The Frequency Domain

4

In Chapter 1 we defined cyclic behavior to mean that a time series realization rises and falls in somewhat of a repetitive fashion. We refer to such data as cyclic or pseudo-periodic. An example of cyclic data is the sunspot data (Figure 1.1) which have an intriguing cyclic behavior of about 11 years. We noted that sunspot cycle lengths varied randomly from cycle to cycle. The DFW monthly temperature data (Figure 1.3) are cyclic data with a fixed 12-month cycle length that can be explained by the earth's rotation around the sun. Such fixed-period cyclic data for which the time axis is tied to the calendar year are referred to as seasonal data. The Air Passengers data (Figure 1.5) are another example of seasonal data. In Chapter 1 we also discussed data for which there was no obvious cyclic behavior. The monthly Dow Jones index (Figure 1.8(d)), West Texas Intermediate crude oil prices (Figure 1.8(e)), and DFW annual temperature data (Figure 1.20(b)) are examples of data with trending or wandering behavior but for which there is no detectable cyclic behavior. The Air Passengers data have both seasonal and trending patterns.

The statistical analysis of time series was discussed in Chapter 3 and the concept of stationarity was introduced. We showed that for stationary time series, it "makes sense" to estimate the mean, variance, and autocorrelations using one realization. Plots of time series realizations and sample autocorrelations provide us with information about the time series behavior. Time series realizations are indexed by "time" while sample autocorrelations are indexed by "lag", which is the difference between time points. Both of these indices have *time* as their basis, and it is natural to think of time series in the *time domain*.

In this chapter, we examine time series from the perspective of their cyclic (or frequency) content. This type of study is said to be based on the *frequency domain*. Frequency domain analysis is based on the fact, shown originally by Joseph Fourier in 1807, that a broad class of functions, $f(x)$, can be written as a linear combination of sines and cosines. See video with the QR code below for an incredible demonstration of the power of these representations.

QR 4.1 Fourier Representation

The spectral density, discussed in Section 4.2, is the main tool we will use for analyzing the frequency content in data. This type of frequency domain analysis is commonplace in science and engineering and is the topic of this chapter. It is our experience that this will likely be the first introduction to the frequency domain analysis for statistics/data analytics students taking their first time series course. We use the frequency domain as a key part of our analyses of time series data throughout the remainder of this book.

DOI: 10.1201/9781003089070-4

4.1 TRIGONOMETRIC REVIEW AND TERMINOLOGY

We are assuming some familiarity with trigonometric functions. However, we first go over some introductory material, much of which will be a review. Figures 4.1(a) and (b) show the familiar sine and cosine waves. We have talked about cyclic /pseudo-periodic time series data. We will use the following terminology:

Periodic Function: A function, $g(t)$, is said to be periodic with period (or cycle length) $p > 0$ if p is the smallest value such that $g(t) = g(t + kp)$ for all integers k. Clearly, functions $\sin(t)$ and $\cos(t)$ are periodic with period (cycle length) 2π. Figures 4.1(a) and (b) show two complete cycles (periods) with the two cycles separated by the vertical lines at 2π. Although Fourier analysis is based on sines and cosines, other periodic functions exist, such as the function plotted in Figure 4.1(c), which is a periodic function of period 10.[1]

Key Point: The terms "cycle length" and "period" will be used interchangeably.

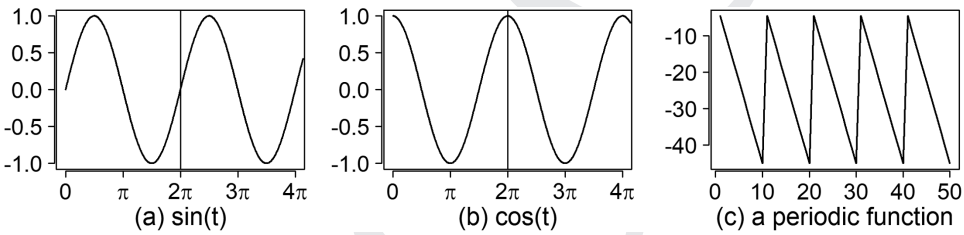


FIGURE 4.1 (a) sin(t) and (b) cos(t), and (c) another periodic function.

Phase Shift: The function sin(t+ℓ) is said to be a sine wave with phase shift ℓ. Such a function still has a cycle length of 2π, but it is "shifted" along the horizontal axis. Figure 4.2(a) overlays the functions $\sin(t)$ and $\sin(t+1)$. At time $t = 0$, $\sin(t+1)$ takes on the value of $\sin(t)$ at $t = 1$. Figure 4.2(b) overlays $\sin(t)$ and $\sin(t+\pi)$. Visually, the two curves are mirror images of each other across a horizontal line at zero.

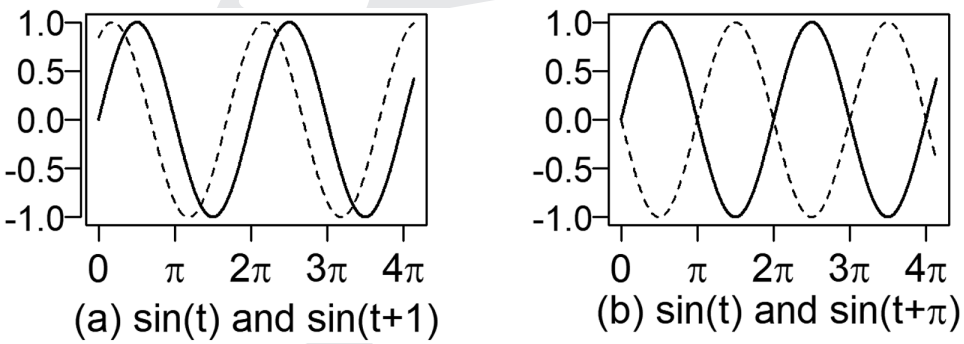


FIGURE 4.2 (a) Overlay showing sin(t) from Figure 4.1(a) (solid) and the phase-shifted sin (t+1) (dashed) lines; (b) Overlay showing sin(t) from Figure 4.1(a) (solid) and the phase-shifted sin (t+π) (dashed) lines.

1 Actual time series data will almost never be "truly periodic". See datasets in the first three chapters. We give this definition as the fundamental behavior which cyclic data approximates.

Frequency (denoted by f)*:* Related to period, another descriptive measure of the behavior of a periodic function is the *frequency*, which can be described in two ways. Frequency is

(a) 1/period (cycle length)
(b) the number of cycles the function goes through in a unit of time

Figure 4.1(a) and Figure 4.3(a) show that $\sin(t)$ has a period of 2π and hence a frequency of $1/(2\pi)$. Figure 4.3(b) indicates that $\sin(2\pi t)$ has period and frequency of one. In Figures 4.3(c) and 4.3(d), we see that $\sin(2\pi 2t)$ has period $1/2$ and frequency two and that $\sin(2\pi(.5)t)$ has period two and frequency $1/2$. Table 4.1 summarizes the information obtained from Figure 4.3.

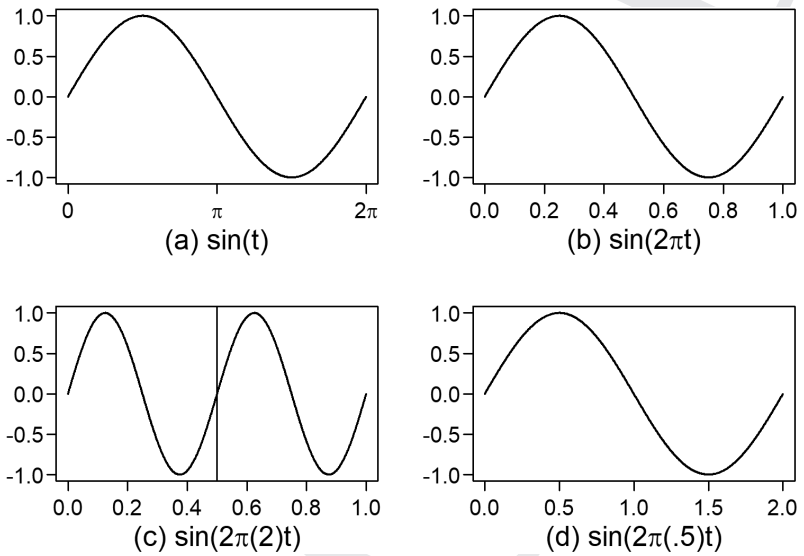


FIGURE 4.3 (a) $\sin(t)$, (b) $\sin(2\pi t)$, (c) $\sin(2\pi 2t)$ and (d) $\sin(2\pi(.5)t)$.

TABLE 4.1 Periods and Frequencies for Selected Sinusoidal Functions

	PERIOD	*FREQUENCY* (f)
(a) $\sin(t)$	2π	$1/(2\pi)$
(b) $\sin(2\pi t)$	1	1
(c) $\sin(2\pi(2)t)$	.5	2
(d) $\sin(2\pi(.5)t)$	2	.5
In General		
$\sin(2\pi ft)$	$1/f$	f

Key Points

1. The function $\sin(2\pi ft)$ has frequency f. That is, it goes through f cycles per unit.
2. If the data are collected annually, monthly, or hourly then the corresponding frequencies measure the number of cycles per year, month, or hour, respectively.
3. In scientific and engineering literature, it is common to measure frequency in Hertz (Hz), which is the number of cycles per second.

Cyclic/Pseudo-periodic Data: As mentioned in Chapter 1 and in the introduction to this chapter, time series data that are likely to be encountered will be "sort of" periodic which we refer to as cyclic or pseudo-periodic data. The sunspot and log-lynx datasets are shown in Figure 4.4(a) and (b), respectively. These datasets have previously been discussed and we have noted that they both have fairly repeatable cycles (but do not satisfy the exacting conditions of a periodic function). The average cycle length for the sunspot data is 11 years so that the sunspot data have a frequency of about $1/11 = .09$. The log-lynx data have a similar cycle length and frequency. Figure 4.4(c) is a plot of a simulated realization that goes through about seven cycles in the realization of length $n = 100$. Consequently, the cycle length is about $100/7 = 14.3$ and the associated frequency is about $f = 7/100 = .07$. The following ***tswge*** code plots the data below:

```
data(sunspot2.0)
data(lynx)
llynx=log10(lynx)
x3=gen.arma.wge(n=100,phi=c(1.75,-.95),sn=203)
par(mfrow=c(1,3),mar=c(4,3,.2,.5))
plotts.wge(sunspot2.0)
plotts.wge(llynx)
plotts.wge(x3,type='l')
```

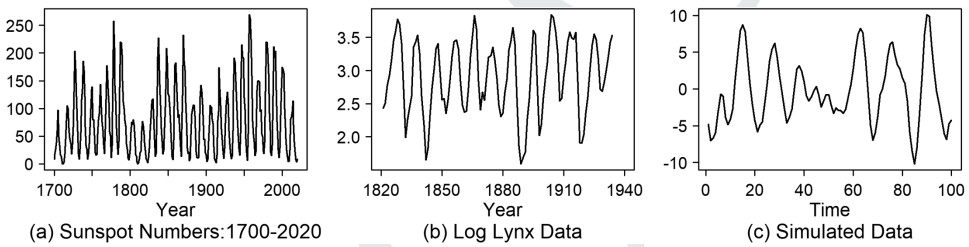


FIGURE 4.4 (a) Sunspot data (1800–present), (b) Log Lynx data, and (c) Simulated data.

