CHAPTER 5

LINEAR PREDICTION

5.1 INTRODUCTION

We saw in Chapter 3 that a tube closed at one end and open at the other supports standing waves that are related to the length of the tube. This concept can be generalized to a concatenation of tubes of differing widths and lengths. For modeling purposes, the vocal tract can be considered as straight (not curved) and, thus, approximated as a variable area tube. The linear wave equation applies, and wave motion in the tract is planar to a good approximation. We can illustrate that wave motion is planar as follows. The wavelength for sound at 4000 Hz is

$$\lambda = \frac{c}{f} = \frac{350 \text{ m/s}}{4000 \text{ Hz}} \cong 8.7 \text{ cm} \tag{5.1.1}$$

Note that 8.7 cm is greater than the average diameter of the vocal tract, which is typically of the order of 2 to 3 cm. Thus, for all practical purposes, we can consider sound propagation in the vocal tract to be planar. Consequently, the vocal tract shape can be approximated by the cross-sectional area as a function of position along the tract.

For vowels, the air flow can reach 700 cm^3/sec. However, the opening of the tract is usually less than 1 cm^2. This means that the Mach number is less than 0.02, which is low, thereby implying that the linear wave equation is a good approximation. Nonlinear effects only become important for the flow through the vocal folds and through narrow constrictions in the vocal tract, as in the production of fricatives.

For an ideal, lossless, rigid tube with a cross-sectional area A(x), one can use the approximate equations of motion and continuity to describe wave propagation in the vocal tract. If u(x, t) is the volume–velocity and p(x, t) is the pressure, then

$$\begin{aligned} \frac{\partial p}{\partial x} &= -\frac{p}{A}\frac{\partial u}{\partial t} && \text{equation of motion} \\ \frac{\partial u}{\partial x} &= -\frac{A}{\rho c^2}\frac{\partial p}{\partial t} && \text{equation of continuity} \end{aligned} \tag{5.1.2}$$

where ρ is the density of air and c is the velocity of sound. Note that these equations are the same as those for an RLC transmission line, and, thus, comparisons can be made between parameters associated with a tube and those with a transmission line, as illustrated in Figure 5.1.

For a tube of length L with a complex exponential excitation, one can use Equation (5.1.2) to calculate the frequency response and impedance of the tube. Upon doing so, we find that the frequency response is

$$\text{frequency response} = \frac{1}{\cos(2\pi fL/c)} \tag{5.1.3}$$

This equation has poles at

$$f_n = \frac{2n+1}{4L}c, \quad n = 0, \pm 1, \pm 2, \ldots \tag{5.1.4}$$

Acoustical parameter	Electrical parameter
p – Pressure	v – Voltage
u – Volume velocity	i – Current
ϱ/A – Air mass inertia (acoustic inductance)	L – Inductance
$A/(\varrho c^2)$ – Air compressibility (acoustic capacitance)	C – Capacitance
Viscous loss	R – Series resistance
Heat conduction loss	G – Shunt resistance
Yielding wall	Z_w – Shunt impedance

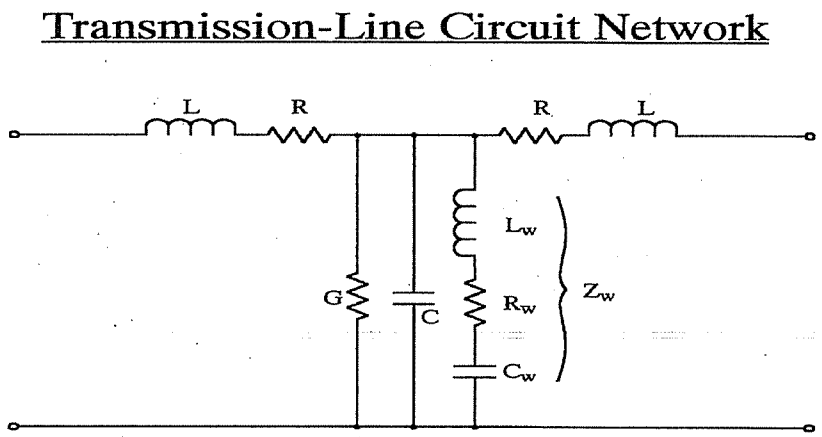


FIGURE 5.1 Analogies between acoustic and electrical transmission line parameters.

which is the result we mentioned in Chapter 3. For $c = 34{,}000$ cm/sec and $L = 17$, $f_n = (2n + 1)/4L \cong (2n + 1)500$ Hz, which are the odd multiples of 500 Hz. These are the natural frequencies, or eigenfrequencies, of the system model and they represent the formants of the vocal tract. Such formants occur regardless of the vocal tract shape. For example, one can concatenate two tubes of different widths and lengths and show that resonances (formants) appear in the system-transfer function just as they do for a single tube. However, the location of the resonances varies with the shape of the tubes. Thus, the formants change as the shape of the vocal tract changes.

In a nonideal tube, there are losses due to viscous friction, thermal conduction, and yielding walls. The effect of yielding walls is to slightly shift the low-frequency formants upward in frequency as well as increase their bandwidth. There is not much effect on the high-frequency formants. For example, the formant at 500 Hz can shift up to 504 Hz and have a bandwidth of 53 Hz. The effect is on the order of 0.8%.

The effect of radiation at the lips and nostrils is that of a high-pass filter approximated by differentiation in the time domain. This effect tends to "lift-up" the amplitude of the formants at the high frequencies.

Another factor is turbulence generated at a constriction. Turbulence occurs when the Reynolds number exceeds a critical value. The definition of the Reynolds number is

$$R_e = \frac{\rho w}{\mu} \frac{u}{A} \tag{5.1.5}$$

where ρ is the density of air, μ is the coefficient of viscosity, and w is the width of the constriction. For the glottis, w is the separation between the vocal folds. For a tract con-

striction, w is the average radius of the constriction. Noise is generated at the constriction only if the Reynolds number is greater than a critical value R_c. When this occurs, the root mean square (rms) value of the generated noise is proportional to the following

$$N \propto (R_e^2 - R_c^2) \tag{5.1.6}$$

At the glottis, the noise is usually considered a series pressure (voltage) source, while at a tract constriction, the noise can be modeled better as a shunt velocity (current) source.

For synthesis there are generally two excitation sources, a periodic excitation source for voiced sounds and a noise generator for unvoiced sounds. There are various waveform models for the periodic excitation. We will discuss these in a later chapter. We shall also consider a detailed model of the vocal tract in a later chapter, where we show how the transfer function is calculated and how the model can be used for speech synthesis.

In summary, the vocal tract can be modeled by a set of resonances (formants or poles) that depend on the vocal tract shape. The bandwidths of the lowest two formants depend primarily on wall losses. The bandwidths of the higher formants depend primarily on viscous friction losses, thermal losses, and losses due to radiation. The nasal tract also has poles. The nasal tract can be considered as parallel with the oral tract. Because the mouth is closed for nasal production, the oral tract can trap frequencies, thereby eliminating these frequencies in the radiated sound. This trapping of frequencies is modeled with zeros in the overall system-transfer function. Furthermore, the excitation is modeled as two sources: one periodic waveform generator (approximated with a two pole transfer function) for voiced sounds and one noise generator for unvoiced sounds. There is also radiation at the lips, which is modeled as a single zero, which is equivalent to differentiation in the time domain. The complete model is summarized in Figure 5.2, where A_V and A_N are gain terms for the voiced and noise source models, respectively. The major point of this model, from the point of view of this chapter, is that the vocal tract model V(z) can be an all pole model. This is why linear prediction modeling is so important in speech analysis, synthesis, and recognition. Therefore, in this chapter we develop an all-pole model for the vocal tract. This model is also used for spectral analysis.

5.2 ESTIMATES OF THE AUTOCORRELATION FUNCTION

Various applications call for autocorrelation estimates, including data modeling such as linear prediction and power spectral estimation, both of which we discuss in this chapter. There are two major estimators for the autocorrelation function that are used on a regular basis, both of which are implemented in MATLAB with the function `xcorr`.

Definition 5.1. The first estimate is called the **unbiased autocorrelation estimate**, and is expressed as

$$\hat{R}_{XX}(k) = \frac{1}{N - |k|} \sum_{i=0}^{N-1-|k|} X(i + |k|)X(i), \quad \text{for } |k| < N \tag{5.2.1}$$

The summation is sometimes expressed as from 1 to $N - |k|$, but the total number of terms

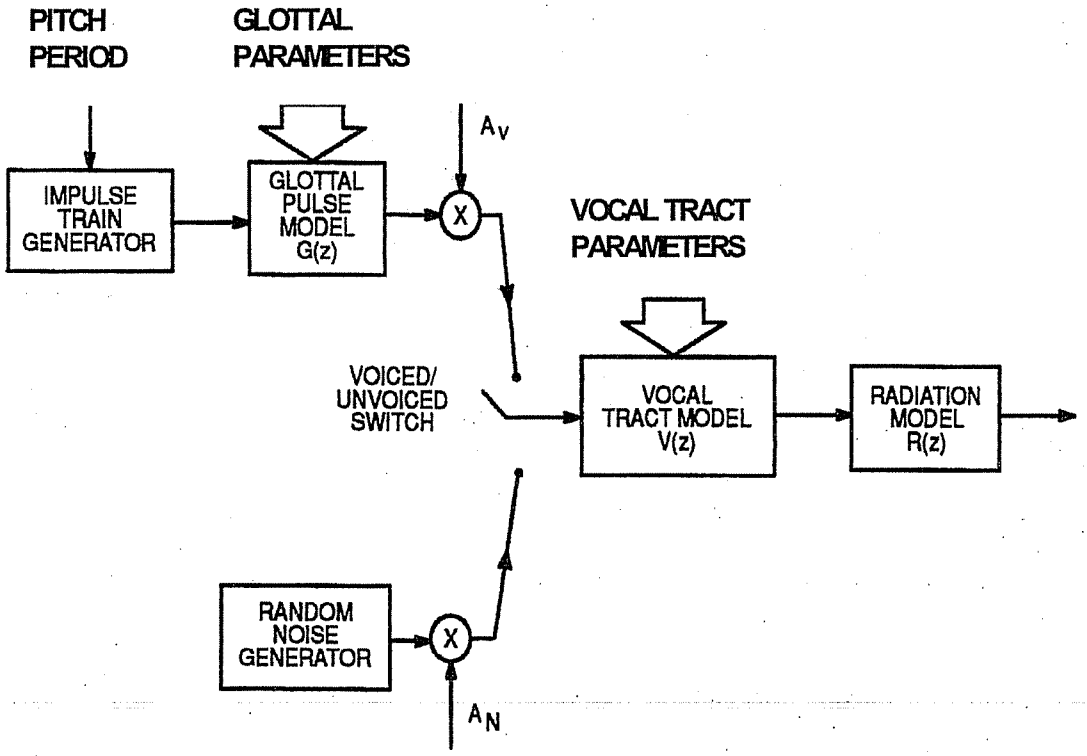


FIGURE 5.2 A model of vocal tract sound production.

is the same, namely, N, which denotes the number of data points available. The choice depends solely on the manner by which the data is indexed, that is, whether the initial value of the data starts at $i = 0$ or $i = 1$. Note that when we calculate Equation (5.2.1), we use an observed data record. Another comment is that we use upper case to denote that the process is a random process. In this case, the random process is a random sequence, since we are dealing with discrete data. Thus, X(n) is a random sequence, while x(n) is a particular member function (or sample function) of the random sequence. Also, the hat over the R denotes an estimate.

This estimate is unbiased, since the expectation of the estimate is the true autocorrelation

$$E[\hat{R}_{XX}(k)] = \frac{1}{N-|k|}\sum_{i=0}^{N-1-|k|} E[X(i+|k|)X(i)] = \frac{1}{N-|k|}\sum_{i=0}^{N-1-|k|} R_{XX}(k) = R_{XX}(k) \tag{5.2.2}$$

where E denotes the expected value. When the random variables are zero mean Gaussian, the variance of this estimate is approximately

$$VAR[\hat{R}_{XX}(k)] = \frac{1}{N-|k|}\sum_{i=-(N-1-|k|)}^{N-1-|k|}\left\{\left(1-\frac{|i|}{N-|k|}\right)(R_{XX}^2(i) + R_{XX}(i+|k|)R_{XX}(i-|k|))\right\} \tag{5.2.3}$$

Provided k is fixed, this estimate of the autocorrelation function approaches zero as N approaches infinity, making this estimate consistent. We discuss this more fully shortly. One problem with this estimate is that its largest value does not always occur at the origin.

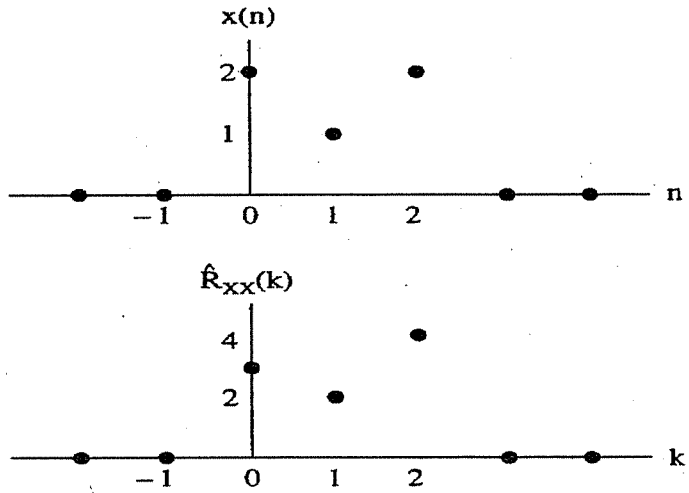


FIGURE 5.3 An example of an autocorrelation estimate that is not positive definite.

