

## Correlation

Our study of signal processing systems has been dominated by the concept of ‘convolution’, and we have somewhat neglected its close relative the ‘correlation’. While formally similar (in fact convolution by a symmetric FIR filter can be considered a correlation as well), the way one should think about the two is different. Convolution is usually between a signal and a filter; we think of it as a system with a single input and stored coefficients. Crosscorrelation is usually between two signals; we think of a system with two inputs and no stored coefficients. The difference may be only in our minds, but nonetheless this mind-set influences the way the two are most often used.

Although somewhat neglected we weren’t able to get this far without mentioning correlations at all. We have already learned that crosscorrelation is a measure of similarity between two signals, while autocorrelation is a measure of how similar a signal is to itself. In Section 5.6 we met the autocorrelation for stochastic signals (which are often quite unlike themselves), and in Section 6.13 we used the crosscorrelation between input and output signals to help identify an unknown system.

Correlations are the main theme that links together the present chapter. We first motivate the concept of correlation by considering how to compare an input signal to a reference signal. We find that the best signal detector is the correlator. After formally defining both crosscorrelation and autocorrelation and calculating some examples, we prove the important Wiener-Khintchine theorem, which relates the autocorrelation to the power spectral density (PSD).

Next we compare correlation with convolution and discover that the optimal signal detector can be implemented as a matched filter. The matched filter was invented for radar and a digression into this important application is worthwhile. The matched filter is good for signal detection, but for cleaning up a partially unknown signal we need the Wiener filter, which is also based on correlations.

There is also a close connection between correlation and prediction. Linear predictive coding is crucial in speech processing, and we present it here in preparation for our later studies.

The Wiener-Khintchine theorem states that correlations are second-order entities. Although these are sufficient for a wide variety of tasks, we end this chapter with a short introduction to the more general higher-order signal processing.

## 9.1 Signal Comparison and Detection

A *signal detector* is a device that alerts us when a desired signal appears. Radar and sonar operate by transmitting a signal and detecting its return after having been reflected by a distant target. The return signal is often extremely weak in amplitude, while interference and noise are strong. In order to be able to reliably detect the presence of the return signal we employ a signal detector whose output is maximized when a true reflection appears. Similar signal detectors are employed in telephony call progress processing, medical alert devices, and in numerous other applications.

Envision a system with a single input that must sound an alarm when this input consists of some specified signal. It is important not to miss any events even when the signal is weak compared to the noise, but at the same time we don't want to encourage false alarms (reporting detection when the desired signal was not really there). In addition, we may need to know as accurately as possible precisely *when* the expected signal arrived.

The signal to be detected may be as simple as a sinusoid of given frequency, but is more often a rather complex, but known signal. It is evident that signal detection is closely related to *signal comparison*, the determination of how closely a signal resembles a reference signal. Signal comparison is also a critically important element in its own right, for example, in digital communications systems. In the simplest of such systems one of several basic signals is transmitted every  $T$  seconds and the receiver must determine which. This can be accomplished by building signal detectors for each of the basic signals and choosing the signal whose respective detector's output is the highest. A more complex example is speech recognition, where we may build detectors for a multitude of different basic sounds and convert the input audio into a string of best matches. Generalization of this technique to images produces a multitude of further applications, including optical character recognition.

From these examples we see that comparison and detection are essentially the same. The simplest detector is implemented by comparing the output of a comparator to a threshold. Complex detectors may employ more sophisticated decision elements, but still require the basic comparison mechanism to function.

Signal detection and comparison are nontrivial problems due to the presence of noise. We know how to build filters that selectively enhance defined frequency components as compared to noise; but how do we build a system that selectively responds to a known but arbitrary reference signal? Our first inclination would be to subtract the input signal  $s_n$  from the desired reference  $r_n$ , thus forming an error signal  $\epsilon_n = r_n - s_n$ . Were the error signal to be identically zero, this would imply that the input precisely matches the reference, thus triggering the signal detector or maximizing the output of the signal comparator. However, for an input signal contaminated by noise  $s_n = r_n + \nu_n$ , we can not expect the instantaneous error to be identically zero, but the lower the energy of the error signal the better the implied match. So a system that computes the energy of the difference signal is a natural comparator.

This idea of using a simple difference is a step in the right direction, but only the first step. The problem is that we have assumed that the input signal is simply the reference signal plus additive noise; and this is too strong an assumption. The most obvious reason for this discrepancy is that the amplitude of the input signal is usually arbitrary. The strength of a radar return signal depends on the cross-sectional area of the target, the distance from the transmitter to the target and the target to the receiver, the type and size of the radar antenna, etc. Communications signals are received after path loss, and in the receiver probably go through several stages of analog amplification, including automatic gain control. A more reasonable representation of the input signal is

$$s_n = A r_n + \nu_n$$

where  $A$  is some unknown gain parameter.

In order to compare the received signal  $s_n$  with the reference signal  $r_n$  it is no longer sufficient to simply form the difference; instead we now have to find a gain parameter  $g$  such that  $r_n - g s_n$  is minimized. We can then use the energy of the resulting error signal

$$\epsilon_n = \min_g (r_n - g s_n)$$

as the final match criterion. How can we find this  $g$ ? Assuming for the

moment that there is no noise, then for every time  $n$  we require

$$g = \frac{r_n}{s_n} = \frac{1}{A} \quad (9.1)$$

in addition to the weaker constraint that the error energy be zero.

$$\sum_n (r_n - g s_n)^2 = 0 \quad (9.2)$$

By opening the square the latter becomes

$$\sum_n r_n^2 - 2g \sum_n r_n s_n + g^2 \sum_n s_n^2 = 0$$

which can be rewritten in the following way.

$$E_r - 2g C_{rs} + g^2 E_s = 0 \quad (9.3)$$

Here  $E_r$  is the energy of the reference signal,  $E_s$  is the energy of the input signal, and  $C_{rs} = \sum_n r_n s_n$  is the *crosscorrelation* between the reference and the input. Among all input signals of given energy the correlation is maximal exactly when the energy of the difference signal is minimal.

Now, from equation (9.1) we can deduce that

$$g^2 = \frac{\sum_n r_n^2}{\sum_n s_n^2} = \frac{E_r}{E_s}$$

which when substituted into (9.3) brings us to the conclusion that

$$C_{rs} = \sqrt{E_r E_s} \quad (9.4)$$

in the absence of noise. When the input signal does not precisely match the reference, due to distortion or noise, we have  $|C_{rs}| < \sqrt{E_r E_s}$ . The cross-correlation  $C_{rs}$  is thus an easily computed quantity that compares the input signal to the reference, even when the amplitudes are not equal. A comparator can thus be realized by simply computing the correlation, and a signal detector can be implemented by comparing it to  $\sqrt{E_r E_s}$  (e.g., requiring  $|C_{rs}| > \sqrt{E_r E_s} - \epsilon$  or  $|C_{rs}| > \gamma \sqrt{E_r E_s}$ ).

Unfortunately we have not yet considered all that happens to the reference signal before it becomes an input signal. In addition to the additive noise and unknown gain, there will also usually be an unknown time shift. For communications signals we receive a stream of signals to compare, each offset by an unknown time delay. For the radar signal the time delay derives

from the round-trip time of the signal from the transmitter to the target and back, and is precisely the quantity we wish to measure. When there is a time shift, a reasonable representation of the input signal is

$$s_n = Ar_{n+m} + \nu_n \quad \forall n$$

where  $A$  is the gain and  $m < 0$  the time shift parameter.

In order to compare the received signal  $s_n$  with the reference signal  $r_n$  we can no longer simply compute a single crosscorrelation; instead we now have to find the time shift parameter  $m$  such that

$$C_{rs}(m) = \sum_n r_{n+m}s_n = \sum_n r_n s_{n-m}$$

is maximal. How do we find  $m$ ? The only way is to compute the crosscorrelation  $C_{rs}(m)$  for all relevant time shifts (also called time ‘lags’)  $m$  and choose the maximal one. It is this

$$C_{rs} = \max_m C_{rs}(m)$$

that must be compared with  $\sqrt{E_r E_s}$  in order to decide whether a signal has been detected.

## EXERCISES

- 9.1.1 Formulate the concept of correlation in the frequency domain starting from spectral difference and taking into account an arbitrary gain of the spectral distribution. What happens if we need to allow an arbitrary spectral shift?
- 9.1.2 Give a complete algorithm for the optimal detection of a radar return  $s_n$  given that the transmitted signal  $r_n$  was sent at time  $T_1$ , returns are expected to be received before time  $T_2$ , and the correlation is required to be at least  $\gamma$ . Note that you can precompute  $E_r$  and compute  $E_s$  and  $C_{rs}(m)$  in one loop.
- 9.1.3 Design an optimal detector for the V.34 probe signal introduced in exercise 2.6.4. The basic idea is to perform a DFT and implement a correlator in the frequency domain by multiplying the spectrum by a comb with 21 pass-bands (of suitable bandwidth). However, note that this is not independent of signal strength. You might try correcting this defect by requiring the correlation to be over 80% of the total signal energy, but this wouldn’t work properly since, e.g., *answer tone* (a pure 2100 Hz tone) would trigger it, being one of the frequencies of the probe signal. What is wrong? How can this problem be solved?

## 9.2 Crosscorrelation and Autocorrelation

The time has come to formally define correlation.

### Definition: crosscorrelation

The crosscorrelation between two real signals  $x$  and  $y$  is given by

$$C_{xy}(\tau) \equiv \int_{-\infty}^{\infty} x(t)y(t-\tau)dt \quad \mathbf{A} \left| \mathbf{D} \quad C_{xy}(m) \equiv \sum_{n=-\infty}^{\infty} x_n y_{n-m} \quad (9.5)$$

where the time shift  $\tau$  or  $m$  is called the *lag*. ■

There is an important special case, called *autocorrelation*, when  $y$  is taken to be  $x$ . It might seem strange to compare a signal with itself, but the lag in equation (9.5) means that we are actually comparing the signal at different times. Thus autocorrelation can assist in detecting periodicities.

### Definition: autocorrelation

The autocorrelation of a real signal  $s$  is given by

$$C_s(\tau) \equiv \int_{-\infty}^{\infty} s(t)s(t-\tau)dt \quad \mathbf{A} \left| \mathbf{D} \quad C_s(m) \equiv \sum_{n=-\infty}^{\infty} s_n s_{n-m} \quad (9.6)$$

and the normalized autocorrelation is defined to be

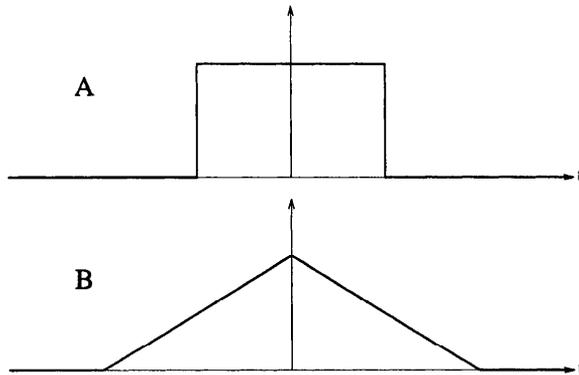
$$c_s(\tau) \equiv \frac{C_s(\tau)}{C_s(0)} \quad \mathbf{A} \left| \mathbf{D} \quad c_s(m) \equiv \frac{C_s(m)}{C_s(0)} \quad (9.7)$$

where  $\tau$  or  $m$  is called the *lag*. ■

These definitions are consistent with those of Section 5.6 for the case of stationary ergodic signals. In practice we often approximate the autocorrelation of equation (5.22) by using equation (9.6) but with the sum only over a finite amount of time. The resulting quantity is called the *empirical autocorrelation*. The correlation is also somewhat related to the *covariance matrix* of vector random variables, and strongly related to the convolution, as will be discussed in the next section.

Before discussing properties of the correlations, let's try calculating a few. The analog rectangular window

$$s(t) = \begin{cases} 1 & |t| < 1 \\ 0 & \text{else} \end{cases}$$



**Figure 9.1:** The autocorrelation of an analog rectangularly shaped signal. In (A) the signal is depicted while the autocorrelation is in (B). Note that the autocorrelation is symmetric and has its maximal value at the origin.

is depicted in Figure 9.1.A. Its autocorrelation is given by the triangular

$$C_s(\tau) = \int_{-\infty}^{\infty} s(t)s(t - \tau)dt = \int_{\max(-1,\tau-1)}^{\min(1,\tau+1)} dt = (2 - |\tau|) \quad (9.8)$$

depicted in Figure 9.1.B. In that figure we see several features that are readily shown to be more general. The autocorrelation is symmetric around time lag zero, and it takes on its maximum value at lag zero, where it is simply the energy  $E_s$ . The autocorrelation is also wider than the original signal, but attacks and decays more slowly.