Non-cyclic (aperiodic) Data: Consider the three datasets plotted in Figure 4.5. As we will see, these figures share the property that there is no discernible cyclic (or even pseudo-cyclic) behavior. Monthly West Texas Intermediate crude oil prices from January 1990 through December 2020 are shown in Figure 4.5(a). As mentioned in Sections 1.2.2.1 and 1.2.2.2, the price of oil has moved around erratically during this time period with no discernible pattern. Figure 4.5(b) shows the Dow Jones monthly closing averages from March 1985 through December 2020. While the DOW seemed to generally trend upward during that period, there were "dips" and pattern changes "along the way". See Sections 1.2.2.1 and 1.2.2.2. Specifically, there is no discernible pseudo-cyclic behavior. Finally, Figure 4.5(c) shows a simulated realization which has a behavior that in places looks pseudo-cyclic (eg. from $t = 75$ through $t = 150$). However, the behavior before or after this time window does not carry on any such pattern.[2]

Key Point: We will often refer to the non-cyclic behavior seen in Figure 4.5 as *aimless wandering*.

2 We use the term "aperiodic" loosely. Non-cyclic behavior is correctly described as aperiodic. Technically, $g(t)$ is an aperiodic function if there does not exist a period $p > 0$ such that $g(t) = g(t + kp)$ for all integers k. Therefore, based on this precise definition, the pseudo-periodic functions in Figure 4.4 are aperiodic. However, we will use the term aperiodic to indicate functions such as those in Figure 4.5 that show no cyclic tendencies. We will discuss this further in Section 4.2.

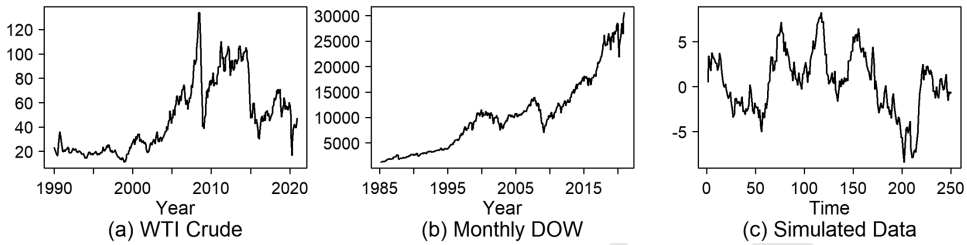


FIGURE 4.5 (a) Monthly West Texas Intermediate Crude price from January 1990 through December 2020, (b) Dow Jones monthly closing averages from March 1985 through December 2020, and (c) simulated data.

4.2 THE SPECTRAL DENSITY

Figure 4.6(a) has the general appearance of time series realizations we will encounter in this text. This signal is the sum of sine waves with different cycle lengths and phase shifts. Close examination of Figure 4.6(a) shows a period of about length 25 (notice that the data seem to go through about four cycles in the realization of length $n = 100$). Shorter term periodic behavior is also visible in the dataset. The question remains, "What frequencies (or cycle lengths) are present in the data?"

Inside Information: Figure 4.6(a) is the sum

$$f(x) = 4\sin\left(2\pi(.04)t\right) + 1.25\sin\left(2\pi(.125)t + 1\right) + .5\sin\left(2\pi(.25)t + 2.5\right). \tag{4.1}$$

That is, $f(x)$ is the sum of the three components. Note that multiplier weights on the three components are 4, 1.25, and .5, respectively. The components are described in Table 4.2.

TABLE 4.2 Characteristics of Components that Sum to Give Figure 4.6(a)

COMPONENT	*FREQUENCY*	*PERIOD*	*PHASE SHIFT*	*MULTIPLIER*
1	.040	25	0.0	4.00
2	.125	8	1.0	1.25
3	.250	4	2.5	.50

What is needed is a tool to identify the frequency content of a stationary time series. The good news is that such a tool exists. The *spectrum* (or in standardized form, the *spectral density*) is the main tool in our tool kit for identifying underlying frequency behavior in stationary time series data.

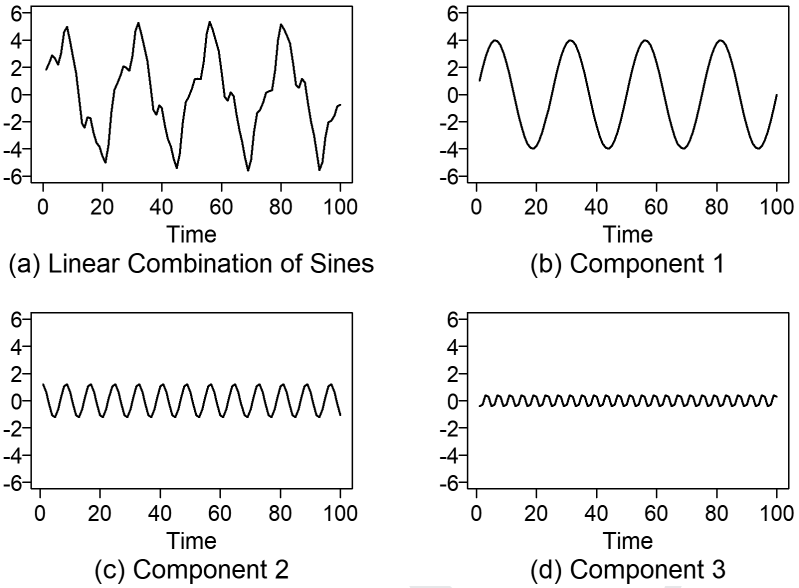


FIGURE 4.6 (a) Sum of the sine waves in (b), (c), and (d).

The graphs in Figure 4.6 can be obtained using the following code:

```
t=1:100
y=rep(0,100)
y1=rep(0,100)
y2=rep(0,100)
y3=rep(0,100)
for(i in 1:100){y1[i]= 4*sin(2*pi*.04*i)}
for(i in 1:100){y2[i]=1.25*sin(2*pi*.125*i+1)}
for(i in 1:100){y3[i]=.5*sin(2*pi*.25*i+2.5)}
y=y1+y2+y3
par(mfrow=c(2,2),mar=c(4,2.5,1.5,.5))
plot(y,type='l',xlab='(a) Linear Combination of Sines')
plot(y1,type='l',xlab='(b) Component 1')
plot(y2,type='l',xlab='(c) Component 2')
plot(y3,type='l',xlab='(d) Component 3')
```

4.2.1 Euler's Formula

The spectrum and spectral density of a time series are defined as functions of frequency. Some trigonometric and complex variables formulas of importance are the following stated here for completeness:

(a) $\sin(-\theta) = -\sin(\theta)$

(b) $\cos(\theta) = \cos(-\theta)$

(c) $e^{i\theta} = \cos(\theta) + i\sin(\theta)$ (Euler's Formula)

(d) $e^{-i\theta} = \cos(\theta) - i\sin(\theta)$

4.2.2 Definition and Properties of the Spectrum and Spectral Density

We are now in position to define the spectrum and spectral density.

Definition 4.1: Let X_t be a stationary time series with autocovariance γ_k and autocorrelation ρ_k. Then for $|f| \leq .5$:

(1) The *spectrum* of X_t is defined by

$$P_X(f) = \sum_{k=-\infty}^{\infty} e^{-2\pi i f k} \gamma_k. \tag{4.2}$$

(2) The *spectral density* of X_t is defined by

$$S_X(f) = \sum_{k=-\infty}^{\infty} e^{-2\pi i f k} \rho_k. \tag{4.3}$$

Using Euler's formula, we obtain the "more pleasing" formulas:

$$P_X(f) = \sigma_X^2 + 2\sum_{k=1}^{\infty} \gamma_k \cos(2\pi f k), \tag{4.4}$$

and

$$S_X(f) = 1 + 2\sum_{k=1}^{\infty} \rho_k \cos(2\pi f k). \tag{4.5}$$

These formulas emphasize that the spectrum and spectral density are real-valued functions which is not apparent from (4.2) and (4.3). The condition that $|f| \leq .5$ will be addressed in Section 4.2.2.1 when we discuss the Nyquist frequency.

Key Point: In this text, we will focus on the spectral density.

Important Properties of Spectral Densities:

1. $S_X(f) \geq 0$
2. $S_X(f) = S_X(-f)$
3. $S_X(f) = 1 + 2\sum_{k=1}^{\infty} \rho_k \cos(2\pi f k)$, where $|f| \leq .5$ (Equation 4.5 above)
4. $\sum_{-.5}^{.5} S_X(f) e^{2\pi i f k} df = \rho_k$

Properties 1 and 2 show that $S_X(f)$ is a non-negative, even function.

Key Point: We plot spectral plots only for $f \geq 0$ because $S_X(f)$ is an even function and the plot for negative values would simply be a mirror image.

Property 3 above says that if the autocorrelation function, ρ_k, $k = 0, 1, 2, \ldots$ is known, then the spectral density can be calculated. On the other hand, Property 4 says that if we know the spectral density, $S_X(f)$, then we can calculate the autocorrelation function.[3]

Key Points

1. Properties 3 and 4 show that ρ_k and $S_X(f)$ contain equivalent mathematical information − if you know one, you can calculate the other.
2. ρ_k and $S_X(f)$ are called *Fourier transform pairs*.

4.2.2.1 The Nyquist Frequency

Definition 4.1 has an interesting restriction on the range of f. Specifically, the definitions for the spectrum and spectral density are subject to the constraint that $|f| \leq .5$. For time series data sampled at unit increments, the shortest period that can be observed is a period of 2. (Think about it.) That is, the highest observable frequency is $f = 0.5$, which is called the *Nyquist* frequency. To better understand the situation, consider the two signals $X_t = \sin(2\pi(.2)t)$ and $Y_t = \sin(2\pi(1.2)t)$, which have frequencies 0.2 and 1.2, respectively. These two signals are defined along a continuum, but we consider the case in which they are sampled at the integers. Figure 4.7 shows an overlay of these two signals, and it is interesting to note that the two signals are equal at integer values of t (shown as dots on the graph). For example, letting $t = 2$,

$$\sin(2\pi(.2)2) = \sin(.8\pi)$$
$$\sin(2\pi(1.2)2) = \sin(2\pi(1+.2)2) = \sin(4\pi+.8\pi) = \sin(.8\pi)$$

because $\sin(\theta + k\pi) = \sin(\theta)$ if k is an even integer. Consequently, the two curves are indistinguishable when sampled at the integers. Consider again the curve $\sin(2\pi(1.2)t)$ (bold in Figure 4.7). If we sample at the integers, we have no way of knowing that this curve passed through more than a full cycle between, for example, $t = 1$ and $t = 2$. This phenomenon is called *aliasing*, because the frequency $f = 1.2$ "falsely appears" to be the same as $f = 0.2$. For data collected at the integers, it would be impossible to detect a period less than two sample units. Consequently, we must restrict frequencies to $|f| \leq .5$. This leads to the following definition.

Definition 4.2: Nyquist Frequency
For data $x_t, t = 1, 2, \ldots, n$, then the Nyquist frequency is $1/2$, and the shortest observable period is 2.[4]

3 In this book, you will not be expected to calculate the integral-based formula in Property 4 or compute the infinite sum in Property 3.