This is illustrated in the following example. For correlation matrices, this means that the estimate is not always positive definite.

EXAMPLE 5.1

Consider the data sequence, x(n), shown in Figure 5.3. The autocorrelation estimate is calculated as follows.

$$\hat{R}_{XX}(0) = \frac{1}{3}[2^2 + 1^2 + 2^2] = 3$$

$$\hat{R}_{XX}(1) = \frac{1}{2}[(2)(1) + (1)(2)] = 2$$

$$\hat{R}_{XX}(2) = \frac{1}{1}[(2)(2)] = 4$$

■

The last example shows that this autocorrelation estimate does not have its largest value at the origin.

Definition 5.2. Another estimate is called the **biased autocorrelation estimate**, and is given as

$$\hat{R}_{XX}(k) = \frac{1}{N}\sum_{i=0}^{N-1-|k|} X(i+|k|)X(i), \quad \text{for } |k| < N \tag{5.2.4}$$

where X(i) is a zero mean Gaussian random sequence. Although this estimate is biased, it is asymptotically unbiased as N increases for k fixed. And the variance is approximately

$$\text{VAR}[\hat{R}_{XX}(k)] = \frac{1}{N^2}\sum_{i=-(N-1-|k|)}^{N-1-|k|} \left\{(N-|i|-|k|)\left(R_{XX}^2(i) + R_{XX}(i+|k|)R_{XX}(i-|k|)\right)\right\} \tag{5.2.5}$$

This estimate is also consistent as N increases, provided k is fixed. It can also be shown that this estimate does have its largest value at the origin, which in correlation matrix terms means that this estimate is positive definite.

Given the above two possible autocorrelation estimates, which is the "best?" Suppose |k| approaches N; then the variance of the unbiased estimate becomes large, because

k, the shift in the data, is large. This shift k is often called the number of lags. When this shift or number of lags is large, then the estimate of the autocorrelation function is unreliable, since there is very little overlap of the data sequence with its shifted version. Consequently, this has lead to the rule of thumb that the number of lags should not be greater than 10% of the length of the data sequence or N/10. On the other hand, the variance of the biased estimate does not become large as k becomes large. In addition, the expected value of this estimate goes to zero, not to the true autocorrelation function. Since the bias is as large as the autocorrelation function we are estimating, we do not obtain a good estimate.

Now let k be fixed and let N increase. Then the variance of the biased estimate decreases as does the bias. The variance of the unbiased estimate also decreases. So we remain perplexed about which estimate is the "best." It is conjectured that the mean square error of the biased estimate is less than that for the unbiased estimate. Thus, it is recommended that the biased autocorrelation estimate be used when possible.

Calculating the autocorrelation estimate involves several practical matters. First, is the data sequence. This is typically of finite length and can be thought of as being observed through a window, where a typical window is rectangular. This process sets the data outside the window to zero. Thus, as we calculate the autocorrelation estimate for increasing k, we have less and less data contributing to the estimate and more and more zeros. This is one reason why the estimate becomes unreliable as the number of lags increases. Figure 5.4 illustrates this point using a sliding rectangular window to represent the data. Note that the total number of data points is N, where the range is zero to $N - 1$.

Sometimes the data may be periodically extended outside the observation window. Here the idea is that this periodic extension is "better" in some sense than using zeros outside the window. In either case, the experimenter must decide where such assumptions are justified.

A rectangular window may be replaced by a window with less sharp leading and trailing edges in many applications. The reason for this is that the fast rise and fall time of the leading and trailing edges of the rectangular window introduce large spectral sidelobes in the spectral estimate, as well as less high frequencies due to the window alone. This type of windowing distorts the spectral estimate. We will discuss this more later.

Both of the autocorrelation estimates can be calculated using the function xcorr in the MATLAB Signal Processing Toolbox. However, they can also be calculated using vector

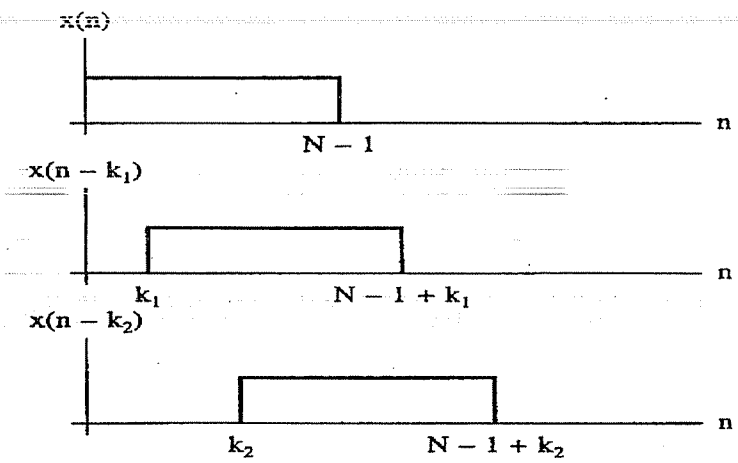


FIGURE 5.4 Illustration of shifting the data when calculating the autocorrelation estimate.

notation as follows. Let the data sequence be represented in vector form as

$$\begin{aligned}
\vec{x}(t)^T|_0 &= [x(0)\ x(1)\ x(2)\cdots x(N-1)\ \overbrace{0\ 0\ 0\cdots 0}^{K-1\text{ zeros}}] \\
\vec{x}(t)^T|_1 &= [0\ x(0)\ x(1)\ x(2)\cdots x(N-1)\ \overbrace{0\ 0\ 0\cdots 0}^{K-2\text{ zeros}}] \\
\vec{x}(t)^T|_2 &= [0\ 0\ x(0)\ x(1)\ x(2)\cdots x(N-1)\ \overbrace{0\ 0\ 0\cdots 0}^{K-3\text{ zeros}}] \\
\vec{x}(t)^T|_{k-1} &= [\overbrace{0\ 0\cdots 0}^{K-1\text{ zeros}}\ x(0)\ x(1)\ x(2)\cdots x(N-1)]
\end{aligned} \tag{5.2.6}$$

Note that Equation (5.2.6) could use either upper or lower case notation for the vectors and their elements. Since the discussion is primarily directed toward formulating data vectors, we adopted the lower case notation. Thus, we have K data vectors, numbered from zero to $K-1$, where $K<N$. The $K-1$ zeros appended to the end of the first data sequence and to the beginning of the last data sequence represents the largest value of the lag to be calculated in the autocorrelation estimate, that is, the number of lags ranges from 0 to $K-1$. The subscript on the vector represents the number of lags, k, in the data sequence. To calculate the biased estimate of the autocorrelation function we do the following.

$$\hat{R}_{XX}(0) = \vec{x}(t)^T|_0\vec{x}(t)|_0 = [x(0)\ x(1)\cdots x(N-1)\ 0\ 0\cdots 0]\begin{bmatrix} x(0) \\ x(1) \\ \cdot \\ \cdot \\ x(N-1) \\ 0 \\ 0 \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

In general, we calculate the autocorrelation estimate in the following manner. Note that N is the length of the data, not the length of the data plus the number of appended zeros.

$$\begin{aligned}
\hat{R}_{XX}(0) &= \frac{1}{N}\vec{x}(t)^T|_0\vec{x}(t)|_0 \\
\hat{R}_{XX}(1) &= \frac{1}{N}\vec{x}(t)^T|_1\vec{x}(t)|_0 \\
\hat{R}_{XX}(k) &= \frac{1}{N}\vec{x}(t)^T|_k\vec{x}(t)|_0
\end{aligned} \tag{5.2.7}$$

One can also create a data matrix, [X], of the data vectors. The matrix has $(N+K-1)$ rows and (K) columns. The autocorrelation matrix, $[\hat{R}_{XX}(k)]$, is calculated by multiply the

transpose of the data matrix by the data matrix as follows.

$$[X] = \begin{bmatrix} x(0) & 0 & 0 & \dots & 0 \\ x(1) & x(0) & 0 & \dots & 0 \\ x(2) & x(1) & x(0) & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & 0 \\ 0 & x(N-1) & & \dots & x(0) \\ 0 & 0 & x(N-1) & \dots & x(1) \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & \dots & x(N-1) \end{bmatrix} \tag{5.2.8}$$

Then

$$[\hat{R}_{XX}(k)] = \frac{1}{N}[X]^T[X] \tag{5.2.9}$$

This matrix calculation yields a $K \times K$ Toeplitz matrix, that is, a matrix that is symmetric about the main diagonal and one that has equal elements on the principal diagonal. This autocorrelation matrix is also positive definite. However, note that the matrix multiplication requires more computations than are necessary since the autocorrelation estimate is symmetric about the origin. In a similar manner we can calculate the unbiased autocorrelation estimate. However, if we do this by matrix multiplication, then the resulting autocorrelation matrix is not Toeplitz, and, furthermore, the matrix may not be positive definite. This is another reason why the unbiased estimate is not usually used.

EXAMPLE 5.2

Consider the data shown in Example 5.1. The biased autocorrelation matrix for $K = 3$ $(K - 1 = 2)$ is

$$[X] = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 2 & 0 \\ 2 & 1 & 2 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix} \quad [X]^T = \begin{bmatrix} 2 & 1 & 2 & 0 & 0 \\ 0 & 2 & 1 & 2 & 0 \\ 0 & 0 & 2 & 1 & 2 \end{bmatrix}$$

$$[\hat{R}_{XX}(k)] = \frac{1}{3}\begin{bmatrix} 2 & 1 & 2 & 0 & 0 \\ 0 & 2 & 1 & 2 & 0 \\ 0 & 0 & 2 & 1 & 2 \end{bmatrix}\begin{bmatrix} 2 & 0 & 0 \\ 1 & 2 & 0 \\ 2 & 1 & 2 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix} = \frac{1}{3}\begin{bmatrix} 9 & 4 & 4 \\ 4 & 9 & 4 \\ 4 & 4 & 9 \end{bmatrix}$$

Thus, the number of lags, k, ranges from 0 to 2 with autocorrelation estimate values of 3, 4/3, and 4/3, respectively. If we had let $K = 4$, then the value of the autocorrelation estimate for lag 3 would be zero. ■

5.3 ESTIMATES OF THE POWER SPECTRUM

5.3.1 Classical Estimation

Many problems in electrical engineering require a knowledge of the distribution of the power of a random process in the frequency domain, for example, the design of filters to remove noise, to cancel signal echoes, or to represent features of a signal for pattern recognition. If the data records are long, then reliable power spectrum estimates can be obtained using standard fast Fourier transform (FFT) techniques. However, if the data records are short, as they usually are, then the task of obtaining reliable, that is, small bias and small variance, power spectrum estimates becomes difficult. Spectral estimators are usually classified as parametric and nonparametric. The latter generally make no assumptions about the statistics of the data other than that it is wide sense stationary or, perhaps, ergodic. The parametric spectral estimators typically use models of the data. In this case, the data may be modeled as a moving average process or as an autoregressive (linear prediction) process. The parametric approach to spectral estimation typically results in estimates with a smaller bias and variance for a given data record length than nonparametric methods. There is no one "best" spectral estimator at this time. One reason for this is that the spectral estimate is data dependent. Therefore, one estimator is good for one type of data, but not for another type of data. If we know *a priori* the type of data we are to analyze, then the choice of the type of spectral estimator to be used can be narrowed. However, we usually do not have such knowledge. Consequently, we often use a "bouquet" of spectral estimators and use our experience to make a judgment as to which estimator is the most accurate for the given data.

The power spectrum is usually defined using expectation, that is, the expected value of the magnitude squared of the Fourier transform of the data, or another definition, is the Fourier transform of the autocorrelation function determined by expectation. In practical situations, we usually have limited knowledge of the data as well as limited data records. So we need estimates of the autocorrelation function, as defined previously.

Perhaps the oldest and most commonly used spectral estimator is the periodogram, which is one of the classical spectral estimators.

Definition 5.3. The **periodogram** is defined as

$$\hat{S}_{XX}(f) = \frac{1}{N}|X_N(f)|^2 = \frac{1}{N}\left|\sum_{n=0}^{N-1} X(n)e^{-j2\pi fn}\right|^2 \tag{5.3.1.1}$$

where X(n) is a random process.

The periodogram is related to the transform of the biased estimate of the autocorrelation function as follows. Suppose we window the data sequence so that

$$X_N(n) = w(n)X(n)$$

where w(n) is the window sequence and is zero for $n < 0$ and $n \geq N$. Then

$$X_N(z) = \sum_{n=-\infty}^{\infty} X_N(n)z^{-n} = \sum_{n=0}^{N-1} w(n)X(n)z^{-n}$$

However, we have

$$\hat{R}_{XX}(k) = \frac{1}{N}\sum_{n=0}^{N-1-|k|} X_N(n+|k|)X_N(n) = \frac{1}{N}[X_N(k) * X_N(-k)]$$

This estimate is zero for $|n| \geq N$ due to the window. The spectral estimate is

$$\hat{S}_{XX}(f) = \sum_{k=-(N-1)}^{N-1} \hat{R}_{XX}(k)z^{-k}|_{z=e^{-j2\pi f}} = \sum_{k=-(N-1)}^{N-1} \left[\frac{1}{N}[X_N(k) * X_N(-k)]\right] z^{-k}|_{z=e^{-j2\pi f}}$$
$$= \frac{1}{N}X_N(z)X_N(z^{-1})|_{z=e^{-j2\pi f}} = \frac{1}{N}|X_N(f)|^2$$

While the periodogram seems to be a reasonable spectral estimator, it has some faults. First, the estimator is biased. If we denote the ensemble average autocorrelation function of the window as $R_{ww}(k)$ and the spectrum of the window as the Fourier transform of this autocorrelation function as $S_{ww}(f)$, then the expected value of the periodogram turns out to be the true spectrum of the data convolved with the power spectrum of the window. Thus, if the window is short, then the bias of the periodogram is large. As the window length in the time domain increases, the bias decreases and approaches the true power spectrum of the data. Furthermore, the variance of the periodogram does not decrease as N increases. This is because no averaging of the estimate is done either by using the expectation operator or by time averaging. So the periodogram is not a good spectral estimator, even for long data records. Nevertheless, it is commonly used.

