# Filters

In everyday parlance a 'filter' is a device that removes some component from whatever is passed through it. A drinking-water filter removes salts and bacteria; a coffee filter removes coffee grinds; an air filter removes pollutants and dust. In electronics the word 'filter' evokes thoughts of a system that removes components of the input signal based on frequency. A notch filter may be employed to remove a narrow-band tone from a received transmission; a noise filter may remove high-frequency hiss or low-frequency hum from recordings; antialiasing filters are needed to remove frequencies above Nyquist before A/D conversion.

Less prevalent in everyday usage is the concept of a filter that emphasizes components rather than removing them. Colored light is created by placing a filter over a white light source; one filters flour retaining the finely ground meal; entrance exams filter to find the best applicants. The electronic equivalent is more common. Radar filters capture the desired echo signals; deblurring filters are used to bring out unrecognizable details in images; narrow-band audio filters lift Morse code signals above the interference.

In signal processing usage a filter is any system whose output spectrum is derived from the input's spectrum via multiplication by a time-invariant weighting function. This function may be zero in some range of frequencies and as a result remove these frequencies; or it may be large in certain spectral regions, consequently emphasizing these components. Or it may half the energy of some components while doubling others, or perform any other arbitrary characteristic.

However, just as a chemical filter cannot create gold from lead, a signal processing filter cannot create frequency components that did not exist in the input signal. Although definitely a limitation, this should not lead one to conclude that filters are uninteresting and their output trivial manipulation of the input. To do so would be tantamount to concluding that sculptors are not creative because the sculpture preexisted in the stone and they only removed extraneous material. In this chapter we will learn how filters are specified in both frequency and time domains. We will learn about fundamental limitations that make the job of designing a filter to meet specifications difficult, but will not cover the theory and implementation of filter design in great detail. Whole books are devoted to this subject and excellent software is readily available that automates the filter design task. We will only attempt to provide insight into the basic principles of the theory so that the reader may easily use any of the available programs.

# 7.1 Filter Specification

Given an input signal, different filters will produce different output signals. Although there are an infinite number of different filters, not every output signal can be produced from a given input signal by a filter. The restrictions arise from the definition of a filter as a linear time-invariant operator. Filters never produce frequency components that did not exist in the input signal, they merely attenuate or accentuate the frequency components that exist in the input signal.

Low-pass filters are filters that pass DC and low frequencies, but block or strongly attenuate high frequencies. *High-pass* filters pass high frequencies but block or strongly attenuate low frequencies and DC. *Band-pass* filters block both low and high frequencies, passing only frequencies in some 'passband' range. *Band-stop* filters do the opposite, passing everything not in a defined 'stop-band'. *Notch* filters are extreme examples of band-stop filters, they pass all frequencies with the exception of one well defined frequency (and its immediate vicinity). *All-pass* filters have the same gain magnitude for all frequencies but need not be the identity system since phases may still be altered.

The above definitions as stated are valid for analog filters. In order to adapt them for DSP we need to specify that only frequencies between zero and half the sampling rate are to be considered. Thus a digital system that blocks low frequencies and passes frequencies from quarter to half the sampling frequency is a high-pass filter.

An ideal filter is one for which every frequency is either in its passband or stop-band, and has unity gain in its pass-band and zero gain in its stop-band. Unfortunately, ideal filters are unrealizable; we can't buy one or even write a DSP routine that implements one. The problem is caused by the sharp jump discontinuities at transitions in the frequency domain that cannot be precisely implemented without peeking infinitely into the



Figure 7.1: Frequency response of ideal and nonideal filters. In (A) we see the low-pass filters, in (B) the high-pass filters, in (C) the band-pass filters and in (D) the band-stop (notch) filters.

future. On the left side of Figure 7.1 we show the frequency response of ideal filters, while the right side depicts more realistic approximations to the ideal response. Realistic filters will always have a finite *transition region* between pass-bands and stop-bands, and often exhibit *ripple* in some or all of these areas. When designing a filter for a particular application one has to specify what amount of ripple and how much transition width can be tolerated. There are many techniques for building both analog and digital filters to specification, but all depend on the same basic principles.

Not all filters are low-pass, high-pass, band-pass, or band-stop, any frequency dependent gain is admissible. The gain of a pre-emphasis filter increases monotonically with frequency, while that of a de-emphasis filter decreases monotonically. Such filters are often needed to compensate for or eliminate the effects of various other signal processing systems.

Filtering in the analog world depends on the existence of components whose impedance is dependent on frequency, usually capacitors and inductors. A capacitor looks like an open circuit to DC but its impedance decreases with increasing frequency. Thus a series-connected capacitor effectively blocks DC current but passes high frequencies, and is thus a low-pass filter. A parallel-connected capacitor short circuits high frequencies but not DC or low frequencies and is thus a high-pass filter. The converse can be said about series- and parallel-connected inductors.

Filtering in DSP depends on mathematical operations that remove or emphasize different frequencies. Averaging adjacent signal values passes DC and low frequencies while canceling out high frequencies. Thus averaging behaves as a low-pass filter. Adding differences of adjacent values cancels out DC and low frequencies but will pass signals with rapidly changing signs. Thus such operations are essentially high-pass filters.

#### FILTERS

One obvious way to filter a digital signal is to 'window' it in the frequency domain. This requires transforming the input signal to the frequency domain, multiplying it there by the desired frequency response (called a 'window function'), and then transforming back to the time domain. In practice the transformations can be carried out by the FFT algorithm in  $O(N \log N)$ time (N being the number of signal points), while the multiplication only requires O(N) operations; hence this method is  $O(N \log N)$  in complexity. This method as stated is only suitable when the entire signal is available in a single, sufficiently short vector. When there are too many points for a single DFT computation, or when we need to begin processing the signal before it has completely arrived, we may perform this process on successive blocks of the signal. How the individually filtered blocks are recombined into a single signal will be discussed in Section 15.2.

The frequency domain windowing method is indeed a straightforward and efficient method of digital filtering, but not a panacea. The most significant drawback is that it is not well suited to real-time processing, where we are given a single input sample, and are expected to return an output sample. Not that it is impossible to use frequency domain windowing for real-time filtering. It may be possible to keep up with real-time constraints, but a processing delay *must* be introduced. This delay consists of the time it takes to fill the buffer (the buffer delay) plus the time it takes to perform the FFT, multiplication, and iFFT (the computation delay). When this delay cannot be tolerated there is no alternative to time domain filtering.

#### **EXERCISES**

7.1.1 Classify the following filters as low-pass, high-pass, band-pass, or notch.

- 1. Human visual system, which has a persistence of  $\frac{1}{20}$  of a second
- 2. Human hearing, which cannot hear under 30 Hz or above 25KHz
- 3. Line noise filter used to remove 50 or 60 Hz AC hum
- 4. Soda bottle amplifying a specific frequency when air is blown above it
- 5. Telephone line, which rejects below 200 Hz and above 3800 Hz
- 7.1.2 Design an MA filter, with an even number of coefficients N, that passes a DC signal (a, a, a, ...) unchanged but completely kills a maximal frequency signal (a, -a, a, -a, ...). For example, for N = 2 you must find two numbers  $g_1$  and  $g_2$  such that  $g_1a + g_2a = a$  but  $g_aa + g_2(-a) = 0$ . Write equations that the  $g_i$  must obey for arbitrary N. Can you find a solution for odd N?
- 7.1.3 Design a moving average digital filter, with an even number of coefficients N, that passes a maximal frequency signal unchanged but completely kills DC. What equations must the  $g_i$  obey now? What about odd N?

274

7.1.4 The squared frequency response of the ideal low-pass filter is unity below the cutoff frequency and zero above.

$$|H(\omega)|^2 = \begin{cases} 1 & \omega < \omega_c \\ 0 & \text{else} \end{cases}$$

What is the full frequency response assuming a delay of N samples?

- 7.1.5 Show that the ideal low-pass filter is not realizable. To do this start with the frequency response of the previous exercise and find the impulse response using the result from Section 6.12 that the impulse response is the FT of the frequency response. Show that the impulse response exists for negative times (i.e., before the impulse is applied), and that no amount of delay will make the system causal.
- 7.1.6 Show that results similar to that of the previous exercise hold for other ideal filter types. (Hint: Find a connection between the impulse response of ideal band-pass or band-stop filters and that of ideal low-pass filters.)
- 7.1.7 The Paley-Wiener theorem states that if the impulse response  $h_n$  of a filter has a finite square sum then the filter is causal if and only if  $\int |\ln |H(\omega)| d\omega$  is finite. Use this theorem to prove that ideal low-pass filters are not realizable.
- 7.1.8 Prove the converse to the above, namely that any signal that is nonzero over some time can't be band-limited.

### 7.2 Phase and Group Delay

The previous section concentrated on the specification of the magnitude of the frequency response, completely neglecting its angle. For many applications power spectrum specification is sufficient, but sometimes the spectral phase can be important, or even critical. A signal's phase can be used for carrying information, and passing such a phase-modulated signal through a filter that distorts phase may cause this information to be lost. There are even many uses for all-pass filters, filters that have unity gain for all frequencies but varying spectral phase!

Let's return to fundamentals. The frequency response  $H(\omega)$  is defined by the relation

$$Y(\omega) = H(\omega)X(\omega)$$

which means that

$$|Y(\omega)| = |H(\omega)||X(\omega)|$$
 and  $\angle Y(\omega) = \angle X(\omega) + \angle H(\omega)$ 

276

or in words, the input spectral magnitude at each frequency is multiplied by the frequency response gain there, while the spectral phase is delayed by the angle of the frequency response at each frequency. If the spectral phase is unchanged by the filter, we say that the filter introduces no phase distortion; but this is a needlessly harsh requirement.

For example, consider the simple delay  $y_n = x_{n-m}$ . This FIR filter is allpass (i.e., the absolute value of its frequency response is a constant unity), but delaying sinusoids effectively changes their phases. By how much is the phase delayed? The sinusoid  $x_n A \sin(\omega n)$  becomes

$$y_n = x_{n-m} = A \sin \left( \omega (n-m) \right) = A \sin (\omega n - \omega m)$$

so the phase delay is  $\omega m$ , which is frequency-dependent. When the signal being delayed is composed of many sinusoids, each has a phase delay proportional to its frequency, so the simple delay causes a spectral phase shift proportional to frequency, a characteristic known as *linear phase*.

Some time delay is often unavoidable; the noncausal FIR filter y = h \* x with coefficients

$$h_{-L}, h_{-L+1}, \ldots h_{-1}, h_0, h_1, \ldots h_{L-1}, h_L$$

introduces no time delay since the output  $y_n$  corresponds to the present input  $x_n$ . If we require this same filter to be causal, we cannot output  $y_n$ until the input  $x_L$  is observed, and so a time delay of L, half the filter length, is introduced.

$$g_0 = h_{-L}, \ g_1 = h_{-L+1}, \ g_L = h_0, \ \ldots \ g_{2L} = h_L$$

This type of delay is called *buffer delay* since it results from buffering the inputs.

