Chapter 5
Design of IIR Filters

5.1 Introduction

IIR filter design primarily concentrates on the magnitude response of the filter and regards the phase response as secondary. The most common design method for digital IIR filters is based on designing an analogue IIR filter and then converting it to an equivalent digital filter.

There are many classes of analogue low-pass filter, such as the Butterworth, Chebyshev and Elliptic filters. The classes differ in their nature of their magnitude and phase responses. The design of analogue filters other than low-pass is based on *frequency transformations*, which produce an equivalent high-pass, band-pass, or band-stop filter from a prototype low-pass filter of the same class. The analogue IIR filter is then converted into a similar digital filter using a relevant transformation method. There are three main methods of transformation, the *impulse invariant* method, the *backward difference* method, and the *bilinear z-transform*.

5.2 IIR Filter Basics

A *recursive filter* involves *feedback*. In other words, the output values are calculated using one or more of the previous outputs, as well as inputs. In most cases a recursive filter has an impulse response which theoretically continues forever. It is therefore referred to as an infinite impulse response (IIR) filter. Assuming the filter is causal, so that the impulse response $h[n] = 0$ for $n < 0$, it follows that $h[n]$ cannot be symmetrical in form. Therefore, an IIR filter cannot display pure linear-phase characteristics like its adversary, the FIR filter.

The finite difference equation and transfer function of an IIR filter is described by Equation 3.3 and Equation 3.4 respectively. In general, the design of an IIR filter usually involves one or more strategically placed poles and zeros in the z-plane, to approximate a desired frequency response. An analogue filter can always be described by a frequency-domain transfer function of the general form, shown in Equation 5.1.

$$H(s) = K\frac{(s-z_1)(s-z_2)(s-z_3)\cdots}{(s-p_1)(s-p_2)(s-p_3)\cdots} \tag{5.1}$$

Where s is the Laplace variable and K is a constant, or gain factor. The filter is characterised by its poles $p_1, p_2, p_3\ldots$, and its zeros $z_1, z_2, z_3\ldots$, which can be plotted in the complex s-plane. The frequency response of the filter $H(\omega)$, can be obtained by replacing $s = j\omega$ into Equation 5.1. The complete response of the filter is then generated by varying ω in Equation 5.2 between 0 and ∞.

$$H(\omega) = K\frac{(j\omega-z_1)(j\omega-z_2)(j\omega-z_3)\cdots}{(j\omega-p_1)(j\omega-p_2)(j\omega-p_3)\cdots} \tag{5.2}$$

5.3 Analogue Low-pass Filters

There are several classes of analogue low-pass filter, three of which are the Butterworth, Chebyshev and Elliptic. These filters differ in the position of their and in the nature of their magnitude and phase responses. Their frequency responses are illustrated in Figure 5.1 below. The Butterworth filter is said to be monotonic at all frequencies (i.e. no local maxima or minima), the Chebyshev is monotonic in the stop-band and equiripple in the pass-band, and an Elliptic filter is equiripple in all bands.

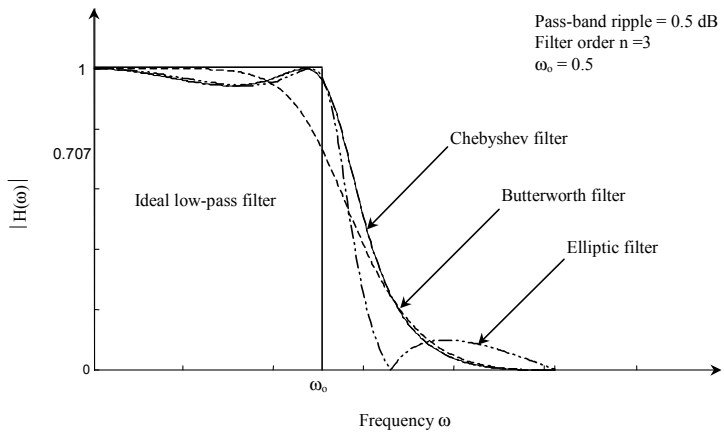

Figure 5.1: Typical frequency responses of various analogue low-pass filters.