Several proposals have been made to improve the power spectral estimate of the periodogram. One such proposal suggests that the data record be segmented into M independent data records each with L points, as shown in Figure 5.5. These data records can overlap if desired, as shown. The periodogram is then calculated for each segmented data record, and the periodogram for the entire data record is the average of the periodograms for each segment. However, the bias of this spectral estimate is larger than the periodogram for the original data record, although the variance of the estimate is reduced by the factor M over that of the periodogram of a single L point segment. Note that this approach has fewer points in the spectral estimate due to the segmentation of the data. Consequently, the spacing between spectral line components is increased, thereby, decreasing the spectral resolution of the estimate.

Since the variance of the spectral estimate of the segmental averaging approach is reduced, this implies that "smoothing" the periodogram of the original (unsegmented) data record may give an improved spectral estimate. Thus, another approach is to segment the autocorrelation estimate and average the segmented autocorrelation estimates, as shown in Figure 5.6. This approach yields a smoothed spectral estimate of the periodogram in that the periodogram is convolved with the power spectrum of a triangular window. However,

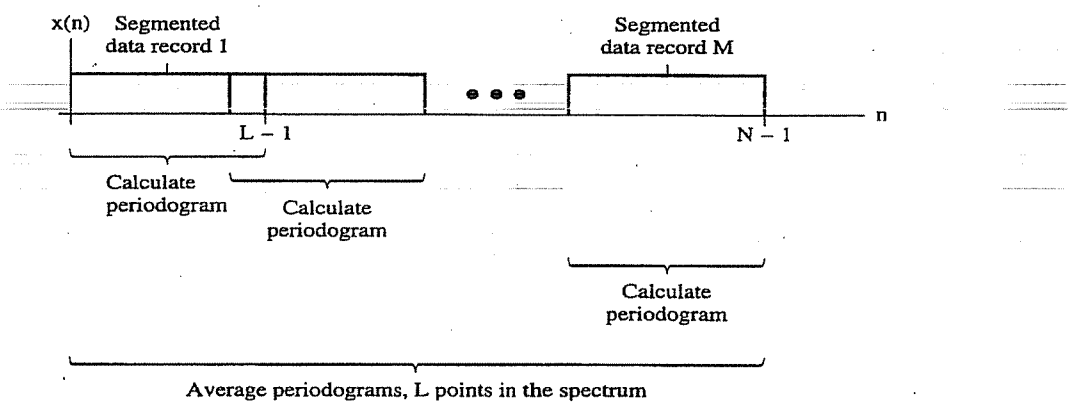


FIGURE 5.5 Illustration of segmenting the data record to improve the statistical stability of the power spectral estimate.

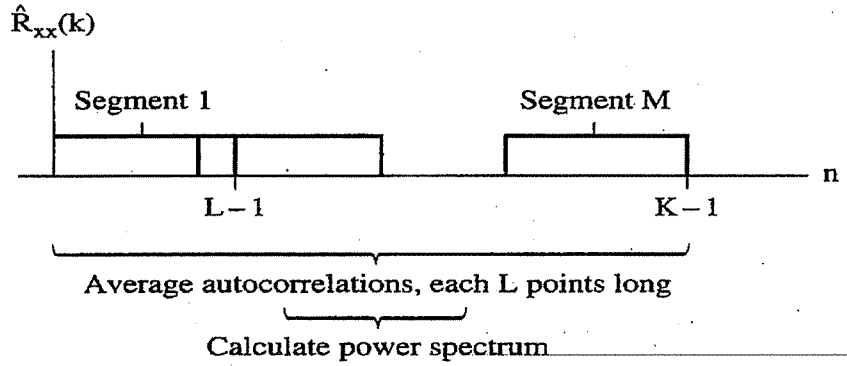


FIGURE 5.6 Illustration of segmenting the autocorrelation function to improve the statistical stability of the power spectral estimate.

since K is 10% of N, and L is less than K, we have even fewer points in the spectral estimate than averaging the periodograms. But, we again achieve improved statistical stability (a reduction in the variance of the estimate), at the cost of reduced spectral resolution.

Appending M zeros to extend the data record from length N to N + M does not result in increased resolution in the spectral estimate. The spectral lines in the transform are closer, thus assisting the interpretation of the spectral envelope. But no new information is added by appending zeros. This process is basically one of interpolation.

Windowing the data record will reduce the sidelobes of the periodogram. However, the bandwidth of the main lobe is increased, which reduces the spectral resolution of the estimate. There are a number of data windows available in the MATLAB Signal Processing Toolbox. The effect of these windows on speech data is easily explored using the software provided in Chapter 2. If the autocorrelation function estimate is windowed before transforming, then the spectral estimate may have negative values. Thus, the transform of the window should be nonnegative.

Another classical spectral estimator is the Blackman–Tukey estimate.

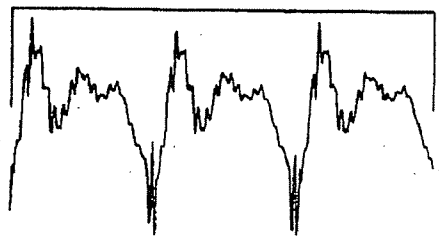
Definition 5.4. The **Blackman–Tukey spectral estimate** is

$$\hat{S}_{XX}(f) = \sum_{k=-M}^{M} w(k)\hat{R}_{XX}(k)e^{-j2\pi fk} \tag{5.3.1.2}$$

where $N - 1 \geq M$, and w(k) is a window applied to the autocorrelation estimate, which is zero for $|k| > M$. The value of M is usually restricted so that $N/10 \geq M$. The window is usually normalized such that $w(0) = 1$ and $1 \geq w(k) \geq 0$ for $k \neq 0$. The windows in MATLAB as well as others can be used. Typically, the biased autocorrelation estimate is used. The Blackman–Tukey spectral estimate is biased, and for certain conditions is asymptotically consistent, since the variance of the estimate does decrease as N increases.

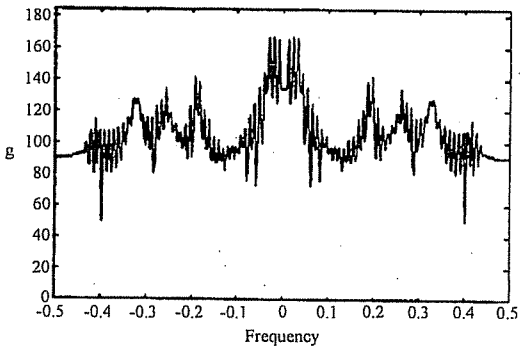
EXAMPLE 5.3

In this example, we compare the periodogram and the Blackman–Tukey spectral estimates for a speech data record of the sustained vowel /IY/ as in see. The data record is shown in Figure 5.7 along with the spectral estimates. All spectra are plotted on the same dB scale. The frequency axis is scaled such that 0.5 is 5000 Hz. The windowed data contain approximately three pitch periods of data, representing about 25 msec of data sampled at 10 kHz. For the periodogram, we calculate the spectrum using a rectangular window and a hamming window

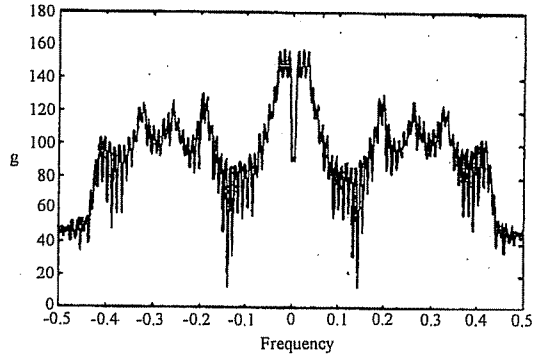


(a) Speech signal with rectangular window.

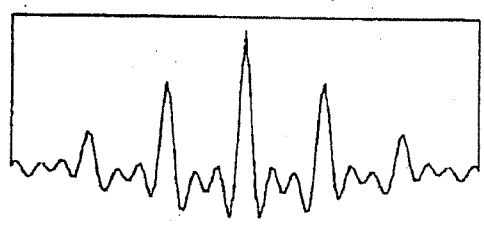
(b) Speech signal with hamming window.



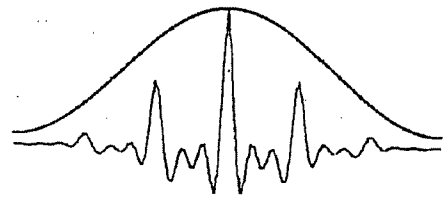
(c) Periodogram of speech signal with rectangular window.



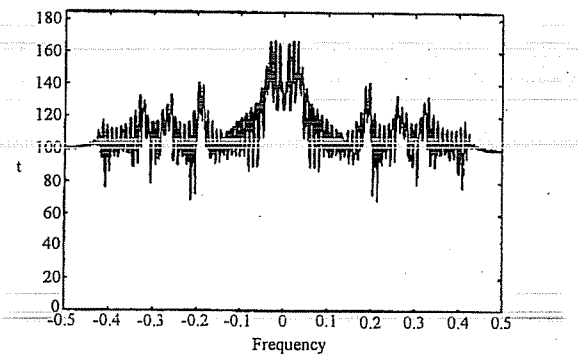
(d) Periodogram of speech signal with hamming window.



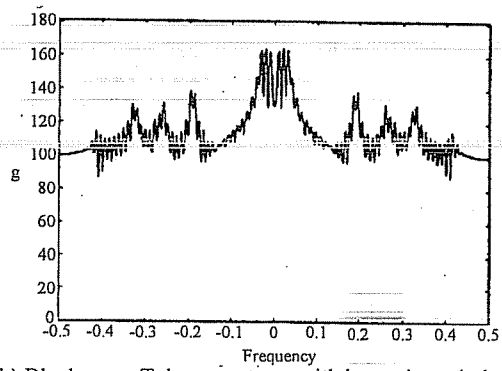
(e) Autocorrelation function with rectangular window.



(f) Autocorrelation function with hamming window.



(g) Blackman–Tukey spectrum with rectangular window.



(h) Blackman–Tukey spectrum with hamming window.

FIGURE 5.7 An example of power spectra for the vowel /IY/.

applied to the data record. For the Blackman–Tukey spectral estimate, we first calculate the biased autocorrelation estimate and then apply the rectangular and hamming windows to the autocorrelation function prior to calculating the power spectrum. The hamming window tends to enhance the peaks and the nulls of the periodogram over that for the rectangular window. The power spectrum of the rectangular windowed autocorrelation function is similar to that for the corresponding periodogram, as one might expect. The Blackman–Tukey power spectrum with the hamming window is the "smoothest" of all four, with perhaps the best defined peaks. We shall return to this example again after we discuss the parametric spectrum methods, at which point, it will be seen that the methods shown here do not provide good estimates of the spectral peaks (formants) around 300, 2200, and 3000 Hz. This example can be done using the software supplied with this book in Chapter 2. ■

5.3.2 Parametric Spectral Estimation

The parametric spectral estimation procedures are relatively new, having appeared within the last 30 years. Their attraction is that they obtain good spectral estimates for short data records. The methods tend to be data adaptive in that they adjust themselves to be least disturbed by power at frequencies other than the one being estimated.

One parametric spectral estimator is called the maximum entropy estimator. The idea for this estimator is that instead of appending zeros to increase the length of the estimated autocorrelation function estimate, the estimated autocorrelation function is to be extrapolated (predicted) beyond the data limited range. The principle used for this extrapolation process is that the spectral estimate must be the most random or have the maximum entropy of any power spectrum that is consistent with the sample values of the calculated autocorrelation function. The objective is to add no information as a result of the prediction process, but yet make an improvement over that obtained by just appending zeros. This spectral estimate is equivalent to linear prediction or autoregression and is the same as the maximum likelihood method for Gaussian data.

