Chapter 7

Time Series Basics

This chapter defines a time series, describes where a time series falls with respect to other stochastic processes, introduces some basic properties of time series, and introduces some basic operations that can be applied to a time series. The presentation given here does not replace a full-semester class on time series analysis, but does provide an elementary introduction to the topic. Each section ends with a brief review of the computational tools available in R for time series analysis.

7.1 The Big Picture

A time series is a sequence of observed data values that are collected over time. The analysis of a time series is an important sub-discipline of statistics and data science that has applications in a variety of areas, including economics, business, science, and engineering.

Classical statistical methods rely on the assumption that the data values collected constitute a simple random sample, which implies that data values are realizations of mutually independent and identically distributed random variables. This is nearly universally not the case in time series analysis because nearby observations collected over time tend to be correlated. Special probability models and associated statistical methods have been developed to account for this correlation. When the focus is on the correlation between observations in the time series, analysis tools from the *time domain* are employed. When the focus is on the periodic behavior in the time series, analysis tools from the *frequency domain* are employed. We begin our exploration of time series with a subsection containing examples.

7.1.1 What is a Time Series?

The essence of a time series is best captured in a sequence of examples. The first example is the monthly number of kilowatt hours required for powering the utilities in my home from 2011 through 2018. The second example is the monthly number of international airline passengers between 1949 and 1960. The third example is a realization of what is known as *Gaussian white noise*. The fourth example is a realization of what is known as a *random walk*.

> **Example 7.1** The monthly number of kilowatt hours to power my home in Williamsburg, Virginia between 2011 and 2018 are given in Table 7.1. Scan the table carefully to see if you can determine any patterns in the time series. To provide more context, here is a little more information about the house that my family lived in between 2011

Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2011	2731	1822	1189	1229	1260	2204	2518	1960	1032	788	1508	2279
2012	1667	1695	1220	872	1189	1164	1851	1789	1370	962	1716	1678
2013	2254	2362	1916	1293	1253	1635	1809	1562	1348	872	1290	2619
2014	3089	2217	2072	1270	1543	1642	1892	1688	1658	984	1609	2577
2015	2712	3363	1887	1494	1260	1127	1865	1741	1430	1588	1535	1626
2016	3004	2344	1969	1431	1456	2029	2294	2036	2173	1132	1834	1713
2017	2583	1810	1728	1145	1253	1696	1936	1875	956	1010	1751	1506
2018	3698	1767	1871	1270	966	1141	1463	1452	1484	1043	1378	1499

Table 7.1: Number of kilowatt hours to power a home.

and 2018. There were no additions made to the home during these years, nor were there any new windows or insulation installed. The two-zone heat pump system that cools the house in the summer and heats the house in the winter did not change during these years. The smallest value in the series occurred in October 2011, when 788 kilowatt hours were consumed. The largest value in the series occurred in January 2018, when a spectacular 3698 kilowatt hours were consumed. January 2018 was one of the coldest Januarys on record in Virginia, which caused the spike in kilowatt hours consumed. The pipes in my neighbor's house burst on one cold night in January. Viewed in table form, the data just looks like a mass of numbers. But plotting the data on a time axis reveals some of the patterns associated with the data values over time.

We use R to plot the observations over time. The first step is to get the time series observations into an R vector. This can be done for a small data set with the `c` function. The R statement below places the time series into the vector `kwh` in chronological order.

```
kwh = c(2731, 1822, 1189, ... , 1499)
```

Alternatively, if the data set is large and is contained in an external file, the `scan` function can be used to read the time series observations from the external file as

```
kwh = scan("kwh.d")
```

where `kwh.d` is a file in the current working directory that contains the energy consumption values in kilowatt hour values, one per line.

The next step is to use the `ts` function to convert the data values in the vector `kwh` to a time series object, which will allow us to use many of the R time series analysis operators included in the base language.

```
kwh.ts = ts(kwh, frequency = 12, start = c(2011, 1))
```

Setting the `frequency` argument in `ts` to 12 lets R know that the data is collected monthly. If the time series consisted of quarterly observations, for example, then `frequency` would be set to 4. Setting the `start` argument to `c(2011, 1)` lets R know that the time series starts in January of 2011. If the first observation in the time series was sampled in March of 2013, for example, then `start` would be set to `c(2013, 3)`.

The next step is to plot the time series, which can be done with the `plot.ts` function.

```
plot.ts(kwh.ts)
```

The time series plot is shown in Figure 7.1. The horizontal axis is the time t, measured in years. The tick marks on the horizontal axis correspond to the January observation from each year. The vertical axis is the observed number of kilowatt hours consumed at time t, denoted by x_t. The points in the time series are connected with lines, but this is largely a matter of personal taste.

Plotting the time series in this fashion is a critical initial step in the analysis of an observed time series. Carefully examining this plot often informs the analyst of the type of probability model that might be appropriate. Unusually small and large observations should be carefully assessed to determine whether the x_t value was recorded properly. Shifts in the heights of the x_t values might correspond to events associated with the time series, such as installing a more efficient heat pump or adding a new room to the house for `kwh.ts`. The plot also allows the analyst to inspect the time series for any *trends* which correspond to gradual increases or decreases in the mean level of the process. In addition, the plot allows the analyst to identify any *seasonal* components, that is, fluctuations which are periodic in nature, that might be present in the time series. Some time series have a change in the variability of the observations that may also be identified in the plot.

There are some preliminary conclusions that can be drawn from the time series plot in Figure 7.1. First, there does appear to be a 12-month cyclical pattern, that is surely influenced by the annual outdoor temperature cycles. The peak energy consumption typically occurs in January. This is consistent with the fact that heat pumps have difficulty during the winter months because there is not much heat to pump, making them inefficient. Second, after accounting for the annual cycle, there does not seem to be any systematic increase or decrease to the amount of energy consumed over the 8-year period. This is consistent with the fact that no energy improvements were made to the home during the 8-year period. There is, of course, short-term change in the mean value of the time series due to the seasonal component of the time series, but there does not

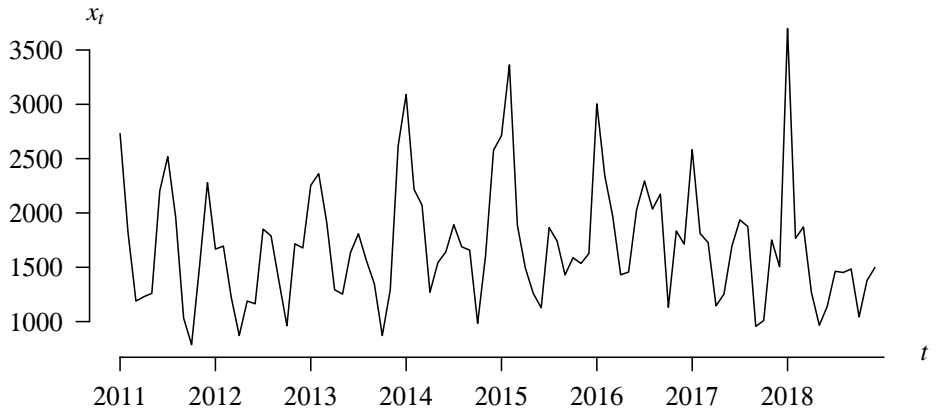

Figure 7.1: Single home energy consumption (in kilowatt hours) 2011–2018.

appear to be any long-term change in the mean value of the series. Third, there is significant random sampling variability associated with the time series. Although forecasts could be made from this data set for values of the time series into the future, they would be fairly imprecise predictions because of the random sampling variability. Some of the variability could be explained by the average outdoor monthly temperature during a particular month, with hot summer months and cold winter months requiring more energy to cool and heat the home. Some of the variability could also be explained by the fact that all months do not have the same number of days, which is easily accounted for. Time series analysts refer to the random sampling variability that remains after all of the *signal* has been accounted for as *noise*. The terms signal and noise are familiar terms in fields such as statistics, data science, astrophysics, and electrical engineering. The term *noise* is analogous to *error* from regression theory.

The home energy consumption example has shown that significant insight concerning a time series can be gleaned by an understanding of the context associated with the time series and the crucial step of making a simple plot of the data values over time. We will return to the home energy consumption time series later for further analysis. The next time series illustrates the increase in international airline travel between 1949 and 1960.

Example 7.2 The number of monthly international airline passengers, in thousands, between January 1949 and December 1960 resides in an R built-in data set named `AirPassengers`. All that is necessary to see the observations is to type the name of the data set. (Notice that all R commands are case sensitive.)

```
AirPassengers
```

The output is shown below.

```
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1949 112 118 132 129 121 135 148 148 136 119 104 118
1950 115 126 141 135 125 149 170 170 158 133 114 140
1951 145 150 178 163 172 178 199 199 184 162 146 166
1952 171 180 193 181 183 218 230 242 209 191 172 194
1953 196 196 236 235 229 243 264 272 237 211 180 201
1954 204 188 235 227 234 264 302 293 259 229 203 229
1955 242 233 267 269 270 315 364 347 312 274 237 278
1956 284 277 317 313 318 374 413 405 355 306 271 306
1957 315 301 356 348 355 422 465 467 404 347 305 336
1958 340 318 362 348 363 435 491 505 404 359 310 337
1959 360 342 406 396 420 472 548 559 463 407 362 405
1960 417 391 419 461 472 535 622 606 508 461 390 432
```

Again, scan these data values and look for patterns. The lowest number of international air travelers was 104,000 in November 1949. The highest number of international air travelers was 622,000 in July 1960. Unlike the previous example, it is not necessary to convert `AirPassengers` from a vector to a time series object. Typing

```
str(AirPassengers)
```

shows that `AirPassengers` is already a time series object, as shown below, by using the `str` (structure) function in R.

```
 Time-Series [1:144] from 1949 to 1961: 112 118 132 129 121 135 ...
```

This useful function tells you that there are $12 \times 12 = 144$ observations in the time series and lists the first few observations. By using the `help` function

```
help(AirPassengers)
```

additional information about the time series reveals that the observations are the number of international airline passengers (in thousands) per month during 1949–1960. Using the `plot.ts` function as in the previous example gives a graph of the time series over time, which is given in Figure 7.2.

Unlike the previous time series, this time series displays a trend. The number of international airline passengers is increasing over time. Although the number of passengers is increasing over time, it is not clear whether the increase is linear, quadratic, or exponential, and this would require further analysis to determine which functional form provides the best model for the increase. As was the case with the time series from the first example, there is also a 12-month cycle that is apparent in the data. The months of July and August–when school is typically not in session–tend to be the busiest months. The cyclic variation appears to be less sinusoidal in nature than the energy consumption time series because there is not a sinusoidal external time series (outdoor temperature) driving the international airline travel time series.

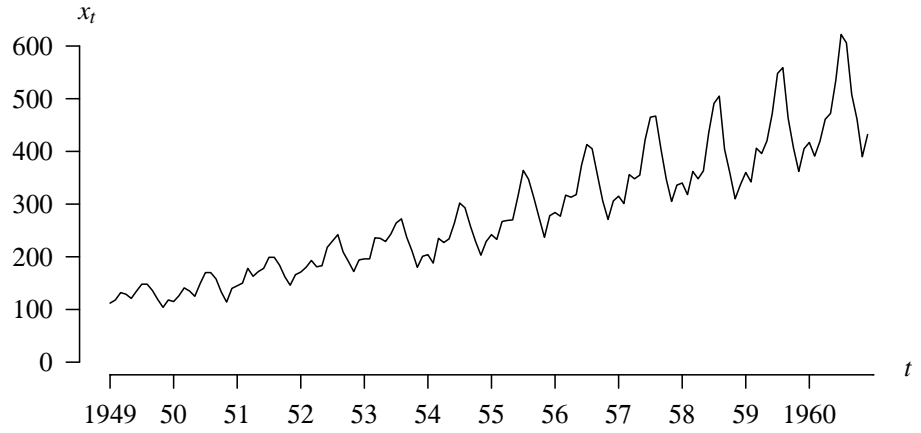

Figure 7.2: International airline passengers (in thousands) 1949–1960.

Before describing a third example of a time series, we define some notation. A common convention in probability theory is to use uppercase letters, such as X and Y, to denote random variables. When encountering a time series, it is often helpful to subscript the random variable denoting the time series observations with t, for time, as X_t. So a time series consisting of the time-ordered observations $X_1, X_2, \ldots, X_n$ can be referred to in the abstract as $\{X_t\}$, which is more compact. But

when referring to a specific realization of a time series (either collected as data or generated by a computer via simulation) consisting of the time-ordered observations $x_1, x_2, \ldots, x_n$, we switch to the lowercase version $\{x_t\}$. This is why the vertical axes in Figures 7.1 and 7.2 were labeled x_t for the $n = 96$ home energy consumption observations from Example 7.1 and the $n = 144$ airline passenger counts from Example 7.2. The notation developed in this paragraph will be apparent in the formal definition of noise, which is defined next.

Instead of analyzing a realization of a time series as in the previous two examples, time series analysts often formulate and fit a *probability model* for a time series. This process is roughly the time series equivalent of fitting a univariate probability distribution, such as the normal distribution, to a data set. We will again refer to the time series as $\{X_t\}$ when constructing such a time series model, but when the index values for t are not obvious by context, we add the additional parameter T, which is the set of allowable values for t using $\{X_t, t \in T\}$. The set T will almost universally be either the set of all integers (when it is necessary to consider observations with negative t values) or the set of all nonnegative integers (when a time origin is necessary).

Most time series cannot be described by a deterministic function. In order to inject randomness into the time series model, it has become common practice to define *noise*, which consists of random shocks that will make a time series model stochastic rather than deterministic. Three varieties of noise used by time series analysts are defined next. In some application areas, noise might be referred to as *error* or *disturbance*.

Definition 7.1 The time series $\{Z_t\}$ that is a sequence of mutually independent random variables $Z_1, Z_2, \ldots, Z_n$, each with population mean 0 and finite population variance σ_Z^2, is known as *white noise*, and is denoted by

$$Z_t \sim WN\left(0, \sigma_Z^2\right).$$

The time series $\{Z_t\}$ that is a sequence of mutually independent and identically distributed random variables $Z_1, Z_2, \ldots, Z_n$, each with population mean 0 and finite population variance σ_Z^2, is known as *iid noise*, and is denoted by

$$Z_t \sim IID\left(0, \sigma_Z^2\right).$$

The time series $\{Z_t\}$ that is a sequence of mutually independent and identically normally distributed random variables $Z_1, Z_2, \ldots, Z_n$, each with population mean 0 and finite population variance σ_Z^2, is known as *Gaussian white noise*, and is denoted by

$$Z_t \sim GWN\left(0, \sigma_Z^2\right).$$

It is clear from Definition 7.1 that the three varieties of noise were defined from the more general case to the more specific case so that

$$GWN \subset IID \subset WN.$$

All three varieties share common population means and variances:

$$E\left[Z_t\right] = 0 \qquad \text{and} \qquad V\left[Z_t\right] = \sigma_Z^2.$$

These three probability models are, in some sense, the simplest possible time series models, although time series that are well-modeled by the three models are very rare in practice. Rather than approximating a real-world time series, they serve as building blocks for more realistic models. They are often used to describe the probability distribution of *error terms* in a probability model for a time series. In the next example, you will see a plot of a realization of Gaussian white noise.

Example 7.3 White noise, iid noise, and Gaussian white noise are just sequences of mutually independent observations centered around zero. A time series of $n = 100$ Gaussian white noise observations with population variance $\sigma_Z^2 = 1$, for example, can be generated and placed in the vector `x` with the R command

```
x = rnorm(100)
```

The choice of $n = 100$ was arbitrary. Defining `x` after an optional call to `set.seed(8)` to establish the random number stream, the values contained in the vector `x` can be converted to a time series with the `ts` function and then plotted as in the two previous examples with the `plot.ts` function. The resulting plot is given in Figure 7.3. The time series that consists of Gaussian white noise values has a minimum value of $x_{90} = -3.015$ and a maximum value of $x_{79} = 2.376$. Of the 100 Gaussian white noise values, 46 are positive and 54 are negative. Since the time series consists of mutually independent and identically distributed random variables, $x_1, x_2, \ldots, x_{100}$ are of no use in predicting the next value in the time series, x_{101}.

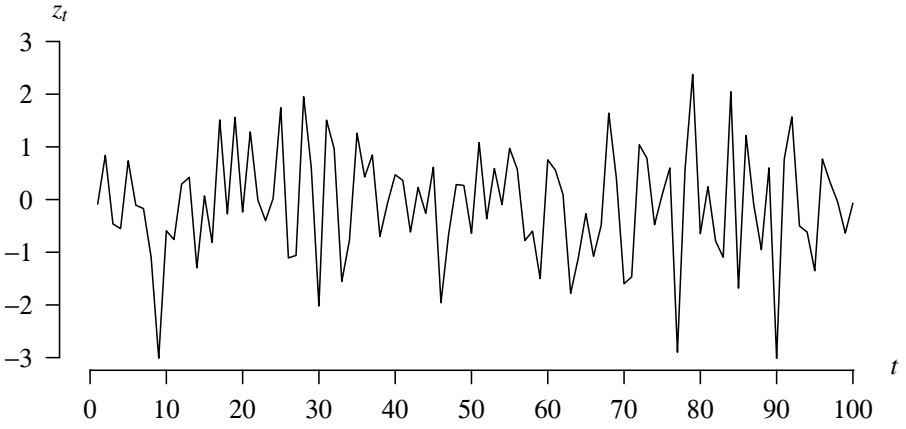

Figure 7.3: Time series plot of $n = 100$ Gaussian white noise observations.

Most time series that are encountered in practice do not behave in a fashion that approximates white noise, iid noise, or Gaussian white noise. It is more often the case that the value of the observation in the time series at time t will depend on the values of one or more of the previous observations in the time series. One time series model that exhibits this dependency is known as a *random walk*, which is defined and illustrated next.

Example 7.4 A time series that is a random walk $\{X_t\}$ is generated by the recursive equation

$$X_t = X_{t-1} + Z_t,$$

where $\{Z_t\}$ is Gaussian white noise. This is to say that the current value of the time series is the previous value of the time series plus a Gaussian white noise term. An algorithm for generating a random walk $\{X_t\}$ using Gaussian white noise is given in the

pseudocode below. Indentation denotes nesting. In the second step, the initial observation, X_1, is arbitrarily set to 0. This step could instead have been $X_1 \leftarrow \mu$, where μ is the mean value of the time series. Alternatively, this start-up condition could instead have been $X_1 \leftarrow Z_1$, which starts the time series at a random position rather than 0. These two alternatives can be combined into $X_1 \leftarrow \mu + Z_1$.

$t \leftarrow 1$
$X_1 \leftarrow 0$
while $(t < n)$
    $t \leftarrow t + 1$
    generate $Z_t \sim N\left(0, \sigma_Z^2\right)$
    $X_t \leftarrow X_{t-1} + Z_t$

This algorithm for generating a random walk can be implemented in R using the code given below. It is assumed that the population variance of the Gaussian white noise is $\sigma_Z^2 = 1$.

```
set.seed(8)
n    = 100
x    = numeric(n)
time = 1
x[1] = 0
while (time < n) {
  time = time + 1
  z = rnorm(1)
  x[time] = x[time - 1] + z
}
x = ts(x)
plot.ts(x)
```

The realization of the random walk stored in the vector `x` is plotted in Figure 7.4. The time series associated with the random walk takes on a decidedly different pattern than that of the Gaussian white noise from Figure 7.3. This realization of a random walk looks quite a bit like some graphs of economic data, such as the daily closing price of a stock or a stock market average. This makes sense because it might be the case that the probability model for the value of the closing price of the stock might be expressed as

$$[\text{today's closing price}] = [\text{yesterday's closing price}] + [\text{noise}]$$

which is equivalent to the random walk model

$$X_t = X_{t-1} + Z_t.$$

The random walk model does such a good job of approximating certain economic data that it can be difficult to distinguish a real set of economic data from a realization of a random walk generated by simulation.

As an illustration, Figure 7.5 contains graphs of three time series. One of these time series is the first $n = 100$ closing values of the Dow Jones Industrial Average during the

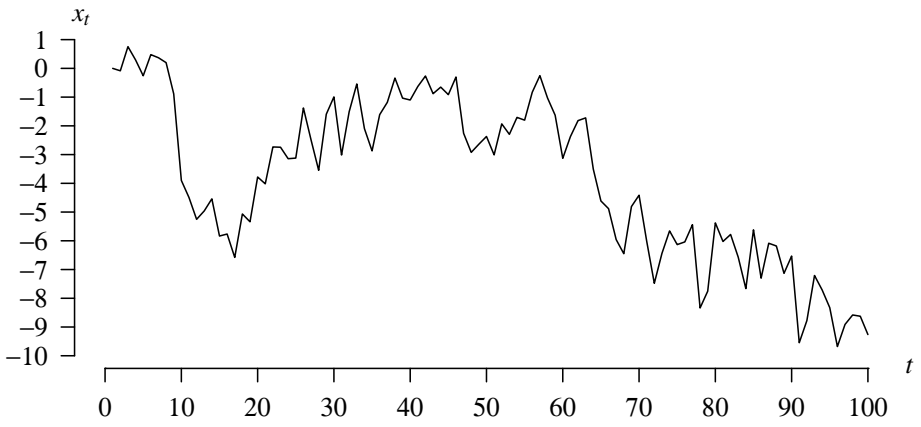

Figure 7.4: Time series plot of $n = 100$ observations from a random walk.

year 2000. The other two are random walks of length $n = 100$ with mutually independent standard normal noise terms generated by simulation. Spend some time looking the three graphs over and try to determine which of the three is the real time series values and which are the random walks generated in R via simulation. In order to make your task of identifying the Dow Jones Industrial Averages more difficult, the labels on the vertical axes have been suppressed, and the scales have been set to stretch from the smallest value to the largest value in the time series.

If you are having a difficult time identifying the stock market average values, it is because the random walk, which is a very simple time series model, is adequately approximating the time evolution of the stock market average values. The real data corresponding to the stock market average closes is in the top graph. The two lower graphs in Figure 7.5 are random walks generated by Monte Carlo simulation.

In this section, we have encountered four examples of time series:

- a time series of $n = 96$ monthly home energy consumption observations,
- a time series of $n = 144$ monthly international airline passenger counts,
- a time series of $n = 100$ observations of Gaussian white noise, and
- two time series of $n = 100$ observations generated from a random walk, which were compared to a time series consisting of $n = 100$ closing values for the Dow Jones Industrial Average.

R has built-in data structures and functions that are useful in the analysis of a time series. Additional tools beyond just the plotting of a time series will be introduced subsequently.

It is often the case that we want to formulate a hypothetical population probability model for a time series from observed values of a time series, such as using the random walk model to model the Dow Jones Industrial Average closing values. This notion of formulating a population probability model is completely analogous to using the normal distribution to approximate the adult heights of Swedish women, for example. Once a tentative model has been identified, any unknown parameters

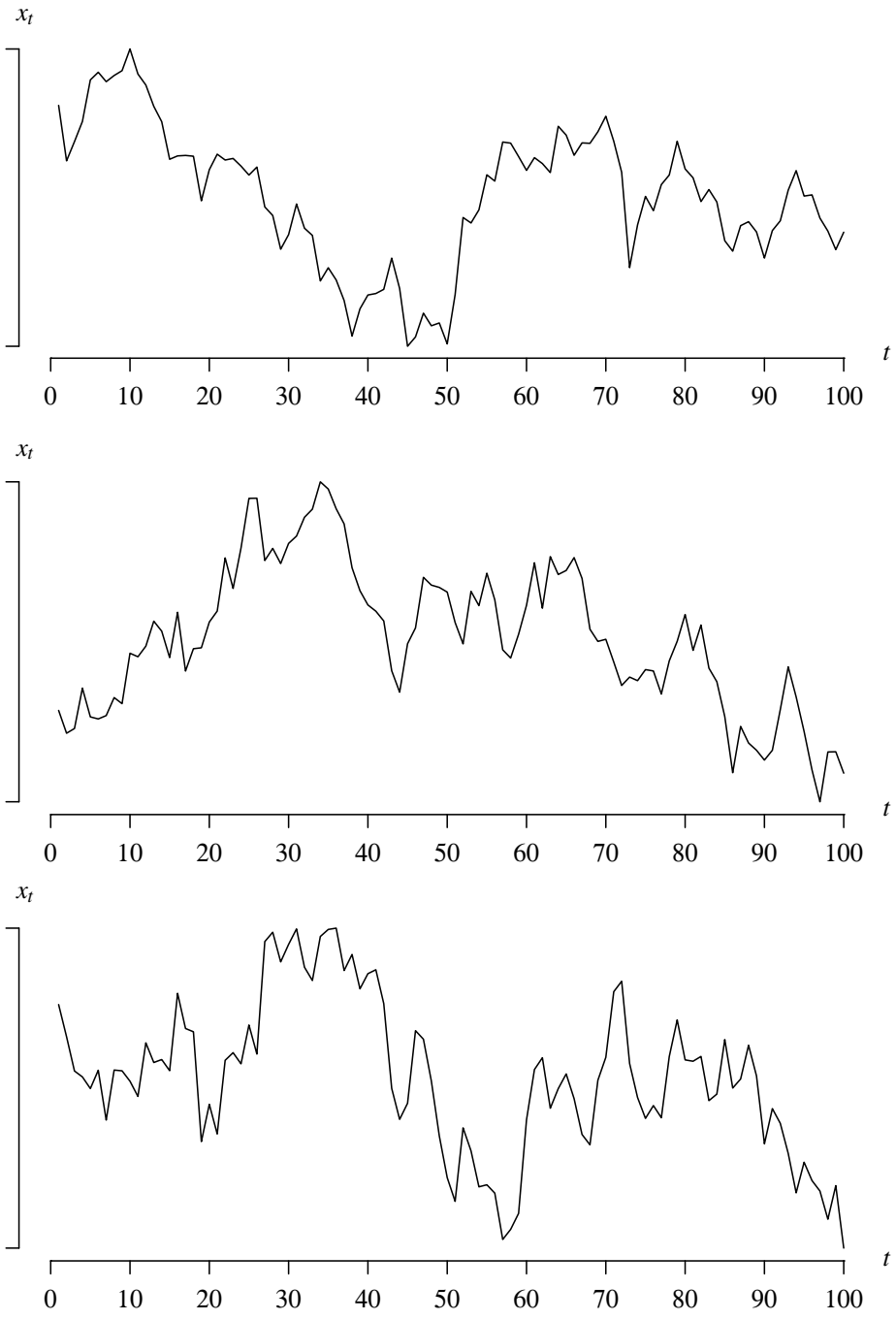

Figure 7.5: Which time series of length $n = 100$ consists of real data?

are estimated and goodness-of-fit tests are conducted. After a fitted probability model is accepted as a reasonable population probability model to describe a time series, a wide variety of statistical inference procedures can be employed. Broad categories of these statistical inference procedures are listed and described in the next subsection.

7.1.2 Why Analyze a Time Series?

There is not one universal purpose for conducting a time series analysis. Some of the more common purposes for conducting a time series analysis include description, explanation, prediction, simulation, control, signal estimation, and segmentation. This list is not comprehensive, but certainly covers the vast majority of the applications of time series analysis.

