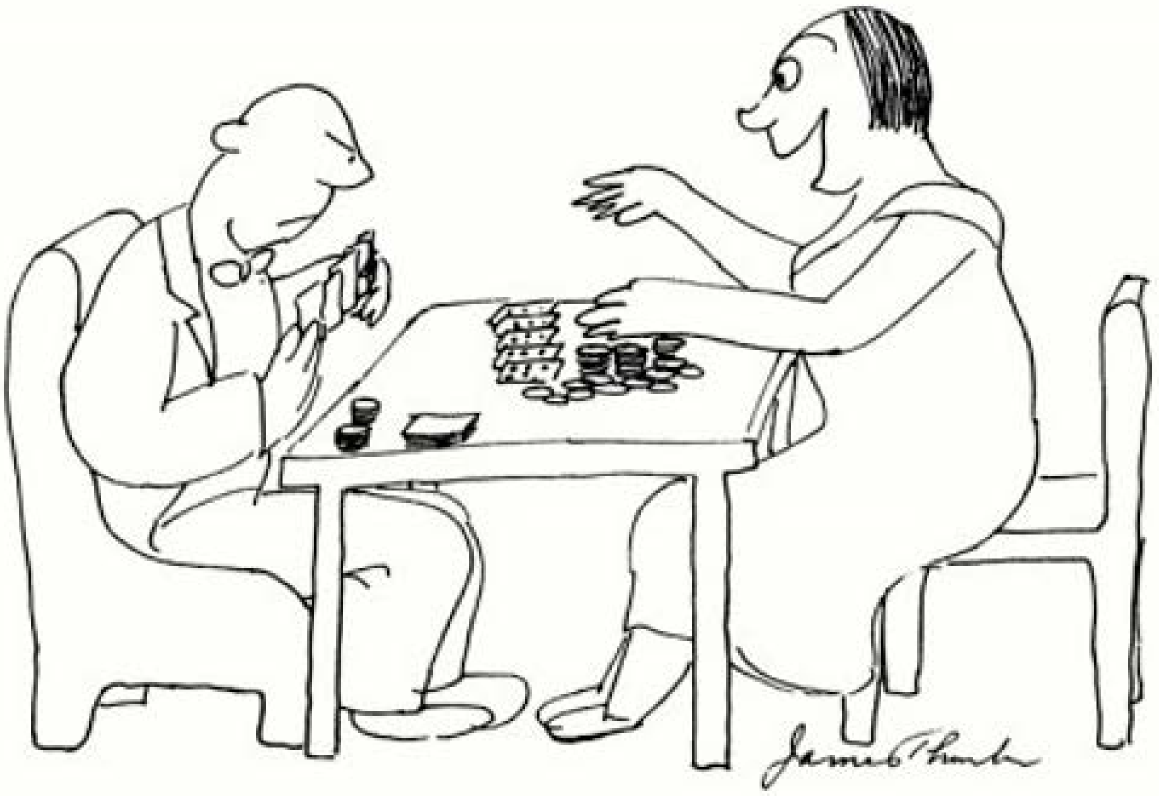


"I drew three more clubs and filled my flush!"

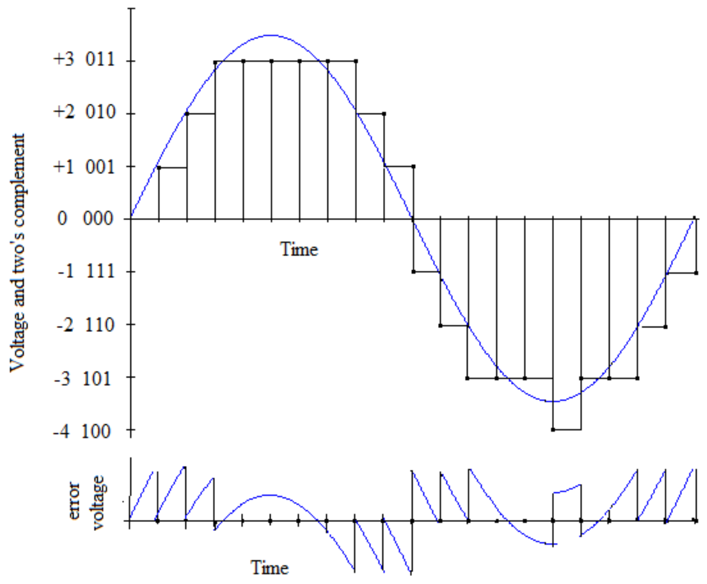


Figure 4.22
The top figure shows a quantized sinusoid as the darker stair stepped curve. The bottom figure shows the quantization error.

