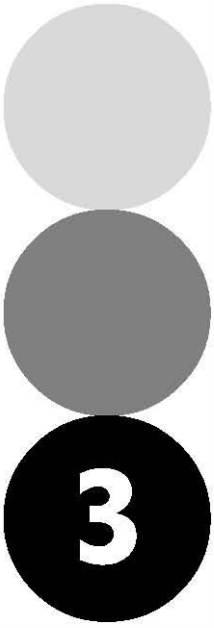

3 Digital Filters

3.0 Introduction

This chapter introduces digital filtering, stressing intuition along with the terminology and notation used to describe digital filters. First, filtering will be discussed in general, followed by definitions and examples of *Finite Impulse Response* (FIR) and *Infinite Impulse Response* (IIR) filters. Then, the general form of a digital filter will be given and discussed. The Z transform will be introduced as a simple algebraic substitution operator, and this will be exploited to define and develop the notion of the transfer function. Zeroes and poles will be defined, and some useful filter forms will be shown. Math-averse readers could possibly skip this chapter for now, returning to it when needed in later chapters, but I'd recommend reading at least until your eyes glaze over.

3.1 Linear Systems, LTI Systems, Convolution

Linearity is a property of systems that allows us to use many powerful mathematical and signal processing techniques to analyze and predict the behavior of these systems. Linearity has two defining criteria:

Homogeneity: if $x \rightarrow y$ then $\alpha x \rightarrow \alpha y$ for any α

Superposition: if $x1 \rightarrow y1$ and $x2 \rightarrow y2$ then $x1 + x2 \rightarrow y1 + y2$

($\rightarrow$ is read "yields", and corresponds to a system operating on x to yield y).

These equations state that a mixture and/or scaling of inputs simply results in a mixture and/or scaling of the outputs. No "new" signals are created by a linear system (we'll have more rigorous means to define "new" later).

A time-invariant system obeys the property:

If $x(n) \rightarrow y(n)$ then $x(n + N) \rightarrow y(n + N)$ for any N,

which simply means that the system doesn't change its behavior with time. Here, $x(n)$ is the chain of samples put into the system, and $y(n)$ is the corresponding chain of output samples. Practically speaking, most systems actually do change over time. A reasonable assumption, however, is that many systems of interest do not change their behavior quickly, and can be treated as time-invariant over time intervals of interest. The bones of the middle ear, acoustics in rooms, a good quality stereo amplifier, and many other systems behave like *Linear Time-Invariant* (LTI) systems over much of their normal operating range. If a system is linear and time-invariant, we can characterize its behavior by measuring its impulse response as shown in Figure 3.1.

The impulse response is defined mathematically as:

$$h(n) = y(n), \text{ for } x(n) = \delta(n)$$

$$\text{where } \delta(n) = \begin{cases} 1, & n = 0, \\ 0, & \text{otherwise.} \end{cases}$$

Linearity and time invariance mean that if we excite a system with an input of 1 at time zero, and 0 thereafter, we can "record" (observe) the output and use that to determine exactly what the system response would be to any arbitrary input. To prove to ourselves that this is true, all we need do is invoke

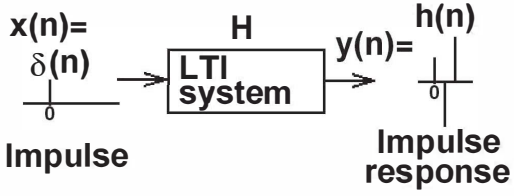

Figure 3.1. Impulse response of a Linear Time-Invariant (LTI) system.

the three properties of LTI systems: homogeneity, superposition, and time-invariance. Thus, any input can be decomposed into a time-ordered set of weighted impulses:

$$x(n) = x_0\delta(n) + x_1\delta(n-1) + x_2\delta(n-2) + x_3\delta(n-3) + \ldots + x_M\delta(n-M).$$

Each input sample can be viewed as a separate weighted (x_0, x_1, etc., are the weights) impulsive input, and the outputs can be viewed as individual outputs, which are weighted versions of the impulse response $h(n)$:

$$y(n) = x_0h(n) + x_1h(n-1) + x_2h(n-2) + x_3h(n-3) + \ldots + x_Mh(n-M)$$
$$= \Sigma_i\, x(i)h(n-i) \qquad \text{denoted by } x(n) * h(n).$$

Figure 3.2 shows the interaction of an input signal with a linear time-invariant system as a decomposition of separate impulse responses. While