5.4 The Bilinear z-transform

One of the most effective and widely used techniques for converting an *analogue* filter into a *digital* equivalent is by means of the bilinear *z*-transform. In this method, we replace *s* in equation (5.1) by the bilinear *z*-transform:

$$F(z) = \frac{z-1}{z+1} \tag{5.3}$$

to give the following function of *z*:

$$H(z) = K\frac{\left[\left(\frac{z-1}{z+1}\right)-z_1\right]\left[\left(\frac{z-1}{z+1}\right)-z_2\right]\left[\left(\frac{z-1}{z+1}\right)-z_3\right]\cdots}{\left[\left(\frac{z-1}{z+1}\right)-p_1\right]\left[\left(\frac{z-1}{z+1}\right)-p_2\right]\left[\left(\frac{z-1}{z+1}\right)-p_3\right]\cdots} \tag{5.4}$$

The frequency response of this *z*-transfer function is obtained by substituting $z = e^{j\Omega}$ in Equation (5.4). The result of doing this is most easily seen by making this substitution first in the function $F(z)$ in equation (5.3):

$$F(\Omega) = \frac{e^{j\Omega}-1}{e^{j\Omega}+1} = \frac{e^{j\Omega/2}-e^{-j\Omega/2}}{e^{j\Omega/2}+e^{-j\Omega/2}} = \frac{j2\sin(\Omega/2)}{2\cos(\Omega/2)} = j\tan\left(\frac{\Omega}{2}\right) \tag{5.7}$$

Substituting this into equation (5.4) we obtain:

$$H(\Omega) = K\frac{[j\tan(\Omega/2)-z_1][j\tan(\Omega/2)-z_2][j\tan(\Omega/2)-z_3]\cdots}{[j\tan(\Omega/2)-p_1][j\tan(\Omega/2)-p_2][j\tan(\Omega/2)-p_3]\cdots} \tag{5.8}$$

The frequency response of a desirable analogue filter was given by Equation (5.2). The function $H(\Omega)$ in equation (5.8) takes all values of the frequency response of the analogue filter, but compressed into the range $0 \le \Omega \le \pi$. Note that the compression of the frequency scale is non-linear. The shape of the *tan* function, as depicted in Figure 5.2, means that the "warping" effect is small near $\Omega = 0$, but increases greatly towards $\Omega = \pi/2$.

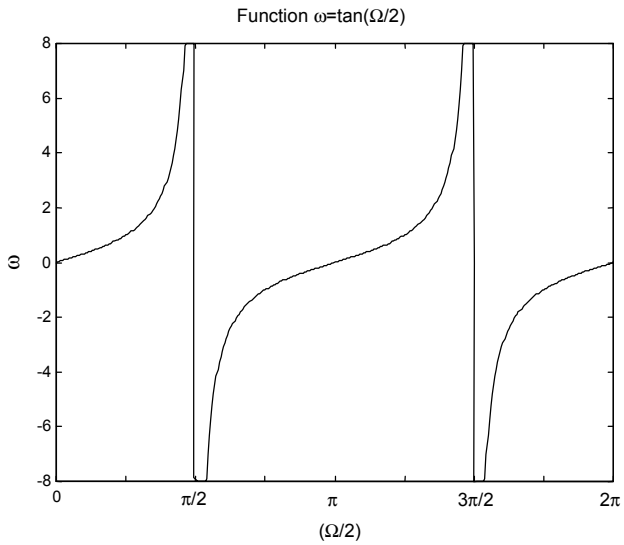

Figure 5.2: The "warping" effect of the tan function.

There are several advantages in using the bilinear z- transform. Firstly, the equiripple amplitude properties of the filters are preserved when the frequency axis is compressed. Secondly, there is no aliasing of the original analogue frequency response. As a result, the response of a low-pass filter falls to zero at $\Omega = \pi$. This is an extremely important feature in many practical applications. The principle of the bilinear z-transform, by making the substitution of Equation 5.6, is illustrated in Figure 5.3 below. It shows that the imaginary axis in the *s*-plane ($s = j\omega$) maps into the unit circle of the *z*-plane.