The *quantized signal to noise ratio* (QSNR) provides a way to measure the quantization error. The QSNR is defined by the equation:

$$QSNR = 10\log\frac{P_s}{P_n} \text{ decibels}$$

In this equation P_s is the signal power and P_n is the noise power. For a sinusoid of amplitude A, the average power is $A^2/2$.

For an A/D converter that is converting an analog signal into a b-bit digital signal we will assume that the sample rate is high enough, relative to the signal frequency, that the maximum quantization error never exceeds $\Delta/2$ where $\Delta = 2A/2^b$.

$$P_n = \frac{1}{T}\int_0^T (\Delta t/T - \Delta/2)^2\,dt$$

Evaluating this integral gives

$$P_n = \frac{\Delta^2}{12}$$

Substituting $2A/2^b$ for Δ gives

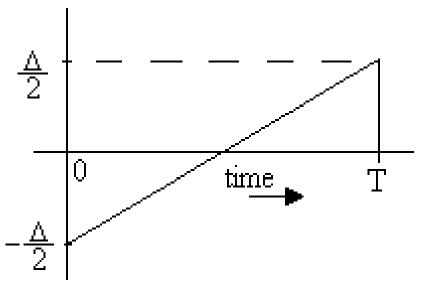


Figure 4.23
We assume that the error signal is a straight line from -Δ/2 to +Δ/2 in the range of one sample period.

$$\frac{P_s}{P_n} = \frac{12(A^2/2)}{4A^2/2^{2b}} = \frac{3}{2} \cdot 2^{2b}$$

Thus, the QSNR can be written in decibels as

$$QSNR \approx 6.02b + 1.76$$

DITHER

Example 4.5
Use MATLAB® to create half of a cycle of a sine wave with a frequency of 0.1 KHz which is sampled at 44.1 KHz. Quantize the sine wave to just 4 bits and plot the results. Add in a dither voltage to the original signal and again quantize the result to just 4 bits. Plot the graph of the dithered signal and compare it to the non-dithered version.

Solution
The MATLAB® m-file below creates a 0.1 KHz sine wave and quantizes it to just 4 bits. A dither signal is added to the sine wave and the result is again quantized to 4 bits. The two graphs are plotted for comparison in Figure 4.24.

```
fs = 44100;T = 1/fs;
f0 = 100;
Bits = 4;
% Create a half cycles of a sine wave at f0.
i = 1:220;
u = 0.5*sin(2*pi*f0*T*i);
%uquant is the quantized version of u.
uquant = u*2^(Bits-1);
uquant = fix(uquant);
uquant = uquant/2^(Bits-1);
%
figure(1);clf;
subplot(2,1,1)
Time = linspace(0,219*T,220);
stem(Time, uquant, 'MarkerEdgeColor', 'none')
axis([0 .005 0 .6])
%
% rand produces a flat distribution between 0 and 1.
```

```
dither = rand(1,length(u))/2^(Bits-1);
%ditherU is the original signal plus the dither.
ditherU =  u + dither;
%dquant is the quantized dithered signal
dquant = ditherU*2^(Bits-1);
dquant = fix(dquant);
dquant = dquant/2^(Bits-1);
subplot(2,1,2)
stem(Time,dquant, 'MarkerEdgeColor', 'none')
axis([0 .005 0 .6])
```

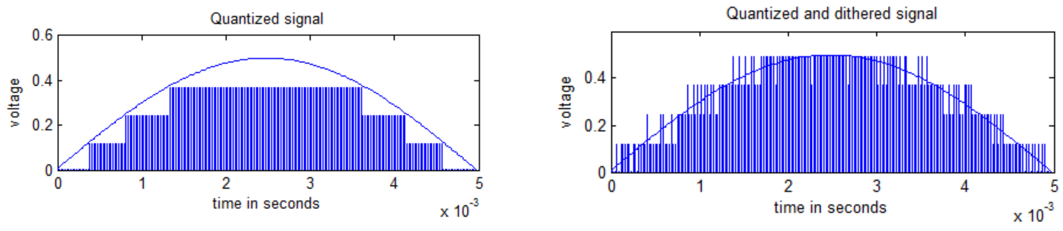


Figure 4.24
This figure shows a 1 KHz sine wave that has been quantized to just 3 bits. In figure at right dithering has been added. The original half sine wave has been superimposed.