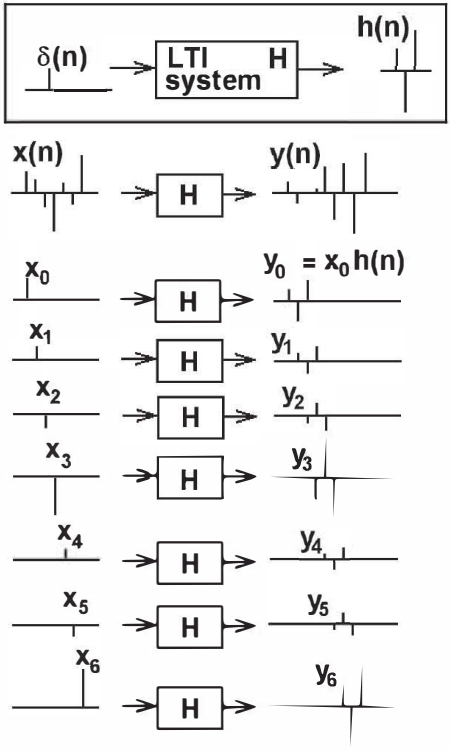

Figure 3.2. Convolution of input with impulse response of Linear Time-Invariant (LTI) system.

seeming quite tedious to calculate (which it is), this operation, called *convolution,* will be very important in that it allows us to use many mathematical tools to analyze and simulate LTI Systems.

3.2 Digital Filters

By forming linear combinations of past input and output samples, digital filters operate on streams of numbers that are uniformly sampled in time (such as samples of audio). Current and past inputs are usually denoted as

$$x(n),\ x(n-1),\ x(n-2),\ \ldots$$

where n is the current time, $n-1$ is the time one sampling period before the current one, $n-2$ is two sampling periods ago, etc. Current and past outputs are usually denoted as

$$y(n),\ y(n-1),\ y(n-2),\ \ldots$$

As discussed in Chapter 1, PCM signals are formed by sampling an analog waveform at regular intervals in time. The sampling intervals are spaced T seconds apart, where T = 1/sampling rate. Thus, relating the integer time indices n, $n+1$, etc., of a sampled signal x to actual times in seconds requires multiplying by the sampling period.

$$x(n) = x_{\text{continuous}}(nT)$$
$$x(n-1) = x_{\text{continuous}}(nT - T)$$
etc., where T = 1/(Sampling Rate)

3.3 FIR Filters

A simple two-point moving average filter can be written as:

$$y(n) = 0.5\ (x(n) + x(n-1)). \tag{3.1}$$

Such a filter is called a FIR (*Finite Impulse Response*), because it operates only on a finite number of delayed versions of its inputs. The number of delays used is referred to as the filter "order." FIR means the filter's impulse response yields only a finite number of nonzero output samples (two successive values of one half in this case). Even though it expresses a sum, Equation 3.1 is called a *difference equation.* Figure 3.3 shows a block signal processing

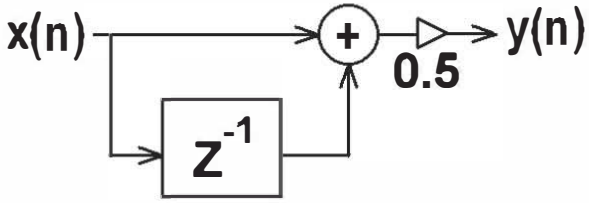

Figure 3.3. Two-point moving average digital filter.

diagram of the two-point moving average filter. The Z^{-1} block in the feedforward block represents a unit sample of delay. We'll find out more about Z and Z^{-1} later on.

Filters of the form of Equation 3.1 are also called nonrecursive, moving average, or all zero (more on that later). Figure 3.4 shows a signal processing block diagram of a general FIR filter. From the discussion of convolution in Section 3.1, note now that an arbitrary (finite length) impulse response can be stored in the coefficients of an FIR filter, and the operation of the filter would then actually perform the convolution. Thus, any LTI system with finite-length

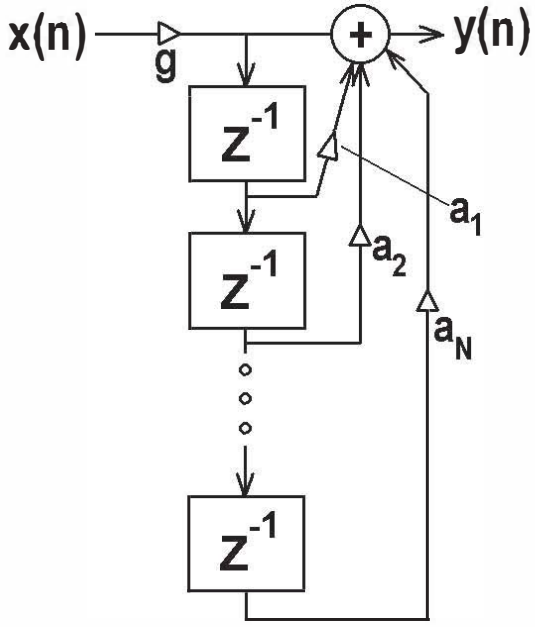

Figure 3.4. A high (Nth) order general FIR digital filter.