Had we used an inverted rectangle (which differs from the original signal by a phase shift)

$$s(t) = \begin{cases} -1 & |t| < 1 \\ 0 & \text{else} \end{cases}$$

we would have found the same autocorrelation. Indeed the generalization of autocorrelation to complex signals,

$$C_s(\tau) \equiv \int_{-\infty}^{\infty} s^*(t)s(t - \tau)dt \quad \mathbf{A} \quad \mathbf{D} \quad C_s(m) \equiv \sum_{n=-\infty}^{\infty} s_n^* s_{n-m} \quad (9.9)$$

can be shown to be *phase blind* (unchanged by multiplying  $s$  by a common phase factor).

What is the autocorrelation of the periodic square wave  $\square(t)$ ? Generalizing our previous result we can show that the autocorrelation is a periodic

triangular wave of the same period. This too is quite general—the autocorrelation of a periodic signal is periodic with the same period; and since the lag-zero autocorrelation is a global maximum, all lags that are multiples of the period have globally maximal autocorrelations. This fact is precisely the secret behind using autocorrelation for determining the period of a periodic phenomenon. One looks for the first nonzero peak in the autocorrelation as an indication of the period. The same idea can be used for finding Fourier components as well; each component contributes a local peak to the autocorrelation.

As our final example, let's try a digital autocorrelation. The signal  $b_n$  is assumed to be zero except for  $n = 1 \dots 13$  where it takes on the values  $\pm 1$ .

$$\dots 0, 0, +1, +1, +1, +1, +1, -1, -1, +1, +1, -1, +1, -1, +1, 0, 0, \dots \quad (9.10)$$

Its autocorrelation is easily computed to be  $C(0) = 13$ ,  $C(m) = 0$  for odd  $m$  in the range  $-13 < m < 13$ ,  $C(m) = 1$  for even nonzero  $m$  in this range, and all other autocorrelations are zero. We see that the autocorrelation is indeed maximal at  $m = 0$  and symmetric, and in addition the highest nonzero-lag correlations are only 1. Signals consisting of  $\pm 1$  values with this last property (i.e., with maximal nontrivial autocorrelation of  $\frac{C(0)}{N}$  or less) are called *Barker codes*, and are useful for timing and synchronization. There is no known way of generating Barker codes and none longer than this one are known.

The definitions for autocorrelation or crosscorrelation given above involve integrating or summing over all times, and hence are not amenable to computation in practice. In any case we would like to allow signals to change behavior with time, and thus would like to allow correlations that are defined for finite time durations. The situation is analogous to the problem that led to the definition of the STFT, and we follow the same tactic here. Assuming a rectangular window of length  $N$ , there are  $N$  terms in the expression for the zero lag, but only  $N - 1$  terms contribute to the lag 1 correlation  $s_1 s_0 + s_2 s_1 + \dots + s_{N-1} s_{N-2}$ , and only  $N - m$  terms in the lag  $m$  sum. So we define the short-time autocorrelation

$$C_s(m) = \frac{1}{N - m} \sum_{n=1}^{N-m} s_n s_{n-m} \quad (9.11)$$

where now the zero lag is the power rather than the energy. This quantity is often called the unbiased empirical autocorrelation when it is looked upon as a numerical estimate of the full autocorrelation.

## EXERCISES

- 9.2.1 What is the connection between autocorrelation defined here for deterministic signals and the autocorrelation we earlier defined for stochastic signals (equation (5.22))?
- 9.2.2 What is the crosscorrelation between a signal  $s(t)$  and the impulse  $\delta(t)$ ?
- 9.2.3 Compute and draw the crosscorrelation between two analog rectangular signals of different widths.
- 9.2.4 Compute and draw the crosscorrelation between two analog triangular signals.
- 9.2.5 Show that  $C_{yx}(m) = C_{xy}(-m)$ .
- 9.2.6 Prove that the autocorrelation is symmetric and takes its maximum value at the origin, where it is the energy. Show that  $|c_{xy}(m)| \leq 1$ .
- 9.2.7 Can you find Barker codes of length 5, 7, and 11? What are their autocorrelations?
- 9.2.8 What is the proper generalization of crosscorrelation and autocorrelation to complex signals? (Hint: The autocorrelation should be phase independent.)
- 9.2.9 Prove that the autocorrelation of a periodic signal is periodic with the same period.
- 9.2.10 Prove that zero mean symmetric signals have zero odd lag autocorrelations.
- 9.2.11 Assume  $y_n = x_{n-l}$ . What are the connections between  $C_{xy}(m)$ ,  $C_y(m)$  and  $C_x(m)$ ?
- 9.2.12 Derive the first few autocorrelation values for  $s_n = A \sin(\omega n + \phi)$ .
- 9.2.13 Generalize the previous exercise and derive the following expression for the general autocorrelation of the sinusoid.

$$C_s(m) = \langle s_n s_{n+m} \rangle = \frac{A^2}{2} \cos(\omega m)$$

### 9.3 The Wiener-Khintchine Theorem

The applications of correlation that we have seen so far derive from its connection with the difference between two signals. Another class of applications originate in the relationship between autocorrelation and power spectrum (see Section 4.5), a relationship known as the Wiener-Khintchine Theorem.

The PSD of a signal is the absolute square of its FT, but it is also can be considered to be the FT of some function. Parseval's relation tells us that integrating the PSD over all frequencies is the same as integrating the square of the signal over all times, so it seems reasonable that the iFT of the PSD is somehow related to the square of the signal.

Could it be that the PSD is simply the FT of the signal squared? The DC term works because of Parseval, but what about the rest? We don't have to actually integrate or sum to find out since we can use the connection between convolution and FT of a product  $\text{FT}(xy) = X * Y$  (equation (4.18) or (4.46)). Using the signal  $s$  for both  $x$  and  $y$  we see that the FT of  $s^2(t)$  is  $S * S = \int S(\omega - \Omega)S(\Omega)d\Omega$ , which is not quite the PSD  $|S|^2 = S * S = S(-\omega)S(\omega)$  (for real signals), but has an additional integration. We want to move this integration to the time side of the equation, so let's try  $s * s$ . From equation (4.19) or (4.47) we see that the FT of  $s * s$  is  $S^2(\omega)$  which is even closer, but has both frequency variables positive, instead of one positive and one negative. So we need something very much like  $s * s$  but with some kind of time variable inversion; that sounds like the autocorrelation!

So let's find the FT of the autocorrelation.

$$\begin{aligned}
 \text{FT}(C_s(t)) &= \text{FT}\left(\int_{-\infty}^{\infty} s(\tau)s(\tau-t)d\tau\right) \\
 &= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} s(\tau)s(\tau-t)d\tau\right) e^{-i\omega t} dt \\
 &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} s(\tau)s(\tau-t)e^{-i\omega t} dt d\tau \\
 &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} s(\tau)s(T)e^{-i\omega(\tau-T)} dT d\tau \\
 &= \int_{-\infty}^{\infty} s(T)e^{i\omega T} dT \int_{-\infty}^{\infty} s(\tau)e^{-i\omega\tau} d\tau \\
 &= S^*(\omega) S(\omega) = |S(\omega)|^2
 \end{aligned}$$

The PSD at last!

We have thus proven the following celebrated theorem.

### The Wiener-Khintchine Theorem

The autocorrelation  $C_s(t)$  and the power spectrum  $S(\omega)$  are an FT pair. ■

Although we proved the theorem for deterministic analog signals, it is more general. In fact, in Section 5.7 we used the Wiener-Khintchine theorem as the definition of spectrum for random signals.

As a corollary to the theorem we can again prove that the autocorrelation is ‘phase blind’, that is, independent of the spectral phase. Two signals with the same power spectral density but different spectral phase will have the same autocorrelation function, and hence an infinite number of signals have the same autocorrelation. Methods of signal analysis that are based on autocorrelation can not differentiate between such signals, no matter how different they may look in the time domain. If we need to differentiate between such signals we need to use the higher-order statistics of Section 9.12.

## EXERCISES

- 9.3.1 The period of a pure sinusoid is evident as a peak in the autocorrelation and hence its frequency is manifested as a peak in the power spectrum. This is the true basis for the connection between autocorrelation and PSD. What can you say about the autocorrelation of a general periodic signal? What is the autocorrelation of the sum of two sinusoidal components? Can you see the PSD connection?
- 9.3.2 Express and prove the Wiener-Khintchine theorem for digital signals.
- 9.3.3 Generalize the Wiener-Khintchine theorem by finding the FT of the cross-correlation of two signals  $x(t)$  and  $y(t)$ .

## 9.4 The Frequency Domain Signal Detector

Simply observing the input signal in the time domain is not a very sensitive method of detecting low-SNR signals, a fact made obvious by looking back at Figure 2.9. Since correlation is a method for detecting weak signals, and correlation is related to spectrum by the Wiener-Khintchine theorem, there should be a way of exploiting the frequency domain for signal detection.

In Section 5.3 we saw how to reduce noise by averaging it out. This would seem to be a purely time domain activity, but there is a frequency domain connection. To see this, consider the simplest case, that of a pure sinusoid in noise. For averaging to optimally reinforce the signal we must first ensure that all the times intervals commence at precisely the same phase in a period, an operation called ‘time registration’. Without registration the signal cancels out just like the noise; with inaccurate registration the signal is only partially reinforced. If we wish to take successive time intervals, accurate registration requires the intervals to be precise multiples of the

sinusoid's basic period. Thus signal emphasis by averaging requires precise knowledge of the signal's frequency.

Now let's see how we can emphasize signals working directly in the frequency domain. In a digital implementation of the above averaging each time interval corresponds to a buffer of samples. Assume that the period is  $L$  samples and let's use a buffer with exactly  $k$  periods. We start filling up the buffer with the input signal consisting of signal plus noise. Once the buffer is filled we return to its beginning, adding the next signal sample to that already there. Performing this addition  $M$  times increases the sinusoidal component by  $M$  but the noise component only by  $\sqrt{M}$  (see exercise 5.3.1). Hence the SNR, defined as the ratio of the signal to noise *energies*, is improved by  $M$ . This SNR increase is called the *processing gain*.

How many input samples did we use in the above process? We filled the buffer of length  $kL$  exactly  $M$  times; thus  $N = kLM$  input samples were needed. We can use a buffer with length corresponding to any integer number of periods  $k$ , but the  $N$  input signal samples are used most efficiently when the buffer contains a single cycle  $k = 1$ . This is because the processing gain  $M = \frac{N}{kL}$  will be maximal for a given  $N$  when  $k = 1$ . However, it is possible to do even better! It is possible to effectively reduce the 'buffer' to a single sample such that  $M = N$ , and obtain the maximal processing gain of  $N$ .

All we have to do is to downmix the signal to DC, by multiplying by a complex exponential and low-pass filtering. The noise will remain zero mean while the sinusoid becomes a complex constant, so that averaging as in Section 6.6 cancels out the noise but reinforces the constant signal. Now, as explained in Section 13.2, this complex downmixing can be performed using the DFT. So by performing a DFT the energy in the bin corresponding to the desired signal frequency increases much faster than all the other bins. In the frequency domain interpretation the processing gain is realized due to the signal being concentrated in this single bin, while the white noise is spread out over  $N$  bins. Thus were the signal and noise energies initially equal, the ratio of the energy in the bin corresponding to the signal frequency to that of the other bins would be  $N$ , the same processing gain deduced from time domain arguments.

So we see that our presumption based on the Wiener-Khintchine theorem was correct; the frequency domain interpretation is indeed useful in signal detection. Although we discussed only the simple case of a single pure sinusoid, it is relatively easy to extend the ideas of this section to more general signals by defining distinctive spectral signatures. Instead of doing this we will return to the time domain and see how to build there a signal detection system for arbitrary signals.