4 If data are sampled at the increments of Δ, then the Nyquist frequency is $1/(2\Delta)$, and the shortest observable period is 2Δ.

Key Point: In order to recover frequencies higher than $1/(2\Delta)$, it is necessary to sample more rapidly.

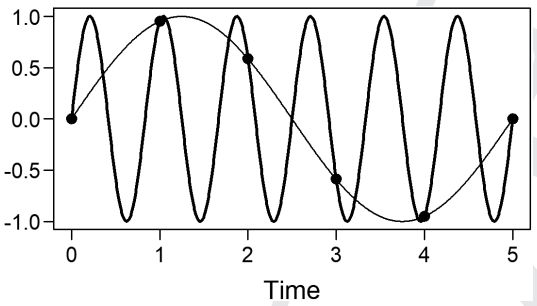


FIGURE 4.7 Plots of $\sin(2\pi(.2)t)$ (thin line) and $\sin(2\pi(1.2)t)$ (bold line).

Figure 4.7 can be obtained using the following code:

```
t=1:1000
tm=t/200
y1=sin(2*pi*.2*tm)
z1=sin(2*pi*1.2*tm)
tp=c(0,1,2,3,4,5)
yp=sin(2*pi*.2*tp)
par(mfrow=c(1,1),mar=c(2.5,2.5,1,1.5))
plot(tm,y1,type='l',xlab='Time',xlim=c(0,5))
points(tm,z1,type='l',lwd=2)
points(tp,yp,pch=19)
```

4.2.2.2 Frequency f = 0

Notice that the restriction $|f| \leq .5$ includes the frequency $f = 0$. Before proceeding, we need to discuss the meaning of $f = 0$. Consider the following list of frequencies and associated cycle lengths.

FREQUENCY	*CYCLE LENGTH*
.1	1/.1=10
.01	100
.001	1,000
.0001	10,000

We see that the closer the frequency gets to $f = 0$, the longer the cycle length. Consequently, $f = 0$ is associated with an "infinite" cycle length. That means there is no periodic content in the data "at all". For example, the realization in Figure 4.8(a) has no periodic content (exhibits "$f = 0$" behavior) and can be characterized as having "aimless wandering" behavior.

If a realization of length $n = 200$ has only aimless wandering, all we can really say is that, if there is a period, it is longer than the realization length, $n = 200$. Figure 4.8(b) shows the realization in Figure 4.8(a) extended to 800 time points. The realization of length $n = 200$ is shown in the shaded region. For this longer realization we see that there is a period of length 250 (marked off by vertical lines on the plot). However,

because in Figure 4.8(a) we only observed the first 200 points, we were unable to detect this period. Note that the frequency associated with a period of 250 is $f = 1/250 = .004$ which is quite close to $f = 0$.

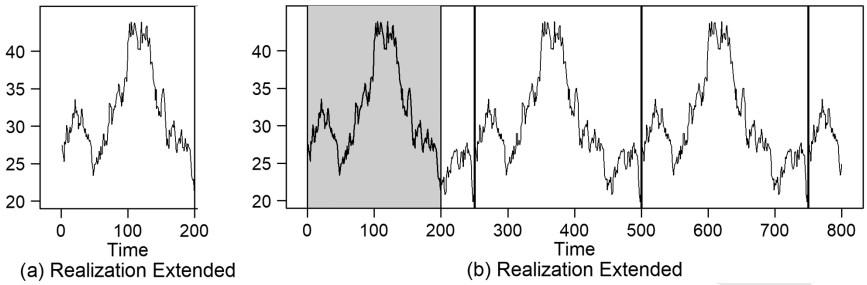


FIGURE 4.8 (a) Realization showing aimless wandering behavior and (b) Realization extended to 800 points showing periodic behavior.

Key Points

1. Data associated with frequency $f = 0$ never have a repeating behavior and is characterized by "aimless wandering" behavior.
2. Given a realization of length n that shows only "aimless wandering", all we can confidently say is that if there is a repeating period in the data, the period length is longer than n.

Example 4.1 Spectral Density for Data with Two Dominant Frequencies

We have claimed that the spectral density is designed to identify the underlying frequency content of a time series realization. In Figure 4.9(a) we show a realization from an *autoregressive* model (these models will be discussed in Chapter 5). Examination of the realization shows that there is a dominant cyclic behavior and that the data go through about 10 cycles in the realization of length $n = 200$. That is, the period length is about 20 so that the associated frequency is about $f = 1/20 = .05$. Further examination of the realization shows that there is a "jagged wiggle" in the data. This behavior is associated with a very short period that corresponds to high frequency (nearly oscillating up and down) behavior. Recall that $f = 1/(\text{period})$, so that the shorter the period length, the greater (higher) the frequency. Figure 4.9(c) shows the "true" spectral density. For the autoregressive model from which the realization is generated, the spectral density in (4.5) has a simple closed-form solution. This solution, which will be given in 5.30, was used to calculate the spectral density shown in Figure 4.9(c).[5]

Recall that the spectral density is designed to identify frequency content in a time series realization. In Figure 4.9(c) we see that the spectral density has a peak at about $f = .05$ and another peak at about $f = .45$. The peak at $f = .05$ is associated with the previously mentioned period lengths of about 20. The peak at $f = .45$ is associated with cycle lengths of approximately $1/.45 = 2.2$. These short cycles are associated with the "high frequency" wiggle in the data. The true autocorrelations in Figure 4.9(b) show a dominant slowly damping sinusoidal behavior with period of about 20. There is very little indication of the high frequency behavior, but close observation shows that ρ_1 appears to be slightly smaller than ρ_2, which is not consistent with the smooth sinusoidal behavior associated with the longer cycle.

5 Spectral densities are typically plotted in log scale. We will discuss the reason for this in Section 4.2.3.4. Spectral densities could be calculated using natural logs or log base 10. However, in this book, we will follow the practice used widely in the scientific community of plotting spectral densities in decibels (dB), which are the standard units for measuring acoustical energy and are calculated as $10 \log_{10} S_X(f)$.

Key Points

1. The spectral density is designed to identify frequency content in stationary time series data.
2. Frequency content in a realization is represented by "peaks" in the spectral density.

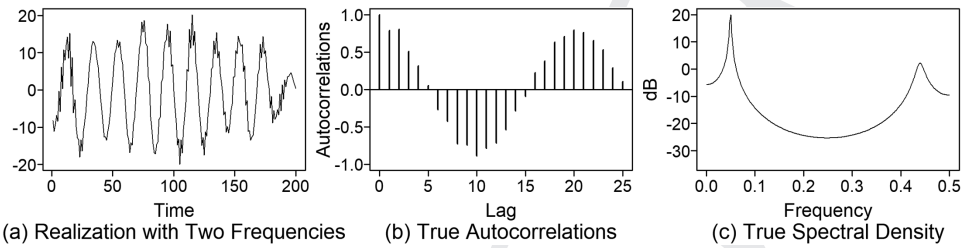


FIGURE 4.9 (a) Realization with two dominant frequencies, (b) true autocorrelations, and (c) the true spectral density for the model from which realization (a) was generated.

The graphs in Figure 4.9 are obtained using the single ***tswge*** command

```
x=plotts.true.wge(n=200,phi=c(0.1300,1.4414,-0.0326,-0.8865),sn=9310)
```

Example 4.2 Spectral Density of White Noise

An example in which the formula for the spectral density in (4.5) can be easily calculated is the case of white noise data. Recall that for white noise, $\rho_0 = 1$ and $\rho_k = 0$ for $k \neq 0$. In this case, $S_X(f) = 1 + 2\sum_{k=1}^{\infty} \rho_k cos(2\pi fk)$ simplifies to $S_X(f) = 1$ for $|f| \geq .5$. The fact that the spectral density is flat indicates that all frequencies are equally present in white noise. Specifically, there are no peaks in the spectral density. Figure 4.10(a) shows the plot for $f \in [-.5,.5]$ while Figure 4.10(b) shows the plot for only $f \in [0,.5]$ as will be our practice in this book.

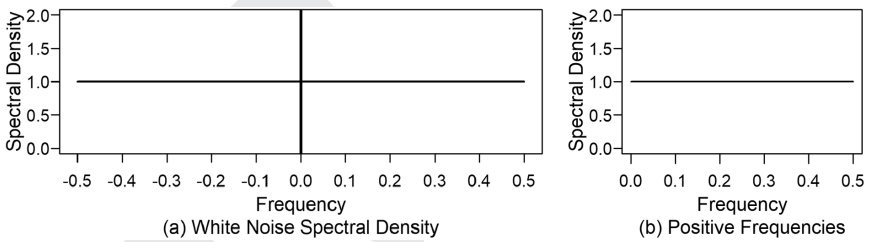


FIGURE 4.10 (a) White noise spectral density and (b) white noise spectral density restricted to positive frequencies.

QR 4.2 Spectral Density for White Noise

4.2.2.3 The Spectral Density and the Autocorrelation Function

Recall that the spectral density, $S_X(f)$, and the autocorrelation function, $\rho_k, k = 0,1,2,\ldots$ contain equivalent mathematical information. That is, if you know one of the functions, you can mathematically compute

the other. Despite their "equivalence", we illustrated in Example 4.1 the fact that the spectral density is a better tool for identifying underlying cyclic behavior.

Key Point: The spectral density is far superior to the autocorrelations as a tool for identifying underlying cyclic behavior (even though they contain essentially equivalent mathematical information).

4.2.3 Estimating the Spectral Density

In practice, given a realization of length n from a time series, we do not have sufficient information to calculate the "true" spectral density in (4.5) which involves an infinite sum of autocorrelations. For one thing, given an actual set of time series data, we do not know the "true" autocorrelations. That is, although (4.5) involves an infinite sum of autocorrelations, ρ_k, we only "know" the estimates, $\hat{\rho}_k$, defined in (3.12), and from (3.10) it follows that for a realization of length n, only $\hat{\rho}_k, k = 1,\ldots,n-1$ can be calculated.

Key Point: In practice, given a realization of time series data, we do not know the true autocorrelations necessary to compute $S_X(f)$ in (4.5). However,

- we can estimate ρ_k using the sample autocorrelations, $\hat{\rho}_k$, defined in (3.12).
- we can only calculate $\hat{\rho}_k, k = 1,\ldots,n-1$.

QR 4.3 Sample Spectral Density

4.2.3.1 The Sample Spectral Density

Based on the above Key Point, the "natural estimate" of the spectral density is obtained by substituting the sample autocorrelations into the formula for $S_X(f)$. Using this strategy, we obtain

$$\hat{S}_X(f) = 1 + 2\sum_{k=1}^{n-1}\hat{\rho}_k \cos(2\pi f k), \; |f| \le .5. \tag{4.6}$$

The estimate in (4.6) is called the *sample spectral density*. Figure 4.11(a) shows the realization in Figure 4.9(a) with a vertical line at $t = 100$. Figure 4.11(b) shows the sample spectral density calculated using (4.6) based on the first 100 time points in Figure 4.11(a) (that is, those that occur before the vertical line at $n = 100$). The two peaks are visible, but the behavior of the sample spectral density is very erratic (variable). While the true spectral density is a smooth curve, the sample spectral density is not at

all smooth. We next calculate the sample spectral density based on the full realization of length $n = 200$ in Figure 4.11(a). The resulting sample spectral density for the longer realization length is plotted in Figure 4.11(c). There is no apparent improvement over the plot in Figure 4.11(b). Most estimators, like the sample mean and sample variance, tend toward the true values as the sample size increases. The behavior of the sample spectral density is unusual and disappointing.