impulse response can be modeled by an FIR filter, provided that the impulse response of the LTI system is bandlimited to the Nyquist frequency.

3.4 IIR Filters

A simple filter that operates on past outputs can be written as

$$y(n) = (g\,x(n)) + (r\,y(n-1)) \tag{3.2}$$

It's easy to show that the impulse response of this filter for $g = 1$ is $r^n = 1.0$, r, r^*r, r^3, etc. This type of response is called an *exponential decay*. It's easy to see why filters of this form are called *Infinite Impulse Response* (IIR) filters, because for a nonzero r, the output technically never goes exactly to zero. If r is negative, the filter will oscillate positive and negative each sample, corresponding to even and odd powers of r. This is called an *exponential oscillation*. If the magnitude of r is greater than one, the filter output will grow without bound. This condition is called *instability*, and such filters are called *unstable*.

Filters of the form of Equation 3.2 are also called recursive, all pole (more on that later), and autoregressive. Figure 3.5 shows a signal processing block diagram of the simple recursive filter described in Equation 3.2. Figure 3.6 shows a higher order IIR filter block diagram.

3.5 The General Filter Form

The most general digital filter operates on both its inputs and outputs, and its difference equation is written:

$$\begin{aligned} y(n) = g(x(n) + a_1x(n-1) + a_2x(n-2) + \ldots + a_Nx(n-N)) \\ -b_1y(n-1) - b_2y(n-2) - \ldots - b_My(n-M). \end{aligned} \tag{3.3}$$

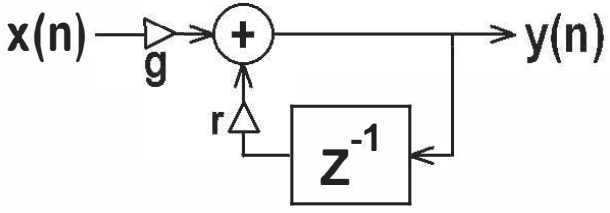

Figure 3.5. First order recursive digital filter.

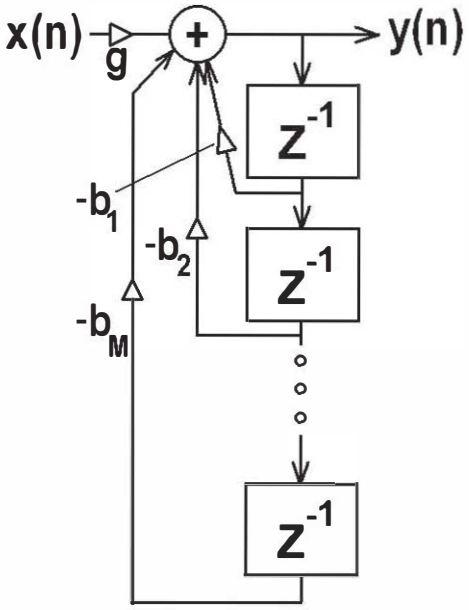

Figure 3.6. High order recursive digital filter.

Note that the length of input sample "history" is not required to be equal to the length of output sample "history," though in practice they are commonly assumed to be equal. The "order" of a filter is equal to the longest delay used in the filter; in the filter of Equation 3.3, the order would be the greater of N or M. Since general digital filters have IIR components as shown in Equation 3.3, such filters are also called IIR filters. Another term for filters with both FIR and IIR parts is *pole-zero filter* (more later). One final commonly used term is *Auto-Regressive Moving Average*, or ARMA. Figure 3.7 shows a signal processing block diagram of the general pole-zero digital filter described in Equation 3.3.