## EXERCISES

- 9.4.1 Express the processing gain in decibels when the DFT is performed using a  $2^m$  point FFT.
- 9.4.2 In the text we tacitly assumed the signal frequency to be precisely at a bin center. If this is not the case a window function  $w_n$  (see Section 13.4) must be employed. Show that with a window the signal energy is enhanced by  $(\sum_n w_n)^2$  while the noise energy is increased by  $\sum_n w_n^2$  thus resulting in a processing gain of the ratio of these two expressions.
- 9.4.3 Build a detector for a signal that consists of the equally weighted sum of two sinusoids. Is it worthwhile taking the phases into account? What if the signal is the weighted sum of the two sinusoids?
- 9.4.4 Extend the technique of the previous exercise and build a DFT-based detector for a completely general signal.

## 9.5 Correlation and Convolution

Although we have not mentioned it until now, you have no doubt noticed the similarity between the expression for digital crosscorrelation in equation (9.5) and that for convolution in equation (6.13). The only difference between them is that in correlation both indices run in the same direction, while in convolution they run in opposite directions. Realizing this, we can now realize our signal comparator as a filter. The filter's coefficients will be the reference signal reversed in time, as in equation (2.16). Such a filter is called a *matched filter*, or a correlator. The name matched filter refers to the fact that the filter coefficients are matched to the signal values, although in reverse order.

What is the frequency response of the matched filter? Reversing a signal in time results in frequency components  $\text{FT}(s(-t)) = S(-\omega)$ , and if the signal is real this equals  $S^*(\omega)$ , so the magnitude of the FT remains unchanged but the phase is reversed.

From the arguments of Section 9.1 the correlator, and hence the theoretically identical matched filter, is the optimum solution to the problem of detecting the appearance of a known signal  $s_n$  contaminated by additive white noise  $x_n = s_n + \nu_n$ .

Can we extend this idea to optimally detect a signal in colored noise? To answer this question recall the joke about the mathematician who wanted a

cup of tea. Usually he would take the kettle from the cupboard, fill it with water, put it on the fire, and when the water boiled, pour it into a cup and drop in a tea bag. One day he found that someone had already boiled the water. He stared perplexed at the kettle and then smiled. He went to the sink, poured out the boiling water, returned the kettle to the cupboard and declared triumphantly: 'The problem has been reduced to one we know how to solve.'

How can we reduce the problem of a signal in colored noise to the one for which the matched filter is the optimal answer? All we have to do is filter the contaminated signal  $x_n$  by a filter whose frequency response is the inverse of this noise spectrum. Such a filter is called a *whitening filter*, because it flattens the noise spectrum. The filtered signal  $x'_n = s'_n + \nu'_n$  now contains an additive white noise component  $\nu'_n$ , and the conditions required for the matched filter to be optimal are satisfied. Of course the reference signal  $s'_n$  is no longer our original signal  $s_n$ ; but finding the matched filter for  $s'_n$  is straightforward.

## EXERCISES

- 9.5.1 Create a sinusoid and add Gaussian white noise of equal energy. Recover the sinusoid by averaging. Experiment with inaccurate registration. Now recover the sinusoid by a DFT. What advantages and disadvantages are there to this method? What happens if the frequency is inaccurately known?
- 9.5.2 Build a matched filter to detect the HPNA 1.0 pulse (see exercise 7.7.4). Try it out by synthesizing pulses at random times and adding Gaussian noise. HPNA 1.0 uses PPM where the information is in the pulse position. How precisely can you detect the pulse's time of arrival?
- 9.5.3 Compare the time domain matched filter with a frequency domain detector based on the FFT algorithm. Consider computational complexity, processing delay, and programming difficulty.

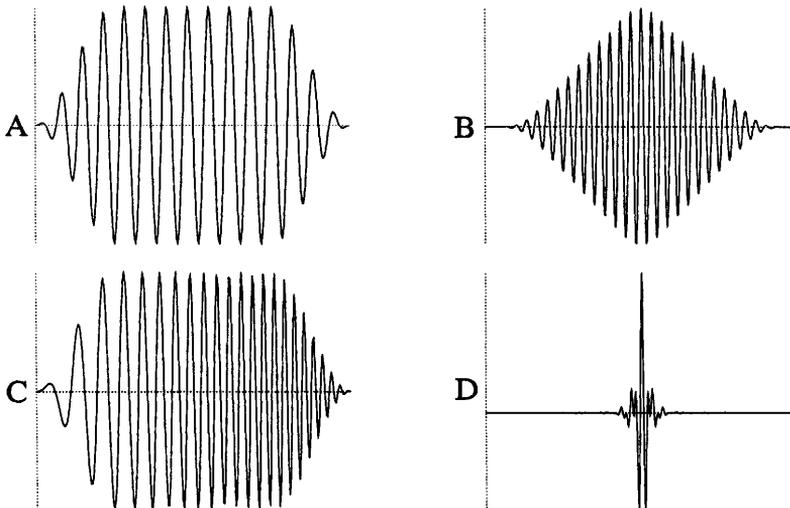
## 9.6 Application to Radar

Matched filters were invented in order to improve the detection of radar returns. We learned the basic principles of radar in Section 5.3 but were limited to explaining relatively primitive radar processing techniques. With

our newly acquired knowledge of matched filters we can now present improved radar signals and receivers.

Radar pulses need to have as much energy as possible in order to increase the probability of being detected, and thus should be long in duration. In order to increase a radar's range resolution we prefer narrow pulses since it's hard to tell when exactly a wide pulse arrives. How can we resolve this conflict of interests? The basic idea is to use a wide pulse but to *modulate* it (i.e., to change its characteristics with time). The output of a filter matched to this modulation can be made to be very short in duration, but containing all the energy of the original pulse.

To this end some radars vary their instantaneous frequency linearly with time over the duration of the pulse, a technique known as *FM chirp*. We demonstrate in Figure 9.2 the improvement chirp can bring in range resolution. The pulse in Figure 9.2.A is unmodulated and hence the matched filter can do no better than to lock onto the basic frequency. The output of such a matched filter is the autocorrelation of this pulse, and is displayed in Figure 9.2.B. Although theoretically there is a maximum corresponding to the perfect match when the entire pulse is overlapped by the matched filter, in practice the false maxima at shifts corresponding to the basic period



**Figure 9.2:** The autocorrelation of pulses with and without chirp. In (A) a pulse with constant instantaneous frequency is depicted, and its wide autocorrelation is displayed in (B). In (C) we present a pulse with frequency chirp; its much narrower autocorrelation is displayed in (D).

make it difficult to determine the precise TOA. In contrast the chirped pulse of Figure 9.2.C does not match itself well at any nontrivial shifts, and so its autocorrelation (Figure 9.2.D) is much narrower. Hence a matched filter built for a chirped radar pulse will have a much more precise response.

Chirped frequency is not the only way to sharpen a radar pulse's autocorrelation. Barker codes are often used because of their optimal autocorrelation properties, and the best way to embed a Barker code into a pulse is by changing its instantaneous phase. **Binary Phase Shift Keying (BPSK)**, to be discussed in Section 18.13, is generated by changing a sinusoidal signal's phase by  $180^\circ$ , or equivalently multiplying the sinusoid by  $-1$ . To use the 13-bit Barker code we divide the pulse width into 13 equal time intervals, and assign a value  $\pm 1$  to each. When the Barker code element is  $+1$  we transmit  $+\sin(\omega t)$ , while when it is  $-1$  we send  $-\sin(\omega t)$ . This Barker BPSK sharpens the pulse's autocorrelation by a factor of 13.

Not all radars utilize pulses; a **Continuous Wave (CW)** radar transmits continuously with constant amplitude. How can range be determined if echo arrives continuously? Once again by modulating the signal, and if we want constant amplitude we can only modulate the frequency or phase (e.g., by chirp or BPSK). Both chirp and BPSK modulation are popular for CW radars, with the modulation sequence repeating over and over again without stopping. CW radars use LFSR sequences rather than Barker codes for a very simple reason. Barker codes have optimal *linear* autocorrelation properties, while maximal-length LFSR sequences can be shown to have optimal *circular* autocorrelation characteristics. Circular correlation is analogous to circular convolution; instead of overlapping zero when one signal extends past the other, we wrap the other signal around periodically. A matched filter that runs over a periodically repeated BPSK sequence essentially reproduces the circular autocorrelation.

## EXERCISES

- 9.6.1 Plot, analogously to Figure 9.2, the autocorrelation of a pulse with a 13-bit Barker code BPSK.
- 9.6.2 What is the circular autocorrelation of the LFSR15 sequence?
- 9.6.3 What is the difference between coherent and incoherent pulse radars? In what way are coherent radars better?

## 9.7 The Wiener Filter

The matched filter provides the optimum solution to the problem of detecting the arrival of a known signal contaminated by noise; but correlation-based filters are useful for other problems as well, for example, removing noise from an unknown signal.

If the signal is known in the matched filter problem, then why do we need to clean it up? The reason is that the signal may be only partially known, and we must remove noise to learn the unknown portion. In one common situation we expect a signal from a family of signals and are required to discover which specific signal was received. Or we might know that the signal is a pure sinusoid, but be required to measure its precise frequency; this is the case for Doppler radars which determine a target's velocity from the Doppler frequency shift.

Let's see how to build a filter to optimally remove noise and recover a signal. Our strategy is straightforward. It is simple to recover a sufficiently strong signal in the presence of sufficiently weak noise (i.e., when the SNR is sufficiently high). When the SNR is low we will design a filter to enhance it; such a filter's design must take into account everything known about the signal and the noise spectra.

Before starting we need some notation. For simplicity we observe the spectrum from DC to some frequency  $F$ . We will denote the original analog signal in time as  $s(t)$  and in frequency as  $S(f)$ . We will call its total energy  $E_s$ . We denote the same quantities for the additive noise,  $\nu(t)$ ,  $\mathcal{V}(f)$ , and  $E_\nu$ , respectively. These quantities are obviously related by

$$\begin{aligned} E_s &= \int_{-\infty}^{\infty} |s(t)|^2 dt = \int_{-\infty}^{\infty} |S(f)|^2 df \\ E_\nu &= \int_{-\infty}^{\infty} |\nu(t)|^2 dt = \int_{-\infty}^{\infty} |\mathcal{V}(f)|^2 df \end{aligned}$$

and if the noise is white then we further define its constant power spectral density to be  $\mathcal{V}_0 = \frac{E_\nu}{F}$  watt per Hz. The overall signal-to-noise ratio is the ratio of the energies

$$SNR = \frac{E_s}{E_\nu} \quad (9.12)$$

but we can define time- and frequency-dependent SNRs as well.

$$SNR(t) = \frac{|s(t)|^2}{|\nu(t)|^2} \quad SNR(f) = \frac{|S(f)|^2}{|\mathcal{V}(f)|^2} \quad (9.13)$$

Finally, the observed signal is the sum of the signal plus the noise.

$$x(t) = s(t) + \nu(t) \quad X(f) = S(f) + \mathcal{V}(f) \quad (9.14)$$

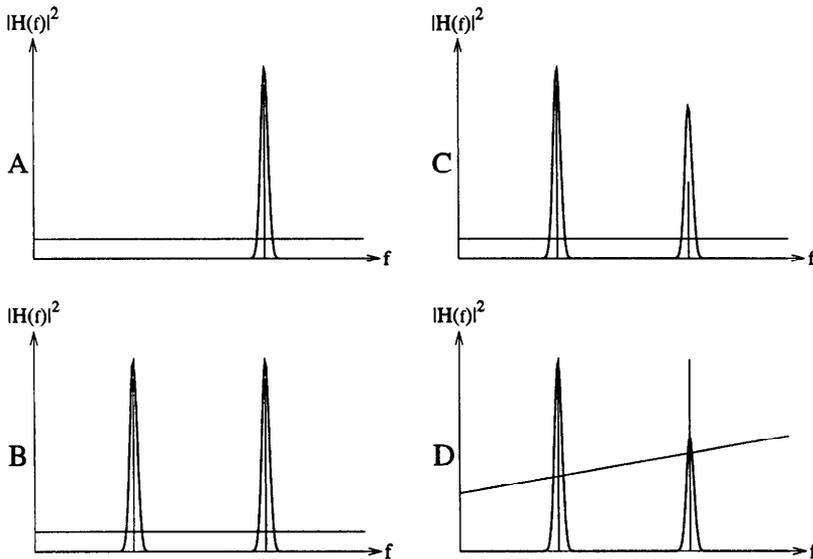
We'll start with the simple case of a relatively pure sinusoid of frequency  $f_0$  in white noise. The signal PSD consists of a single narrow line (and its negative frequency conjugate), while the noise PSD is a constant  $\mathcal{V}_0$ ; accordingly the SNR is  $\frac{E_s}{\mathcal{V}_0}$ . What filter will optimally detect this signal given this noise? Looking at the frequency-dependent SNR we see that the signal stands out above the noise at  $f_0$ ; so it makes sense to use a narrow band-pass filter centered on the sinusoid's frequency  $f_0$ . The narrower the filter bandwidth  $BW$ , the less noise energy is picked up, so we want  $BW$  to be as small as possible. The situation is depicted in Figure 9.3.A where we see the signal PSD represented as a single vertical line, the noise as a horizontal line, and the optimum filter as the smooth curve peaked around the signal. The signal-to-noise ratio at the output of the filter

$$\begin{aligned} SNR_{out} &= \frac{\int_{-\infty}^{\infty} |H(f)S(f)|^2 df}{\int_{-\infty}^{\infty} |H(f)\mathcal{V}(f)|^2 df} \quad (9.15) \\ &= \frac{\int_{f_0 - \frac{BW}{2}}^{f_0 + \frac{BW}{2}} |H(f)S(f)|^2 df}{\int_{f_0 - \frac{BW}{2}}^{f_0 + \frac{BW}{2}} |H(f)\mathcal{V}(f)|^2 df} = \frac{E_s}{\mathcal{V}_0 BW} \end{aligned}$$

is greater than that at the input by a factor of  $\frac{F}{BW}$ . For small  $BW$  this is a great improvement in SNR and allows us to detect the reference signal even when buried in very high noise levels.

