 100, 2 and $\phi_1 = .9$.

8.1.1.5 Other Methods for Testing for Trend in Time Series Data

Many other tests are available for testing for trend in the presence of correlated residuals. Among these, Beach and MacKinnon (1978) developed an ML-based method, and Cowpertwaite and Metcalfe (2009) recommend a generalized least squares (GLS) approach in the presence of AR(1) residuals. Consider the case in which the true model is $(1 - 0.9B)X_t = a_t$, and suppose tests are run at the 5% level of significance. Table 8.2 shows that the CO method incorrectly detects a trend about 21% of the time. Woodward et al. (1997) showed that the ML approach of Beach and MacKinnon (1978) detects a trend about 17% of the time, and further simulations, not shown here, indicate that the GLS approach detects a trend about 12% of the time. The simulations discussed in Woodward, et al. (1997) show that when X_t is a stationary AR(1) with $\phi_1 = .9$ or higher, realizations of length $n = 1000$ or more may be required in order for the CO or ML tests to have observed significance levels close to the nominal levels. Further discussion of these inflated significance level issues can be found in Park and Mitchell (1980), Woodward and Gray (1993), Woodward et al. (1997), and Woodward (2013).

Key Point: We recommend using the WBG approach because it controls the observed significance levels at close to the nominal level and is easily implemented using ***tswge*** function **`wbg.boot.wge`**.

8.1.2 Fitting Line+Noise Models to Data

Section 8.1.1 discussed the issue of determining whether a deterministic linear signal should be included in the model for a given time series realization. After this decision has been made, a final model will need to be derived for the data. Consider the model

$$X_t = b_0 + b_1 t + Z_t,$$

previously given in (8.1) where Z_t is a zero-mean AR(p) model. When analyzing a set of time series data for which we believe the appropriate model may include a deterministic linear component, we recommend using the following procedure, which follows the strategy in Section 8.1.1.

Procedure: First check for the existence of a deterministic linear trend. That is, test $H_0 : b_1 = 0$ (we recommend using the WBG method)

(a) ***If*** $\boldsymbol{H_0 : b_1 = 0}$ ***is rejected:*** Then fit a line+noise model to the data using the following steps:

(i) Fit a least squares line to the data $\hat{s}_t = \hat{b}_0 + \hat{b}_1 t$ using the command

```
xa=slr.wge(x1)
# among the output are xa$b0hat, xa$b1hat # SLR estimates b0hat, b1hat
```

(ii) Find the residuals $\hat{Z}_t = X_t - \hat{b}_0 - \hat{b}_1 t$. The output from **`slr.wge`** contains these residuals in **`xa$res`**

(iii) Fit an AR model to the residuals using Burg estimates

```
xa.aic=aic.burg.wge(xa$res,p=1:5)
```

(iv) The final fitted model is $X_t = \hat{b}_0 + \hat{b}_1 t + \hat{Z}_t$, where $\hat{Z}_t$ is the maximum likelihood AR fit to the residuals and $\hat{\sigma}_a^2$ is the white noise variance estimate.

(b) ***If*** $\boldsymbol{H_0 : b_1 = 0}$ ***is not rejected:*** Then the evidence suggests that the fitted model does not contain a linear signal, and we fit an ARMA(p,q) or ARIMA(p,d,q) model to X_t as discussed in Chapters 6 and 7.

Key Points:

1. The SLR, CO, and WBG tests are designed to test whether $b_1 = 0$.
2. If it is concluded that $b_1 \neq 0$, then regardless of the testing method used to come to that decision, the final model is obtained by
   - Using **slr.wge** to fit a regression line to the data
   - Finding the residuals, $\hat{Z}_t$, from the regression line and fitting an AR(p) model to them.
   - Although we have used Burg estimates up until this point, it is common practice to use ML estimates for the final model.

Example 8.2 Fitting models to datasets x1 and x2 in Figure 8.2(a) and (b), respectively.

(a) Realization **x1** from $X_{1t} = 10 + .1t + Z_{1t}$ where Z_{1t} is the AR(1) process $(1 - .9B)Z_t = a_t$. Recall that the realization, **x1**, is generated by the command

```
x1=gen.sigplusnoise.wge(n=100,b0=10,b1=.1,phi=.9,sn=65987)
```

Note: We know that this realization came from a model with a linear signal, but we analyze the data as if we do not know this.

Procedure: *First check for the existence of a deterministic linear signal.* That is, test $H_0 : b_1 = 0$ (we recommend using the WBG method) using the command

```
wbg.boot.wge(x1)
# the resulting p-value is .030
# repeated issuance of the command produces similar p-values #try it!
```

Conclusion: *We reject* $H_0 : b_1 = 0$ (at the $\alpha = .05$ level) and decide to fit a model containing a linear trend. Fit the model using the following steps:

(i) Fit a least squares line to the data $\hat{s}_t = \hat{b}_0 + \hat{b}_1 t$ using the command

```
xa=slr.wge(x1)
# among the output are xa$b0hat, xa$b1hat # SLR estimates b0hat, b1hat
# xa$b0hat=12.028 xa$b1hat=.059
```

(ii) Find the residuals $\hat{Z}_t = X_t - \hat{b}_0 - \hat{b}_1 t$: For the current dataset $\hat{Z}_t = X_t - 12.028 - .059t$.

The output from **slr.wge** contains these residuals in **xa$res.**

(iii) Fit an AR model to the residuals (using ML estimates for the final model)

```
xa.aic=aic.wge(xa$res,p=0:5,q=0:0)
```

AIC selects the AR(3) model below for the residuals (even though the original model was an AR(1))

$$(1 - .98B + .08B^2 + .14B^3)\hat{Z}_t = a_t$$

where $\hat{\sigma}_a^2 = .971$.

(iv) The final fitted model is

$$X_t = 12.028 + .059t + \hat{Z}_t,$$

where $\left(1 - .98B + .08B^2 + .14B^3\right)\hat{Z}_t = a_t$ with $\hat{\sigma}_a^2 = .971$.

Note: The above sequence of commands was used to illustrate the steps involved in fitting line+noise models to data. After deciding to include a line in the final model, the ***tswge*** command

```
fit.sig=fore.sigplusnoise.wge(x1,linear=TRUE)
```

can be used to fit a line+noise model to the data using AIC to select p, and MLE to estimate the parameters of the pth order AR model fit to $\hat{Z}_t$. As the name of the command suggests, **`fore.sigplusnoise.wge`** also computes forecasts which will be discussed in Section 8.1.3. For now, among the output from the above command are

```
fit.sig$b0hat=12.028
fit.sig$b1hat=.059
fit.sig$phi.z= 0.98 -0.08 -0.14
fit.sig$wnv=.971
```

which are consistent with the final model in (iv) above.

(b) Realization **`x2`** from $X_{2t} = 10 + Z_{2t}$ where Z_{2t} is the AR(1) process $\left(1 - .95B\right)Z_t = a_t$. Recall that the realization, **`x2`**, is generated using the commands

```
z2=gen.arma.wge(n=100,phi=.95,sn=6587)
x2=10+z2
```

Note: We know that this realization came from a stationary model without a linear signal, but the analysis will proceed as if this information is unknown.

Procedure: *Check for the existence of a deterministic linear signal.* That is, test $H_0 : b_1 = 0$ (again, we recommend using the WBG method). We issue the commands

```
wbg.boot.wge(x2)
# the resulting p-value is .12
# repeated issuance of the command produces similar p-values
```

Conclusion: We do not reject $H_0 : b_1 = 0$ (at $\alpha = .05$) and decide to model the data using techniques for modeling ARMA and ARIMA models. The data, sample autocorrelations, and Parzen spectral density estimate obtained using **`plotts.sample.wge(x2)`** are shown in Figures 8.3(a)–(c).

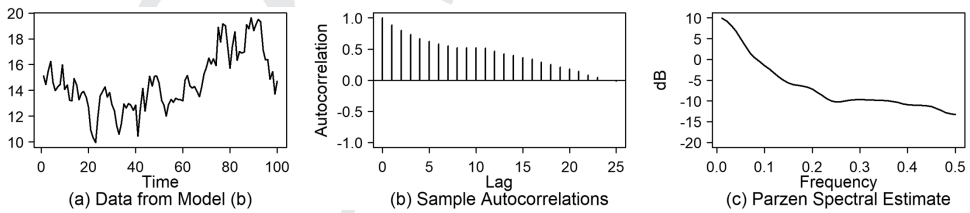

FIGURE 8.3 (a) Data from Model (b), (b) sample autocorrelations of the data in (a), and (c) Parzen spectral density estimate of the data in (a).

The slowly damping sample autocorrelations are suggestive of a possible unit root, but because this dataset is simulated data and we have no physical reason to include a unit root, we simply fit an ARMA(p,q) model to the data. Using the command

```
aic.wge(x2,p=0:8,q=0:2)
```

AIC selects the AR(1) model $(1-.878B)(X_t-14.656)=a_t$ where $\hat{\sigma}_a^2=1.073$.

QR 8.4 Line+Noise Model Fitting

A final note: In Section 8.1.3 we will discuss forecasting with line+noise models. We again note that these forecasts "look, smell, and act" like *extrapolation* in a simple linear regression setting. Be careful when projecting a line (or other deterministic function) beyond the range of the independent variables (in this case, "time").

8.1.3 Forecasting Using Line+Noise Models

In this section we discuss forecasting future values using a fitted line+noise model. Assume that, using the techniques of Section 8.1.2, a time series, X_t, has been modeled using a line+noise model

$$X_t=\hat{b}_0+\hat{b}_1t+\hat{Z}_t, \tag{8.6}$$

where $\hat{Z}_t$ is modeled as a zero-mean AR(p) model $\hat{\phi}(B)\hat{Z}_t=\hat{a}_t$. Suppose data are observed for times $t=1,2,\ldots,t_0$ and forecasts $\hat{X}_{t_0}(\ell)$ are desired for $\ell=1,\ldots,k$. Forecasts are obtained as follows:

(1) *Forecast the residual series, $\hat{Z}_t$:*
Because $\hat{Z}_t=X_t-\hat{b}_0-\hat{b}_1t$ is modeled as a stationary AR(p) process, forecasts $\hat{Z}_{t_0}(\ell)$, $\ell=1,\ldots,k$ can be found using **`fore.arma.wge`** based on its stationary model fit using techniques of Section 6.2.