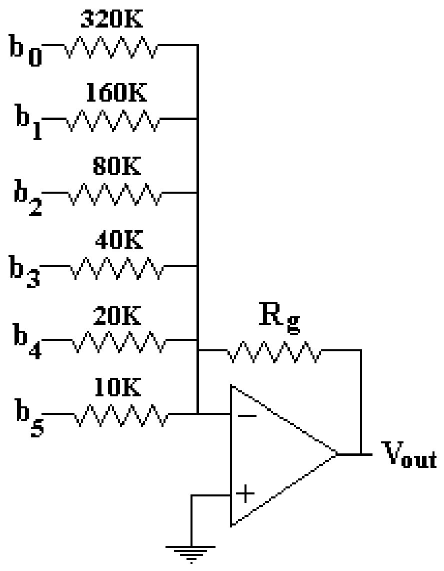


Figure 4.26
A weighted resistor type of D/A converter. In this converter the resistors are the weighting factors and the op amp sums the current.

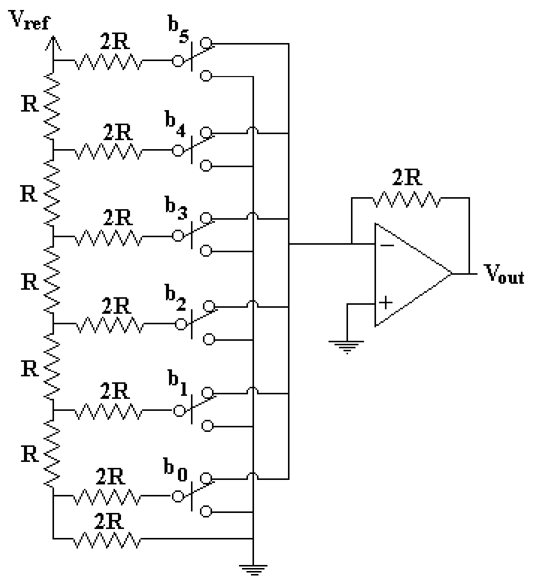


Figure 4.27
A six bit ladder type D/A converter. Note that only two different values of resistors are used. The digital input bits (b0 to b5) set the switches. The switches are shown in the "1" bit position.

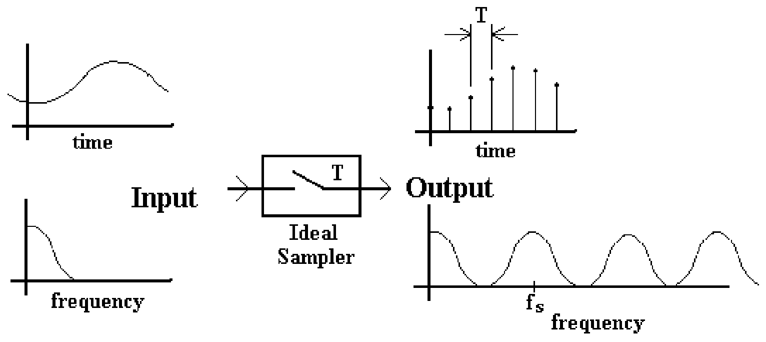


Figure 4.28
This figure shows the typical input and output signals for an ideal sampler. Both time domain and frequency domain are shown. The sample rate is $f_s = 1/T$.

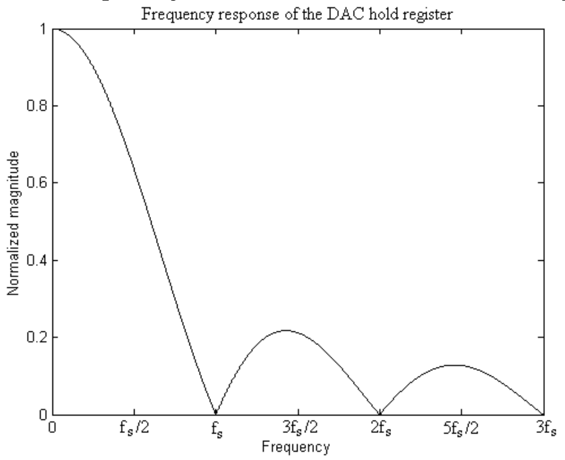


Figure 4.29
Frequency response function for a register and D/A converter which produces a stair step output.

4.6 Anti-Imaging Filters

The frequency spectrum of a signal which passes through an ideal sampler looks identical to that of the signal going into the sampler except that it is repeated multiple times in frequency space. The spacing is determined by the sampling frequency as shown in Figure 4.28.