It is not difficult to show that if the impulse response is symmetric (or antisymmetric) then the linear phase shift resulting from buffer delay is the only phase distortion. Applying the symmetric noncausal FIR filter with an odd number of coefficients

$$h_L, h_{L-1}, \ldots, h_1, h_0, h_1, \ldots, h_{L-1}, h_L$$

to a complex exponential  $e^{i\omega n}$  we get

$$y_{n} = \sum_{m=-L}^{+L} h_{|m|} e^{i\omega(n-m)} = h_{0} e^{i\omega n} + 2e^{i\omega n} \sum_{m=1}^{L} h_{|m|} \cos(m\omega)$$

so that the frequency response is real and thus has zero phase delay.

$$H(\omega) = h_0 + 2\sum_{m=1}^{L} h_{|m|} \cos(m\omega)$$

We can force this filter to be causal by shifting it by L

$$g_0 = h_L, \ g_1 = h_{L-1}, \ \ldots \ g_L = h_0, \ \ldots \ g_{2L} = h_L$$

and the symmetry is now somewhat hidden.

$$g_0 = g_{2L}, \ g_1 = g_{2L-1}, \ \ldots \ g_m = g_{2L-m}$$

Once again applying the filter to a complex exponential leads to

$$y_n = \sum_{m=0}^{2L} g_m e^{i\omega(n-m)} = g_L e^{i\omega(n-L)} + 2e^{i\omega n} e^{-i\omega L} \sum_{m=0}^{L-1} g_m \cos(m\omega)$$

so that the frequency response is

$$H(\omega) = \left(g_L + 2\sum_{m=0}^{L-1} g_m \cos(m\omega)\right) e^{-i\omega L} = |H(\omega)|e^{-i\omega L}$$

(the important step is isolating the imaginary portion) and the filter is seen to be linear-phase, with phase shift corresponding to a time delay of L.

The converse is true as well, namely all linear-phase filters have impulse responses that are either symmetric or antisymmetric. We can immediately conclude that causal IIR filters cannot be linear-phase, since if the impulse response continues to the end of time, and must be symmetric, then it must have started at the beginning of time. This rules out the filter being causal.

From now on we will not consider a linear phase delay (constant time delay) to be phase 'distortion'. True phase distortion corresponds to nonlinearities in the phase as a function of frequency. To test for deviation from linearity it is useful to look at the first derivative, since linear phase response will have a constant derivative, and deviations from linearity will show up as deviations from a constant value. It is customary to define the group delay

$$\tau(\omega) = -\frac{d}{d\omega} \angle H(\omega) \tag{7.1}$$

where the phase must be unwrapped (i.e., the artificial discontinuities of  $2\pi$  removed) before differentiation. What is the difference between phase delay and group delay?



Figure 7.2: The difference between phase delay and group delay. In (A) we see the input signal, consisting of the sum of two sinusoids of nearly the same frequency. (B) depicts the output of a filter with unity gain, phase delay of  $\pi$ , and zero group delay, while the graph in (C) is the output of a filter with unity gain, no phase delay, but nonzero group delay. Note that the local phase in (C) is the same as that of the input, but the position of the beat amplitude peak has shifted.

In Figure 7.2 we see the effect of passing a signal consisting of the sum of two sinusoids of nearly the same frequency through two filters. Both filters have unity gain in the spectral area of interest, but the first has maximal phase delay and zero derivative (group delay) there. The second filter has zero phase delay but a group delay of one-half the beat period. Both filters distort phase, but the phase distortions are different at the frequency of the input signal.

#### EXERCISES

- 7.2.1 Show that an antisymmetric FIR filter  $(h_n = -h_{-n})$  has zero phase and when made causal has linear phase.
- 7.2.2 Prove that all linear-phase filters have impulse responses that are either symmetric or antisymmetric.
- 7.2.3 Assume that two filters have phase delay as a function of frequency  $\Phi_1(\omega)$  and  $\Phi_2(\omega)$ . What is the phase delay of the two filters in series? What about the group delay?
- 7.2.4 In a non-real-time application a nonlinear-phase filter is run from the end of the signal buffer toward the beginning. What phase delay is introduced?

**FILTERS** 

- 7.2.5 Stable IIR filters cannot be truly linear-phase. How can the result of the previous exercise be used to create a filter with linear phase based on IIR filtering? How can this technique be used for real-time linear-phase IIR filtering with delay? (Hint: Run the filter first from the beginning of the buffer to the end, and then back from the end toward the beginning.)
- 7.2.6 What is the phase delay of the IIR filter of equation (6.39)? What is the group delay?
- 7.2.7 Can you think of a use for all-pass filters?

# 7.3 Special Filters

From the previous section you may have received the mistaken impression that all filters are used to emphasize some frequencies and attenuate others. In DSP we use filters to implement almost every conceivable mathematical operation. Sometimes we filter in order to alter the time domain characteristics of a signal; for example, the simple delay is an FIR filter, although its specification is most natural in the time domain. The DSP method of detecting a narrow pulse-like signal that may be overlooked is to build a filter that emphasizes the pulse's particular shape. Conversely, a signal may decay too slowly and be in danger of overlapping other signals, in which case we can narrow it by filtering. In this section we will learn how to implement several mathematical operations, such as differentiation and integration, as filters.

A simple task often required is smoothing, that is, removing extraneous noise in order to recover the essential signal values. In the numerical analysis approach smoothing is normally carried out by approximating the data by some appropriate function (usually a polynomial) and returning the value of this function at the point of interest. This strategy works well when the chosen function is smooth and the number of free parameters limited so that the approximation is not able to follow all the fluctuations of the observed data. Polynomials are natural in most numeric analysis contexts since they are related to the Taylor expansion of the function in the region of interest. Polynomials are not as relevant to DSP work since they have no simple frequency domain explanation. The pertinent functional form is of course the sum of sinusoids in the Fourier expansion, and limiting the possible oscillation of the function is equivalent to requiring these sinusoids to be of low frequency. Hence the task of smoothing is carried out in DSP by lowpass filtering. The new interpretation of smoothing is that of blocking the high-frequency noise while passing the signal's energy.

The numerical analysis and DSP approaches are not truly incompatible. For the usual case of evenly sampled data, polynomial smoothing can be implemented as a filter, as was shown for the special case of a five-point parabola in exercise 6.6.5. For that case the smoothed value at time n was found to be the linear combination of the five surrounding input values,

$$y_n = a_2 x_{n-2} + a_1 x_{n-1} + a_0 x_n + a_1 x_{n+1} + a_2 x_{n+2}$$

which is precisely a symmetric MA filter. Let's consider the more general case of optimally approximating 2L + 1 input points  $x_n$  for  $n = -L \ldots + L$  by a parabola in discrete time.

$$y_n = a_2 n^2 + a_1 n + a_0$$

For notational simplicity we will only consider retrieving the smoothed value for n = 0, all other times simply requiring shifting the time axis.

The essence of the numerical analysis approach is to find the coefficients  $a_2, a_1$ , and  $a_0$  that make  $y_n$  as close as possible to the 2L + 1 given  $x_n$   $(n = -L \dots + L)$ . This is done by requiring the squared error

$$\epsilon = \sum_{n=-L}^{+L} (y_n - x_n)^2 = \sum_{n=-L}^{+L} (a_2 n^2 + a_1 n + a_0 - x_n)^2$$

to be minimal. Differentiating with respect to a, b, and c and setting equal to zero brings us to three equations, known as the *normal equations* 

$$B_{00}a_0 + B_{01}a_1 + B_{02}a_2 = C_0$$
  

$$B_{10}a_0 + B_{11}a_1 + B_{12}a_2 = C_1$$
  

$$B_{20}a_0 + B_{21}a_1 + B_{22}a_2 = C_2$$
  
(7.2)

where we have defined two shorthand notations.

$$B_{ij} = \sum_{n=-L}^{+L} n^{i+j}$$
 and  $C_i = \sum_{n=-L}^{+L} n^i x_n$ 

The *B* coefficients are universal, i.e., do not depend on the input  $x_n$ , and can be precalculated given *L*. It is obvious that if the data are evenly distributed around zero  $(n = -L, -L + 1, \ldots - 1, 0, +1, \ldots L - 1, L)$  then

 $B_{ij} = 0$  when i + j is odd, and the other required values can be looked up in a good mathematical handbook.

$$B_{00} = \sum_{n=-L}^{+L} 1 = 2L + 1 \equiv \mathcal{B}_0$$
  

$$B_{02} = B_{20} = B_{11} = \sum_{n=-L}^{+L} n^2 = \frac{L(L+1)(2L+1)}{3} \equiv \mathcal{B}_2$$
  

$$B_{22} = \sum_{n=-L}^{+L} n^4 = \frac{L(L+1)(2L+1)(3L^2+3L-1)}{15} \equiv \mathcal{B}_4$$

The three C values are simple to compute given the inputs.

$$C_0 = \sum_{n=-L}^{+L} x_n$$

$$C_1 = \sum_{n=-L}^{+L} n x_n$$

$$C_2 = \sum_{n=-L}^{+L} n^2 x_n$$

In matrix notation the normal equations are now

$$\begin{pmatrix} \mathcal{B}_0 & 0 & \mathcal{B}_2 \\ 0 & \mathcal{B}_2 & 0 \\ \mathcal{B}_2 & 0 & \mathcal{B}_4 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} C_0 \\ C_1 \\ C_2 \end{pmatrix}$$
(7.3)

and can be readily solved by inverting the matrix

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} \mathcal{D}_0 & 0 & \mathcal{D}_2 \\ 0 & \mathcal{D}_1 & 0 \\ \mathcal{D}_2 & 0 & \mathcal{D}_4 \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ C_2 \end{pmatrix}$$
(7.4)

and the precise expressions for the  $\mathcal{D}$  elements are also universal and can be found by straightforward algebra.

$$D = \mathcal{B}_0 \mathcal{B}_2 \mathcal{B}_4 - \mathcal{B}_2^3$$
$$\mathcal{D}_0 = \frac{\mathcal{B}_2 \mathcal{B}_4}{D}$$
$$\mathcal{D}_1 = \frac{1}{\mathcal{B}_2^2}$$
$$\mathcal{D}_2 = -\frac{\mathcal{B}_2^2}{D}$$
$$\mathcal{D}_3 = \frac{\mathcal{B}_0 \mathcal{B}_2}{D}$$

#### FILTERS

Now that we have found the coefficients  $a_0, a_1$ , and  $a_2$ , we can finally find the desired smoothed value at n = 0

$$y_0 = a_2 = \mathcal{D}_0 C_0 + \mathcal{D}_2 C_2 = \sum_{n=-L}^{+L} (\mathcal{D}_0 + \mathcal{D}_2 n^2) x_n$$

which is seen to be a symmetric MA filter. So the numerical analysis approach of smoothing by parabolic approximation is equivalent to a particular symmetric MA filter, which has only a single adjustable parameter, L.

Another common task is the differentiation of a signal,

$$y(t) = \frac{d}{d\tau}x(\tau) \tag{7.5}$$

a common use being the computation of the instantaneous frequency from the phase using equation (4.67). The first approximation to the derivative is the finite difference,

$$y_n = x_n - x_{n-1}$$

but for signals sampled at the Nyquist rate or only slightly above the sample times are much too far apart for this approximation to be satisfactory. The standard numerical analysis approach to differentiation is derived from that for smoothing; first one approximates the input by some function, and then one returns the value of the derivative of that function. Using the formalism developed above we can find that in the parabolic approximation, the derivative at n = 0 is given by

$$y_0 = a_1 = \mathcal{D}_1 C_1 = \sum_{n=-L}^{+L} (\mathcal{D}_1 n) x_n$$

which is an antisymmetric MA filter, with coefficients proportional to |n|!The antisymmetry is understandable as a generalization of the finite difference, but the idea of the remote coefficients being more important than the adjacent ones is somewhat hard to embrace. In fact the whole idea of assuming that values of the derivative to be accurate just because we required the polynomial to approximate the signal values is completely ridiculous. If we do not require the derivative values to be close there is no good reason to believe that they will be; quite the contrary, requiring the polynomial approximation to be good at sampling instants will cause the polynomial to oscillate wildly in between these times, resulting in meaningless derivative estimates.

282



Figure 7.3: Frequency and impulse responses of the ideal differentiation filter.