Now let's complicate matters a bit by considering a signal with two equal spectral components, as in Figure 9.3.B. Should we use a filter that captures both spectral lines or be content with observing only one of them? The two-component filter will pass twice the signal energy but twice the noise energy as well. However, a filter that matches the signal spectrum may enhance the time-dependent SNR; the two signal components will add constructively at some time, and by choosing the relative phases of the filter components we can make this peak occur whenever we want. Also, for finite times the noise spectrum will have local fluctuations that may cause a false alarm in a single filter, but the probability of that happening simultaneously in both filters is much smaller. Finally, the two-component filter can differentiate better between the desired signal and a single frequency sinusoid masquerading as the desired signal.

Were one of the frequency components to be more prominent than the other, we would have to compensate by having the filter response  $H(f)$  as



**Figure 9.3:** Expected behavior of an optimum filter in the frequency domain. In all the figures we see the PSD of the reference signal and noise, as well as the Wiener filter. The various cases are discussed in the text.

depicted in Figure 9.3.C. This seems like the right thing to do, since such a filter emphasizes frequencies with high SNR. Likewise Figure 9.3.D depicts what we expect the optimal filter to look like for the case of two equal signal components, but non-white noise.

How do we actually construct this optimum filter? It's easier than it looks. From equation (9.14) the spectrum at the filter input is  $S(f) + \mathcal{V}(f)$ , so the filter's frequency response must be

$$H(f) = \frac{S(f)}{S(f) + \mathcal{V}(f)} \quad (9.16)$$

in order for the desired spectrum  $S(f)$  to appear at its output. This frequency response was depicted in Figure 9.3. Note that we can think of this filter as being built of two parts: the denominator corresponds to a whitening filter, while the numerator is matched to the signal's spectrum. Unlike the whitening filter that we met in the matched filter detector, here the entire signal plus noise must be whitened, not just the noise.

This filter is a special case of the *Wiener filter* derived by Norbert Wiener during World War II for optimal detection of radar signals. It is a special case because we have been implicitly assuming that the noise and signal are

uncorrelated. When the noise can be correlated to the signal we have to be more careful.

This is not the first time we have attempted to find an unknown FIR filter. In Section 6.13 we found that the hard system identification problem for FIR filters was solved by the Wiener-Hopf equations (6.63). At first it seems that the two problems have nothing in common, since in the Wiener filter problem only the input is available, the output being completely unknown (otherwise we wouldn't need the filter), while in the system identification case both the input and output were available for measurement! However, neither of these statements is quite true. Were the output of the Wiener filter completely unspecified the trivial filter that passes the input straight through would be a legitimate solution. We do know certain characteristics of the desired output, namely its spectral density or correlations. In the hard system identification problem we indeed posited that we intimately knew the input and output signals, but the solution does not exploit this much detail. Recall that only the correlations were required to find the unknown system.

So let's capitalize on our previous results. In our present notation the input is  $x_n = s_n + \nu_n$  and the desired output  $s_n$ . We can immediately state the Wiener-Hopf equations in the time domain

$$C_{sx}(q) = \sum_k h_k C_x(q - k)$$

so that given  $C_{sx}$  and  $C_x$  we can solve for  $h$ , the Wiener filter in the time domain. To compare this filter with our previous results we need to transfer the equations to the frequency domain, using equation (4.47) for the FT of a convolution.

$$P_{sx}(\omega) = H(\omega)P_x(\omega)$$

Here  $P_{sx}(\omega)$  is the FT of the crosscorrelation between  $s(t)$  and  $x(t)$ , and  $P_x(\omega)$  is the PSD of  $x(t)$  (i.e., FT of its autocorrelation). Dividing we find the full Wiener filter.

$$H(\omega) = \frac{P_{sx}(\omega)}{P_x(\omega)} \quad (9.17)$$

For uncorrelated noise  $P_{sx}(\omega) = P_s(\omega)$  and  $P_x(\omega) = P_s(\omega) + P_n(\omega)$  and so the full Wiener filter reduces to equation (9.16).

The Wiener filter only functions when the signals being treated are stationary (i.e.,  $P_{sx}$  and  $P_s$  are not functions of time). This restriction too can be lifted, resulting in the *Kalman filter*, but any attempt at explaining its principles would lead us too far astray.

## EXERCISES

- 9.7.1 Assume that the signal  $s(t)$  has constant PSD in some range but the noise  $\nu(t)$  is narrow-band. Explain why we expect a Wiener filter to have a notch at the disturbing frequency.
- 9.7.2 An alternative to the SNR is the ‘signal-plus-noise-to-noise-ratio’ S+NNR. Why is this ratio of importance? What is the relationship between the overall S+NNR and SNR? What is the relationship between the Wiener filter and the frequency-dependent S+NNR and SNR?

## 9.8 Correlation and Prediction

A common problem in DSP is to predict the next signal value  $s_n$  based on the values we have observed so far. If  $s_n$  represents the closing value of a particular stock on day  $n$  the importance of accurate prediction is obvious. Less obvious is the importance of predicting the next value of a speech signal. It’s not that I impolitely do not wish to wait for you to finish whatever you have to say; rather the ability to predict the next sample enables the *compression* of digitized speech, as will be discussed at length in Chapter 19. Any ability to predict the future implies that less information needs to be transferred or stored in order to completely specify the signal.

If the signal  $s$  is white noise then there is no correlation between its value  $s_n$  and its previous history (i.e.,  $C_s(m) = 0 \quad \forall m \neq 0$ ), and hence no prediction can improve on a guess based on single sample statistics. However, when the autocorrelation is nontrivial we can use past values to improve our predictions. So there is a direct connection between correlation and prediction; we can exploit the autocorrelation to predict what the signal will most probably do.

The connection between correlation and prediction is not limited to autocorrelation. If two signals  $x$  and  $y$  have a nontrivial crosscorrelation this can be exploited to help predict  $y_n$  given  $x_n$ . More generally, the causal prediction of  $y_n$  could depend on previous  $y$  values,  $x_n$ , and previous  $x$  values. An obvious example is when the crosscorrelation has a noticeable peak at lag  $m$ , and much information about  $y_n$  can be gleaned from  $x_{n-m}$ .

We can further clarify the connection between autocorrelation and signal prediction with a simple example. Assume that the present signal value  $s_n$  depends strongly on the previous value  $s_{n-1}$  but only weakly on older values. We further assume that this dependence is *linear*,  $s_n \approx b s_{n-1}$  (were we to

take  $s_n \approx b s_{n-1} + c$  we would be forced to conclude  $c = 0$  since otherwise the signal would diverge after enough time). Now we are left with the problem of finding  $b$  given an observed signal. Even if our assumptions are not very good, that is, even if  $s_n$  *does* depend on still earlier values, and/or the dependence is not really linear, and even if  $s_n$  depends on other signals as well, we are still interested in finding that  $b$  that gives the best linear prediction given only the previous value.

$$\tilde{s}_n = b s_{n-1} \quad (9.18)$$

What do we mean by *best* prediction? The best definition of *best* is for the Mean Squared Error (MSE)

$$d_n^2 = (s_n - \tilde{s}_n)^2 = (s_n - b s_{n-1})^2 = s_n^2 - 2b s_n s_{n-1} + b^2 s_{n-1}^2$$

to be as small as possible, on the average. We are now in familiar territory. Assuming the signal to be time-invariant we average over all time

$$\langle d_n^2 \rangle = \langle s_n^2 \rangle - 2b \langle s_n s_{n-1} \rangle + b^2 \langle s_{n-1}^2 \rangle = (1 + b^2) C_s(0) - 2b C_s(1)$$

and then differentiate and set equal to zero. We find that the optimal linear prediction is

$$b = \frac{C_s(1)}{C_s(0)} = c_s(1) \quad (9.19)$$

the normalized autocorrelation coefficient for lag 1. Substituting this back into the expression for the average square error, we find

$$\langle d_n^2 \rangle = \frac{C_s^2(0) - C_s^2(1)}{C_s(0)} \quad (9.20)$$

so that the error vanishes when the lag 1 correlation equals the energy.

## EXERCISES

- 9.8.1 Wiener named his book *The Extrapolation, Interpolation and Smoothing of Stationary Time Series with Engineering Applications*. Wiener's 'extrapolation' is what we have called 'prediction'. What did he mean by 'interpolation' and 'smoothing'?
- 9.8.2 Find the optimal linear prediction coefficients when two lags are taken into account.

$$\tilde{s}_n = b_1 s_{n-1} + b_2 s_{n-2}$$

## 9.9 Linear Predictive Coding

Signal coding, that is, compression of the amount of information needed to represent a signal, is an important application of DSP. To see why, consider the important application of digital speech. A bandwidth of 4 KHz is required so we must sample at 8000 samples per second; with 16 bit samples this requires 128 Kb/s, or just under 1 MB of data every minute. This data rate cannot be transferred over a telephone connection using a modem (the fastest telephone-grade modems reach 56 Kb/s) and would even be a tremendous strain on storage facilities. Yet modern speech compression techniques (see Chapter 19) can reduce the required rate to 8 Kb/s or less with only barely noticeable quality degradation.

Let's call the signal to be compressed  $s_n$ . If  $s$  is not white noise then it is at least partially linearly predictable based on its  $M$  previous values.

$$s_n = Ge_n + \sum_{m=1}^M b_m s_{n-m} \quad (9.21)$$

Here  $e_n$  is the portion of the signal not predictable based on the signal's own history,  $G$  is an arbitrarily introduced gain, and  $b_m$  are called the Linear Predictive Coding (LPC) coefficients. Note that most people use  $a$  for these coefficients, but we reserve  $a$  for FIR coefficients; some people use a minus sign before the sum (i.e., use what we call  $\beta$  coefficients).

Equation (9.21) has a simple interpretation; the signal  $s_n$  is obtained by filtering the unpredictable signal  $e_n$  by a all-pole filter with gain  $G$  and coefficients  $b_m$ . The  $e_n$  is called the 'excitation' signal since it 'excites' the filter into operation. Since the filter is all-pole it enhances certain excited frequencies; these amplified frequencies are responsible for the non-flat spectrum and nontrivial autocorrelation of predictable signals. For speech signals (see Section 11.3) the excitation  $e_n$  is the 'glottal excitation'; for voiced speech (e.g., vowels) this is a periodic set of pulses created by the vocal chords, while for unvoiced speech (e.g., **h**) it is a noise-like signal created by constricting the passage of air. For both cases the mouth and nasal cavities act as a filter, enhancing frequencies according to their geometry.