Perhaps the simplest way to introduce the parametric spectral estimation procedure is with data models. One way to generate a data model is to excite a filter and measure the filter output. For example, Figure 5.8 shows a filter with both poles and zeros [an autoregressive-moving average (ARMA) model] such that the output is given as

$$X(n) = -\sum_{i=1}^{p} a_i X(n-i) + \sum_{i=0}^{q} b_i E(n-i) \qquad (5.3.2.1)$$

Note that we still denote a random process with upper case notation and a particular member function of the random process with lower case notation. Thus, Equation (5.3.2.1) denotes the difference equation for an ARMA random process data model, which we can also write as

$$X(z) = -\sum_{i=1}^{p} a_i z^{-i} X(z) + \sum_{i=0}^{q} b_i z^{-i} E(z) \qquad (5.3.2.2)$$

The minus sign in front of the a_i coefficients is one convention that is also used in MATLAB. The input E(n) to the filter is the driving function, which is typically white noise, while the output X(n) is the random process to be modeled. This model is called an

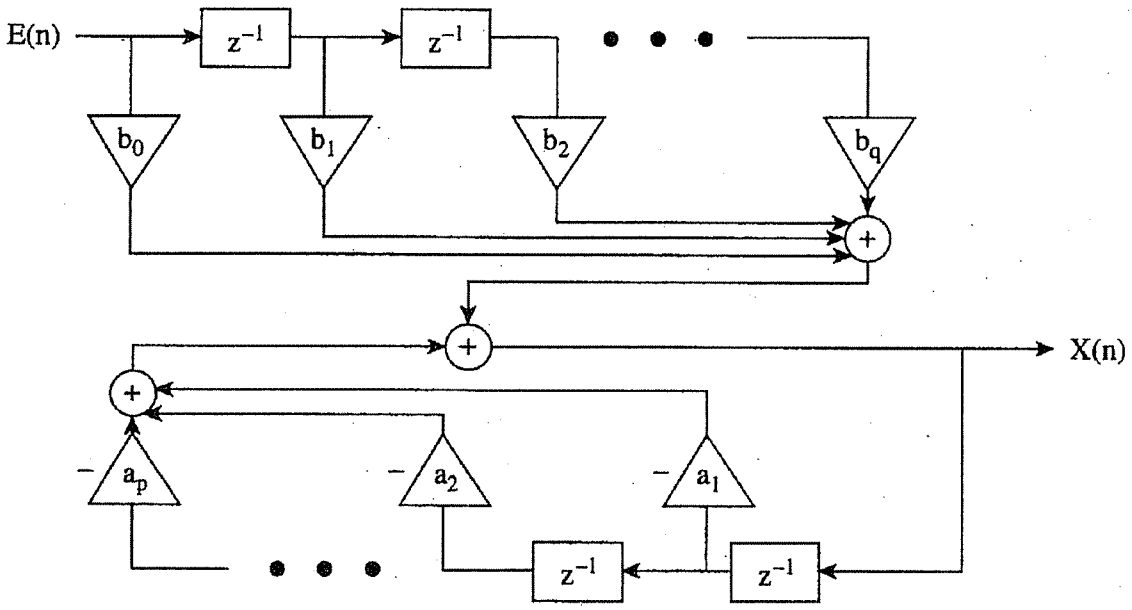


FIGURE 5.8 An ARMA filter model.

ARMA(p, q) process. The transfer function for the ARMA filter is

$$H(z) = \frac{X(z)}{E(z)} = \frac{\sum_{i=0}^{q} b_i z^{-i}}{\sum_{i=0}^{p} a_i z^{-i}} = \frac{B(z)}{A(z)} \tag{5.3.2.3}$$

where B(z) represents the moving average (MA) branch of the filter, while A(z) represents the autoregressive (AR) branch, which is also called the linear prediction (LP) branch. Note that there are q zeros in the MA branch of the filter, and p poles in the AR branch.

Since

$$X(z) = H(z)E(z) \tag{5.3.2.4}$$

The power spectrum is expressed as

$$S_{XX}(f) = S_{EE}(f)|H(f)|^2 \tag{5.3.2.5}$$

This is not an estimate, since this last step can be done using ensemble averaging. However, we will soon address the estimation problem.

If all of the $a_i = 0$, except for the unity coefficient in A(z), which is usually called $a_0 = 1$, then we have an all zero, MA, model of order q, that is

$$X(n) = \sum_{i=0}^{q} b_i E(n-i) \tag{5.3.2.6}$$

which is shown in Figure 5.9.

The MA(q) power spectrum for a white noise excitation with variance σ_{EE}^2 is

$$S_{MA}(f) = S_{XX}(f) = \sigma_{EE}^2 |B(f)|^2 \tag{5.3.2.7}$$

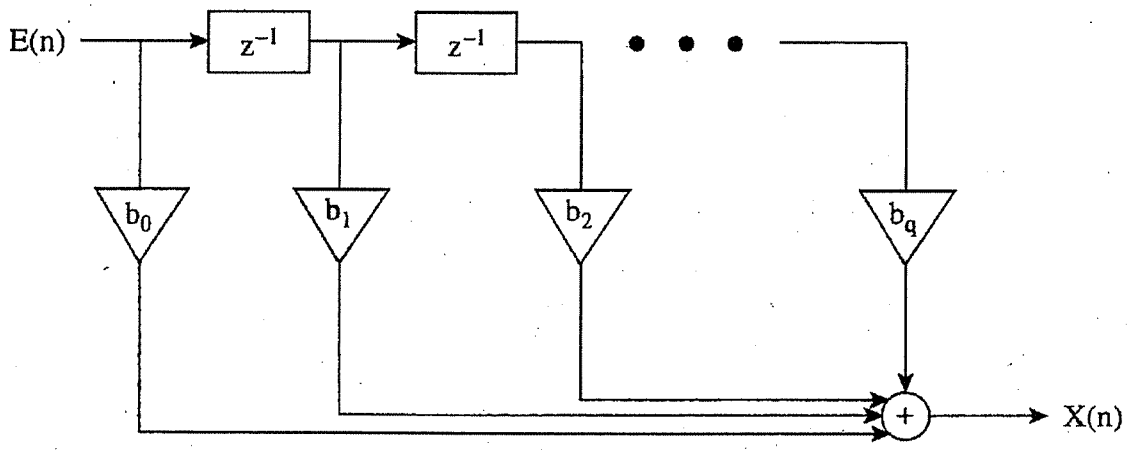


FIGURE 5.9 A MA(q) filter model.

If all the $b_i = 0$ except for $b_0 = 1$, then

$$X(n) = -\sum_{i=1}^{p} a_i X(n-i) + E(n) \tag{5.3.2.8}$$

which is an all pole, AR(p), process, also called a LP(p) process, shown in Figure 5.10, and the power spectrum is

$$S_{AR}(f) = S_{XX}(f) = \frac{\sigma_{EE}^2}{|A(f)|^2} \tag{5.3.2.9}$$

The estimation process requires a procedure for estimating the a_i and b_i coefficients given a data record. There are several theorems that tell us more about these data models. One theorem is called the Wold decomposition theorem, which says that a wide sense stationary random process can be decomposed into a component that is random and one that is deterministic. Another theorem says that any ARMA or AR process can be represented by a unique MA process of infinite order, i.e., MA(∞). The third theorem says that any ARMA or MA process can be represented by an AR(∞) process. These theorems tell us that if we should select the wrong model to represent the data, we can still obtain an adequate approximation, although not an "optimum" approximation, if we use a high enough model

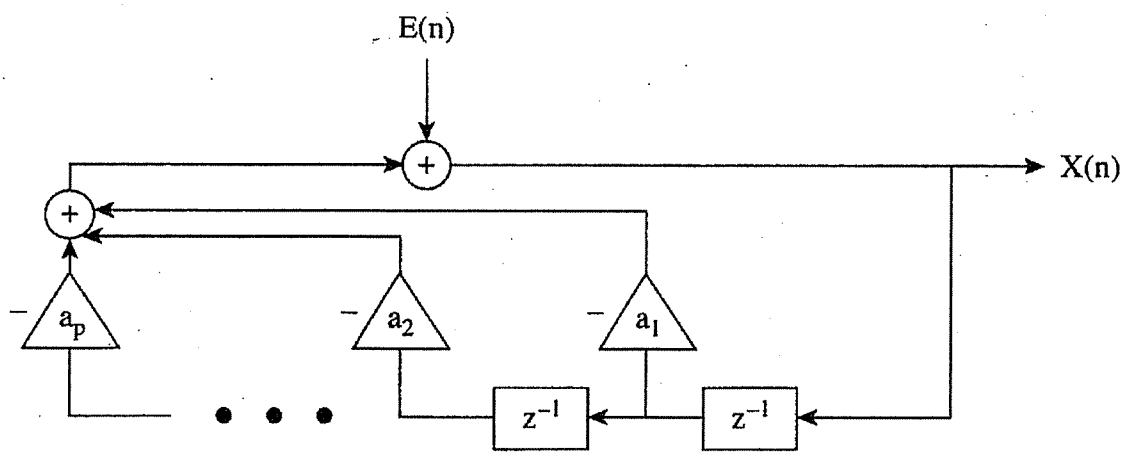


FIGURE 5.10 An AR(p) filter model.

order. In practical terms this is not satisfactory, since if we have to improve the model by increasing its order toward infinity, then we also have to consider a larger data set from which to estimate reliably the parameters of the model. And one purpose of data modeling for spectral analysis is to obtain a good spectral estimate with as few model parameters as possible. So from a practical point of view, it is not a good idea to think that if the result from a low-order model is not satisfactory in some sense, that the model order can be increased until a "good" result is obtained. A better idea is to try another data model.

EXAMPLE 5.4

Consider a first order AR model, such that

$$X(n) = -a_1 X(n-1) + E(n)$$

Then

$$X(z) = -a_1 z^{-1} X(z) + E(z)$$

which can be expressed as

$$X(z) = \frac{E(z)}{1 + a_1 z^{-1}} = z\frac{E(z)}{z + a_1}$$

where $|a_1| < 1$. This process has a simple pole at $z = -a_1$, and depending on the value of a_1, the process will be lowpass or highpass, as shown in Figure 5.11. The process cannot be bandpass unless the process has more than one pole. The power spectrum for a white noise input with variance σ^2_{EE} is given below and is sketched in Figure 5.11, where the sampling frequency is f_s. Note that we use f_s and F_s interchangeably.

$$S_{XX}(f) = \frac{\sigma^2_{EE}}{|1 + a_1 e^{-j2\pi f}|^2}$$

■

Further discussion of linear prediction (LP) and linear prediction spectral estimation follows. We also discuss ARMA and MA spectral estimates that are used in the software provided with this book. The reader can also consult Appendix 6 as well as Kay (1988).

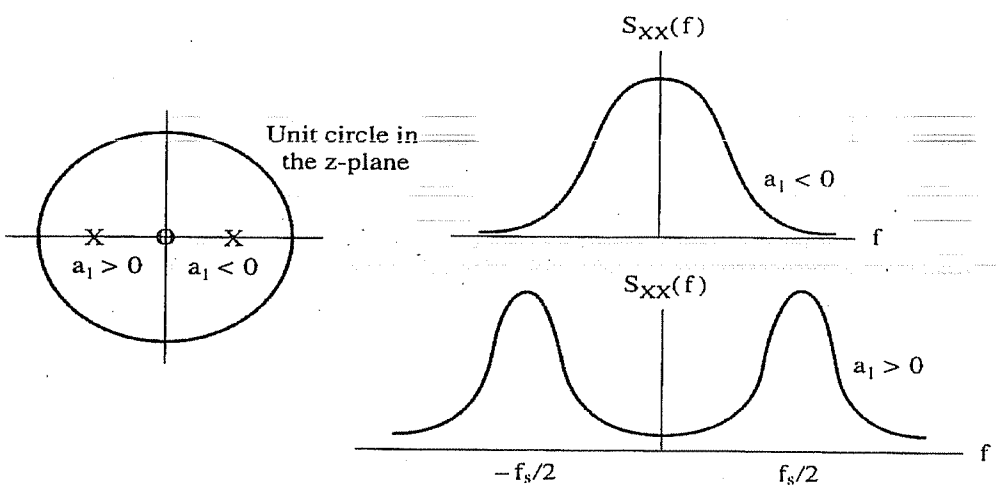


FIGURE 5.11 The power spectral density of an AR(1) model.

5.3.3 Least Squares Estimation

The method we will use in this chapter for estimating the autoregression (AR) parameters is the method of least squares. We will focus on AR estimation only. The method of least squares estimation may be introduced with the following simple situation. We conduct an experiment by measuring specific data values of a function, x(n). We wish to design a filter that will estimate the data at time n using one previous data sample at time n − 1, i.e.,

$$\hat{x}(n) = -\hat{a}x(n-1) \tag{5.3.3.1}$$

where $\hat{x}(n)$ and x(n) are real discrete sequences, and $\hat{a}$ is a parameter to be estimated. The term $\hat{x}(n)$ is an estimate of the true value x(n). Consequently, we have an error, defined as

$$e(n) = x(n) - \hat{x}(n) = x(n) + \hat{a}x(n-1) \tag{5.3.3.2}$$

Note that this model is predicting the true value of the data at time n using a weighted value of the data at time n − 1. This model is called linear prediction (LP) or autoregression (AR). The method of least squares says that the constant $\hat{a}$ can be chosen so that we can minimize the sum of the errors over some interval from n = 0 to n = N − 1 (or 1 to N), i.e., we minimize the total squared error.

$$\begin{aligned} E_T^2 &= \sum_{n=0}^{N-1} e^2(n) = \sum_{n=0}^{N-1} (x(n) - \hat{x}(n))^2 \\ &= \sum_{n=0}^{N-1} (x(n) + \hat{a}x(n-1))^2 \end{aligned} \tag{5.3.3.3}$$

The total squared error is sometimes normalized by the number of data points, N. However, this has no effect on the solution, as we show below. The total squared error is often called the prediction error or the prediction error power.

A word about notation is appropriate here. The method of least squares assumes that we are dealing with data and does not use statistics. Consequently, the lower case notation is used for functions. It is possible to consider X(n) to be a random process and solve for the parameter $\hat{a}$ using expectation. This procedure minimizes the mean square error and gives the true value (not an estimate) for the parameter.