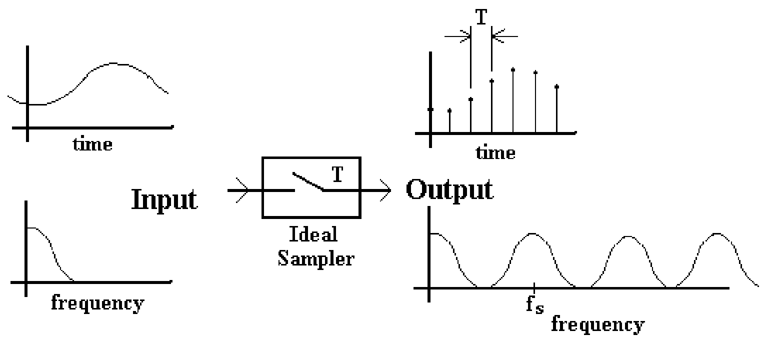


Figure 4.28
This figure shows the typical input and output signals for an ideal sampler. Both time domain and frequency domain are shown. The sample rate is $f_s = 1/T$.

It is evident from examination of the frequency spectrum of the sampled signal that we could recover the original spectrum if we could eliminate the unwanted repeated images of the original frequency spectrum. To do this we need an ideal low pass filter which passes all frequencies below $f_s/2$ with a gain of 1 and stops all frequencies above $f_s/2$.

It is impossible to construct an ideal low pass filter that operates in real time since such a filter has an impulse response which is noncausal. We can, however, approximate such a filter. Indeed, holding the DSP output in a register between samples to produce the stair step effect from the D/A converter provides some low pass filtering.

The register-D/A converter arrangement has a frequency response function shown in Figure 4.29 (See Problem 4.11 for a derivation). In many cases this is all the filtering that is necessary for an application. The stair step output form is suitable for digital control of a d.c. motor for example. In applications where the hold register alone does not produce suitable filtering, more complex output arrangements must be made.

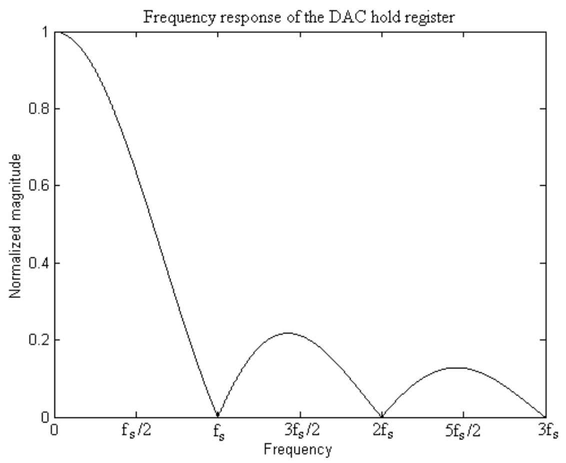


Figure 4.29
Frequency response function for a register and D/A converter which produces a stair step output.

In some systems, it is possible to compensate for the attenuation in the 0 to $f_S/2$ band by adding some gain to the digital output *before* it gets to the D/A. Alternatively, compensation may added after the D/A in the form of an analog filter. A third alternative is to use a different type of converter such as the Sigma-Delta converter.

Example 4.6
A signal sampled at f_s goes to a D/A converter and then to a hold register. What should be done to the signal before it goes to the D/A to compensate for the effects of the hold register.

Solution
Figure 4.29 shows the filtering effects of the hold register. To compensate for the effects of this filtering we need to add gain before the D/A such that the product of the gain and the filter is unity from 0 to $f_S/2$. The equation for this curve is given by

$$H(\omega) = (1 - e^{-j\omega T}) / j\omega T$$

To get the frequency response of the compensating filter we do the following:

```
   fs = 1000;T = 1/fs;
   f = 1:fs/2;w = 2*pi*f;
   H = (1 - exp(-j*w*T))./(j*w);
   Ideal = ones(1, length(H));
   figure(1);clf;
   %Normalize magnitude
   H = abs(H)/max(abs(H));
   plot(f, H);
   hold on;
   %Divide normalized magnitude into 1 to get compensator
   Hc = Ideal./abs(H);
   plot(f, Hc);
   xlabel('Frequency');
   ylabel('Normalized magnitude');
   title('Response of the D/A and hold register');
```

Figure 4.30 shows the result.