Differentiation is obviously a linear and time-invariant operation and hence it is not surprising that it can be performed by a filter. To understand this filter in the frequency domain note that the derivative of  $s(t) = e^{i\omega t}$ is  $i\omega s(t)$ , so that the derivative's frequency response increases linearly with frequency (see Figure 7.3.A) and its phase rotation is a constant 90°.

$$H(\omega) = i\omega \tag{7.6}$$

This phase rotation is quite expected considering that the derivative of sine is cosine, which is precisely such a 90° rotation. The impulse response, given by the iFT of the frequency response,

$$h(t) = \frac{1}{2\pi} \int_{-\pi}^{\pi} i\omega e^{i\omega t} d\omega$$
  
$$= \frac{i}{2\pi} \left( \frac{e^{i\pi t}}{it} (\pi - \frac{1}{in}) - \frac{e^{-i\pi t}}{it} (-\pi - \frac{1}{in}) \right)$$
  
$$= \frac{\cos(\pi t)}{t} - \frac{\sin(\pi t)}{\pi t^2}$$

is plotted in Figure 7.3.B.

We are more interested in digital differentiators than in the analog one just derived. When trying to convert the frequency response to the digital domain we run into several small snags. First, from the impulse response we see that the ideal differentiator is unrealizable. Second, since the frequency response is now required to be periodic, it can no longer be strictly linear, but instead must be sawtooth with discontinuities. Finally, if the filter has an



Figure 7.4: Frequency and impulse responses of digital differentiation filters with even and odd numbers of coefficients. In (A) we see the frequency response of an odd length differentiator; note the linearity and discontinuities. (B) is the impulse response for this case. In (C) we see the real and imaginary parts of the frequency response of an even length differentiator. (D) is its impulse response; note that fewer coefficients are required.

even number of coefficients it can never reproduce the derivative at precisely time t = 0, but only one-half sample before or after. The frequency response for a time delay of  $-\frac{1}{2}$  is

$$H(\omega) = i\omega e^{i\omega(t+\frac{1}{2})} \tag{7.7}$$

which has both real and imaginary parts but is no longer discontinuous. We now need to recalculate the impulse response.

$$h(t) = \frac{1}{2\pi} \int_{-\pi}^{\pi} i\omega e^{i\omega(t+\frac{1}{2})} d\omega$$
$$= \frac{-\cos(\pi t)}{\pi(t+\frac{1}{2})^2}$$

The frequency and impulse responses for the odd and even cases are depicted in Figure 7.4. We see that FIR differentiators with an even number of coefficients have no discontinuities in their frequency response, and hence their coefficients vanish quickly. In practical applications we must truncate after a finite number of coefficients. For a given amount of computation an even-order differentiator has smaller error than an odd-order one.

After studying the problem of differentiation it will come as no surprise that the converse problem of integration

$$y(t) = \int_{-\infty}^{t} x(\tau) \, d\tau \tag{7.8}$$

can be implemented by filtering as well. Integration is needed for the recovery of running phase from instantaneous frequency, and for discovering the cumulative effects of slowly varying signals. Integration is also a popular function in analog signal processing where capacitors are natural integrators; DSP integration is therefore useful for simulating analog circuits.

The signal processing approach to integration starts by noting that the integral of  $s(t) = e^{i\omega t}$  is  $\frac{1}{i\omega}s(t)$ , so that the required frequency response is inversely proportional to the frequency and has a phase shift of 90°.

$$H(\omega) = \frac{1}{i\omega} \tag{7.9}$$

The standard Riemann sum approximation to the integral

$$\int_0^{nT} x(t) dt \approx T(x_0 + x_1 + \dots x_{n-1})$$

is easily seen to be an IIR filter

$$y_n = y_{n-1} + Tx_n (7.10)$$

and we'll take T = 1 from here on. What is the frequency response of this filter? If the input is  $x_n = e^{i\omega n}$  the output must be  $y_n = H(\omega)e^{i\omega n}$  where  $H(\omega)$  is a complex number that contains the gain and phase shift. Substituting into the previous equation

$$H(\omega)e^{\mathbf{i}\omega n} = y_n = y_{n-1} + x_n = H(\omega)e^{\mathbf{i}\omega(n-1)} + e^{\mathbf{i}\omega n}$$

we find that

$$H(\omega) = \frac{1}{1 - e^{-i\omega}}$$
$$|H(\omega)|^2 = \frac{1}{2(1 - \cos(\omega))}$$
$$\angle H(\omega) = \frac{1}{2}(\pi + \omega)$$



Figure 7.5: The (squared) frequency response of integrators. The middle curve is that of the ideal integrator, the Riemann sum approximation is above it, and the trapezoidal approximation below.

which isn't quite what we wanted. The phase is only the desired  $\frac{\pi}{2}$  at DC and deviates linearly with  $\omega$ . For small  $\omega$ , where  $\cos(\omega) \sim 1 - \frac{1}{2}\omega^2$ , the gain is very close to the desired  $\frac{1}{\omega}$ , but it too diverges at higher frequencies (see Figure 7.5). What this means is that this simple numeric integration is relatively good when the signal is extremely oversampled, but as we approach Nyquist both gain and phase response strongly deviate.

A slightly more complex numeric integration technique is the trapezoidal rule, which takes the average signal value  $(x_{n-1} + x_n)$  for the Riemann rectangle, rather than the initial or final value. It too can be written as an IIR filter.

$$y_n = y_{n-1} + \frac{1}{2}(x_{n-1} + x_n) \tag{7.11}$$

Using the same technique we find

 $H(\omega)e^{i\omega n} = y_n = y_{n-1} + \frac{1}{2}(x_{n-1} + x_n) = H(\omega)e^{i\omega(n-1)} + \frac{1}{2}(e^{i\omega(n-1)} + e^{i\omega n})$  which means that

$$H(\omega) = \frac{i}{2} \frac{1}{\tan(\frac{\omega}{2})}$$
$$|H(\omega)|^2 = \frac{1}{4\tan^2(\frac{\omega}{2})}$$
$$\angle H(\omega) = \frac{\pi}{2}$$

so that the phase is correct, and the gain (also depicted in Figure 7.5) is about the same as before. This is not surprising since previous signal values contribute just as in the Riemann sum, only the first and last values having half weight.

Integrators are always approximated by IIR filters. FIR filters cannot be used for true integration from the beginning of all time, since they forget everything that happened before their first coefficient. Integration over a finite period of time is usually performed by a 'leaky integrator' that gradually forgets, which is most easily implemented by an IIR filter like that of equation (6.39). While integration has a singular frequency response at DC, the frequency response of leaky integration is finite.

Our final special filter is the Hilbert transform, which we introduced in Section 4.12. There are two slightly different ways of presenting the Hilbert transform as a filter. We can consider a real filter that operates on x(t)creating y(t) such that z(t) = x(t) + iy(t) is the analytic representation, or as a complex filter that directly creates z(t) from x(t). The first form has an antisymmetric frequency response

$$H(\omega) = -i \operatorname{sgn}(\omega) = \begin{cases} -i & \omega > 0\\ 0 & \omega = 0\\ i & \omega < 0 \end{cases}$$
(7.12)

which means  $|H(\omega)|^2 = 1$  and its phase is  $\pm \frac{\pi}{2}$ . The impulse response for delay  $\tau$  is not hard to derive

$$h(t) = \frac{2}{\pi} \frac{\sin^2\left(\frac{\pi}{2}(t-\tau)\right)}{t-\tau}$$
(7.13)

except for at t = 0 where it is zero. Of course the ideal Hilbert filter is unrealizable. The frequency response of the second form is obtained by summing  $X(\omega)$  with i times the above.

$$H(\omega) = \begin{cases} 2 & \omega > 0 \\ 0 & \omega \le 0 \end{cases}$$
(7.14)

The factor of two derives from our desire to retain the original energy after removing half of the spectral components.

The Hilbert transform can be implemented as a filter in a variety of ways. We can implement it as a noncausal FIR filter with an odd number of coefficients arranged to be antisymmetric around zero. Its impulse response

$$h(t) = \frac{2}{\pi} \frac{\sin^2(\frac{\pi}{2}t)}{t}$$

288



Figure 7.6: Imaginary portion of the frequency response of a realizable digital Hilbert filter with zero delay. The ideal filter would have discontinuities at both DC and half the sampling frequency.

decays slowly due to the frequency response discontinuities at  $\omega = 0$  and  $\omega = \pi$ . With an even number of coefficients and a delay of  $\tau = -\frac{1}{2}$  the frequency response

$$H(\omega) = -i \operatorname{sgn}(\omega) e^{-i\frac{\omega}{2}}$$

leads to a simpler-looking expression;

$$h(t) = \frac{1}{\pi(t+\frac{1}{2})}$$

but simplicity can be deceptive, and for the same amount of computation odd order Hilbert filters have less error than even ones.

The trick in designing a Hilbert filter is bandwidth reduction, that is, requiring that it perform the 90° phase shift only for the frequencies absolutely required. Then the frequency response plotted in Figure 7.6 can be used as the design goal, rather than the discontinuous one of equation (7.12).

#### EXERCISES

- 7.3.1 Generate a signal composed of a small number of sinusoids and approximate it in a small interval by a polynomial. Compare the true derivative to the polynomial's derivative.
- 7.3.2 What are the frequency responses of the polynomial smoother and differentiator? How does the filter length affect the frequency response?
- 7.3.3 What is the ratio between the Riemann sum integration gain and the gain of an ideal integrator? Can you explain this result?
- 7.3.4 Show that the odd order Hilbert filter when discretized to integer times has all even coefficients zero.

### 7.4 Feedback

While FIR filters can be implemented in a *feedforward* manner, with the input signal flowing through the system in the forward direction, IIR filters employ *feedback*. Feedforward systems are simple in principle. An FIR with N coefficients is simply a function from its N inputs to a single output; but feedback systems are not static functions; they have dynamics that make them hard to predict and even unstable. However, we needn't despair as there are properties of feedback systems that can be easily understood.

In order to better understand the effect of feedback we will consider the simplest case, that of a simple amplifier with instantaneous feedback. It is helpful to use a graphical representation of DSP systems that will be studied in detail in Chapter 12; for now you need only know that in Figure 7.7 an arrow with a symbol above it represents a gain, and a circle with a plus sign depicts an adder.

Were it not for the feedback path (i.e., were a = 0) the system would be a simple amplifier y = Gx; but with the feedback we have

$$y = Gw \tag{7.15}$$

where the intermediate signal is the sum of the input and the feedback.

$$w = x + ay \tag{7.16}$$

Substituting

$$y = G(x + ay) = Gx + aGy$$

and solving for the output

$$y = \frac{G}{1 - aG} x \tag{7.17}$$

we see that the overall system is an amplifier like before, only the gain has been enhanced by a denominator. This gain obtained by closing the feedback



Figure 7.7: The DSP diagram of an amplifier with instantaneous feedback. As will be explained in detail in Chapter 12, an arrow with a symbol above it represents a gain, a symbol above a filled circle names a signal, and a circle with a plus sign depicts an adder. The feedforward amplifier's gain is G while the feedback path has gain (or attenuation) a.



**Figure 7.8:** An amplifier with delayed feedback. As will be explained in detail in Chapter 12, a circle with  $z^{-N}$  stands for a delay of N time units. Here the feedforward amplifier's gain is G while the feedback path has delay of N time units and gain (or attenuation) a.

loop is called the *closed loop gain*. When a is increased above zero the closed loop gain increases.

What if a takes precisely the value  $a = \frac{1}{G}$ ? Then the closed loop gain explodes! We see that even this simplest of examples produces an instability or 'pole'. Physically this means that the system can maintain a finite output even with zero input. This behavior is quite unlike a normal amplifier; actually our system has become an *oscillator* rather than an amplifier. What if we subtract the feedback from the input rather than adding it? Then for  $a = \frac{1}{G}$  the output is exactly zero.