A necessary condition for a relative minimum is that the partial derivative with respect to $\hat{a}$ is zero. Then

$$\frac{\partial E_T^2}{\partial \hat{a}} = 0 = 2\sum_{n=0}^{N-1} (x(n) + \hat{a}x(n-1))x(n-1) \tag{5.3.3.4}$$

which gives

$$\hat{a}\sum_{n=0}^{N-1} x(n-1)x(n-1) = -\sum_{n=0}^{N-1} x(n)x(n-1) \tag{5.3.3.5}$$

Note that if there had been a multiplicative factor of $\frac{1}{N}$ in (5.3.3.3), then this would have no effect on the solution to (5.3.3.4). Equation (5.3.3.5) can be expressed in terms of autocorrelation estimates as follows.

$$\hat{a} = \frac{-\frac{1}{N}\sum_{n=0}^{N-1} x(n)x(n-1)}{\frac{1}{N}\sum_{n=0}^{N-1} x(n-1)x(n-1)} = -\frac{\hat{R}_{XX}(1)}{\hat{R}_{XX}(0)} \tag{5.3.3.6}$$

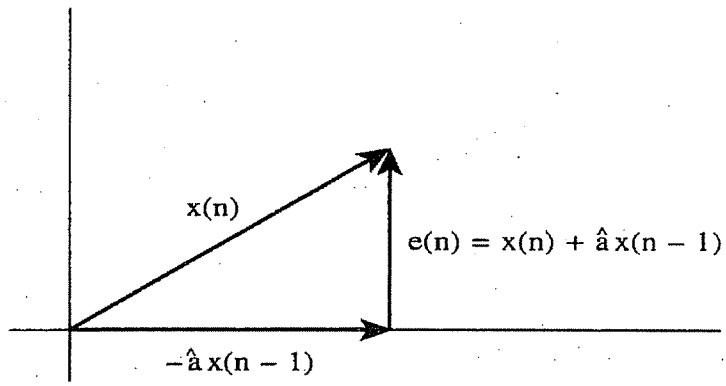


FIGURE 5.12 The principal of orthogonality. The data is orthogonal to the error.

where we will continue to use $\hat{R}_{XX}(k)$ to denote an estimate of the autocorrelation function. Equation (5.3.3.6) gives the least squares estimate for the parameter â. This estimate gives the best linear prediction of x(n) using only one weighted past data sample. The solution is best in the sense that the total error, that is, the sum of the squared error terms, is minimized.

We will soon generalize the above results for p $\hat{a}_i$ coefficients. However, first we can observe that (5.3.3.4) provides us with the result that the error is orthogonal to the data in a least squares sense, that is

$$\sum_{n=0}^{N-1}(x(n) + \hat{a}x(n-1))x(n-1) = \sum_{n=0}^{N-1} e(n)x(n-1) = 0 \qquad (5.3.3.7)$$

This **principle of orthogonality** is illustrated in Figure 5.12.

Next we note that Equation (5.3.3.5) is called the **normal equation**, which becomes multiple equations when we consider p $\hat{a}_i$ coefficients. The method of least squares makes no assumption about the statistics of the random sequence. However, Gauss proved a theorem, named after him years ago, that says that if the errors are uncorrelated with zero mean and the same variance, then the optimum estimate for the coefficient is the value that minimizes the square of the error between the observed data and the true value. The estimate is optimum in the sense that any linear function can be estimated with the minimum mean square error using the estimator.

We now form a more general linear prediction model using p$\hat{a}_i$ coefficients, and p past values of the data to predict the value of the data at time n. The data are given, and may or may not be from an AR process. This knowledge is often unknown. Nevertheless, we are modeling the data as if it were an AR data model. Therefore,

$$\hat{x}(n) = -\sum_{k=1}^{p} \hat{a}_k x(n-k) \qquad (5.3.3.8)$$

Then, we have

$$x(n) = -\sum_{k=1}^{p} \hat{a}_k x(n-k) + e(n) = \hat{x}(n) + e(n) \qquad (5.3.3.9)$$

which we illustrate in Figure 5.13.

The z-transform of Equation (5.3.3.9) is

$$X(z) = -\left[\sum_{k=1}^{p} \hat{a}_k z^{-k}\right] X(z) + E(z) = \hat{X}(z) + E(z) \qquad (5.3.3.10)$$

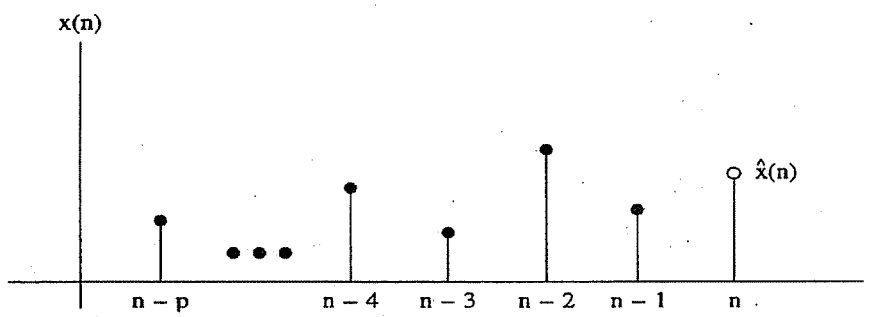


FIGURE 5.13 Linear prediction model.

which can be expressed as

$$X(z) = \frac{E(z)}{1 + \left[\sum_{k=1}^{p} \hat{a}_k z^{-k}\right]} = \frac{E(z)}{A(z)} \tag{5.3.3.11}$$

where

$$A(z) = 1 + \sum_{k=1}^{p} \hat{a}_k z^{-k} \tag{5.3.3.12}$$

Equation (5.3.3.11) is implemented as a filter in Figure 5.14. This form is often called the forward filter model, since E(z), the input excitation, is filtered by $\frac{1}{A(z)}$ to produce X(z), the LP data.

We can also write Equation (5.3.3.10) in the following form.

$$E(z) = \left[1 + \sum_{k=1}^{p} \hat{a}_k z^{-k}\right] X(z) = A(z)X(z) \tag{5.3.3.13}$$

This is the form used in Figure 5.15, where the data, X(z), is "inverse" filtered by A(z) to yield the excitation, E(z). Consequently, this form is called the inverse filter model.

The method of least squares determines the coefficients $\hat{a}_i$ such that the sum of the errors over some interval is minimized. The interval over which we observe the data is what distinguishes the two methods that are used to solve for the $\hat{a}_i$ coefficients. For the **autocorrelation** method, it is assumed that the data are windowed such that the data are

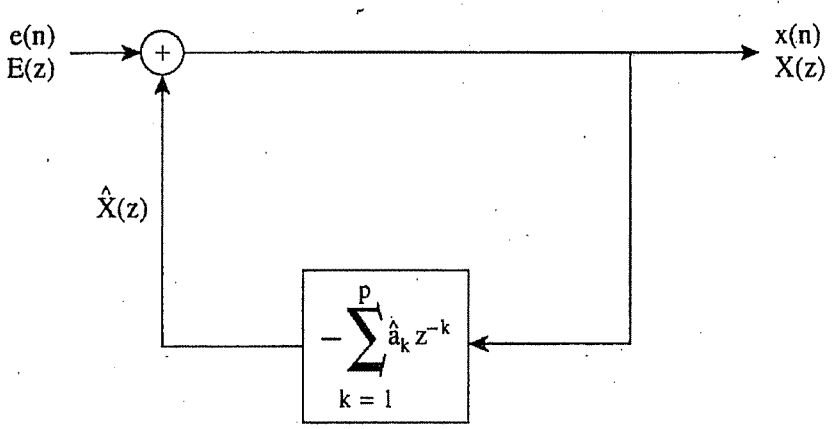


FIGURE 5.14 The forward filter model.

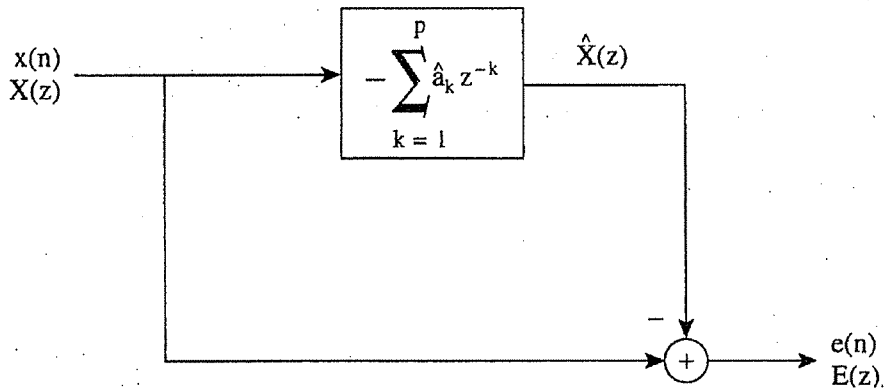


FIGURE 5.15 The inverse filter model.

zero outside the window. Note that in this case the error is likely to be large at both the beginning and ending of the estimation interval, since we are trying to predict the true data using zeros because we have windowed the data. This is why the data window is typically tapered at both ends. This causes the data to gradually increase at the beginning of the data record and similarly gradually decrease at the end, thereby, reducing the error for these segments. The total squared error (total prediction error, or total prediction power) over this interval is

$$E_T^2 = \sum_{n=0}^{N-1} e^2(n) = \sum_{n=0}^{N-1} (x(n) - \hat{x}(n))^2 = \sum_{n=0}^{N-1} \left(x(n) + \sum_{k=1}^{p} \hat{a}_k x(n-k) \right)^2 \qquad (5.3.3.14)$$

where x(n) is the windowed data. The minimization is done by

$$\frac{\partial E_T^2}{\partial \hat{a}_i} = 0, \quad i = 1, 2, \ldots, p$$

which gives

$$\sum_{k=1}^{p} \hat{a}_k \sum_{n=0}^{N-1} x(n-k)x(n-i) = -\sum_{n=0}^{N-1} x(n)x(n-i), \quad \text{for } i = 1, 2, \ldots, p \qquad (5.3.3.15)$$

When these equations are derived using expectation, they are known as the **normal equations**, and are also sometimes referred to as the **Yule–Walker equations** and the **Wiener–Hopf equations**. We will refer to them by the same names. The equations can be expressed in a more compact form using the autocorrelation estimates as follows.

$$\sum_{k=1}^{p} \hat{a}_k \hat{R}_{XX}(i-k) = -\hat{R}_{XX}(i), \quad i = 1, 2, \ldots, p \qquad (5.3.3.16)$$

where

$$\hat{R}_{XX}(k) = \frac{1}{N} \sum_{n=0}^{N-1-|k|} x(n)x(n+|k|) \qquad (5.3.3.17)$$

Note that the $\frac{1}{N}$ term is not needed as we mentioned previously. This is only a scale factor and does not affect the solution of the equations for the $\hat{a}_i$. Equation (5.3.3.16) is a set of p

equations with p unknowns and can be solved using matrix techniques as follows:

$$\begin{bmatrix} \hat{R}_{XX}(0) & \hat{R}_{XX}(-1) & \dots & \hat{R}_{XX}(-(p-1)) \\ \hat{R}_{XX}(1) & \hat{R}_{XX}(0) & \dots & \hat{R}_{XX}(-(p-2)) \\ \vdots & \vdots & \dots & \vdots \\ \hat{R}_{XX}(p-1) & \hat{R}_{XX}(p-2) & \dots & \hat{R}_{XX}(0) \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_p \end{bmatrix} = - \begin{bmatrix} \hat{R}_{XX}(1) \\ \hat{R}_{XX}(2) \\ \vdots \\ \hat{R}_{XX}(p) \end{bmatrix} \tag{5.3.3.18}$$

These equations can be solved by matrix inversion, but this is not as efficient as the Levinson algorithm, which solves the equations for the $\hat{a}_i$ in a recursive manner. Algorithms for doing this are available in MATLAB and take advantage of the fact that the autocorrelation matrix is Toeplitz. MATLAB has a function called lpc that returns the $\hat{a}_i$ coefficients given the data. Note, that since the autocorrelation function is real and even, then the terms above the main diagonal in the correlation matrix in Equation (5.3.3.18) can be replaced with terms with a positive argument.

The autocorrelation method is guaranteed to give a stable LP filter, that is, one such that the poles of $\frac{1}{A(z)}$ are all within the unit circle in the z-plane. Since we are assuming real data, the $\hat{a}_i$ coefficients are real, implying that the poles of $\frac{1}{A(z)}$ appear in complex conjugate pairs. Thus, for example, if p is even, then we will have an even number of coefficients and an even number of poles, with one-half of the poles being in the upper half of the unit circle in the z-plane and one-half of the poles being in the lower half. Of course, one or more pairs of poles could fall on the real line in the z-plane. If p is odd, then at least one root will be on the real line.

An additional equation is often added to the above equations for the $\hat{a}_i$, namely, the equation for the total squared error (prediction error), which from Equations (5.3.3.13), (5.3.3.15), and (5.3.3.17) is

$$E_T^2 = \hat{R}_{XX}(0)a_0 + \sum_{k=1}^{p} \hat{a}_k \hat{R}_{XX}(-k) \tag{5.3.3.19}$$

where $a_0 = 1$. Again the estimates with negative arguments in Equation (5.3.3.19) can be replaced with the corresponding estimates with positive arguments. This error will vary with the order p. The reason for introducing a_0 is that this is a convenience when the matrix equations are augmented for the additional unknown, E_T^2, as follows.

$$\begin{bmatrix} \hat{R}_{XX}(0) & \hat{R}_{XX}(-1) & \dots & \hat{R}_{XX}(-p) \\ \hat{R}_{XX}(1) & \hat{R}_{XX}(0) & \dots & \hat{R}_{XX}(-(p-1)) \\ \vdots & \vdots & \dots & \vdots \\ \hat{R}_{XX}(p) & \hat{R}_{XX}(p-1) & \dots & \hat{R}_{XX}(0) \end{bmatrix} \begin{bmatrix} a_0 \\ \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_p \end{bmatrix} = \begin{bmatrix} E_T^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5.3.3.20}$$

These equations are also often called the **extended** or **augmented normal equations**.