3.6 The Z Transform

A common analytical tool for digital filters is the Z transform representation. As we said before, we'll define Z^{-1} (Z to the minus 1) as a single sample of delay, and in fact, Z^{-1} is sometimes called the *Delay Operator*. To transform a filter using the Z transform, simply capitalize all variables x and y, and replace all time indices $(n-a)$ with the appropriate time delay operator Z^{-a}. Thus, the Z transformed version of Equation 3.3 would be written:

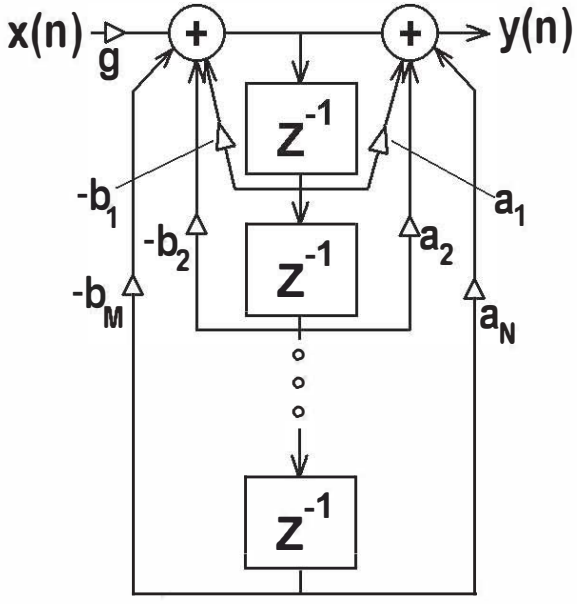

Figure 3.7. General pole-zero (IIR) filter.

$$Y = g(X + a_1XZ^{1} + a_2XZ^{2} + \ldots + a_NXZ^{N}) \qquad (3.4)$$
$$-b_1YZ^{1} - b_2YZ^{2} - \ldots - b_MYZ^{M}.$$

We'll see in subsequent sections and chapters how the Z transform can be used for analyzing and manipulating signals and filters.

3.7 The Transfer Function

A powerful relationship used for analyzing digital filters is the *transfer function*, which is found by solving for the ratio of output (Y) to input (X) in the Z-transformed filter expression. The transfer function for Equation 3.4 can be solved by using simple algebra:

$$Y(1 + b_1Z^{-1} + b_2Z^{-2} + \ldots + b_MZ^{-M})$$
$$= gX(1 + a_1Z^{-1} + a_2Z^{-2} + \ldots + a_NZ^{-N}) \qquad (3.5)$$

$$H = \frac{Y}{X} = \frac{g(1 + a_1Z^{-1} + a_2Z^{-2} + \cdots + a_NZ^{-N})}{1 + b_1Z^{-1} + b_2Z^{-2} + \ldots + b_MZ^{-M}}. \qquad (3.6)$$

The transfer function is notated as H. H is the Z transform of the time-domain impulse response function $h(n)$. Transformation of x and y into the Z domain gives us a tool for talking about a function H (the Z transform of h) that takes X as input and yields Y.

3.8 Zeroes and Poles

Looking at the numerator of Equation 3.6 as a polynomial in Z^{-1}, there will be N values of Z^{-1} that make the numerator equal to zero, and the transfer function will be zero at these values. These values that make the polynomial equal to zero are potentially complex numbers: $Re + jIm$, where Re and Im are called the *real* and *imaginary* parts, and $j = \sqrt{(-1)}$. The zero values are called *zeroes of the filter* because they cause the gain of the transfer function to be zero. The two-dimensional (real and imaginary) space of possible values of Z is called the *z-plane*.

Similarly, the denominator will have M values of Z^{-1} that make it zero, and these values cause the filter gain to be infinite at those values. These M values are called *poles of the filter* (like tent poles sticking up in the transfer function, with infinite height where the denominator is zero). Poles are important because they can model resonances in physical systems (we'll see that in the next chapter). Zeroes model signal cancellations, as in the destructive interference discussed in Chapter 2.

3.9 First Order One-Zero and One-Pole Filters

The simple two-point moving average filter was defined in Equation 3.1, and shown in Figure 3.3. A more general form of the first order one-zero filter is shown in Figure 3.8, and is described by the following equations:

$$y(n) = g(x(n) + ax(n - 1)) \tag{3.7}$$

$$Y/X = g(1 + aZ^{-1}). \tag{3.8}$$

This filter has a single zero at $Z = -a$, and exhibits a maximum gain of $g \cdot (1 + |a|)$. Figure 3.9 shows the gain responses versus frequency of this filter for various values of a (with g set to $1/(1 + |a|)$ to normalize the maximum gain). Such plots are called *spectral magnitude plots*, because they show the magnitude of the gain of the filter at each frequency, from zero Hz up to one half of the sampling rate (the maximum unaliased frequency). We will see a lot more on spectra and spectral plots in Chapters 5 and 6.