The next step in understanding feedback is to add some delay to the feedback path, as depicted in Figure 7.8. Now

$$y_n = Gw_n$$

with

$$w_n = x_n + a y_{n-N}$$

where N is the delay time. Combining

$$y_n = G(x_n + ay_{n-N}) = Gx_n + aGy_{n-N}$$
(7.18)

and we see that for constant signals nothing has changed. What happens to time-varying signals? A periodic signal  $x_n$  that goes through a whole cycle, or any integer number of whole cycles, during the delay time will cause the feedback to precisely track the input. In this case the amplification will be exactly like that of a constant signal. However, consider a sinusoid that goes through a half cycle (or any odd multiple of half cycles) during the delay time. Then  $y_{n-N}$  will be of opposite sign to  $y_n$  and the feedback will destructively combine with the input; for aG = 1 the output will be zero! The same is true for a periodic signal that goes through a full cycle (or any multiple) during the delay time, with negative feedback (i.e., when the feedback term is subtracted from rather than added to the input).

So feedback with delay causes some signals to be emphasized and others to be attenuated, in other words, feedback can *filter*. When the feedback produces a pole, that pole corresponds to some frequency, and only that frequency will build up without limit. When a 'zero' is evoked, no matter how much energy we input at the particular frequency that is blocked, no output will result. Of course nearby frequencies are also affected. Near a pole sinusoids experience very large but finite gains, while sinusoids close to a zero are attenuated but not eliminated.

With unity gain negative feedback it is possible to completely block a sinusoid; can this be done with  $aG \neq 1$ ? For definiteness let's take  $G = 1, a = \frac{1}{2}$ . Starting at the peak of the sinusoid  $x_0 = 1$  the feedback term to be subtracted a cycle later is only  $ay_{n-N} = \frac{1}{2}$ . Subtracting this leads to  $w = \frac{1}{2}$ , which a cycle later leads to the subtraction of only  $ay_{n-N} = \frac{1}{4}$ . In the steady state the gain settles down to  $\frac{2}{3}$ , the prediction of equation (7.17) with a taken to be negative. So nonunity gain in the negative feedback path causes the sinusoid to be attenuated, but not notched out. You may easily convince yourself that the gain can only be zero if aG = 1. Similarly nonunity gain in a positive feedback path causes the sinusoid to be amplified, but not by an infinite amount.

So a sinusoid cannot be completed blocked by a system with a delayed negative feedback path and nonunity feedback gain, but is there a nonsinusoidal signal that *is* completely notched out? The only way to compensate for nonunity gain in the feedback term to be subtracted is by having the signal vary in the same way. Hence for aG > 1 we need a signal that increases by a factor of aG after the delay time N, i.e.,

$$s_n = e^{+(\ln aG)\frac{n}{N}} \sin\left(2\pi \frac{n}{N}\right)$$

while for aG < 1 the signal needs to decrease in the same fashion. This is a general result; when the feedback gain is not unity the signals that are optimally amplified or notched are exponentially growing or damped sinusoids.

Continuing our argument it is easy to predict that if there are several delayed feedback paths in parallel then there will be several frequency regions that are amplified or attenuated. We may even put filters in the feedback path, allowing feedback at certain frequencies and blocking it at others. Indeed this is the way filters are designed in analog signal processing; feedback paths of various gains and phases are combined until the desired effect is approximated.



Figure 7.9: The general feedback amplifier. The boxes represent general filters, with transfer functions as marked. The amplifier's transfer function is H while that of the feedback path is F.

In the most general setting, consider a digital system with transfer function  $H(z^{-1})$  to which we add a feedback loop with transfer function  $F(z^{-1})$ , as depicted in Figure 7.9. The closed loop transfer function is given by

$$H'(\mathbf{z}^{-1}) = \frac{H(\mathbf{z}^{-1})}{1 - F(\mathbf{z}^{-1})H(\mathbf{z}^{-1})}$$
(7.20)

which has a pole whenever the denominator becomes zero (i.e., for those z for which  $F(z^{-1})H(z^{-1}) = 1$ ). The value of z determines the frequency of the oscillation.

#### EXERCISES

- 7.4.1 When the microphone of an amplification system is pointed toward the speaker a squealing noise results. What determines the frequency of the squeal? Test your answer. What waveform would you expect?
- 7.4.2 A feedback pole causes an oscillation with frequency determined by the delay time. This oscillation is sustained even without any input. The system is linear and time-invariant, and so is a filter; as a filter it cannot create energy at a frequency where there was no energy in the input. Resolve this paradox.
- 7.4.3 What is the effect of a delayed feedback path with unity gain on a sinusoid of frequency close, but not equal, to the instability? Plot the gain as a function of frequency (the frequency response).
- 7.4.4 Find a signal that destabilizes a system with a delayed positive feedback path and nonunity feedback gain.
- 7.4.5 Show that for  $G = \frac{1}{2}$ , a = 1 a sinusoid of frequency corresponding to the delay is amplified by the gain predicted by equation (7.17).

- 7.4.6 What is the effect of a delayed feedback path with nonunity gain G on a sinusoid of frequency corresponding to the delay? Plot the effective gain as a function of G.
- 7.4.7 Simulate a system that has a causal MA filter in the feedback path. Start with a low-pass filter, then a high-pass, and finally a band-pass. Plot the frequency response.

# 7.5 The ARMA Transfer Function

In Section 6.14 we defined the transfer function of a filter. The transfer function obeys

$$Y(z) = H(z)X(z)$$

where X(z) is the zT of the input to the filter and Y(z) is the zT of the output. Let's find the transfer function of an ARMA filter. The easiest way to accomplish this is to take the z transform of both sides of the general ARMA filter in the symmetric form (equation (6.46))

$$\sum_{m=0}^{M} \beta_m y_{n-m} = \sum_{l=0}^{L} \alpha_l x_{n-l}$$

the zT of the left side being

$$\sum_{n=-\infty}^{\infty} \left\{ \sum_{m=0}^{M} \beta_m y_{n-m} \right\} z^{-n} = \sum_{m=0}^{M} \beta_m \sum_{n=-\infty}^{\infty} y_{n-m} z^{-n} = \sum_{m=0}^{M} \beta_m z^{-m} \sum_{\nu=-\infty}^{\infty} y_{\nu} z^{-\nu} = \left\{ \sum_{m=0}^{M} \beta_m z^{-m} \right\} Y(z)$$

and similarly that of the right side.

$$\sum_{n=-\infty}^{\infty} \left\{ \sum_{l=0}^{L} \alpha_{l} x_{n-l} \right\} z^{-n} = \sum_{l=0}^{L} \alpha_{l} \sum_{n=-\infty}^{\infty} x_{n-l} z^{-n} = \sum_{l=0}^{L} \alpha_{l} z^{-l} \sum_{\nu=-\infty}^{\infty} x_{\nu} z^{-\nu} = \left\{ \sum_{l=0}^{L} \alpha_{l} z^{-l} \right\} X(z)$$

Putting these together

$$\left\{\sum_{m=0}^{M}\beta_m z^{-m}\right\}Y(z) = \left\{\sum_{l=0}^{L}\alpha_l z^{-l}\right\}X(z)$$

#### FILTERS

and comparing with equation (6.65) we find that the transfer function is the ratio of two polynomials in  $z^{-1}$ .

$$H(z) = \frac{\sum_{l=0}^{L} \alpha_l z^{-l}}{\sum_{m=0}^{M} \beta_m z^{-m}}$$
(7.21)

This can be also expressed in terms of the coefficients in equation (6.45)

$$H(z) = \frac{\sum_{l=0}^{L} a_l z^{-l}}{1 - \sum_{m=1}^{M} b_m z^{-m}}$$
(7.22)

a form that enables one to build the transfer function 'by inspection' from the usual type of difference equation.

For an AR filter L = 0 and neglecting an uninteresting gain (i.e., taking  $a_0 = 1$ )

$$H(z) = \frac{1}{1 - \sum_{m=1}^{M} b_m z^{-m}}$$
(7.23)

while for an MA filter all the  $b_m$  are zero and the transfer function is a polynomial.

$$H(z) = \sum_{l=0}^{L} a_l z^{-l} \tag{7.24}$$

It is often burdensome to have to deal with polynomials in  $z^{-1}$ , so we express the transfer function in terms of z instead.

$$H(z) = z^{M-L} \frac{\sum_{l=0}^{L} \alpha_l z^{L-l}}{\sum_{m=0}^{M} \beta_m z^{M-m}}$$
(7.25)

We see that H(z) is a rational function of z.

The fact that the transfer function H(z) of the general ARMA filter is a rational function, has interesting and important ramifications. The fundamental theorem of algebra tells us that any polynomial of degree M can be completely factored over the complex numbers

$$\sum_{i=0}^{D} c_i x^i = G \prod_{i=1}^{D} (x - \zeta_i)$$

where the D roots  $\zeta_i$  are in general complex numbers. When the coefficients  $c_i$  are real, the sum itself is always real, and so the roots must either be real, or appear in complex conjugate pairs. Thus we can rewrite the transfer function of the general ARMA filter as

$$H(z) = G \frac{\prod_{l=1}^{L} (z - \zeta_l)}{\prod_{m=1}^{M} (z - \pi_m)}$$
(7.26)

294

where the roots of the numerator  $\zeta_l$  are called 'zeros' of the transfer function, and the roots of the denominator  $\pi_m$  its 'poles'. So other than an simple overall gain G, we need only specify the zeros and poles to completely determine the transfer function; no further information is needed.

From equations 7.23 and 7.24 we see that the transfer function of the MA filter has zeros but no poles while that of the AR filter has poles but no zeros. Hence the MA filter is also called an all-zero filter and the AR filter is called an all-pole filter.

What is the meaning of these zeros and poles? A zero of the transfer function is a complex number  $\zeta = re^{i\omega}$  that represents a complex (possibly decaying or increasing) exponential signal that is attenuated by the ARMA filter. Poles  $\pi$  represent complex exponential signals that are amplified by the filter. If a zero or pole is on the unit circle, it represents a sinusoid that is either completely blocked by the filter or destabilizes it.

Since the positions in the complex plane of the zeros and poles provide a complete description of the transfer function of the general ARMA system, it is conventional to graphically depict them using a *pole-zero plot*. In such plots the position of a zero is shown by a small filled circle and a pole is marked with an X. Poles or zeros at z = 0 or  $z = \infty$  that derive from the  $z^{M-L}$  factor in equation (7.25) are not depicted, but multiple poles and/or zeros at the same position are. This single diagram captures everything one needs to know about a filter, except for the overall gain.

A few examples are in order. First consider the causal equally weighted L+1-point average MA filter (since we intend to discard the gain we needn't normalize the sum).

$$y_n = \sum_{l=0}^L x_{n-l}$$

By inspection the transfer function is

$$H(z) = \sum_{l=0}^{L} z^{-l} = \frac{1 - z^{-1^{-L-1}}}{1 - z^{-1}} = \frac{1}{z^{L}} \frac{z^{L+1} - 1}{z - 1}$$

and we seem to see L poles at the origin, the L+1 zeros of  $z^{L+1}-1$  and a pole at z = 1. The zeros are the L + 1 roots of unity,  $z = e^{i2\pi \frac{k}{L+1}}$ , one of which is z = 1 itself; hence that zero cancels the putative pole at z = 1. The Lpoles at the origin are meaningless and may be ignored. We are therefore left with L zeros equally spaced around the unit circle (not including z = 1), as displayed in Figure 7.10.A. It is not difficult to verify that the corresponding sinusoids are indeed blocked by the averaging MA filter.



Figure 7.10: The pole-zero plots of two simple systems. In (A) we see the pole-zero plot for the MA filter that averages with equal weights eight consecutive input values. In (B) is the simple AR low-pass filter  $y_n = (1 - \beta)x_n + \beta y_{n-1}$ .

Our second example is our favorite AR filter of equation (6.39).

$$y_n = (1 - \beta)x_n + \beta y_{n-1} \qquad 0 \le \beta < 1$$

By inspection we can write

$$H(z) = \frac{(1-\beta)}{1-\beta z^{-1}} = z\frac{1-\beta}{z-\beta}$$

which has a trivial zero at the origin and a single pole at  $\beta$ , as depicted in Figure 7.10.B.