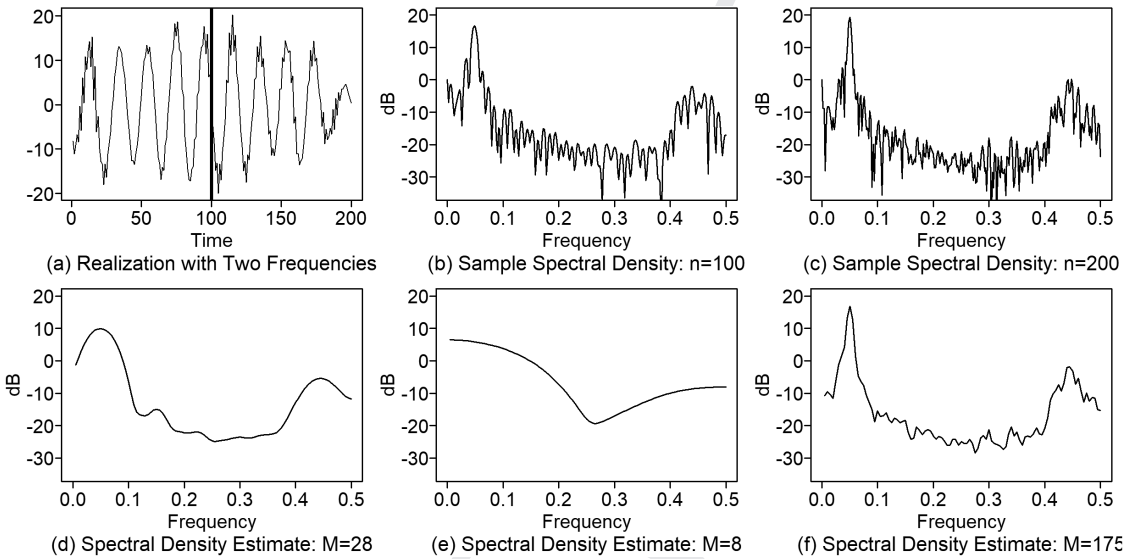


FIGURE 4.11 (a) Realization in Figure 4.9(a), (b) sample spectral density based on the first 100 points (to the left of the vertical line) in (a), (c) sample spectral density for the entire realization in (a); (d), (e), and (f): Parzen spectral density estimates for $M = 28$ (default value), 8 and 175, respectively, for the 200 point realization in (a).

The plots in Figure 4.11 can be obtained using the following R code.

```
x=gen.arma.wge(n=200,phi=c(0.1300,1.4414,-0.0326,-0.8865),sn=9310)
# this generates an AR(4) realization (details in Chapter 5)
par(mfrow=c(2,3))
plotts.wge(x)
abline(v=100)
mtext(side=1,line=1.4)
sample.spec.wge(x[1:100])
sample.spec.wge(x)
parzen.wge(x)
parzen.wge(x,trunc=8)
parzen.wge(x,trunc=175)
```

For a realization, **x**, of length *n*, the sample spectral density can be obtained using the ***tswge*** command **sample.spec.wge(x).**

Key Points

1. The sample spectral density does not "improve" as the realization length increases.
   − that is, it does not converge to the true spectral density as *n* increases.
2. Another spectral estimate related to the sample spectral density is the periodogram. See Woodward, et al. (2017) for a discussion of the periodogram.

QR 4.4
Periodogram

4.2.3.2 Smoothing the Sample Spectral Density

To understand the problem associated with the non-convergence of the sample spectral density, we recall that the quality of $\hat{\rho}_k$ decreases as k increases. Note that from (3.12), given a realization of length n, the sample autocorrelation at lag k is given by

$$\hat{\rho}_k = \frac{\frac{1}{n}\sum_{t=1}^{n-k}(x_t - \bar{x})(x_{t+k} - \bar{x})}{\hat{\sigma}_X^2}$$

As discussed in Section 3.3.2.3, using this general formula for $\hat{\rho}_k$, it follows that

$$\hat{\rho}_1 = \frac{\frac{1}{n}\sum_{t=1}^{n-1}(x_t - \bar{x})(x_{t+1} - \bar{x})}{\hat{\sigma}_X^2},$$

because there are $n-1$ data pairs in the realization separated by one time period. The $n-1$ pairs are x_1 and x_2, x_2 and x_3,…, x_{n-1} and x_n. Thus, $\hat{\rho}_1$ is "a sort of average" of $n-1$ cross products $(x_t - \bar{x})(x_{t-1} - \bar{x})$. On the other hand,

$$\hat{\rho}_{n-1} = \frac{\frac{1}{n}(x_n - \bar{x})(x_1 - \bar{x})}{\hat{\sigma}_X^2},$$

because x_1 and x_n are the only pair of data values separated by $n-1$ time points. Consequently, we do not expect $\hat{\rho}_{n-1}$ to be a very good estimator of ρ_{n-1}. This is analogous to estimating the population mean, μ, of a population based on a sample of size $n = 1$. And notice that the problem with estimating ρ_{n-1} based on one cross-product occurs whether $n = 100$ or $n = 500{,}000$.

This issue with the inability to obtain reliable estimates of ρ_{n-1} (and any ρ_k where k is close to n) gives us a clue regarding how to improve the sample spectral density. Through the years, several smoothing methods have been developed that have two characteristics:

(1) The sum in (4.6) is truncated at some value $M < n-1$.
(2) A "smoothing" or "window" function, λ_k, is used to minimize the impact of $\hat{\rho}_k$ as k increases. That is, $\lambda_k \to 0$ as $k \to \infty$.

The general formula for a "smoothed" spectral density estimator is

$$\hat{S}_X(f) = \lambda_0 + 2\sum_{k=1}^{M}\lambda_k\hat{\rho}_k\cos(2\pi fk), \; |f| \le .5. \tag{4.7}$$

Implementation involves a choice for M and a window function. A common choice for M is $2\sqrt{n}$. Although there are many window functions, we have chosen to implement the Parzen window in ***tswge***. The Parzen window is defined as

$$\begin{aligned}\lambda_k &= 1-6\left(\frac{k}{M}\right)^2+6\left(\frac{k}{M}\right)^3, \quad 0\le k\le M/2\\ &= 2\left[1-\left(\frac{k}{M}\right)\right]^3, \; M/2<k\le M\\ &= 0,\; k>M,^{6}\end{aligned} \tag{4.8}$$

and is implemented in ***tswge*** functions **`parzen.wge`** and **`plotts.sample.wge`**. Using **`parzen.wge(x)`**, the Parzen spectral density estimate is obtained using the default value $M = 2\sqrt{n}$. You can also specify a truncation value, say $M = 8$, by using the command **`parzen.wge(x,trunc=8)`**.

Example 4.1 (revisited) Figure 4.11(d) shows the Parzen spectral density estimate for the data set considered in Example 4.1 (shown originally in Figure 4.9(a)) using the default value $M = 2\sqrt{200} = 28$. In this plot the two peaks at $f = .05$ and $f = .45$ are visible, although not as sharp as in the true spectral density in Figure 4.9(c). Recall that Figure 4.11(c) is the sample spectral density for the realization of length $n = 200$ shown in Figure 4.9(a). The Parzen window has "smoothed" the sample spectral density dramatically. Figures 4.11(e) and (f) show Parzen spectral density estimates for $M = 8$ and $M = 175$, respectively. The spectral estimate for $M = 8$ uses only the first eight sample autocorrelations, and consequently loses much of the detail in the spectral density. In this case, there is a peak in the spectral density at $f = 0$, with a "plateau" from about $f = .4$ to $f = .5$. Clearly, $M = 8$ is not an optimal choice. The Parzen spectral density estimate for $M = 175$ shown in Figure 4.11(f) has sharp peaks but with much of the "jagged" appearance of the sample spectral density. The use of $M = 2\sqrt{n}$ is simply a guide. For this particular dataset, $M = 50$ seems to produce a smooth spectral estimate with sharper peaks than for $M = 28$. Try it!

Key Point: The windowed spectral estimators (properly used) can do an excellent job of smoothing the sample spectral density and providing information about frequency content in the data.

4.2.3.3 Parzen Spectral Density Estimate vs Sample Autocorrelations

Figure 4.12(a) is a plot of the "two-frequency" realization considered in Example 4.1, while Figures 4.12(b) and (c) are plots of the associated sample autocorrelations and Parzen spectral estimator (with default truncation point $M = 28$), respectively. We noted that the model-based spectral density is a better tool than the autocorrelations for identifying underlying frequency content. Figure 4.12 shows that similarly, the Parzen spectral density estimate does a better job of detecting frequency content than do the sample autocorrelations. The two frequencies show up clearly in Figure 4.12(c), while there is absolutely no indication of the high frequency behavior in the sample autocorrelations! The plots in Figure 4.12 can be obtained using the following R code:

```
x=gen.arma.wge(n=200,phi=c(0.1300,1.4414,-0.0326,-0.8865),sn=9310)
# this generates an AR(4) realization (details in Chapter 5)
plotts.sample.wge(x)
```

6 The Parzen window has the property that the weights, λ_k, damp to zero as k gets large. Parzen's window uses a cubic damping function. Other windows include the Bartlett window (triangular damping),Tukey's window (a tapered cosine), and a host of others, mostly named after the researcher. We have chosen to use the Parzen window, although there are a variety of alternatives that might be preferable in certain circumstances. Emmanuel Parzen was a giant in the history of time series analysis and a friend of the first author. For those interested in investigating spectral windows, see Jenkins and Watts (1968) as a good starting point.

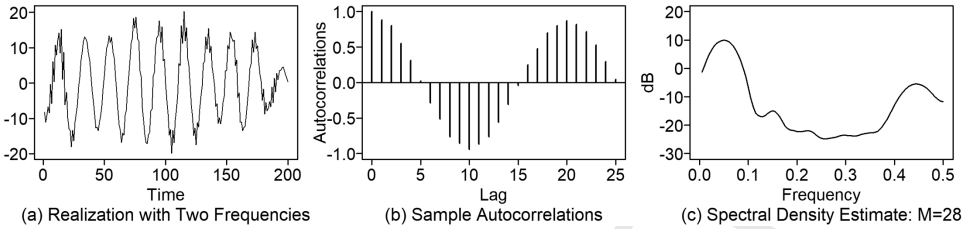


FIGURE 4.12 (a) Realization in Figure 4.9(a), (b) sample autocorrelations, and (c) Parzen spectral density estimate with $M = 28$ calculated from the data in (a).