In order to compress the signal we need an algorithm for finding the  $M + 1$  parameters  $G$  and  $b_m$  given a buffer of  $N$  samples of the signal  $\{s_n\}_{n=0}^{N-1}$ . Looking carefully at equation (9.21) we note a problem. There are too many unknowns. In order to uniquely determine the coefficients  $b_m$  we need to know both the observed speech signal  $s_n$  and the excitation  $e_n$ . Unfortunately, the latter signal is usually inaccessible; for speech signals

obtaining it would require swallowing a microphone so that it would be close to the vocal chords and before the vocal tract. We thus venture forth under the assumption that the excitation is identically zero. This is true most of the time for a pulse train excitation, only erring for those time instants when the impulse appears. It is obviously not a good approximation for many other cases.

Under the assumption of zero excitation we get the homogeneous recursion

$$s_n = \sum_{m=1}^M b_m s_{n-m} \quad (9.22)$$

for which  $s = 0$  (the zero signal) is a solution. It is the only solution if the excitation was truly always zero; but due to the IIR nature of the filter, other possibilities exist if the excitation was once nonzero, even if zero during the duration of the present buffer. For speech the excitation is not truly zero, so even when we find the coefficients  $b_m$  we can only approximately predict the next signal value.

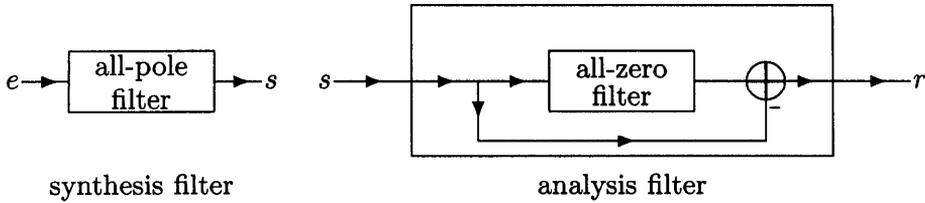
$$\tilde{s}_n = \sum_{m=1}^M b_m s_{n-m} \quad (9.23)$$

The error of this approximation is called the *residual signal*

$$r_n = s_n - \tilde{s}_n = s_n - \sum_{m=1}^M b_m s_{n-m} = \sum_{m=0}^M \beta_m s_{n-m} \quad (9.24)$$

(where  $\beta_0 \equiv 1$  and  $\beta_m = -b_m$ ), and the correct LPC coefficients minimize this residual. Note that the residual is obtained by FIR filtering the input signal, with the filter coefficients being precisely  $\beta_m$ . This all-zero filter is usually called the 'LPC analysis filter' and it is the inverse filter of the 'LPC synthesis filter' that synthesizes the speech from the excitation (see Figure 9.4). The analysis filter is also called the 'LPC whitening filter', the residual being much whiter than the original speech signal, since the linear predictability has been removed.

There is another way of looking at the residual signal. Rather than taking no excitation and treating the residual as an error signal, we can pretend that there *is* excitation but take the error to be precisely zero. What must the excitation be for  $\tilde{s}_n$  to be the correct signal value? Comparing equations (9.24) and (9.21) we see that  $r_n = Ge_n$ , the residual is simply the excitation amplified by the gain. Thus when analyzing voiced speech we see that the residual is usually small but displays peaks corresponding to the vocal chord pulses.



**Figure 9.4:** LPC synthesis and analysis filters. The synthesis filter synthesizes the signal  $s_n$  from the excitation  $e_n$ , while the analysis filter analyzes incoming signal  $s_n$  and outputs the residual error signal  $r_n$ . The synthesis and analysis filters are inverse systems to within a gain.

One final remark regarding the residual. In speech compression terminology the residual we defined is called the *open-loop* residual. It can be calculated only if the original speech samples  $s_n$  are available. When decompressing previously compressed speech these samples are no longer available, and we can only attempt to predict the present signal value based on past *predicted values*. It is then better to define the *closed-loop* residual

$$r_n^c = s_n + \sum_{m=1}^M b_m \tilde{s}_{n-m}$$

and minimize *it* instead.

Returning to our mission, we wish to find coefficients  $b_m$  that minimize the residual of equation (9.24). In order to simultaneously minimize the residual  $r_n$  for all times of interest  $n$ , we calculate the MSE

$$E = \sum_n r_n^2 = \sum_n \left( s_n - \sum_{m=1}^M b_m s_{n-m} \right)^2 \quad (9.25)$$

and minimize it with respect to the  $b_m$  ( $m = 1 \dots M$ ). This minimization is carried out by setting all  $M$  partial derivatives equal to zero

$$\frac{\partial E}{\partial b_m} = 0$$

which leads us to the following set of  $M$  equations.

$$\sum_n \left( \sum_{l=1}^M b_l s_{n-m} s_{n-l} - s_n s_{n-m} \right) = 0 \quad (9.26)$$

Moving the sum on  $n$  inside we can rewrite these

$$\sum_{l=1}^M b_l \sum_n s_{n-m} s_{n-l} = \sum_n s_n s_{n-m} \quad (9.27)$$

in which the signal enters only via autocorrelations  $C_s$ .

$$\sum_{l=1}^M C_s(|m-l|) b_l = C_s(m) \quad (9.28)$$

These are, of course, the Yule-Walker equations for the LPC coefficients.

The sum in the autocorrelations *should* run over all times  $n$ . This is problematic for two reasons. First, we are usually only given an input signal buffer of length  $N$ , and even if we are willing to look at speech samples outside this buffer, we cannot wait forever. Second, many signals including speech are stationary only for short time durations, and it is only sensible to compute autocorrelations over such durations. Thus we must somehow limit the range of times taken into account in the autocorrelation sums. This can be done in two ways. The brute-force way is to artificially take all signal values outside the buffer to be zero for the purposes of the sums. A somewhat more gentle variant of the same approach uses a window function (see Section 13.4) that smoothly reduces the signal to zero. The second way is to retain the required values from the previous buffer. The first way is called the autocorrelation method and is by far the most popular; the second is called the covariance method and is less popular due to potential numerical stability problems.

The autocorrelation method allows the sum in the MSE to be over all times, but takes all signal values outside the buffer  $s_0 \dots s_{N-1}$  to be zero. Since the error  $e_n$  in equation (9.24) depends on  $M+1$  signal values, it can only be nonzero for  $n = 0 \dots N+M-1$ . Accordingly, the MSE is

$$E = \sum_{n=0}^{N+M-1} r_n^2$$

and the correlations appearing in it have these limits.

$$C_s(m, l) \equiv \sum_{n=0}^{N+M-1} s_{n-m} s_{n-l} = C_s(|m-l|)$$

Writing the Yule-Walker equations in matrix notation

$$\begin{pmatrix} C_s(0) & C_s(1) & \dots & C_s(M-1) \\ C_s(1) & C_s(0) & \dots & C_s(M-2) \\ C_s(2) & C_s(1) & \dots & C_s(M-3) \\ \vdots & \vdots & \vdots & \vdots \\ C_s(M-1) & C_s(M-2) & \dots & C_s(0) \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_M \end{pmatrix} = \begin{pmatrix} C_s(1) \\ C_s(2) \\ C_s(3) \\ \vdots \\ C_s(M) \end{pmatrix} \quad (9.29)$$

we see that the matrix is symmetric and Toeplitz. In the next section we will study a fast method for solving such equations.

The MSE in the covariance method is taken to be

$$E = \sum_{n=0}^{N-1} r_n^2$$

and here we don't assume that the signal was zero for  $n < 0$ . We must thus access  $N + M$  signal values, including  $M$  values from the previous buffer. Equations (9.27) are still correct, but now the sums over  $n$  no longer lead to genuine autocorrelations due to the limits of the sums being constrained differently.

$$C_s(m, l) \equiv \sum_{n=0}^{N-1} s_{n-m} s_{n-l} = C_s(l, m)$$

In particular  $C_s$  although symmetric is no longer a function of  $|l - m|$ , but rather a function of  $l$  and  $m$  separately. Writing these equations in matrix form we get a matrix that is symmetric but not Toeplitz.

$$\begin{pmatrix} C_s(1, 1) & C_s(1, 2) & \dots & C_s(1, M) \\ C_s(1, 2) & C_s(2, 2) & \dots & C_s(2, M) \\ C_s(1, 3) & C_s(2, 3) & \dots & C_s(3, M) \\ \vdots & \vdots & \vdots & \vdots \\ C_s(1, M) & C_s(2, M) & \dots & C_s(M, M) \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_M \end{pmatrix} = \begin{pmatrix} C_s(0, 1) \\ C_s(0, 2) \\ C_s(0, 3) \\ \vdots \\ C_s(0, M) \end{pmatrix} \quad (9.30)$$

The fast methods of solving Toeplitz equations are no longer available, and the Cholesky decomposition (equation (A.94)) is usually employed.

Since general covariance matrices are of this form this method is called the covariance method, although no covariances are obviously present. For  $N \gg M$  the difference between using  $N$  samples and using  $N + M$  samples becomes insignificant, and the two methods converge to the same solution. For small buffers the LPC equations can be highly sensitive to the boundary conditions and the two methods may produce quite different results.

## EXERCISES

- 9.9.1 What is the approximation error for the covariance method?
- 9.9.2 Equation (9.22) predicts  $s_n$  based on  $M$  previous values  $s_{n-1}, s_{n-2}, \dots, s_{n-M}$  and is called the *forward predictor*. We can also ‘predict’ (postdict?)  $s_{n-M}$  based on the next  $M$  values  $s_{n-M+1}, \dots, s_{n-1}, s_n$ . This surprising twist on LPC is called backward linear prediction. Modify equation (9.22) for this case (call the coefficients  $c_m$ ). What is the residual?
- 9.9.3 Show that the MSE error can be written  $E = \sum_n s_n^2 + \sum_{m=1}^M b_m \sum_n s_n s_{n-m}$  and thus for the autocorrelation method  $E = C_s(0) + \sum_{m=1}^M b_m C_s(m)$ .
- 9.9.4 Show that assuming the input to be an impulse  $G\delta_{n,0}$  the gain is given by the error as given in the previous exercise.
- 9.9.5 Use the LPC method to predict the next term in the sequence  $1, \alpha, \alpha^2, \alpha^3, \dots$  for various  $0 < \alpha < 1$ . Repeat for  $\alpha > 1$ . Does the LPC method always correctly predict the next signal value?

## 9.10 The Levinson-Durbin Recursion

Take an empty glass soft-drink bottle and blow over its mouth. Now put a little water in the bottle and blow again. The frequency produced is higher since the wavelength that resonates in the cavity is shorter (recall our discussion of wavelength in Section 7.9). By tuning a collection of bottles you can create a musical instrument and play recognizable tunes.

The bottle in this experiment acts as a filter that is excited by breath noise. Modeling the bottle as a simple cylinder, the frequency it enhances is uniquely determined by its height. What if we want to create a signal containing two different frequencies? One way would be to blow over two different bottles separately (i.e., to place the filters in parallel). From our studies of filters we suspect that there may be a way of putting the filters in series (cascade) as well, but putting two cylinders one after the other only makes a single long cylinder. In order to get multiple frequencies we can use cylinders of different cross-sectional areas, the resonant frequencies being determined by the widths rather than the heights.