(2) *Convert the forecasts in (1) to forecasts, $\hat{X}_{t_0}(\ell)$, for $X_{t_0+\ell}$:*
The forecasts, $\hat{Z}_{t_0}(\ell)$, can be converted to forecasts, $\hat{X}_{t_0}(\ell)$, for $X_{t_0}+\ell$ using the formula

$$\hat{X}_{t_0}(\ell)=\hat{b}_0+\hat{b}_1t+\hat{Z}_{t_0}(\ell). \tag{8.7}$$

Notes:

(a) If a line+noise model has been fit to a set of data, then the tacit assumption is that this fitted line can be appropriately "extrapolated" into the future.
(b) Because $\hat{Z}_t$ is modeled as a stationary process with zero mean, the eventual forecasts, $\hat{Z}_{t_0}(\ell)$, will tend toward zero. Consequently, the eventual forecasts, $\hat{X}_{t_0}(\ell)$, will tend toward the fitted line.

QR 8.5 Line+Noise Forecasting

Example 8.3 Forecasting realization `x1` using the model in (8.7).
Assume that a decision has been made to fit a line+noise model to the **`x1`** data and we want to forecast 25 steps ahead beyond the end of the realization.The estimates of the fitted line from Example 8.2 Model (a) are found by issuing the command

```
xa=slr.wge(x1)
```

From the output we obtain

$\hat{b}_0$ is 12.028 (**`xa$b0hat`**)
$\hat{b}_1$ is .059 (**`xa$b1hat`**)
$\hat{z}_t$, the residual series (**`xa$res`**)

AIC selected an AR(3) model for the residual series using

```
xa.aic=aic.wge(xa$res,p=0:5)
```

The output includes:

AIC selection: 3 (**`xa.aic$aic`**)
$\hat{\phi}_1, \hat{\phi}_2$, and $\hat{\phi}_3$: 0.98, –0.08, and –0.14 (**`xa.aic$phi`**)
$\hat{\sigma}_a^2$: .971 (**`xa.aic$vara`**)

(1) *Forecast the residual series:* Forecast the residual series using the command

```
f.z1=fore.arma.wge(xa$res,xa.aic$phi,n.ahead=25)
```

The forecasts for the residual series along with 95% limit lines are shown in Figure 8.4(a).

(2) *Convert the forecasts in (1) to forecasts,* $\hat{X}_{t_0}(\ell)$:

The forecasts for 25 steps ahead of the line+noise dataset , **`x1`**, can be calculated using the commands below.

```
tf=101:125
f.x1=f.z1$f+xa$b0hat+xa$b1hat*tf
```

The vector **`f.x1`** contains the forecasts $\hat{X}_{t_0}(\ell)$, ℓ =1, …, 25.

Note:

(1) The commands above were provided to show the forecast procedure step-by-step.
(2) In practice the forecasts for the line+noise fit can be easily obtained using the command

```
fore.sigplusnoise.wge(x1,n.ahead=25,limits=TRUE)
```

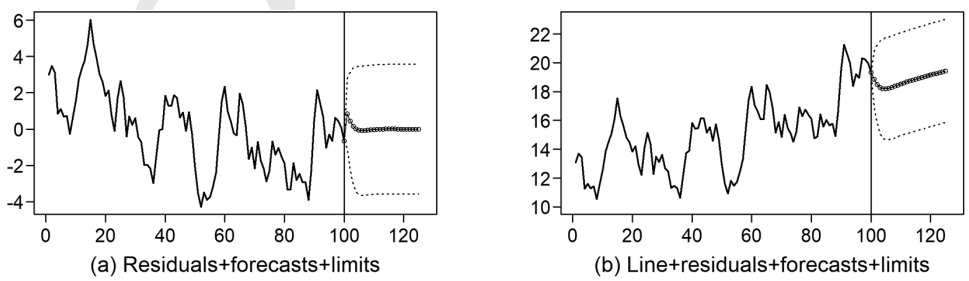

FIGURE 8.4 (a) Forecasts up to 25 steps ahead for residual series $\hat{Z}_t$ along with 95% prediction limits; (b) corresponding forecasts and 95% prediction limits for the line+noise signal in `x1`.

Key Points:

1. As mentioned previously, the fact that we *have decided* to include a deterministic linear component in our model tacitly implies that the eventual forecasts will follow the fitted line.
   - This is extrapolation!
2. Before including a line in a model to be used for forecasting, you should be certain that "it belongs there" and that you have good reason to believe that current forces impacting the data will continue.

8.2 COSINE SIGNAL+NOISE MODELS

In this section, we address the case in which the signal, s_t, is a function of the form $s_t = C\cos(2\pi ft + U)$. The signal+noise model in this case is

$$X_t = C_0 + C_1\cos(2\pi ft + U) + Z_t \tag{8.8}$$

where Z_t is a zero-mean stationary process which we will model using an autoregressive model.[5]

Figure 8.5(a) shows a realization from the model

$$X_t = 5\cos(2\pi(.1)t + \pi/3) + Z_t, \tag{8.9}$$

where $(1-.75B)Z_t = a_t$, and $\sigma_a^2 = 1.5$. Note that $C_0 = 0$. The realization was created using the ***tswge*** commands

```
z=gen.arma.wge(n=100,phi=.75,sn=921,vara=1.5)
time=1:100
x.cos=5*cos(2*pi*.1*time+pi/3)
x.cos=x.cos+z
```

QR 8.6 Cosine+ Noise Model

Figure 8.5(a) shows the cosine-based cyclic behavior and shows that the amplitudes of the peaks and troughs vary from cycle to cycle. Figure 8.5(b) shows the underlying cosine signal while Figure 8.5(c) is a plot of the noise realization, z_t. The plot in Figure 8.5(a) is simply the sum of the signals plotted in Figures 8.5(b) and (c).

5 Woodward et al. (2017) refer to the model in (8.8) as a harmonic component model. Also, because $\sin(\theta) = \cos\left(\frac{\pi}{2} - \theta\right)$ the "cosine signal+noise models" could just as easily be called "sine signal+noise" models or "sinusoidal" signal+noise models.

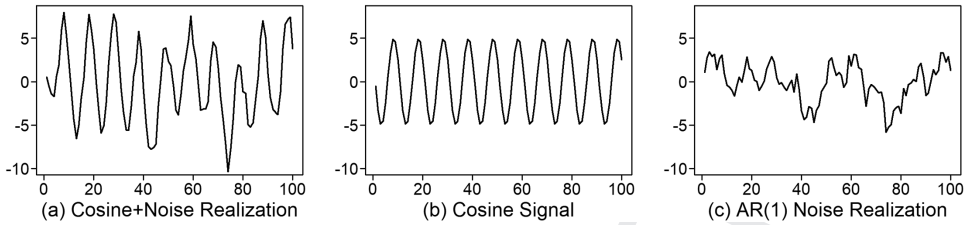

FIGURE 8.5 (a) Realization from cosine signal+noise model in (8.9), (b) plot of the deterministic curve $5\cos\left(2\pi(.1)t+\pi/3\right)$, and (c) realization from AR(1) model $(1-.75B)Z_t = a_t$ with $\sigma_a^2 = 1.5$.

Cosine signal+noise models could also contain cosines at several different frequencies. We restrict our analyses to a cosine signal at a single frequency.

Key Point:

1. As was the case with the line+noise models, a major step in the modeling process is deciding whether to include a (deterministic) cosine signal in the model
2. We will address the important "decision" issue in Section 8.2.3.

8.2.1 Fitting a Cosine Signal+Noise Model to Data

Before fitting a model of the form (8.8) to a set of data, it is common to re-write (8.8) in the form

$$X_t = C_0 + A\cos\left(2\pi ft\right) + B\sin\left(2\pi ft\right) + Z_t, \tag{8.10}$$

where $A = \cos\left(U\right)$ and $B = -\sin\left(U\right)$.[6] If the *decision* has been made to fit a model of the form (8.10) to a dataset $\mathbf{x}$, then we recommend the following procedure:

(i) *Estimate the parameters of* $C_0 + A\cos\left(2\pi ft\right) + B\sin\left(2\pi ft\right)$:

That is, C_0, A, B, and f need to be estimated. The presence of the parameter, f, causes this to be a nonlinear estimation problem.

(a) *Estimate* f : Our experience is that if the sinusoidal term is sufficiently strong in the signal, then estimating f can be accomplished by examining the data, finding the location of the dominant peak in the spectral density, and/or using the factor table and domain knowledge.

(b) *Estimate* C_0, A *and* B : Notice that if f is given (or an estimated value is selected for use) then (8.10) is no longer nonlinear, and can be viewed as a multiple linear regression,

$$X_t = C_0 + AH_1\left(t\right) + BH_2\left(t\right) + Z_t. \tag{8.11}$$

6 Use the trigonometric identity $\cos\left(F+G\right) = \cos\left(F\right)\cos\left(G\right) - \sin\left(F\right)\sin\left(G\right)$. In our case, $F = 2\pi ft$ and $G = U$ in (8.8). Consequently, in (8.10), A = cos(*U*) and $B = -\sin\left(U\right)$. Given the above identities, then $C_1\cos\left(2\pi ft+U\right)$ in (8.8) can be written as $A\cos\left(2\pi ft\right) + B\sin\left(2\pi ft\right)$ where $A = C_1\cos\left(U\right)$ and $B = -C_1\sin\left(U\right)$.

where $H_1(t) = \cos(2\pi ft)$ and $H_2(t) = \sin(2\pi ft)$. Letting $\hat{f}$ denote the estimate for f, the estimated independent variables are $\hat{H}_1(t) = \cos(2\pi \hat{f}t)$ and $\hat{H}_2(t) = \cos(2\pi \hat{f}t)$. Multiple linear regression techniques are then applied to find estimates $\hat{C}_0$, $\hat{A}$ and $\hat{B}$

(ii) *Calculate the residuals,* $\hat{Z}_t$: The residuals are $\hat{Z}_t = X_t - \hat{C}_0 - \hat{A}\cos(2\pi \hat{f}t) - \hat{B}\sin(2\pi \hat{f}t)$.