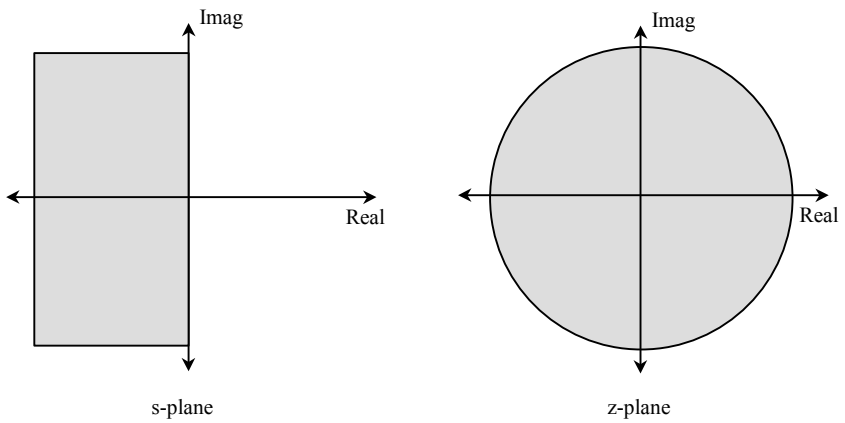

Figure 5.3: Illustration of s-plane to z-plane mapping using the bilinear z-transform.

The substitution maps the left-hand side of the *s*-plane to the inside of the unit circle in the *z*-plane. This ensures that the Nyquist stability criterion is obeyed and therefore filter stability is preserved. To overcome the frequency "warping" introduced by the bilinear z-transform, it is common practice to *pre-warp* the specification of the analogue filter, so that after warping they will be located at the desired frequencies. For example, suppose we wish to design a digital low-pass filter with a cut-off frequency Ω_c. We first transform this frequency to the analogue-domain cut-off frequency ω_{ac}, using the pre-warping relationship of Equation (5.9).

$$\omega_{ac} = k \cdot \tan\left(\frac{\Omega_c}{2}\right) \qquad k = 1 \quad or \quad \frac{2}{T} \tag{5.9}$$

We then proceed to design the analogue filter using the corresponding cut-off frequency, obtained from Equation (5.9). After the analogue filter has been transformed using the bilinear z-transform, the resulting digital filter will have its cut-off frequency in the correct place. Since pre-warping is performed in the beginning of the design procedure, and bilinear transformation is performed at the end, the value of *k* is immaterial.

5.5 Frequency Transformations

The design of analogue filters other than low-pass is usually achieved by designing a low-pass filter of the desired class e.g. Butterworth, Chebyshev, or Elliptic, and then transforming the resulting filter to get the desired frequency response e.g. high-pass, band-pass, or band-stop. This is accomplished by substituting the frequency-domain transfer function $H(s)$ with one of the relevant *frequency transformations* listed below. Where ω_2 and ω_1 are the band-edge frequencies of the desired filter and are also positive parameters satisfying $\omega_2 > \omega_1$.

Low-pass to *Low-pass* transformation: $$s \Rightarrow \frac{s}{\omega_{ac}} \tag{5.10}$$

Low-pass to *High-pass* transformation $$s \Rightarrow \frac{\omega_{ac}}{s} \tag{5.11}$$

Low-pass to *Band-pass* transformation $$s \Rightarrow \frac{s^2 + \omega_1\omega_2}{(\omega_2 - \omega_1)s} \tag{5.12}$$

Low-pass to *Band-stop* transformation $$s \Rightarrow \frac{(\omega_2 - \omega_1)s}{s^2 + \omega_1\omega_2} \tag{5.13}$$

5.6 Summary of IIR Filter Design Using the Bilinear z-transform

- Use the digital filter specification to determine a suitable normalised frequency-domain transfer function $H(s)$.
- Determine the cut-off frequency of the digital filter Ω_c.
- Obtain the equivalent analogue filter cut-off frequency ω_{ac} using the pre-warping function of Equation 5.9.
- Denormalise the analogue filter by frequency scaling $H(s)$, with one of the appropriate frequency transformations e.g. $s \Rightarrow s/\omega_{ac}$ etc.
- Apply the bilinear z-transform to obtain the digital filter transfer function $H(z)$ by replacing s with $(z - 1)/(z + 1)$.