As our last example we choose a general first-order section, that is, an ARMA system with a single zero and a single pole.

$$y_n = a_0 x_n + a_1 x_{n-1} + b_1 y_{n-1}$$

This is a useful system since by factorization of the polynomials in both the numerator and denominator of the transfer function we can break down any ARMA filter into a sequence of first-order sections in cascade. By inspection the transfer function

$$H(z) = \frac{a_0 + a_1 z^{-1}}{1 - b_1 z^{-1}} = a_0 \frac{z + \frac{a_1}{a_0}}{z - b_1}$$

has its zero at  $z = -\frac{a_1}{a_0}$  and its pole at  $z = b_1$ . To find the frequency response we substitute  $z = e^{i\omega}$ 

$$H(\omega) = \frac{a_0 + a_1 e^{-i\omega}}{1 - b_1 e^{-i\omega}}$$

which at DC is  $\frac{a_0+a_1}{1-b_1}$  and at Nyquist  $\omega = \pi$  is  $\frac{a_0-a_1}{1+b_1}$ . To find the impulse response we need the inverse zT, which generally is difficult to calculate. Here it can be carried out using a trick

$$H(z) = a_0 \frac{(z - b_1) + (\frac{a_1}{a_0} + b_1)}{z - b_1}$$
  
=  $a_0 \left( 1 + \frac{(\frac{a_1}{a_0} + b_1)z^{-1}}{1 - b_1 z^{-1}} \right)$   
=  $a_0 \left( 1 + (\frac{a_1}{a_0} + b_1)z^{-1} + (\frac{a_1}{a_0} + b_1)b_1 z^{-2} + (\frac{a_1}{a_0} + b_1)b_1^2 z^{-3} + \dots \right)$ 

and the desired result is obtained.

$$h_n = \begin{cases} a_0 & n = 0\\ (a_1 + a_0 b_1) b_1^{n-1} & n \neq 0 \end{cases}$$

#### EXERCISES

- 7.5.1 Sometimes it is useful to write difference equations as  $y_n = Gx_n + \sum a_l x_{n-l} + \sum b_m y_{n-m}$  where G is called the *gain*. Write the transfer function in rational-function- and factored-form for this case.
- 7.5.2 Derive equation (7.21) more simply than in the text by using the time shift relation for the zT.
- 7.5.3 Consider the system with a single real pole or zero. What signal is maximally amplified or attenuated? Repeat for a complex pole or zero.
- 7.5.4 Calculate the transfer function H(z) for the noncausal MA system of equation (6.35). Relate this to the transfer function of the causal version and to the frequency response (equation (6.36)) previously calculated.
- 7.5.5 Show that stable ARMA filters have all their poles inside the unit circle.
- 7.5.6 Prove that real all-pass filters have poles and zeros in conjugate reciprocal locations.
- 7.5.7 Show that the first-order section is stable when  $|b_1| < 1$  both by considering the pole and by checking the impulse response.
- 7.5.8 Plot the absolute value of the frequency response of the first-order section for frequencies between DC and Nyquist. When is the filter low-pass (passes low frequencies better than highs)?
- 7.5.9 If the maximum input absolute value is 1, what is the maximal output absolute value for the first-order section? If the input is white noise of variance 1, what is the variance of the output of the first-order section?

7.5.10 In general, when breaking down ARMA systems into first-order sections the zeros and poles may be complex. In such cases we most often use real-valued second-order sections instead.

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 - b_1^{-1} - b_2 z^{-2}}$$

What is the frequency response for the second-order section with complex conjugate poles?

### 7.6 Pole-Zero Plots

The main lesson from the previous section was that the positions of the zeros and poles of the transfer function determine an ARMA filter to within a multiplicative gain. The graphical depiction of these positions such as in Figure 7.10 is called a pole-zero plot. Representing filters by pole-zero plots is analogous to depicting signals by the z-plane plots introduced in Section 4.10. Indeed there is a unique correspondence between the two since z-plane plots contain a complete frequency domain description of signals, and filters are specified by their effect in the frequency domain.

The pole-zero plot completely specifies an ARMA filter except for the overall gain. At first sight it may seem strange that the positions of the zeros and poles are enough to completely specify the transfer function of an ARMA filter. Why can't there be two transfer functions that have the same zeros and poles but are different somewhere far from these points? The fact is that were we to allow arbitrary systems then there could indeed be two different systems that share zeros and poles; but the transfer function of an ARMA filter is constrained to be a rational function and the family of rational functions does not have that much freedom. For instance, suppose we are given the position of the zeros of an MA filter,  $\zeta_1, \zeta_2 \dots \zeta_L$ . Since the transfer function is a polynomial, is must be

$$H(z) = G \prod_{l=1}^{L} (z - \zeta_l)$$

since any other polynomial will have different zeros.

In addition to being mathematically sufficient, pole-zero plots are graphically descriptive. The pole-zero plot provides the initiated at a glance everything there is to know about the filter. You might say that the pole-zero plot picture is worth a thousand equations. It is therefore worthwhile to become proficient in 'reading' pole-zero plots.

We can place restrictions on the poles and zeros before we even start. Since we wish real inputs to produce real outputs, we require all the coefficients of the ARMA filter to be real. Now real-valued rational functions will have poles and zeros that are either real valued, or that come in complex conjugates. For example, the three zeros 1, 1 + i and 1 - i form the real polynomial  $(z-1)(z-i)(z+i) = z^3 - z^2 + z - 1$ , while were the two complex zeros not complex conjugates the resulting polynomial would be complex! So the pole-zero plots of ARMA systems with real-valued coefficients are always mirror-symmetric around the real axis.

What is the connection between the pole-zero plots of a system and its inverse? Recall from equation (6.17) that when the output of a system is input to its inverse system the original signal is recovered. In exercise 6.14.3 we saw that the transfer function of the concatenation of two systems is the product of their respective transfer functions. So the product of the transfer functions of a system and its inverse must be unity, and hence the transfer functions reciprocals of each other. Hence the pole-zero plot of the inverse system is obtained by replacing all poles with zeros and all zeros with poles. In particular it is easy now to see that the inverse of an all-zero system is all-pole and vice versa.

In Section 7.4 we saw what it means when a pole or a zero is on the unit circle. A zero means that the frequency in question is swallowed up by the system, and nearby frequencies are attenuated. A pole means that the system is capable of steady state output without input at this frequency, and nearby frequencies are strongly amplified. For this reason poles on the unit circle are almost always to be avoided at all costs.

What if a pole or zero is inside the unit circle? Once again Section 7.4 supplied the answer. The signal that is optimally amplified or blocked is a damped sinusoid, exactly the basic signal represented by the pole or zero's position in the z-plane. If the pole or zero is outside the unit circle the signal most affected is the growing sinusoid represented by that point. Although we don't want poles on the unit circle, we want them even less outside it. A pole corresponding to an exponentially growing sinusoid would mean that we have an unstable system that could explode without notice. Thus IIR system designers must always ensure that all poles are inside the unit circle.

FILTERS

The pole-zero plot directly depicts the transfer function, but the frequency response is also easily inferred. Think of the unit circle as a circular railroad track with its height above sea level representing the gain at the corresponding frequency. In this analogy poles are steep mountains and zeros are craters. As the train travels around the track its height increases and decreases because of proximity to a mountain or crater. Of course at any position there may be several poles and/or craters nearby, and the overall height is influenced by each of them according to its distance from the train. Now let's justify this analogy. Substituting  $z = e^{i\omega}$  into equation (7.26) we find that the frequency response of an ARMA systems is

$$H(\omega) = G \frac{\prod_{l=1}^{L} (e^{i\omega} - \zeta_l)}{\prod_{m=1}^{M} (e^{i\omega} - \pi_m)}$$
(7.27)

with magnitude and angle given by the following.

$$|H(\omega)| = G \frac{\prod_{l=1}^{L} |e^{i\omega} - \zeta_l|}{\prod_{m=1}^{M} |e^{i\omega} - \pi_m|}$$
  
$$\angle H(\omega) = \sum_{l=1}^{L} \angle (e^{i\omega} - \zeta_l) - \sum_{m=1}^{M} \angle (e^{i\omega} - \pi_m)$$

The  $l^{\text{th}}$  factor in the numerator of the magnitude is the distance between the point on the unit circle and the  $l^{\text{th}}$  zero, and the  $m^{\text{th}}$  factor in the denominator is the distance to the  $m^{\text{th}}$  pole. The magnitude is seen to be the product of the distances to all the zeros divided by the product to all the poles. If one of the zeros or poles is very close it tends to dominate, but in general the train's height is influenced by all the mountains and craters according to their distances from it. The  $l^{\text{th}}$  term in the numerator of the angle is the direction of the vector between the point on the unit circle and the  $l^{\text{th}}$  zero, and the  $m^{\text{th}}$  term in the denominator is the angle to the  $m^{\text{th}}$ pole. Therefore the phase of the frequency response is seen to be the sum of the angles to all the zeros minus the sum of the angles to the poles. If one of the zeros or poles is very close its angle changes rapidly as the train progresses, causing it to dominate the group delay.

Suppose we design a filter by some technique and find that a pole is outside the unit circle. Is there some way to stabilize the system by moving it back inside the unit circle, without changing the frequency response? The answer is affirmative. Let the pole in question be  $\pi_0 = Pe^{i\theta}$ . You can convince yourself that the distance from any point on the unit circle to  $\pi_0$  is exactly  $P^2$  times the distance to  $\pi'_0 = \frac{1}{P}e^{i\theta}$ , the point along the same radius

but with reciprocal magnitude. Thus to within a gain term (that we have been neglecting here) we can replace any pole outside the unit circle with its 'reciprocal conjugate'  $\pi'_0$ . This operation is known as 'reflecting a pole'. We can also reflect a zero from outside the unit circle inward, or from the inside out if we so desire. For real filters we must of course reflect both the pole and its complex conjugate.

Let's see how the concept of a pole-zero plot enables us to design some useful filters. Assume we want a DC blocker, that is, a filter that blocks DC but passes AC frequencies. A first attempt might be to simply place a zero at DC

$$H(z) = z - 1 = z(1 - z^{-1}) \qquad \Rightarrow \qquad y_n = x_n - x_{n-1}$$

discarding the term representing a zero at z = 0; but this filter is simply the finite difference, with frequency response

$$|H(\omega)|^2 = |1 - e^{-i\omega}|^2 = 2(1 - \cos \omega)$$

not corresponding to a sharp notch. We can sharpen the response by placing a pole on the real axis close to, but inside, the unit circle. The reasoning behind this tactic is simple. The zero causes the DC frequency response to be zero, but as we move away from  $\omega = 0$  on the unit circle we immediately start feeling the effects of the pole.

$$H(z) = \frac{z-1}{z-\beta} = \frac{1-z^{-1}}{1-\beta z^{-1}} \implies y_n = \beta y_{n-1} + (x_n - x_{n-1})$$

Here  $\beta < 1$  but the closer  $\beta$  is to unity the sharper the notch will be. There is a minor problem regarding the gain of this filter. We would like the gain to be unity far away from DC, but of course pole-zero methods cannot control the gain. At z = -1 our DC blocker has a gain of

$$\frac{z-1}{z-\beta} = \frac{-1-1}{-1-\beta} = \frac{1}{1-\frac{\alpha}{2}}$$

where we defined the small positive number  $\alpha$  via  $\beta = 1 - \alpha$ . We can compensate for this gain by multiplying the x terms by a factor  $g = 1 - \frac{\alpha}{2}$ .

$$y_n = (1 - \alpha) y_{n-1} + (1 - \frac{1}{2}\alpha) (x_n - x_{n-1})$$

In addition to a DC blocker we can use the same technique to make a notch at any frequency  $\Omega$ . We need only put a conjugate pair of zeros on the unit circle at angles corresponding to  $\pm \Omega$  and a pair of poles at the same

302 FILTERS

angles but slightly reduced radius. We can also make a sharp band-pass filter by reversing the roles of the zeros and poles. Wider band-pass or band-stop filters can be approximated by placing several poles and/or zeros along the desired band. Every type of frequency-selective filter you can imagine can be designed by careful placement of poles and zeros.