(iii) *Fit an AR model to the residuals:* We will use AIC to identify the AR order and find the ML estimates of the parameters of the AR model.

(iv) *Specify the final model:* The final fitted model is

$$X_t = \hat{C}_0 + \hat{A}\cos(2\pi \hat{f}t) + \hat{B}\sin(2\pi \hat{f}t) + \hat{Z}_t. \tag{8.12}$$

The final model should specify the order and parameter estimates of the AR model fit to the data, and should include the estimated white noise variance.

Example 8.4 Fitting a Cosine Signal+Noise Model to the Data in Figure 8.5(a)

We use the steps listed above to fit a cosine signal+noise model to the data in Figure 8.5(a) which were generated from the model in (8.9). The code for generating the data is given following equation (8.9) and the data to be analyzed are in **x.** As discussed above, before estimating the parameters we express the model in (8.9) in the form given in (8.10). Using the footnote, $A = 5\cos(\pi/3) = 5(.5) = 2.5$ and $B = -5\sin\left(\frac{\pi}{3}\right) = -5(.866) = -4.33$. In the form of (8.10), model (8.9) becomes

$$X_t = 2.5\cos(2\pi ft) - 4.33\sin(2\pi ft) + Z_t, \tag{8.13}$$

where $(1 - .75B)Z_t = a_t$ with $\sigma_a^2 = 1.5$.

Figure 8.6(a) is a replot of the signal+noise realization in Figure 8.5(a) along with the sample autocorrelations and Parzen spectral density estimate.

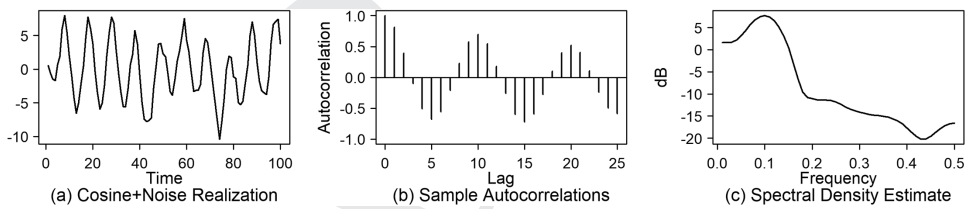

FIGURE 8.6 (a) Cosine signal+noise realization in Figure 8.5(a), (b) sample autocorrelations of the data in (a), and (c) Parzen spectral density estimate of the data in (a).

(i) *Estimate the parameters of* $C_0 + A\cos(2\pi ft) + B\sin(2\pi ft)$:

The parameter estimation procedure involves the following two steps:

(a) *Estimate* f: Examining the data it can be seen that the data pass through about 10 cycles in the realization of length $n = 100$, suggesting a frequency of $f = .1$. The spectral density provides supporting information in that it has a peak at about $f = .1$. The sample autocorrelations, which do not always provide informative frequency information, have a damped sinusoidal behavior with a period of about 10, again suggesting a frequency of about $f = .1$.

Table 8.4 shows the factor table for an AR(10) fit to the data in Figure 8.5(a) and 8.6(a) using Burg estimates. Overfit models with $p = 8$, 12, and 14 all had a primary system frequency of $f = .099$. Consequently, we choose to use the estimate, $\hat{f} = .1$. In the R code below we denote **f.est=.1.**

TABLE 8.4 Factor Table for AR(10) Overfit to Cosine Signal+Noise Data

AR-FACTOR	*ROOTS*	$\lvert r\rvert^{-1}$	f_0
$1-1.61B+.97B^2$	$.83\pm.59i$	.99	.099
$1-.92B$	1.08	.92	0
$1+1.54B+.66B^2$	$-1.16\pm.39i$	.81	.448
$1+.76B+.59B^2$	$-.64\pm1.13i$	.77	.332
$1-.26B+.53B^2$	$.25\pm1.35i$	.73	.221
$1-.33B$	3.06	.33	0

(b) *Estimate* $C_0, A,$ *and* B:

$\hat{H}_1(t) = \cos\left(2\pi\hat{f}t\right)$ and $\hat{H}_2(t) = \cos\left(2\pi\hat{f}t\right)$ are computed using the following code.

(Recall that the data are in **x.cos**.)

```
n=length(x.cos)
h1=rep(0,n)
h2=rep(0,n)
f.est=.1
for(t in 1:n) h1[t]=cos(2*pi*f.est*t)
for(t in 1:n) h2[t]=sin(2*pi*f.est*t)
```

The parameters $C_0, A,$ and B are estimated using the R command

```
h.est=lm(x.cos~h1+h2)
```

The estimates for $C_0, A,$ and B are found in **h.est$coefficients[k]**, *k*=**1,2**, and **3**, respectively. Specifically, we obtain the following

```
h.est$coefficients[1] # -.0339
h.est$coefficients[2] # 2.7590
h.est$coefficients[3] # -4.6649
```

(ii) *Calculate the residuals,* $\hat{Z}_t$: The residuals, $\hat{Z}_t = X_t - \hat{C}_0 - \hat{A}\cos\left(2\pi\hat{f}t\right) - \hat{B}\sin\left(2\pi\hat{f}t\right)$, are calculated using the commands:

```
z.cos=rep(0,n)
C0=h.est$coefficients[1]
A=h.est$coefficients[2]
B=h.est$coefficients[3]
for(t in 1:n) z.cos[t]=x.cos[t]-C0-A*h1[t]-B*h2[t]
```

(iii) *Fit an AR model to the residuals*: The command

```
z.cos.aic=aic.wge(z.cos,p=1:5)
```

selects an AR(1) model. The following command estimates the parameters of the AR(1) model, computes the residuals, $\hat{a}_t$, and returns the parameter estimates for the selected model.

```
ar.cos=est.ar.wge(z.cos,p=z.cos.aic$p)
```

Among the output are the following:

```
ar.cos$phi [1] 0. 8315
ar.cos$avar [1] 1.365834
```

The model fit to the residuals is

$$(1-.83B)\hat{Z}_t = a_t, \tag{8.14}$$

where $\hat{\sigma}_a^2 = 1.37$ which is close to the true value, $\sigma_a^2 = 1.5$.[7] The residuals, $\hat{a}_t$, are contained in **`ar.cos$res`**. These are not plotted here, but they appear to be white noise.

(iv) *Specify the final model:* The final fitted model is

$$X_t = -.03 + 2.76\cos(2\pi(.1)t) - 4.66\sin(2\pi(.1)t) + \hat{Z}_t, \tag{8.15}$$

where $\hat{Z}_t$ satisfies the AR(1) model in (8.14). Examination of the fitted model in (8.15) shows that it is quite similar to the actual model in (8.13).

8.2.2 Forecasting Using Cosine Signal+Noise Models

Consider the fitted cosine signal+noise model

$$X_t = \hat{C}_0 + \hat{A}\cos(2\pi\hat{f}t) + \hat{B}\sin(2\pi\hat{f}t) + \hat{Z}_t. \tag{8.16}$$

where $\hat{Z}_t$ is modeled as a zero-mean AR(p) model, $\hat{\phi}(B)\hat{Z}_t = a_t$, using the techniques discussed in Chapter 6. In this section we develop forecasts based on the fitted model in (8.16). The forecast procedure for cosine signal+noise models is similar to that used for the line+noise models in Section 8.1.4. There are basically two steps:

(1) ***Forecast the residual series, $\hat{Z}_t$:***

Because $\hat{Z}_t = X_t - \hat{C}_0 - \hat{A}\cos(2\pi\hat{f}t) - \hat{B}\sin(2\pi\hat{f}t)$ is modeled as a stationary AR(p) process, forecasts $\hat{Z}_{t_0}(\ell)$, $\ell = 1,\ldots,k$, can found using **`fore.arma.wge`** based on the stationary model fit to $\hat{Z}_t$ using methods discussed in Section 6.2.

(2) ***Convert the forecasts in (1) to forecasts, $\hat{X}_{t_0}(\ell)$, for $X_{t_0+\ell}$:***

The forecasts, $\hat{Z}_{t_0}(\ell)$, can be converted to forecasts $\hat{X}_{t_0}(\ell)$ for $X_{t_0+\ell}$ using the formula

$$\hat{X}_{t_0}(\ell) = \hat{C}_0 + \hat{A}\cos(2\pi\hat{f}t) + \hat{B}\sin(2\pi\hat{f}t) + \hat{Z}_{t_0}(\ell). \tag{8.17}$$

7 The sample mean of the `ar.cos$res` data was zero to over 4 decimal places which is also very close to the true value.

Notes:

(a) If a cosine signal+noise model has been fitted to a set of data, then the tacit assumption is that this fitted signal can be appropriately "extrapolated" into the future.

(b) Because $\hat{Z}_t$ is modeled as a stationary process with zero mean, the eventual forecasts, $\hat{Z}_{t_0}(\ell)$, will tend toward zero. Consequently, the eventual forecasts, $\hat{X}_{t_0}(\ell)$, will tend toward the fitted cosine curve,

$$\hat{C}_0 + \hat{A}\cos\left(2\pi\hat{f}t\right) + \hat{B}\sin\left(2\pi\hat{f}t\right)$$

as ℓ gets large.