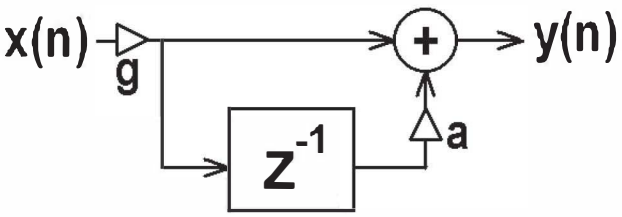

Figure 3.8. General one-zero filter.

As can be seen from Figure 3.9, positive values of a cause the filter to favor low frequencies over high. Such filters are called low pass filters. Negative values of a cause the filter to be a high pass filter, as shown in Figure 3.8. If a is set to –1 and g to T (the sampling period), the one-zero filter becomes the digital approximation to differentiation (delta in adjacent x values divided by delta time). We'll use this in later chapters.

The simple first order one-pole filter was shown in Figure 3.5 and is described by the equations below:

$$y(n) = gx(n) + ry(n-1) \tag{3.9}$$

$$Y/X = g / (1 - rZ^{-1}). \tag{3.10}$$

This filter has a single pole at $Z = r$, and exhibits a maximum gain of $g/(1 - |r|)$. Figure 3.10 shows the gain responses versus frequency of this filter for various values of r (g is set to $1 - |r|$ to normalize the filter maximum gain). As we observed before, the absolute value of r must be less than one in order for the filter to be stable. As can be seen from Figure 3.10, positive values of r cause the filter to favor low frequencies (low pass), while negative values cause the filter to favor high frequencies (high pass). If both g and r

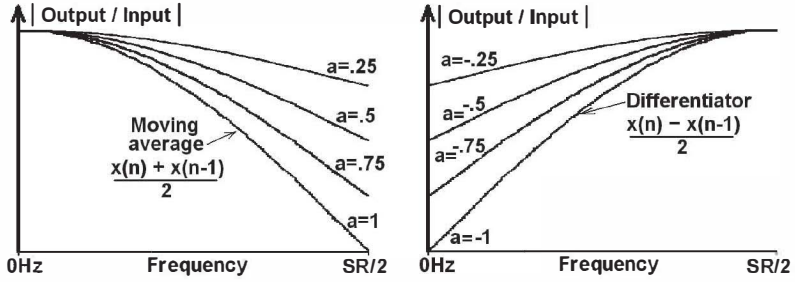

Figure 3.9. Gain versus zero location for one-zero filter.

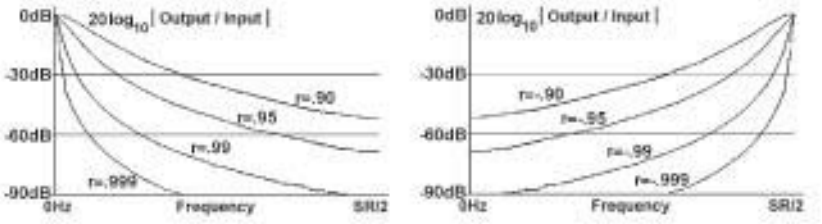

Figure 3.10. Gain versus pole location for one-pole filter.

((11))

are set to 1.0 (which is unstable), the first order one-pole filter implements the *digital integrator*, which simply sums up all prior input signals.

The plots in Figure 3.10 show amplitude mapped into a logarithmic space. This is often done in audio and for some other signal representations because the dynamic range of the signals found is often quite large. For example, our hearable range of audio signals is immense: The amplitude ratio of the quietest sound we can hear to the loudest sound we can tolerate without pain or damage is one to one million (a range of 140 dB, see Chapter 1)! Our one-pole filter, set with $r = 0.99$ can range in amplitude response from 1.0 at zero frequency, down to about 0.0035 at SRATE/2. Plotting that on a simple amplitude scale wouldn't be meaningful, so we plot it on the dB scale. Each factor of ten in amplitude is equal to 20 dB change, so our gain of 0.005 would correspond to −46 dB.

3.10 The Second Order Pole/Zero (BiQuad) Filter

A special form of digital filter is the *BiQuad*, so named because the numerator and denominator of the transfer function are both second order, or quadratic polynomials in Z^{-1}. Any polynomial can be factored into first and second order polynomials with real coefficients; thus, all that's needed to factor the transfer function of Equation 3.5 or 3.6 is first and second order building blocks. The second order two-pole blocks correspond to resonators, or oscillators with exponential damping, and are very powerful building blocks for digital filter systems. The second order two-zero building blocks are anti-resonators, capable of placing a pair of complex zeroes anywhere in the z-plane. A two-zero block combined with a two-pole block makes up a BiQuad.