#### EXERCISES

- 7.6.1 Although practically every filter you meet in practice is ARMA, they are *not* the most general LTI system. Give an example of a linear time-invariant system that is not ARMA.
- 7.6.2 Make a pole-zero plot for the system

$$H(z) = \frac{(z-\alpha)(z-\frac{1}{\alpha})}{(z-r\alpha)(z-\frac{r}{\alpha})}$$

where  $\alpha = e^{\mathbf{i}\Omega}$  and  $r \stackrel{<}{\sim} 1$ . Sketch the frequency response. What kind of filter is this?

- 7.6.3 Why did we call  $\pi'_0$  the reciprocal conjugate? Prove that the distance from any point on the unit circle to  $\pi_0$  is exactly  $P^2$  times the distance to the reciprocal conjugate  $\pi'_0$ .
- 7.6.4 A stable system whose inverse is stable as well is said to be *minimum phase*. What can you say about the pole-zero plot of a minimum phase system?
- 7.6.5 Prove that reflecting poles (or zeros) does not change the frequency response.
- 7.6.6 What can be said about the poles and zeros of an all-pass filter? What is the connection between this question and the previous one?
- 7.6.7 A notch filter can be designed by adding the outputs of two all-pass filters that have the same phase everywhere except in the vicinity of the frequency to be blocked, where they differ by 180°. Design a notch filter of the form  $H(z) = \frac{1}{2}(1 + A(z))$  where A(z) is the transfer function of an all-pass filter. How can you control the position and width of the notch?
- 7.6.8 Consider the DC blocking IIR filter  $y_k = 0.9992(x_k x_{k-1}) + 0.9985y_{k-1}$ . Draw its frequency response by inputting pure sinusoids and measuring the amplitude of the output. What is its pole-zero plot?

# 7.7 Classical Filter Design

Classical filter design means analog filter design. Why are we devoting a section in a book on DSP to analog filter design? There are two reasons. First, filtering is one of the few select subjects in analog signal processing about which every DSP expert should know something. Not only are there always analog antialiasing filters and reconstruction filters, but it is often worthwhile to perform other filtering in the analog domain. Good digital filters are notoriously computationally intensive, and in high-bandwidth systems there may be no alternative to performing at least some of the filtering using analog components. Second, the discipline of analog filter design was already well-developed when the more complex field of digital filter design was first developing. It strongly influenced much of the terminology and algorithms, although its stranglehold was eventually broken.



Figure 7.11: Desired frequency response of the analog low-pass filter to be designed. The pass-band is from f = 0 to the pass-band edge  $f_p$ , the transition region from  $f_p$  to  $f_s$ , and the stop-band from the top-band edge  $f_s$  to infinity. The frequency response is halfway between that of the pass-band and that of the stop-band at the cutoff frequency  $f_c$ . The maximal ripple in the pass-band is  $\delta_p$  and in the stop-band  $\delta_s$ .

We will first focus on the simplest case, that of an analog low-pass filter. Our ideal will be the ideal low-pass filter, but that being unobtainable we strive toward its best approximation. The most important specification is the cutoff frequency  $f_c$ , below which we wish the signal to be passed, above which we wish the signal to be blocked. The pass-band and stop-band are separated by a transition region where we do not place stringent requirements on the frequency response. The end of the pass-band is called  $f_p$  and the beginning

#### FILTERS

of the stop-band  $f_s$ . Other specifications for a practical implementation are the transition width  $\Delta = f_s - f_p$ , the maximal deviation from unity gain in the pass-band  $\delta_p$ , and the maximal amplitude in the stop-band  $\delta_s$ . In a typical analog filter design problem  $f_c$  (or  $f_p$  or  $f_s$ ) and the maximal allowed values for  $\Delta$ ,  $\delta_p$ , and  $\delta_s$  are given. Figure 7.11 depicts the ideal and approximate analog low-pass filters with these parameters.

Designing an analog filter essentially amounts to specifying the function H(f) whose square is depicted in the figure. From the figure and our previous analysis we see that

$$\begin{aligned} |H(0)|^2 &= 1 \\ |H(f)|^2 &\approx 1 & \text{for } f < f_c \\ |H(f)^2 &\approx 0 & \text{for } f > f_c \\ |H(f)|^2 &\to 0 & \text{for } f \to \infty \end{aligned}$$

are the requirements for an analog low-pass filter. The first functional forms that come to mind are based on arctangents and hyperbolic tangents, but these are natural when the constraints are at plus and minus infinity, rather than zero and infinity. Classical filter design relies on the form

$$|H(f)|^2 = \frac{1}{1+p(f)}$$
(7.28)

where p(f) is a polynomial that must obey

$$p(0) = 0$$
  
$$p(f) \xrightarrow{f \to \infty} \infty$$

and be well behaved. The classical design problem is therefore reduced to the finding of this polynomial.

In Figure 7.11 the deviation of the amplitude response from the ideal response is due entirely to its smoothly decreasing from unity at f = 0 in order to approach zero at high frequencies. One polynomial that obeys the constraints and has no extraneous extrema is the simple quadratic

$$p(f) = \left(\frac{f}{f_c}\right)^2$$

which when substituted back into equation (7.28) gives the 'slowest' filter depicted in Figure 7.12. The other filters there are derived from

$$p(f) = \left(\frac{f}{f_c}\right)^{2N}$$



Figure 7.12: Frequency response of analog Butterworth low-pass filters. From bottom to top at low frequencies we have order  $N = 1, 2, 3, 5, 10, 25, \infty$ .

and are called the Butterworth low-pass filters of order n. It is obvious from the figure that the higher n is the narrower the transition.

Butterworth filters have advantages and disadvantages. The attenuation monotonically increases from DC to infinite frequency; in fact the first 2N-1 derivatives of  $|H(f)|^2$  are identically zero at these two points, a property known as 'maximal flatness'. An analog Butterworth filter has only poles and is straightforward to design. However, returning to the design specifications, for the transition region  $\Delta$  to be small enough the order N usually has to be quite high; and there is no way of independently specifying the rest of the parameters.

In order to obtain faster rolloff in the filter skirt we have to give something up, and that something is the monotonicity of  $|H(f)|^2$ . A Butterworth filter 'wastes' a lot of effort in being maximally flat, effort that could be put to good use in reducing the size of the transition region. A filter that is allowed to oscillate up and down a little in either the pass-band, the stop-band or both can have appreciably smaller  $\Delta$ . Of course we want the deviation from our specification to be minimal in some sense. We could require a minimal squared error between the specification and the implemented filter

$$\epsilon^2 = \int |H_{spec}(\omega) - H_{impl}(\omega)|^2 d\omega$$

but this would still allow large deviation from specification at some frequencies, at the expense of overexactness at others. It makes more sense to require *minimax error*, i.e., to require that the maximal deviation from specification

$$\max_{\omega} |H_{spec}(\omega) - H_{impl}(\omega)|$$

306



Figure 7.13: Frequency response of low-pass equiripple designs. In (A) we see an FIR filter designed using the Remez algorithm for comparison purposes. In (B) we the IIR Chebyshev design, in (C) the inverse Chebyshev and in (D) the elliptical design.

be minimal. Achieving true minimax approximation is notoriously difficult in general, but approximation using Chebyshev polynomials (see Appendix A.10) is almost the same and straightforward to realize. This approximation naturally leads to equiripple behavior, where the error oscillates around the desired level with equal error amplitude, as shown in Figure 7.13.

The Chebyshev (also known as Chebyshev I) filter is equiripple in the pass-band, but maximally flat in the stop-band. It corresponds to choosing the polynomial

$$p(f) = \delta^2 T_N^2 \left(\frac{f}{f_p}\right)$$

and like the Butterworth approximation, the analog Chebyshev filter is allpole. The inverse Chebyshev (or Chebyshev II) filter is equiripple in the stop-band but maximally flat in the pass-band. Its polynomial is

$$p(f) = \delta^2 \frac{T_N^2 \left(\frac{f_s}{f_p}\right)}{T_N^2 \left(\frac{f_s}{f}\right)}$$

The Chebyshev filter minimax approximates the desired response in the pass-band but not in the stop-band, while the inverse Chebyshev does just

the opposite. For both types of Chebyshev filters the parameter  $\delta$  sets the ripple in the equiripple band. For the inverse Chebyshev, where the equiripple property holds in the stop-band, the attenuation is determined by the ripple; lower ripple means higher stop-band rejection.

Finally, the *elliptical* filter is equiripple in both pass-band and stopband, and so approximates the desired response in the minimax sense for all frequencies. Its 'polynomial' is not a polynomial at all, but rather a rational function  $U_N(\frac{f}{f_p})$ . These functions are defined using the elliptical functions (see Appendices A.8 and A.10). Taking the idea from equation (A.59), we define the function

$$U_{r;k,q}(u) \equiv \operatorname{sn}_k\left(r\operatorname{sn}_q^{-1}(u)\right)$$
(7.29)

and when r and the complete elliptical integrals  $K_k$  and  $K_q$  obey certain relations that we will not go into here, this function becomes a rational function.

$$U_N(u) = a^2 \begin{cases} \frac{(u_1^2 - u^2)(u_3^2 - u^2) \cdots (u_{2N-1}^2 - u^2)}{(1 - u_1^2 u^2)(1 - u_3^2 u^2) \cdots (1 - u_{2N-1}^2 u^2)} & N \text{ even} \\ \frac{u(u_2^2 - u^2)(u_4^2 - u^2) \cdots (u_{2N-1}^2 u^2)}{(1 - u_2^2 u^2)(1 - u_4^2 u^2) \cdots (1 - u_{2N}^2 u^2)} & N \text{ odd} \end{cases}$$
(7.30)

This rational function has several related interesting characteristics. For u < 1 the function lies between -1 and +1. Next,

$$U_N\left(\frac{1}{u}\right) = \frac{1}{U_N(u)}$$

and its zeros and poles are reciprocals of each other. Choosing all the N zeros in the range  $0 < \zeta < 1$  forces all N poles to fall in the range  $1 < \pi < \infty$ . Although the zeros and poles are not equally spaced, the behavior of

$$|H(f)|^2 = \frac{1}{1 + U_N(\frac{f}{f_p})}$$

is equiripple in both the pass-band and the stop-band.

It is useful to compare the four types of analog filter—Butterworth, Chebyshev, inverse Chebyshev, and elliptical. A very strong statement can be made (but will not be proven here) regarding the elliptical filter; given any three of the four parameters of interest (pass-band ripple, stop-band ripple, transition width, and filter order) the elliptical filter minimizes the remaining parameter. In particular, for given order N and ripple tolerances the elliptical filter can provide the steepest pass-band to stop-band transition. The Butterworth filter is the weakest in this regard, and the two Chebyshev types are intermediate. The Butterworth filter, however, is the best approximation to the Taylor expansion of the ideal response at both DC and infinite frequency. The Chebyshev design minimizes the maximum pass-band ripple, while the inverse Chebyshev maximizes the minimum stop-band rejection.

The design criteria as we stated them do not address the issue of phase response, and none of these filters is linear-phase. The elliptical has the worst phase response, oscillating wildly in the pass-band and transition region (phase response in the stop-band is usually unimportant). The Butterworth is the smoothest in this regard, followed by the Chebyshev and inverse Chebyshev.