Key Points:

1. The *tacit assumption* that this cosine signal can be appropriately "extrapolated" into the future is not one to be made lightly.
2 The important decision whether to use a cosine signal+noise model will be discussed in Section 8.2.3.

Example 8.5 Forecasting realization `x.cos` (shown in Figure 8.5(a) and 8.6(a)) using the fitted model in (8.15). In this example we will produce two types of forecasts:

(A) Forecasts 50 steps beyondthe end of the series with prediction limits.
(B) Forecasts of the last k values in the dataset for comparison of actual data values.

The forecasts will be made under the assumption that the cosine signal+noise model (8.15) has been fit to the data in **`x.cos`**.

(A) The forecasts are based on the two-step procedure below:

(1) *Forecast the residual series,* $\hat{Z}_t$:

The following commands are used to compute and plot forecasts, $\hat{Z}_{t_0}(\ell), \ell = 1,\ldots,25$ along with 95% forecast limits. These forecasts and 95% prediction limits are plotted in Figure 8.7(a). Recall that the residuals are in the file **`z.cos$res`** and that the parameters of the AR(1) model fit to the data in **`z.cos$res`** are contained in **`z.cos$phi.`** The following command forecasts residuals up to 50 steps ahead.

```
f.ar.cos=fore.arma.wge(ar.cos$res,ar.cos$phi,n.ahead=50)
```

Because $\hat{Z}_{100} = -1.02$ is below the sample mean (zero), the forecasts trend upward toward zero. Specifically, $\hat{Z}_{100}(1) = -.085$.

(2) *Convert the forecasts in (1) to forecasts,* $\hat{X}_{t_0}(\ell)$, *for* $X_{t_0+\ell}$: The forecasts up to 50 steps ahead for the cosine signal+noise dataset, **`x.cos`**, can be calculated using the commands below.

```
f.x.cos=rep(0,length(x.cos))
for (t in 1:length(x.cos))
f.x.cos[t]=f.ar.cos$f[t]+C0+A*h1[t]+B*h2[t]
```

Note that the forecasts in Figure 8.7(b) are simply the forecasts *of the residuals* in Figure 8.7(a) plus the deterministic function, $H(t) = -.03 + 2.76\cos(2\pi(.1)t) - 4.66\sin(2\pi(.1)t)$. That is, the forecast $\hat{X}_{100}(1) = H(t) + \hat{Z}_{100}(1)$

$$= -.03 + 2.76\cos(2\pi(.1)t) - 4.66\sin(2\pi(.1)t) + \hat{Z}_{100}(1),$$

so that

$$\hat{X}_{100}(1) = -.03 + 2.76\cos(2\pi(.1)(101)) - 4.66\sin(2\pi(.1)(101)) + \hat{Z}_{t_0}(1) = -1.39.$$

The forecasts tend toward the sinusoidal function, $H(t)$, as t gets large.

8.2.2.1 Using fore.sigplusnoise.wge:

While the above sequence of code illustrates the *procedure* for finding the cosine signal+noise forecasts, the forecasts in Figure 8.7(b) can more be easily obtained using the command

```
fore.sigplusnoise.wge(x.cos,linear=FALSE,freq=.1,n.ahead=50,limits=TRUE)
```

Note:
The command **`fore.sigplusnoise.wge`** is designed to provide forecasts using either line+ noise or cosine signal+noise models. The line+noise model is the default option. See Appendix 8A for more details.

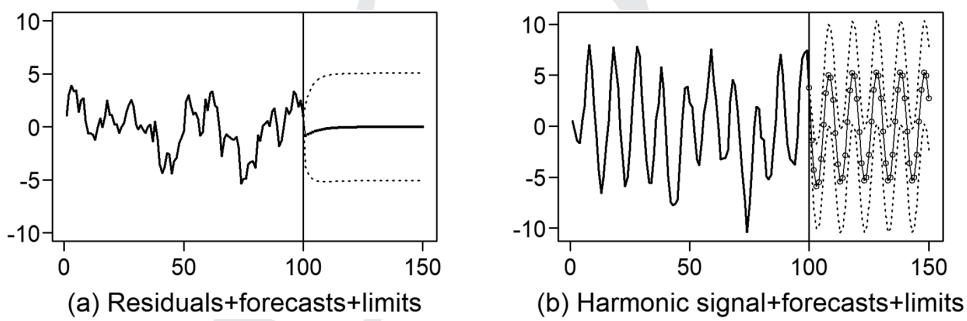

FIGURE 8.7 (a) Forecasts up to 50 steps ahead for residual series `z.cos` along with 95% prediction limits; (b) corresponding forecasts and prediction limits for the signal+noise signal `x.cos`.

(B) The following code uses the fitted model to calculate the "last k" forecasts for $k = 10, 43, 67$, and 84. In this case, we used the entire dataset to obtain model (8.15) and based all forecasts on this model.

```
fore.sigplusnoise.wge(x.cos,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
10,limits=FALSE)
fore.sigplusnoise.wge(x.cos,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
43,limits=FALSE)
fore.sigplusnoise.wge(x.cos,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
67,limits=FALSE)
fore.sigplusnoise.wge(x.cos,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
84,limits=FALSE)
```

Figure 8.8 shows the forecasts (open circles connected by dotted lines) for the last 10, 43, 67, and 84 values. In each case the forecast cycles are "in sync" with the actual cycles, even for the forecasts for 84 steps ahead.

Key Point: The cosine signal+noise model produces realizations that have cyclic behavior with *fixed cycle length* in the terminology of Section 1.2.1.

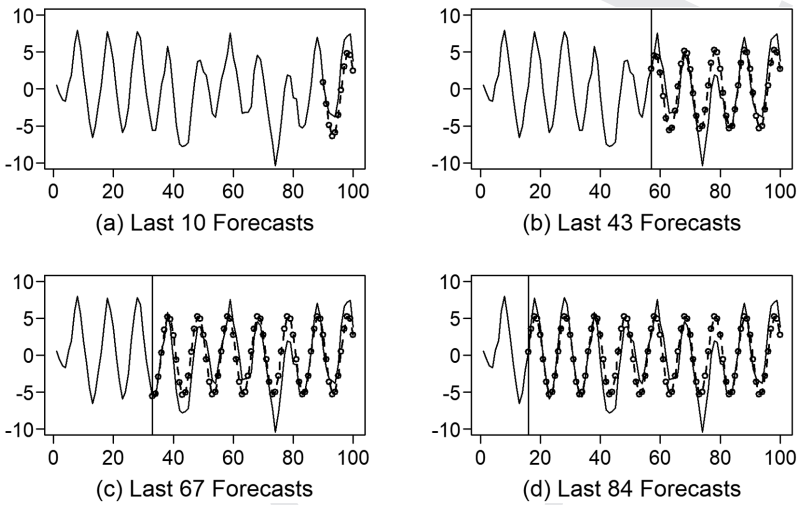

FIGURE 8.8 Forecasts for last k values in x.cos dataset for $k = 10, 43, 67$, and 84, respectively.

8.2.3 Deciding Whether to Fit a Cosine Signal+Noise Model to a Set of Data

In this section we address issues similar to those in Section 8.1.1. In that section we discussed the decision whether to include a line (deterministic linear signal) when modeling a dataset that has trending behavior. It was emphasized that this is a difficult decision that is critically important, especially when using the fitted model to forecast into the future. For example, we showed that an AR(1) model with ϕ_1 close to or equal to +1 (but not greater than +1) often has realizations with random trends that have the appearance of (deterministic) linear trending behavior.[8] Simulation studies in Section 8.1.1 show that standard tests for linear trend have difficulty distinguishing between "random trends" in AR models and deterministic trends in line+noise models. We recommend the use of the WBG bootstrap-based test for distinguishing between the two types of trend. However, we noted that the decision should not be based solely on any testing procedure.

8 Actually, realizations from (stationary) AR (p) and ARMA (p,q) models with a factor close to +1 can also exhibit the random trending behavior observed for the AR(1) models discussed in Section 8.1.1.

In the cosine setting there is a similar uncertainty:

(a) Cosine signal+noise models produce realizations with cyclic-type behavior.
(b) AR(2) models associated with a pair of complex roots fairly close to the unit circle produce realizations with cyclic-type behavior based on the associated system frequency.[9]

See Section 5.1.2.4.
As an illustration of point (b) we have generated a realization from the AR(2) model,

$$\left(1-1.55B+.925B^2\right)X_t = a_t, \tag{8.18}$$

with $\sigma_a^2 = .8$. This realization is generated using the ***tswge*** command

```
x.ar2=gen.arma.wge(n=100,phi=c(1.55,-.925),vara=.8,sn=281)
```

The cosine signal+noise signal realization, **`x.cos`**, is plotted in Figure 8.9(a) along with the AR(2) realization, **`x.ar2`**, in Figure 8.9(b).

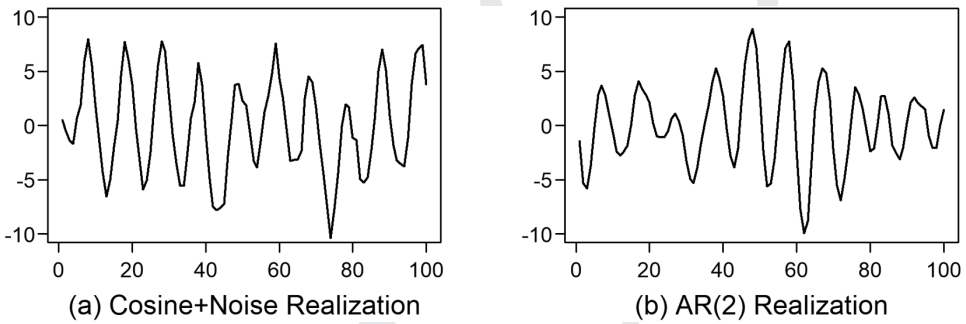

FIGURE 8.9 (a) Cosine signal+noise realization, `x.cos`, and (b) AR(2) realization, `x.ar2`.