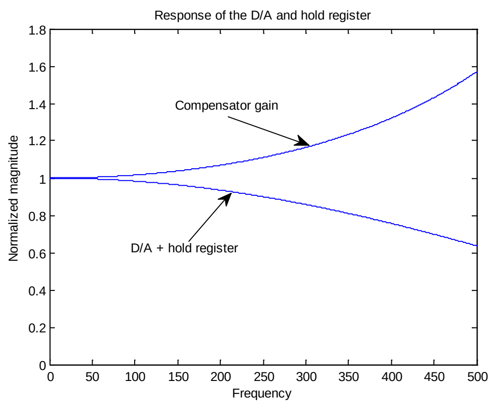


Figure 4.30
Compensation that is to be added to the signal before it goes to the D/A and the hold register.

Thus we see that a digital signal processor may be preceded by an analog low pass filter on the front end and an analog low pass filter on the back end. The front end filter serves to attenuate spurious frequency components from the incoming signal that are above $f_s/2$. The back end filter serves as a reconstruction filter by removing those extra frequency components introduced by the sampling process.

In some applications oversampling of the output signal can be used to reduce the cost and improve the quality of an analog reconstruction filter. Intuitively, if we consider a simple RC low pass filter with a sampled signal on the input, the capacitor stores charge at the sample points and uses that charge to "fill in" between the sample points to smooth the signal. If the sample frequency is higher, the distance between samples is smaller and the size of the capacitor used can be reduced. This leads to lower cost and smaller size.

For audio systems the sample frequency is typically a few tens of kilohertz. Increasing this sample frequency to a few hundred kilohertz reduces the size of the analog reconstruction filter. The process of increasing the sample rate of a given signal is referred to as *oversampling* or *interpolation* since the value of the new inserted samples is interpolated from the neighboring values.

Interpolation filters find application in CD players and in some control applications. A CD player may have four times over sampling which means that an interpolation filter is used prior to the signal going to the speaker to make it appear as if the sample rate was four times the sample rate of the information stored on the disk. Likewise, in control systems, at low frequencies, it is often computationally easier to use an interpolation filter prior to sending a control signal to a continuous time motor than it is to provide the high quality analog filter that may otherwise be necessary.

Linear interpolation provides an easy way to think about how interpolation works. In Figure 4.31, linear interpolation is applied to an impulse response function.

To make a practical linear interpolator we will insert zeros between the sample points of the incoming sequence (see Figure 4.32). An interpolation filter will replace the zeros in the sequence with the average value of the previous point and the next point. Since the filter cannot look into the future, a single delay element will have to be introduced in the input sequence.

Although the linear interpolation filter is simple to implement it does not always provide satisfactory results. A better solution may be found by examining the problem in the frequency domain.

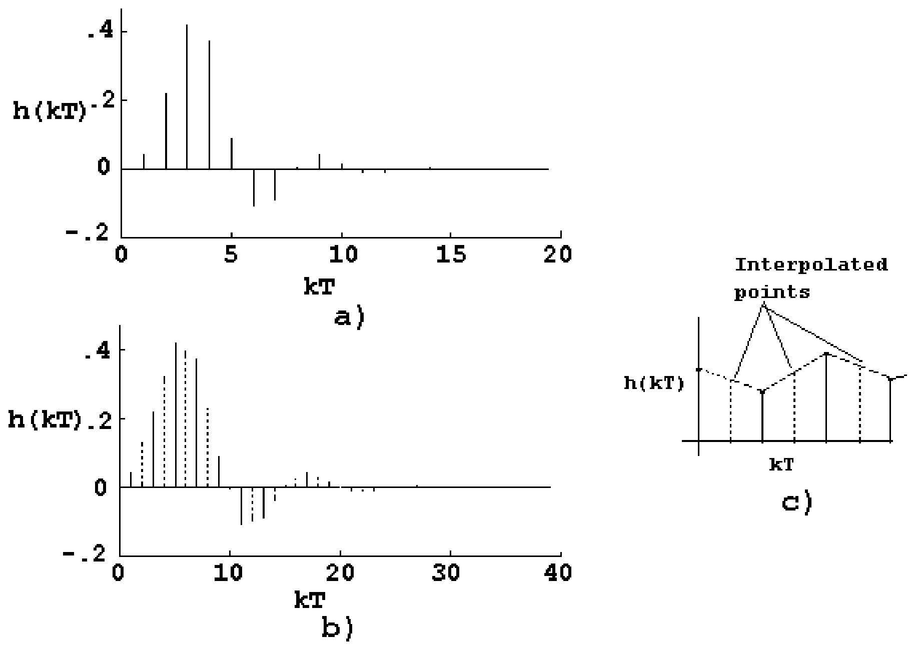


Figure 4.31
This figure illustrates linear interpolation. a) an original impulse response function. b) The impulse response function with interpolated values inserted. Note that the sample rate has doubled. c) Linear interpolation.