Although this entire section focused on analog low-pass filter, the principles are more general. All analog filters with a single pass-band and/or stop-band can be derived from the low-pass designs discussed above. For example, we can convert analog low-pass filter designs into high-pass filters by the simple transformation  $f \rightarrow \frac{1}{f}$ . Digital filters are a somewhat more complex issue, to be discussed in the next section. For now it is sufficient to say that IIR filters are often derived from analog Butterworth, Chebyshev, inverse Chebyshev, or elliptical designs. The reasoning is not that such designs are optimal; rather that the theory of the present section predated DSP and early practitioners prefered to exploit well-developed theory whenever possible.

#### EXERCISES

- 7.7.1 Show that a Butterworth filter of order N is maximally flat.
- 7.7.2 All Butterworth filters have their half gain (3 dB down) point at  $f_c$ . Higher order N makes the filter gain decrease faster, and the speed of decrease is called the 'rolloff'. Show that for high frequencies the rolloff of the Butterworth filter is 6 dB per octave (i.e., the gain decreases 6 dB for every doubling in frequency) or 20 dB per decade. How should N be set to meet a specification involving a pass-band end frequency  $f_p$ , a stop-band start frequency  $f_s$ , and a maximum error tolerance  $\delta$ ?
- 7.7.3 Show that the 2N poles of  $|H(f)|^2$  for the analog Butterworth filter all lie on a circle of radius  $f_c$  in the s-plane, are equally spaced, and are symmetric with respect to the imaginary axis. Show that the poles of the Chebyshev I filter lie on an ellipse in the s-plane.
- 7.7.4 The HPNA 1.0 specification calls for a pulse consisting of 4 cycles of a 7.5 MHz square wave filtered by a five-pole Butterworth filter that extends from 5.5 MHz to 9.5 MHz. Plot this pulse in the time domain.

7.7.5 The frequency response of a certain filter is given by

$$|H(f)|^2 = \frac{f_c^{\alpha}}{f^{\alpha} + f_c^{\alpha}}$$

where  $\alpha$  and  $f_c$  are parameters. What type of filter is this and what is the meaning of the parameters?

7.7.6 Repeat the previous exercise for these filters.

$$|H(f)|^2 = \frac{f^{\alpha}}{f^{\alpha} + f^{\alpha}_c}$$
$$|H(f)|^2 = \frac{f^{\alpha} + f^{\alpha}_d}{f^{\alpha} + f^{\alpha}_c}$$

- 7.7.7 Show that in the pass-band the Chebyshev filter gain is always between  $\frac{1}{\sqrt{1-\delta^2}}$  and  $\frac{1}{\sqrt{1+\delta^2}}$  so that the ripple is about  $4\delta^2$  dB. Show that the gain falls monotonically in the stop-band with rolloff 20N dB per decade but always higher than the Butterworth filter of the same order.
- 7.7.8 We stated that an analog low-pass filter can be converted into a high-pass filter by a simple transformation of the frequency variable. How can band-pass and band-stop filters be similarly designed by transformation?

# 7.8 Digital Filter Design

We will devote only a single section to the subject of digital filter design, although many DSP texts devote several chapters to this subject. Although the theory of digital filter design is highly developed, it tends to be highly uninspiring, mainly consisting of techniques for constrained minimization of approximation error. In addition, the availability of excellent digital filter design software, both full graphic applications and user-callable packages, makes it highly unlikely that you will ever need to design on your own. The aim of this section is the clarification of the principles behind such programs, in order for the reader to be able to use them to full advantage.

Your first reaction to the challenge of filter design may be to feel that it is a trivial pursuit. It is true that finding the frequency response of a given filter is a simple task, yet like so many other inverse problems, finding a filter that conforms to a frequency specification is a more difficult problem. From a frequency domain specification we can indeed directly derive the impulse response by the FT, and the numeric values of the impulse response are 310

undeniably FIR filter coefficients; but such an approach is only helpful when the impulse response quickly dies down to zero. Also numeric transformation of N frequency values will lead to a filter that obeys the specification at the exact frequencies we specified, but at in-between frequencies the response may be far from what is desired. The main trick behind filter design is how to constrain the frequency response of the filter so that it does not significantly deviate from the specification at *any* frequency.

We should note that this malady is not specific to time domain filtering. Frequency domain filtering uses the FT to transfer the signal to the frequency domain, performs there any needed filtering operation, and then uses the iFT to return to the time domain. We can only numerically perform a DFT for a finite number of signal values, and thus only get a finite frequency resolution. Multiplying the signal in the frequency domain enforces the desired filter specification at these frequencies only, but at intermediate frequencies anything can happen. Of course we can decide to double the number of signal times used thus doubling the frequency resolution, but there would still remain intermediate frequencies where we have no control. Only in the limit of the LTDFT can we completely enforce the filter specification, but that requires knowing the signal values at all times and so is an unrealizable process.

At its very outset the theory and practice of digital filter design splits into two distinct domains, one devoted to general IIR filters, and the other restricted to linear-phase FIR filters. In theory the general IIR problem is the harder one, and we do not even know how to select the minimum number of coefficients that meet a given specification, let alone find the optimal coefficients. Yet in practice the FIR problem is considered the more challenging one, since slightly suboptimal solutions based on the methods of the previous section can be exploited for the IIR problem, but not for the FIR one.

Let's start with IIR filter design. As we mentioned before we will not attempt to directly optimize filter size and coefficients; rather we start with a classical analog filter design and bring it into the digital domain. In order to convert a classical analog filter design to a digital one, we would like to somehow digitize. The problem is that the z-plane is not like the analog (Laplace) s-plane. From Section 4.10 we know that the sinusoids live on the imaginary axis in the s-plane, while the periodicity of digital spectra force them to be on the unit circle in the z-plane. So although the filter was originally specified in the frequency domain we are forced to digitize it in the time domain. The simplest time domain property of a filter is its impulse response, and we can create a digital filter by evenly sampling the impulse response of any of the classical designs. The new digital filter's transfer function can then be recovered by z transforming this sampled impulse response. It is not hard to show that a transfer function thus found will be a rational function, and thus the digital filter will be ARMA. Furthermore the number of poles is preserved, and stable analog filters generate stable digital filters. Unfortunately, the frequency response of the digital filter will not be identical to that of the original analog filter, because of aliasing. In particular, the classical designs do not become identically zero at high frequencies, and so aliasing cannot be avoided. Therefore the optimal frequency domain properties of the analog designs are not preserved by impulse response sampling.

An alternative method of transforming analog filters into digital ones is the bilinear mapping method. The basic idea is to find a mapping from the *s*-plane to the *z*-plane and to convert the analog poles and zeros into the appropriate digital ones. For such a mapping to be valid it must map the imaginary axis  $s = i\omega$  onto the unit circle  $z = e^{i\omega}$ , and the left half plane into the interior of the unit circle. The mapping (called 'bilinear' since the numerator and denominator are both linear in s)

$$z = \frac{1+s}{1-s}$$
(7.31)

does just that. Unfortunately, being nonlinear it doesn't preserve frequency, but it is not hard to find that the analog frequency can be mapped to the digital frequency by

$$\omega_{\text{analog}} = \tan(\frac{1}{2}\omega_{\text{digital}}) \tag{7.32}$$

thus compressing the analog frequency axis from  $-\infty$  to  $\infty$  onto the digital frequency axis from  $-\pi$  to  $+\pi$  in a one-to-one manner. So the bilinear mapping method of IIR filter design goes something like this. First 'prewarp' the frequencies of interest (e.g.,  $f_p$ ,  $f_c$ ,  $f_s$ ) using equation (7.32). Then design an analog filter using a Butterworth, Chebyshev, inverse Chebyshev, or elliptical design. Finally, transform the analog transfer function into a digital one by using the bilinear mapping of equation (7.31) on all the poles and zeros.

FIR filters do not directly correspond to any of the classical designs, and hence we have no recourse but to return to first principles. We know that given the required frequency response of a filter we can derive its impulse response by taking the iLTDFT

$$h_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(e^{\mathbf{i}\omega}) e^{\mathbf{i}\omega n} \, d\omega \tag{7.33}$$

#### FILTERS

and that these  $h_n$  are the coefficients of the convolution in the time domain. Therefore, the theoretical frequency responses of the ideal low-pass, high-pass, band-pass, and band-stop filters already imply the coefficients of the ideal digital implementation. Assuming a noncausal filter with an odd number of coefficients, it is straightforward to find the following.

$$\begin{aligned} \text{low-pass:} \quad h_n &= \begin{cases} \frac{\omega_c}{\pi} & n=0\\ \frac{\omega_a}{\pi}\operatorname{sinc}(n\omega_c) & n\neq 0 \end{cases} \\ \text{high-pass:} \quad h_n &= \begin{cases} 1 - \frac{\omega_c}{\pi} & n=0\\ -\frac{\omega_c}{\pi}\operatorname{sinc}(n\omega_c) & n\neq 0 \end{cases} \end{aligned}$$
(7.34)
$$\text{band-pass:} \quad h_n &= \begin{cases} \frac{\omega_2 - \omega_1}{\pi} & n=0\\ \frac{\omega_2}{\pi}\operatorname{sinc}(n\omega_2) - \frac{\omega_1}{\pi}\operatorname{sinc}(n\omega_1) & n\neq 0 \end{cases} \\ \text{band-stop:} \quad h_n &= \begin{cases} 1 + \frac{\omega_1 - \omega_2}{\pi} & n=0\\ \frac{\omega_1}{\pi}\operatorname{sinc}(n\omega_1) - \frac{\omega_2}{\pi}\operatorname{sinc}(n\omega_2) & n\neq 0 \end{cases} \end{aligned}$$

Unfortunately these  $h_n$  do not vanish as |n| increases, so in order to implement a *finite* impulse response filter we have to truncate them after some |n|.

Truncating the FIR coefficients in the time domain means multiplying the time samples by a rectangular function and hence is equivalent to a convolution in the frequency domain by a sinc. Such a frequency domain convolution causes blurring of the original frequency specification as well as the addition of sidelobes. Recalling the Gibbs effect of Section 3.5 and the results of Section 4.2 regarding the transforms of signals with discontinuities, we can guess that multiplying the input signal by a smooth window

$$h_n' = w_n h_n \tag{7.35}$$

rather than by a sharply discontinuous rectangle should reduce (but not eliminate) the ill effects.

What type of window should be used? In Section 13.4 we will compare different window functions in the context of power spectrum estimation. Everything to be said there holds here as well, namely that the window function should smoothly increase from zero to unity and thence decrease smoothly back to zero. Making the window smooth reduces the sidelobes of the window's FT, but at the expense of widening its main lobe, and thus widening the transition band of the filter. From the computational complexity standpoint, we would like the window to be nonzero over only a short time duration; yet even nonrectangular windows distort the frequency response by convolving with the window's FT, and thus we would like this FT to be as narrow as possible. These two wishes must be traded off because the uncertainty theorem limits how confined the window can simultaneously be in the time and frequency domains. In order to facilitate this trade-off there are window families (e.g., kaiser and Dolph-Chebyshev) with continuously variable parameters.

So the windowing method of FIR filter design goes something like this.

```
Decide on the frequency response specification
Compute the infinite extent impulse response
Choose a window function:
trade off transition width against stop-band rejection
trade off complexity against distortion
Multiply the infinite extent impulse response by the window
```

The window design technique is useful when simple programming or quick results are required. However, FIR filters designed in this way are not optimal. In general it is possible to find other filters with higher stop-band rejection and/or lower pass-band ripple for the same number of coefficients. The reason for the suboptimality is not hard to find, as can be readily observed in Figure 7.14. The ripple, especially that of the stop-band, decreases as we move away from the transition. The stop-band attenuation specification that must be met constrains only the first sidelobe, and the stronger rejection provided by all the others is basically wasted. Were we able to find



Figure 7.14: FIR design by window method vs. by Remez algorithm. (A) is the frequency response of a 71-coefficient low-pass filter designed by the window method. (B) is a 41-coefficient filter designed by the Remez algorithm using the same specification. Note the equiripple characteristic.