- **Description.** Time series analysis is often useful for *describing* the time evolution of the observations in a time series. Plotting the values in the time series, as we have done in the four examples, is a critical first step for observing trends, seasonal effects, extreme observations, etc. The time series plot also allows an analyst to easily identify outliers in a time series and decide whether these outliers were due to a coding error that occurred when inputting the time series or just random extreme observations. More sophisticated techniques that are helpful in describing a time series, such as the sample autocorrelation function to detect and quantify serial dependence in the values of a time series, or the periodogram to detect and quantify cyclic variation in the values of a time series, will be introduced subsequently.

- **Explanation.** It is often the case that one time series can be used to explain another time series. The home energy consumption time series from Example 7.1, for instance, might be partially explained by a time series of monthly average outdoor temperatures in Williamsburg, Virginia in 2011–2018. Another time series that might partially explain the home energy consumption values is the average number of hours of daylight in Williamsburg in a particular month.

- **Prediction.** Certain application areas, such as quantitative finance and seismology, engage in the analysis of a time series for the purpose of *forecasting*. The prediction of the next value of the time series, or perhaps the value of the time series h time units into the future, is often of interest. In quantitative finance, predicting the future value of a particular stock based on its history to date might be of interest. In seismology, predicting the time of the next earthquake might be of interest. Forecasted values are typically given by a point estimate and an associated confidence interval that measures the precision of the point estimate.

- **Simulation.** Simulating a time series in a discrete-event simulation might be the ultimate goal. A time series model that adequately mimics the real-world time series is critical in building a valid simulation model. As an example of such a simulation, financial planners often turn to simulation to estimate the probability that an individual or a married couple will have enough money to pay their expenses in retirement. This simulation requires, among other elements, a time series model that is capable of generating the annual inflation rate over the lifetimes of the individual or couple. The generation of simulated future annual inflation rates is based on building a time series model from previous annual inflation rates. Other elements, such as annual stock market returns or interest rates, would require a separate time series model. The values in these various time series are often correlated.

- **Control.** Time series analysis can be performed with the goal of controlling a particular variable. Examples include keeping ball bearing diameters between two prescribed thresholds,

keeping delivery times below a prescribed threshold, and keeping unemployment in an economy between two thresholds. A branch of quality control known as *statistical process control* refers to the time series plots given earlier as *control charts*.

- **Signal estimation.** Certain application areas, such as astrophysics and electrical engineering, are particularly interested in separating signal from noise. The techniques using spectral analysis, which is presented subsequently, are particularly adept at detecting cyclic variation in a time series. Sometimes a very weak signal can be detected in a very noisy time series using these techniques.

- **Segmentation.** Economists often find it useful to classify a period of economic activity as a period of expansion or a period of contraction. They do so by breaking a time series into a sequence of segments. The challenge here is to identify the boundary points at which times the economy switches from expansion to contraction and then back again. Determining these points in time in which the changes in the time series model occur is one of the goals of segmentation.

A time series is just one instance of a process that evolves randomly over time known as a *stochastic process*. The next section classifies stochastic processes based on whether time passes in a discrete or continuous fashion, and whether the variable of interest at each time step is discrete or continuous.

7.1.3 Where Does Time Series Analysis Fall in the Modeling Matrix?

The purpose of this subsection is to step back and consider where time series analysis fits in the larger arena of stochastic processes. The common elements between the four time series we have encountered so far are that time is measured as an integer (representing months for the first two time series, the first 100 positive integers for the Gaussian white noise, and trading days for the stock market averages) and the values of the time series observations are measured on a continuous scale. So a *time series* is a sequence of observed data values, measured on a continuous scale, which are collected over time. In most instances, the observations are taken at equally-spaced points in time. The observations in the time series are denoted generically by

$$X_1, X_2, \ldots, X_n,$$

and can be referred to more compactly as just $\{X_t\}$. Table 7.2 shows the position that time series analysis resides in the 2×2 table in which the nature of time (discrete or continuous) defines the rows and the nature of the observed variable (discrete or continuous) defines the columns. Time series analysis occupies the discrete-time, continuous-state entry in the table. Popular stochastic

		state	
		discrete	continuous
time	discrete	Markov chains	a time series model
	continuous	continuous-time Markov chains	Brownian motion

Table 7.2: Four types of stochastic models in the modeling matrix.

process models, such as Markov chains which are often used to model discrete-time and discrete-state stochastic processes, occupy the other three positions in the table.

The defining characteristic of a time series model is that time is measured on a discrete scale and the observations are measured on a continuous scale. You have seen four examples of time series. Here are examples of applications of stochastic models in the other three boxes in Table 7.2.

- The classic example of a discrete-time, discrete-state stochastic process with two states is the weather on a particular day. If the two states are "rainy" and "sunny" (actually "not rainy" should be the second state so as to partition the state space so that a cloudy day with no rain is classified as "not rainy"), then a Markov chain is a potential model for the evolution of the weather from one day to the next.

- A continuous-time, discrete-state stochastic process that we have all encountered is that of a single-server queueing system. The state of the system is the number of customers in the system. If the number of customers in the system can either go up by one (via a customer arrival) or down by one (via a customer departure), then the state of the system is discrete. Furthermore, since customer arrivals and departures can occur at any instant, time is measured on a continuous scale.

- One well-known example of a continuous-time, continuous-state stochastic process is *Brownian motion*, which is named after Scottish botanist Robert Brown (1753–1858). He described the motion in 1827 while observing the pollen of the plant Clarkia pulchella immersed in water through a microscope. Physicist Albert Einstein (1879–1955) explained that the motion of the pollen was caused by individual water molecules in 1905. Brownian motion can be thought of as a random walk in which the time between subsequent observations approaches zero.

Figure 7.6 contains a 2×2 array of graphs that are analogous to the 2×2 array of stochastic process models in Table 7.2. The values plotted are one particular *realization*, also known as a *sample path*, of a stochastic process. A stochastic process can be thought of as a probability model which evolves over time. The dashed lines indicate that time or state is measured discretely. In this sense, the techniques for analyzing a time series represent 25% of the techniques for analyzing all of the stochastic processes.

7.1.4 Computing

We review some of the R functions that have been used in this section and also introduce some additional functions that are useful in time series analysis. All of these functions are available in the base distribution of R, so they are immediately available upon initiating an R session. These functions will typically be illustrated here for the built-in time series of monthly international air passenger counts from 1949 to 1960 named `AirPassengers` which was first encountered in Example 7.2, but they could be applied to any time series.

The time series function `ts` is used to convert data to the internal time series data structure in R. It takes arguments for the time series observations, the time of the first observation, the time of the last observation, the number of observations per unit of time, etc. As an illustration, the quarterly time series observations contained in the vector named `data` which begin in the second quarter of 1984 is converted to a time series named `x` via the `ts` function with the R command

```
x = ts(data, start = c(1984, 2), frequency = 4)
```

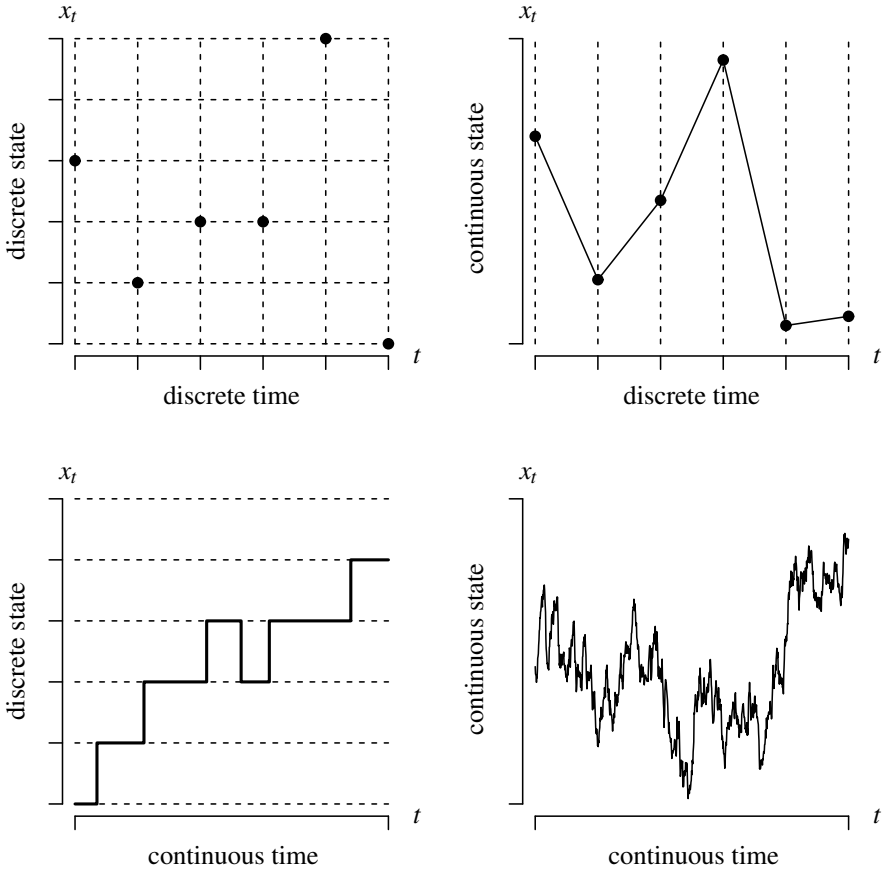

Figure 7.6: Graphs of realizations of four classes of stochastic processes.

Once a time series is in the R time series data structure, the `plot.ts` function can be used to provide a plot of the time series, for example,

```
plot.ts(AirPassengers)
```

The `ts.plot` command can be used to plot several time series on a single set of axes, for example,

```
ts.plot(ldeaths, mdeaths, fdeaths)
```

where the three time series that are built into R being plotted are monthly total deaths, male deaths, and female deaths from bronchitis, emphysema, and asthma in the United Kingdom from 1974 to 1979.

The next group of time series functions can be thought of as *utilities*, which are used to extract information about a time series. Illustrations of the application of these utility functions on the `AirPassengers` time series are given below.

```
length(AirPassengers)
```

```
start(AirPassengers)
end(AirPassengers)
frequency(AirPassengers)
deltat(AirPassengers)
time(AirPassengers)
cycle(AirPassengers)
```

The `length` function returns the length of a time series, which in this case is 144 observations. The `start` function returns the time associated with the first observation in the time series, which in this case is a vector comprised of the two elements `1949` and `1`. The `end` function returns the time associated with the last observation in the time series. The `frequency` function shows the period length of a time series, which in this case is 12 because `AirPassengers` consists of monthly data. The `deltat` function returns the time increment associated with the time series, which in this case is $1/12$. The `time` function returns the time values for each observation as a time series, which in this case is $1949, 1949\frac{1}{12}, 1949\frac{2}{12}, \ldots, 1960\frac{11}{12}$. The `cycle` function returns integers indicating the position of each observation in a cycle, which in this case is 12 iterations of the first 12 integers.

7.2 Basic Properties of a Time Series

A time series has some unique properties that will require some special tools to aid in its modeling and analysis. Central to these properties is the notion of *stationarity*, which is the subject of one of the subsections that follow. Once stationarity is established for a time series, then the population autocorrelation function, and its statistical counterpart, the sample autocorrelation function, can be helpful in characterizing a time series. These autocorrelation functions give the correlation as a function of the distance between the values in the time series. We begin by defining the population autocovariance and autocorrelation.

7.2.1 Population Autocovariance and Autocorrelation

Traditional statistical methods rely on the assumption that observations are mutually independent and identically distributed. In most practical time series applications, this assumption is violated because adjacent or nearby values in a time series are correlated. Thus, the special analysis tools for time series known as the population autocovariance function and the population autocorrelation function are introduced in this subsection. Before motivating and defining these new notions, we briefly review the definitions of population covariance and correlation from probability theory in the next paragraph. The link to time series analysis will be made subsequently.

The defining formula for the population covariance between the random variables X and Y is

$$\text{Cov}(X,Y) = E\left[(X-\mu_X)(Y-\mu_Y)\right],$$

where $\mu_X = E[X]$ is the expected value of X and $\mu_Y = E[Y]$ is the expected value of Y. The units on the population covariance are the units of X times the units of Y. When X and Y are independent random variables,

$$\text{Cov}(X,Y) = E\left[(X-\mu_X)(Y-\mu_Y)\right] = E\left[X-\mu_X\right]E\left[Y-\mu_Y\right] = (\mu_X-\mu_X)(\mu_Y-\mu_Y) = 0.$$

Zero covariance does not imply that X and Y are independent. The population correlation between the random variables X and Y is

$$\rho = \text{Corr}(X,Y) = \frac{\text{Cov}(X,Y)}{\sigma_X\sigma_Y} = \frac{E\left[(X-\mu_X)(Y-\mu_Y)\right]}{\sigma_X\sigma_Y} = E\left[\left(\frac{X-\mu_X}{\sigma_X}\right)\left(\frac{Y-\mu_Y}{\sigma_Y}\right)\right],$$

where σ_X is the population standard deviation of X and σ_Y is the population standard deviation of Y. The population correlation is a measure of the linear association between X and Y. The population correlation is unitless and satisfies $-1 \leq \rho \leq 1$, where the extremes indicate a perfect linear association.

A time series $\{X_t\}$ consists of a sequence of observations $X_1, X_2, \ldots, X_n$ indexed over time. Since we are working with time series *models* rather than time series data values, the time series values will typically be set in uppercase in this subsection. The observations are continuous random variables that have been drawn from some population probability distribution. This probability distribution can be described by a joint probability density function

$$f(x_1, x_2, \ldots, x_n)$$

or an associated joint cumulative distribution function

$$F(x_1, x_2, \ldots, x_n) = P(X_1 \leq x_1, X_2 \leq x_2, \ldots, X_n \leq x_n).$$

The most well-known probability distribution for modeling n random variables $X_1, X_2, \ldots, X_n$ is the *multivariate normal distribution*, which is illustrated in the next example.

Example 7.5 Consider an $n \times 1$ vector of random variables $\boldsymbol{X} = (X_1, X_2, \ldots, X_n)'$, an associated $n \times 1$ vector of population means $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_n)'$, and an associated $n \times n$ variance–covariance matrix $\boldsymbol{\Sigma}$. Matrix notation makes the expression of the joint probability density function of $X_1, X_2, \ldots, X_n$ much more compact than an entirely algebraic approach. The random vector $\boldsymbol{X}$ has the multivariate normal distribution if its joint probability density function has the form

$$f(x_1, x_2, \ldots, x_n) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})} \qquad (x_1, x_2, \ldots, x_n) \in \mathcal{A},$$

where

- $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)'$,
- $\boldsymbol{\Sigma}^{-1}$ is the inverse of the variance–covariance matrix,
- $|\boldsymbol{\Sigma}|$ is the determinant of the variance–covariance matrix,
- the support is

$$\mathcal{A} = \{(x_1, x_2, \ldots, x_n) \,|\, -\infty < x_i < \infty, \text{ for } i = 1, 2, \ldots, n\},$$

- the parameter space is

$$\Omega = \{(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \,|\, \boldsymbol{\mu} \in \mathcal{R}^n, \boldsymbol{\Sigma} \text{ is an } n \times n \text{ symmetric, positive semi-definite matrix}\}.$$

Although the multivariate normal distribution has some very appealing mathematical and statistical properties, it has one very significant drawback when it comes to being used as a time series model. That drawback concerns the number of parameters. There are n mean parameters $\mu_1, \mu_2, \ldots, \mu_n$ and $n(n+1)/2$ parameters in the symmetric variance–covariance matrix $\boldsymbol{\Sigma}$. If an analyst has collected just a single realization of a time series $x_1, x_2, \ldots, x_n$, then there are many more parameters to estimate than data values. So one of the goals for the rest of the section is to establish properties of a time series which allow us to formulate *parsimonious models* that adequately model a

particular time series with as few parameters as possible. We begin the process of establishing these properties by defining the *population mean function* and the *population autocovariance function* associated with a time series $\{X_t\}$. As you will see, the focus is on the first two population moments associated with the time series.

Definition 7.2 A time series $\{X_t\}$ has a *population mean function*

$$\mu(t) = E[X_t]$$

provided that the expected values exist for all values of the index t.

So as long as the observations in the time series have expected values that are finite, the population mean function gives the expected observed value of the time series at time t. In other words, the mean values

$$\mu(1), \mu(2), \ldots, \mu(n)$$

are the expected values of $X_1, X_2, \ldots, X_n$. This defines what is essentially the first moment of the time series. The second moment of the time series is defined by the population autocovariance function.

Definition 7.3 A time series $\{X_t\}$ has a *population autocovariance function*

$$\gamma(s,t) = \text{Cov}(X_s, X_t)$$

provided that the population covariances exist for all values of the indexes s and t.

Notice that the order associated with the arguments in the population autocovariance function is immaterial, so $\gamma(s,t) = \gamma(t,s)$. Notice also that when the two arguments in the population autocovariance function are identical, the expression reduces to the population variance, that is,

$$\gamma(s,s) = \text{Cov}(X_s, X_s) = V[X_s].$$

The prefix "auto" means "self." This prefix is attached to covariance to signify that the population covariance is being taken between two members of the same time series. The value of $\gamma(s,t)$ is the population covariance between two snapshots of the same time series at times s and t.

We now consider a sequence of three examples in which we (a) define a time series model, (b) calculate the population mean function, and (c) calculate the population autocovariance function. The three examples, which will be in order of increasing complexity, are

- white noise,
- a three-point moving average, and
- a random walk.

We begin with a process that consists of just white noise.

Example 7.6 Recall from Definition 7.1 that the time series $\{Z_t\}$ that is a sequence of mutually independent random variables $Z_1, Z_2, \ldots, Z_n$, each with population mean 0 and population variance σ_Z^2, is known as white noise, and is denoted by

$$Z_t \sim WN\left(0, \sigma_Z^2\right).$$

Assume that our time series of interest $\{X_t\}$ is just this white noise; that is, $X_t = Z_t$. Find the population mean function and the population autocovariance function.

The population mean function is

$$\mu(t) = E[X_t] = E[Z_t] = 0$$

because each term in the white noise time series has expected value 0. The population autocovariance function is

$$\gamma(s,t) = \text{Cov}(X_s, X_t) = \text{Cov}(Z_s, Z_t) = \begin{cases} \sigma_Z^2 & t = s \\ 0 & t \neq s \end{cases}$$

because the observations in the time series are mutually independent random variables and

$$\gamma(s,s) = \text{Cov}(X_s, X_s) = \text{Cov}(Z_s, Z_s) = V[Z_s] = \sigma_Z^2$$

when $t = s$.

So the population mean and population autocovariance functions take on a particularly tractable form in the case of a time series that consists of white noise terms. We now consider calculating the population mean and population autocovariance functions for a three-point moving average of white noise.

Example 7.7 We again let the time series $\{Z_t\}$ denote mutually independent random variables $Z_1, Z_2, \ldots, Z_n$, each with population mean 0 and population variance σ_Z^2. This is again white noise, and is denoted by

$$Z_t \sim WN\left(0, \sigma_Z^2\right).$$

This time, however, our time series of interest $\{X_t\}$ is a three-point moving average of the white noise, that is,

$$X_t = \frac{Z_{t-1} + Z_t + Z_{t+1}}{3}$$

for $t = 2, 3, \ldots, n-1$. Find the population mean function and the population autocovariance function.

The population mean function is

$$\mu(t) = E[X_t] = E\left[\frac{Z_{t-1} + Z_t + Z_{t+1}}{3}\right] = \frac{1}{3}\left(E[Z_{t-1}] + E[Z_t] + E[Z_{t+1}]\right) = 0$$

because each term in the white noise time series has expected value 0. The population autocovariance function is more difficult than in the previous example because identical white noise terms are used in adjacent three-point moving averages. Assuming that all appropriate expected values exist, we rely on the formula

$$\text{Cov}\left(\sum_{i=1}^{n} a_i X_i, \sum_{j=1}^{m} b_j Y_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{m} a_i b_j \text{Cov}(X_i, Y_j)$$

to help with the calculations. The population autocovariance function is

$$\begin{aligned}\gamma(s,t) &= \text{Cov}\,(X_s, X_t)\\ &= \text{Cov}\left(\frac{Z_{s-1}+Z_s+Z_{s+1}}{3}, \frac{Z_{t-1}+Z_t+Z_{t+1}}{3}\right)\\ &= \frac{1}{9}\text{Cov}\,(Z_{s-1}+Z_s+Z_{s+1}, Z_{t-1}+Z_t+Z_{t+1})\,.\end{aligned}$$

It is clear that $\gamma(s,t)=0$ when $|t-s|>2$ because there is no overlap in the white noise terms. The mutual independence of $Z_1, Z_2, \ldots, Z_n$ implies $\text{Cov}\,(Z_i, Z_j)=0$ when $i \neq j$. So let's check the other cases individually using the formula concerning the population covariance between sums of random variables. First, the case of $t=s$:

$$\begin{aligned}\gamma(s,s) &= \frac{1}{9}\text{Cov}\,(Z_{s-1}+Z_s+Z_{s+1}, Z_{s-1}+Z_s+Z_{s+1})\\ &= \frac{1}{9}\left[\text{Cov}\,(Z_{s-1},Z_{s-1})+\text{Cov}\,(Z_s,Z_s)+\text{Cov}\,(Z_{s+1},Z_{s+1})\right]\\ &= \frac{1}{9}\left(V\,[Z_{s-1}]+V\,[Z_s]+V\,[Z_{s+1}]\right)\\ &= \frac{1}{9}\left(\sigma_Z^2+\sigma_Z^2+\sigma_Z^2\right)\\ &= \frac{1}{9}\cdot 3\sigma_Z^2\\ &= \frac{\sigma_Z^2}{3}\end{aligned}$$

based on the mutual independence of $Z_1, Z_2, \ldots, Z_n$. Next, consider the case of $t=s+1$:

$$\begin{aligned}\gamma(s,s+1) &= \frac{1}{9}\text{Cov}\,(Z_{s-1}+Z_s+Z_{s+1}, Z_s+Z_{s+1}+Z_{s+2})\\ &= \frac{1}{9}\left[\text{Cov}\,(Z_s,Z_s)+\text{Cov}\,(Z_{s+1},Z_{s+1})\right]\\ &= \frac{1}{9}\left(V\,[Z_s]+V\,[Z_{s+1}]\right)\\ &= \frac{1}{9}\left(\sigma_Z^2+\sigma_Z^2\right)\\ &= \frac{1}{9}\cdot 2\sigma_Z^2\\ &= \frac{2\sigma_Z^2}{9}.\end{aligned}$$

Finally, consider the case of $t=s+2$:

$$\begin{aligned}\gamma(s,s+2) &= \frac{1}{9}\text{Cov}\,(Z_{s-1}+Z_s+Z_{s+1}, Z_{s+1}+Z_{s+2}+Z_{s+3})\\ &= \frac{1}{9}\text{Cov}\,(Z_{s+1},Z_{s+1})\\ &= \frac{1}{9}V\,[Z_{s+1}]\\ &= \frac{\sigma_Z^2}{9}.\end{aligned}$$

So to summarize, using the symmetry of the population autocovariance function in its arguments, the population autocovariance function is

$$\gamma(s,t) = \begin{cases} \sigma_Z^2/3 & |t-s| = 0 \\ 2\sigma_Z^2/9 & |t-s| = 1 \\ \sigma_Z^2/9 & |t-s| = 2 \\ 0 & |t-s| > 2. \end{cases}$$

The effect of using common terms from the time series Z_t consisting of white noise in constructing the three-point moving average time series X_t is apparent in the positive values in the population autocovariance function. There is positive population autocovariance at lag 0 ($t = s$), slightly weaker positive population autocovariance at lag 1 ($|t-s| = 1$), still slightly weaker positive population autocovariance at lag 2 ($|t-s| = 2$), and zero population autocovariance at lags greater than 2. The decreasing magnitude of the population autocovariance function is due to the fewer common terms in the three-point moving average as the distance between values in $\{X_t\}$ increases. The diagram in Figure 7.7 conveys the intuition associated with the values in the population autocovariance function. The brackets show the mutually independent values of the white noise terms $Z_1, Z_2, \ldots, Z_n$ used in each term in the three-point moving average time series. Terms in the three-point moving average that are three time units apart, such as X_2 and X_5, have no white noise terms in common, and hence have population autocovariance zero.

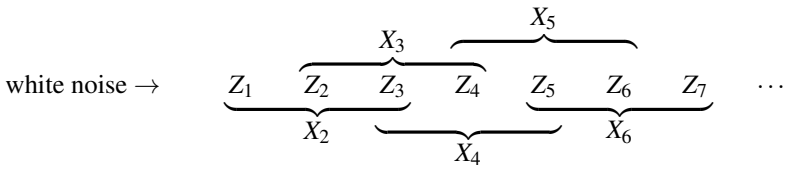

Figure 7.7: Relationship between white noise and three-point moving average.

The third and final example concerns the calculation of the population mean function and the population autocovariance function for a random walk.

Example 7.8 We now return to the random walk model first introduced in Example 7.4. The time series model for a random walk $\{X_t\}$ is the recursive equation

$$X_t = X_{t-1} + Z_t,$$

where $\{Z_t\}$ is white noise. A graph of a realization of a random walk was given in Figure 7.4. Find the population mean function and the population autocovariance function.

The first step is to write the model in a slightly different fashion. The random walk model can be written as a summation of the white noise terms:

$$X_t = \sum_{i=1}^{t} Z_i.$$

This formula can be verified by plugging it back into the original random walk model, yielding

$$\sum_{i=1}^{t} Z_i = \sum_{i=1}^{t-1} Z_i + Z_t.$$

This alternative formulation of the random walk model aids in the derivation of the population mean function and the population autocovariance function. Using the alternative formulation, the population mean function is

$$\mu(t) = E[X_t] = E\left[\sum_{i=1}^{t} Z_i\right] = \sum_{i=1}^{t} E[Z_i] = 0$$

because each term in the white noise time series has expected value 0. Again using the alternative formulation of the random walk model and the result from the previous example concerning the population covariance of sums of random variables, the population autocovariance function is

$$\begin{aligned}
\gamma(s,t) &= \text{Cov}(X_s, X_t) \\
&= \text{Cov}\left(\sum_{i=1}^{s} Z_i, \sum_{j=1}^{t} Z_j\right) \\
&= \sum_{i=1}^{s}\sum_{j=1}^{t} \text{Cov}(Z_i, Z_j) \\
&= \sum_{i=1}^{\min\{s,t\}} V[Z_i] \\
&= \min\{s,t\}\,\sigma_Z^2.
\end{aligned}$$

The population autocovariance function $\gamma(s,t)$ is the smaller of the arguments s and t multiplied by the population variance of the white noise.

The three examples have illustrated how to find the population mean function and the population autocovariance function for a time series model. Sometimes the population autocorrelation function is also of interest because population correlation is unitless and always lies between -1 and 1. The population autocorrelation function is defined next.