The **covariance method** for solving for the coefficients is similar to the autocorrelation method. The major difference is that the covariance method does not assume that the data are windowed, and that while N samples of the data are available, the error is windowed such that N − p samples of the error are available. This affects the calculation of the "autocorrelation" values, since we now have

$$E_T^2 = \sum_{n=p}^{N-1} e^2(n) = \sum_{n=p}^{N-1} (x(n) - \hat{x}(n))^2 = \sum_{n=p}^{N-1} \left(x(n) + \sum_{k=1}^{p} \hat{a}_k x(n-k) \right)^2 \tag{5.3.3.21}$$

The normal equations have the same form, namely

$$\sum_{k=1}^{p} \hat{a}_k \hat{R}_{XX}(i-k) = -\hat{R}_{XX}(i), \quad i = 1, 2, \ldots, p \tag{5.3.3.22}$$

However, the autocorrelation estimates are different, so a different notation is used to distinguish the two methods, that is,

$$\hat{C}_{XX}(i,k) = \frac{1}{N-p} \sum_{n=p}^{N-1} x(n-i)x(n-k) \tag{5.3.3.23}$$

and

$$\hat{C}_{XX}(i,0) = \frac{1}{N-p} \sum_{n=p}^{N-1} x(n-i)x(n) \tag{5.3.3.24}$$

where $1 \leq i \leq p$, $0 \leq k \leq p$, and the data record is N long. No window is applied to the data because we assume we have sufficient data to calculate the desired correlation values. The matrix equations have the same form, namely

$$\begin{bmatrix} \hat{C}_{XX}(1,1) & \hat{C}_{XX}(1,2) & \ldots & \hat{C}_{XX}(1,p) \\ \hat{C}_{XX}(2,1) & \hat{C}_{XX}(2,2) & \ldots & \hat{C}_{XX}(2,p) \\ \vdots & \vdots & \ldots & \vdots \\ \hat{C}_{XX}(p,1) & \hat{C}_{XX}(p,2) & \ldots & \hat{C}_{XX}(p,p) \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_p \end{bmatrix} = - \begin{bmatrix} \hat{C}_{XX}(1,0) \\ \hat{C}_{XX}(2,0) \\ \vdots \\ \hat{C}_{XX}(p,0) \end{bmatrix} \tag{5.3.3.25}$$

The correlation matrix is no longer Toeplitz, but has the properties of a covariance matrix, from which the name for this method is derived. The solution for this method does not use the Levinson algorithm, but instead uses Cholesky decomposition (or other methods), which are also available in MATLAB.

The covariance method may yield estimates of the $\hat{a}_i$ coefficients such that the LP filter, $\frac{1}{A(z)}$, does not have all of its poles inside the unit circle, therefore, the filter may be unstable.

Return to Equation (5.3.3.20) and note that all of the equations are prediction equations for the autocorrelation function. In particular, the last equation in the set of equations represented by (5.3.3.20) can be written as

$$\hat{R}_{XX}(p) + \hat{a}_1 \hat{R}_{XX}(p-1) + \cdots + \hat{a}_p \hat{R}_{XX}(0) = 0 \tag{5.3.3.26}$$

This equation can be considered a prediction equation for $\hat{R}_{XX}(p)$ using the past values of the autocorrelation function and the prediction coefficients. It can be shown that for lag values greater than p, that

$$\hat{R}_{XX}(k) + \hat{a}_1 \hat{R}_{XX}(k-1) + \cdots + \hat{a}_p \hat{R}_{XX}(k-p) = 0 \tag{5.3.3.27}$$

for $k = p+1, p+2, \ldots$. Therefore, the linear prediction equations lead to a correlation prediction equation, whereby the correlation function can be extended. This is the basis for the maximum entropy spectral estimator mentioned earlier.

5.3.4 Power Spectral Estimation With Linear Prediction

Linear prediction is a data model. Consequently, it is frequently used to estimate the spectrum of data that are modeled using linear prediction techniques.

Definition 5.5. The **LP power spectral estimate** is the reciprocal of the square of the absolute value of the transform of the LP coefficients.

$$\hat{S}_{XX}(f) = \frac{1}{|A(z)|^2} = \frac{1}{\left|1 + \sum_{i=1}^{p} \hat{a}_i z^{-i}\right|^2}, \quad \text{for } z = e^{j2\pi f} \tag{5.3.4.1}$$

where the $\hat{a}_i$ coefficients are the estimates obtained using the least squares method above, and the number of coefficients, p, is the same as the number of poles of $\frac{1}{A(z)}$. Since A(z) is a polynomial in z, the roots of this polynomial are the zeros of A(z), which in turn are the poles of $\frac{1}{A(z)}$.

Why is Equation (5.3.4.1) considered a good spectral estimate? We explain this as follows. The error can be expressed as

$$E(z) = A(z)X(z) \tag{5.3.4.2}$$

where the A(z) in this equation represents the z-transform of the estimated coefficients. Then

$$|E(z)|^2 = |A(z)|^2 |X(z)|^2 \tag{5.3.4.3}$$

and the true power spectrum is

$$S_{XX}(f) = |X(z)|^2, \quad \text{for } z = e^{j2\pi f} \tag{5.3.4.4}$$

The total squared error can be expressed as

$$E_T^2 = \frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} |E(e^{j2\pi f})|^2 \, d\omega = \frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} \frac{S_{XX}(f)}{\hat{S}_{XX}(f)} \, d\omega \tag{5.3.4.5}$$

where T is the sampling interval or $\frac{1}{T}$ is F_s, the sampling frequency. Now note that minimizing the error is equivalent to minimizing the integral of the ratio of the true power spectrum to the estimate. This ratio is positive. Both global and local errors are minimized by minimizing the integral. A local error may be thought of as an error at a particular frequency. Globally, the total error is determined by how well the true spectrum and the estimated spectrum match over the entire frequency range, regardless of the shape of the spectrum. Local errors are a measure of the difference between these two spectra at a given frequency. If the true spectrum is greater than the estimate (model), then the total error is larger than if the opposite is true.

Thus, after minimizing E_T^2, we would expect the estimate to match the true spectrum better in those regions where the true spectrum is greater than the estimate. Thus, the estimate, or model spectrum, should follow the true spectrum in the vicinity of the peaks, or local maxima. However, this is not necessarily the case in regions where minima (notches) occur, since the local error is already small and the emphasis is on reducing the largest errors. Thus, the estimate should be a good estimate of the spectral envelope of the true spectrum. Since it is usually the peaks in the spectrum that we want to estimate, the LP estimate is often used.

Spectral flatness is defined as the spectrum of the prediction error

$$E_f = \frac{\exp\left\{\frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} \log|E(\omega)|^2\, d\omega\right\}}{\frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} |E(\omega)|^2\, d\omega} = \frac{\exp\left\{\frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} \log|E(\omega)|^2\, d\omega\right\}}{E_T^2} \quad (5.3.4.6)$$

where $|E(\omega)|^2$ is the spectrum of the prediction error. The spectral flatness is the ratio of the geometric mean to the arithmetic mean of the spectrum and can assume values between 0 and 1. It is 1 for a constant spectrum. Note that the spectral flatness is inversely proportional to the mean squared prediction error. Thus, minimizing the mean squared prediction error in the time domain is equivalent to finding an all pole filter such that the spectral flatness of the prediction error is maximized. The idea of maximizing the spectral flatness is used in the manual inverse filtering procedures in the software in Chapter 2. Maximizing the spectral flatness of the error in glottal inverse filtering to eliminate (or at least reduce) the formant frequencies reduces the error in the estimates of the linear prediction filter coefficients. This procedure is difficult to achieve automatically, so it is implemented with user manual interaction.

One problem with the LP method is the estimation of the number of coefficients, p, to be used in the model. While there are criteria for this, it is still debated which is the best. One such criterion is the Akaike information criterion (AIC), which is defined as $AIC(p) = (N)\ln(E_T^2) + 2p$, where p is the model order, N is the number of data points available, ln is the natural log, E_T^2 is the total squared error for the pth order model given in Equation (5.3.3.19). The value of AIC can be plotted versus p to determine the "best" model order. The AIC procedure is implemented in the software for this book (see Chapter 2 and Appendix 6). A trial and error procedure is also often used, whereby, the choice of p is determined by the value that gives the smallest error for reasonable variations in p. This is done by varying p and calculating the error for each value of p. Then, we select the value of p that gives the least error. Another problem with the LP method is that it does not give good estimates of sinusoids. However, there are other methods for doing this, such as subspace methods, which use eigenvector decomposition techniques. These methods are covered in other texts (see Kay, 1988; Therrien, 1992).

A major usage of linear prediction, besides spectral estimation, is for speech coding, where the method is called linear predictive coding (LPC). For example, suppose a 10th order model is used to code (represent) 100 samples of speech data. The coefficients are transmitted over a communication channel and the speech is reconstructed (synthesized) using the LPC coefficients in a filter that is excited by a model of the speech excitation waveform. The excitation model may be a discrete multipulse random sequence distributed over a pitch period. This technique can produce very high-quality speech, with a reduction in data of ten to one or so. Similar applications are used in some speaking toys and talking calculators, as well as in speech applications software for personal computers.

The spectral estimate using the LP coefficients determined by the covariance method is usually better (less bias and less variance) than the spectral estimate using the coefficients determined by the autocorrelation method. This is because no data window is used with the covariance method.

While we do not cover the MA or ARMA parameter estimation techniques in depth, an outline of the procedures is briefly presented. The MA method models the data with q zeros. Consequently, the MA power spectral estimation method is useful for estimating broad peaks or sharp nulls in the spectrum, while poles are used to estimate sharp peaks. Durbin's algorithm is often used to estimate the coefficients of the MA model. The idea of

this algorithm is to first fit a large order, say L, AR model to the data, where $q \ll L \ll N$. Then let the AR parameters serve as a data sequence to estimate the MA parameters. Thus, the algorithm calculates the autocorrelation matrix using the AR parameters as data and then solves the normal equations for the MA parameters, which are the unknowns. Thus, we have an equation similar to Equation (5.3.3.18), but one that involves the $\hat{b}_i$ coefficients. The idea for this algorithm comes from the fact that an MA model is equivalent to an infinite order AR model. So that $B(z) = \frac{1}{A(z)}$, or $B(z)A(z) = 1$. By taking inverse transforms, we obtain an equation involving the AR coefficients and the MA coefficients that can be arranged into a form like the AR (or LP) equations.

The ARMA model is used to estimate spectra where there are both poles and zeros, which calls for a technique to estimate both the peaks and the nulls in the spectrum. Estimating the ARMA parameters is more difficult and involves nonlinear equations. There are no optimal solution methods available, so suboptimal methods are used. One procedure first estimates an AR(p) model using the data. The data is then filtered by A(z) to produce an approximate MA model. This data is then used to estimate the MA(q) parameters. Another method solves a set of nonlinear equations.

The software introduced in Chapter 2 has algorithms for both MA and ARMA spectral estimates; see Appendix 6 as well.

EXAMPLE 5.5

In this example, we compare in Figure 5.16 the magnitude squared of the FFT (periodogram) of a data record of the vowel /IY/ with the LP spectral estimate with $p = 11$. The LP spectrum matches the envelope of the periodogram well, but not the notches, as expected. Furthermore, the LP spectrum is smoother than the other spectral estimates we have examined. The dB and the frequency scales are the same as for Figure 5.7. Also shown are the pole locations in the z-plane. ■

EXAMPLE 5.6

Consider the following autocorrelation matrix and determine the LP coefficients using MATLAB.

$$\begin{bmatrix} 1.0 & 0.3 & 0.09 \\ 0.3 & 1.0 & 0.3 \\ 0.09 & 0.3 & 1.0 \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \end{bmatrix} = - \begin{bmatrix} 0.3 \\ 0.09 \\ 0.027 \end{bmatrix}$$

The $\hat{a}_i$ are -0.3, 0, and 0 for $i = 1, 2, 3$. From Equation (5.3.3.19), the total error is $1 + \hat{a}_1(0.3) + \hat{a}_2(0.09) + \hat{a}_3(0.027)$, which is 0.91. The spectral estimate is found from Equation (5.3.4.1), and is plotted in Figure 5.17. It might be surprising that only coefficient $\hat{a}_1$ is nonzero. However, the autocorrelation function for a first order LP process is proportional to $(-a)^{|k|}$, and such a process has only one pole. In this example $\hat{a}_1 = a = -0.3$, so the autocorrelation function is proportional to $(0.3)^{|k|}$, which agrees with the data given. ■

EXAMPLE 5.7

This example uses the function lpc in MATLAB to determine the power spectral estimate of the data shown in Figure 5.18, where the spectrum is also plotted. This algorithm uses the estimate for the biased autocorrelation function, which is calculated using the data. The same results are obtained if one calculates the autocorrelation function, then calculates the LP coefficients, and then calculates the spectrum. So this MATLAB function is quite convenient to use. The reader is encouraged to examine this function carefully. It is used in the software in Chapter 2. ■