Example 4.3 Parzen spectral density estimate for the "sum-of-sines data" in Figure 4.6(a).
We return to the "sum-of-sines" dataset shown in Figure 4.6(a), which is shown again in Figure 4.13(a). As mentioned earlier, this is actually the deterministic curve specified in (4.1) which is a sum of three sinusoidal components with frequencies .04, .125, and .25, respectively. Using the following commands, we obtain the plot in Figure 4.13.

```
y1=rep(0,100)
y2=rep(0,100)
y3=rep(0,100)
y=rep(0,100)
for(t in 1:100){y1[t]= 4*sin(2*pi*.04*t)}
for(t in 1:100){y2[t]=1.25*sin(2*pi*.125*t+1)}
for(t in 1:100){y3[t]=.5*sin(2*pi*.25*t+2.5)}
y=y1+y2+y3
par(mfrow=c(1,3))
plottsw.wge(y)
parzen.wge(y,trunc=20)
parzen.wge(y,trunc=30)
```

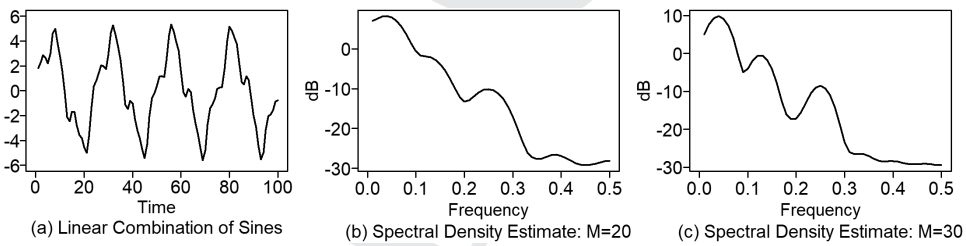


FIGURE 4.13 (a) Sum-of-sines curve previously shown in Figure 4.6(a); (b) and (c) Parzen spectral density estimates for the data in (a) using $M = 20$ (default) and $M = 30$, respectively.

Note that the Parzen spectral density estimator does not "know" the inside information about the frequencies in (4.1). Based on the data, for which it was difficult to visually discern the various frequencies in the data, the Parzen spectral density estimate in Figure 4.13(b) has a weak peak at about $f = .04$, more like an inflection point at about $f = .125$, and a peak at about $f = .25$. For $n = 100$, the "common" choice for M is $2\sqrt{100} = 20$, and this is the truncation point, M, which the command **`parzen.wge(y)`** uses by default. Trying another value for M, we issued the command:

```
parzen.wge(y,trunc=30)
```

and obtained Figure 4.13(c) which shows the three frequency components more clearly than in Figure 4.13(b). Notice also that the multipliers in (4.1) for frequencies .04, .125, and .25 decreased from 4

to 1.25 to .5, respectively. Correspondingly, the peaks in the spectral density in Figure 4.13(c) decrease in height (and again, the Parzen spectral estimator was not "provided" with that information).

Key Point: Higher peaks tend to indicate stronger behavior at the associated frequency.

Parzen spectral density estimate vs sample autocorrelations: Before leaving this example, we show the sample autocorrelations and the Parzen spectral density estimate in Figures 4.14(b) and (c) for the sum-of-sines data, which are plotted again in Figure 4.14(a). We note that the sample autocorrelations in Figure 4.14(b) have the form of a damped sinusoid with a period of about 25 which corresponds to a period of $f = 1/25 = .04$. There is absolutely no indication in the sample autocorrelations of the other two frequencies ($f = .125$ and $f = .25$). However, all three frequency components are identified by the spectral estimate in Figure 4.14(c). This re-emphasizes the fact that the spectral density is the proper tool for identifying frequency content in data. Letting **y** denote the vector obtained using the above code for the sum-of-sines, the plots in Figure 4.14 can be obtained using the ***tswge*** command:

```
plotts.sample.wge(y,trunc=30)
```

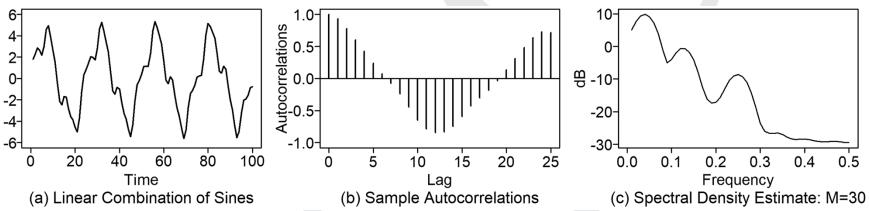


FIGURE 4.14 (a) Sum-of-sines data and corresponding (b) sample autocorrelations and (c) Parzen spectral estimate ($M = 30$).

4.2.3.4 Why We Plot Spectral Densities in Log Scale

Recall that the previous spectral densities have been plotted in log scale (in our case, dB). To understand the reasoning for this, consider Figure 4.15. Figure 4.15(a) is the Parzen spectral density estimate using $M = 30$ for the sum-of-sines data shown previously in Figure 4.14(a) and plotted in dB. Figure 4.15(b) shows the Parzen spectral estimator with $M = 30$ plotted in non-log scale. In this plot, we see that the main peak (at $f = .04$) dominates the plot, and the other two peaks are barely visible.

Key Point: Plotting spectral densities in log scale allows secondary peaks to be more visible.

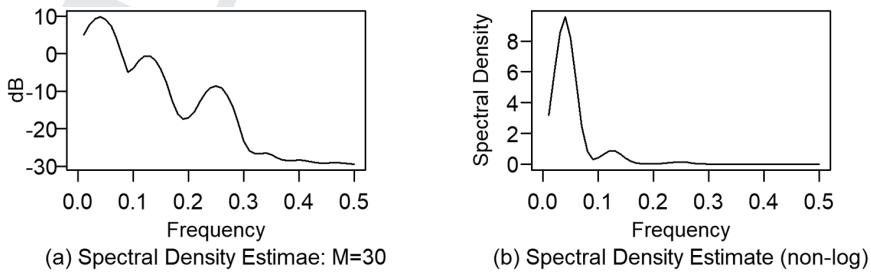


FIGURE 4.15 (a) Parzen spectral density estimate using $M = 30$ for the sum-of-sines data in Figure 4.14(a) plotted in dB, and (b) Parzen spectral density estimate in (a) but plotted in non-log scale.

Example 4.4 Spectral density estimates for real datasets.
In this example, we will find spectral estimates for some datasets of interest.

(1) Datasets Shown in Figure 4.4
Figures 4.4(a) and (b) show plots of the sunspot data and the log-lynx data. These were given as examples of cyclic/pseudo-periodic data with cycle lengths of about 11 and 10, respectively. That is, the data have some cyclic-type behavior, but the data are not technically periodic. In Figure 4.16, we show these datasets along with their sample autocorrelations and Parzen spectral density estimates, using the default truncation point = $2\sqrt{n}$. As expected, both plots of the Parzen spectral density estimate in Figure 4.16(c) and (e) have peaks at about $f = .10$. The spectral density for the sunspot data also shows a peak at $f = 0$. Examination of the sunspot data shows that there is some longer cyclic or wandering behavior related to the peaks. The log-lynx data have a peak only at $f = .10$, consistent with the observation that there seems to be no other pattern in the data. For both datasets, the sample autocorrelations show a damped sinusoidal behavior with period of about 10. The plots in Figure 4.16(a)–(c) can be obtained using the commands:

```
data(ss2.0)
plotts.sample.wge(ss2.0)
```

while those in Figures 4.16(d)–(f) are obtained using the code

```
data(llynx)
plotts.sample.wge(llynx)
```

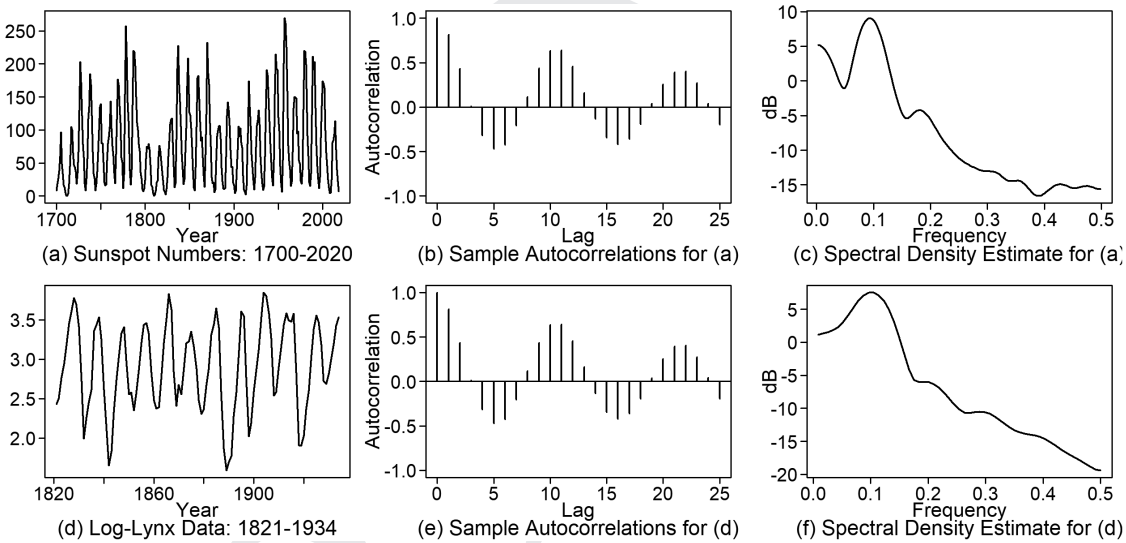


FIGURE 4.16 (a) Sunspot data (1700–present), (b) sample autocorrelations, and (c) Parzen spectral density for (a), (d) log-lynx data, (e) sample autocorrelations and (f) Parzen spectral density estimate for (c).

Key Point: It is worth repeating the fact that frequency content in a realization is represented by "peaks" in the spectral density.

(2) Datasets Shown in Figure 4.5
In Figure 4.5 we show datasets for the monthly West Texas intermediate crude oil price from January 1990 through December 2020, and the monthly DOW closing price from March 1985 through December

2020. These plots were shown to illustrate aperiodic data which have a wandering behavior. The crude oil prices have an unpredictable wandering behavior with upward trending behavior ending in a dramatic peak of about $120/barrel in the summer of 2008 followed by an equally dramatic drop to $34 dollars/barrel toward the end of that year. The prices increased and then remained at about $100/barrel in the years 2012−2014 but have remained low since that time with a catastrophic drop to about $15/barrel in the spring of 2020. The general picture is one of aperiodic and volatile behavior. The DOW data are also aperiodic with a general tendency to increase but with "dips along the way". Figure 4.17 shows these datasets along with the Parzen spectral density estimates using the default value for truncation point, M, in each case. The spectral density for each of the datasets has a distinct peak at $f = 0$, which would be expected. Additionally, the sample autocorrelations have a slowly damping exponential appearance for both datasets. Specifically, there is a strong positive correlation between data values close to each other in time. This is true in the crude oil data despite a few dramatic changes in oil price. A horizontal line is drawn at the mean of the data values in Figures 4.17(a) and (d), where it can be seen that there tend to be long sections of time at which the data values stay above or below the mean.