The realizations are quite similar. The system frequency associated with (8.18) is

$$f_0 = (1/2\pi)\cos^{-1}\left(\frac{1.55}{2\sqrt{.925}}\right) = .1,$$

which is the frequency associated with the cosine signal+noise model in Equation 8.9. We saw in Section 8.1.4 that the decision to include a deterministic linear trend component implies that forecasts of the future will eventually fall along the fitted line. In a similar manner, the decision to include a cosine signal in a time series model implies that forecasts will eventually follow the fitted cosine curve.

Assume that we do not know the source of the realization in Figure 8.9(b), and that because of its cyclic behavior, we decide to fit a cosine signal+noise model of the form (8.12). We use the ***tswge*** command

```
x.ar2.h=fore.sigplusnoise.wge(x.ar2,linear=FALSE, f=.1)
```

and obtain the following:

```
x.ar2.h$b[1]  0.0042891 (C0)
x.ar2.h$b[2] -0.4445196 (A)
x.ar2.h$b[3] -3.4193501 (B)
```

9 AR (p) and ARMA (p,q) models with an irreducible 2nd-order factor associated with roots "close" to the unit circle will also tend to show cyclic-type behavior.

so that the fitted model is

$$X_t = .00 - .44\cos\left(2\pi(.1)t\right) - 3.42\sin\left(2\pi(.1)t\right) + \hat{Z}_t, \tag{8.19}$$

where AIC selects an AR(3) model for $\hat{Z}_t$. We use the cosine signal+noise model for forecasting the realization **x.ar2**. Figures 8.10(a)–(d) show forecasts from four different forecast origins and are obtained using the commands

```
fore.sigplusnoise.wge(x.ar2,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
10,limits=FALSE)
fore.sigplusnoise.wge(x.ar2,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
43,limits=FALSE)
fore.sigplusnoise.wge(x.ar2,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
67,limits=FALSE)
fore.sigplusnoise.wge(x.ar2,linear=FALSE,freq=.1,lastn=TRUE,n.ahead=
84,limits=FALSE)
```

The forecasts in Figure 8.10 show that although the system frequency of the generating AR(2) model is $f_0 = .1$, forecasts based on a deterministic cosine signal+noise model are quite poor. For example, the forecasts in Figure 8.10(b) tend to be "backward". That is, they forecast "up" when the data are "down" and vice versa. We noted this type of behavior in Example 6.15 when forecasting the sunspot data. Also note that forecasts of the last 43, 67, and 84 points stay "in sync" with the actual data until about the last two cycles.

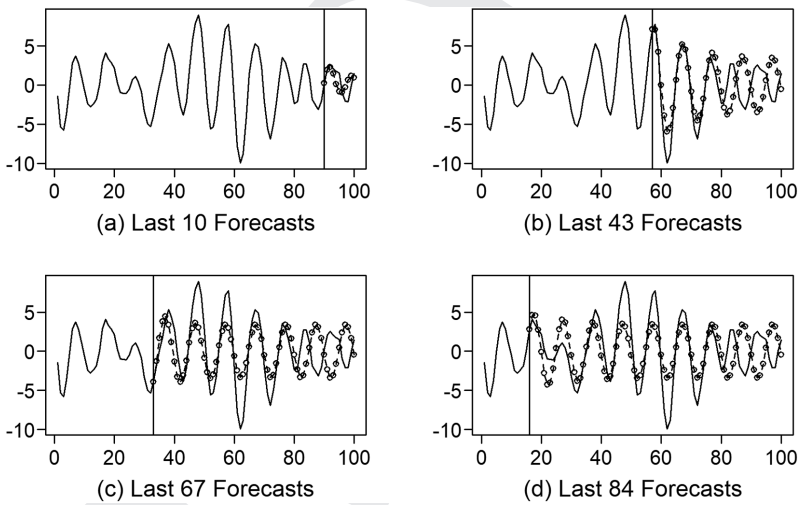

FIGURE 8.10 Forecasts for last k values in `x.ar2` using a cosine signal+noise model fit to the data. Forecasts are shown for $k = 10, 43, 67$, and 84, respectively.

8.2.3.1 A Closer Look at the Cyclic Behavior

Realizations **x.cos** in Figure 8.9(a) and **x.ar2** in Figure 8.9(b) are both described as "cyclic". That is, they show a cyclic behavior but are not "perfectly" sinusoidal. The major difference between the two realizations is that, in the language of Section 1.2.1, **x.cos** has fixed cycle length and **x.ar2** does not. Figure 8.11 is similar to Figures 1.2 and 1.4 for the sunspot data and DFW monthly temperatures, respectively. Figures 8.11(a) and (b) show cycle lengths for the nine full cycles in **x.cos** and in **x.ar2,** respectively. There is very little variation in cycle lengths for the **x.cos** data. Nearly all were length 10 and while cycle four was of length 11 it was "adjusted for" in cycle six, which is of length 9. The first six

cycles in **x.ar2** are just as consistently close to 10 as those in **x.cos**. However, cycle seven is of length 7 and cycles six to nine are all 9 or less and there seems to be no adjustment in cycle lengths to get back in "sync". This variability in cycle length is the cause of the poor forecasts toward the end of the dataset and is not "adjusted for". This behavior is a characteristic of autoregressive data of this type which do not have fixed cycle lengths.

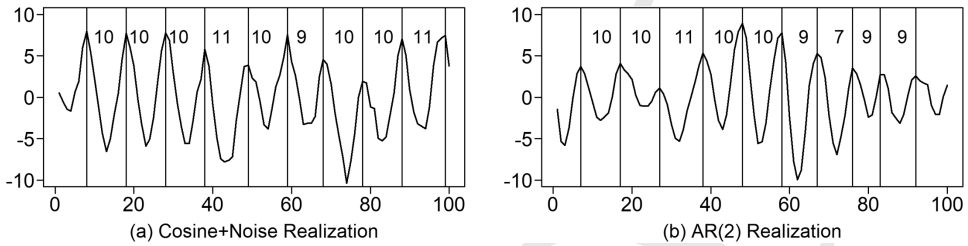

FIGURE 8.11 (a) Realization `x.cos` and (b) `x.ar2` showing cycles and cycle lengths for each of the nine full cycles in each dataset.

Key Point: The amount of variability among cycle lengths provides useful information when deciding whether a cosine signal+noise model should be fit to a given set of data.

We next fit an AR model to the data in **x.ar2.** AIC selects the AR(3) model, in unfactored and factored form,

$$\left(1-.728B+1.230B^2-.193B^3\right)\hat{X}_t=\left(1-1.152B+.908B^2\right)\left(1+.213B\right)\hat{Z}_t=a_t,$$

where the estimated white noise variance is $\hat{\sigma}_a^2=.622$. Thus the dominant 2nd-order factor is very close to the factor in the AR(2) model in (8.18). Figure 8.12 shows the forecasts analogous to those shown in Figure 8.10 from the cosine signal+noise fit.

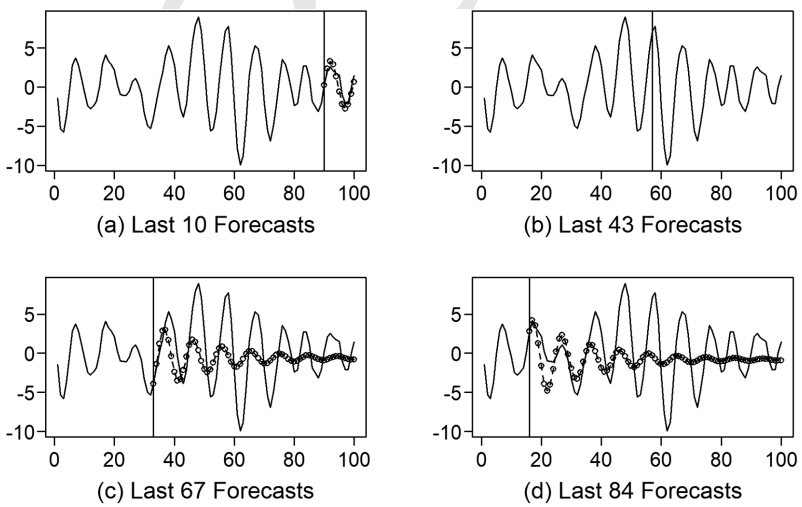

FIGURE 8.12 Forecasts for last k values in `x.ar2` using the AR(3) model fit to the data. Forecasts are shown for $k=10,43,67$ and 84, respectively.

These forecasts, based on an AR(p) fit, "understand" that cycle-length variability may occur. Since the farther we wish to forecast into the future the less confident we are of the timings of peaks and troughs, it is reasonable that forecasts at longer steps-ahead are close to the mean.

Key Point: Before using a cosine signal+noise model to forecast cyclic behavior very far into the future, you should have physical or empirical evidence concerning the permanence of the cyclic behavior.

Example 8.6 Re-examination of the DFW monthly temperature data, sunspot data, and lynx data
Each of these datasets has cyclic behavior. We will examine each with respect to whether a cosine signal+noiose model should be fit to the data.

(a) Dallas Ft. Worth Monthly Temperature Data

In Example 7.3 we fit a nonstationary ARIMA model to the DFW monthly temperature data in ***tswge*** file **`dfw.mon`**. This dataset consists of the average monthly temperature (in degrees Fahrenheit) for the DFW area from January 1900 through December 2020. Consistent with Example 7.3 we will use the data from January 2000 through December 2020 which is in the ***ts*** file created in the following.

```
dfw.2000=window(dfw.mon,start=c(2000,1))
```

The data have a cyclic behavior with a cycle length of 12 and are plotted in Figure 8.13(a). In Example 7.3 it was noted that the "seasonal" component in **`dfw.2000`** could be captured using the single factor $1-1.732B+B^2$ (which is associated with a system frequency of $f=1/12$), in contrast to data such as the **`AirPassengers`** data for which we used a factor $1-B^{12}$.[10] The reason for the difference is that the "seasonal" component of **`dfw.2000`** has a cyclic behavior that is sinusoidal in nature. This leads us to consider a cosine signal+noise model for the **`dfw.2000`** data. Figure 8.12(a) shows the **`dfw.2000`** data and Figure 8.13(b) displays the cycle lengths. Not surprisingly, because of the stability of seasonal data the cycle lengths were 11, 12, or 13 for every cycle. See Figure 1.4 and the associated discussion about temperature patterns in the DFW area. Monthly temperature data such as **`dfw.2000`** seem ideal for cosine signal+noise models.