Definition 7.4 A time series $\{X_t\}$ has a *population autocorrelation function*

$$\rho(s,t) = \text{Corr}(X_s, X_t) = \frac{\text{Cov}(X_s, X_t)}{\sqrt{V[X_s]\,V[X_t]}} = \frac{\gamma(s,t)}{\sqrt{\gamma(s,s)\gamma(t,t)}}$$

provided that the population covariance exists for all indexes s and t.

We now revisit the previous three examples to compute the population autocorrelation function for the white noise, three-point moving average, and random walk models.

Example 7.9 The white noise model used

$$Z_t \sim WN\left(0, \sigma_Z^2\right)$$

and a time series $\{X_t\}$ which was just white noise, that is, $X_t = Z_t$. Find the population autocorrelation function.

The population autocovariance function for the white noise time series from Example 7.6 was

$$\gamma(s,t) = \begin{cases} \sigma_Z^2 & t = s \\ 0 & t \neq s. \end{cases}$$

Since $\gamma(s,s) = \sigma_Z^2$, the population autocorrelation function is

$$\rho(s,t) = \frac{\gamma(s,t)}{\sqrt{\gamma(s,s)\gamma(t,t)}} = \begin{cases} 1 & t = s \\ 0 & t \neq s. \end{cases}$$

There is perfect positive population correlation between each observation in the time series and itself because $\rho(s,s) = 1$. Furthermore, there is zero population correlation between distinct terms in the time series because $\rho(s,t) = 0$ for all $t \neq s$.

Although it lacks practical application in most real-world settings, the population autocorrelation function in the case of white noise is one of the most fundamental population autocorrelation functions possible. Since iid noise and Gaussian white noise are subsets of white noise, they also share this same population autocorrelation function. The next example considers the three-point moving average.

Example 7.10 Find the population autocorrelation function $\rho(s,t)$ for the three-point moving average time series model

$$X_t = \frac{Z_{t-1} + Z_t + Z_{t+1}}{3},$$

where

$$Z_t \sim WN\left(0, \sigma_Z^2\right).$$

The population autocovariance function for the three-point moving average time series from Example 7.7 was

$$\gamma(s,t) = \begin{cases} \sigma_Z^2/3 & |t-s| = 0 \\ 2\sigma_Z^2/9 & |t-s| = 1 \\ \sigma_Z^2/9 & |t-s| = 2 \\ 0 & |t-s| > 2. \end{cases}$$

Since $\gamma(s,s) = \sigma_Z^2/3$, the population autocorrelation function is

$$\rho(s,t) = \frac{\gamma(s,t)}{\sqrt{\gamma(s,s)\gamma(t,t)}} = \begin{cases} 1 & |t-s| = 0 \\ 2/3 & |t-s| = 1 \\ 1/3 & |t-s| = 2 \\ 0 & |t-s| > 2. \end{cases}$$

There is perfect positive population correlation between each observation in the time series and itself because $\rho(s,s) = 1$. The population autocorrelation function is positive and decreases linearly for lags 1 and 2 because of the common terms in the 3-point moving average, as illustrated previously in Figure 7.7. There is 0 population autocorrelation for lags of 3 or more because the moving averages contain no common white noise terms.

The third and final example concerns the calculation of the population autocorrelation function for a random walk model for a time series $\{X_t\}$.

Example 7.11 Find the population autocorrelation function $\rho(s,t)$ for the random walk time series model

$$X_t = X_{t-1} + Z_t,$$

where

$$Z_t \sim WN\left(0, \sigma_Z^2\right).$$

The population autocovariance function for the random walk time series from Example 7.8 was

$$\gamma(s,t) = \min\{s,t\}\,\sigma_Z^2.$$

Since $\gamma(s,s) = s\sigma_Z^2$, the population autocorrelation function is

$$\rho(s,t) = \frac{\gamma(s,t)}{\sqrt{\gamma(s,s)\gamma(t,t)}} = \frac{\min\{s,t\}\,\sigma_Z^2}{\sqrt{s\sigma_Z^2 \cdot t\sigma_Z^2}} = \frac{\min\{s,t\}}{\sqrt{st}}.$$

Since $\rho(s,s) = s/s = 1$, this can be written as

$$\rho(s,t) = \begin{cases} 1 & t = s \\ \min\{s,t\}/\sqrt{st} & t \neq s. \end{cases}$$

Once again, there is perfect positive population correlation between each observation in the time series and itself because $\rho(s,s) = 1$. This will be the case with any time series model.

This ends the introduction to three important functions that are associated with a time series model:

- the population mean function $\mu(t) = E[X_t]$,
- the population autocovariance function $\gamma(s,t) = \mathrm{Cov}(X_s, X_t)$, and
- the population autocorrelation function $\rho(s,t) = \mathrm{Corr}(X_s, X_t)$.

An important property of a time series, known as *stationarity*, will be defined and illustrated in the next subsection. A stationary time series is one in which there is no long-term change in the probability mechanism governing the time series. Knowing that a time series is stationary will have an important effect on $\mu(t)$, $\gamma(s,t)$, and $\rho(s,t)$.

7.2.2 Stationarity

A time series $\{X_t\}$ is stationary if the underlying probability mechanism that governs the time series is independent of a shift in time. In other words, if you select two different time windows in which to view a number of observations from the time series, the probability distribution of the observations in those two time windows will be identical.

Definition 7.5 The time series $\{X_t\}$ is *strictly stationary* if

$$X_1, X_2, \ldots, X_n$$

and the shifted observations in the time series

$$X_{k+1}, X_{k+2}, \ldots, X_{k+n}$$

have the same joint probability distribution for all integers k and all positive integers n.

A strictly stationary time series is also known as a *strongly stationary* or *completely stationary* time series. The next two examples contain the type of thought experiment that is appropriate for determining whether a time series is strictly stationary.

Example 7.12 Strict stationarity implies that the probability mechanism that governs the time series does not change with a shift in time. Would the time series of monthly international airline passengers (in thousands) contained in the `AirPassengers` time series in R be likely to have been drawn from a strictly stationary time series model?

Here is the thought associated with making such a judgment. Consider a specific instance of the time series from Definition 7.5 with $n = 3$ and $k = 18$ in order to develop a counterexample. So the question is whether

$$X_1, X_2, X_3$$

and the shifted observations in the time series

$$X_{19}, X_{20}, X_{21}$$

have the same trivariate probability distribution. In the case of the `AirPassengers` observed time series, these values correspond to January, February, and March of 1949 versus July, August, and September of 1950. The first three values in the time series are

$$x_1 = 112, x_2 = 118, x_3 = 132,$$

and the three time series observations shifted 18 months into the future are

$$x_{19} = 170, x_{20} = 170, x_{21} = 158.$$

From a cursory inspection, the three earlier values in the time series appear to be less than the three later values. In addition, Figure 7.2 showed a significant upward trend in the time series as time progresses. Furthermore, a careful inspection of the values in the `AirPassengers` time series from Figure 7.2 reveals that the annual peak travel occurs during the months of July and August. Based on this evidence, we conclude that the `AirPassengers` time series is *not* drawn from a strictly stationary time series. The hypothesis of an underlying stationary time series model can be rejected because of the trend and seasonal component that are clearly apparent in Figure 7.2. The underlying probability mechanism governing the time series appears to be changing over time.

The discussion above would indicate that very few time series which occur in practice would be strictly stationary. The previous example asks whether a *realization* appears to be drawn from a stationary time series model. The next example gives a simple time series model which is strictly stationary.

Example 7.13 Consider the time series from Example 7.3 which consists of Gaussian white noise with population variance $\sigma_Z^2 = 1$, that is,

$$X_t \sim GWN(0, 1).$$

Is this time series strictly stationary?

As in the last example, consider $n = 3$ and $k = 18$ from Definition 7.5. If the time series is strictly stationary, then

$$X_1, X_2, X_3$$

and the shifted observations in the time series

$$X_{19}, X_{20}, X_{21}$$

have the same trivariate probability distribution. In the case of Gaussian white noise with $\sigma_X^2 = \sigma_Z^2 = 1$, $(X_1, X_2, X_3)'$ has a trivariate normal distribution with 3×1 vector of population means $\boldsymbol{\mu} = (0, 0, 0)'$, and an associated 3×3 variance–covariance matrix which is the identity matrix. Using the formulation from Example 7.5, the joint probability density function of X_1, X_2, X_3 is

$$f(x_1, x_2, x_3) = \frac{1}{(2\pi)^{3/2}} e^{-\left(x_1^2 + x_2^2 + x_3^2\right)/2} \qquad -\infty < x_1 < \infty, -\infty < x_2 < \infty, -\infty < x_3 < \infty.$$

Because the values in the time series model are mutually independent and identically distributed, this is also the joint probability density function of X_{19}, X_{20}, X_{21}. So for this particular choice of n and k, the conditions of Definition 7.5 are satisfied. The probability mechanism governing X_1, X_2, X_3 is exactly the same as the probability mechanism governing X_{19}, X_{20}, X_{21}, so the realization of such a process in Figure 7.3 displays no trend, no seasonality, no change in variability, and no change in the marginal distributions of $X_1, X_2, \ldots, X_n$. But the choices of $n = 3$ and $k = 18$ were arbitrary. The joint probability distributions would be identical regardless of the choices for n and k, so we conclude that a time series consisting of Gaussian white noise is strictly stationary.

So the international airline passengers data set, just from observing the time series, is not strictly stationary. The Gaussian white noise process is strictly stationary. There are several implications of a strictly stationary time series, some of which are listed below.

- The initial n values of the time series $X_1, X_2, \ldots, X_n$ and their associated observations shifted k time units to the left or right $X_{k+1}, X_{k+2}, \ldots, X_{k+n}$ having the same joint probability distribution implies that each must have the same joint cumulative distribution function, that is,

  $$P(X_1 \le x_1, X_2 \le x_2, \ldots, X_n \le x_n) = P(X_{k+1} \le x_1, X_{k+2} \le x_2, \ldots, X_{k+n} \le x_n)$$

  for all values of $x_1, x_2, \ldots, x_n$.

- The marginal distribution of each value in the time series is identical. Symbolically,

  $$P(X_s \le x) = P(X_t \le x)$$

  for all integer time values s and t and all real-valued x.

- The population mean function $\mu(t) = E[X_t]$ is constant in time; that is, there is no trend.

- The population autocovariance function $\gamma(s,t) = \text{Cov}(X_s, X_t)$ is constant with respect to a shift; that is,
$$\gamma(s,t) = \gamma(s+k, t+k).$$
- Any time series consisting of mutually independent and identically distributed random variables must be strictly stationary.

Strict stationarity is a lot to ask of a time series, and is difficult to establish based on an observed realization of a time series. So time series analysts have defined a weaker version of strict stationarity which we will refer to here as just stationarity. Other terms used for this type of stationarity are

- weakly stationary,
- second-order stationary, and
- covariance stationary.

Whereas a strictly stationary time series required that the entire multivariate distribution remain the same on any time window, a stationary time series only places requirements on the first and second moments. The population mean function must be constant over time, and the population autocovariance function must depend only on the lag between the observations.

Definition 7.6 A time series $\{X_t\}$ is said to be *stationary* if the following two conditions are satisfied.

(a) The population mean function $\mu(t) = E[X_t]$ exists and is constant in t; that is, there is a real-valued constant c such that $E[X_t] = c$ for all values of t.

(b) The population autocovariance function $\gamma(s,t) = \text{Cov}(X_s, X_t)$ exists and depends only on $|t-s|$; that is, for integers s_1, t_1, s_2, and t_2,
$$\gamma(s_1, t_1) = \gamma(s_2, t_2)$$
if $|t_1 - s_1| = |t_2 - s_2|$.

The first condition implies that the time series has no trend because each observation in the time series has the same expected value. The second condition implies that the population covariance between two observations is a function of only the absolute difference between the two time indexes of the observations. This second condition implies that a stationary time series model only requires a single argument, which is known as the lag k, when defining the population autocovariance and autocorrelation function. We will use the same names for these functions, but only use a single argument when the time series is stationary.

Definition 7.7 For a stationary time series $\{X_t\}$, the population autocovariance function is
$$\gamma(k) = \text{Cov}(X_t, X_{t+k}),$$
and the population autocorrelation function is
$$\rho(k) = \text{Corr}(X_t, X_{t+k}) = \frac{\gamma(k)}{\gamma(0)}$$
for $k = 0, \pm 1, \pm 2, \ldots$.

The population autocorrelation function for a stationary time series provides an important reflection of the structure of a time series model. It gives the modeler a view of how observations in the time series are correlated based on their distance away from one another in the time series. The next example relates the population *correlation matrix* and the population autocorrelation function for a time series model.

Example 7.14 Consider the stationary time series $\{X_t\}$ with population correlation matrix

$$\begin{bmatrix} 1.0 & 0.4 & -0.2 & 0.1 \\ 0.4 & 1.0 & 0.4 & -0.2 \\ -0.2 & 0.4 & 1.0 & 0.4 \\ 0.1 & -0.2 & 0.4 & 1.0 \end{bmatrix}.$$

There is a population and a sample version of this matrix, but we will refer to this matrix as a population correlation matrix. This particular matrix is a special population correlation matrix because it corresponds to a stationary time series with equal-valued elements that are equal distance from the diagonal. The lag 2 population autocorrelations, for example, are all -0.2, and are two positions away from the diagonal elements in the population correlation matrix. The population autocorrelation function values for lags 4 and higher are all zero. Some notes on the population correlation matrix for a stationary time series model are given below.

- The population correlation matrix is symmetric and positive definite, with ones on the diagonal, and identical elements at a fixed number of entries from the diagonal.
- The population correlation between adjacent observations in the time series is given by the elements that are just off of the diagonal.
- This particular population correlation matrix has positive eigenvalues $\lambda_1 = 1.5$, $\lambda_2 = 1.3685$, $\lambda_3 = 1$, and $\lambda_4 = 0.1315$, which is consistent with the matrix being positive definite.

Convert this population correlation matrix to a population autocorrelation function.

Figure 7.8 shows the population correlation matrix rotated 45° clockwise. With this rotation, the identical elements are now aligned vertically. The vertical dashed lines show how the elements of the matrix are translated to a population autocorrelation function, which has nonzero spikes at lags $k = -3, -2, \ldots, 3$. The associated population autocorrelation function is symmetric. As expected, the lag k population autocorrelation satisfies $-1 \leq \rho(k) \leq 1$ for $k = 0, \pm 1, \pm 2, \ldots$. There is no information conveyed by including the population autocorrelation values for negative values of k. It is convention in time series analysis that the spike of height 1 associated with lag $k = 0$ is included in the graph of the population autocorrelation function. Furthermore, we always extend the vertical axis from -1 to 1 so that all population autocorrelation functions are viewed on an equal footing. Figure 7.9 shows the format that we will use for the plot of the population autocorrelation function for nonnegative lags k from this point forward.

You might have noticed that the word *population* precedes autocorrelation function. This convention is not universal, but we do so in order to distinguish the population autocorrelation function from its statistical counterpart, the *sample* autocorrelation function. An analogy in the realm of univariate probability distributions is the distinction between the population mean μ, which is a constant, and its statistical counterpart, the sample mean $\bar{X}$, which is a random variable. In the same

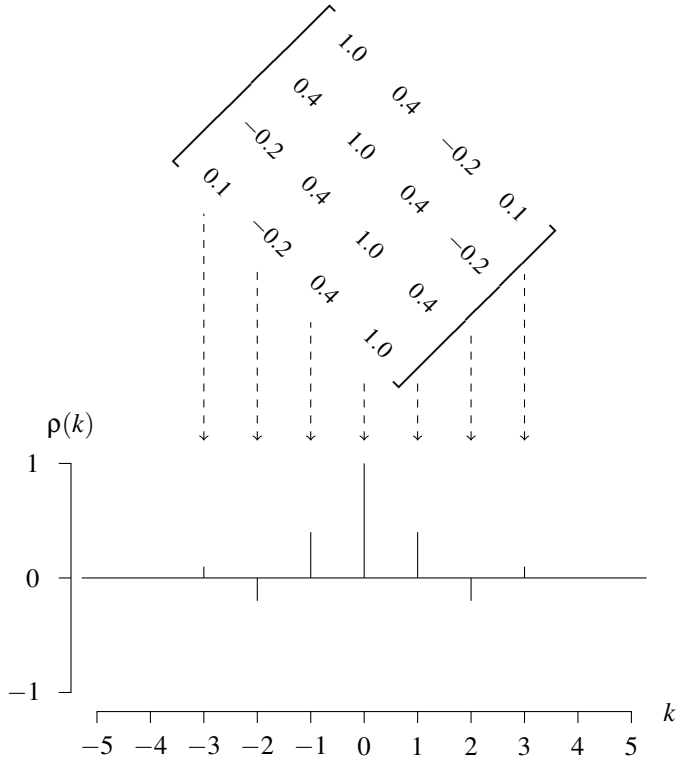

Figure 7.8: Mapping a correlation matrix to an autocorrelation function.

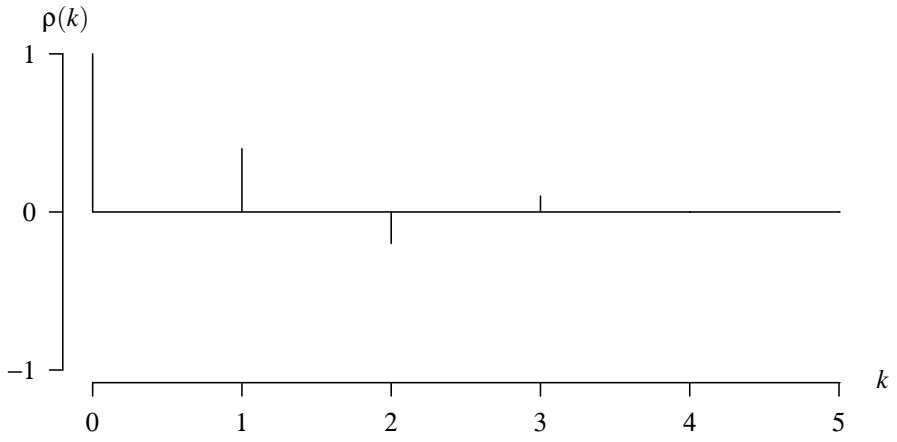

Figure 7.9: Population autocorrelation function for nonnegative lag values.

sense, the population autocorrelation function $\rho(k)$ is a sequence of population correlations, which are fixed constants that are indexed by the lag k. The sample autocorrelation function, which will be introduced in a subsequent section as r_k, is a sequence of sample correlations, which are random variables that are indexed by the lag k.

Several properties of the population autocorrelation function for *all* stationary time series models are given next.

Theorem 7.1 For a stationary time series $\{X_t\}$ with population mean μ and population autocorrelation function $\rho(k)$,

- $\rho(0) = 1$,
- $-1 \leq \rho(k) \leq 1$ for $k = 0, \pm 1, \pm 2, \ldots$,
- $\rho(k) = \rho(-k)$ for $k = 0, 1, 2, \ldots$,
- $\rho(k)$ is unitless, and
- $\rho(k)$ does not uniquely determine an underlying time series model.

Proof Consider a stationary time series $\{X_t\}$ with mean μ, population autocovariance function $\gamma(k)$, and population autocorrelation function $\rho(k)$.

- The lag zero population autocorrelation is $\rho(0) = 1$ because

$$\rho(0) = \text{Corr}(X_t, X_t) = \frac{\text{Cov}(X_t, X_t)}{\sigma_{X_t}\sigma_{X_t}} = \frac{V[X_t]}{\sigma_{X_t}^2} = \frac{\gamma(0)}{\gamma(0)} = 1.$$

- The lag k population autocorrelation must lie on the closed interval $[-1, 1]$. In other words, $-1 \leq \rho(k) \leq 1$, because $\rho(k)$ is defined as a population correlation. This can also be proved by first principles as follows. The inequality

$$V[c_1 X_t + c_2 X_{t+k}] \geq 0$$

holds for any real-valued constants c_1 and c_2 because all variances are nonnegative. This is equivalent to

$$c_1^2 V[X_t] + c_2^2 V[X_{t+k}] + 2c_1 c_2 \text{Cov}(X_t, X_{t+k}) \geq 0.$$

or

$$\left(c_1^2 + c_2^2\right)\sigma_{X_t}^2 + 2c_1 c_2 \gamma(k) \geq 0.$$

When $c_1 = c_2 = 1$, this inequality reduces to $\sigma_{X_t}^2 + \gamma(k) \geq 0$, which implies that $\rho(k) = \gamma(k)/\sigma_{X_t}^2 \geq -1$. Similarly, when $c_1 = 1$ and $c_2 = -1$, the inequality reduces to $\sigma_{X_t}^2 - \gamma(k) \geq 0$, which implies that $\rho(k) = \gamma(k)/\sigma_{X_t}^2 \leq 1$. Combining these two inequalities gives $-1 \leq \rho(k) \leq 1$.

- Since the time series $\{X_t\}$ is stationary,

$$\rho(k) = \text{Corr}(X_t, X_{t+k}) = \text{Corr}(X_{t-k}, X_t) = \rho(-k)$$

for $k = 0, 1, 2, \ldots$.

- The lag k population autocorrelation is unitless because the units of the numerator of
$$\rho(k) = \frac{\text{Cov}(X_t, X_{t+k})}{\sigma_{X_t}\sigma_{X_{t+k}}}$$
are the square of the units of X_t, and the units of both σ_{X_t} and $\sigma_{X_{t+k}}$ are the units of X_t. Thus, the units cancel and $\rho(k)$ is unitless.
- This final property can be proved by counterexample. Consider two time series models. The first is $X_t \sim GWN(0, 1)$ (that is, Gaussian white noise with $\sigma_{X_t} = 1$). The second is $X_t \sim IID(0, 1)$, for example, iid noise with $\sigma_{X_t} = 1$ and error terms $U\left(-\sqrt{3}, \sqrt{3}\right)$. These two time series models have identical population autocorrelation functions but are not identical time series models. Therefore, $\rho(k)$ does not uniquely determine an underlying time series model. □

These properties of $\rho(k)$ have important implications in time series analysis. The first result from Theorem 7.1 indicates that there is perfect positive population autocorrelation between an observation and itself (that is, an observation at lag $k = 0$). The initial spike in the population autocorrelation function at $\rho(0) = 1$ is generally included in a graph of the population autocorrelation function, although it conveys no information. The second result indicates that all population autocorrelation functions must lie between -1 and 1. Subsequent plots of $\rho(k)$ will always stretch the vertical axis from -1 to 1 so that they can easily be compared with one another. The third result indicates that $\rho(k)$ is an even function in k, so although k can be any integer, it is common practice to only graph $\rho(k)$ for $k = 0, 1, 2, \ldots$ because we know that the reflection about the $\rho(k)$ axis is identical. There is no need to graph the population autocorrelation function for negative lags because no additional information is conveyed. The fourth result explains why $\rho(k)$ tends to be more popular than $\gamma(k)$ because it is free of the units selected for X_t. The fifth result indicates that a time series model cannot be determined from its population autocorrelation function. Every stationary time series model has a population autocorrelation function, but knowing the autocorrelation function does not necessarily determine the underlying time series model.

We can now revisit the three examples from the previous subsection, namely white noise, a three-point moving average, and a random walk, to see if they are stationary time series models. In addition, we will make a plot of their population autocorrelation functions if they happen to be stationary.

Example 7.15 Consider the white noise time series model

$$Z_t \sim WN\left(0, \sigma_Z^2\right),$$

and the time series of interest is just $\{X_t\} = \{Z_t\}$. Determine whether this time series model is stationary, and plot the population autocorrelation function if it is stationary.

Recall from Example 7.6 that the population mean function for the white noise time series was

$$\mu(t) = 0$$

for all values of t, so the first condition of Definition 7.6 is satisfied. Recall also that the population autocovariance function was

$$\gamma(s,t) = \begin{cases} \sigma_Z^2 & t = s \\ 0 & t \neq s. \end{cases}$$

Since the value of $\gamma(s,t)$ depends only on $|t-s|$, the second condition of Definition 7.6 is satisfied, and we conclude that this time series model is stationary. Because the time series model is stationary, the population autocovariance function can be written in terms of the single argument k, the lag, as

$$\gamma(k) = \begin{cases} \sigma_Z^2 & k = 0 \\ 0 & k = 1, 2, \ldots . \end{cases}$$

Since $\gamma(0) = \sigma_Z^2$, the population autocorrelation function written in terms of the lag k is

$$\rho(k) = \begin{cases} 1 & k = 0 \\ 0 & k = 1, 2, \ldots . \end{cases}$$

It would be perfectly reasonable to consider the range of k to be $k = 0, \pm 1, \pm 2, \ldots$, but Theorem 7.1 indicates that the population autocorrelation function for a stationary time series model is *always* an even function, so we will only report the nonnegative values of k. A graph of $\rho(k)$ for the white noise process is shown in Figure 7.10. A horizontal line has been drawn at $\rho(k) = 0$ for reference. There is a single spike of height 1 at lag $k = 0$ which indicates that each observation is perfectly positively correlated with itself. There are spikes of height 0 at $k = 1, 2, \ldots$, which indicates that distinct observations in the time series are uncorrelated, as expected from the time series model consisting of white noise values.

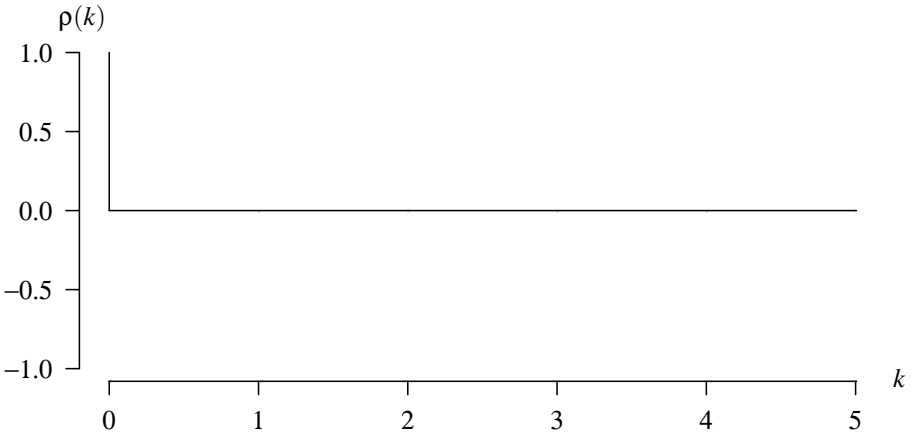

Figure 7.10: Population autocorrelation function for a white noise time series.

The population autocorrelation function for the white noise time series model is identical to that for iid noise and Gaussian white noise because those time series models are subsets of the white noise time series model. We now consider the three-point moving average model.

Example 7.16 Consider the white noise time series model

$$Z_t \sim WN\left(0, \sigma_Z^2\right),$$

and the time series of interest $\{X_t\}$ is the three-point moving average of the white noise; that is,

$$X_t = \frac{Z_{t-1} + Z_t + Z_{t+1}}{3}.$$

Determine whether this time series model is stationary, and plot the population autocorrelation function if it is stationary.