If we send a sound wave down a pipe that consists of a sequence of cylinders of different cross-sectional areas  $A_i$ , at each interface a certain amount of acoustic energy continues to travel down the pipe while some is reflected back toward its beginning. Let's send a sinusoidal acoustic wave

down a pipe consisting of two cylinders. Recalling from Section 7.9 that traveling waves are functions of  $s - vt$ , we can express the incoming wave for this one directional case as follows.

$$\psi_{inc}^{[1]}(x, t) = A \sin(x - vt) \quad (9.31)$$

The reflected wave in the first cylinder will be sinusoid of the same frequency but traveling in the opposite direction and reduced in amplitude

$$\psi_{ref}^{[1]}(x, t) = kA \sin(x + vt) \quad (9.32)$$

where the reflection coefficient  $k$  is the fraction of the wave that is reflected. Since the energy is proportional to the signal squared, the fraction of the wave's energy that is reflected is  $k^2$ , while the wave energy that continues on to the second cylinder is whatever remains.

$$E_2 = (1 - k^2)E_1 \quad (9.33)$$

Now for a little physics. The  $\psi$  for sound waves can represent many different physical quantities (e.g., the average air particle displacement, the air particle velocity, the pressure). We'll assume here that it represents the velocity. Physically this velocity must be continuous across the interface between the two sections, so at the interface the following must hold.

$$\psi_{inc}^{[1]}(x, t) - \psi_{ref}^{[1]}(x, t) = \psi^{[2]}(x, t)$$

The derivative of the velocity is the acceleration, which is proportional to the force exerted on the air particles. The pressure, defined as the force per unit area, must be continuous at the interface, implying that the following must hold there.

$$\frac{\dot{\psi}_{inc}^{[1]}(x, t) - \dot{\psi}_{ref}^{[1]}(x, t)}{A_1} = \frac{\dot{\psi}^{[2]}(x, t)}{A_2}$$

Combining these two equations results in

$$\frac{1 + k}{A_1} = \frac{1 - k}{A_2}$$

and rearranging we find an expression for the reflection coefficient in terms of the cross-sectional areas.

$$k = \frac{A_1 - A_2}{A_1 + A_2}$$

Let's check to see that this result is reasonable. If the second cylinder shrinks to zero area (closing off the pipe) then  $k = 1$  and the wave is entirely reflected, as it should be. If there really is no interface at all (i.e.,  $A_1 = A_2$ ) then  $k = 0$  and no energy is reflected. If  $A_2 \gg A_1$  then  $k = -1$ , which seems unreasonable at first; but an open-ended pipe has zero pressure at its end, and so the wave reflects but with a phase reversal.

It isn't hard to generalize our last result to a pipe with many sections. The reflection coefficient at the interface between section  $i$  and section  $i + 1$  is

$$k_i = \frac{A_i - A_{i+1}}{A_i + A_{i+1}} \quad (9.34)$$

What does all this have to do with solving the Yule-Walker equations for the LPC coefficients in the autocorrelation method? The LPC coefficients  $b_m$  are not the only way of describing an all-pole system; the area ratios, the reflection coefficients, and many others (including an interesting set to be discussed in the next section) can be used instead. Since all of these parameter sets contain exactly the same information, it follows that we can derive any set from any other set. Many of the parameter sets are related by linear transformations, and hence the conversion is equivalent to multiplying by a matrix. We will now show that the connection between the reflection and LPC coefficients can be expressed as a recursion that is the most efficient way of deriving both.

How can equation (9.29) be solved recursively? For simplicity we'll drop the subscript identifying the signal, but we have to add superscripts identifying the recursion depth. The first case is simple (for further simplicity we have dropped the subscript)

$$C(0)b_1^{[1]} = C(1) \quad \longrightarrow \quad b_1^{[1]} = \frac{C(1)}{C(0)}$$

and its MSE is

$$E_1 = C(0) - b_1^{[1]}C(1) = C(0)(1 - k_1^2)$$

where we have defined  $k_1 \equiv b_1^{[1]}$ . Let's assume we have already solved the  $m^{\text{th}}$  case

$$\begin{pmatrix} C(0) & C(1) & \dots & C(m-1) \\ C(1) & C(0) & \dots & C(m-2) \\ \vdots & \vdots & \ddots & \vdots \\ C(m-1) & C(m-2) & \dots & C(0) \end{pmatrix} \begin{pmatrix} b_1^{[m]} \\ b_2^{[m]} \\ \vdots \\ b_m^{[m]} \end{pmatrix} = \begin{pmatrix} C(1) \\ C(2) \\ \vdots \\ C(m) \end{pmatrix}$$

and let's write this  $\underline{\underline{C}}^{[m]} \underline{\underline{b}}^{[m]} = \underline{\underline{c}}^{[m]}$ . We are now interested in the  $(m + 1)^{\text{th}}$  case

$$\left( \begin{array}{cccc|c} C(0) & C(1) & \dots & C(m-1) & C(m) \\ C(1) & C(0) & \dots & C(m-2) & C(m-1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ C(m-1) & C(m-2) & \dots & C(0) & C(1) \\ \hline C(m) & C(m-1) & \dots & C(1) & C(0) \end{array} \right) \begin{pmatrix} b_1^{[m+1]} \\ b_2^{[m+1]} \\ \vdots \\ b_m^{[m+1]} \\ b_{m+1}^{[m+1]} \end{pmatrix} = \begin{pmatrix} C(1) \\ C(2) \\ \vdots \\ C(m) \\ C(m+1) \end{pmatrix}$$

where we have drawn in delimiters that divide the equations into two parts:

$$\begin{pmatrix} C(0) & C(1) & \dots & C(m-1) \\ C(1) & C(0) & \dots & C(m-2) \\ \vdots & \vdots & \vdots & \vdots \\ C(m-1) & C(m-2) & \dots & C(0) \end{pmatrix} \begin{pmatrix} b_1^{[m+1]} \\ b_2^{[m+1]} \\ \vdots \\ b_m^{[m+1]} \end{pmatrix} + b_{m+1}^{[m+1]} \begin{pmatrix} C(m) \\ C(m-1) \\ \vdots \\ C(1) \end{pmatrix} = \begin{pmatrix} C(1) \\ C(2) \\ \vdots \\ C(m) \end{pmatrix} \tag{9.35}$$

and

$$\begin{pmatrix} C(m) & C(m-1) & \dots & C(1) & C(0) \end{pmatrix} \begin{pmatrix} b_1^{[m+1]} \\ b_2^{[m+1]} \\ \vdots \\ b_m^{[m+1]} \\ b_{m+1}^{[m+1]} \end{pmatrix} = C(m+1) \tag{9.36}$$

Now multiply equation (9.35) by the inverse of the autocorrelation matrix of the  $m^{\text{th}}$  iteration  $(\underline{\underline{C}}^{[m]})^{-1}$  and use the results of that iteration.

$$\begin{pmatrix} b_1^{[m+1]} \\ b_2^{[m+1]} \\ \vdots \\ b_m^{[m+1]} \end{pmatrix} + b_{m+1}^{[m+1]} (\underline{\underline{C}}^{[m]})^{-1} \begin{pmatrix} C(m) \\ C(m-1) \\ \vdots \\ C(1) \end{pmatrix} = (\underline{\underline{C}}^{[m]})^{-1} \begin{pmatrix} C(1) \\ C(2) \\ \vdots \\ C(m) \end{pmatrix}$$

Defining  $\underline{\underline{J}}$  as the matrix that reverses row order

$$\underline{\underline{J}} = \begin{pmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \end{pmatrix}$$

and noting that it commutes with Toeplitz matrices, we can finally write the following recursion for the LPC coefficients

$$\begin{pmatrix} b_1^{[m+1]} \\ b_2^{[m+1]} \\ \vdots \\ b_m^{[m+1]} \end{pmatrix} = (\underline{\underline{I}} - k_{m+1}\underline{\underline{J}}) \begin{pmatrix} b_1^{[m]} \\ b_2^{[m]} \\ \vdots \\ b_m^{[m]} \end{pmatrix} \quad (9.37)$$

where  $k_m \equiv b_m^{[m]}$ .

In the statistics literature the  $k$  variables are called 'partial correlation' or PARCOR coefficients, since they can be shown to measure the correlation between the forward and backward prediction errors (see exercise 9.9.2). Later we will show that they are exactly the reflection coefficients.

Were we to know  $k_{m+1}$  this recursion would produce all the other new  $b^{[m+1]}$  given the old  $b^{[m]}$ . So we have reduced the problem of finding the LPC coefficients to the problem of finding the PARCOR coefficients. Yet it is obvious from equation (9.36) that the converse is also true,  $k_{[m+1]}$  can be derived from the lower  $b^{[m+1]}$  coefficients. So let's derive a recursion for the  $k$ s and try to eliminate the  $b$ s.

First we rewrite equation (9.36) as

$$\begin{pmatrix} C(1) & C(2) & \dots & C(m) \end{pmatrix} \underline{\underline{J}} \begin{pmatrix} b_1^{[m+1]} \\ b_2^{[m+1]} \\ \vdots \\ b_m^{[m+1]} \end{pmatrix} + k_{m+1}C(0) = C_{(m+1)}$$

which can be written (with obvious notation) as follows.

$$\underline{\underline{c}} \cdot \underline{\underline{J}} \underline{\underline{b}}^{[m+1]} + k_{m+1}C(0) = C(m+1)$$

Now we substitute the  $b^{[m+1]}$  vector from equation (9.37)

$$\underline{c} \cdot \underline{J}(\underline{I} - k_{m+1}\underline{J})\underline{b}^{[m]} + k_{m+1}C(0) = C(m+1)$$

and finally solve for  $k_{m+1}$  (noting  $\underline{J}^2 = \underline{I}$ )

$$\begin{aligned} k_{m+1} &= \frac{C(m+1) - \underline{c} \cdot \underline{J}\underline{b}^{[m]}}{C(0) - \underline{c} \cdot \underline{b}^{[m]}} \\ &= \frac{C(m+1) - \underline{c} \cdot \underline{J}\underline{b}^{[m]}}{E_m} \end{aligned}$$

identifying the MSE in the denominator. After following all the above the reader will have no problem proving that the MSE obeys the simplest recursion of all.

$$E_{m+1} = (1 - k_{m+1}^2)E_m \quad (9.38)$$

Let's now group together all the recursive equations into one algorithm that computes the  $k$  and  $b$  coefficients for successively higher orders until we reach the desired order  $M$ .

Given the signal autocorrelations  $C(0)$  through  $C(M)$

Start with  $E_0 = C(0)$

for  $m \leftarrow 1$  to  $M$

$$b^{[m]} \leftarrow k_m = \frac{C(m) - \sum_{\mu=1}^{m-1} C(m-\mu)b_{\mu}^{[m-1]}}{E_{m-1}}$$

for  $\mu = m-1$  down to 1

$$b_{\mu}^{[m]} \leftarrow b_{\mu}^{[m-1]} - k_m b_{m-\mu}^{[m-1]}$$

$$E_m \leftarrow (1 - k_m^2)E_{m-1}$$

for  $\mu \leftarrow 1$  to  $M$

$$b_{\mu} \leftarrow b_{\mu}^{[M]}$$

To see how the algorithm works let's run through it for the case of two coefficients.

$$\begin{pmatrix} C(0) & C(1) \\ C(1) & C(0) \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} C(1) \\ C(2) \end{pmatrix}$$

The first iteration is easy.

$$\begin{aligned} E_0 &= C(0) \\ b_1^{[1]} &= k_1 = \frac{C(1)}{C(0)} \end{aligned}$$

Now we can perform the second iteration

$$\begin{aligned}
 E_1 &= \frac{C^2(0) - C^2(1)}{C(0)} \\
 b_2^{[2]} &= k_2 = \frac{C(2)C(0) - C^2(1)}{C^2(0) - C^2(1)} \\
 b_1^{[2]} &= b_1^{[1]} - k_2 = \frac{C(1)C(0) - C(1)C(2)}{C^2(0) - C^2(1)}
 \end{aligned}$$

and we have found the desired coefficients  $b_1$  and  $b_2$ .

We finish the section by fulfilling our promise to show that the  $k$  are the reflection coefficients. If we implement the LPC analysis filter (the FIR filter that converts the signal into the residual as a multistage lattice filter) then equation (9.38) tells us how the energy of the residual decreases. Comparing this with equation (9.33) completes the identification.

## EXERCISES

- 9.10.1 Prove equation (9.34) for a pipe with multiple sections taking into account the reflected wave from the next interface.
- 9.10.2 Transmission lines have both voltage and current traveling waves, the ratio between the voltage and current being the impedance  $Z$ . At a splice where the impedance changes a reflected wave is generated. Express the reflection coefficient in terms of the impedances. Explain the limiting cases of shorted and open circuited cables.
- 9.10.3 Prove equation (9.38) for the MSE.
- 9.10.4 Solve the three-coefficient problem on paper using the Levinson-Durbin recursion.
- 9.10.5 Show that the complexity of the Levinson-Durbin algorithm is  $O(M^2)$  rather than  $O(M^3)$  as for non-Toeplitz systems.
- 9.10.6 Levinson originally solved the more general problem of solving the equations  $\underline{\underline{T}}x = \underline{y}$  where  $\underline{\underline{T}}$  is Toeplitz but unrelated to  $\underline{y}$ . Generalize the recursion to solve this problem. (Hint: You will need another set of recursions.) How much more computationally complex is the solution?