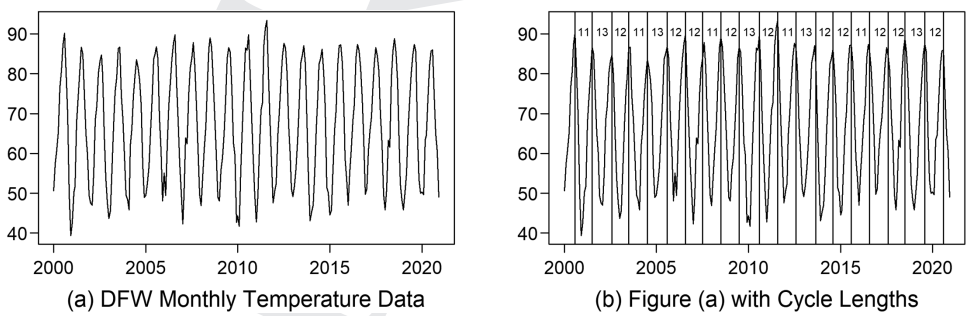

FIGURE 8.13 (a) DFW temperature data in `DFW.2000` and (b) Figure (a) showing cycle lengths.

The command

```
fore.sigplusnoise.wge(dfw.2000,linear=FALSE,f=.0833)
```

10 Recall that $1-1.732B+B_2$ is one of the factors of $1-B^{12}$.

produces the cosine signal+noise model

$$X_t = 67.35 - 16.97\cos\left(2\pi(.1)t\right) - 10.26\sin\left(2\pi(.1)t\right) + \hat{Z}_t, \tag{8.20}$$

where AIC selects the AR(1) model $\left(1-0.34B\right)\hat{Z}_t = a_t$, with $\hat{\sigma}_a^2 = 9.91$.[11]

(a.1) Forecasts of* `dfw.2000` *data using cosine signal+noise model (8.20)

Figure 8.14 shows forecasts for the last 12, 36, 84, and 132 months using the cosine signal+noise model in (8.20). The forecasts stay "in sync" with the actual data, and they estimate the peaks and troughs well.

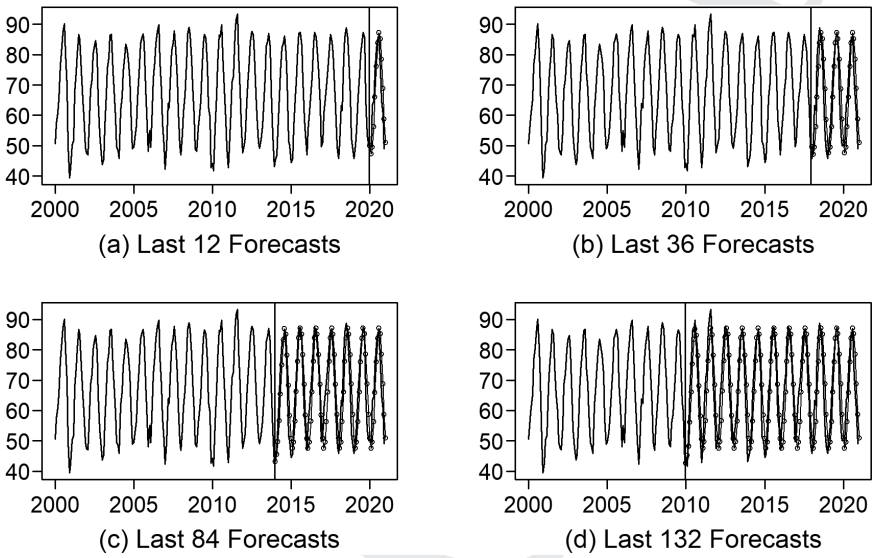

FIGURE 8.14 DFW monthly temperature with forecasts for last 12, 36, 84, and 132 months.

(b) Sunspot Data

The sunspot data have been analyzed in various places in this book. They are introduced in Section 1.2.1 and in Example 6.15 forecasts were obtained although it was noted that these data have more variability in the peaks than in the troughs, so that an ARMA(p,q) model will not be able to explain this nonlinear behavior. We encountered a similar situation with the lynx data, and it is common to analyze the "log lynx" data (which we did). The logs of the sunspot data are not typically analyzed because there are two instances of zero sunspot activity. In this example we will examine the idea of fitting a cosine signal+noise model to the sunspot data so it will be helpful to remove the asymmetry from the data. In order to remove the asymmetry, we will go against tradition and analyze $\log(x_t + 10)$ where x_t is the sunspot number at year t. In the following we will refer to $\log(x_t + 10)$ as the "log sunspot data".

Figure 8.15(a) shows the sunspot data in `sunspot2.0`, while Figures 8.15(b) and (c) are plots of the associated sample autocorrelations and Parzen spectral density estimate, respectively. Figure 8.15(d) is a plot of $\log(x_t + 10)$ (`sunspot2.0`+10) along with sample autocorrelations and Parzen spectral density estimate. Figure 8.15(d) has a more symmetric ARMA-like appearance, and it also has the appearance of a cosine signal+noise realization. Interestingly, the sample autocorrelations and spectral estimate are very similar in appearance to those associated with the "raw" sunspot data.

11 The sample mean is close to zero (within 4 decimal places) as expected

AIC selects an AR(9) model for the log sunspot data. A factor table for this model (not shown) indicates a 2nd-order factor associated with frequency $f = .092$ (period 1/.092=10.9) and a 1st-order factor associated with frequency $f = 0$. This is consistent with the factor table shown in Table 5.6 for the AR(9) fit to the raw sunspot data.

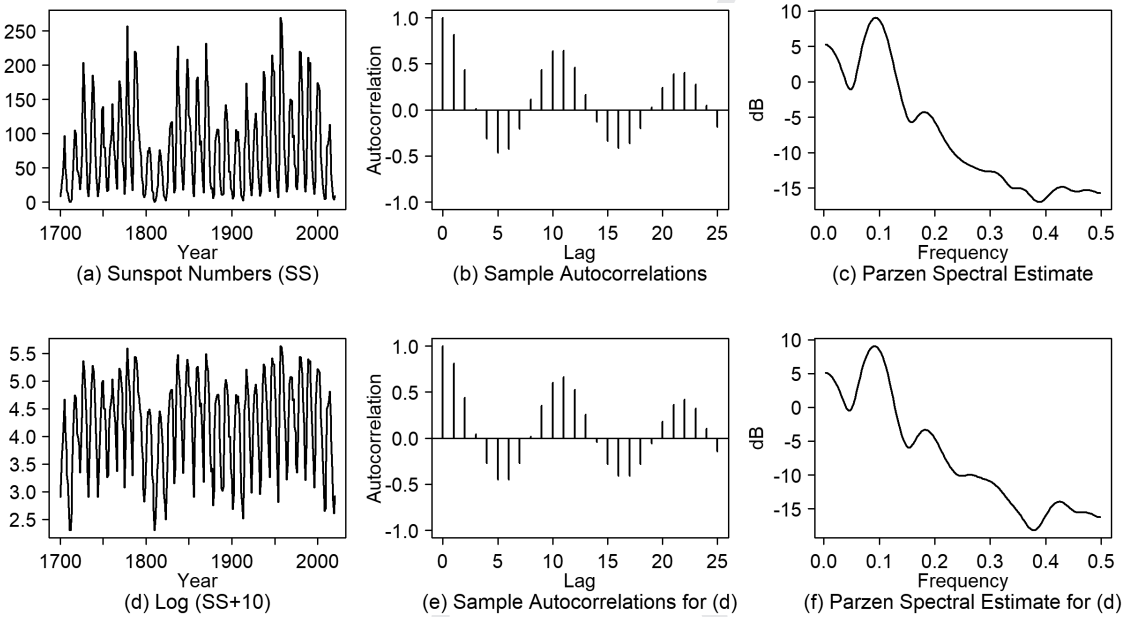

FIGURE 8.15 (a – c) sunspot data in `sunspot2.0`, sample autocorrelations, and Parzen spectral density estimate; (d–f) log sunspot, sample autocorrelations, and Parzen spectral density estimate.

We next consider whether a cosine signal+noise model is appropriate for the "log sunspot data". Figure 8.16 is a plot of the log sunspot data along with cycle lengths. The plot shows quite a lot of variability in cycle lengths, which go from a low of 7 to a high of 17 making a viable cosine signal+noise model unlikely. Any cosine fit to the data will produce forecasts that become out of sync with the data and will very likely at times forecast a peak when the data are at a trough.

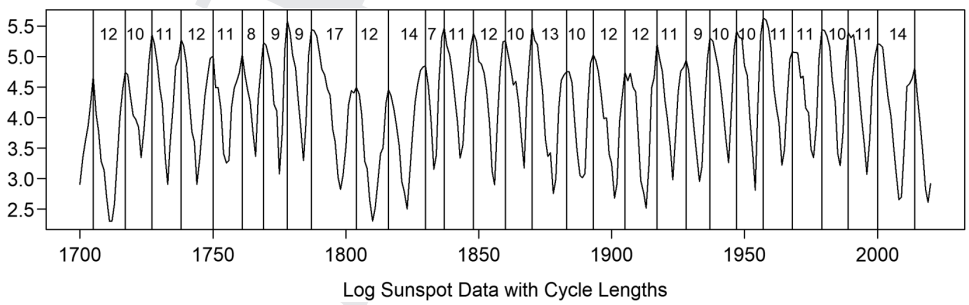

FIGURE 8.16 Log sunspot data showing cycle lengths for each sunspot cycle.