5.6.1 Example

Design a digital filter equivalent of a 2nd order Butterworth low-pass filter with a cut-off frequency f_c = 100 Hz and a sampling frequency f_s = 1000 samples/sec. Derive the finite difference equation and draw the realisation structure of the filter. Given that the analogue prototype of the frequency-domain transfer function $H(s)$ for a Butterworth filter is:

$$H(s) = \frac{1}{s^2 + \sqrt{2} \cdot s + 1}$$

The normalised cut-off frequency of the digital filter is given by the following equation:

$$\Omega_c = \frac{2\pi f_c}{f_s} = \frac{2\pi 100}{1000} = 0.628$$

Now determine the equivalent analogue filter cut-off frequency ω_{ac}, using the pre-warping function of Equation 5.9. The value of K is immaterial so let $K = 1$.

$$\omega_{ac} = K \cdot \tan\left(\frac{\Omega_c}{2}\right) = 1 \cdot \tan\left(\frac{0.628}{2}\right)$$

$$\omega_{ac} = 0.325 \quad rads/\sec$$

Now denormalise the frequency-domain transfer function $H(s)$ of the Butterworth filter, with the corresponding low-pass to low-pass frequency transformation of Equation 5.10. Hence the transfer function of the Butterworth filter becomes:

$$H(s) = \frac{1}{\left[\frac{s}{0.325}\right]^2 + \sqrt{2}\cdot\left[\frac{s}{0.325}\right] + 1}$$

Next, convert the analogue filter into an equivalent digital filter by applying the bilinear z-transform. This is achieved by making a substitution for s in the transfer function.

$$s = \frac{z-1}{z+1} \equiv \frac{1-z^{-1}}{1+z^{-1}}$$

$$H(z) = \frac{1}{\frac{1}{0.325^2}\cdot\left[\frac{1-z^{-1}}{1+z^{-1}}\right]^2 + \frac{\sqrt{2}}{0.325}\cdot\left[\frac{1-z^{-1}}{1+z^{-1}}\right] + 1}$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{0.067 + 0.135z^{-1} + 0.067z^{-2}}{1 - 1.1429z^{-1} + 0.4127z^{-2}}$$

The finite difference equation of the filter is found by inverting the transfer function.

$$y(n) = 1.1429y(n-1) - 0.4127y(n-2) + 0.067x(n) + 0.135x(n-1) + 0.067x(n-2)$$

The transfer equation H(z) above, resembles the direct structure of Equation 3.13, from Chapter 3. So the realisation of this filter follows the same format as Figure 3.9, where the corresponding coefficients a_1, a_2, b_0, b_1, and b_2 are taken from the Equation above.

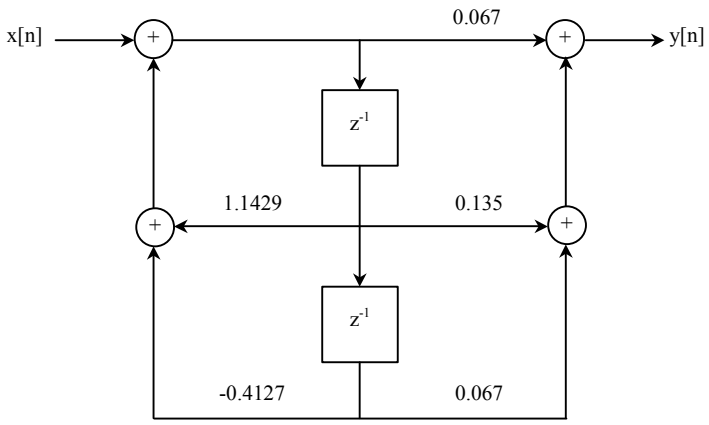

Figure 5.4: Direct realisation for a 2nd order Butterworth equivalent filter.