Recall from Example 7.7 that the population mean function for the white noise time series was

$$\mu(t) = 0$$

for all values of t, so the first condition of Definition 7.6 is satisfied. Recall also that the population autocovariance function was

$$\gamma(s,t) = \begin{cases} \sigma_Z^2/3 & |t-s| = 0 \\ 2\sigma_Z^2/9 & |t-s| = 1 \\ \sigma_Z^2/9 & |t-s| = 2 \\ 0 & |t-s| > 2. \end{cases}$$

Since the value of $\gamma(s,t)$ depends only on $|t-s|$, the second condition of Definition 7.6 is satisfied, and we conclude that this time series model is stationary. Because the time series model is stationary, the population autocovariance function can be written in terms of the single argument k, the lag, as

$$\gamma(k) = \begin{cases} \sigma_Z^2/3 & k = 0 \\ 2\sigma_Z^2/9 & k = 1 \\ \sigma_Z^2/9 & k = 2 \\ 0 & k = 3, 4, \ldots . \end{cases}$$

Since $\gamma(0) = \sigma_Z^2/3$, the population autocorrelation function written in terms of the lag k is

$$\rho(k) = \begin{cases} 1 & k = 0 \\ 2/3 & k = 1 \\ 1/3 & k = 2 \\ 0 & k = 3, 4, \ldots . \end{cases}$$

A graph of $\rho(k)$ for the three-point moving average model is shown in Figure 7.11. As with all population autocorrelation functions, there is a spike of height 1 at lag $k = 0$, which indicates that each observation is perfectly positively correlated with itself. The spikes at $k = 1$ and $k = 2$ reflect the effect of the nearby moving averages being functions of common white noise observations. Observations in $\{X_t\}$ that are three or more indexes apart are uncorrelated because they do not contain any common white noise terms. This corresponds to $\rho(k) = 0$ for $k = 3, 4, \ldots$.

The previous example concerning a three-point moving average of white noise generalizes to an m-point moving average of white noise, where m is an odd, positive integer. The more general time series model is also stationary, and the population autocorrelation function also decreases linearly, and cuts off at lag m. The derivation of this result is given as an exercise at the end of this chapter. The third example considers a random walk time series model.

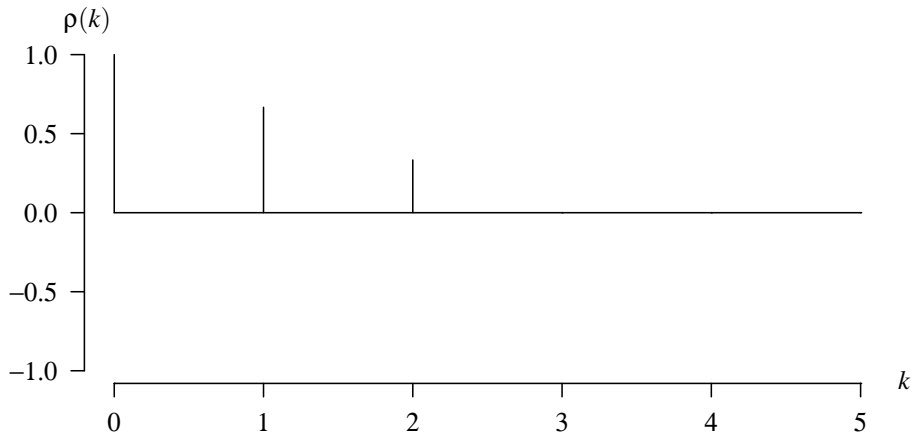

Figure 7.11: Population autocorrelation function for a three-point moving average time series.

Example 7.17 Consider the white noise time series model

$$Z_t \sim WN\left(0, \sigma_Z^2\right),$$

and the time series of interest $\{X_t\}$ is the random walk model; that is,

$$X_t = X_{t-1} + Z_t.$$

Determine whether this time series model is stationary, and plot the population autocorrelation function if it is stationary.

Recall from Example 7.8 that the population mean function for the white noise time series was

$$\mu(t) = 0$$

for all values of t, so the first condition of Definition 7.6 is satisfied. Recall also that the population autocovariance function was

$$\gamma(s,t) = \min\{s,t\}\,\sigma_Z^2.$$

Since the value of $\gamma(s,t)$ does *not* depend only on $|t-s|$, the second condition of Definition 7.6 is *not* satisfied, so we conclude that this time series model is nonstationary. Because the time series model is nonstationary, we are not able to write the population autocovariance function in terms of the single argument k. An example of the population autocorrelation function not being a function of the lag is

$$\gamma(1,4) = \sigma_Z^2 \qquad \text{and} \qquad \gamma(2,5) = 2\sigma_Z^2.$$

Equivalently,

$$\text{Cov}\,(X_1, X_4) = \sigma_Z^2 \qquad \text{and} \qquad \text{Cov}\,(X_2, X_5) = 2\sigma_Z^2.$$

Since observations that are three time indexes apart have different values of the population autocovariance function, $\gamma(s,t)$ does not depend only on $|t-s|$.

The statistical analogs of the population autocovariance and autocorrelation functions are the sample autocovariance and autocorrelation functions, which are calculated from an observed time series from a stationary model. These two functions are defined in the next subsection.

7.2.3 Sample Autocovariance and Autocorrelation

This section takes up the estimation of the population autocovariance function and the population autocorrelation function from a single realization of a time series denoted by the observations $x_1, x_2, \ldots, x_n$. In addition to the vital plot of a time series, a plot of the sample autocorrelation function, which is known as the *correlogram*, can yield additional insight concerning the underlying probability model governing the time series. The approach that we will take here is to review the sample versions of the covariance and correlation in terms of data pairs in the next paragraph, and then adapt these notions to their associated analogs in time series analysis.

This paragraph reviews the estimation of the population covariance and correlation from a data set of data pairs (X_i, Y_i), for $i = 1, 2, \ldots, n$. The population covariance is estimated by the *sample covariance*

$$\widehat{\text{Cov}}(X, Y) = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y}),$$

where $\bar{X}$ and $\bar{Y}$ are the sample means of the associated sample values:

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i \qquad \text{and} \qquad \bar{Y} = \frac{1}{n} \sum_{i=1}^{n} Y_i.$$

This formula is the statistical analog to the formula

$$\text{Cov}(X, Y) = E[(X - \mu_X)(Y - \mu_Y)]$$

from probability theory. There are two formulas for estimating the population variance from a random sample—one with n in the denominator and one with $n - 1$ in the denominator. Since n is required to be fairly large in time series analysis, the choice between the two is not critical. The formula with $n - 1$ in the denominator is more prevalent in statistics because

$$E\left[\frac{1}{n-1} \sum_{i=1}^{n} (X_i - \bar{X})^2\right] = \sigma_X^2$$

for mutually independent and identically distributed observations $X_1, X_2, \ldots, X_n$; that is, the sample variance S^2 is an unbiased estimator of the population variance σ_X^2. We use n in the denominator here because, in spite of being a biased estimator of the population variance in the non-times-series setting, it leads to certain terms dropping out of a subsequent formula. The population variances can be estimated by the maximum likelihood estimators

$$\hat{\sigma}_X^2 = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})^2 \qquad \text{and} \qquad \hat{\sigma}_Y^2 = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \bar{Y})^2.$$

An estimate for the population correlation ρ is given by the *sample correlation*

$$r = \hat{\rho} = \frac{\widehat{\text{Cov}}(X, Y)}{\hat{\sigma}_X \hat{\sigma}_Y} = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\left[\sum_{i=1}^{n} (X_i - \bar{X})^2\right]\left[\sum_{i=1}^{n} (Y_i - \bar{Y})^2\right]}}.$$

Notice that the $1/n$ terms in the numerator and the denominator cancel. Had a denominator of $n-1$, rather than n, been used in the formulas for $\widehat{\text{Cov}}(X,Y)$, $\hat{\sigma}_X^2$, and $\hat{\sigma}_Y^2$, the same cancellation would occur. Table 7.3 summarizes the results from Section 7.2.1 and this paragraph.

	covariance	correlation
population	$E\left[(X-\mu_X)(Y-\mu_Y)\right]$	$\dfrac{E\left[(X-\mu_X)(Y-\mu_Y)\right]}{\sigma_X \sigma_Y}$
sample	$\dfrac{1}{n}\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})$	$\dfrac{\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\left[\sum_{i=1}^{n}(X_i-\bar{X})^2\right]\left[\sum_{i=1}^{n}(Y_i-\bar{Y})^2\right]}}$

Table 7.3: Population and sample covariance and correlation.

We now translate the concepts from the previous paragraph into the context of the analysis of a time series. Consider the estimation of $\gamma(k)$ and $\rho(k)$ from a realization of observations $x_1, x_2, \ldots, x_n$, which are assumed to be observed values from a stationary time series model. The *lag k sample autocovariance*, which estimates $\gamma(k)$, is

$$c_k = \frac{1}{n}\sum_{t=1}^{n-k}(x_t-\bar{x})(x_{t+k}-\bar{x}),$$

where $\bar{x}$ is the sample mean of the observations in the time series. This is not a universal choice for c_k. Since there are $n-k$ terms in the summation, some time series analysts prefer to divide by $n-k$ rather than n. Because of the two different options for the denominator, it is important to only calculate c_k for k values that are significantly smaller than n. Generally speaking, there should be at least 60 to 70 observations in a time series to use the techniques described here. Having a large value of n means that having n or $n-k$ in the denominator is not critical for small values of k. The units on c_k are the square of the units of the observations in the time series. Notice that when $k=0$, the lag 0 sample autocovariance reduces to

$$c_0 = \frac{1}{n}\sum_{t=1}^{n}(x_t-\bar{x})^2,$$

which is an estimate for $\gamma(0)=\sigma_X^2$. The *lag k sample autocorrelation*, which estimates $\rho(k)$, is

$$r_k = \frac{c_k}{c_0}$$

for integer values of k which are significantly smaller than n. As was the case with $\rho(k)$, the lag k sample autocorrelation is a unitless quantity. When $k=0$, $r_0 = c_0/c_0 = 1$, as desired. The notation developed here to calculate $\gamma(k)$ and $\rho(k)$ for a stationary time series model and to estimate these functions with c_k and r_k for an observed time series $x_1, x_2, \ldots, x_n$ is summarized in Table 7.4.

	lag k autocovariance	lag k autocorrelation
population	$\gamma(k) = E\left[(X_t - \mu_X)(X_{t+k} - \mu_X)\right]$	$\rho(k) = \dfrac{\gamma(k)}{\gamma(0)}$
sample	$c_k = \dfrac{1}{n}\displaystyle\sum_{t=1}^{n-k}(x_t - \bar{x})(x_{t+k} - \bar{x})$	$r_k = \dfrac{c_k}{c_0}$

Table 7.4: Population and sample lag k autocovariance and autocorrelation.

Computing Sample Autocovariance and Autocorrelation

We now consider the estimation of the lag k sample autocovariance c_k and the lag k sample autocorrelation r_k in R. We write an R function named `autocovariance` below that has two arguments: the vector containing the time series `x` and the lag `k`. The first statement in the function uses the `length` function to determine the number of observations in the time series. The second statement uses the `mean` function to calculate the sample mean of the values in the time series. The third statement uses the formula

$$c_k = \frac{1}{n}\sum_{t=1}^{n-k}(x_t - \bar{x})(x_{t+k} - \bar{x})$$

to calculate the lag k sample autocovariance.

```
autocovariance = function(x, k) {
  n = length(x)
  xbar = mean(x)
  sum((x[1:(n - k)] - xbar) * (x[(k + 1):n] - xbar)) / n
}
```

We can now write an R function named `autocorrelation` below that has the same arguments as the `autocovariance` function. It uses the formula

$$r_k = \frac{c_k}{c_0}$$

to calculate the lag k sample autocorrelation.

```
autocorrelation = function(x, k) {
  autocovariance(x, k) / autocovariance(x, 0)
}
```

Time series analysts typically plot the sample autocorrelation function values for the first few lags. This plot is known as either the *sample autocorrelation function* or the *correlogram*. We illustrate the calculation and plotting of the correlogram for a simulated time series whose elements are Gaussian white noise, so $\sigma_X = \sigma_Z$. Recall that Gaussian white noise, denoted by

$$X_t \sim GWN\left(0, \sigma_Z^2\right),$$

consists of mutually independent $N\left(0, \sigma_Z^2\right)$ observations. Recall from Example 7.15 that the population autocovariance function is

$$\gamma(k) = \begin{cases} \sigma_Z^2 & k = 0 \\ 0 & k = 1, 2, \ldots \end{cases}$$

and the population autocorrelation function is

$$\rho(k) = \begin{cases} 1 & k = 0 \\ 0 & k = 1, 2, \ldots . \end{cases}$$

We expect the sample autocorrelation function to be similar to the population autocorrelation function except for random sampling variability. The R code below plots the correlogram for the first 20 lags for a time series that consists of $n = 100$ observations of a Gaussian white noise time series. We have assumed here that the population variance of the Gaussian white noise is equal to one (that is, $\sigma_Z = 1$). The first statement in the R code below uses the `set.seed` function to set the random number seed to `8`. The second statement uses the `rnorm` function to generate a time series consisting of 100 mutually independent standard normal random variates. The vector `correlogram` is initialized to a vector of length 21. This will hold the lag 0 sample autocorrelation function value (which is always $r_0 = 1$) and the sample autocorrelation function values for lags 1 to 20. The `autocorrelation` function defined previously will compute the sample autocorrelation values. Finally, the `plot` function is used to plot the sample autocorrelation function. Using the `type = "h"` argument in the call to `plot` graphs the sample autocorrelation values as spikes. This is largely a matter of personal taste. Some time series analysts prefer to connect them with a line. We take the spike approach to emphasize that a non-integer value for the lag has no meaning in the context described here. The `ylim = c(-1, 1)` argument is included so that the entire potential range of the sample autocorrelation values $-1 \leq r_k \leq 1$ is included. The `abline` function is used to draw a horizontal line at $r_k = 0$, and two other dashed lines that will be described subsequently.

```
set.seed(8)
n = 100
x = rnorm(n)
correlogram = numeric(21)
for (i in 1:21) correlogram[i] = autocorrelation(x, i - 1)
plot(0:20, correlogram, type = "h", ylim = c(-1, 1))
abline(h = 0)
abline(h = c(-1, 1) * 2 / sqrt(n), lty = 2)
```

The plot of the time series and the correlogram for the first 20 lags are given in Figure 7.12. The time series plot displays the typical pattern for Gaussian white noise. The observations are mutually independent, so equally likely to be positive or negative. There are just a handful of observations more than 2 units away from the population mean function $\mu(t) = E[X_t] = 0$. The correlogram is exactly what we anticipated for a time series consisting of Gaussian white noise based on our population autocorrelation function $\rho(k)$, which was one at lag zero and zero at all other lags. We have $r_0 = 1$, as expected, and then small spikes associated with values of r_k at other lag values k that reflect the random sampling variability in the specific time series values that were generated by the `rnorm` function. The correlogram has a horizontal line drawn at correlation 0 to make it clearer which spikes are positive and which are negative. In addition, the correlogram would be identical if all of the points in the time series were translated to have arbitrary population mean μ rather than population mean 0. Correlograms are not influenced by a shift in the time series. Since drawing a

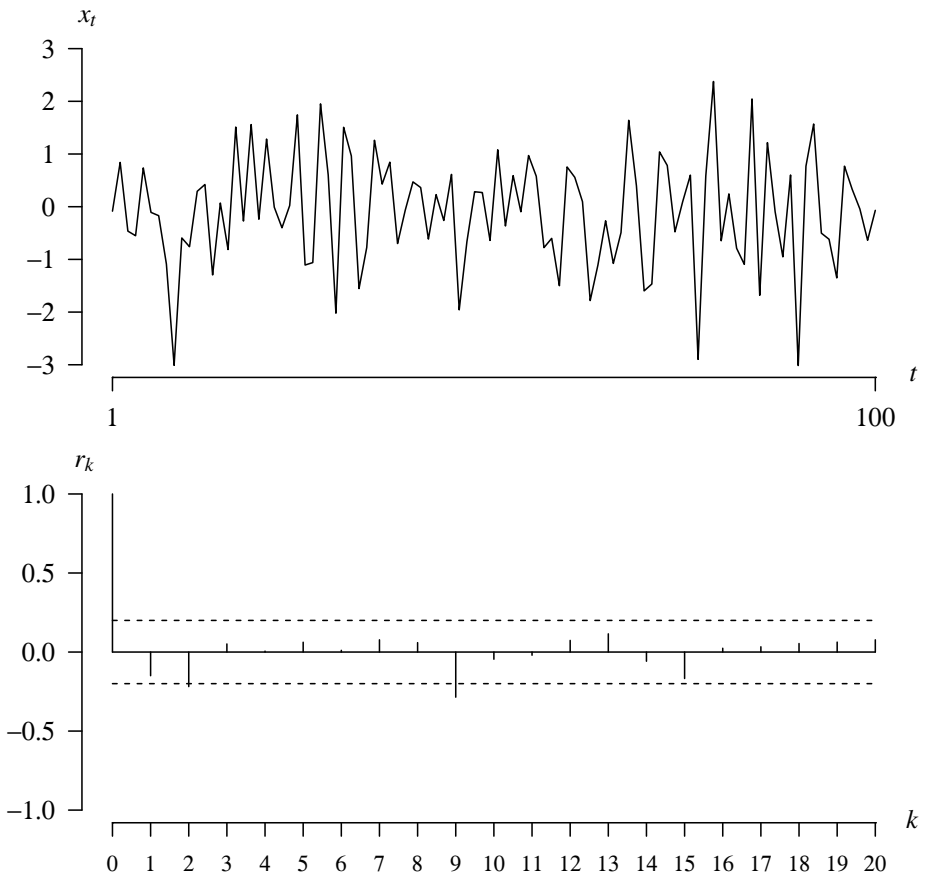

Figure 7.12: Time series plot and correlogram for $n = 100$ Gaussian white noise observations.

correlogram occurs so frequently in the analysis of a time series, R has automated the process with the `acf` function (acf is an abbreviation for autocorrelation function). After a call to `set.seed(8)`, all of the previous calculations can be performed with the single R statement

```
acf(rnorm(100), ylim = c(-1, 1), lag.max = 20)
```

When you make this call to the `acf` function, you will notice that two dashed lines are drawn just above and just below the horizontal line $r_k = 0$, as was the case in Figure 7.12. These two lines are 95% confidence limits that are helpful for determining whether the sample autocorrelation function values are statistically different from zero. Even for a time series consisting of Gaussian white noise, the probability distribution of r_k is complicated because the formula for r_k is complicated. However, when the time series $\{X_t\}$ consists of mutually independent observations, the population mean and variance of r_k are

$$E[r_k] \cong \frac{1}{n}$$

and

$$V[r_k] \cong \frac{1}{n}$$

for $k = 1, 2, \ldots$, and these values are asymptotically normally distributed. This means that an approximate two-sided 95% confidence interval for r_k when n is large and k is significantly less than n is

$$\frac{1}{n} - 1.96\frac{1}{\sqrt{n}} < r_k < \frac{1}{n} + 1.96\frac{1}{\sqrt{n}},$$

where 1.96 is the 0.975 fractile of the standard normal distribution. Since n is typically large in time series analysis, the $1/n$ term is often assumed to be small enough to ignore. Furthermore, if 1.96 is rounded to 2, then this approximate 95% confidence interval simplifies to

$$-\frac{2}{\sqrt{n}} < r_k < \frac{2}{\sqrt{n}},$$

The limits at $\pm 2/\sqrt{n}$ are plotted in Figure 7.12 as dashed lines at $\pm 2/\sqrt{100} = \pm 0.2$. We notice that the spikes in the correlogram in Figure 7.12 fall outside of the confidence interval limits for lag 2 (just barely) and lag 9. We should not be concerned about this occurring. Since these are approximate 95% confidence intervals, we would expect to have about 1 in 20 values fall outside of the limits even if we had mutually independent observations in the time series. Since it appears that there is little or no pattern to the spikes in Figure 7.12, we conclude that the two spikes that exceeded the confidence limits are just due to random sampling variability. Significant spikes at lower lags, for example, lag 1 and lag 2 should be scrutinized more carefully than other lone significant spikes, such as the one that we saw at lag 9. Furthermore, a significant spike at a lag associated with possible seasonal variation (for example, lag 12 for monthly data with a suspected annual variation) should also be scrutinized more carefully than other statistically significant spikes.

Correlogram Examples

Experience is critical in interpreting correlograms. Four examples are given here to illustrate the recommended thought process associated with the interpretation of a time series and its correlogram. The four examples are

- a time series with a linear trend illustrated by the population of Australia from 1971–1993,
- a time series of the first 100 Dow Jones Industrial Average closing values in the year 2000,
- a time series of chemical yields, and
- a seasonal time series illustrated by the home energy consumption values from 2011–2018.

For all four time series, we (a) plot the time series, (b) plot the associated correlogram, (c) interpret the statistically significant spikes in the correlogram, and (d) interpret the shape of the spikes in the correlogram.

> **Example 7.18** This example considers the calculation of the sample autocorrelation function for a time series with a linear trend. The time series consists of $n = 89$ quarterly observations, which are the quarterly number of Australian residents (in thousands) from the second quarter of 1971 through the second quarter of 1993. This time series is built into R and has the name `austres`. Plot the time series and associated correlogram, and interpret the significance of the spikes and shape formed by the values of r_k.
>
> We can view the observations in the time series by just typing the name of the data set.
>
> ```
> austres
> ```

The output from this command is given below.

```
        Qtr1    Qtr2    Qtr3    Qtr4
1971         13067.3 13130.5 13198.4
1972 13254.2 13303.7 13353.9 13409.3
1973 13459.2 13504.5 13552.6 13614.3
1974 13669.5 13722.6 13772.1 13832.0
1975 13862.6 13893.0 13926.8 13968.9
1976 14004.7 14033.1 14066.0 14110.1
1977 14155.6 14192.2 14231.7 14281.5
1978 14330.3 14359.3 14396.6 14430.8
1979 14478.4 14515.7 14554.9 14602.5
1980 14646.4 14695.4 14746.6 14807.4
1981 14874.4 14923.3 14988.7 15054.1
1982 15121.7 15184.2 15239.3 15288.9
1983 15346.2 15393.5 15439.0 15483.5
1984 15531.5 15579.4 15628.5 15677.3
1985 15736.7 15788.3 15839.7 15900.6
1986 15961.5 16018.3 16076.9 16139.0
1987 16203.0 16263.3 16327.9 16398.9
1988 16478.3 16538.2 16621.6 16697.0
1989 16777.2 16833.1 16891.6 16956.8
1990 17026.3 17085.4 17106.9 17169.4
1991 17239.4 17292.0 17354.2 17414.2
1992 17447.3 17482.6 17526.0 17568.7
1993 17627.1 17661.5
```

The plot of the time series and the plot of the sample autocorrelation function are graphed one above the another using the R `plot.ts` and `acf` functions. The `par` function called with the argument `mfrow = c(2, 1)` creates a template for a 2×1 matrix of graphs.

```
par(mfrow = c(2, 1))
plot.ts(austres, type = "p", cex = 0.4)
abline(h = mean(austres))
acf(austres)
```

The default on the `plot.ts` function is to connect the time series values with lines. That default is modified here by setting the `type` argument to `"p"` in order to just plot points instead. The `cex` (character expand) parameter controls the size of the points. A horizontal line has been added to the time series plot using the `abline` function at the sample mean value of the time series, $\bar{x} = 15,273$, which will aid in the interpretation of the values of r_k. The plots are displayed in Figure 7.13. The time series is plotted as just points because of the linear growth in the population. The first 46 of the $n = 89$ observations are below the sample mean, and the remainder are above the sample mean. Consider now the calculation of c_1, the lag 1 sample autocovariance. The formula for c_1 is

$$c_1 = \frac{1}{n} \sum_{t=1}^{n-1} (x_t - \bar{x})(x_{t+1} - \bar{x}).$$

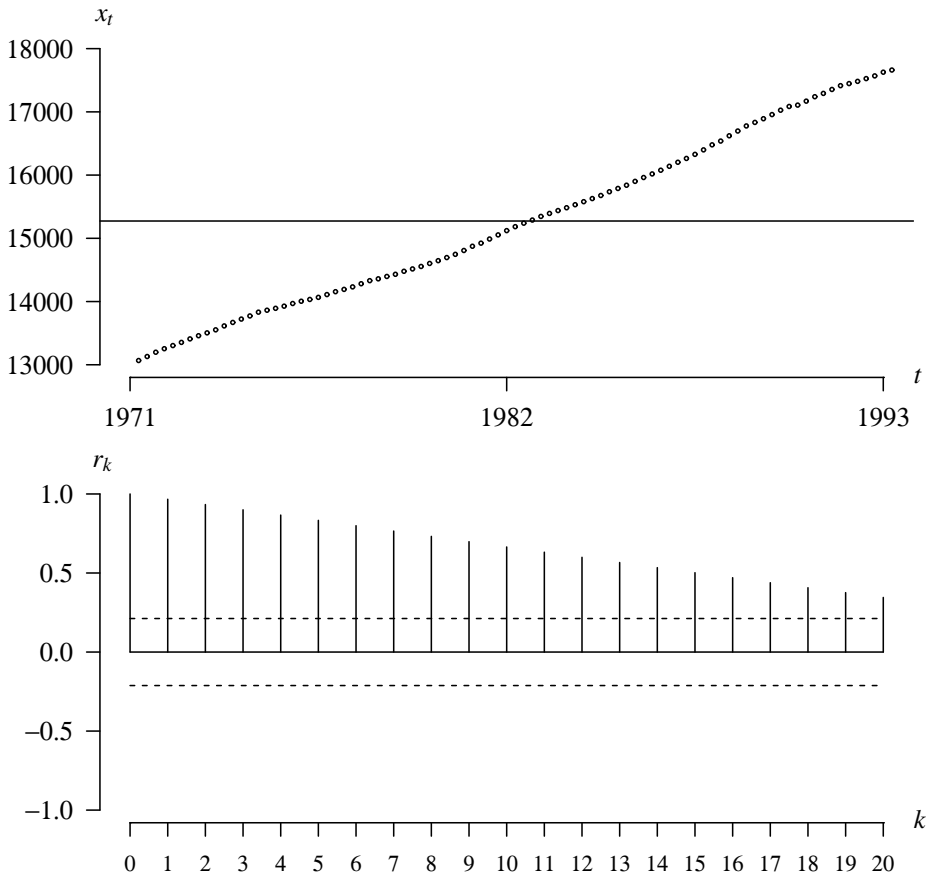

Figure 7.13: Time series plot and correlogram for $n = 89$ quarterly population observations.