314

an *equiripple* approximation we could either reduce the maximum error or alternatively reduce the required number of coefficients.

As in classical filter design the equiripple property ensures that the maximal deviation from our amplitude specification be minimal. Techniques for solving the minimax polynomial approximation problem are reviewed in Appendix A.12. In the early seventies McClellan, Parks, and Rabiner published a paper and computer program that used the Remez exchange algorithm for FIR design. This program has become the most widely used tool in FIR design, and it is suggested that the reader obtain a copy (or a full up-to-date program with user interface and graphics based on the original program) and become proficient in its use.

Before concluding this chapter we should answer the question that must have occurred to you. When should FIR filters be used and when IIR? As a general rule integrators are IIR, while differentiators are FIR. Hilbert transforms are usually FIR although IIR designs are sometimes used. As to frequency-selective filters, the answer to this question is often (but not always) easy. First recall from Section 7.2 that FIR filters can be linear-phase, while IIR filters can only approach this behavior. Hence, if phase response is critical, as in many communications systems (see Chapter 18), you may be forced to use FIR filters (although the trick of exercise 7.2.5 may be of use). If phase response is not of major importance, we can generally meet a specification using either FIR or IIR filters. From the computational complexity point of view, IIR filters almost always end up being significantly more efficient, with elliptical filters having the lowest computational requirements. The narrower the transitions the more pronounced this effect becomes. However, these elliptical filters also have the worst phase response, erratically varying in the vicinity of transitions.

#### EXERCISES

- 7.8.1 Some digital filter design programs assume a sampling frequency (e.g., 8000 Hz). Can these programs be used to design filters for systems with different sampling frequencies?
- 7.8.2 Obtain a good filter design program and design an IIR low-pass filter using the four classical types from Section 7.7. What happens as you force the transition region to shrink in size? What is the effect of  $f_p$  for a given transition region width? Plot the phase response and group delay. How can you make the phase response more linear?
- 7.8.3 Design a low-pass FIR using the same criteria as in the previous exercise. Compare the amount of computation required for similar gain characteristics.

- 7.8.4 Repeat the previous two questions for a narrow band-pass filter.
- 7.8.5 An extremely narrow FIR low-pass filter requires a large number of coefficients, and hence a large amount of computation. How can this be reduced?
- 7.8.6 The impulse responses in equations (7.34) were for odd N. For even N the ideal frequency responses must be shifted by a half-integer delay  $e^{\frac{i}{2}}$  before applying equation (7.33). Find the ideal impulse responses for even N.
- 7.8.7 What are the coefficients of the ideal differentiator and Hilbert filters for even and odd N?

# 7.9 Spatial Filtering

Up to now we have dealt with filters that are frequency selective—filters that pass or block, amplify or attenuate signals based on frequency. In some applications there are other signal characteristics that help differentiate between signals, and these can be used along with frequency domain filtering, or by even by themselves when we need to separate signals of the same frequency. One such characteristic is the geographical position of the signal's source; if we could distinguish between signals on that basis we could emphasize a specific signal while eliminating interference from others not colocated with it.

A wave is a signal that travels in space as well as varying in time, and consequently is a function of the three-dimensional spatial coördinates sas well as being a function of time t. At any particular spatial coördinate the wave is a signal, and at any particular time we see a three-dimensional spatially varying function. A wave that travels at a constant velocity vwithout distortion is a function of the combination s - vt; traveling at exactly the right speed you 'move with the wave'. The distance a periodic wave travels during a single period is called the wavelength  $\lambda$ . Light and radio waves travel at the speed of light (approximately  $3 \cdot 10^8$  meters per second), so that a wavelength of one meter corresponds to a frequency of 300 MHz.

Directional antennas, such as the TV antennas that clutter rooftops, are spatially selective devices for the reception and/or transmission of radio waves. Using carefully spaced conducting elements of precise lengths, transmitted radiation can be focused in the desired direction, and received signals arriving from a certain direction can be amplified with respect to those from other angles. The problem with such directional antennas is that changing the preferred direction involves physically rotating the antenna to point the desired way. *Beamforming* is a technique, mainly utilized in transmission and reception of sonar and radar signals, for focusing transmitted energy or amplifying received energy without having to physically rotate antennas. This feat is performed by combining a number of omnidirectional sensors (antennas, microphones, hydrophones, or loudspeakers depending on the type of wave).



Figure 7.15: Beamforming to separate two sinusoidal signals of the same frequency. The sensor array consists of two antennas separated by the distance traveled by the wave during half a period. Each sensor is connected to a phase shifter and the phase shifted signals are summed.

In the simplest example of the principle involved we need to discriminate between two sinusoidal waves of precisely the same frequency and amplitude but with two orthogonal directions of arrival (DOAs) as depicted in Figure 7.15. Wave  $x_1$  impinges upon the two sensors at the same time, and therefore induces identical signals  $y_1$  and  $y_2$ . Wave  $x_2$  arrives at the lower sensor before the upper, and accordingly  $y_1$  is delayed with respect to  $y_2$  by a half period. Were the reception of wave  $x_1$  to be preferred we would set both phase shifters to zero shift;  $y_1$  and  $y_2$  would sum when  $x_1$ is received, but would cancel out when  $x_2$  arrives. Were we to be interested in wave  $x_2$  we could set  $\Delta \Phi_2$  to delay  $y_2$  by one half period, while  $\Delta \Phi_1$ would remain zero; in this fashion  $x_1$  would cause  $y_1$  and  $y_2$  to cancel out, while  $x_2$  would cause them to constructively interact. For waves with DOA separations other than 90° the same idea applies, but different phase shifts need to be employed.



Figure 7.16: A wave impinging upon an array of M = 5 sensors spaced d apart. The parallel lines represent the peaks of the sinusoids and hence there is one wavelength  $\lambda$  between each pair. The wave arrives at angle  $\theta$  from the normal to the line of the sensors. It is obvious from the geometry that when  $\lambda = d \sin \theta$  the wave takes the same value at all of the sensors.

The device just described is a rudimentary example of a phased array, and it has the advantage of eliminating mechanical motors and control mechanisms. Switching between different directions can be accomplished essentially instantaneously, and we may also simultaneously recover signals with multiple DOAs with the same array, by utilizing several different phase shifters. We can enhance directivity and gain of a phased array by using more than two sensors in the array. With an array with M sensors, as in Figure 7.16, at every time n we receive M signals  $y_{mn}$  that can be considered a vector signal  $\underline{y}_n$ . To enhance a signal of frequency  $\omega$  impinging at angle  $\theta$ we need a phase delay of  $\kappa = 2\pi \frac{d}{\lambda} \sin \theta$  between each two consecutive sensors. We could do this by successive time delays (resulting in a *timed array*) but in a phased array we multiply the  $m^{\text{th}}$  component of the vector signal by a phase delay  $e^{-i\kappa m}$  before the components are combined together into the output  $z_n$ .

$$z_n = \sum_{m=0}^{M-1} y_{mn} e^{-i\kappa m}$$
(7.36)

Forgetting the time dependence for the moment, and considering this as a function of the DOA variable  $\kappa$ , this is seen to be a spatial DFT! The sensor number m takes the place of the time variable, and the DOA  $\kappa$  stands in for the frequency. We see here the beginnings of the strong formal resemblance between spatial filtering and frequency filtering.

Now what happens when a sinusoidal wave of frequency  $\omega$  and DOA  $\phi$ 



Figure 7.17: Angle response of a phased array. We depict the square of the response in dB referenced to the zero degree response for a phased array with M = 32 and  $\pi \frac{d}{\lambda} = \frac{3}{4}$ . The phased array is pointed to  $\theta = 0$  and the horizontal axis is the angle  $\phi$  in degrees.

is received? Sensor m sees at time n

$$y_{mn} = A e^{+i\varphi + i\omega n + im\frac{2\pi d}{\lambda}\sin\phi}$$
$$= A e^{i\varphi} e^{i\omega n} e^{ikm}$$

where  $\varphi$  is the phase at the first sensor, and k is the DOA variable corresponding to angle  $\phi$ . Substituting this into equation (7.36)

$$z_n = \sum_{m=0}^{M-1} A e^{i\varphi} e^{i\omega n} e^{ikm} e^{-i\kappa m}$$

$$= A e^{i\varphi} e^{i\omega n} \sum_{m=0}^{M-1} e^{i(k-\kappa)m}$$

$$= A e^{i\varphi} e^{i\omega n} \frac{1 - e^{iM(k-\kappa)}}{1 - e^{i(k-\kappa)}}$$

$$= A e^{i\varphi} e^{i\omega n} e^{i\frac{1}{2}M(k-\kappa)} e^{-i\frac{1}{2}(k-\kappa)} \frac{\sin\frac{1}{2}M(k-\kappa)}{\sin\frac{1}{2}(k-\kappa)}$$

where we have performed the sum using (A.48), symmetrized, and substituted (A.8). The phased array *angle response* is the square of this expression

$$|z|^{2} = \left(\frac{\sin\frac{1}{2}M(k-\kappa)}{\sin\frac{1}{2}(k-\kappa)}\right)^{2} = \left(\frac{\sin M\pi\frac{d}{\lambda}(\sin\phi-\sin\theta)}{\sin\pi\frac{d}{\lambda}(\sin\phi-\sin\theta)}\right)^{2}$$
(7.37)

and is plotted in Figure 7.17.

So the phased array acts as a spatial filter that is really quite similar to a regular frequency domain filter. The angle response of equation (7.37) is analogous to the frequency response of a frequency filter, and the high sidelobes in Figure 7.17 can be attenuated using techniques from filter design, such as windowing.

Our discussion has focused on simple sinusoidal waves; what if we need to pull in a complex wave? If the wave consists of only two frequency components, we can build two separate phased arrays based on the same sensors and add their results, or equivalently a single phased array with two delays per sensor. A little thought should be sufficient to convince you that arbitrary waves can be accommodated by replacing the simple phase delay with full FIR filters. In this way we can combine spatial and frequency filtering. Such a combined filter can select or reject a signal based on both its spectral and spatial characteristics.

#### EXERCISES

- 7.9.1 Direction fixing can also be performed using time of arrival (TOA) techniques, where the time a signal arrives at multiple sensors is compared. We use both phase differences and TOA to locate sound sources with our two ears, depending on the frequency (wavelength) of the sound. When is each used? How is elevation determined? (Hint: The external ear is not symmetric.) Can similar principles be exploited for SONAR echolocation systems?
- 7.9.2 Bats use biological sonar as their primary tool of perception, and are able to hunt insects at night (making the expression *blind as a bat* somewhat frivolous). At first, while searching for insects, they emit signals with basic frequency sweeping from 28 KHz down to 22 KHz and duration of about 10 milliseconds. Once a target is detected the sounds become shorter (about 3 milliseconds) in duration but scan from 50 KHz down to 25 KHz. While attempting to capture the prey, yet a third mode appears, of lower bandwidth and duration of below 1 millisecond. What is the purpose of these different cries? Can similar principles be used for fighter aircraft radar?

# **Bibliographical Notes**

Filters and filter design are covered in all standard DSP texts [186, 185, 200, 167], and chapters 4 and 5 of [241], as well as books devoted specifically to the subject [191]. Many original papers are reprinted in [209, 40, 41].

The original Parks-McClellan FIR design program is described and (FOR-TRAN) source code provided in [192, 165]. Extensions and portings of this code to various languages are widely available. After the original article appeared, much follow-on work appeared that treated the practical points of designing filters, including differentiators, Hilbert transforms, etc. [208, 213, 212, 78, 207].

Exercise 7.2.5 is based on [197].