The cosine signal+noise model fit to the log sunspot data is interesting in that the cosine component is weak and most of the cyclic behavior in the model is accounted for by Z_t. The forecasts shown in Figure 8.17 are consistent with this interpretation.

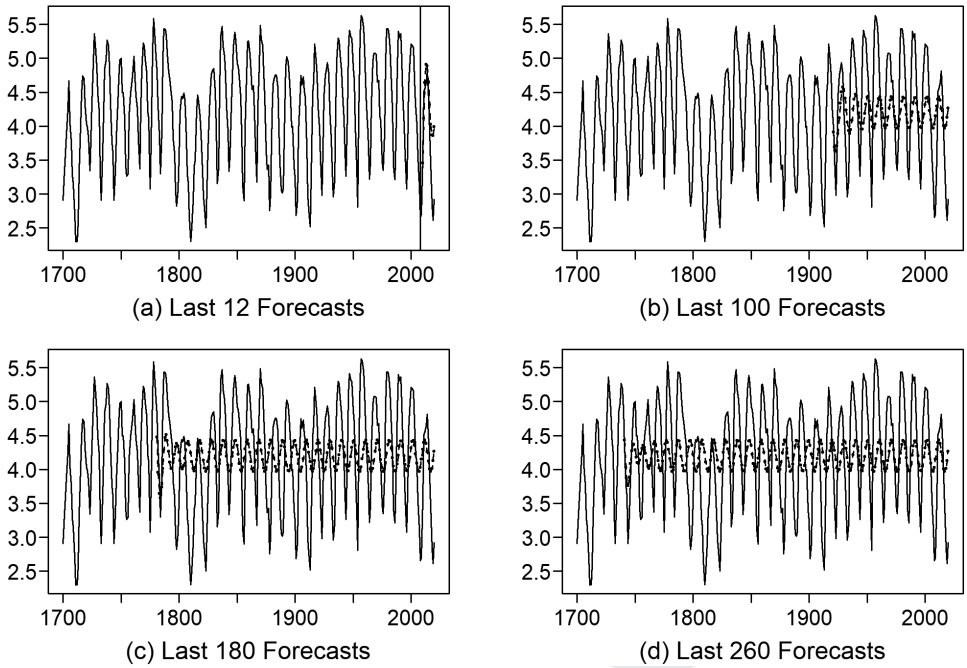

FIGURE 8.17 Log sunspot data with forecasts for last 12, 100, 240, and 280 years using a cosine signal+noise model fit to the data.

(c) Lynx Data

We return to the lynx dataset examined in Examples 5.3, 6.5, and 6.13. The feature of the data that has caused it to be of interest is the regular cyclic behavior of the data with cycle lengths of about 10 years. In Example 5.3 it was noted that because of the asymmetry in the actual lynx data, researchers have historically analyzed the log (base 10) of the number of lynx trapped. The lynx data are contained in Base R dataset **`lynx`**. The log-lynx data are shown in Figure 8.18(a).

Researchers have debated whether an AR model or a cosine signal+noise model is appropriate for modeling the log-lynx data. Campbell and Walker (1977) fit a cosine signal+noise model of the form (7.32) to the data while Tong (1977) fit an AR(11) model as we did in Example 6.5.[12] Priestley (1962) proposed the $P(\lambda)$ test, which was designed to determine whether a cosine signal should be included in a model. Bhansali (1979) used the $P(\lambda)$ test and concluded that the log-lynx data contained a cosine signal. The factored form of Tong's model has a 2nd-order factor which has roots very close to the unit circle (the absolute reciprocal is 0.98) associated with frequency $0.104 (= 1/9.6)$. Woodward and Gray (1983) generated realizations from Tong's model and showed that the $P(\lambda)$ test usually incorrectly "detected" a cosine signal in these AR(11) realizations. Consequently the fact that the $P(\lambda)$ test detected a cosine component does not definitely imply the presence of one.

Figure 8.18(a) shows a plot of the log-lynx data and Figure 8.18(b) shows the cycle lengths in the log-lynx data. There it can be seen that the cycle lengths are consistently 9 or 10 with the exception of the last cycle of 12.

12 Tong's model is $\left(1-1.13B+.51B^2-.23B^3+.29B^4-.14B^5+.14B^6-.08B^7+.04B^8-.13B^9-.19B^{10}+.31B^{11}\right)\left(X_t-2.90\right)=a_t$. This model is very close to the YW model obtained using **`est.ar.wge`**.

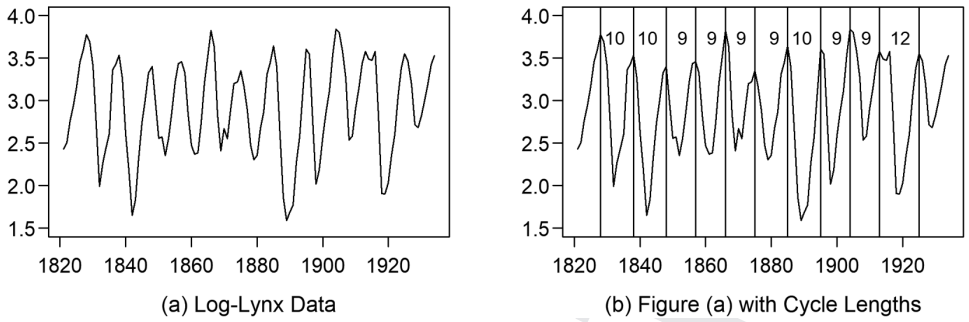

FIGURE 8.18 (a) Log-lynx data in **`llynx`** and (b) Figure (a) showing cycle lengths.

Using ***tswge*** command **`fore.sigplusnoise.wge`**, the resulting cosine signal+noise model is

$$X_t = 2.91 - .093\cos\left(2\pi(.103)t\right) - .607\sin\left(2\pi(.103)t\right) + \hat{Z}_t, \tag{8.21}$$

where $\left(1 - 1.065B + 0.376B^2\right)\hat{Z}_t = a_t$, with $\hat{\sigma}_a^2 = .04$. The last 12, 36, 60 and 84 step-ahead forecasts using (7.41) are shown in in Figure 8.19. There we see that the forecasts stay "in sync". Biologists suggest a possible explanation for the cyclic behavior of the lynx data is that it is related to the corresponding cycle for the horseshoe hare, which is a staple for the lynx. However, the explanation for cyclic behavior is not as explainable as that for the DFW monthly temperature data. Given the information above, it is clear that the cosine signal+noise model and the ARMA(2,3) (or AR(11)) provide reasonable fits to the log-lynx data. We re-emphasize the following Key Points.

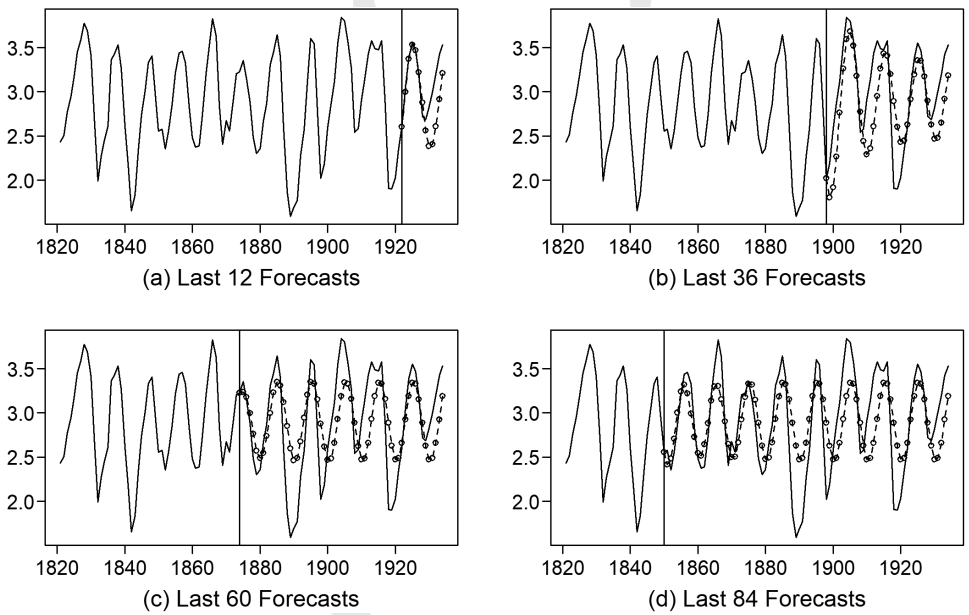

FIGURE 8.19 Log-lynx data with cosine signal+noise forecasts for last 12, 36, 60, and 84 years.

Key Points:

1. Before using a cosine signal+noise model to forecast cyclic behavior very far into the future, you should have physical or empirical evidence concerning the permanence of the cyclic behavior.
2. Tong(1977) argued that it was more reasonable to use ARMA-type models (which are capable of describing random phase shifts) than a fixed cycle-length model with an unusual cycle length such as 1/.104=9.6 years. Tong's reasoning is compelling, but "All models are wrong, but some are useful." (Maybe we've already said that.) That is, neither model is correct and both may be helpful.

Figure 8.20 shows a plot analogous to Figure 8.19 for a realization generated from the ARMA(2,3) model fit to the log-lynx data. We see that the cycle length variability is slightly more than that of the log-lynx data, and this is reflected by forecasts getting out of sync. However, other realizations from the ARMA(2,3) model displayed forecast behavior very similar to that for the log-lynx data.