5.7 Z-Plane Poles and Zeros

A very useful representation of a z-transform is obtained by plotting its *poles* and *zeros* in the complex plane. It is quite easy to visualise the frequency response from such a diagram and it also gives a good indication of the degree of stability of a system.

The frequency-selective properties of first and second-order systems can be controlled by the appropriate choice of the pole-zero locations. Poles are particularly effective in this respect because when they are placed close to the unit circle they produce sharp, well-defined peaks in the frequency response. Usually an equal number of zeros are then placed at the z-plane origin (0, 0) to ensure that the impulse response begins at $n = 0$. The frequency response of a first-order system is defined by Equation 5.14 below; it has one real pole at the location $z = \alpha$ and one real zero at the origin $z = 0$. The frequency response of a second-order system is also defined by Equation 5.15; it has two poles (either both real or a

complex conjugate pair) at the locations $z = r \cdot \exp(j\theta)$ and $z = r \cdot \exp(-j\theta)$. In addition, it also has two zeros at the origin $z = 0$.

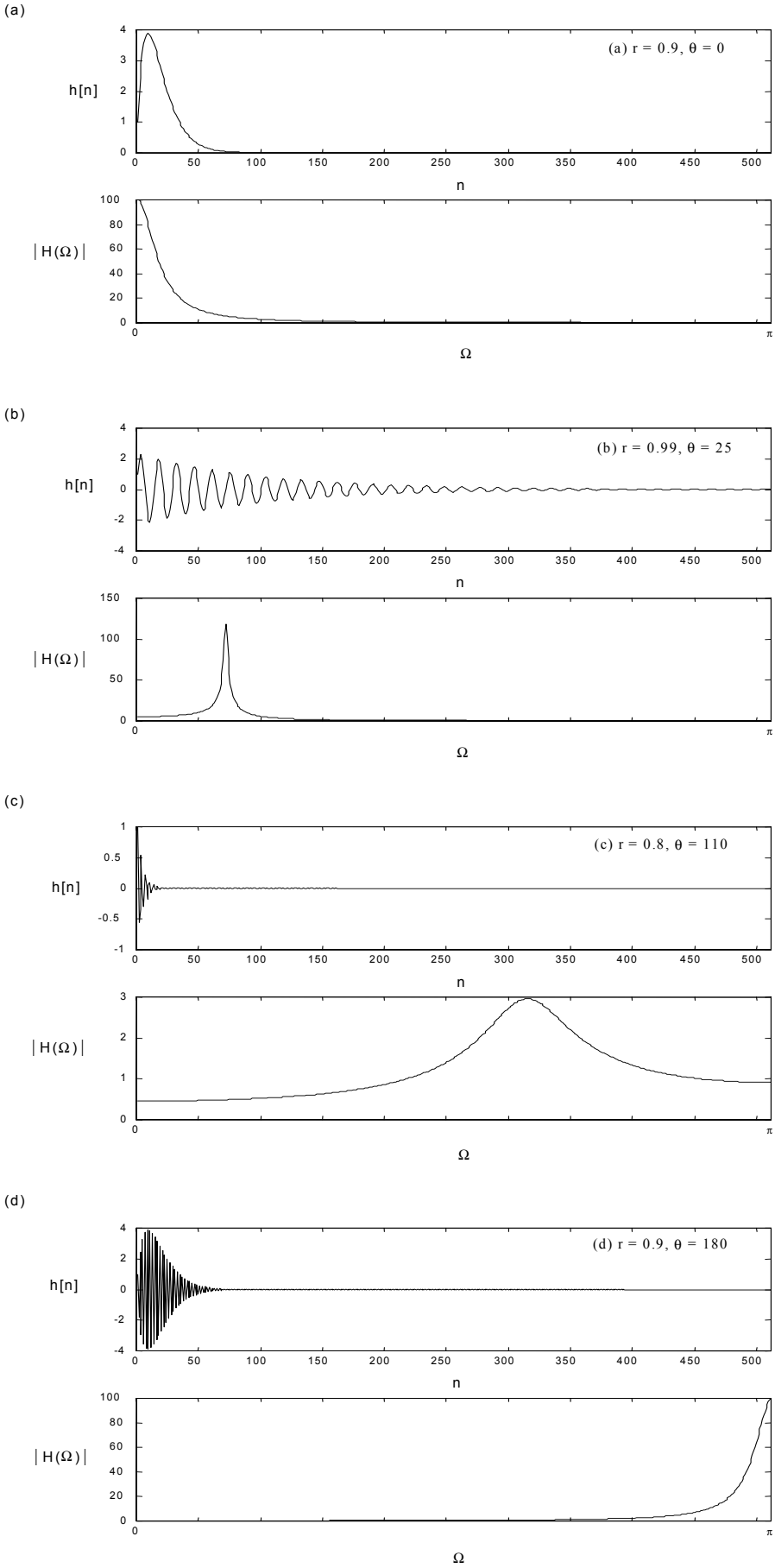

Figure 5.5: The impulse and frequency responses of several second-order systems.

$$H_1(z) = \frac{1}{1-\alpha z^{-1}} \tag{5.14}$$

$$H_2(z) = \frac{1}{[1-r\cdot\exp(j\theta)z^{-1}][1-r\cdot\exp(-j\theta)z^{-1}]} \tag{5.15}$$

By using the trigonometric representation of the *exponential* function, Equation 5.15 can be re-written as Equation 5.16, after multiplying out the denominator.

$$H_2(z) = \frac{1}{[1-2r\cdot\cos\theta\cdot z^{-1}+r^2\cdot z^{-2}]} \tag{5.16}$$