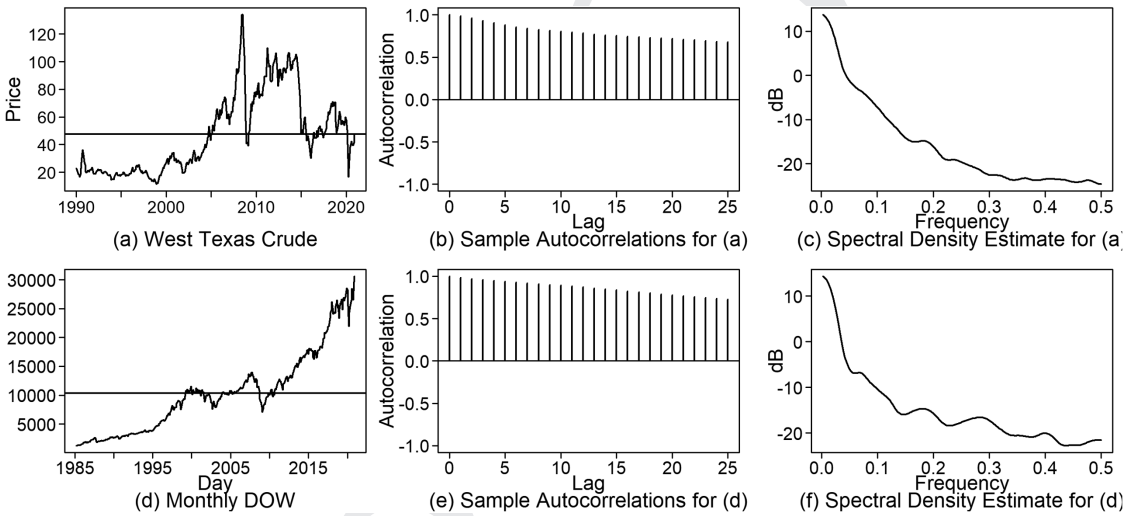


FIGURE 4.17 (a) Monthly West Texas Intermediate Crude price from January 1990 through December 2020 (b) sample autocorrelations, and (c) Parzen spectral density estimate for (a). (d) Dow Jones monthly closing price from March 1985 through December 2020, (e) sample autocorrelations, and (f) Parzen spectral density estimate for (c).

Key Point: A spectral density whose only peak is at $f = 0$, as in Figures 4.17(c) and (f), indicates that the realization shows no cyclic behavior associated with a period of less than n.

(3) A White Noise Dataset

In Figure 4.10(b) we plotted the spectral density for white noise and showed it to be the constant $S_X(f) = 1, 0 \le f \le .5$. We note that if we had plotted this spectral density in dB, then the constant horizontal line would have been at $10log_{10}(1) = 0$. Figure 4.18(a) shows a white noise realization of length $n = 100$. This realization has the typical random behavior associated with white noise. It is important to note that frequency behavior is difficult to recognize in Figure 4.18(a), but this is not because there is no cyclic at all (as in Figures 4.17(a) and (c)). On the contrary, all frequencies are present in white noise with equal intensities.

Figure 4.18(b) is a plot of the Parzen spectral density estimate in dB (using the default value $M = 20$). At first glance, the spectral density plot is disappointing in that it does not look flat at all. However, notice

that the plot is centered around zero and that there seem to be about as many dips as peaks. Also, note the vertical scale on Figure 4.18(b). In this plot, dB ranges from about −3.5 to 2. Contrast this with the spectral plots in Figure 4.17(c) and (f), where dB ranges from −20 to 10, which is similar to the dB range for the spectral density plots in Figure 4.16. That is, the spectral density in Figure 4.18(b) does not vary much from zero. In Figure 4.18(c), we show this spectral density plot with dB ranging from −10 to 10. Given this rescaling of the spectral density, it is seen that the Parzen spectral density estimate is fairly flat and is centered around the horizontal line at dB= 0.

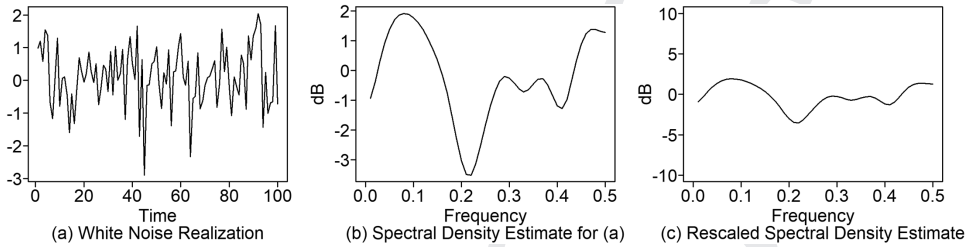


FIGURE 4.18 (a) A white noise realization, (b) the Parzen spectral density estimate for the data in (a), and (c) Figure 4.18(b) with expanded vertical scale.

Key Point: All frequencies are present in white noise with equal intensities.

4.3 SMOOTHING AND FILTERING

Filters are used in many fields of science and engineering to "filter out" certain types of frequencies from a dataset. The centered moving average smoother discussed in Chapter 2 is a type of filter. In Chapter 2, we discussed smoothing of time series data to remove "noisy" parts of a realization and retain (or pass through) the true "signal". The main smoother discussed in that section is the centered moving average smoother. Equations 2.1 and 2.3 give the general formulas for centered moving average smoothers of odd and even orders, respectively.

4.3.1 Types of Filters

Figure 4.19(a) shows the monthly West Texas intermediate crude oil price shown in Figure 4.17(a), among other places. Figure 4.19(b) shows the result of a 9th-order moving average smoother applied to the crude oil data. There we see that the data have been smoothed (that is, the high frequency "wiggle" has been removed) while the major movements of oil prices described in the discussion of Figure 4.17(a) are "passed through". Because the moving average smoother passed through the low-frequency part of the data and removed the high-frequency part of the data, we say that the moving average smoother is an example of a *low-pass filter*.

In Figure 4.19(c) we show the differenced crude oil data. That is, we calculate the differences $x_t - x_{t-1}$, where x_t is the crude oil price at month t. Note that $x_1 - x_0$ cannot be computed, so we are only able to calculate the differences $x_t - x_{t-1}$ for $t = 2,\ldots,n$. Taking a slight liberty with the time index notation, we define the difference $x_t - x_{t-1}$ to be d_{t-1}, so that we have the differenced data $d_t, t = 1,2,\ldots,n-1$. The differenced data are interesting and appear to have done the exact opposite of smoothing. That is, the differencing of

the data retained (or passed through) the "high-frequency wiggle" and tended to remove the low-frequency wandering behavior. Because of this fact, we say that differencing the data is a form of *high-pass filtering*.

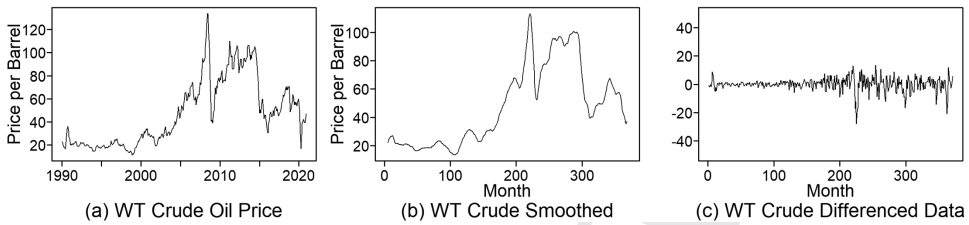


FIGURE 4.19 (a) Monthly West Texas Intermediate Crude oil price from January 1990 through December 2020, (b) Smoothed version of (a) using a centered moving average smoother of order 9, and (c) Differenced crude oil price data.

```
# Note: This code is for WT Crude from Jan 1990 through December 2020
data(wtcrude2020)
n=length(wtcrude2020)
parz.wt=parzen.wge(wtcrude2020,plot=FALSE)
s9=ma.smooth.wge(wtcrude2020,order=9,plot=FALSE)
parz.s9=parzen.wge(s9$smooth[5:114],plot=FALSE)
d1.wt=rep(NA,n)
for(t in 2:n) d1.wt[t-1]=wtcrude2020[t]-wtcrude.2020[t-1]
parz.d1=parzen.wge(d1.wt[1:n-1],plot=FALSE)
par(mfrow=c(1,3),mar=c(4.5,3,2,.5))
plot(wtcrude2020,type='l',xlab='(a) WT Crude Oil Price')
plot(s9$smooth,type='l',xlab='(b) WT Crude Smoothed')
plot(d1.wt,type='l',xlab='(c) WR Crude Differenced data',ylim=c(-50,50))
```

Key Points

1. *Low-pass filters* pass through lower frequencies and remove higher frequencies
   - low-pass filtered data has a smoother appearance than the original data
   - a centered moving average smoother is an example of a low-pass filter.
2. *High-pass filters* pass through higher frequencies and remove lower frequencies
   - high-pass filtered data have a noisy/high-frequency appearance
   - a difference is an example of a high-pass filter
   - high-pass filters are often used to remove low-frequency noise, eliminate humming sounds in audio signals, and so forth.
3. *Band-pass filters* pass through behavior in a certain band (or range) of frequencies.
4. *Band-stop filters* remove behavior in a particular band of frequencies and retain the frequencies outside this band.

The two filters we have discussed to this point are the centered moving average smoother which is a low-pass filter, and the first difference which is a high-pass filter.

Example 4.5 Smoothing and differencing the two-frequency data in Example 4.1

In this example, we consider filtering the data in Figure 4.12 which consist of distinct low-frequency behavior at $f = .05$ and high-frequency behavior at $f = .45$. Figures 4.20(a) and (b) are plots of the data and Parzen spectral density estimate, which have been shown previously but are included here for purposes of comparison. In Example 4.1 we noted the behavior associated with the ten cycles, each of approximate length 20. This is seen as a peak in the Parzen spectral density estimate at about $f = .05$. Similarly, the

high-frequency behavior is evidenced by the "noisy wiggle" in the data and a peak in the Parzen spectral density estimate at about $f = .45$.

Figures 4.20(c) and (d) show the smoothed data after applying a 6th-order centered moving average smoother. The effect of the smoothing (low-pass filter) is to remove the "wiggle" from the realization and essentially eliminate the peak in the Parzen spectral density estimate at $f = .45$.

Figure 4.20(e) shows the differenced data $(x_t - x_{t-1})$. In this figure, the high-frequency behavior is accentuated but there is still some indication of the ten cycles that were present in the original data. The Parzen spectral density estimate in Figure 4.20(f) shows that the peak at $f = .45$ is higher than the peak at $f = .05$ (indicating that the peak at $f = .45$ is more dominant, but there is still a distinct peak at $f = .05$).