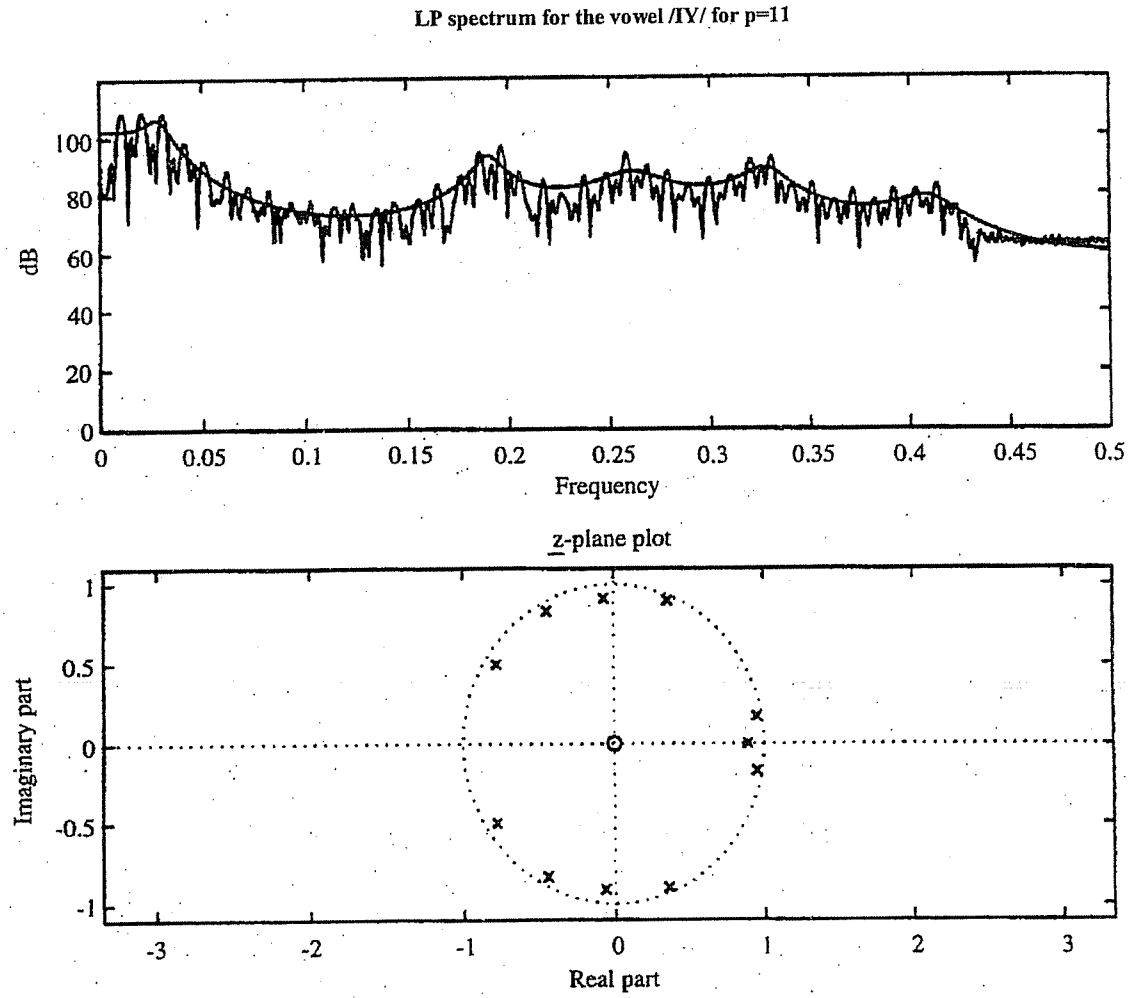


FIGURE 5.16 Comparison of the periodogram and the LP spectrum for the vowel /IY/.

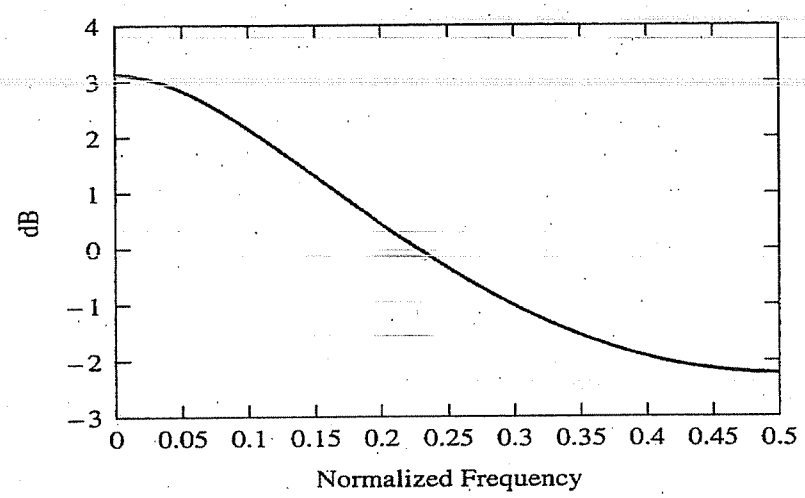


FIGURE 5.17 The spectrum for Example 5.6.

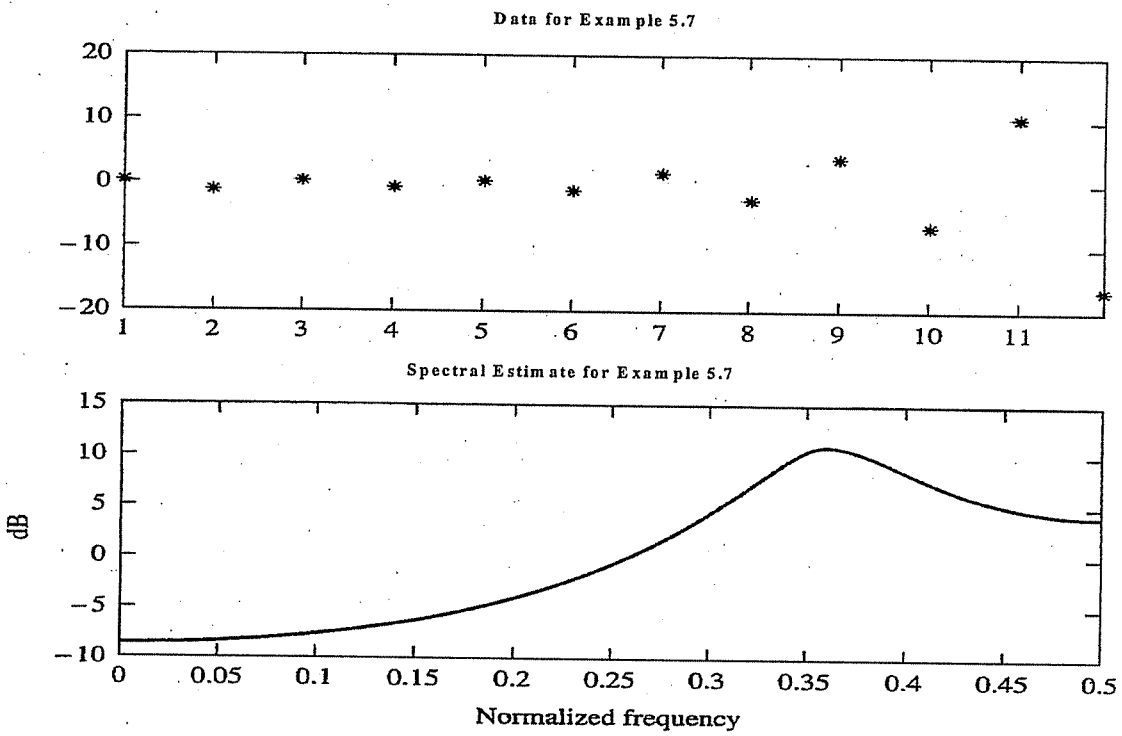


FIGURE 5.18 Results of lpc in Matlab for the data shown.

5.4 ALGORITHMS

We mentioned previously that the Levinson algorithm is used to solve the normal equations. What does this algorithm do? Suppose it is our task to find the pth order prediction filter coefficients for

$$\hat{x}(n) = \sum_{i=1}^{p} \hat{a}_k x(n-k)$$

The Levinson algorithm finds recursively the coefficients for the first-order filter, which we denote as $\hat{a}_1^1$, where the superscript denotes the filter order and the subscript denotes the coefficient number for the order. Next, the algorithm finds the coefficients for the second-order filter, $\hat{a}_1^2$ and $\hat{a}_2^2$, and so on, until the pth order filter, $\hat{a}_1^p, \hat{a}_2^p, \ldots, \hat{a}_p^p$. Note that there are several notations for the filter coefficients besides the one used here. Two others are $\hat{a}_p[1], \hat{a}_p[2], \ldots, \hat{a}_p[p]$, which is used in Kay (1988) and $\hat{a}_{p,1}, \hat{a}_{p,2}, \ldots, \hat{a}_{p,p}$. Thus, the Levinson algorithm gives all the lower order filter coefficients recursively, and it is terminated upon reaching the desired pth order filter. This algorithm requires fewer multiplications and additions than matrix inversion and is thus faster. However, the algorithm does require that the matrix of equations be Toeplitz. Levinson's algorithm is given as follows.

Initialize recursion

$$\hat{a}_1^1 = -\frac{\hat{R}_{XX}(1)}{\hat{R}_{XX}(0)}$$

$$[E_T^2]^1 = \text{prediction error power for first order filter} = \left(1 - (\hat{a}_1^1)^2\right)\hat{R}_{XX}(0)$$

Increase order to i = 2, Levinson recursion

$$\hat{a}_i^i = -\frac{\hat{R}_{XX}(i) + \sum_{j=1}^{i-1} \hat{a}_j^{i-1} \hat{R}_{XX}(i-j)}{(E_T^2)^{i-1}} = k_i = \text{reflection coefficient}$$

$$\hat{a}_j^i = \hat{a}_j^{i-1} + \hat{a}_i^i \hat{a}_{i-j}^{i-1}, \quad \text{for } j = 1, 2, \ldots, i-1$$

$$(E_T^2)^i = \left(1 - (\hat{a}_i^i)^2\right)(E_T^2)^{i-1}$$

Increase order by one

if order = $p + 1$ exit

The reflection coefficient is discussed in a later chapter. However, it is determined by having a concatenation of tubes for a vocal tract model. The reflection coefficient determines the amount of volume–velocity reflected at a junction of two tubes. The ith reflection coefficient is given as

$$k_i = \frac{A_{i+1} - A_i}{A_{i+1} + A_i}$$

where A_i is the cross-sectional area of the ith uniform tube. The reflection coefficient is $-1 \leq k_i \leq 1$. Thus, the reflection coefficient and, therefore, the ith coefficient of the ith order linear prediction filter determine the amount of volume–velocity reflected at a lossless tube junction. The reflection coefficient is also denoted as r_i. The reflection coefficients are also known as the partial correlation coefficients or PARCOR coefficients or the Itakura–Saito reflection coefficients. In the lattice filter problem the computation of the linear prediction coefficients becomes a problem of computing the PARCOR coefficients, which is the correlation coefficient between the forward and backward prediction errors.

The Cholesky decomposition algorithm is used to solve a positive definite matrix, [A], that is not Toeplitz. The idea for this algorithm is to decompose [A] such that $[A] = [L][D][L]^H$, where [L] is a lower triangular matrix with ones along the main diagonal, [D] is a diagonal matrix with real and positive elements on the main diagonal, and H denotes Hermitian transpose. The set of equations $[A]\vec{x} = \vec{b}$ is solved by two stages of back substitution. Again this algorithm is more efficient than matrix inversion techniques.

Burg's spectral estimator was derived by minimizing the sum of the forward and backward prediction errors in the linear prediction problem. In the material above, we derived the linear prediction filter coefficients by considering only the forward prediction error. However, one can define a backward prediction error in a similar manner. Burg's algorithm derives a set of reflection coefficients. However, these coefficients differ from those estimated by the PARCOR method. The advantage of Burg's approach is that the algorithm finds the reflection coefficients directly from the data, without having to calculate the autocorrelation coefficients. Furthermore, this approach does not need to window the data. The Burg spectral estimator is implemented in the software in Chapter 2.

5.5 ADDITIONAL NOTES ON LP

In the LP model, the error is orthogonal to the prediction term (or data). This is not always valid with real data. For example, voiced speech is a situation where the error is not orthogonal to the prediction term. However, if the speaker's pitch is low, then the error is nearly orthogonal to the prediction term, that is, the cross-correlation function between the speech signal and the error is approximately a delta function. Another way of interpreting this is

that the impulse response of the filter has sufficient time to decay (or damp out) before the next glottal excitation occurs. For male voices, this situation is approximately true. However for female and children's voices, this is not the case. The condition of orthogonality is met for the unvoiced case.

An accepted rule for selecting the order of the LP model is due to Markel and Gray (1976).

$$p = F_s + (4 \text{ or } 5), \quad \text{for voiced speech}$$
$$p = F_s, \quad \text{for unvoiced speech}$$

where F_s is the sampling frequency in kHz. Thus, there is one pole pair per kHz of the Nyquist bandwidth, which assumes that there is one formant per kHz. The extra 4 or 5 poles provide "spectral balance" as well as some freedom to adjust the spectrum for spectral roll-off due to the source spectrum.

In speech analysis, especially inverse filtering, the speech signal is often pre-emphasized prior to inverse filtering. There are two major reasons for this procedure. The first is due to the model of the speech production process. Recall that we have assumed that the source can be modeled with two poles and the radiation is modeled as a zero. Since the system is linear, the product of the source-transfer function with the radiation-transfer function leaves one pole, provided we assume one pole of the source approximately cancels the radiation zero. If we now pre-emphasize the speech signal by multiplying it with a transfer function that has a zero at the remaining pole location of the source, then we are left with only the poles of the model that represents the vocal tract filter. A second reason is based on preventing numerical instability. In the autocorrelation method, if the speech signal contains primarily low frequencies, then it is highly predictable. Thus, a large order LP model will result in an ill-conditioned autocorrelation matrix. Thus, pre-emphasis tends to whiten the spectrum, making it flatter and removing some of the spectral tilt due to the source or other causes.