## 9.11 Line Spectral Pairs

Another set of parameters that contain exactly the same information as the LPC coefficients are the **Line Spectral Pair (LSP)** frequencies. To introduce them we need to learn a mathematical trick that can be performed on the polynomial in the denominator of the LPC system function.

A polynomial of degree  $M$

$$a(x) = \sum_{m=0}^M p_m x^m = a_0 + a_1 x + a_2 x^2 + \dots + a_{M-2} x^{M-2} + a_{M-1} x^{M-1} + a_M x^M$$

is called 'palindromic' if  $a_m = a_{M-m}$ , i.e.,

$$a_0 = a_M \quad a_1 = a_{M-1} \quad a_2 = a_{M-2} \quad \text{etc.}$$

and 'antipalindromic' if  $a_m = -a_{M-m}$ , i.e.,

$$a_0 = -a_M \quad a_1 = -a_{M-1} \quad a_2 = -a_{M-2} \quad \text{etc.}$$

so  $1 + 2x + x^2$  is palindromic, while  $x + x^2 - x^3$  is antipalindromic. It is not hard to show that the product of two palindromic or two antipalindromic polynomials is palindromic, while the product of an antipalindromic polynomial with a palindromic one is antipalindromic.

We will now prove that every real polynomial that has all of its zeros on the unit circle is either palindromic or antipalindromic. The simplest cases are  $x+1$  and  $x-1$ , which are obviously palindromic and antipalindromic, respectively. Next consider a second degree polynomial with a pair of complex conjugate zeros on the unit circle.

$$\begin{aligned} a(x) &= (x - e^{i\phi})(x - e^{-i\phi}) \\ &= x^2 - e^{-i\phi}x - e^{i\phi}x + e^{i\phi}e^{-i\phi} \\ &= x^2 - 2\cos(\phi) + 1 \end{aligned}$$

This is obviously palindromic.

Any real polynomial that has  $k$  pairs of complex conjugate zeros will be the product of  $k$  palindromic polynomials, and thus palindromic. If a polynomial has  $k$  pairs of complex conjugate zeros and the root  $+1$  it will also be palindromic, while if it has  $-1$  as a root it will be antipalindromic. This completes the proof.

The converse of this statement is not necessarily true; not every palindromic polynomial has all its zeros on the unit circle. The idea behind the

LSPs is to define palindromic and antipalindromic polynomials that *do* obey the converse rule. Let's see how this is done.

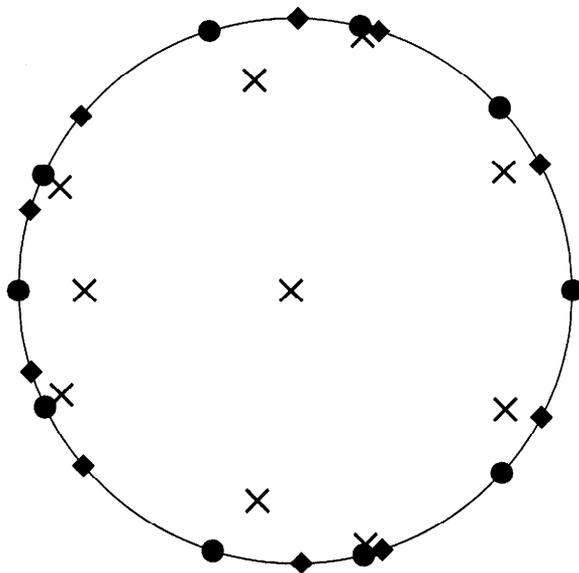
Any arbitrary polynomial  $a(x)$  can be written as the sum of a palindromic polynomial  $p(x)$  and an antipalindromic polynomial  $q(x)$

$$a_m = \frac{1}{2}(p_m + q_m) \quad \text{where} \quad \begin{aligned} p_m &= a_m + a_{M-m} \\ q_m &= a_m - a_{M-m} \end{aligned} \quad (9.39)$$

(if  $M$  is even the middle coefficient appears in  $p_m$  only). When we are dealing with polynomials that have their constant term equal to unity, we would like the polynomials  $p_m$  and  $q_m$  to share this property. To accomplish this we need only pretend for a moment that  $a_m$  is a polynomial of order  $M + 1$  and use the above equation with  $a_{M+1} = 0$ .

$$a_m = \frac{1}{2}(p_m + q_m) \quad \text{where} \quad \begin{aligned} p_m &= a_m + a_{M+1-m} \\ q_m &= a_m - a_{M+1-m} \end{aligned} \quad (9.40)$$

Now  $a_0 = p_0 = q_0 = 1$  but  $p_m$  and  $q_m$  are polynomials of degree  $M + 1$ .



**Figure 9.5:** The zeros of a polynomial and of its palindromic and antipalindromic components. The Xs are the zeros of a randomly chosen tenth order polynomial (constrained to have its zeros inside the unit circle). The circles and diamonds are the zeros of the  $p(x)$  and  $q(x)$ . Note that they are all on the unit circle and are intertwined.

Formally we can write the relationships between the polynomials

$$a(x) = \frac{1}{2} (p(x) + q(x)) \quad \text{where} \quad \begin{pmatrix} p(x) \\ q(x) \end{pmatrix} = a(x) \pm x^{M+1} a(x^{-1})$$

and it is not hard to show that if all the zeros of  $a(x)$  are inside the unit circle, then all the zeros of  $p(x)$  and of  $q(x)$  are *on* the unit circle. Furthermore, the zeros of  $p(x)$  and  $q(x)$  are intertwined, i.e., between every two zeros of  $p(x)$  there is a zero of  $q(x)$  and vice versa. Since these zeros are on the unit circle they are uniquely specified by their angles. For the polynomial in the denominator of the LPC frequency response these angles represent frequencies, and are called the LSP frequencies.

Why are the LSP frequencies a useful representation of the all-pole filter? The LPC coefficients are not a very homogeneous set, the higher-order  $b_m$  being more sensitive than the lower-order ones. LPC coefficients do not quantize well; small quantization error may lead to large spectral distortion. Also the LPC coefficients do not interpolate well; we can't compute them at two distinct times and expect to accurately predict them in between. The zeros of the LPC polynomial are a better choice, since they all have the same physical interpretation. However, finding these zeros numerically entails a complex two-dimensional search, while the zeros of  $p(x)$  and  $q(x)$  can be found by simple one-dimensional search techniques. In speech applications it has been found empirically that the LSP frequencies quantize well and interpolate better than all other parameters that have been tried.

## EXERCISES

- 9.11.1 Let's create a random polynomial of degree  $M$  by generating  $M + 1$  random numbers and using them as coefficients. We can now find the zeros of this polynomial and plot them in the complex plane. Verify empirically the hard-to-believe fact that for large  $M$  most of the zeros are close to the unit circle (except for large negative real zeros). Change the distribution of the random number generator. Did anything change? Can you explain why?
- 9.11.2 Prove that if all the zeros of  $a(x)$  are inside the unit circle, then all the zeros of  $p(x)$  and of  $q(x)$  are *on* the unit circle. (Hint: One way is write the  $p$  and  $q$  polynomials as  $a(x)(1 \pm h(x))$  where  $h(x)$  is an all-pass filter.) Prove that the zeros of  $p(x)$  and  $q(x)$  are intertwined. (Hint: Show that the phase of all-pass filter is monotonic, and alternately becomes  $\pi$  (zero of  $p$ ) and  $0$  (zero of  $q$ ).)
- 9.11.3 A pipe consisting of  $M + 1$  cylinders that is completely open or completely closed at the end has its last reflection coefficient  $k_{M+1} = \pm 1$ . How does this relate to the LSP representation?

- 9.11.4 Generate random polynomials and find their zeros. Now build  $p(x)$  and  $q(x)$  and find their zeros. Verify that if the polynomial zeros are inside the unit circle, then those of  $p$  and  $q$  are on the unit circle. Is there a connection between the angles of the polynomial zeros and those of the LSPs?
- 9.11.5 The Greek mathematician Apollonius of Perga discovered that given two points in the plane  $z_1$  and  $z_2$ , the locus of points with distances to  $z_1$  and  $z_2$  in a fixed ratio is circle (except when the ratio is fixed at one when it is a straight line). Prove this theorem. What is the connection to LSPs?

## 9.12 Higher-Order Signal Processing

The main consequence of the Wiener-Khintchine theorem is that most of the signal processing that we have learned is actually only ‘power spectrum’ processing. For example, when we use frequency selective filters to enhance signals we cannot discriminate between signals with the same power spectrum but different spectral phase characteristics. When we use correlations to solve system identification problems, we are really only recovering the square of the frequency response. We have yet to see methods for dealing with signals with non-Gaussian distributions or non-minimum-phase attributes of systems.

In this section we will take a brief look at a theory of signal processing that *does* extend beyond the power spectrum. We will assume that our signals are stochastic and stationary and accordingly use the probabilistic interpretation of correlations, first introduced in Section 5.6. There we defined the moment functions, definitions we repeat here in slightly modified form.

$$M_s^{[k]}(m_1, m_2, \dots, m_k) \equiv \langle s_n s_{n+m_1} \cdots s_{n+m_k} \rangle \quad (9.41)$$

The  $k^{\text{th}}$  moment function of the digital stationary stochastic signal  $s$  is the average of the product of  $k + 1$  signal values, at time lags defined by the moment function’s parameters.

The first-order moment function is simply

$$M_s^{[1]} = \langle s_n \rangle$$

the signal’s average (DC) value. The second-order moment function is

$$M_s^{[2]}(m) = \langle s_n s_{n+m} \rangle = C_s(-m)$$

the autocorrelation (recall the probabilistic interpretation of autocorrelation of equation (5.22)). The third-order moment function is a new entity.

$$M_s^{[3]}(m_1, m_2) = \langle s_n s_{n+m_1} s_{n+m_2} \rangle$$

Normal signal processing exploits only the first and second moments; higher-order signal processing utilizes the third and higher moments as well.

The moment functions have an especially simple interpretation for the special case of a stochastic signal that can only take on values 0 and 1. The first moment, the signal's average value, can be interpreted as the probability that the signal takes on the value 1; if the average is  $\frac{1}{2}$  this means that the 0 and 1 values are equally probable. The second moment, the autocorrelation, relates the signal's values at two different times separated by  $m$ . Its interpretation is the probability that the signal takes on value 1 at any two times separated by  $m$ . If the signal is white (i.e., the 0 or 1 value is chosen independently at each time instant), then the autocorrelation will be  $\frac{1}{2}$  (fifty-fifty) for all nonzero time lags. A higher correlation at time lag  $m$  means that the signal's being 1 at time  $n$  encourages the probability that it will be 1 at time  $n + m$  as well, while a lower correlation indicates that a 1 at one time inhibits a second 1. A periodic signal will have its second moment function equal to unity for a lag equaling the period of any multiple thereof, since the probability of matching values is a certainty. As correlations of nonperiodic function normally die down for large enough lags, the two events become independent for large  $m$ .

$$M_s^{[2]}(m) \xrightarrow{m \rightarrow \infty} (M_s^{[1]})^2 \quad (9.42)$$

The interpretation of the third moment function is now clear. It is the probability that the 0-1 stochastic signal takes on the value 1 at all three times  $n$ ,  $n + m_1$ , and  $n + m_2$ . If both  $m_1$  and  $m_2$  are very large we expect the third moment to equal the mean cubed, while if  $m_1$  is small enough for there to be nontrivial correlations, but  $m_2$  still large, then we expect a slightly more complex expression.

$$M_s^{[3]}(m_1, m_2) \xrightarrow{m_2 \rightarrow \infty} M_s^{[2]}(m_1) M_s^{[1]} \quad (9.43)$$

However, the third moment can be significantly different from this as well. For instance, a signal that is generated by

$$s_n = \Theta \left( \nu_n + \alpha_1 s_{n-m_1} + \alpha_2 s_{n-m_2} + \dots \right)$$

(where  $\nu_n$  is some driving noise signal) will have a nontrivial third moment function with just these lags.

Similarly the fourth and higher moment functions give the probability that the signal takes on 1 values at four or more times. In practice, interpretation of numeric moment function data is complex because of the contributions from lower-order moments, as in equations (9.42) and (9.43). For example, if 0 and 1 are equally probable, we expect to observe 1 at two different times with a probability of one-quarter; only deviations from this value signify that there is something special about the lag between the two times. Likewise, to really understand how connected four different times are, we must subtract from the fourth moment function all the contributions from the third-order moments, but these in turn contain portions of second-order moments and so on. The way to escape this maize of twisty little passages is to define *cumulants*.