Consider adjacent observations in the time series (that is, observations that are just one lag apart). The first two observations, x_1 and x_2, are both less than $\bar{x}$, so the product $(x_1 - \bar{x})(x_2 - \bar{x})$ makes a positive contribution to c_1. Likewise, x_2 and x_3 make a positive contribution to c_1. Likewise, x_3 and x_4 make a positive contribution to c_1. In fact, all of the adjacent observations make a positive contribution to c_1 except for x_{46} and x_{47}, which are on opposite sides of $\bar{x}$, so this pair makes a negative contribution. It is for this reason that c_1 will be positive for this particular time series, and the associated correlation r_1 will be positive and statistically significant. The terms in c_1 and c_0 are very similar for this time series, so r_1 will be close to 1.

Now consider observations in the time series that are two lags apart. There are now $n-2$ terms in the summation for c_2. The first two observations, x_1 and x_3, are both less than $\bar{x}$, so the product $(x_1 - \bar{x})(x_3 - \bar{x})$ makes a positive contribution to c_2. Likewise, x_2 and x_4 make a positive contribution to c_2. Likewise, x_3 and x_5 make a positive contribution to c_2. In fact, all of the observations make a positive contribution to c_1 except for two pairs, x_{45} and x_{47} and x_{46} and x_{48}, which are on opposite sides of $\bar{x}$. These pairs make a

negative contribution. So there will be a significant positive value for r_2, but it will be slightly smaller in magnitude than r_1. This pattern continues for $r_3, r_4, \ldots, r_{32}$ as the r_k values are a decreasing value of k. Then at lag 33, which is beyond the lags displayed in Figure 7.13, there is an approximately equal number of positive and negative terms in the summation to calculate c_{33}. This results in r_{33} being the first negative value in the correlogram. So r_{33} and the sample autocorrelation function values that follow it are negative. So in conclusion, a time series with a linear trend (either increasing or decreasing) has a correlogram in which the initial spikes of r_k are linearly decreasing in k. The correlogram for a time series with a linear or nonlinear trend will not have a traditional interpretation that will be seen in the other examples because the trend overwhelms the values in the correlogram. It is often the case that the trend is first removed, and then the correlogram of the detrended series is analyzed. It is a good exercise to use the `acf` function with a bigger `lag.max` argument than the default to see what the autocorrelation function does for larger values of k.

The next example considers a time series that has statistically significant positive autocorrelation values at small lags.

Example 7.19 Consider again the time series of the first $n = 100$ Dow Jones Industrial Averages during 2000 that was first detailed in Example 7.4. Plot the time series and associated correlogram, and interpret the significance of the spikes and shape formed by the values of r_k.

The time series (with a horizontal line drawn at the mean value $\bar{x} = 10{,}766$) and the correlogram are shown in Figure 7.14. Consider the lag 1 sample autocorrelation. Considering the adjacent observations in the time series plot, the vast majority lie on the same side of $\bar{x}$. This results in a statistically significant positive lag 1 sample autocorrelation r_1. Similar thinking should convince you that there will also be a statistically significant positive lag 2 sample autocorrelation r_2. This time series exhibits runs of significant length above and below the population mean, so it has a dozen statistically significant initial spikes on the correlogram. So a time series that is well-modeled by a random walk (as shown in Example 7.4) has a correlogram with statistically significant early positive sample autocorrelation values.

The next example considers a stationary time series in which adjacent observations tend to be on opposite sides of the sample mean.

Example 7.20 Consider the time series consisting of $n = 70$ consecutive yields from a batch chemical process from page 32 of Box and Jenkins (1976) given in Table 7.5 (read row-wise). Plot the time series and associated correlogram. Interpret the statistical significance of the spikes and the shape formed by the values of r_k.

The time series plot of the yields, along with a horizontal line at $\bar{x} = 51.1$, is given in the top graph in Figure 7.15. The bottom graph contains the associated correlogram. The time series plot indicates that a large yield is followed by a small yield in a majority of the observations, so we expect a negative lag 1 sample autocorrelation function value. The lag 1 sample autocorrelation function value is $r_1 = -0.390$. Since observations that are two apart tend to be on the same side of the sample mean, the lag 2 sample autocorrelation function value is $r_2 = 0.304$. So a time series which tends to jump above and below the sample mean results in a correlogram whose r_k values alternate

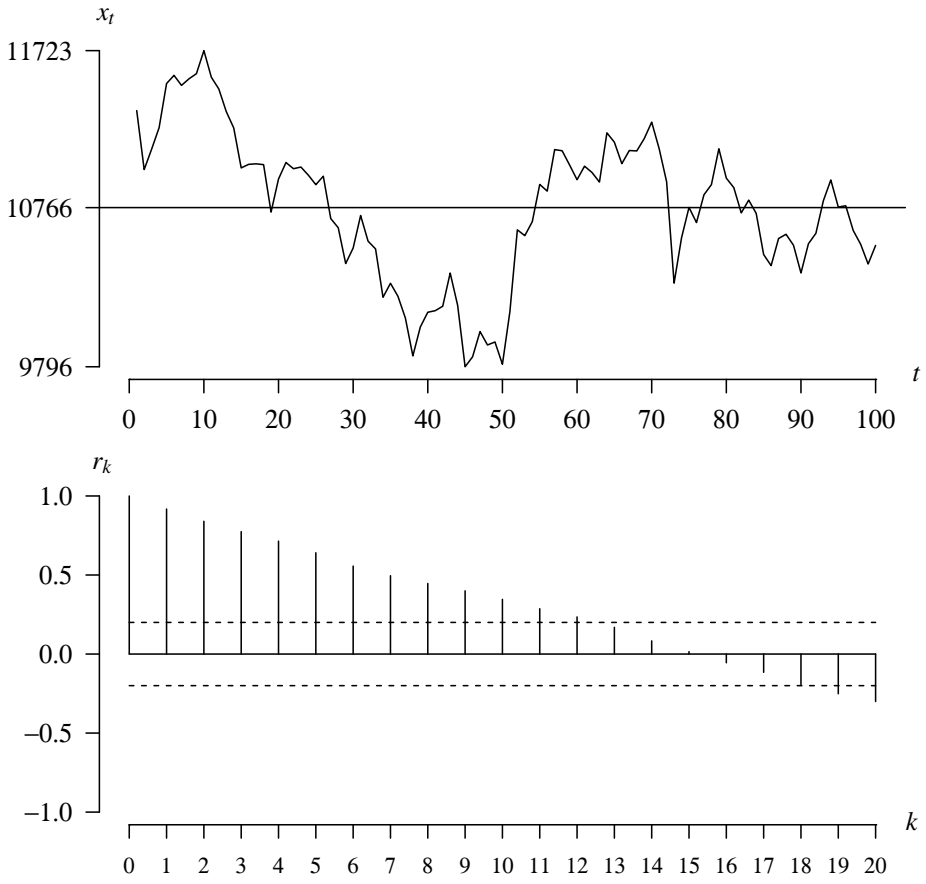

Figure 7.14: Time series plot and correlogram for $n = 100$ DJIA closing prices.

in sign. These are the only two values of the correlogram that show a statistically significant difference from zero because they lie outside of the 95% confidence limits. The dashed horizontal lines corresponding to 95% confidence bounds that determine

47	64	23	71	38	64	55	41	59	48	71	35	57	40
58	44	80	55	37	74	51	57	50	60	45	57	50	45
25	59	50	71	56	74	50	58	45	54	36	54	48	55
45	57	50	62	44	64	43	52	38	59	55	41	53	49
34	35	54	45	68	38	50	60	39	59	40	57	54	23

Table 7.5: A time series of $n = 70$ consecutive yields from a chemical process.

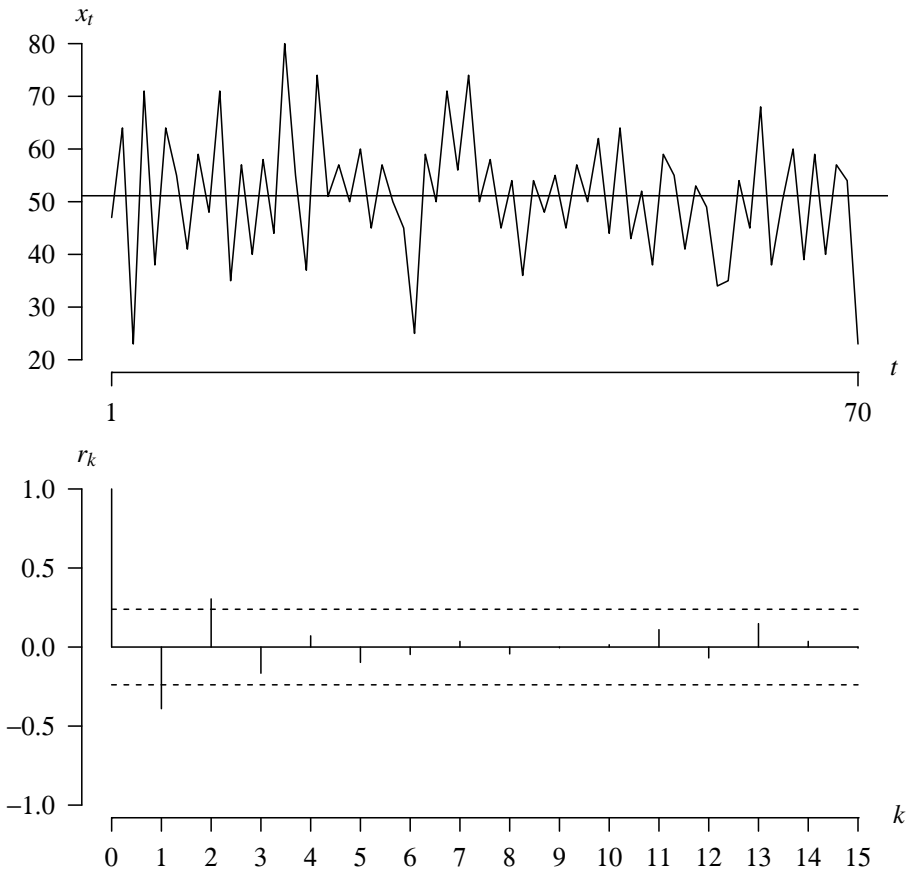

Figure 7.15: Time series plot and correlogram of $n = 70$ yields from a chemical process.

statistical significance are drawn on the correlogram at heights

$$\pm\frac{2}{\sqrt{70}} \cong \pm 0.239.$$

These two sample autocorrelation function values might be due to overcorrection by the personnel running the batch chemical process.

The final example illustrates the impact of a time series with a seasonal component on the shape of the correlogram.

Example 7.21 Consider again the home energy consumption time series from Example 7.1 consisting of $n = 84$ monthly observations (measured in kilowatt hours) collected between 2011 and 2018. Plot the time series and associated correlogram. Interpret the statistical significance of the spikes and the shape formed by the values of r_k.

The time series and correlogram are shown in Figure 7.16, with a horizontal line drawn at the mean value $\bar{x} = 1703$ kilowatt hours. The time series displays a seasonal pattern, with higher energy consumption during the winter months and the summer months. The winter months tend to draw more energy than the summer months. The correlogram for a time series with a seasonal component is also cyclic, with a frequency that matches the frequency in the time series. Since the summer and winter months draw more energy from the heat pump, the cycle on the correlogram repeats itself with a wavelength of 6. The fact that the magnitude of r_{12} is larger than the magnitude of r_6 is due to the fact that the winter months draw more energy than the summer months. For this particular time series, the shape of the correlogram does not provide much information beyond confirming that this is a time series with a seasonal component. The more valuable information is typically gleaned by first detrending the time series (that is, removing the seasonal component) and inspecting the correlogram of the detrended time series. The statistically significant sample autocorrelation function value at lag 3, which is $r_3 = -0.397$, indicates that energy consumption in months that differ by 3 (for example,

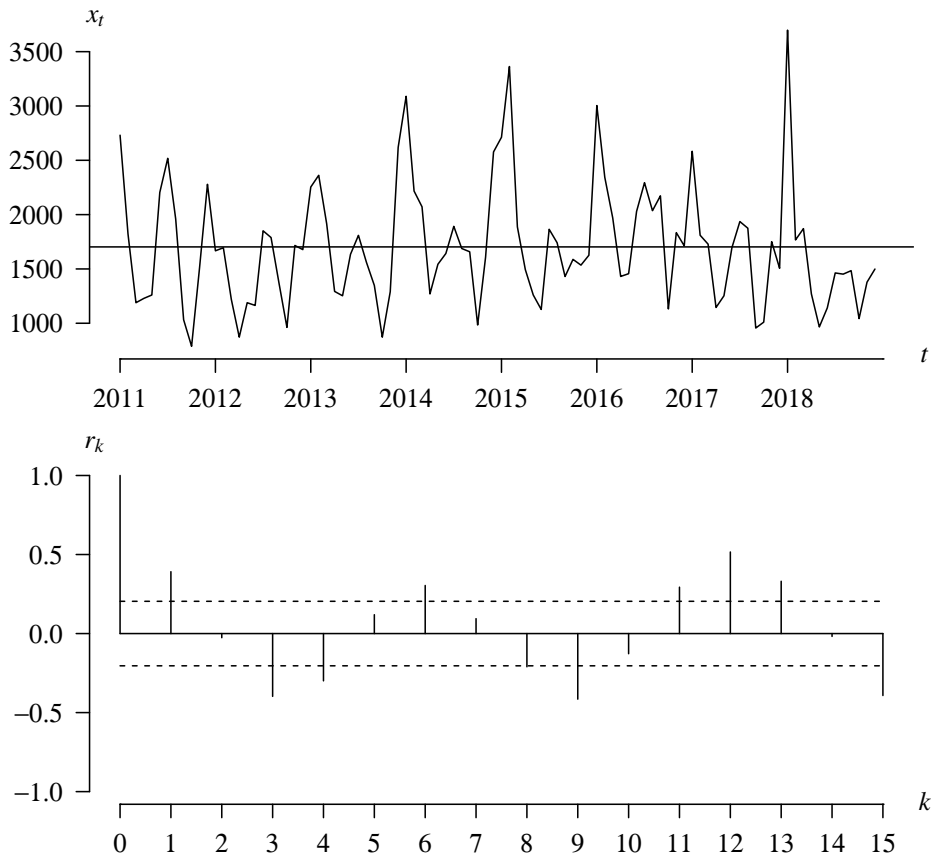

Figure 7.16: Time series plot and correlogram for $n = 84$ home energy consumption values.

February and May), tend to have energy consumptions that lie on the opposite side of the sample mean.

To summarize the section thus far, the population mean function $\mu(t) = E[X_t]$ is the expected value of the time series at time t, and indicates whether a trend and/or cyclic variation is present. The population autocovariance and population autocorrelation functions reflect the linear relationship between two values, X_s and X_t, in the time series. A time series is stationary if the population mean function is constant in t and the population autocovariance function $\gamma(s,t) = \text{Cov}(X_s, X_t)$ depends only on $|t-s|$. For a stationary time series, the population autocovariance function and the population autocorrelation function can be written as a function of the lag $k = |t-s|$ as $\gamma(k)$ and $\rho(k)$. The sample autocorrelation function estimates the population autocorrelation function.

Occasions arise in time series analysis in which we are also interested in the correlation between X_t and X_{t+k} with the linear dependency on the values between these two values, namely $X_{t+1}, X_{t+2}, \ldots, X_{t+k-1}$, removed. This leads to what is known as the partial autocorrelation function, which is presented next.

7.2.4 Population Partial Autocorrelation

One key characteristic of nearly all time series is that nearby observations tend to be correlated. This makes the notion of autocorrelation crucial in time series analysis because it captures the correlation between observations in a stationary time series that are separated in time by a prescribed number of lags. The population autocorrelation function was introduced in Section 7.2.1; its statistical counterpart, the sample autocorrelation function, was introduced in Section 7.2.3.

It is often difficult to distinguish between population autocorrelation functions for two different competing, tentative stationary time series models in practice because

- the population autocorrelation functions for the two models are nearly identical, and/or
- there is significant sampling variability in the sample autocorrelation functions making it difficult to determine which of the two models provides a better fitted model.

As will be seen in subsequently, the sample autocorrelation function is particularly helpful for determining the number of terms to include in a popular time series model known as a *moving average model*. However, the sample autocorrelation function is much less helpful for determining the number of terms to include in another popular time series model known as an *autoregressive model*. A second type of autocorrelation function, the *partial autocorrelation function*, is an ancillary diagnostic tool to help determine the number of terms to include in an autoregressive model. As was the case with the autocorrelation function, there is a population and a sample version of the partial autocorrelation function. The population partial autocorrelation function is introduced in this subsection.

The notion behind partial autocorrelation is intuitive. Before describing the interpretation of partial autocorrelation in the context of time series analysis, we present a scenario involving just partial correlation in a more general setting. Let's say you are interested in the correlation between a full-time employee's age, X, and their annual income, Y. Intuition suggests that this correlation is positive because annual income tends to rise with age. But there are other factors that influence income, such as the employee's education level achieved, the number of years on the job, specific industry of employment, etc. To simplify, let's consider just one of these factors, say, the employee's years of education achieved, Z. The population partial correlation is the population correlation between age X and annual income Y with the linear relationship associated with the number of

years of education Z removed. We are effectively controlling for the influence of education as we measure the correlation between X and Y. We regress X on Z to obtain $\tilde{X}$. We regress Y on Z to obtain $\tilde{Y}$. This regression is in the sense of probability rather than its statistical counterpart (which typically uses least squares for parameter estimation). Finally, we calculate the population partial correlation $\text{Corr}\left(X-\tilde{X}, Y-\tilde{Y}\right)$, which is the population correlation between X and Y with the linear influence of Z removed. Like the ordinary population correlation, the population partial correlation falls in the closed interval $[-1, 1]$. The extreme values on this interval correspond to perfect negative population correlation and perfect positive population correlation, respectively. To summarize, the partial correlation measures the degree of linear association between two variables, with the linear association of one or more other variables removed.

Returning to the time series context, the partial autocorrelation reflects the relationship between observations at a particular lag in a time series with the linear relationship associated with intervening observations removed. The variables whose influence is being removed are the observations between the two values of the time series of interest. Stated in another fashion, the partial autocorrelation at lag k is the population correlation between two observations in the time series that are k time units apart after the removal of the linear influence of the time series observations at lags $1, 2, \ldots, k-1$. As was the case with autocorrelation, we want to find the population and sample versions of the partial autocorrelation. We will then have an inventory of possible population partial autocorrelation shapes that we can match to sample partial autocorrelation functions, which will help determine the number of terms to include in a time series model. The main role of the sample partial autocorrelation function is to determine the number of terms to include in an autoregressive model.

We now develop some general notation concerning partial autocorrelation. Although many authors use ϕ_{kk} to denote the population lag k partial autocorrelation, we will instead use $\rho^*(k)$ to emphasize that this quantity is still a correlation and to use the symbol ϕ exclusively for the coefficients in an autoregressive time series model. The superscript $*$ is used to distinguish the population partial autocorrelation function from the population autocorrelation function $\rho(k)$. Likewise, we will use r_k^* to denote the sample lag k partial autocorrelation in the next subsection. The superscript $*$ is used here to distinguish the sample partial autocorrelation function from the sample autocorrelation function r_k.

The next example illustrates the calculation of a population partial autocorrelation for a stationary time series model.

Example 7.22 Consider the stationary time series model

$$X_t = 0.8X_{t-1} + Z_t,$$

where $\{Z_t\} \sim WN\left(0, \sigma_Z^2\right)$. The current value in the time series is 0.8 times the previous value in the time series, plus a random shock of white noise Z_t. This time series model is similar to the random walk time series model that was introduced in Example 7.4 and analyzed in Examples 7.8, 7.11, and 7.17. The random walk time series model was determined to be nonstationary. The one small difference between this time series model and the random walk is the 0.8 coefficient associated with the X_{t-1} term. This small alteration makes this time series model stationary. What is the population lag 2 partial autocorrelation?

The population lag 2 partial autocorrelation is the population correlation between X_t and X_{t-2} with the linear effect of the intervening observation X_{t-1} removed. This is the population correlation between $X_t - 0.8X_{t-1}$ and $X_{t-2} - 0.8X_{t-1}$, which can be written

as

$$\begin{aligned}\rho^*(2) &= \text{Corr}\,(X_t - 0.8X_{t-1}, X_{t-2} - 0.8X_{t-1}) \\ &= \frac{\text{Cov}\,(X_t - 0.8X_{t-1}, X_{t-2} - 0.8X_{t-1})}{\sqrt{V\,[X_t - 0.8X_{t-1}]\,V\,[X_{t-2} - 0.8X_{t-1}]}} \\ &= \frac{\text{Cov}\,(Z_t, X_{t-2} - 0.8X_{t-1})}{\sqrt{V\,[Z_t]\,V\,[X_{t-2} - 0.8X_{t-1}]}} \\ &= \frac{0}{\sqrt{V\,[Z_t]\,V\,[X_{t-2} - 0.8X_{t-1}]}} \\ &= 0\end{aligned}$$

because the population covariance between the white noise term at time t, which is Z_t, and the linear combination of the two previous values of the time series, which is $X_{t-2} - 0.8X_{t-1}$, is zero.

We now pivot from the calculation of population partial autocorrelation for a specific time series model to the calculation of the population partial autocorrelation for a general stationary time series model. Let's begin with the calculation of the lag 1 population partial autocorrelation $\rho^*(1)$ for a stationary time series model. By definition, this is the population correlation between X_t and X_{t-1} after removing the linear effect of any observations between X_t and X_{t-1}. But there aren't any observations between X_t and X_{t-1}, so the lag 1 population partial autocorrelation is simply the lag 1 population autocorrelation: $\rho^*(1) = \rho(1)$. The population partial autocorrelation for higher lags uses the best linear estimate of each of the two values of interest as a function of the intervening values. Minimizing the associated mean square error, the population partial autocorrelation can be determined by solving a set of linear equations. Using Cramer's rule to solve these equations, the population lag 2 partial autocorrelation function value is given by the ratio of determinants

$$\rho^*(2) = \frac{\begin{vmatrix} 1 & \rho(1) \\ \rho(1) & \rho(2) \end{vmatrix}}{\begin{vmatrix} 1 & \rho(1) \\ \rho(1) & 1 \end{vmatrix}}.$$

Notice that the denominator is the determinant of the correlation matrix of any two adjacent observations. The numerator is the determinant of this same matrix with the last column replaced by the first two population autocorrelation values. This pattern continues for the population lag 3 partial autocorrelation function value, which is

$$\rho^*(3) = \frac{\begin{vmatrix} 1 & \rho(1) & \rho(1) \\ \rho(1) & 1 & \rho(2) \\ \rho(2) & \rho(1) & \rho(3) \end{vmatrix}}{\begin{vmatrix} 1 & \rho(1) & \rho(2) \\ \rho(1) & 1 & \rho(1) \\ \rho(2) & \rho(1) & 1 \end{vmatrix}}.$$

Again, the denominator is the determinant of the correlation matrix of any three sequential observations. The numerator is the determinant of this same matrix with the last column replaced by the first three population autocorrelation values (where the lag number matches the row number). This pattern continues for higher lag values, which leads to the following definition.

Definition 7.8 For a stationary time series model, the lag 0 population partial autocorrelation is 1, the lag 1 population partial autocorrelation is $\rho(1)$, and the lag k population partial autocorrelation is

$$\rho^*(k) = \frac{\begin{vmatrix} 1 & \rho(1) & \rho(2) & \cdots & \rho(1) \\ \rho(1) & 1 & \rho(1) & \cdots & \rho(2) \\ \rho(2) & \rho(1) & 1 & \cdots & \rho(3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho(k-1) & \rho(k-2) & \rho(k-3) & \cdots & \rho(k) \end{vmatrix}}{\begin{vmatrix} 1 & \rho(1) & \rho(2) & \cdots & \rho(k-1) \\ \rho(1) & 1 & \rho(1) & \cdots & \rho(k-2) \\ \rho(2) & \rho(1) & 1 & \cdots & \rho(k-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho(k-1) & \rho(k-2) & \rho(k-3) & \cdots & 1 \end{vmatrix}},$$

for $k = 2, 3, \ldots$.

The next example illustrates the calculation of the population partial autocorrelation function for a stationary time series model.

Example 7.23 Consider the time series model for $\{X_t\}$ described by

$$X_t = Z_t - Z_{t-1} + \frac{1}{2}Z_{t-2},$$

where $\{Z_t\} \sim WN\left(0, \sigma_Z^2\right)$. The current value of the time series is a linear combination of the current and two previous shock values. Find the population autocorrelation function and the population partial autocorrelation function for the first eight lags.

The population mean function is

$$\mu(t) = E[X_t] = E\left[Z_t - Z_{t-1} + \frac{1}{2}Z_{t-2}\right] = E[Z_t] - E[Z_{t-1}] + \frac{1}{2}E[Z_{t-2}] = 0.$$

The population autocovariance function is

$$\begin{aligned}
\gamma(s,t) &= \text{Cov}(X_s, X_t) \\
&= E\left[(X_s - E[X_s])(X_t - E[X_t])\right] \\
&= E[X_s X_t] \\
&= E\left[\left(Z_s - Z_{s-1} + \frac{1}{2}Z_{s-2}\right)\left(Z_t - Z_{t-1} + \frac{1}{2}Z_{t-2}\right)\right] \\
&= E[Z_s Z_t] - E[Z_s Z_{t-1}] + \frac{1}{2}E[Z_s Z_{t-2}] - E[Z_{s-1} Z_t] + E[Z_{s-1} Z_{t-1}] - \\
&\quad \frac{1}{2}E[Z_{s-1} Z_{t-2}] + \frac{1}{2}E[Z_{s-2} Z_t] - \frac{1}{2}E[Z_{s-2} Z_{t-1}] + \frac{1}{4}E[Z_{s-2} Z_{t-2}].
\end{aligned}$$

When $t = s$,

$$\gamma(t,t) = E\left[Z_t^2\right] + E\left[Z_{t-1}^2\right] + \frac{1}{4}E\left[Z_{t-2}^2\right] = V[Z_t] + V[Z_{t-1}] + \frac{1}{4}V[Z_{t-2}] = \frac{9}{4}\sigma_Z^2$$

because of the mutual independence of the white noise terms and the fact that the expected value of each white noise term is zero. When $|t-s|=1$, for example, when $t=s-1$,

$$\gamma(s,t) = -E\left[Z_{s-1}Z_t\right] - \frac{1}{2}E\left[Z_{s-2}Z_{t-1}\right] = -\frac{3}{2}\sigma_Z^2.$$

When $|t-s|=2$, for example, when $t=s-2$,

$$\gamma(s,t) = \frac{1}{2}E\left[Z_{s-2}Z_t\right] = \frac{1}{2}\sigma_Z^2.$$

When $|t-s|=3,4,\ldots$, the population autocovariance is $\gamma(s,t)=0$ because each expected value in the expansion of $\gamma(s,t)$ contains independent random variables. Since the population mean function is constant in time and the population autocovariance is a function of the lag $k=|t-s|$ (as required by Definition 7.6), we have established that the time series model is stationary with population autocovariance function

$$\gamma(k) = \begin{cases} 9\sigma_Z^2/4 & k=0 \\ -3\sigma_Z^2/2 & k=1 \\ \sigma_Z^2/2 & k=2 \\ 0 & k=3,4,\ldots. \end{cases}$$

Since $\rho(k)=\gamma(k)/\gamma(0)$, the population autocorrelation function is

$$\rho(k) = \begin{cases} 1 & k=0 \\ -2/3 & k=1 \\ 2/9 & k=2 \\ 0 & k=3,4,\ldots, \end{cases}$$

Notice that the population autocorrelation function is independent of the population variance of the white noise σ_Z^2.