Plots in Figure 4.20 can be obtained using the following commands:

```
x=gen.arma.wge(n=200,phi=c(0.1300,1.4414,-0.0326,-0.8865),sn=9310)
n=length(x)
par(mfrow=c(3,2),mar=c(3,2.5,1,1.5))
plot(x,type='l',ylim=c(-20,20))
parzenww.wge(x)
s6=ma.smooth.wge(x,order=6,plot=FALSE)
plot(s6$smooth,type='l',ylim=c(-20,20))
parzenww.wge(s6$smooth[4:197])
for(t in 2:n) d1[t-1]=x[t]-x[t-1]
plot(d1,type='l',ylim=c(-20,20))
parzenww.wge(d1[1:n-1])
```

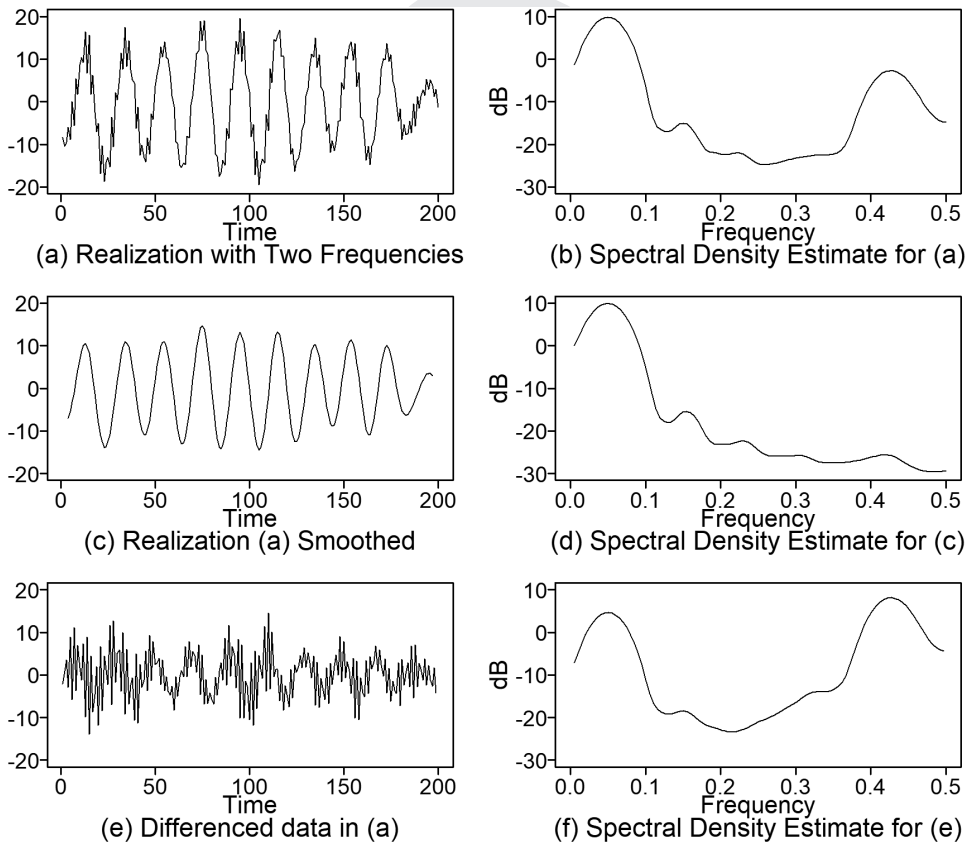


FIGURE 4.20 (a) Two-frequency data discussed in Example 4.1, (b) Parzen spectral estimate of the "two-frequency" data, (c) data after applying a 6th-order moving average smoother, (d) Parzen spectral density estimate of the data in (c), (e) data in (a) differenced, and (f) Parzen spectral density estimate of the data in (e).

Key Points: Regarding the data in Figure 4.20(a):
1. The 6th-order moving average smoother did a good job of removing the high-frequency behavior and retaining a "smoother" version of the low-frequency behavior.
2. The difference acted as a high-pass filter, but it did not sufficiently remove the low-frequency behavior that was dominant in the realization.

4.3.2 The Butterworth Filter

In Example 4.5, we noted that while differencing is a method of high-pass filtering, it is not a very good one. That is, while it did accentuate the high-frequency behavior in the data in Example 4.5, it did not really remove the low-frequency signal. We say that the $f = .05$ behavior "leaked" through the difference filter. What is needed is a method of filtering that does a better job of retaining a certain range of frequency behavior while at the same time removing parts of the signal related to unwanted frequencies. One of the more common filters used by scientists and engineers for this purpose is the Butterworth filter (see Butterworth, 1930; Porat, 1997).[7]

The low-pass Butterworth filter is designed to remove frequencies above a specified "cutoff" frequency, and a high-pass Butterworth filter works similarly. While the Butterworth filter can perform a variety of filtering methods, the ***tswge*** function **`butterworth.wge`** is designed to produce a low-pass or high-pass Butterworth filter on an R vector. The **`butterworth.wge`** function has the general form

```
butterworth.wge(x,order,type,cutoff,plot)
```

where type is either **`'l'`** for low-pass or **`'h'`** for high-pass. The parameter **`cutoff`** specifies the cutoff frequency, and **`order`** specifies the order of the Butterworth filter. That is, if we apply a Butterworth filter to remove frequencies above $f = .2$, we could apply a low-pass Butterworth filter with **`cutoff=.25`**. We have found that a good choice is **`order=4`**, which is the default. The function also provides the option to plot the original and filtered data, and **`plot=TRUE`** is the default.

Example 4.5 (revisited) Using the Butterworth filter on the two-frequency data
In this example, we apply low-pass and high-pass Butterworth filters in an attempt to recover a smooth pseudo-sinusoidal signal via a low-pass Butterworth filter along with a signal representing the high-frequency component of the data without the low frequency sinusoidal component. The plots in Figure 4.21 were obtained using the following code:

```
x=gen.arma.wge(n=200,phi=c(0.1300,1.4414,-0.0326,-0.8865),sn=9310)
par(mfrow=c(3,2),mar=c(3,2.5,1,1.5))
plot(x,type='l',ylim=c(-20,20))
parzenww.wge(x)
low=butterworth.wge(x,type='l',cutoff=.25,plot=FALSE)
plot(low$x.filt,type='l',ylim=c(-20,20))
parzenww.wge(low$x.filt)
high=butterworth.wge(x,type='h',cutoff=.25,plot=FALSE)
plot(high$x.filt,type='l',ylim=c(-20,20))
parzenww.wge(high$x.filt)
```

QR 4.5 Butterworth Filter – Example 4.5

7 Letting x_t and y_t denote the original and filtered data, respectively, the Butterworth filter is a recursive filter defined by $y_t = \sum_{k=0}^{N} \alpha_k x_{t-k} - \sum_{k=0}^{N} \beta_k y_{t-k}$, where the α_ks and $\beta_k s$ are selected by the function to provide the appropriate filter.

Figure 4.21 shows that the Butterworth filter did precisely what it was intended to do. That is, the low-pass filter smoothed the data and eliminated the peak in the spectral density at $f = .45$ while retaining the peak at $f = .05$. Similarly, the high-pass filter eliminated the low-frequency pseudo-sinusoidal behavior and retained the high-frequency wiggle. The peak in the spectral density at $f = .05$ is removed, while the one at $f = .45$ is retained. In particular, the high-pass filter performed much better than the difference filter.

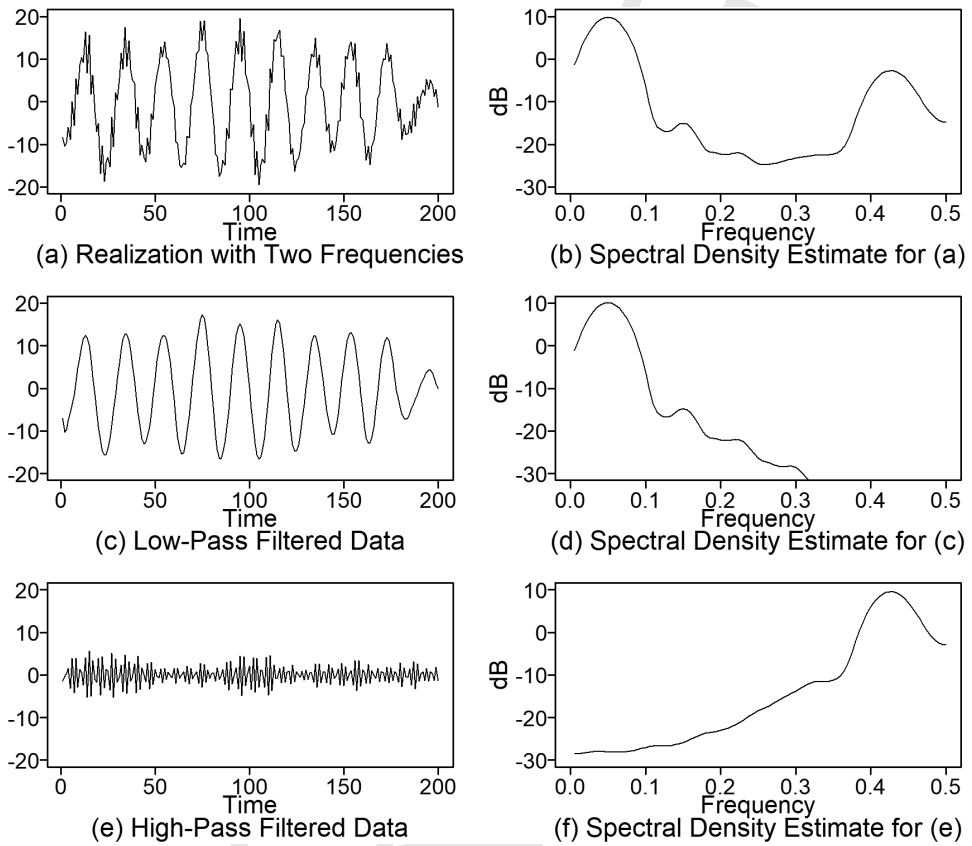


FIGURE 4.21 (a) Two-frequency data discussed in Example 4.1, (b) Parzen spectral estimate of the "two-frequency" data, (c) data in (a) after applying low-pass Butterworth filter, (d) Parzen spectral density estimate of the data in (c), (e) data in (a) after applying high-pass Butterworth filter, and (f) Parzen spectral density estimate of the data in (e).

Key Points

1. The Butterworth filter can be "fine-tuned" to retain and remove desired frequencies.
2. Through use of the cutoff frequency, the Butterworth filter can be designed to remove (or retain) specified frequency ranges.
   - the centered moving average smoother and the difference do not have this type of "frequency control".
3. The Butterworth filter has "end-effects" issues. The ***tswge*** function **`butterworth.wge`** does not control for these, so data values toward the ends of the filtered data should not be viewed as reliable.

- The larger the filter order, the more values are subject to end-effects problems.
- This is like the centered moving average filter, which cannot be applied to ends of the data.
- The number of undefined values increases as the order increases.

4. The Butterworth filter (either low-pass or high-pass) can be best implemented by understanding the frequency behavior in the data.

4.4 CONCLUDING REMARKS

In this chapter we have provided tools for understanding the frequency content of data. We formalized the concepts of frequency and period, and we discussed cyclic, pseudo-periodic, and aperiodic data. Our treatment is atypical in introductory applied time series texts in that we discuss the frequency domain early and then use the associated tools throughout the remainder of the book to better understand models, forecasts, and so forth.

We discussed the information available in the spectral density, and we presented the Parzen spectral density estimator as a tool for estimating the spectral density given a set of time series data. As we discuss time series models in the following chapters, we will often use information provided by the spectral density to understand and formulate models.

Smoothing is commonly used in business and economic applications to isolate key features of a dataset. For example, centered moving average smoothing was discussed in Chapter 2 as a technique for removing noisy components of data to better observe the dominant behavior. In this chapter we discussed filtering as a generalization of smoothing. We introduced the Butterworth filter, which also provides the ability to smooth data (that is, low-pass filter), but in this case, smoothing is done by controlling which "high-frequency" behavior is to be removed. The Butterworth high-pass filter was introduced as a powerful tool for emphasizing subtle high-frequency behavior that is obscured by the dominating low-frequency features of the data.

APPENDIX 4A

TSWGE FUNCTIONS

(a) **`parzen.wge(x, dbcalc, trunc, plot)`** calculates and alternatively plots the spectral density estimate based on the Parzen window.