The exact definition of the cumulant is a bit tricky since we have to keep track of all possible groupings of the time instants that appear in the moment function. For this purpose we use the mathematical concept of a 'partition', which is a collection of nonempty sets whose union is a given set. For example, in the third moment there are three time instances  $n_0 = n$ ,  $n_1 = n + m_1$ , and  $n_2 = n + m_2$ , and these can be grouped into five different partitions.  $P_1 = \{(n_1, n_2, n_3)\}$ ,  $P_2 = \{(n_1), (n_2, n_3)\}$ ,  $P_3 = \{(n_2), (n_1, n_3)\}$ ,  $P_4 = \{(n_2), (n_1, n_2)\}$ , and  $P_5 = \{(n_1), (n_2), (n_3)\}$ . We'll use the symbol  $S_{ij}$  for the  $j^{\text{th}}$  set of partition  $P_i$  (e.g.,  $S_{21} = (n_1)$  and  $S_{42} = (n_1, n_2)$ ),  $N_i$  the number of such sets ( $N_1 = 1$ ,  $N_2 = N_3 = N_4 = 2$  and  $N_5 = 3$ ), and  $N_{ij}$  for the number of elements in a set (e.g.,  $N_{11} = 3$ ,  $N_{51} = 1$ ). We can now define the cumulant

$$C_s^{[k]} = \sum_i (-1)^{N_i-1} (N_i - 1)! \prod_{j=1}^i M_s^{[N_{ij}]}(S_{ij}) \quad (9.44)$$

where the sum is over all possible partitions of the  $k$  time instants.

It will be convenient to have a special notation for the signal with its DC component removed,  $\hat{s} \equiv s - \langle s \rangle$ . The first few cumulants can now be expressed as follows:

$$C_s^{[1]} = M_s^{[1]} = \langle s_n \rangle$$

as expected,

$$C_s^{[2]}(m) = M_s^{[2]}(m) - (M_s^{[1]})^2 = \langle \hat{s}_n \hat{s}_{n+m} \rangle$$

which is the autocovariance rather than the autocorrelation,

$$\begin{aligned}
C_s^{[3]}(m_1, m_2) &= \\
M_s^{[3]}(m_1, m_2) - M_s^{[1]} \left( M_s^{[2]}(n) + M_s^{[2]}(n+m_1) + M_s^{[2]}(n+m_2) \right) + 2(M_s^{[1]})^3 \\
&= \langle \hat{s}_n \hat{s}_{n+m_1} \hat{s}_{n+m_2} \rangle
\end{aligned}$$

a surprisingly simple result,

$$\begin{aligned}
C_s^{[4]}(m_1, m_2, m_3) &= M_s^{[4]}(m_1, m_2, m_3) - \dots - 6(M_s^{[1]})^4 = \\
&\langle \hat{s}_n \hat{s}_{n+m_1} \hat{s}_{n+m_2} \hat{s}_{n+m_3} \rangle \\
&- C_s^{[2]}(m_1) C_s^{[2]}(m_2-m_3) - C_s^{[2]}(m_2) C_s^{[2]}(m_3-m_1) - C_s^{[2]}(m_3) C_s^{[2]}(m_1-m_2)
\end{aligned}$$

which is somewhat more complex. For the special case of a zero mean signal and  $m_1 = m_2 = m_3$ ,  $C^{[2]}$  is the variance,  $C^{[3]}$  the ‘skew’, and  $C^{[4]}$  the ‘kurtosis’.

Other than their interpretability, the cumulants are advantageous due to their convenient characteristics. The most important of these, and the reason they are called ‘cumulants’, is their additivity.

$$C_{x+y}^{[k]}(m_1, m_2, \dots, m_{k-1}) = C_x^{[k]}(m_1, m_2, \dots, m_{k-1}) + C_y^{[k]}(m_1, m_2, \dots, m_{k-1})$$

It is easy to see that this characteristic is not shared by the moment functions. Another nice feature is their blindness to DC components

$$C_{x+\alpha}^{[k]}(m, m_2, \dots, m_{k-1}) = C_x^{[k]}(m_1, m_2, \dots, m_{k-1})$$

where  $\alpha$  is any constant. Like the moments, cumulants are permutation blind

$$C_s^{[k]}(m_{\sigma_1}, m_{\sigma_2}, \dots, m_{\sigma_{k-1}}) = C_s^{[k]}(m_1, m_2, \dots, m_{k-1})$$

where  $\sigma_i$  is any permutation of  $1 \dots k-1$ ; and scale according to their order.

$$C_{g^k s}^{[k]}(m_1, m_2, \dots, m_{k-1}) = g^k C_s^{[k]}(m_1, m_2, \dots, m_{k-1})$$

If the signal is symmetrically distributed then all odd-order cumulants vanish. If a signal is Gaussianly distributed all cumulants above the second-order vanish.

Higher-order spectra are defined in analogy with the Wiener-Khinchine theorem. Just as the spectrum is the FT of  $C_s^{[2]}(m)$ , the *bispectrum* is defined to be the two-dimensional FT of  $C_s^{[3]}(m_1, m_2)$ , and the *trispectrum* the three-dimensional FT of  $C_s^{[4]}(m_1, m_2, m_3)$ . It can be shown that for signals with finite energy, the general *polyspectrum* is given by a product of FTs.

$$S^{[k]}(\omega_1, \omega_2 \dots \omega_{k-1}) = S(\omega_1) S(\omega_2) \dots S(\omega_{k-1}) S^*(\omega_1 + \omega_2 + \dots + \omega_{k-1})$$

Now that we have defined them, we can show that cumulants are truly useful. Assume that we have a non-Gaussian signal distorted by Gaussian noise. Standard signal processing does not take advantage of the higher-order statistics of the signal, and can only attempt to separate the signal from the noise in the power spectral domain. However, cumulants of the third and higher orders of the noise will vanish exactly, while those of the signal will not, thus providing a more powerful tool for recovery of such a signal. For example, higher-order matched filters can be used as sensitive detectors of the arrival of non-Gaussian signals in Gaussian noise.

We know from Section 8.1 that intermodulation products are produced when two sinusoids enter a nonlinearity. Assume we observe several frequency components in the output of a possibly nonlinear system; is there any way to tell if they are intermodulation frequencies rather than independent signals that happen to be there? The fingerprint of the phenomenon is that intermodulation products are necessarily phase coupled to the inputs; but such subtle phase relations are lost in classical correlation-based analysis. By using higher-order cumulants intermodulation frequencies can be identified and the precise nature of system nonlinearities classified.

In Sections 6.12 and 6.13 we saw how to perform correlation-based system identification when we had access to a system's input and output. Sometimes we may desire to identify a system, but can only observe its output. Amazingly, this problem may be tractable if the input signal is non-Gaussian. For example, if the unknown system is an  $N$  tap FIR filter,

$$y_n = \sum_{m=0}^{N-1} h_m x_{n-m} + \nu_n$$

the input  $x$  is zero mean but with nonzero third-order cumulant, and the output  $y$  is observed contaminated by additive Gaussian (but not necessarily white) noise  $\nu$ , then the system's impulse response can be derived solely from the output's third-order cumulants.

$$h_m = \frac{C_y^{[3]}(N-1, m)}{C_y^{[3]}(N-1, 0)} \quad (9.45)$$

This amazing result is due to the input's third-order cumulant (assumed nonzero) appearing in the numerator and denominator and hence cancelling out, and can be generalized to higher-order cumulants if needed. A related result is that cumulant techniques can be used for *blind equalization*, that is, constructing the inverse of an unknown distorting system, without access to the undistorted input.

## EXERCISES

9.12.1 Find all the partitions of four time instants and express  $C_s^{[4]}(m_1, m_2, m_3)$  in terms of moments.

9.12.2 Consider the three systems ( $0 < a, b < 1$ )

$$\begin{aligned}y_n &= x_n - (a + b)x_{n-1} + abx_{n-2} \\y_n &= x_n - (a + b)x_{n+1} + abx_{n+2} \\y_n &= -ax_{n+1} + (1 + ab)x_n - bx_{n-1}\end{aligned}$$

What are the system functions for these systems? Which system is minimum phase, which maximum phase, and which mixed phase? Take  $x_n$  to be a zero mean stationary white noise signal, with  $\langle x_n x_{n+m} \rangle = \delta_{m0}$  and  $\langle x_n x_{n+m_1} x_{n+m_2} \rangle = \delta_{m_1 m_2}$ . Show that the output signals from all three systems have the same autocorrelations. Prove that for all three systems the same frequency response is measured. Why is this result expected? Show that the third-order moments are different.

9.12.3 Prove equation (9.45).

9.12.4 There is another way of defining cumulants. Given the  $k$  signal values

$$s_n, s_{n+m_1}, \dots, s_{n+m_{k-1}}$$

we posit  $k$  dummy variables  $w_0 \dots w_{k-1}$  and define the following function, known as the characteristic function.

$$\Phi(w_0 \dots w_{k-1}) = \left\langle e^{i(w_0 s_n + w_1 s_{n+m_1} + \dots + w_{k-1} s_{n+m_{k-1}})} \right\rangle$$

The cumulants are the coefficients of the Taylor expansion of the logarithm of this function. Derive the first few cumulants according to this definition and show that they agree with those in the text. Derive the additivity property from this new definition.

9.12.5 In the text we mentioned the application of higher-order signal processing to the identification of intermodulation products. Let  $\varphi_1, \varphi_2$  and  $\varphi_3$  be independent uniformly distributed random variables and define two stochastic signals

$$\begin{aligned}s_n^{[a]} &= \cos(\omega_1 n + \varphi_1) + \cos(\omega_2 n + \varphi_2) \cos((\omega_1 + \omega_2)n + (\varphi_1 + \varphi_2)) \\s_n^{[b]} &= \cos(\omega_1 n + \varphi_1) + \cos(\omega_2 n + \varphi_2) \cos((\omega_1 + \omega_2)n + \varphi_3)\end{aligned}$$

each of which has three spectral lines, the highest frequency being the sum of the lower two. The highest component of  $s^{[a]}$  could be an intermodulation product since it is phase-locked with the other two, while that of  $s^{[b]}$  is an unrelated signal. Show that both signals have the same autocorrelation and power spectrum, but differ in their third-order cumulants.

## Bibliographical Notes

Matched filters are covered in most books on communications theory, e.g. [242, 95].

Wiener's first expositions of the Wiener-Khintchine theorem were in mathematical journals [276] but he later wrote an entire book on his discoveries [277]. The co-discoverer of the theorem was Aleksandr Khintchine (or Khinchin), whose *Mathematical Foundations of Information Theory* was translated into English from the original Russian in 1957.

The second volume of Norbert Wiener's autobiography [280] has fascinating background information on Wiener's work at MIT during the World War II years. His 1942 report, entitled *Extrapolation, Interpolation and Smoothing of Stationary Time Series*, was suppressed because of possible military applications, and finally released only in 1949 [278]. Even though written to be more understandable than the former paper, its mathematics, more familiar to physicists than engineers, was so difficult to the latter audience that it was commonly called the 'yellow peril'. Levinson both explained Wiener's results to a wider audience [146] and translated the formalism to the digital domain. While accomplishing this second task he invented his recursion [147], although digital hardware capable of computing it did not exist at the time.

The invention of LPC is due to Bishnu Atal of Bell Labs [10], who was mainly interested in its use for compression of speech [9]. The LSP frequencies are due to Itakura of NTT Research Labs [109] (but don't bother checking the original reference, it's only an abstract).

Higher-order signal processing is the subject of a book [181] and numerous review articles [173, 182]. [33] discusses partitions in a simple way, and includes source code for computing the number of partitions of  $n$  objects. Cumulants were introduced in statistics by Fisher in the 1930s and in use in physics at about the same time. The idea of higher-order spectra as the FT of cumulants dates back to Kolmogorov, but the nomenclature 'polyspectra' is due to Tukey. The use of cumulants for output-only system identification is due to Georgios Giannakis [72]. A few references to the extensive literature on applications of cumulants include noise cancellation [49]; system identification [73, 65]; blind equalization [235, 236]; and signal separation [286, 287, 108].