We now turn to calculation the population partial autocorrelation function. The lag 0 population partial autocorrelation is $\rho^*(0)=1$. From Definition 7.8, the lag 1 population partial autocorrelation is $\rho^*(1)=\rho(1)=-2/3$. The lag 2 population partial autocorrelation is the ratio of the determinants of two 2×2 matrices:

$$\rho^*(2) = \frac{\begin{vmatrix} 1 & \rho(1) \\ \rho(1) & \rho(2) \end{vmatrix}}{\begin{vmatrix} 1 & \rho(1) \\ \rho(1) & 1 \end{vmatrix}} = \frac{\begin{vmatrix} 1 & -2/3 \\ -2/3 & 2/9 \end{vmatrix}}{\begin{vmatrix} 1 & -2/3 \\ -2/3 & 1 \end{vmatrix}} = \frac{-2/9}{5/9} = -\frac{2}{5} = -0.4.$$

The lag 3 population partial autocorrelation is the ratio of the determinants of two 3×3 matrices:

$$\rho^*(3) = \frac{\begin{vmatrix} 1 & \rho(1) & \rho(1) \\ \rho(1) & 1 & \rho(2) \\ \rho(2) & \rho(1) & \rho(3) \end{vmatrix}}{\begin{vmatrix} 1 & \rho(1) & \rho(2) \\ \rho(1) & 1 & \rho(1) \\ \rho(2) & \rho(1) & 1 \end{vmatrix}} = \frac{\begin{vmatrix} 1 & -2/3 & -2/3 \\ -2/3 & 1 & 2/9 \\ 2/9 & -2/3 & 0 \end{vmatrix}}{\begin{vmatrix} 1 & -2/3 & 2/9 \\ -2/3 & 1 & -2/3 \\ 2/9 & -2/3 & 1 \end{vmatrix}} = \frac{-8/243}{7/27} = -\frac{8}{63} \cong -0.1270.$$

Computing determinants by hand gets more tedious as the size of the matrices grows. The R code below automates this process, using the `det` function to calculate the determinants. After executing the code, the vector `rhostar` contains the first eight population partial autocorrelation values. Examine the subscripts carefully because there is a lag zero autocorrelation but R begins its subscripts at 1.

```
rho = c(1, -2 / 3, 2 / 9, 0, 0, 0, 0, 0, 0)
rhostar = rho
for (k in 2:(length(rho) - 1)) {
  a = matrix(1, k, k)
  for (i in 1:(k - 1)) a[abs(row(a) - col(a)) == i] = rho[i + 1]
  denominator = det(a)
  a[ , k] = rho[2:(k + 1)]
  numerator = det(a)
  rhostar[k + 1] = numerator / denominator
}
rhostar
```

The calculations in this example have been automated in the `ARMAacf` function in R. The `ar` and `ma` parameters will be described in a subsequent chapter, but notice that the elements in the `ma` vector are the coefficients of Z_{t-1} and Z_{t-2} in the original time series model. The first R command below calculates the values of $\rho(1), \rho(2), \ldots, \rho(8)$, and the second R command calculates the values of $\rho^*(1), \rho^*(2), \ldots, \rho^*(8)$ because the `pacf` argument in the call to `ARMAacf` is set to `TRUE`.

```
ARMAacf(ar = 0, ma = c(-1, 1 / 2), lag.max = 8)
ARMAacf(ar = 0, ma = c(-1, 1 / 2), lag.max = 8, pacf = TRUE)
```

The resulting population autocorrelation and partial autocorrelation functions are plotted in Figure 7.17. The population autocorrelation function cuts off at lag 2 and the population partial autocorrelation function has correlations that appear to behave in a

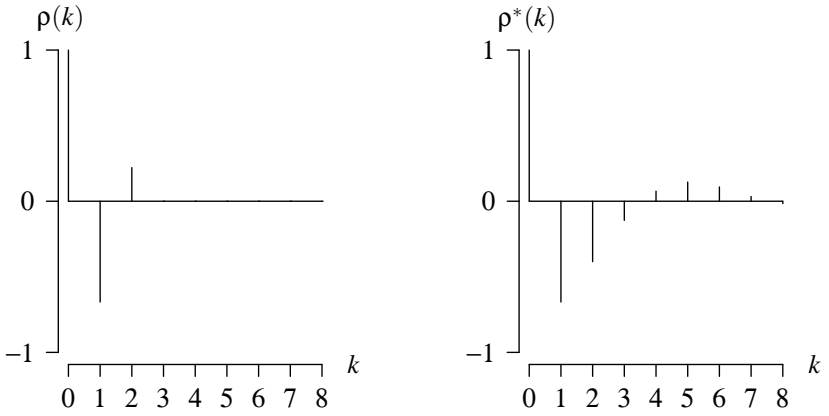

Figure 7.17: Population autocorrelation and partial autocorrelation functions.

damped sinusoidal fashion. Time series analysts refer to this type of population partial autocorrelation function as one that "tails off."

7.2.5 Sample Partial Autocorrelation

We now transition from considering the *population* partial autocorrelation function to considering its statistical counterpart, the *sample* partial autocorrelation function. Calculating the sample partial autocorrelation function is just a matter of replacing the population values with the sample values in the determinants given in Definition 7.8.

Definition 7.9 For a stationary time series model, the lag 0 sample partial autocorrelation is 1, the lag 1 sample partial autocorrelation is r_1, and the lag k sample partial autocorrelation is

$$r^*(k) = \frac{\begin{vmatrix} 1 & r_1 & r_2 & \cdots & r_1 \\ r_1 & 1 & r_1 & \cdots & r_2 \\ r_2 & r_1 & 1 & \cdots & r_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{k-1} & r_{k-2} & r_{k-3} & \cdots & r_k \end{vmatrix}}{\begin{vmatrix} 1 & r_1 & r_2 & \cdots & r_{k-1} \\ r_1 & 1 & r_1 & \cdots & r_{k-2} \\ r_2 & r_1 & 1 & \cdots & r_{k-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{k-1} & r_{k-2} & r_{k-3} & \cdots & 1 \end{vmatrix}},$$

for $k = 2, 3, \ldots$.

Given an observed time series, the R code from the previous example that used the `det` function to calculate the determinants could be used to perform these calculations. The sample partial autocorrelation function can be calculated much more efficiently, however, by using the built-in `pacf` function in R, as illustrated in the next example. The lag in which the sample partial autocorrelation function values become statistically indistinguishable from zero can be determined by using the approximate result that for a time series of white noise values, $r_k^* \sim N(0, 1/n)$, for $k = 1, 2, \ldots$. For this reason, the `pacf` function in R plots dashed lines at the approximate 95% bands at $\pm 1.96/\sqrt{n}$. These dashed lines are useful to a time series analyst in determining which sample partial autocorrelation values differ significantly from zero.

Example 7.24 Plot the time series, sample autocorrelation function, and sample partial autocorrelation function for the $n = 70$ chemical yield values from Example 7.20, repeated in Table 7.6 for convenience. The values in the time series should be read row-wise.

It would be reasonable to simply use the code from the previous example to compute the sample partial autocorrelation function, but we instead illustrate the use of R's built-in `pacf` function here. In addition, the `layout` function can be used to stretch the time series plot from left-to-right on the graphic, but yet compress the plots of the sample autocorrelation function and the sample partial autocorrelation function. The elements in the matrix given as the first argument to `layout` indicate the plot number being displayed. Stretching the time series plot is important in order to visually detect

47	64	23	71	38	64	55	41	59	48	71	35	57	40
58	44	80	55	37	74	51	57	50	60	45	57	50	45
25	59	50	71	56	74	50	58	45	54	36	54	48	55
45	57	50	62	44	64	43	52	38	59	55	41	53	49
34	35	54	45	68	38	50	60	39	59	40	57	54	23

Table 7.6: A time series of $n = 70$ consecutive yields from a chemical process.

patterns in the time series. Nothing is lost by horizontally compressing the sample autocorrelation function and the sample partial autocorrelation function as long as the spike values r_k and r_k^* are distinct on the plots.

```
chemical = scan("chemical.d")
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
plot.ts(chemical)
acf(chemical)
pacf(chemical)
```

If you prefer confidence limits other than the default 95% limits, both `acf` and `pacf` accept a `ci` argument that accepts any argument between 0 and 1, but defaults to 0.95. The resulting plots are displayed in Figure 7.18. There are two statistically significant spikes in the sample autocorrelation function and one statistically significant spike in the sample partial autocorrelation function.

There will be more examples of computing the partial autocorrelation function and its interpretation subsequently.

7.2.6 Computing

The R `plot.ts` function generates a plot of a realization of a time series, which is an important initial step in formulating an appropriate stochastic model for the time series. Many time series analysts prefer to also see the sample autocorrelation and partial autocorrelation functions along with the plot of the time series. The `layout` function can be used to stretch the plot of the time series horizontally, while horizontally compressing the plots of the sample autocorrelation and partial autocorrelation functions. The `acf` function computes the sample autocorrelation function and has arguments that control the number of lags to display, whether to suppress the plot, etc. The `pacf` function computes the sample partial autocorrelation function and has similar arguments. Notice that the `acf` function includes the lag 0 sample autocorrelation, which is always 1, but the `pacf` function does not include the lag 0 sample partial autocorrelation. The statements below apply these functions to the built-in R `AirPassengers` time series.

```
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
plot.ts(AirPassengers)
acf(AirPassengers)
pacf(AirPassengers)
```

The vertical axes on all three plots are autoscaled. Use the `ylim = c(-1, 1)` argument in the `acf` and `pacf` functions in order to stretch the vertical axis from -1 to 1. Finally, the `ARMAacf` function can be used to compute the population autocorrelation and partial autocorrelation function values for a prescribed time series model.

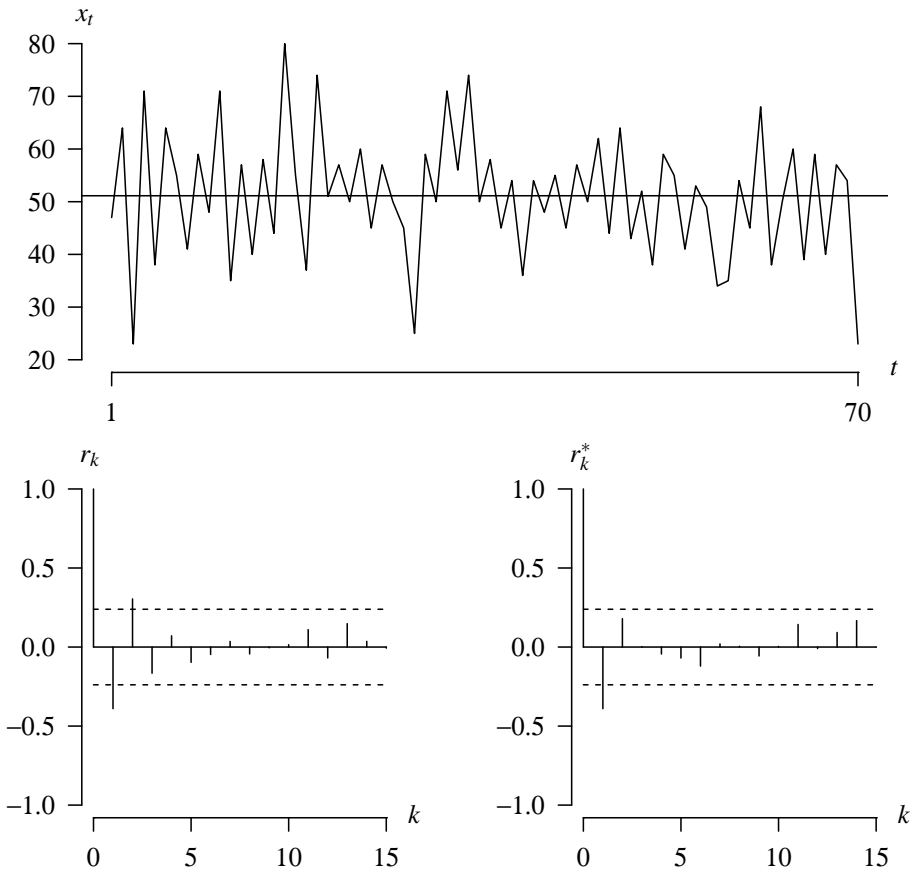

Figure 7.18: Time series plot, r_k, and r_k^* for $n = 70$ yields from a chemical process.

7.3 Operations on a Time Series

This section considers operations that can be performed on a time series. The first subsection introduces *filters* that can be applied to a time series. We have already encountered an example of a filter when we considered a three-point moving average of a time series consisting of Gaussian white noise. The second subsection introduces *decomposition*, which is the process of breaking an observed time series into its component parts. The `AirPassengers` time series that is built into R, for example, can be decomposed into an trend, a seasonal component, and any remaining noise in the process once the trend and seasonal components have been removed. The third subsection concerns R functions that can be helpful in implementing these operations.

7.3.1 Filtering

This section takes up the important topic of *filtering*, which can be thought of as the process of converting one observed time series $\{x_t\}$ to another time series $\{y_t\}$. The mathematical operations required to convert one time series to another can be abstractly depicted as

$$\{x_t\} \xrightarrow{\text{filter}} \{y_t\}$$

It is often the case that several of these filters can be applied sequentially to a particular observed time series. Filter 1, for example, converts $\{x_t\}$ to $\{y_t\}$, and then Filter 2 converts $\{y_t\}$ to $\{z_t\}$.

$$\{x_t\} \xrightarrow{\text{filter 1}} \{y_t\} \xrightarrow{\text{filter 2}} \{z_t\}$$

The resultant time series $\{z_t\}$ is not associated with the white noise terms Z_t from Definition 7.1. The purpose of such a series of filters applied to a time series might be to remove the trend with the first filter, and then to remove some seasonal variation with the second filter. If the resulting time series $\{z_t\}$ looks like random noise values, then the two filters applied in series have successfully removed the trend and the seasonal variation.

Three different general classes of filters will be considered: transformations, detrending, and linear filters. These classes of filters allow for the manipulation of a times series for a particular purpose, such as smoothing or variance stabilization.

Transformations

One simple filter that can be applied to a time series is to apply a transformation to each element of the time series. Two transformations that are commonly applied to a time series are the logarithmic and square root transformations, which are

$$y_t = \ln x_t$$

and

$$y_t = \sqrt{x_t}.$$

Some common purposes of applying such a transformation are to

- stabilize the variance of the time series (for example, when larger values of x_t tend to have greater variability than smaller values of x_t),
- make the trend and seasonal components of a time series appear to be additive, rather than multiplicative, in nature, and
- make the values in the filtered time series appear to be similar to white noise, iid noise, or Gaussian white noise (see Definition 7.1). The advantage to having values of the fitted time series be approximately mutually independent and normally distributed is to enable the use of easier statistical inference procedures concerning, for example, forecasted values.

The transformation of the values in a time series given in the next example is a variance-stabilizing transformation which makes significant improvement to a time series in terms of its visualization and interpretation.

> **Example 7.25** The Dow Jones Industrial Average (DJIA), also known as the Dow 30, was devised by Charles Dow and was initiated on May 26, 1896. The average bears Dow's name and that of statistician and business associate Edward Jones. The DJIA is the average stock price of 30 U.S.-based, publicly traded companies, adjusted for stock splits and the swapping of companies in and out of the average so that it adequately reflects the composition of the domestic stock market. These adjustments are made by altering the average's denominator for historical continuity, which is now much less than 30.

The evolution of the DJIA is not a true reflection of the yield of the 30 stocks because two important factors are not incorporated into the average. First, the average does not factor in dividends that are paid by some of the 30 companies. Second, the average does not factor in inflation, which erodes the true return that a stock investment provides. If dividends were factored into the calculation, the DJIA would be much higher than it is presently; if inflation were factored into the calculation, the DJIA would be much lower than it is presently.

This example considers the time series plot of the average annual DJIA closing values during the 20th century. This plot is generated with the R code given below.

```
x = 1901:2000
y = ts(scan("dow.d"))
plot.ts(x, y)
```

The file `dow.d` contains the 100 annual average closing values. The resulting graph is shown in Figure 7.19.

The DJIA had a sample mean closing value of 69.52 during 1901 and a sample mean closing value of 10731.15 during 2000. The linear scale that is used in Figure 7.19 obscures most of the variability of the DJIA during the first half of the century. The graph can be made more meaningful by using a logarithmic scale on the vertical axis. This is accomplished by calling the `plot.ts` function as `plot.ts(x, y, log = "y")`, resulting in the graph shown in Figure 7.20. This is equivalent to plotting the filtered time series

$$y_t = \log_{10} x_t$$

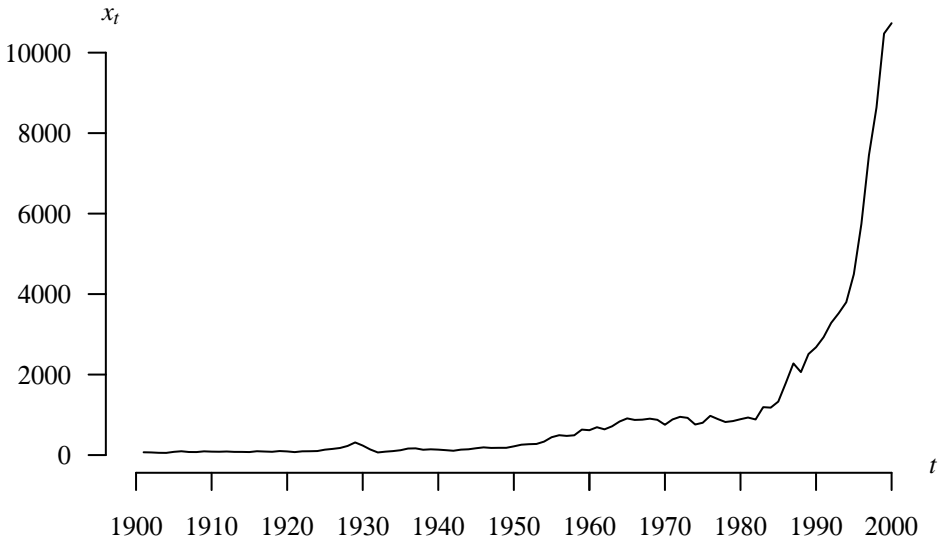

Figure 7.19: Dow Jones Industrial Average (1901–2000).

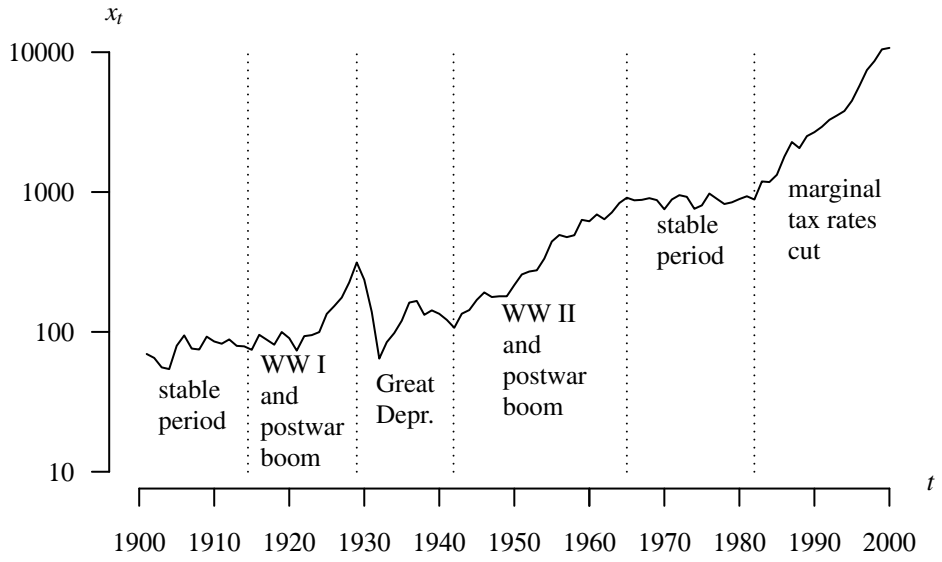

Figure 7.20: Dow Jones Industrial Average (1901–2000) with logarithmic scale.

on a linear vertical scale. An equal percent change now covers the same vertical distance with the logarithmic vertical scale. Labels have been added to help highlight events that might have influenced the DJIA.

The stock market crash in October of 1929 is much more pronounced in Figure 7.20. The DJIA had peaked with a close of 381.20 on September 3, 1929. The market bottomed out on July 8, 1932 when it closed at 41.20, which corresponds to a loss of almost 90%. Each of the two World Wars fought during the twentieth century was followed by a sustained bull market in the DJIA. The top marginal income tax rate was lowered from 70% to 28% and the federal budget was brought into balance in the 1980s and 1990s, resulting in a prolonged growth in the DJIA.

Detrending Filters

When a consistent trend in a time series is apparent, as was the case with the international airline passengers time series from Example 7.2, an analyst typically would like to estimate the trend. Once the trend has been estimated, the residual time series remaining after detrending can often be fitted to a time series model. There are two popular types of filters that can be used to detrend a time series: curve fitting and differencing. These will be considered individually. Time series analysts often use the term *secular trend* to describe a long-term, non-periodic trend, but we will refer to it as just a trend here.

One way to detrend a time series is to fit a curve that approximates the mean value of the time series. As a simple example, consider a time series that appears to have a linear trend. In this case a simple linear regression statistical model

$$X_t = \beta_0 + \beta_1 t + \varepsilon_t$$

can be fitted to the time series in order to estimate the slope β_0 and intercept β_1 of the regression line. The time t plays the role of the predictor in the regression model; the time series observations X_t play the role of the response in the regression model. It is also possible to have a non-linear trend in a time series. A quadratic trend in time, for example, could be modeled via

$$X_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \varepsilon_t.$$

Note that this model is linear in the β parameters. The statistical models used to detrend a time series are not limited to just polynomials in time. It is also possible to have an exponential model such as

$$X_t = \beta_0 e^{\beta_1 t} + \varepsilon_t.$$

This model is not linear in the β parameters. The potential statistical models are endless. A working knowledge of regression modeling is helpful in constructing an appropriate model for formulating, estimating, and assessing a model for the trend in a time series.

Example 7.26 This example considers the simplest case of detrending a data set, which is a linear trend. The data set consists of $n = 89$ quarterly observations which are the quarterly number of Australian residents (in thousands) from March 1971 to March 1993 which was first encountered in Example 7.18. This time series is built into R and has the name `austres`. Use simple linear regression to estimate the trend in the data set and calculate the detrended time series.

The raw data values and the time series plot are given in Example 7.18. The time series plot is repeated in Figure 7.21 for convenience, plotting individual points but not connecting them with lines. It is clear that the population of Australia is increasing in a linear fashion over this time period.

The next step is to fit a simple linear regression model to the time series model

$$X_t = \beta_0 + \beta_1 t + \varepsilon_t.$$

This can be accomplished in R using the `lm` (for *linear model*) function.

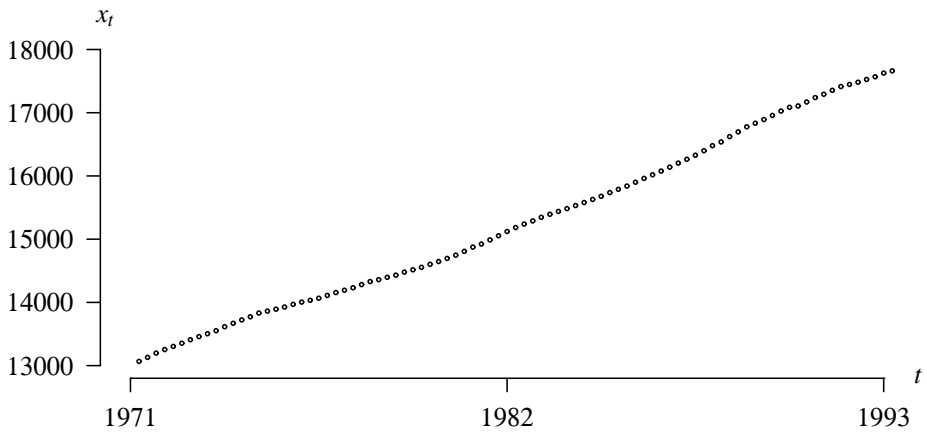

Figure 7.21: Quarterly population of Australia (in thousands) 1971–1993.

```
plot.ts(austres, type = "p", cex = 0.4)
fit = lm(austres ~ seq(1971.25, 1993.25, by = 0.25))
abline(fit$coefficients)
coef(fit)
```

The fitted slope and intercept of the least square regression line are

$$\hat{\beta}_0 = -399{,}861 \qquad \text{and} \qquad \hat{\beta}_1 = 209.426.$$

The interpretation of the estimated intercept is that in the year 0 the population of Australia was negative 400 million. (This is a good illustration that the model should not be extrapolated significantly outside of the range of the time values in the time series.) The interpretation of the estimated slope is that the population of Australia increases by an estimated 209,426 each year over the time window 1971–1993. The plot that includes the regression line plotted via the `abline` function is given in Figure 7.22. The regression line reveals some very slight nonlinear trends in the time series that were not immediately apparent in the original time series plot in Figure 7.21.

The final step in the analysis is to examine the residual time series after detrending. Viewing that residual series as a filter, the new time series after detrending is

$$y_t = x_t - \left(\hat{\beta}_0 + \hat{\beta}_1 t\right).$$

The time series $\{y_t\}$ can be calculated and plotted with the additional R statements

```
austres.detrend = austres - fit$fitted
plot.ts(austres.detrend)
```

This residual series is plotted in Figure 7.23. The residual series is connected by lines. In addition, a horizontal dashed line is added at $y_t = 0$. Clearly, the residual series does not consist of mutually independent noise terms. Its shape might have been influenced by Australian immigration policies or the Australian economy between 1971 and 1993.

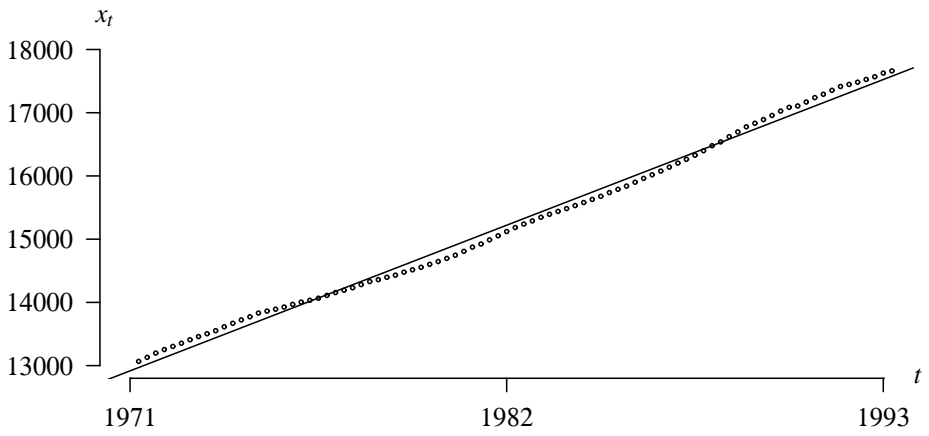

Figure 7.22: Quarterly population of Australia (in thousands) 1971–1993 with regression line.