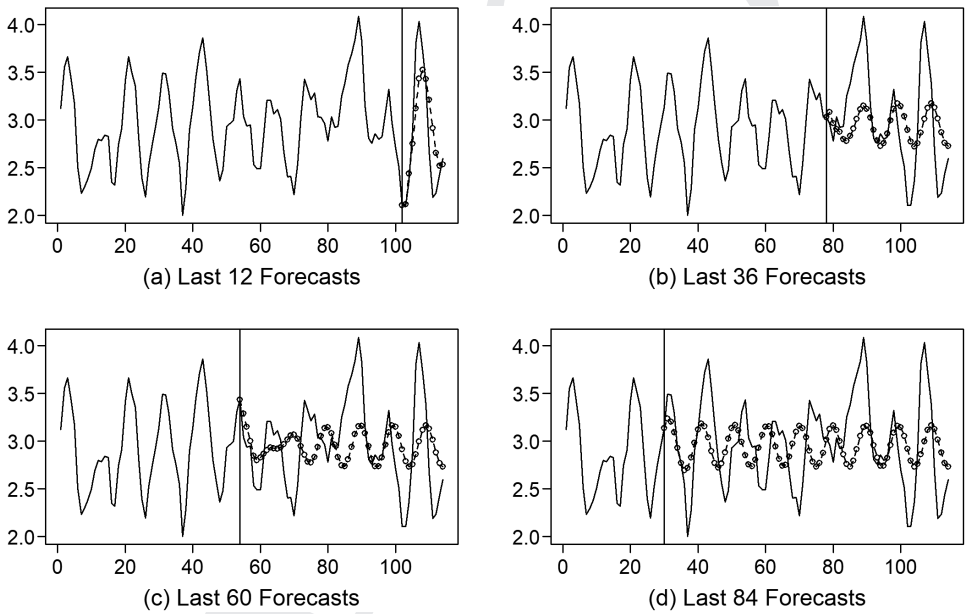

FIGURE 8.20 Realization from the ARMA(2,3) model fit to the log-lynx data with forecasts for last 12, 36, 60, and 84 years.

8.3 CONCLUDING REMARKS

In this chapter we discussed signal+noise models for which the mean changes with time. While many "signals" exist, in this chapter, we focused on the line and the sinusoidal (cosine) curve. The crucial (and sometimes difficult!) question that the analyst must answer is whether it is reasonable to expect that the line or sinusoidal pattern is expected to continue. Tests and graphical displays were described which can help answer this question.

Procedures to fit the appropriately chosen model and derive forecasts were presented in several examples of simulated and real data.

APPENDIX 8A

(a) **co.wge(x,maxp)** is a function that performs a Cochrane-Orcutt (CO) test for trend in the time series realization, **x**. The test involves fitting an AR model to the residuals from the fitted line. The order, p, is selected by AIC, and **maxp** specifies the maximum order for p allowed. The autoregressive parameters are estimated using Burg estimates to avoid problems with roots inside the unit circle.

x is a vector containing the time series realization $x_t, t = 1,\ldots,n$.
maxp is the maximum order p for the AR model fit to the residuals from the least squares line.

Output contains the following:

$z are the residuals from the fitted line.
$b0hat is the estimated y-intercept of the fitted line using the CO method.
$b1hat is the estimated slope of the fitted line using the CO method.
$z.order is the order, p, fit to the residuals.
$z.phi are the coefficients of the AR model fit to the residuals.
$pvalue is the p-value of the CO test for the significance of the slope.
$tco is the Cochrane-Orcutt test statistic.

(b) **fore.sigplusnoise.wge(x,linear,method,freq,max.p,n.ahead, lastn,plot, alpha,limits)** is a function that fits a signal+noise model and forecasts **n.ahead** steps ahead for models of the form

(i) $X_t = b_0 + b_1 t + Z_t$ where Z_t satisfies a zero-mean AR process.
(ii) $X_t = C_0 + A\cos(2\pi ft) + B\sin(2\pi ft) + Z_t$ where Z_t satisfies a zero-mean AR process.

Note that signals are restricted to linear or (single) sinusoidal types. You can calculate and plot forecasts **n.ahead** steps beyond the end of the series or you can forecast the last **n.ahead** data values. In the linear model above, all parameters are estimated including the order (p) and associated parameters of the AR(p) model fit to the residuals from the fitted line. In the cosine model the user must specify f. All other parameters are estimated.

x = realization
linear = logical variable. If **linear = TRUE** (default) then the forecasts will be based on the line+noise model above. If **linear = FALSE** then the cosine signal+noise model is fit to the data.
freq is the variable f that must be specified by the user if **linear=FALSE** (default is **freq = 0**).
max.p is the maximum order autoregressive model to be fit (based on AIC) to the residuals.
n.ahead = the number of steps ahead you want to forecast (default = 10)
lastn = logical variable. If **lastn = FALSE** (default) then the forecasts are for $x(n+1), x(n+2)$, …, $x(n+$**n.ahead**$)$ where **n** is the length of the realization. If **lastn = TRUE**, then the program forecasts the last **n.ahead** values in the realization. (default = **FALSE**)
plot = logical variable. If **plot = TRUE** (default) then the forecasts will be plotted along with the realization. If **plot = FALSE** then no plots are output.
alpha specifies the level of the prediction limits (default=.05)

limits = logical variable. If **limits = TRUE** then the 95% forecast limits will be plotted along with the forecasts. (default = **TRUE)**

Example:
Suppose a realization **x** is modeled using a cosine signal+noise model with frequency .0833. The following command will fit the cosine signal+noise model, then fit an AR(p) model to the residuals, and forecast 20 steps beyond the end of the data and plot the results.

```
xfore=fore.sigplusnoise.wge(x,linear=FALSE,freq=0.0833,n.ahead=20,
lastn=FALSE,limits=FALSE)
```

In a real data situation in which you believe the realization may be a cosine signal+noise model, you will typically be able to obtain a good estimate of **freq** using a spectral density estimate or fitting a high order AR model and checking the factor table for the dominant frequency.

(c) **gen.sigplusnoise.wge(n,b0,b1,coef,freq,psi,phi,vara,plot,sn)** is a function that generates a realization from the signal-plus-noise model

$$X_t = b_0 + b_1 t + coef[1]\cos(2\pi freq[1]t + psi[1]) + coef[2]\cos(2\pi freq[2]t + psi[2]) + Z_t$$

n = length of realization to be generated (t in the above formula specifies the integers from 1 to n)
b0 is the y-intercept
b1 is the slope of a linear regression component (default = 0)
coef is a two-component vector specifying the coefficients (if only one cosine term is desired, define **coef[2] = 0**) (default = **c(0,0)**)
freq is a two-component vector specifying the frequency components (0 to .5) (default = **c(0,0)**)
psi is a two-component vector specifying the phase shift (0 to 2π) (default = **c(0,0)**)
phi is a vector containing the AR model for the noise process.
vara is the variance of the noise (default = 1).
plot is a logical variable. **plot = TRUE** (default) produces a plot of the generated realization.
sn determines the seed number used in the simulation. (default = 0) (see Note below)

Note: **sn=0** (default) produces results based on a randomly selected seed. If you want to reproduce the same realization on subsequent runs of **gen.sigplusnoise.wge** then set **sn** to the same positive integer value on each run.

Example:
The command

```
x=gen.sigplusnoise.wge(n=100,coef=c(1.5,3.5),freq=c(.05,.2),psi=
c(1.1,2.8),
phi=.7,vara=2)
```

calculates $X_t = 1.5\cos(2\pi(.05)t + 1.1) + 3.5\cos(2\pi(.2)t + 2.8) + Z_t$, where $(1 - .7B)Z_t = a_t$, and $\sigma_a^2 = 1$.

(d) **slr.wge(x,tstart)** is a function that uses base R function **lm** to fit a simple linear regression line to the data in vector x.

x is a vector containing the time series realization $x_t, t = 1,\ldots,n$
tstart is the index of the first data value in **x**. The independent variable is **time= tstart:length(x)**

Output:

$time contains the values of the independent variable.

That is, the value **tstart, tstart+1, …, tstart+length(x)-1**

$b0hat is the least squares estimate of the y-intercept of the fitted line.
$b1hat is the least squares estimate of the slope of the fitted line.
$pvalue is the *p*-value for the test of the significance of the slope.
$tstatistic is the *t*-test statistic for the test of significance of the slope.

(e) **wbg.boot.wge(x,nb,alpha,pvalue=TRUE,sn=0)**

This function performs a Woodward-Bottone-Gray (WBG) test for trend in the time series realization, **x**. The test involves bootstrap replications of the AR model fit to the original data. The maximum value allowed for *p* is 5. Burg estimates are used to avoid problems with roots inside the unit circle and because of the computer intensive nature of the test.

x is a vector containing the time series realization $x_t, t = 1,\ldots,n$.
nb is the number of Bootstrap replications (default is 399).
alpha is the significance level of the test.
pvalue=TRUE prints out the *p*-value of the test.
sn determines the seed number used in the simulation. (default = 0).

Output contains the following:
$p is the AR order used for the bootstrap simulations.
$phi are the coefficients of the AR model fit to the data.
$pv is the p-value of the test.

EXERCISES

8.1 Generate and plot four realizations from the model

$$\left(1-.79B-.70B^2+.255B^3+.24B^4\right)X_t = a_t,$$

where $\sigma_a^2 = 1$.

(a) Do any of these realizations have a linear-trending behavior? Explain why or why not, based on the model.
(b) Use the Cochrane-Orcutt test to test for trend in these datasets.

8.2 Consider the following datasets:
(a) **dfw.yr**
(b) **wtcrude2020**
(c) **tesla**
(d) **bitcoin**
(1) Fit an ARMA or ARIMA (not linear trend) model to the data. Explain why you have decided to (or not to) include a unit root in your model.
(2) Test the datas for trend using Cochrane-Orcutt (**co.wge**) and **wbg.boot.wge**.
(3) For each model decide which model you prefer and why.

8.3 Using the **patemp** dataset,
(1) fit a cosine+noise model
(2) Forecast three "cycles" beyond the end of the dataset.
(3) "Forecast" the last three cycles, six, and nine cycles. Compare the forecasts with the true values using plots and RMSEs.