The BiQuad in the time and Z transform domains looks like:

$$y(n) = g(x(n) + a_1 x(n-1) + a_2 x(n-2)) \\ -b_1 y(n-1) - b_2 y(n-2) \tag{3.11}$$

$$\frac{Y}{X} = \frac{g(1 + a_1 Z^{-1} + a_2 Z^{-2})}{1 + b_1 Z^{-1} + b_2 Z^{-2}}. \tag{3.12}$$

This filter has two poles and two zeroes. Depending on the values of the *a* and *b* coefficients, the poles and zeroes can be placed in fairly arbitrary positions around the *z*-plane, but not completely arbitrary if the *a* and *b* coefficients are real numbers (not complex). Remember from quadratic equations in algebra that the roots of a second order polynomial can be found by the formula: $(-a_1 +/- (a_1^2 - 4a_2)^{1/2}] / 2$ for the zeroes, and similarly for the poles using the *b* coefficients. It turns out that for complex roots, the positions will always end up as a *complex conjugate pair* of the form *Re +/– jIm*. Filters with two poles are called *resonators*, or *phasors*.

For practical use in sound processing, there is a wonderful formulation of the BiQuad that deals more directly with resonator parameters:

$$\frac{Y}{X} = \frac{g(1 - 2r_z \cos(2\pi Freq_z T) Z^{-1} + r_z^2 Z^{-2})}{1 - 2r_p \cos(2\pi Freq_p T) Z^{-1} + r_p^2 Z^{-2}} \tag{3.13}$$

$$\begin{aligned} y(n) = g(x(n) - 2r_z \cos(2\pi Freq_z T) x(n-1) + r_z^2 x(n-2)) \\ +2r_p \cos(2\pi Freq_p T) y(n-1) - r_p^2 y(n-2). \end{aligned} \tag{3.14}$$

This describes the filter coefficients in terms of an exponential damping parameter (r_z for the zeroes, r_p for the poles) and a center frequency of resonance (antiresonance for the zeroes), which is $Freq_z$ for the zeroes and $Freq_p$ for the poles. We can now control aspects of the filter more directly from these parameters, knowing that once we decide on r_z and $Freq_z$, we can convert to a_1, a_2, b_1, and b_2 directly. Figure 3.11 shows the BiQuad in block diagram form.

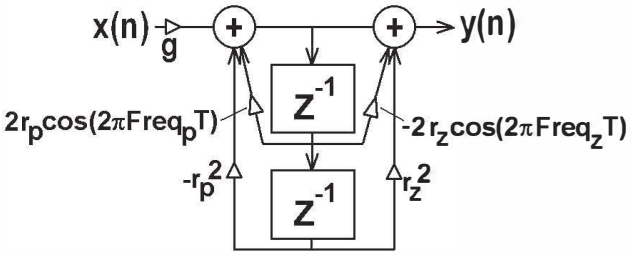

Figure 3.11. BiQuad filter block diagram.

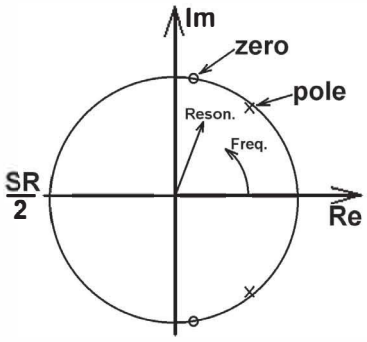

Figure 3.12. *Z*-plane pole/zero plot of BiQuad with zeroes at 1.0, 5000 Hz, and poles at 0.99, 3000 Hz (sampling rate is 22050).

Just as with the filter of Equation 3.2, r_p must be strictly less than one, and for the resonance formulation it is usually kept nonnegative, because we can use the frequency variable to swing the position around in an arc anywhere in the z-plane. $Freq_z$ and $Freq_p$ can take on any value from zero to one half the sampling rate (the Nyquist frequency). $Freq_z$ and $Freq_p$ can actually take on any value, but they will alias to frequencies within the Nyquist range, because of the "modulo 2π" nature of the cosine functions in Equation 3.13. Figure 3.12 shows the z-plane pole/zero plot of a Biquad filter with r_z set to 1.0; $Freq_z$ set to 5000 Hz; r_p set to 0.99; and $Freq_p$ set to 3000 Hz (sampling rate is 22050 Hz). The resonance parameters, r, are reflected by the radial distance from the origin in the z-plane. The frequency parameters determine the angle of the pole/zero locations.