`x` is a vector containing the time series realization $x_t, t = 1,\ldots,n$
`dbcalc` is a logical variable. **`dbcalc = "TRUE"`** (default) calculates the smoothed spectral estimator on the log scale (in dB)
`trunc` (integer ≥0) specifies the truncation point.
`trunc = 0` (default) (or no value for **trunc** is specified) indicates that the default truncation point is $M = 2\sqrt{n}$
`trunc > 0` is a user-specified truncation point, i.e. calculations will be based on **M = trunc**

plot is a logical variable. **plot = "TRUE"** (default) produces a plot of the smoothed spectral estimator (in log or non-log scale depending on dB).

Example:

```
data(ss2.0); ssparz=parzen.wge(ss2.0)
```

calculates and plots the smoothed spectral estimator (in dB) for the sunspot data (**ss2.0**) using the default truncation point, $M = 2\sqrt{319} = 36$. The vectors **ssparz$freq** and **ssparz$pgram** contain the frequencies and associated Parzen-based spectral estimates.

(b) **butterworth.wge(x,order,type,cutoff)** is a ***tswge*** R function that requires functions in the CRAN package **signal** which should be installed on your machine before calling **butterworth.wge.**

x is a vector containing the time series realization $x_t, t = 1,\ldots,n$

order is the order of the Butterworth filter you want to use (default = 4). **type** should be either **"low", "high", "pass",** or **"stop"** for a low-pass, high-pass, band-pass, or band-stop (notch) filter, respectively.

cutoff

- if **type = "high"** or **"low",** then **cutoff** is the scalar cutoff frequency
- if **type = "stop"** or **"band",** then **cutoff** is a 2-component vector containing the lower and upper cutoff frequencies

Example: The dataset **wages** is contained in ***tswge***. The ***tswge*** R commands to perform the low-pass filter described in Example 2.4 are

```
data(wages)
wages05 = butterworth.wge(wages,order = 4,type = "low", cutoff = 0.05)
```

The filtered data set is **wages05$x.filt.**

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

$$x_t = b0 + b1(t) + coef[1]*cos(2\pi freq[1]t + psi[1]) + coef[2]*cos(2\pi freq[2]t + psi[2]) + z_t$$

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

Notes

(1) The AR process, z_t, serves as the "noise" in the above signal-plus-noise model. We will see in Chapter 5 that an AR process is "driven" by a white noise process, a_t. In ***tswge***, we assume the white noise process to be normally distributed with zero mean. The variance of the white noise process is specified using **vara**.

(2) If you want the "noise" in the signal-plus-noise model to be white noise, then use the default value for **phi** (=0).
plot is a logical variable. **plot = TRUE** (default) produces a plot of the generated realization.
sn determines the seed used in the simulation. (default = 0) (see note below)

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

Example: The command

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

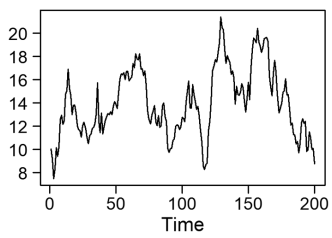
calculates

$$x_t = 1.5\cos\left(2\pi(.05)t + 1.1\right) + 3.5\cos\left(2\pi(.2)t + 2.8\right) + z_t,$$

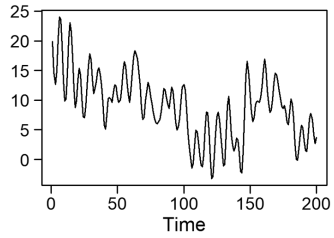
where z_t is a realization from the AR(1) process $z_t = .7z_{t-1} + a_t$, where a_t is a zero-mean white noise process with variance $\sigma_a^2 = 2$. The realization is based on a randomly selected seed (based on **sn**) and the resulting signal-plus-noise realization is placed in the vector **x**.

PROBLEMS

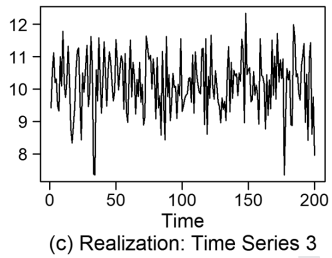
1. For each of the following datasets, use **plotts.sample.wge** to plot the data, sample autocorrelations, and Parzen spectral density estimate (using the default truncation point $M = 2\sqrt{n}$). For each dataset, explain how the sample autocorrelations and Parzen spectral density estimate describe (or fail to describe) the behavior in the data:
   (a) **dfw.mon**
   (b) **wtcrude2020**
   (c) **sunspot2.0**
   (d) **llynx** (log of the lynx data)
   (e) **doppler**
   (f) Simulated data **f=gen.arma.wge(n=200,phi=c(2.55,-2.42,.855),mu=10, sn=7285)**
   (g) Simulated data **g=gen.arma.wge(n=200,phi=c(.65,.62,-.855),mu=50, sn=5698)**
   (h) Simulated data **h=gen.sigplusnoise.wge(n=250, coef=c(2,1),freq= c(.05,.28), psi=c(1.2,2),phi=.9,sn=278)**
2. Following are displayed three sets of figures, each containing four plots. The first set shows four realizations of length $n = 200$, each generated from a time series model. The second and third sets contain four autocorrelation functions and four spectral densities, respectively. Match each realization with the corresponding autocorrelations and spectral density. Explain your answers.



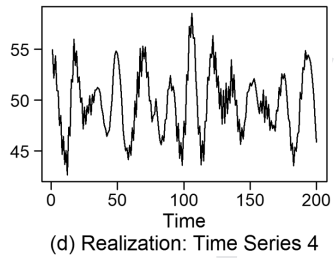
(a) Realization: Time Series 1



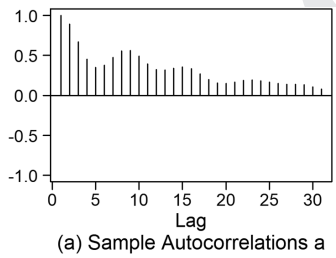
(b) Realization: Time Series 2



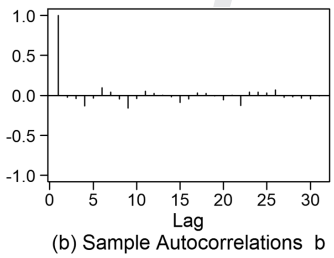
(c) Realization: Time Series 3



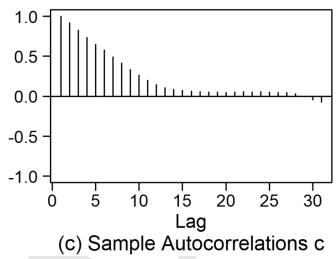
(d) Realization: Time Series 4



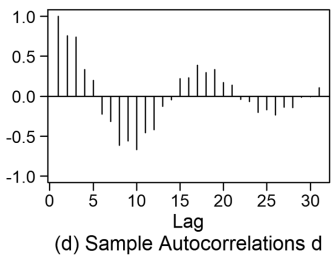
(a) Sample Autocorrelations a



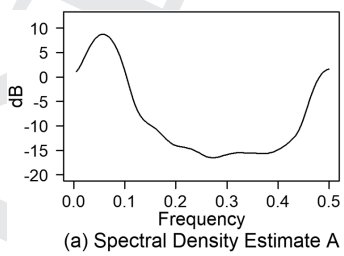
(b) Sample Autocorrelations b



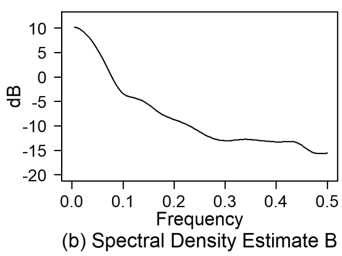
(c) Sample Autocorrelations c



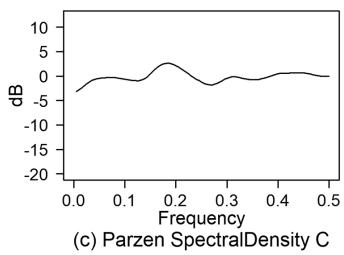
(d) Sample Autocorrelations d



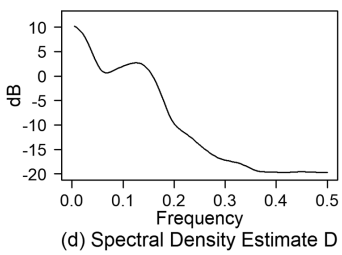
(a) Spectral Density Estimate A



(b) Spectral Density Estimate B



(c) Parzen SpectralDensity C



(d) Spectral Density Estimate D

3. Using the ***tswge*** command below, generate a time series realization.

```
y=gen.sigplusnoise.wge(n=100,coef=c(2,0),freq=c(.1,0),psi=
c(0,0),phi=0,sn=17)
```

   (a) Using the description for **gen.sigplusnoise.wge** in the appendix to this chapter, write down the equation satisfied by **y**.
   (b) Plot the data in vector **y**.
   (c) Filter the data in **y** with a high-pass Butterworth filter with cutoff=.05. Plot the filtered data, and describe and explain the results.
   (d) Filter the data in **y** with a low-pass Butterworth filter with cutoff=.05. Plot the filtered data, and describe and explain the results.
   (e) Filter the data in **y** with a high-pass Butterworth filter with cutoff=.15. Plot the filtered data, and describe and explain the results.
   (f) Filter the data in **y** with a low-pass Butterworth filter with cutoff=.15. Plot the filtered data, and describe and explain the results.
   (g) Apply a centered moving average smoother of order 5 to the data in **y**. Plot the smoothed data, and describe and explain the results.

4. Using the ***tswge*** command below, generate a time series realization.

```
x=gen.arma.wge(n=50,phi=c(1.83000,-1.97500,1.82085,-0.97510),sn=2787)
```

   In the following, you will be asked to discuss the frequency behavior in the data and Parzen spectral density estimate. In each case, we recommend that you use **plotts.sample.wge** on the dataset (or filtered dataset) under discussion.

   (a) Discuss the frequency behavior visible in the realization (**x**) and the Parzen spectral density estimate.
   (b) Use the Butterworth filter to perform a low-pass filter using cutoff=.12. Discuss the effect of the filter by discussing the frequency behavior visible in the filtered data and associated Parzen spectral density estimate.
   (c) Use the Butterworth filter to perform a high-pass filter using cutoff=.12. Discuss the effect of the filter by discussing the frequency behavior visible in the filtered data and associated Parzen spectral density estimate.
   (d) Use the Butterworth filter to perform a low-pass filter using cutoff=.23. Discuss the effect of the filter by discussing the frequency behavior visible in the filtered data and associated Parzen spectral density estimate. How does performance of this filter compare with the low-pass filter using cutoff=.12, and why?
   (e) Use the Butterworth filter to perform a high-pass filter using cutoff=.07. Discuss the effect of the filter by discussing the frequency behavior visible in the filtered data and associated Parzen spectral density estimate. How does performance of this filter compare with the high-pass filter using cutoff=.12, and why?
   (f) Use the Butterworth filter to perform a low-pass filter using cutoff=.3. Discuss the effect of the filter by discussing the frequency behavior visible in the filtered data and associated Parzen spectral density estimate. How would you explain the effect of the filter?
   (g) Use the Butterworth filter to perform a high-pass filter using cutoff=.3. Discuss the effect of the filter by discussing the frequency behavior visible in the filtered data and associated Parzen spectral density estimate. How would you explain the effect of the filter? (Notice the end-effects in the filtered data. How might you adjust for the end-effects?)