By changing the parameters r and θ, the impulse response $h[n]$ and frequency response magnitude $|H(\Omega)|$ vary. Some typical results are illustrated in Figure 5.3 for various second-order systems. In illustration (a) of Figure 5.3, the values of $r = 0.9$, $\theta = 0$ show that this configuration is a low-pass system with a second-order pole on the real axis in the z-plane. The choice of $r = 0.9$ gives a moderately selective frequency response. In illustration (b), $r = 0.99$ and $\theta = 25$ deg. The poles are now much closer to the unit circle, giving a very selective frequency-domain characteristic $|H(\Omega)|$. In the time-domain, the impulse response is prolonged, with the frequency of oscillation corresponding to $\theta = 25$ deg, which relates to 14 Hz. Diagram (c) illustrates the results for $r = 0.8$, $\theta = 110$ deg. This system is much less selective in the frequency domain, so its impulse response is short. Finally, (d) has $r = 0.9$ and $\theta = 180$ deg, producing a high-pass counterpart of the low-pass system shown in (a), but with the frequency response centered at $\Omega = \pi$.

5.8 Finite Word Length Effects in IIR Filters

In general IIR filters are much more difficult to analyse than FIR filters because of the feedback structure. However, both types of filter suffer from the same problems and have the same sources of noise due to finite word length effects. The extent of filter degradation depends on the length of the word and the type of arithmetic (fixed or floating point) used to perform the filtering operation. A summary of the main four sources of noise and their corresponding effects on IIR filter performance are summarised in Table 5.1.