Figure 3.13 shows the spectrum (top) of a random noise input to the BiQuad Filter, along with the spectrum of the filtered noise signal. *White noise* is named similarly to white light, where all frequencies are present in relatively equal amplitude, as shown in the magnitude versus frequency plot. Passing white noise through a filter results in what is called *colored noise*, in direct analogy to passing light through a color filter. Note the peak in the filtered output spectrum at 3000 Hz corresponding to the pole resonances, and the dip at 5000 Hz corresponding to the zero antiresonances.

Figure 3.14 shows some superimposed frequency responses of a Biquad with g set to 0.1; r_p set to 0.99 and 0.97; the zeroes set at special locations (+/–1 on the real axis); and $Freq_p$ swept from 0 to 4000 Hz in 500 Hz steps (sampling rate is 8000 Hz). Locating the zeroes at those locations of frequency = 0 and frequency = SRATE/2 helps to keep the total filter gain nearly constant, independent of the frequency of the resonator.

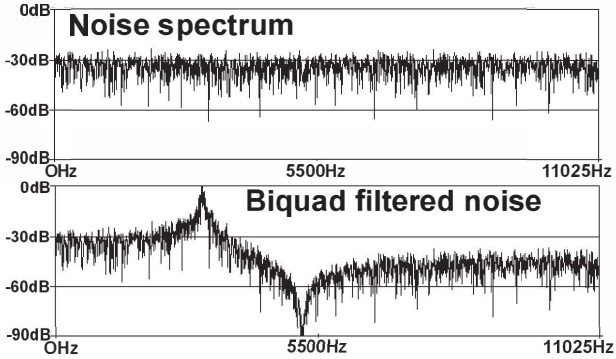

Figure 3.13. White noise input spectrum (top) versus BiQuad filtered output spectrum (bottom). The BiQuad has a resonator pole pair at 3000 Hz with $r = 0.99$, and a zero pair at 5000 Hz with $r = 1.0$ (same filter as shown in the z-plane view of Figure 3.12).

3.11 A Little on Filter Topologies

If you've had enough of digital filters, difference equations, the z-plane, and all that for now, you can skip the next sections and go on to Chapter 4. However, if you just can't get enough of this stuff, read on for some notes about implementing digital filters.

Digital filters are described by simple linear algebraic equations. As such they can be factored in a number of ways, and thus a given filter might be implemented in a wide variety of ways. As an example, we will use the filter shown in Figure 3.15 (hereafter we'll let $g = 1$ for convenience).

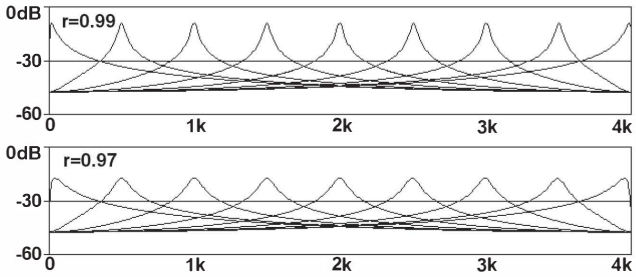

Figure 3.14. BiQuad transfer function magnitudes with zeroes set at +/–1.0 (one zero at DC and one at $SRATE/2$); $r = 0.99$ (top plot), and $r = 0.97$ (lower plot). Resonance frequencies are set to 0, 500, 1000, 1500, 2000, 2500, 3000, 3500, and 4000 Hz (sample rate = 8000).

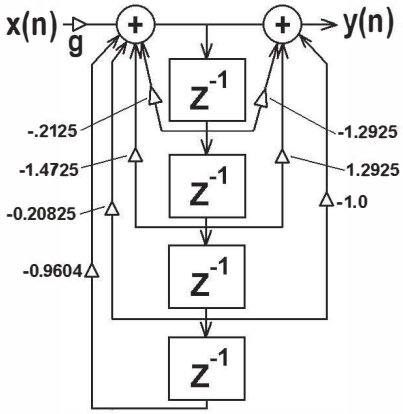

Figure 3.15. A fourth order IIR, third order FIR digital filter.