Since the coefficients of A(z) are real, then the roots of A(z) appear in complex conjugate pairs with one pair denoted as $z_i = m_i e^{j\theta_i}$ and $z_i^* = m_i e^{-j\theta_i}$. Then an estimate of the formant frequency is

$$f_i = \frac{\omega_i}{2\pi} = \frac{1}{2\pi}\frac{1}{T}[\arg(z_i)] \text{ Hz} \tag{5.5.1}$$

and the formant bandwidth estimate is

$$b_i = \frac{1}{\pi}\frac{1}{T}|\log(m_i)| \text{ Hz} \tag{5.5.2}$$

When the order of the filter, p, is twice the number of formants in the frequency range 0 to $\frac{1}{2T}$, where T is the sampling interval, the roots of A(z) in the equation are almost exactly the formants. If p is larger than the number of formants, the roots include not only the formants but also some spurious poles corresponding to small peaks in the spectrum and real poles of zero or $\frac{1}{2T}$ Hz. The real poles represent the slope of the spectral envelope. If p is less than the number of formants, some adjacent low-level formants, which very often appear in the high-frequency range, are combined together and approximated with one pole of wide bandwidth. For a telephone voice line, the frequency range is up about 3 to 4 kHz, and usually, p is 8 to 10, but for a high-fidelity voice band that covers 5 to 8 kHz, p is 12 to 16.

Generally, a spurious formant has a relatively wider bandwidth than a true formant. Real poles often appear at 0 or $\frac{1}{2T}$ Hz. They represent the overall slope of the spectral envelope. In all cases, careful selection of the poles is necessary to extract the true formants

from the roots of A(z) in order to maintain continuity of formant frequency movement in a frame to frame analysis sequence.

The log area coefficients are given as

$$g_i = \log\left[\frac{A_{i+1}}{A_i}\right] = \log\left[\frac{1-k_i}{1+k_i}\right], \quad 1 \le i \le p \tag{5.5.3}$$

We can express the PARCOR coefficients as

$$k_i = \frac{1-e^{g_i}}{1+e^{g_i}} \tag{5.5.4}$$

The log area coefficients have found use in speaker recognition systems.

5.6 SUMMARY

Two estimators for the autocorrelation function were presented and discussed. We concluded that the biased estimator is the best.

There are two major power spectral estimators: parametric and non-parametric. The non-parametric spectral estimators include the periodogram and the Blackman–Tukey methods. These methods are often included in a class of classical estimators because they are among the first to be used. The parametric spectral estimators use data modeling methods and are generally considered superior to the classical methods. The parametric procedures require a means to estimate the parameters of the data model that is selected. We used the method of least square, which requires no assumptions about the statistics of the data.

This chapter basically provides a theoretical discussion of some of the software features introduced in Chapter 2 and discussed in Appendix 6.

Consult Furui (1989) and Furui and Sondhi (1992) for additional details.

PROBLEMS

5.1 Show that Equation (5.3.3.20) is equivalent to Equation (5.3.3.18) plus Equation (5.3.3.19).

5.2 This problem is to be completed using analytical methods so that you will gain some insight into the appropriate equations. Suppose you have a sample from a wide sense stationary random sequence as follows.

$$0, 1, 0, -1, 2$$

Determine, $\hat{R}(0)$, $\hat{R}(1)$, $\hat{R}(2)$ using the biased autocorrelation estimate. Next determine the correlation matrix for the autocorrelation method. Finally, determine the sample mean.

5.3 A random process, X(n), is to be approximated by a straight line using the estimate, $\hat{X}(n) = a + bn$. Determine the least squares estimates for a and b if N samples of the error and X(n) are available. Next, suppose you are given the data sequence

$$0, 1, 0, -1, 2, -3, 5, 0, -7, 8$$

Determine the least square straight line fit to the data.

5.4 Suppose a zero mean random sequence, X(n), has an autocorrelation function $R_{XX}(k) = c^{|k|}$. An estimate of X(n) is

$$\hat{X}(2) = -\hat{a}_1 X(1) - \hat{a}_2 X(0)$$

What is the linear prediction estimate for X(2)? What is the total squared error?

5.5 A simple moving average filter processes a zero mean, unit variance random process X(n) such that the filter output is $Y(n) = (\frac{1}{2})[X(n) + X(n-1)]$ for $n = 0, \ldots, N-1$, where $X(-1) = 0$ and $X(N) = 0$ and the X(n) are independent, identically distributed random variables. Find the covariance (autocorrelation) matrix of Y(n). Now let the variance for X(n) be σ_{XX}^2; find the autocorrelation function of Y(n) and the power spectral density.

5.6 Let $R_{XX}(k) = (0.3)^{|k|}$ for $k = 0, 1, 2$. If we model the data with an LP model, we obtain $\hat{a}_1 = -0.3$, $\hat{a}_2 = 0$, $\hat{a}_3 = 0$. Consider the autocorrelation prediction Equation (5.3.3.27) and calculate $\hat{R}_{XX}(3)$. Discuss the implications of this result.

5.7 Derive the following impulse response for the first order $\frac{1}{A(z)}$ prediction filter.

$$h(n) = \delta(n) - \hat{a}_1 h(n-1)$$

where $\delta(n)$ is the unit pulse. Let $h(-1) = 0$. Let $\hat{a}_1 = -0.3$. Analytically generate 10 samples of h(n). Now suppose you have a data sequence

$$0.2, 0.5, 1.1, 0.7, -0.1$$

Determine the error sequence. Calculate the correlation matrix using $\hat{a}_1$ only. (There are two unknowns, and one equation. What can you do?) Repeat the problem for a second-order filter, where

$$h(n) = \delta(n) - \hat{a}_1 h(n-1) - \hat{a}_2 h(n-2)$$

where $h(-1) = h(-2) = 0$ and $\hat{a}_1 = -0.625$ and $\hat{a}_2 = 0.25$.

5.8 The purpose of this problem is to estimate the filter coefficients for a second-order filter, i.e., $p = 2$. Assume you are given $\hat{R}_{XX}(0)$, $\hat{R}_{XX}(1)$, and $\hat{R}_{XX}(2)$. Determine the filter coefficients by matrix inversion. Repeat the estimation using Levinson's algorithm.

5.9 Calculate and plot the periodogram for each of the windows in MATLAB, Bartlett, triangle, kaiser, hanning, hamming, chebyshev, boxcar (rectangular), and blackman. Which window has the lowest sidelobes? Which window has the narrowest main lobe? Which window has the widest main lobe?

5.10 Generate a 100 point data sequence using randn in MATLAB. Calculate the total squared error for a linear prediction model of the data for $p = 1, 2, \ldots, 10$. Plot the error versus p. What value of p has the smallest error? Compare your results with the Akaike information criterion.

5.11 Calculate the spectrograms for the files m0125s.dat and m0127s. Identify and label the vowel regions.

5.12 Generate a 100 point random sequence using rand in MATLAB. Use a first-order LP filter to filter this random sequence. The filter is

$$A(z) = \frac{1}{1 + a_1 z^{-1}}$$

Let $a_1 = -0.1, -0.3$, and -0.7. Calculate and compare the autocorrelation function for the output of the three filters. Do the results agree with the theory? Repeat this problem using randn in MATLAB. Are there any significant differences in your results?

5.13 Suppose we have the following correlation matrix

$$\begin{bmatrix} 1.0 & 0.3 \\ 0.3 & 1.0 \end{bmatrix}$$

Determine the first order LP model. Calculate $\hat{R}_{XX}(2)$ using the autocorrelation prediction Equation (5.3.3.27). Now suppose you are told the correlation matrix is

$$\begin{bmatrix} 1.0 & 0.3 & (0.3)^4 \\ 0.3 & 1.0 & 0.3 \\ (0.3)^4 & 0.3 & 1.0 \end{bmatrix}$$

Compare $\hat{R}_{XX}(2)$ to the true value. Now determine the second order LP model. Calculate $\hat{R}_{XX}(3)$ using Equation (5.3.3.27). Plot and compare the true values of the autocorrelation function with the predicted values.

5.14 Analyze a segment of the vowel /IY/ in the data file m0103s.dat. Use LP models with p = 10, 11, 12, 13, and 14. Window the data with a hamming window prior to analysis. Calculate and plot the LP spectra and compare with the periodogram. Calculate and plot the total squared error versus the order, p. The first three resonances for this vowel (/IY/ as in see) occur at approximately 270, 2290, and 3010 Hz. Which model order appears to give the best results? Does this agree with the periodogram results? Does this agree with the error versus p results? Now that you have the "best" LP model for this data, predict and plot the unwindowed data beyond the last data value in the file, m0103s.dat. Comment on this prediction. For how many samples does the prediction appear to work well? Repeat this problem for the vowel /AA/ (as in Bach) in file m0113s.dat. The first three resonances for this vowel occur at approximately 570, 840, and 2410 Hz. Note that this problem can be done using the software in Chapter 2.

5.15 Apply a hamming window to the speech file m0103s.dat. Find the LP filter $\frac{1}{A(z)}$. Now inverse filter the speech signal and plot the data sequence, e(n). Discuss your result including why the sequence has peaks where it does. Hint: compare the location of the peaks in e(n) with the speech data. This problem can be done using the software in Chapter 2.

5.16 The purpose of this problem is to compare the various spectral analysis techniques for the vowel /IY/, file m0103s.dat. Use the software in Chapter 2. Load the file. Calculate and plot the following: spectra: FFT, periodogram, Blackman–Tukey. Use the following parameter settings: Frame length: 128, Hamming window, Select starting point (which is to be near the middle of the data file). Repeat using frame lengths of 256, 512, and 1024. Compare and discuss your results. Does frame length have an effect?

5.17 Repeat Problem 5.16 but change the hamming window to a rectangular window. Compare and discuss your results. Next compare and discuss the results of Problem 5.16 with the results obtained in this problem. What do you conclude about frame length and window type?

5.18 The purpose of this problem is to compare the autocorrelation and covariance algorithms for LP spectral analysis for the vowel /IY/, file m0103s.dat. Use the Chapter 2 software. Load the file. Calculate the following: LP analysis: autocorrelation and covariance. Use the following parameter settings: frame length: 256, number of poles: 10, 11, 12, 13, 14, 15, 16, select starting point (which is to be near the middle of the data file). Compare and discuss the results for the two algorithms and the seven different settings for the number of poles. Which algorithm appears to be "best"? What value of number of poles appears to be "best"?

5.19 The purpose of this problem is to calculate the model order selection for LP analysis. Use the vowel /IY/, file m0103s.dat. Use the Chapter 2 software. Load the file. Select the LP analysis property window. The settings in the property window are not important. Simply press the Order Selection button. The analyses for the three criteria are calculated and displayed for p = 10 to 14. What value of p gives the least error? Compare these results with the results of Problem 5.18.

5.20 The purpose of this problem is to compare the results for two ARMA spectral analysis techniques with the LP results for Problem 5.18. Use the vowel /IY/, file m0103s.dat. Use the Chapter 2 software. Load the file. Calculate the following: ARMA: Akaike MLE and MYWE. Use the following parameter settings: Frame length: 256, Number of poles: 10, 12, 14, Number of zeros: 2, 4, 6, Hamming window, Select starting point (which is to be the middle of the data file). Compare and discuss the results for this problem with those for Problem 5.18. Does the ARMA analysis improve the estimation of the spectral envelope over that using LP analysis?

5.21 The purpose of this problem is to compare the results for MA analysis with the LP for Problem 5.18. Use the vowel /IY/, file m0103s.dat. Use the Chapter 2 software. Load the file. Calculate the MA spectrum. Use the following parameter settings: Frame length: 256, Number of zeros: 4, 6, 8, Hamming window, Select starting point (which is to be near the middle of the data file). Compare and discuss the results for this problem with those for Problem 5.18. Is LP analysis superior to MA analysis for speech?

5.22 The purpose of this problem is to compare the results for inverse filtering using three algorithms: pitch synchronous (manual), pitch asynchronous (manual), and asynchronous analysis. Use the vowel /IY/, file m0103s.dat. Use the Chapter 2 software. Load the file. Calculate the inverse filtered waveforms and the pitch and formant frequency contours. Follow the instructions in Chapter 2. Use the following parameter settings: Property window: Frame length: 256, Overlap: 56, Poles: 14, Zeros: 0 and 4, Between marks: (select a segment near the middle of the data file of about 2000 to 3000 samples in length). Compare and discuss your results. Does the inverse filtered waveform appear reasonable (ignore any dc offset)? How about the pitch and formant contours? Compare the formant contours with the spectrogram for the data analyzed.

5.23 The purpose of this problem is to compare the Burg spectral analysis algorithm with the LP spectral analysis algorithm for the vowel /IY/, file m0103s.dat. Use the Chapter 2 software. Load the file. Calculate the Burg analysis using the following parameter settings: Frame length: 256, Number of poles: 10, 11, 12, 13, 14, 15, 16, Select starting point (which is to be near the middle of the data file). Compare and discuss the results for the Burg algorithm and the seven different settings for the number of poles with those for Problem 5.18. Which algorithm appears to be "best"? What value of number of poles appears to be "best"?

RESEARCH PROBLEMS

R5.1 Read the paper in Appendix 6 on WRLS-VFF analysis. Conduct a WRLS-VFF analysis of the data used in Problem 5.22. Compare the results with those of Problem 5.22.

R5.2 Obtain a copy of the Hermansky paper on perceptual linear prediction (PLP). Read the paper. Repeat Problem 5.18 using the PLP method. The algorithm is available in the Chapter 2 software. Compare the results of this problem with those for Problem 5.18.

R5.3 Derive the Levinson algorithm. Help is available in several texts.