Source of noise	Affect on performance	Reduction techniques
A/D conversion.	Quantisation noise = $q^2/12$.	• Increase number of bits. • Use multirate techniques.
Arithmetic round off.	Causes low level limit cycles i.e. oscillations at the filter output, or output stuck at a nonzero value, even when there is no input.	• Use double word length for intermediate results. • Optimise filter structure to include error spectral shaping. • Add a dither signal before rounding.
Coefficient quantisation.	Modifies position of the poles and zeros, may cause instability and a change in the frequency response.	• Use sufficient Nos. of bits in fixed-point representation. • Optimise selection of filter coefficients. • Use floating-point arithmetic.
Arithmetic overflow.	Incorrect output signal.	• Scale filter coefficients (at cost of reduced SNR). • Detect and use "maximum" rather than "overflowed" value. • Use floating-point arithmetic.

Table 5.1: Finite word length effects in IIR filters.

5.8.1 Arithmetic Round-Off

Arithmetic round off, can cause low-level limit cycles to occur in IIR filters. These can cause oscillations of the filter output, or the output to remain stuck at a non-zero value, even when there is no input. For example, consider the

following output of a 1st order IIR filter, as shown in Figure 5.6, with a 4-bit data and register length. Notice how the output oscillates between [-2, 2].

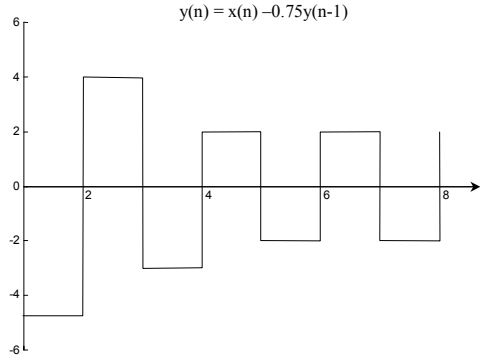

Figure 5.6: Low-level limit cycles caused by arithmetic round off.

Low-level limit cycles, as illustrated by Figure 5.6, can be reduced by using longer registers or by adding a dither signal before rounding. In addition, arithmetic round off can also be reduced by utilising feedback and feedforward paths in the 2nd order section, often known as error spectral shaping (ESS).

5.8.2 Coefficient Quantisation

If an error is introduced during coefficient quantisation, it can cause the poles and zeros to deviate from their expected positions and changes the desired frequency response. If a pole position is moved outside of the unit circle then this will cause instability in the filter. Now let us examine the effects of finite word lengths on the position of the pole-zero placement in the *z*-domain of the unit circle.

5.8.2.1 First-Order System

Consider a first-order system with a single pole at position $z = b$ and a zero at the origin, as depicted in Figure 5.7 below.

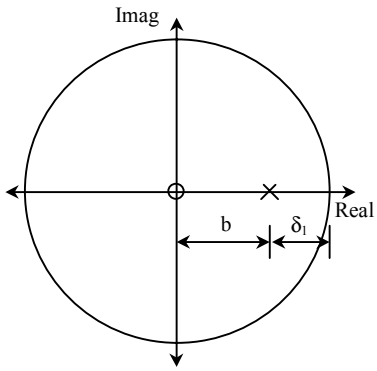

Figure 5.7: A single pole at $z = b$ in the z domain, and a zero at the origin.

The z-transfer function of this first-order filter is given by the equation below:

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

The change δ_1 in b that would cause the pole to lie at $z = 1$, is defined below:

$$1 - (b + \delta_1)z^{-1} = 0$$

$$1 - (b + \delta_1) = 0$$

$$\delta_1 = 1 - b$$

As an example, let us assume that the position of the pole was located at $b = 0.95$. Then, from the equation above, $\delta_1 = 0.05$. Now let us assume that the specification of the filter coefficient is not permitted to exceed 1% of the value of δ_1. Therefore the precision of the filter coefficient has to be accurate to within 0.0005. The minimum number of bits that is required to meet this specification, after rounding, is given below:

$$x = \log_2\left(\frac{1}{0.0005}\right) = \log_2(2000) = \frac{\log_{10}(2000)}{\log_{10}(2)} = 10.966 \approx 11$$

Furthermore, an additional bit has to be added to x for the *mantissa* (or sign) of the filter coefficient, so the total number of bits required to meet this specification is 12.