The filter of Figure 3.15 has a difference equation (for $g = 1$) of

$$y(n) = x(n) - 1.2925\,x(n-1) + 1.2925\,x(n-2) - x(n-3) - 0.2125\,y(n-1) - 1.4725\,y(n-2) - 0.20825\,y(n-3) - 0.9604\,y(n-4).$$

We can write out the Z transform of this filter as:

$$\frac{Y}{X} = \frac{1 - 1.2925Z^{-1} + 1.2925Z^{-2} - Z^{-3}}{1 + 0.2125Z^{-1} + 1.4725Z^{-2} + 0.20825Z^{-3} + 0.9604Z^{-4}}.$$

Noting that the numerator is of third order, and the denominator is of fourth order, we can factor each into first and second order filter segments:

$$\frac{Y}{X} = \frac{(1 - 0.2925Z^{-1} + Z^{-2}) \cdot (1 - Z^{-1})}{(1 - 0.6Z^{-1} + 0.98Z^{-2}) \cdot (1 + 0.8125Z^{-1} + 0.98Z^{-2})}.$$

Now that we've factored it, we can compare the sections to our standard BiQuad forms given in Equation 3.11 and see that this filter gives us two resonances: one at $r = 0.99, f = 3/8$ SRATE; and one at $r = 0.99, f = 7/8$ SRATE. It also gives us one complex zero pair at $r = 1.0, f = 5/7$ SRATE. Finally, it implements a single zero at $f = 0$. Since the transfer function is just a product of these, we can rewrite it as a chain of simple filter segments:

$$Y/X = (1 - 0.2925\,Z^{-1} + Z^{-2}) \cdot (1 - Z^{-1}) \cdot 1/(1 - 0.6Z^{-1} + 0.98Z^{-2}) \cdot 1/(1 + 0.8125Z^{-1} + 0.98Z^{-2}).$$

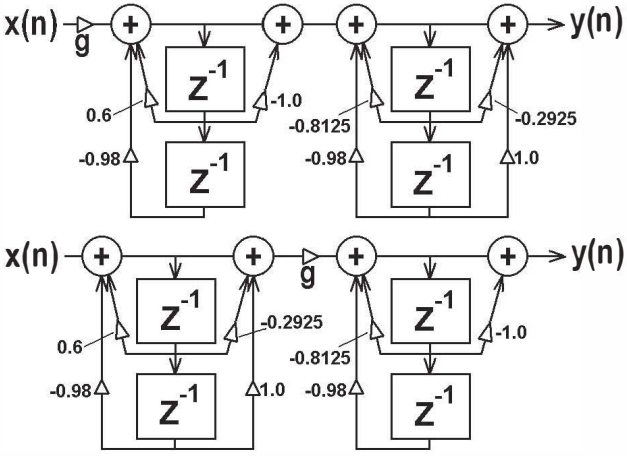

Figure 3.16. Two topological variants of the filter shown in Figure 3.15. Linearity says we can reorder the sections any way we like.

The mathematics of LTI systems tells us that these can be implemented in any order. Two of many such possibilities are shown in Figure 3.16. Note that the gain term can also go at any point in the chain.

13

The form shown in Figure 3.15 is called the *direct form*, and the forms shown in Figure 3.16 are all called *factored cascade forms*. There are many other forms for filters, including parallel. You might ask, "If the math says they're all the same, then why would we care about different forms?" Well, it turns out that the math is only strictly true in the pure case of infinite precision computation. For finite word sizes, even floats in computers, then rounding, truncation, quantization, etc., can all make a difference in filter implementations. Some topologies do better with finite precision computation. However, for the rest of this book we won't much care about this, because it's really not that critical of an issue for floating point computations on low order filters.

3.12 Conclusion

Digital filters operate on sampled signals by forming linear combinations of past inputs and outputs. LTI (linear time-invariant) systems are common in the acoustical world, and any LTI system can be modeled by a digital filter. Simple low order filters can be used to implement simple high and low pass functions, as well as implementing the averaging, differentiation, and integration operations. The second order pole-zero filter (called the "BiQuad")

is a convenient and flexible form, allowing independent control of resonance and antiresonance. The next chapter will look at the first actual physical model in this book and relate the simulation of that physical model (and more complex systems) to digital filters.

Reading:

Ken Stieglitz. *A Digital Signal Processing Primer*. Menlo Park: Addison Wesley, 1996.

Lawrence Rabiner and Bernard Gold. *Theory and Application of Digital Signal Processing*. Englewood Cliffs: Prentice Hall, 1974.

Code:

filter.c

GUIResoLab

Sounds:

[Track 11] One-Pole Filtered Speech, r = 0.9 ,0.95, 0.99, –0.9, –0.95, –0.99.

[Track 12] BiQuad Filtered Noise.

[Track 13] Noise Filtered Through Topologies of Figures 3.15–3.16.