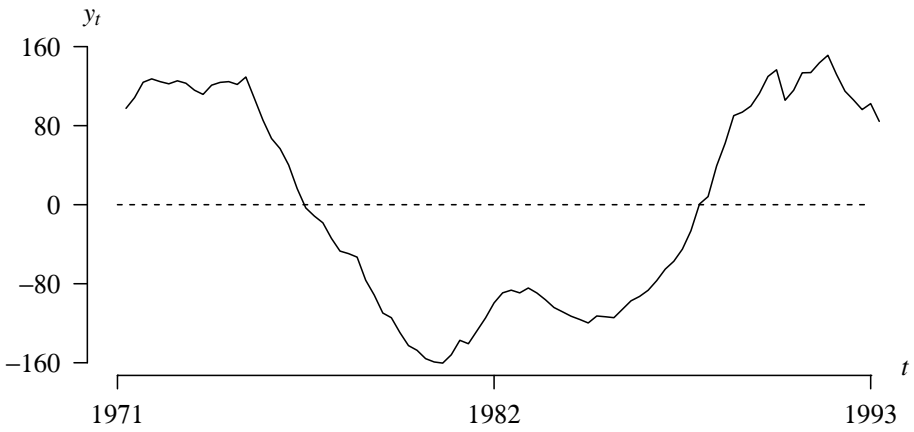

Figure 7.23: Residual time series after detrending for the quarterly population of Australia.

A second way to detrend a time series is to use differencing. The difference operator ∇, is defined as a filter by

$$y_t = \nabla x_t = x_t - x_{t-1} = (1-B)x_t,$$

where B is the backshift operator defined by $Bx_t = x_{t-1}$. Differencing a time series is the discrete analog of taking a derivative of a continuous function. So a time series that exhibits a linear trend, for example, will pass through the differencing filter ∇ and result in a time series without a trend. Notice that there will be one fewer observation in the new time series after applying this filter. If the original time series observations are $x_1, x_2, \ldots, x_n$, then the differenced series will be $y_2, y_3, \ldots, y_n$. Likewise, a time series with a quadratic trend can be detrended by applying the differencing operator ∇ twice to the original time series:

$$y_t = \nabla^2 x_t = \nabla\big(\nabla x_t\big) = \nabla\big(x_t - x_{t-1}\big) = \big(x_t - x_{t-1}\big) - \big(x_{t-1} - x_{t-2}\big) = x_t - 2x_{t-1} + x_{t-2}.$$

This detrending filter can be written with the backshift operator as $y_t = (1-2B+B^2)x_t$. There will be two fewer observations in the new time series after applying this filter. If the original time series observations are $x_1, x_2, \ldots, x_n$, then the twice differenced series will be $y_3, y_4, \ldots, y_n$.

A time series that exhibits a seasonal component can have a seasonal differencing filter applied. Monthly observations from a time series with an annual seasonal component, for example, can have the seasonal differencing filter

$$y_t = \nabla_{12} x_t = x_t - x_{t-12} = \big(1 - B^{12}\big)\, x_t$$

applied to eliminate the seasonal effects. There will be 12 fewer observations in the new time series after applying this filter. If the original time series observations are $x_1, x_2, \ldots, x_n$, then the time series observations that result from applying the filter associated with the ∇_{12} operator are $y_{13}, y_{14}, \ldots, y_n$.

We now illustrate the application of a differencing filter. The next example applies a single difference filter to the Australian population time series.

Example 7.27 Consider again the quarterly population of Australia between 1971 and 1993 from Example 7.26 given in the R built-in data set `austres`. Apply the single

difference filter $y_t = \nabla x_t = x_t - x_{t-1}$ and make a time series plot of the differenced time series.

The filter

$$y_t = \nabla x_t = x_t - x_{t-1}$$

is appropriate for detrending because the time series $\{x_t\}$ is approximately linear, as seen in Figure 7.22. The `diff` function in R differences the time series. So the differenced time series $\{y_t\}$ can be plotted with the single R statement

```
plot.ts(diff(austres))
```

The plot of the differenced time series is given in Figure 7.24. A dashed horizontal line has height equal to the slope of the line (with respect to quarters) connecting the first and last points in the time series [that is, a horizontal line at $(x_n - x_1)/(n-1) = 52.2$] can be added with the `abline` function with an `h` (for horizontal) and `lty = 2` (for a dashed line) arguments.

```
n = length(austres)
abline(h = (austres[n] - austres[1]) / (n - 1), lty = 2)
```

The original time series increases by $4 \cdot 52.2 = 208.8$, or $208,800$ residents annually. This is roughly equal to the slope of the regression line $\hat{\beta}_1 = 209.4$, or an increase of $209,400$ residents annually from Example 7.26. The filtered time series $\{y_t\}$ does not appear to exhibit any long-term trend, which was the original purpose of using the differencing filter.

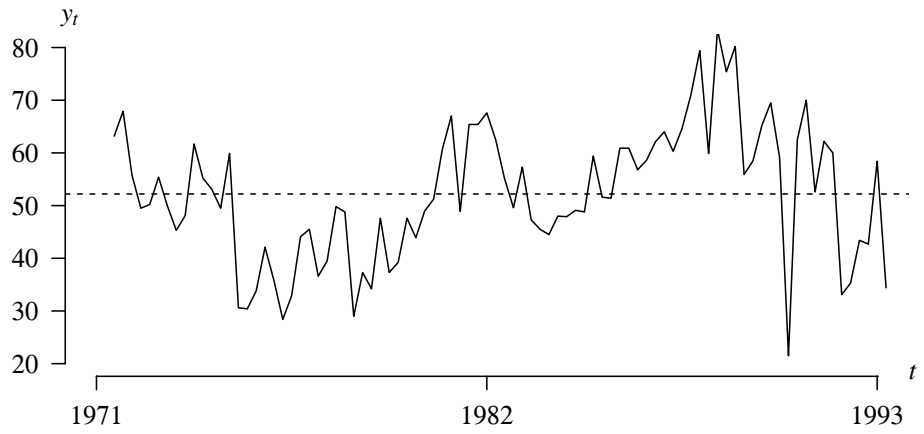

Figure 7.24: Filtered time series after differencing the quarterly population of Australia.

So far, two classes of filters have been introduced. The first class is known as a transformation; the second class is known as a detrending filter. Two types of detrending filters were introduced: curve fitting and differencing. We now turn to a third class of filter which is known as a linear filter. Differencing is a special case of a linear filter.

Linear Filters

Linear filters provide a wide array of options for converting the original time series $\{x_t\}$ to another time series $\{y_t\}$. The general form of a linear filter is

$$y_t = \sum_{s=t_0}^{t_1} c_s x_{t+s},$$

where the coefficients c_s are real-valued constants. It is often the case that $t_0 < 0$ and $t_1 > 0$, which means that the time series $\{y_t\}$ is a linear combination of the chronological *current*, *previous*, and *future* values of $\{x_t\}$ in time.

One purpose of a linear filter is *smoothing* the original time series by using what is known as a *moving average*. When the coefficients sum to one, written symbolically as

$$\sum_{s=t_0}^{t_1} c_s = 1,$$

this linear filter is a moving average. One elementary example of a *symmetric moving average* is when $t_0 = -t_1$ with identical weights

$$c_s = \frac{1}{2t_1 + 1}.$$

In this case, the smoothed time series $\{y_t\}$ is the arithmetic mean of

- the current value of the time series $\{x_t\}$,
- the t_1 previous values of the time series $\{x_t\}$,
- the t_1 future values of the time series $\{x_t\}$,

for a total of $2t_1 + 1$ values averaged. The symmetric moving average $\{y_t\}$ will have $2t_1$ fewer observations than the original time series $\{x_t\}$ because the average cannot be computed for the first and last t_1 observations in $\{x_t\}$. The smoothed values of the first 100 Dow Jones Industrial Average closing values during the year 2000, introduced in Example 7.4, will be illustrated next.

Example 7.28 Consider the time series $\{x_t\}$ consisting of the first $n = 100$ closing values of the Dow Jones Industrial Average during the year 2000 that appeared in the top graph of Figure 7.5. Graph the original time series $\{x_t\}$ and the linear filter which is a symmetric moving average of five adjacent values (that is, $t_1 = 2$)

$$y_t = \frac{x_{t-2} + x_{t-1} + x_t + x_{t+1} + x_{t+2}}{5}.$$

Notice that the coefficients in this linear filter are all $1/5$. This symmetric moving average is sometimes known as a *five-point moving average*.

The first step is to write an R function to compute the five-point moving average. The R function `movingAverage5` given below calculates the five-point moving average.

```
movingAverage5 = function(x) {
  n = length(x)
  (x[1:(n - 4)] + x[2:(n - 3)] + x[3:(n - 2)] + x[4:(n - 1)] + x[5:n]) / 5
}
```

The original time series of Dow Jones Industrial Averages and the five-point moving average are plotted in Figure 7.25. The original time series $\{x_t\}$ is plotted as a solid black line connecting the points. The linear filter $\{y_t\}$ smooths the original time series and is given by the thicker gray curve, which is a piecewise linear function that connects the five-point moving average points, given as dots within the gray curve. The original time series consists of $n = 100$ points. The five-point moving average loses two points at the beginning and two points at the end, resulting in just the points $y_3, y_4, \ldots, y_{98}$. The moving average successfully smooths the original time series. By averaging the current value, two previous values, and two future values, the significant variations in the original time series are damped, and the trend of the Dow Jones Industrial during the year 2000 is more apparent with the five-point moving average.

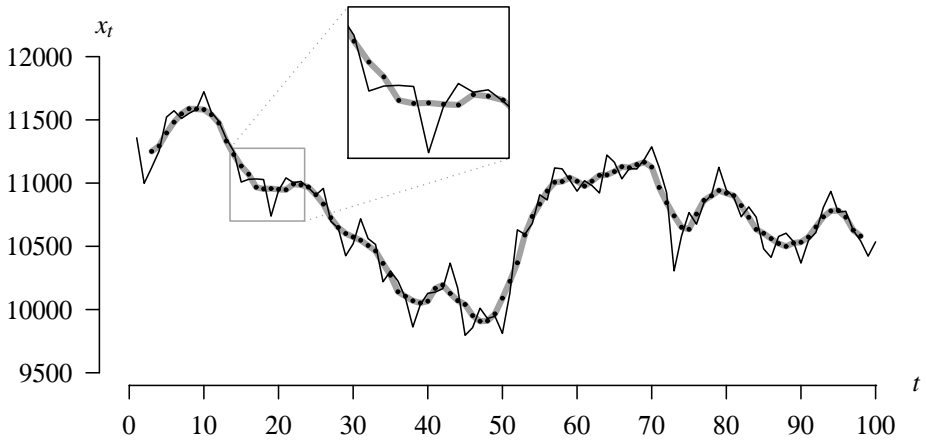

Figure 7.25: The first $n = 100$ DJIA closes in 2000 and a five-point moving average.

We now illustrate the case in which filters are applied to a time series in a serial fashion as illustrated below.

$$\{x_t\} \xrightarrow{\text{filter 1}} \{y_t\} \xrightarrow{\text{filter 2}} \{z_t\}$$

Let the values in the original time series be

$$x_1, x_2, \ldots, x_n.$$

Consider the linear filter which is a three-point moving average (in which $t_1 = 1$)

$$y_t = \frac{x_{t-1} + x_t + x_{t+1}}{3}.$$

This linear filter results in the time series $\{y_t\}$ consisting of the observations

$$\frac{x_1 + x_2 + x_3}{3}, \frac{x_2 + x_3 + x_4}{3}, \ldots, \frac{x_{n-2} + x_{n-1} + x_n}{3}.$$

Now consider applying this same linear filter again, but this time to $\{y_t\}$:

$$z_t = \frac{y_{t-1} + y_t + y_{t+1}}{3}.$$

The second linear filter results in the time series $\{z_t\}$ consisting of the observations

$$\frac{x_1+2x_2+3x_3+2x_4+x_5}{9}, \frac{x_2+2x_3+3x_4+2x_5+x_6}{9}, \ldots, \frac{x_{n-4}+2x_{n-3}+3x_{n-2}+2x_{n-1}+x_n}{9}$$

when written in terms of the original time series $\{x_t\}$. Notice that the serial application of the two linear filters is the same as the application of a single linear filter with the coefficients

$$\frac{1}{9}, \frac{2}{9}, \frac{3}{9}, \frac{2}{9}, \frac{1}{9}.$$

The R `convolve` function calculates the coefficients of the linear filter associated with two linear filters applied to a time series. In the example described here, the coefficients can be determined with the R statements

```
a = c(1 / 3, 1 / 3, 1 / 3)
b = c(1 / 3, 1 / 3, 1 / 3)
convolve(a, b, type = "open")
```

The application of two linear filters in sequence is illustrated in the next example.

Example 7.29 Consider again the time series $\{x_t\}$ consisting of the first $n = 100$ closing values of the Dow Jones Industrial Average during the year 2000. Graph the original time series $\{x_t\}$ and two serial applications of the five-point moving average

$$y_t = \frac{x_{t-2}+x_{t-1}+x_t+x_{t+1}+x_{t+2}}{5}.$$

As in the case of the three-point moving average, we can use the R `convolve` function to calculate the coefficients in the convolution of the two linear filters.

```
a = rep(1 / 5, 5)
convolve(a, a, type = "open")
```

The `convolve` function results in the coefficients

$$\frac{1}{25}, \frac{2}{25}, \frac{3}{25}, \frac{4}{25}, \frac{5}{25}, \frac{4}{25}, \frac{3}{25}, \frac{2}{25}, \frac{1}{25}$$

associated with the two linear filters applied in series. These coefficients sum to one as expected. The convolution of the two moving average filters remains a moving average. The application of the two linear filters in series has two potential benefits over the five-point moving average alone:

- the serial application of the two linear filters provides more smoothing than in the previous example because more observations are used in the moving average, and the coefficients are all smaller than in the five-point moving average, and
- the serial application of the two linear filters provides a mechanism in which more distant observations get less weight than nearby observations.

The result of applying this moving average to the Dow Jones Industrial Average closes is shown in Figure 7.26. As expected, this filter provides more smoothing than the five-point moving average from the previous example.

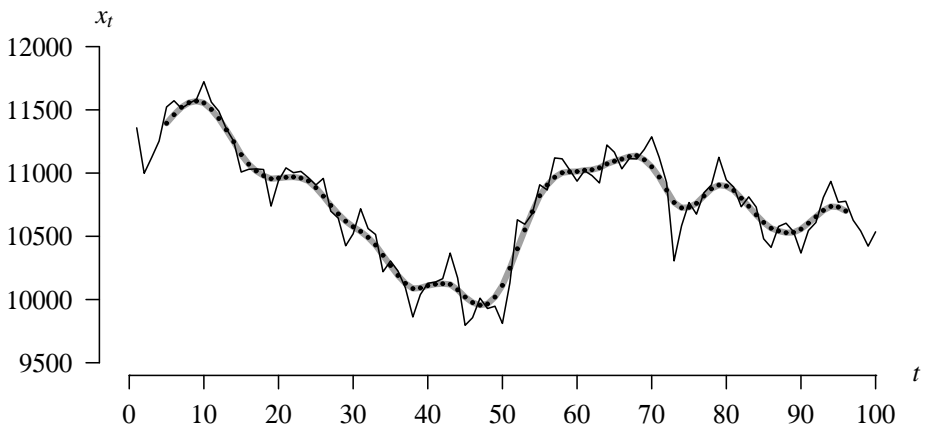

Figure 7.26: The first $n = 100$ DJIA closes in 2000 and a nine-point moving average.

Some experimentation is often necessary to achieve a moving average that provides the appropriate amount of smoothing. The amount of smoothing desired is problem specific. Another type of symmetric moving average that places the most weight on the current value x_t and decreasing weight to more distant observations is to use the terms in the expansion of

$$\left(\frac{1}{2}+\frac{1}{2}\right)^{2t_1}$$

as coefficients in a weighted moving average. When $t_1 = 2$, for example, the coefficients are

$$\frac{1}{16}, \frac{4}{16}, \frac{6}{16}, \frac{4}{16}, \frac{1}{16}.$$

The numerators can be recognized as one row in Pascal's triangle. The weights must sum to 1 because $1/2+1/2=1$.

For a time series without a trend that contains a seasonal component, a special linear filter can be applied. Consider a time series of monthly observations with seasonal variation. A common way to eliminate the seasonal component is the linear filter

$$y_t = \frac{\frac{1}{2}x_{t-6}+x_{t-5}+x_{t-4}+\cdots+x_{t+4}+x_{t+5}+\frac{1}{2}x_{t+6}}{12}.$$

This linear filter has 13 coefficients

$$\frac{1}{24}, \frac{1}{12}, \frac{1}{12}, \ldots, \frac{1}{12}, \frac{1}{12}, \frac{1}{24},$$

which places a weight of $1/12$ on the current observation x_t and each observation within five months of x_t and splits the weight between the two months that are six months before and six months after the current observation. Notice that the filtered time series $\{y_t\}$ will have 12 fewer observations than the original time series $\{x_t\}$ because the moving average loses six points at the beginning of the time series and six points at the end of the time series. This seasonal weighted average will be illustrated in the next example for the monthly home energy consumption time series.

Example 7.30 Consider the time series $\{x_t\}$ of $n = 96$ monthly home energy consumption observations, in kilowatt hours, from Example 7.1. Compute the seasonal moving average

$$y_t = \frac{\frac{1}{2}x_{t-6} + x_{t-5} + x_{t-4} + \cdots + x_{t+5} + x_{t+6} + \frac{1}{2}x_{t+6}}{12}$$

and plot the original time series and the seasonal moving average on the same set of axes.

The original time series and the seasonal moving average are graphed in Figure 7.27. The seasonal moving average effectively removes the seasonal component, revealing a slight upward trend in the first half of the time series and a slight downward trend toward the end of the time series. Since the time series was collected over an eight-year period, there are a total of $8 \cdot 12 - 12 = 84$ observations in the seasonal moving average. An observation from each of the 12 months plays a role in every value calculated in the seasonal moving average $\{y_t\}$.

The R code below calculates the seasonal moving average of the energy consumption values, which are stored in the file named `kwh.d`. The coefficients of the seasonal moving average which control the weights allocated to each value in the time series are stored in the `w` vector. The seasonal moving average values are stored in the `y` vector.

```
x = scan("kwh.d")
w = c(1 / 24, rep(1 / 12, 11), 1 / 24)
n = length(x)
y = numeric(n - 12)
for (i in 1:(n - 12)) y[i] = sum(w * x[i:(i + 12)])
print(y)
```

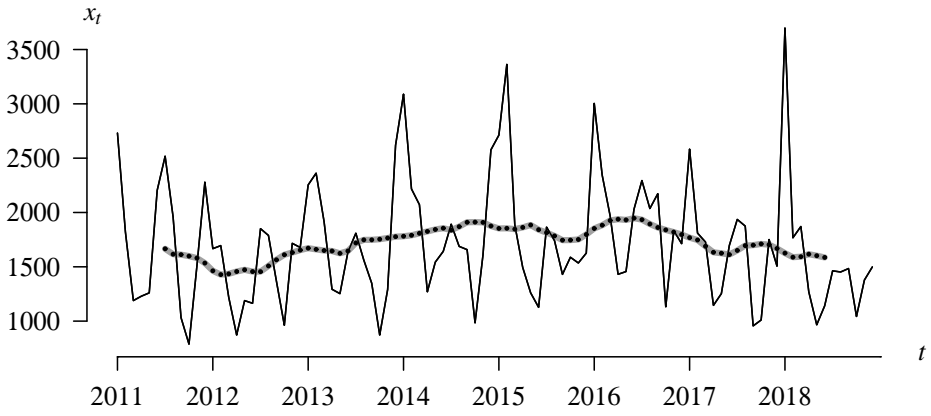

Figure 7.27: Home energy consumption time series and seasonal moving average.

All of the linear filters we have encountered so far have been *symmetric moving averages*. Each of them has reached as far into the past as they have into the future. One weakness of these filters is

that in many settings we often do not have any future observations. So in many practical problems the linear filter

$$y_t = \sum_{s=t_0}^{t_1} c_s x_{t+s}$$

is rewritten to avoid any observations in the future as

$$y_t = \sum_{s=t_0}^{0} c_s x_{t+s}$$

for some negative integer index t_0. The most well-known of these filters is known as the *exponentially-weighted moving average*, often abbreviated EWMA, which can be written as

$$y_t = \sum_{s=-\infty}^{0} c_s x_{t+s},$$

where the weights c_s are given by

$$c_s = \alpha(1-\alpha)^{-s}$$

for $0 < \alpha < 1$ and $s = -\infty, \ldots, -2, -1, 0$. The weights in the exponentially-weighted moving average must sum to one because they form a geometric series:

$$\sum_{s=-\infty}^{0} c_s = \sum_{s=-\infty}^{0} \alpha(1-\alpha)^{-s} = \alpha \sum_{s=0}^{\infty} (1-\alpha)^s = \frac{\alpha}{1-(1-\alpha)} = 1.$$

The exponentially-weighted moving average gives weight α to the current observation, and then geometrically declining weights to previous observations. So the parameter α can be thought of as a dial or tuning parameter which controls the amount of smoothing. Large values of α mean very little smoothing; small values of α mean significant smoothing. While it is daunting to think about values of a time series $\{x_t\}$ running back in time to $-\infty$, it is possible to avoid the infinite summation. The exponentially-weighted moving average filter can be rewritten as

$$\begin{aligned} y_t &= \alpha x_t + \alpha(1-\alpha)x_{t-1} + \alpha(1-\alpha)^2 x_{t-2} + \cdots \\ &= \alpha x_t + (1-\alpha)\left[\alpha x_{t-1} + \alpha(1-\alpha)x_{t-2} + \cdots\right] \\ &= \alpha x_t + (1-\alpha) y_{t-1}; \end{aligned}$$

that is, the exponentially-weighted moving average is α times the current value in the time series x_t plus $1-\alpha$ times the previous value in the moving average. Arbitrarily setting $y_1 = x_1$ to initiate this recursive relationship, the initial terms in the moving average are

$$\begin{aligned} y_2 &= \alpha x_2 + (1-\alpha) y_1 \\ y_3 &= \alpha x_3 + (1-\alpha) y_2 \\ &\vdots \qquad\qquad \vdots \end{aligned}$$

Notice that the extreme case of $\alpha = 1$ is possible in this recursive equation, and this corresponds to a moving average that is identical to the original time series: $y_1 = x_1$, $y_2 = x_2$, $y_3 = x_3$, etc. This extreme case corresponds to no smoothing at all.

The next example applies the exponentially-weighted moving average to the first $n = 100$ Dow Jones Industrial Average closing observations during the year 2000.

Example 7.31 Consider yet again the time series $\{x_t\}$ consisting of the first $n = 100$ closing values of the Dow Jones Industrial Average during the year 2000. Graph the original time series $\{x_t\}$ and the exponentially-weighted moving average with $\alpha = 0.2$ on the same set of axes.

The original time series $\{x_t\}$ and the exponentially-weighted moving average $\{y_t\}$ are plotted in Figure 7.28. The exponentially-weighted moving average values are generated by

$$y_t = \alpha x_t + (1-\alpha)y_{t-1} = 0.2x_t + 0.8y_{t-1}$$

for $t = 2, 3, \ldots, n$. The first point in the exponentially-weighted moving average, y_2, for example, is a convex combination of x_1 and x_2 with coefficients 0.8 and 0.2. Unlike the symmetric moving averages shown in the previous examples, Figure 7.28 shows that the exponentially-weighted moving average smooths the original time series, but also lags the original time series because it is not a symmetric moving average. Adjusting α can make this exponentially-weighted moving average respond more quickly or more slowly than that shown in Figure 7.28.

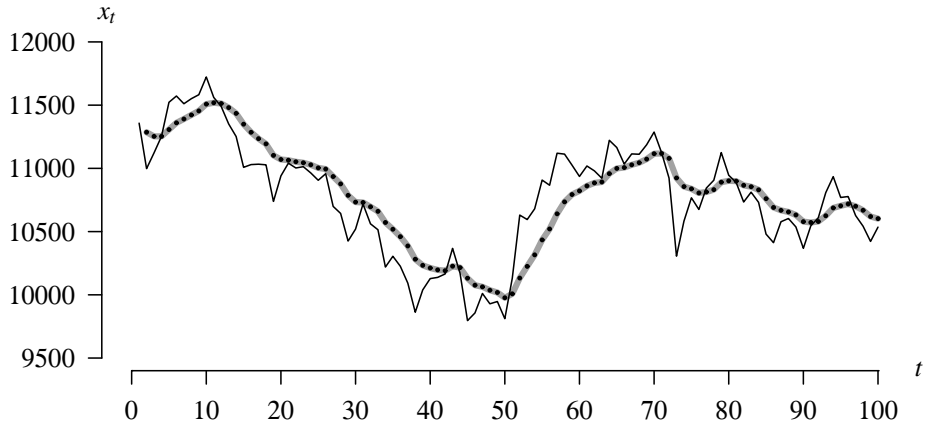

Figure 7.28: The first $n = 100$ DJIA closes in 2000 and an exponentially-weighted moving average.

This concludes the discussion of filters that can be applied to a time series. The three classes of filters that were presented in this section are

- transformations,
- detrending filters, and
- linear filters.

Filters can be applied for several different purposes, including: (a) to stabilize the variance of a time series whose variance increases with time, (b) to stabilize the variance of a time series in which larger values are more variable than smaller values, (c) to make error terms look approximately normally distributed, (d) to express a time series with a seasonal component as an additive model, (e) to detrend a time series containing a trend, (f) to estimate and eliminate a seasonal component

in a time series without a trend, and (g) to smooth a time series. Many of these reasons for using a filter on a time series will be discussed subsequently.

The next section uses two filters in series in order to decompose a time series into trend and seasonal components. Decomposition will be applied to the international air travel time series and the home energy consumption time series from the first two examples in this chapter.

7.3.2 Decomposition

The time series plots on the four examples from Section 7.1.1 have revealed certain types of patterns that appear in many time series. The partial list below contains the most common types of variation in time series. One common approach to decomposing a time series into these various types of variation is to remove the detected types of variation one-by-one until only noise (that is, random variation) remains.

- **Trend.** The time series consisting of the number of international airline passengers between 1949 and 1961 from Example 7.2 had a clear and obvious long-term systemic increase in its mean value as air travel became more popular over that time period. Detecting trends and including them in a time series model is an important part of the analysis of a time series that will be illustrated in the next example.