5.8.2.2 Double Pole at z=b

Now let us consider a second-order system with a double pole at position $z = b$ and two zeros at the origin. The z-transfer function of this second-order filter is given by this equation:

$$H(z) = \frac{1}{(1-bz^{-1})^2} = \frac{1}{1-2bz^{-1}+b^2z^{-2}} = \frac{1}{1+a_1z^{-1}+a_2z^{-2}}$$

The δ_2 change in coefficient a_1 that would cause one of the two poles to lie at $z = 1$, is defined by equation

$1+(a_1+\delta_2)z^{-1}+a_2z^{-2} = 0$, evaluated at $z = 1$:

$$1+(a_1+\delta_2)+a_2 = 0$$

$$\delta_2 = -(1+a_1+a)_2 = -(1-2b+b^2) = -(1-b)^2$$

Using the same value for $b = 0.95$, then δ_2 can be evaluated:

$$\delta_2 = -(0.05)^2 = -0.0025$$

with a corresponding coefficient wordlength requirement of:

$$x = \log_2\left(\frac{1}{0.000025}\right) = \log_2(4000) = \frac{\log_{10}(40000)}{\log_{10}(2)} = 15.287 \approx 16 \text{ (+1 for the sign bit)}$$

We can therefore conclude that fewer bits are required by implementing the filter as a cascade of two first-order sections rather than a single second-order section. It is fairly easy to generalise this result to higher orders and state that implementing a digital filter as a cascade of first or second-order sections always results in shorter coefficient wordlength requirements than if it were implemented as a single high order section.

5.8.2.3 Second-Order System

Now let us examine the case of a second-order complex conjugate pole pair with a double zero at the origin, as illustrated in Figure 5.8. The z-transfer function of this second-order filter is given by the equation below:

$$H(z) = \frac{1}{1+a_1z^{-1}+a_2z^{-2}} = \frac{1}{[1-2r\cdot\cos\theta\cdot z^{-1}+r^2\cdot z^{-2}]} \text{ where } r = a_2^{0.5} \text{ and } \theta = \cos^{-1}(-a_1/2r)$$

For stability the poles must lie within the unit circle, satisfying the conditions:

$0 \le |a_2| < 1$ and $|a_1| \le 1 + a_2$ (derivation not given here, and is not required for this course)

As an example, let us consider the second-order Butterworth filter designed in section 5.6.1. The coefficient values turned out to be a_1= -1.1429 and a_2 = 0.4127. Calculate δ_1 and δ_2 (corresponding to the two conditions above) that would put the poles on the unit circle.

$0 \le |0.4127+\delta_1| < 1$ resulting in the condition $|\delta_1| \le 0.4127$

$1.1429 + \delta_2 \le 1 + 0.4127$ resulting in the condition $\delta_2 \le 0.2698$

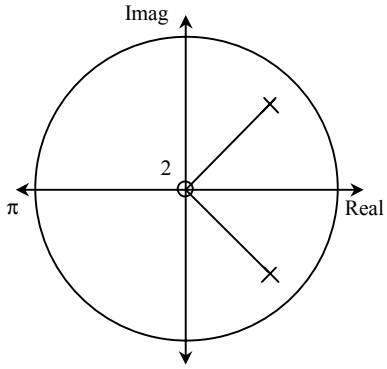

Figure 5.8: A complex conjugate pole pair and a double zero at the origin.

Now let us assume that the specification of the filter coefficient is not permitted to exceed 1% of the lowest value of δ (δ_2 = 0.2698). Therefore the precision of the filter coefficient has to be accurate to within 0.002698. The minimum number of bits that is required to meet this specification, after rounding up, is given below:

$$\frac{1.1429}{0.002698} \Rightarrow \frac{2}{0.002698} = 741.2898 = 2^x$$

$$x = \frac{\log_{10}(741.2898)}{\log_{10}(2)} = 9.5339 = 10 \quad \text{(rounded up)}$$

Furthermore, an additional bit has to be added to x for the *mantissa* (or sign) of the filter coefficient, so the total number of bits required to meet this specification is 11.