- **Seasonal variation.** Both the home energy consumption time series from Example 7.1 and the international airline passenger time series from Example 7.2 exhibit seasonal variation. The period associated with the seasonal variation in both cases was one year. A single time series is capable of having multiple seasonal variation cycles. Outdoor temperature, for example, has both an annual cycle (warmer during the summer and cooler during the winter) and a daily cycle (warmer during the day and cooler during the night). The frequency of the daily cycle is 365 times greater than the frequency of the annual cycle. (To be more careful, the frequency is actually 365.2422 times greater.)

- **Other cyclical variation.** There are other types of cyclical variation that have an unknown period that might be included in a mathematical model that describes the time series. Economists, for example, often refer to *business cycles* that might influence the values in a time series. Business cycles typically have a varying and unknown period generally ranging from a few years to decades.

- **Remaining variation.** Once the trend, seasonal variation, and other cyclical variation have been removed from the original time series, a time series with no trend, seasonal, or cyclic variation is obtained. Once this new time series is obtained, it is common practice to plot these values to see how closely they approximate noise terms. The final step in constructing a time series model for the original process is often fitting a time series model to the residual time series, which reflects the noise terms in the time series model.

We now consider mathematical models for decomposing a time series into these constituent parts. Just as a probability model like the normal or exponential distribution is used to approximate the probability distribution from which a data set is drawn in classical statistics, we want to develop a probability model for the time series $\{X_t\}$. This probability model will be more complicated than the random walk model because we would like it to include both trends and seasonal variation. An *additive model* to describe $\{X_t\}$ is

$$X_t = m_t + s_t + \varepsilon_t,$$

where the m_t term models the trend, the s_t term models the seasonal variation with fixed period, and the ε_t term models the noise. In an ideal probabilistic modeling sequence, once the trend and seasonality terms have been estimated and removed from the time series, only random variation remains. We have ignored other cyclic variation (for example, business cycles) in this particular mathematical model. Notice that the trend term m_t and the seasonal variation term s_t are set in lowercase because these are assumed to be deterministic functions of t in the model. The stochastic element of the time series model comes from the ε_t term. A *multiplicative model* to describe $\{X_t\}$ is

$$X_t = m_t \cdot s_t \cdot \varepsilon_t.$$

The multiplicative model is often appropriate in the case in which the variance of the time series increases with time. This is because taking the natural logarithm of both sides of this model yields

$$\ln X_t = \ln m_t + \ln s_t + \ln \varepsilon_t,$$

which is an additive model for the time series $\{\ln X_t\}$.

Decomposing a time series into its constituent parts can be preformed in R with the `decompose` function. The additive model is the default. The next example applies the `decompose` function to the time series consisting of the international airline passengers.

Example 7.32 Consider again the time series of the number of international airline passengers (in thousands) between 1949 and 1961 from Example 7.2. Decompose this time series into its trend, seasonal, and random components using the R `decompose` function.

Since the plot of the time series in Figure 7.2 shows that the variance of the time series is increasing with time, we elect to use the multiplicative model

$$X_t = m_t \cdot s_t \cdot \varepsilon_t.$$

The call to the `decompose` function applied to the `AirPassengers` built-in data set is

```
fit = decompose(AirPassengers, type = "multiplicative")
```

The fitted model that extracts the trend and seasonal components, and calculates the residual time series once those two components are extracted, is held in a list named `fit`. The `type` argument in the call to the `decompose` function is used to invoke the multiplicative model. A plot of the original time series and its decomposition into its component parts is obtained by the additional R statement

```
plot(fit)
```

The associated plot is given in Figure 7.29, which gives four time series stacked one above the another. The first time series, labeled *observed*, is the original time series $\{x_t\}$, the number of monthly international airline passengers (in thousands). The lowercase variable name x_t is used here because these are observed values of the time series. The second time series, labeled *trend*, is the estimate of the trend $\{m_t\}$ returned by the `decompose` function. The third time series, labeled *seasonal*, is the estimate of the seasonal component of the multiplicative model $\{s_t\}$ returned by the `decompose` function. The seasonal component is identical from one year to the next with period 12 extracted

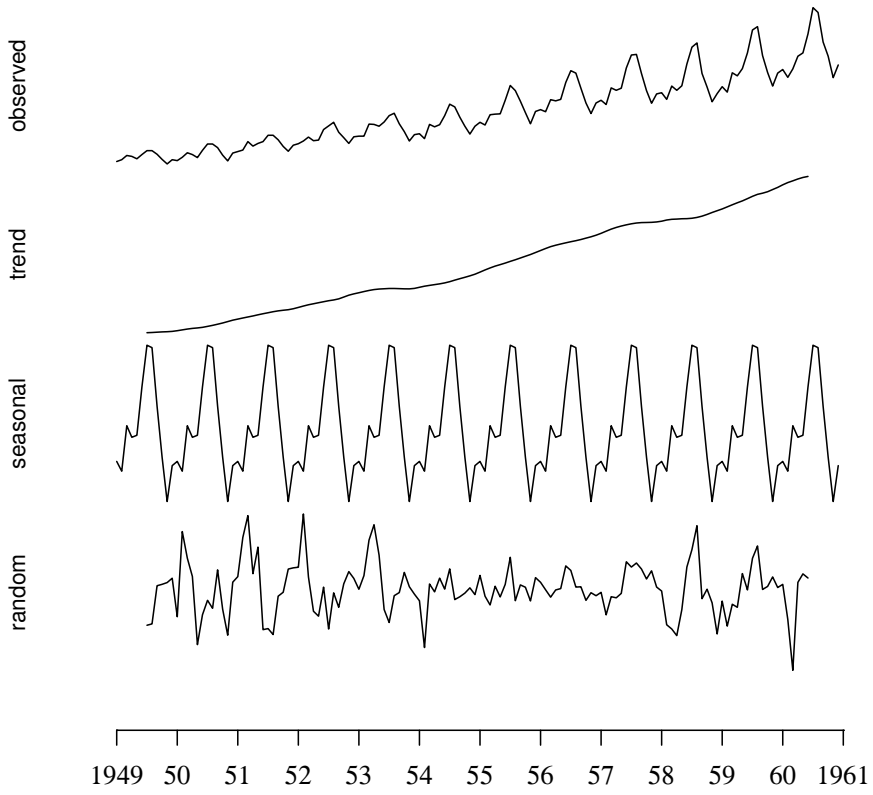

Figure 7.29: Decomposition of the international airline passengers time series.

from the `AirPassengers` time series. Finally, the fourth time series, labeled *random*, is the remaining time series $\{e_t\}$ once the trend and seasonal components of the model have been removed.

A call to the `str` (structure) function

```
str(fit)
```

reveals that the object named `fit` is a list that consists of six components. One of these components is named `seasonal`, which contains the seasonal components of the time series. This means that `fit$seasonal` is a time series, which is identical over every year in the time series. In order to investigate the seasonal component, Figure 7.30 contains one cycle of the seasonal component of the decomposed model. The additional R statement which can be used to plot the first cycle of the seasonal component is given below. The subscripts `1:12` extract the values in the first annual cycle.

```
plot.ts(fit$seasonal[1:12])
```

The resulting graph displayed in Figure 7.30 shows that the largest number of international airline passengers during the years from 1949 through 1960 occurs in the months of July and August when school is typically not in session. It also shows a local maximum in March which might correspond to families traveling over spring break week. The global minimum occurs in the month of November.

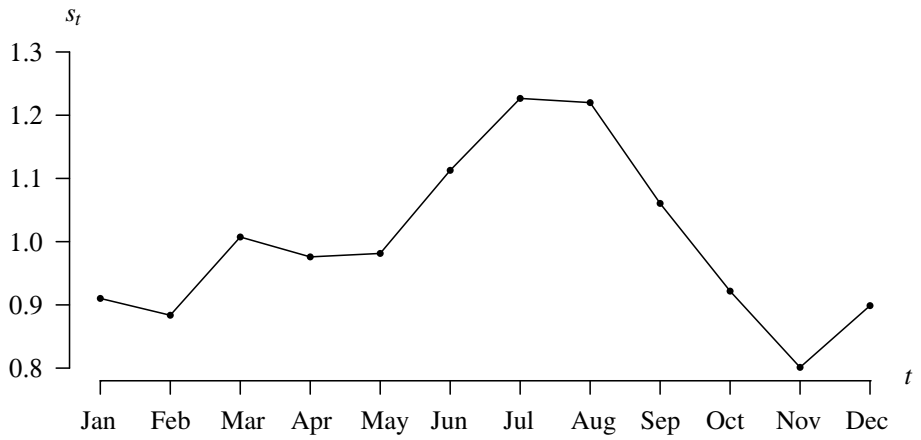

Figure 7.30: Seasonal component of the international airline passengers time series.

The international airline passengers data set had its own distinct signature for its seasonal component. The next example plots the analogous seasonal component for the home energy consumption data.

Example 7.33 Consider the home energy consumption data from Example 7.1. Using similar code to the previous example, we can plot the seasonal component of the time series with the R statements below.

```
kwh    = scan("kwh.d")
kwh.ts = ts(kwh, frequency = 12, start = c(2011, 1))
fit    = decompose(kwh.ts)
plot.ts(fit$seasonal[1:12])
```

The first statement reads the monthly energy consumption observations into a vector named `kwh`. The second statement uses the R `ts` function to convert the observations into a time series. The third statement uses the R `decompose` function to decompose the time series into its constituent trend, seasonal, and remaining variation components using an additive model (the default). The last statement uses the R `plot.ts` function to plot the first 12 elements of the seasonal component of the decomposed time series. The resulting graph shown in Figure 7.31 reveals a distinctly different seasonal pattern than the associated graph for the international airline passengers data set. The peak energy consumption is clearly in January. This peak is consistent with the intuition for the time series because (a) the outdoor temperature in the winter in Williamsburg is further from a comfortable indoor temperature than the outdoor in the summer in Williamsburg, and

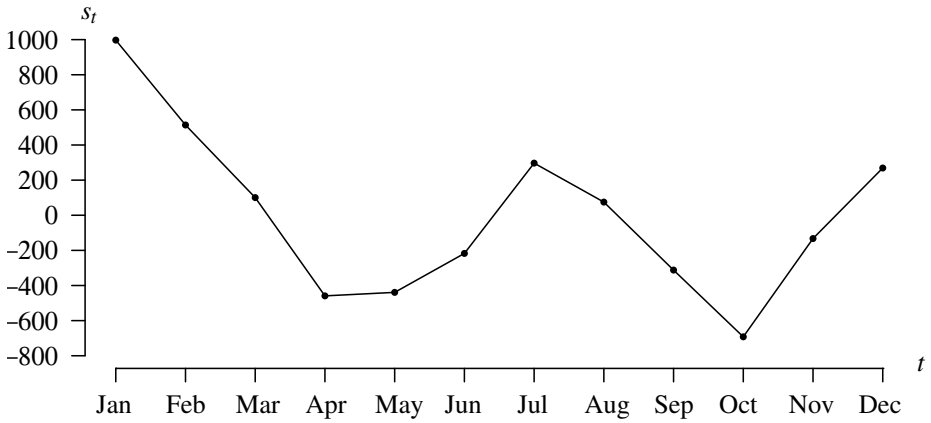

Figure 7.31: Seasonal component of the home energy consumption time series.

(b) the heat pumps are more energy efficient in the summer than they are in the winter because there is more heat available to capture. Not surprisingly, the two peaks in the time series occur in the winter and summer when demand on the heat pumps is the greatest. Since an additive model was selected, the units on the vertical axis are in kilowatt hours. The energy consumption bump associated with the number of kilowatt hours consumed in January, for example, is about 1000 kilowatt hours above the average monthly energy consumption over the entire eight-year period.

Recall that the additive model is

$$X_t = m_t + s_t + \varepsilon_t$$

and the `decompose` function has provided a fitted time series for the trend component m_t, the seasonal component s_t, and the error component ε_t. This indicates that the original time series can be reconstructed with the R statement

```
fit$trend + fit$seasonal + fit$random
```

which yields the original time series `fit$x`, with the exception of some `NA` values at the extreme values, which will be investigated next.

It is important to not treat a function like `decompose` as a just a black box that decomposes a time series without knowing the internal workings of the function. It is crucial to know exactly what is going on inside of `decompose` for (a) proper interpretation of the output of the function, and (b) the ability to modify the function. In that light, we now show the intermediate steps that occurred in `decompose` using the object names in `decompose` that resulted in the values plotted in Figure 7.31.

```
kwh      = scan("kwh.d")
kwh.ts   = ts(kwh, frequency = 12, start = c(2011, 1))
l        = length(kwh.ts)
```

```
f        = frequency(kwh.ts)
trend    = filter(kwh.ts, c(0.5, rep(1, f - 1), 0.5) / f)
season   = kwh.ts - trend
periods  = l / f
index    = seq(1, l, by = f) - 1
figure   = numeric(f)
for (i in 1:f) figure[i] = mean(season[index + i], na.rm = TRUE)
figure   = figure - mean(figure)
seasonal = ts(rep(figure, periods + 1)[seq(l)], start = start(kwh.ts),
              frequency = f)
```

Try typing these statements into R and viewing the resulting objects. The bullet points below give a line-by-line explanation of the algorithm associated with this R code.

- The first statement reads the home energy consumption time series into the vector `kwh`.
- The second statement converts the vector named `kwh` to a time series named `kwh.ts` with monthly values beginning in January of 2011 via the `ts` function.
- The third statement calculates the length of the time series as $l = 96$.
- The fourth statement extracts the frequency of the time series as $f = 12$.
- The fifth statement uses the `filter` function to apply a 13-point moving average to the original time series, which results in a time series named `trend`. The extreme values in this moving average are identical months separated by one year, each getting weight $1/24$, and the interior 11 months each get weight $1/12$. Notice that the first six and last six values of the resulting time series `trend` are `NA`, as expected. The 13-point moving average is first reported in July of 2011, using the 13 values from the original time series from January 2011 to January 2012. Each value in `trend` is effectively an annual average of energy consumption, so this is a fairly flat time series for the energy consumption time series data because there does not appear to be any significant trend. The values in `trend` are plotted in Figure 7.27.
- The sixth statement creates a time series named `season` which is the difference between the original time series `kwh.ts` and the time series `trend`. In time series analysis, this step is known as *detrending*. This new time series `season` isolates the empirical seasonal component, which will change from one year to the next. The remaining R statements are designed to average these seasonal components.
- The seventh statement calculates the number of periods in the original time series: $l/f = 96/12 = 8$.
- The eighth statement creates a vector named `index` that will be used in the calculation of the seasonal component averages. For the energy consumption data, the eight elements of `index` are $0, 12, 24, \ldots, 84$.
- The ninth statement uses the `numeric` function to initialize the 12-element vector named `figure`, which will contain the seasonal component averages.
- The tenth statement contains a `for` loop which uses the `mean` function to calculate the seasonal component averages for each of the 12 months.
- The eleventh statement centers these seasonal component averages around zero.

- The twelfth statement uses the `ts` function to create a time series named `seasonal` which contains 8 periods of the seasonal component averages with a frequency of $f = 12$ and a start time of January of 2011. These are the values that are plotted in Figure 7.31.

View the `decompose` function by simply typing

```
decompose
```

In addition to the R statements described above, you will see (a) error trapping at the top of the function, (b) several conditional statements to account for the additive and the multiplicative models, and (c) code that will adjust to a time series consisting of incomplete periods.

The figures that we have seen so far have plotted a time series $\{x_t\}$ or a filter applied to create another time series $\{y_t\}$. The critical initial step of plotting the time series and making a careful examination of the plot should never be skipped.

7.3.3 Computing

Regression can be used to fit a model to a time series using the built-in `lm` (linear model) function in R. This is illustrated for the `AirPassengers` time series below.

```
fit = lm(AirPassengers ~ time(AirPassengers))
plot.ts(AirPassengers)
abline(fit$coefficients)
fitted(fit)
```

The first statement sets the object `fit` to a list that contains the results of a simple linear regression of `time(AirPassengers)` as the independent variable (the predictor) and `AirPassengers` as the dependent variable (the response). The second statement plots the time series in the usual fashion using the `plot.ts` function. The third statement appends the plot with the regression line using the `abline` function. Finally, the fitted values can be extracted by the call to `fitted(fit)` as shown above or by using `fit$fitted.values`.

A second way to remove a trend from a time series $x_1, x_2, \ldots, x_n$ is differencing. The difference operator ∇ defined by

$$\nabla x_t = x_t - x_{t-1} = (1 - B)x_t$$

(where B is the backshift operator defined by $Bx_t = x_{t-1}$) can be used to remove a linear trend. The R function `diff` can be used to difference a time series. The following statement creates a time series that contains the differences between adjacent values in the `AirPassengers` time series.

```
diff(AirPassengers)
```

There will be one fewer observation in the differenced time series than in the original time series. A quadratic trend in a time series can be detrended by applying the differencing operator ∇ twice to the original time series:

$$\nabla^2 x_t = \nabla\big(\nabla x_t\big) = \nabla\big(x_t - x_{t-1}\big) = \big(x_t - x_{t-1}\big) - \big(x_{t-1} - x_{t-2}\big) = x_t - 2x_{t-1} + x_{t-2}.$$

This detrending filter can be written with the backshift operator as $(1 - 2B + B^2)x_t$. Second-order differences, denoted by $\nabla^2 x_t$, can be calculated by applying the `diff` function twice:

```
diff(diff(AirPassengers))
```

or by using the `differences` argument in the `diff` function:

```
diff(AirPassengers, differences = 2)
```

Monthly observations from a time series with an annual seasonal component, for example, can have the seasonal differencing filter

$$\nabla_{12} x_t = x_t - x_{t-12} = \left(1 - B^{12}\right) x_t$$

applied to eliminate the seasonal effects. The `lag` argument is added to the `diff` function in order to do this type of seasonal differencing.

```
diff(AirPassengers, lag = 12)
```

There will be 12 fewer observations in the new time series after applying this filter.

The backshift operator B, which is useful in writing differencing operations compactly, is defined by $Bx_t = x_{t-1}$, or more generally as $B^m x_t = x_{t-m}$. A single application of the B operator can be achieved by a call to the `lag` function.

```
lag(AirPassengers)
```

The first observation in the resulting time series is December of 1948. The time series has simply been shifted back in time by one month. To apply the B operator twice, denoted by B^2, the second argument should be set to 2.

```
lag(AirPassengers, 2)
```

The first observation in the resulting time series is November of 1948. In order to shift a time series forward in time, a negative value for the second argument is used.

```
lag(AirPassengers, -3)
```

The first observation in the resulting time series is April of 1949.

The intersection of several time series can be achieved with the `ts.intersect` function, which is illustrated below.

```
ts.intersect(lag(AirPassengers), AirPassengers, lag(AirPassengers, -1))
```

The elements of the resulting intersection of the three time series will only be defined on common time values. A related function named `ts.union`, will take the union of the constituent time series, appending an `NA` to positions in any of the constituent time series without observations.

Occasions arise in which only a subset of a time series is of interest. The `window` function shown below extracts the portion of the `AirPassengers` time series between January of 1951 and June of 1957.

```
window(AirPassengers, start = c(1951, 1), end = c(1957, 6))
```

A linear filter can also be applied to a time series using the `filter` function. The example below calculates a 12-point moving average of the `AirPassengers` time series.

```
filter(AirPassengers, filter = rep(1 / 12, 12), sides = 1)
```

A time series can be decomposed into trend, seasonal, and random components using the `decompose` function.

```
decompose(AirPassengers)
```

These components can be viewed by embedding this command into the plot function.

```
plot(decompose(AirPassengers))
```

This provides only a graph of the four components. The `decompose` function provides a rather elementary way of decomposing time series. A more sophisticated approach using the loess (locally estimated scatterplot smoothing) method is the `stl` (which is an abbreviation for seasonal decomposition of a time series by loess) function.

```
stl(AirPassengers, s.window = "periodic")
```

7.4 Exercises

7.1 White noise (*WN*), iid noise (*IID*), and Gaussian white noise (*GWN*) were introduced in Definition 7.1. The three classes are related by

$$GWN \subset IID \subset WN.$$

Indicate the strongest class of noise associated with the following three time series.

(a) $X_1, X_2, \ldots, X_n$ are mutually independent and identically distributed $N(0, 1)$ random variables.

(b) $X_1, X_2, \ldots, X_n$ are mutually independent random variables with $X_t \sim N(0, 1)$ when t is even and $X_t \sim U\left(-\sqrt{3}, \sqrt{3}\right)$ when t is odd.

(c) $X_1, X_2, \ldots, X_n$ are mutually independent and identically distributed $U(-2, 2)$ random variables.

7.2 Let $X_1, X_2, \ldots, X_n$ be n observations from the random walk model described in Example 7.4. Find $V\left[\bar{X}\right]$, where $\bar{X}$ is the sample mean.

7.3 The realization of the random walk in Example 7.4 was generated by using a `while` loop in R. Write R code to generate the same time series values without using a loop.

7.4 The time series of the number of monthly accidental deaths in the United States from 1973 to 1978 is given in `USAccDeaths` in R. Make a plot of the time series values and comment on any features you can glean from the plot.

7.5 Classify each of the following stochastic processes by time (discrete or continuous) and state (discrete or continuous).

(a) The number of eastbound cars stopped at a particular stoplight over time.

(b) The location of a taxi cab (classified as city, airport, or suburbs) at the end of each passenger's ride.

(c) The temperature of a puppy measured at 20-minute intervals.

(d) A person's internal body temperature over time.

(e) The number of goldfish on inventory at a local pet shop.

7.6 Consider the random variables $X_1, X_2, \ldots, X_n$ and $Y_1, Y_2, \ldots, Y_m$ with associated finite population means, population variances, and population covariances. Show that

$$\text{Cov}\left(\sum_{i=1}^{n} a_i X_i, \sum_{j=1}^{m} b_j Y_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{m} a_i b_j \text{Cov}\,(X_i, Y_j)$$

for real-valued constants $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$.

7.7 The time series $\{X_t\}$ consists of the number of spots on the up face of sequential rolls of a fair die.

(a) Is this time series strictly stationary?

(b) What is the population mean function?

(c) What is the population autocovariance function?

(d) What is the population autocorrelation function?

7.8 Consider the (tiny) time series X_1, X_2, X_3, whose values are the cumulative number of spots showing in three rolls of a fair die. More specifically, if R_1, R_2, and R_3 denote the outcomes of the three rolls, then $X_1 = R_1$, $X_2 = R_1 + R_2$, and $X_3 = R_1 + R_2 + R_3$.

(a) What is the population mean function?

(b) Is this time series strictly stationary?

(c) Is this time series stationary?

(d) What is the population variance–covariance matrix of X_1, X_2, X_3?

(e) Perform a Monte Carlo simulation which provides convincing numerical evidence that the population variance–covariance matrix from part (d) is correct.

7.9 Argue whether the time series of monthly home energy consumption observations from Example 7.1 is a stationary time series.

7.10 Consider the time series $\{X_t\}$ consisting of white noise terms defined by

$$X_t \sim \begin{cases} N(0,1) & t \text{ odd} \\ U\left(-\sqrt{3}, \sqrt{3}\right) & t \text{ even.} \end{cases}$$

(a) Is this time series strictly stationary?

(b) Is this time series stationary?

7.11 The energy consumption time series introduced in Example 7.1 was given in number of kilowatt hours per month. The varying number of days per month was not taken into account in the analysis performed on this data set in this chapter. Adjust the time series so that the varying month length has been taken into account and answer the following.

(a) The original time series had the maximum monthly energy consumption in January of 2018. Which month has the maximum monthly energy consumption in the adjusted time series?

(b) Make a plot of the time series using the units average daily number of kilowatt hours.

(c) Use the R `acf` function to calculate the sample autocorrelation at lag 3. Interpret the sign of the sample lag 3 autocorrelation.

7.12 The R built-in data set `nhtemp` contains the average annual temperatures in New Hampshire from 1912 to 1971. Plot the time series and correlogram.

7.13 Consider a time series of $n = 100$ observations which are Gaussian white noise with $\sigma_Z = 1$. Use Monte Carlo simulation to estimate the probability that the lag 3 sample autocorrelation falls between $-z_{\alpha/2}/\sqrt{100}$ and $z_{\alpha/2}/\sqrt{100}$, where $\alpha = 0.05$. Report your estimate to 3-digit accuracy.

7.14 The sojourn time of a customer in a single-server queue is defined as the waiting time plus the service time. Consider 100 consecutive sojourn times for customers in a single-server queue with exponential times between arrivals with population mean 1 and exponential service times with population mean 0.9. (This is a special case of what is known in queueing theory as an M/M/1 queue. The first M is for Markov, and indicates that the times between arrivals is exponentially distributed. The second M is also for Markov, and indicates that the service times are exponentially distributed. The 1 indicates that there is a single server.) Also assume that the first 1000 customer sojourn times have been discarded so that the system has "warmed up." A realization of these 100 consecutive sojourn times can be generated in R and placed into the vector `x` with the statements

```
install.packages("simEd")
library(simEd)
x = ssq(maxArrivals = 1100, saveSojournTimes = TRUE,
        showProgress = FALSE, seed = 12345)$sojournTimes[1001:1100]
```

(a) Write a paragraph that outlines whether or not the stationarity assumption is appropriate in this setting.

(b) Before running a simulation, predict whether the sample lag 3 autocorrelation will be positive, zero, or negative.

(c) Make three runs of this simulation with three different seeds and plot the three sample autocorrelation functions on the same set of axes (plot them as points connected by lines rather than spikes) for the first 20 lags.

7.15 Consider a stationary time series $\{X_t\}$. For $k = 1, 2, \ldots$, the lag k population partial autocorrelation $\rho^*(k)$ can be written as the last component of the vector defined by

$$\Gamma_k^{-1}\gamma_k,$$

where Γ_k is the $k \times k$ variance–covariance matrix of any k elements of the time series and $\gamma_k = \big(\gamma(1), \gamma(2), \ldots, \gamma(k)\big)'$. Show that this way of calculating the lag k population partial autocorrelation is equivalent to that given in Definition 7.8 for $k = 1$ and $k = 2$.

7.16 Consider a three-point moving average.

(a) What are the coefficients associated with three of these moving averages applied in series.

(b) Check your solution using the R `convolve` function.

(c) Apply this series of three filters to the `AirPassengers` data set and plot the smoothed series.

7.17 The logarithm and square root transformations are commonly used on a time series whose variability increases with time. Propose a transformation for a time series whose variability decreases with time.

7.18 Find the population autocorrelation function for an m-point moving average of a white noise time series, where m is an odd, positive integer.

7.19 The R `decompose` function can be used to decompose a time series. Another R function named `stl` is a more sophisticated function for decomposing a time series. Apply this function to the built-in R time series `JohnsonJohnson`, which contains the quarterly earnings, in dollars, for one share of stock in Johnson & Johnson from 1960 to 1980. Plot the trend and seasonal components of the decomposed time series. Which quarter tends to have the highest quarterly earnings? Considering the seasonal part of the decomposed model, what is the difference between the best and worst quarter's earnings to the nearest penny?