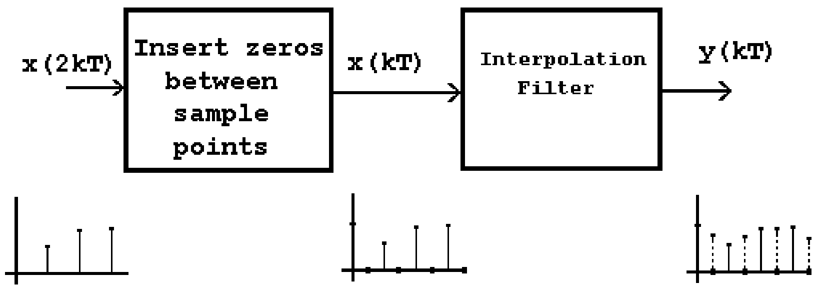


Figure 4.32
Implementation of a linear interpolation filter.

In Figure 4.33 the signal labeled |H(ω)| represents the magnitude function of a sampled signal. This function is necessarily periodic due to the sampling and the aliasing. If |H(ω)| is passed through an ideal low pass filter the result is a single copy of the frequency spectrum. Thus, an ideal low pass filter can serve as an ideal interpolation filter.

In the time domain the impulse response of an ideal low pass filter can be found as

$$h_{Ideal}(kT) = \int_{-\infty}^{\infty} H_I(j\omega)e^{j\omega t}d\omega = \int_{-\omega_S/2}^{\omega_S/2} (1)e^{j\omega t}d\omega$$

$$h_{Ideal}(t) = \frac{1}{2\pi jt}e^{j\omega t}\Big|_{-f\omega_s/2}^{f\omega_s/2} = \frac{e^{j\omega_s t/2} - e^{-j\omega_s t/2}}{2\pi jt} = \frac{2j\sin(\omega_s t/2)}{2\pi jt} = \frac{\omega_s}{2\pi}\text{sinc}(\omega_s t/2) \qquad 4.5$$

In the frequency domain the ideal filter is multiplied times the sampled spectrum. In the time domain this multiplication corresponds to convolution. We can write the impulse response of an interpolator as

$$h_{\text{int}}(t) = h(t) * \frac{\omega_s}{2\pi}\text{sinc}(\omega_s t/2)$$

or

$$h_{\text{int}}(t) = \frac{\omega_s}{2\pi}\int_{-\infty}^{\infty} h(\tau)\cdot\text{sinc}[\omega_s(t-\tau)/2]d\tau$$

In terms of a discrete time function this equation can be written as

$$h_{\text{int}}(t)\Big|_{t=kT} = h_{\text{int}}(kT) = \frac{\omega_s}{2\pi}\sum_{k=-\infty}^{\infty} h(\tau)\cdot\text{sinc}[\omega_s(kT-\tau)/2] \qquad 4.6$$

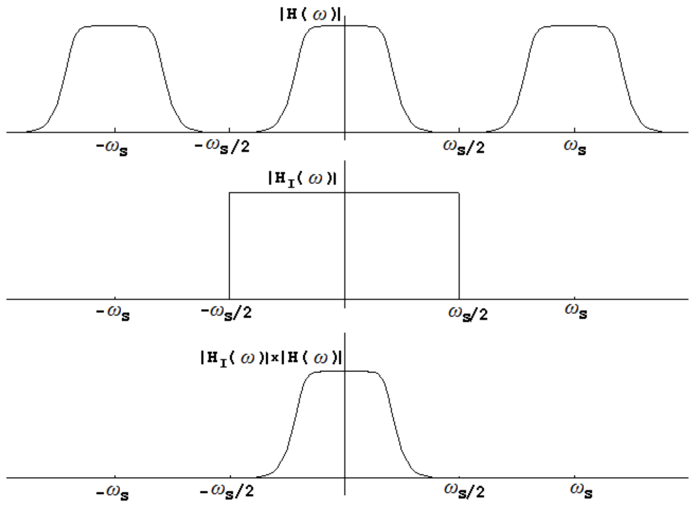
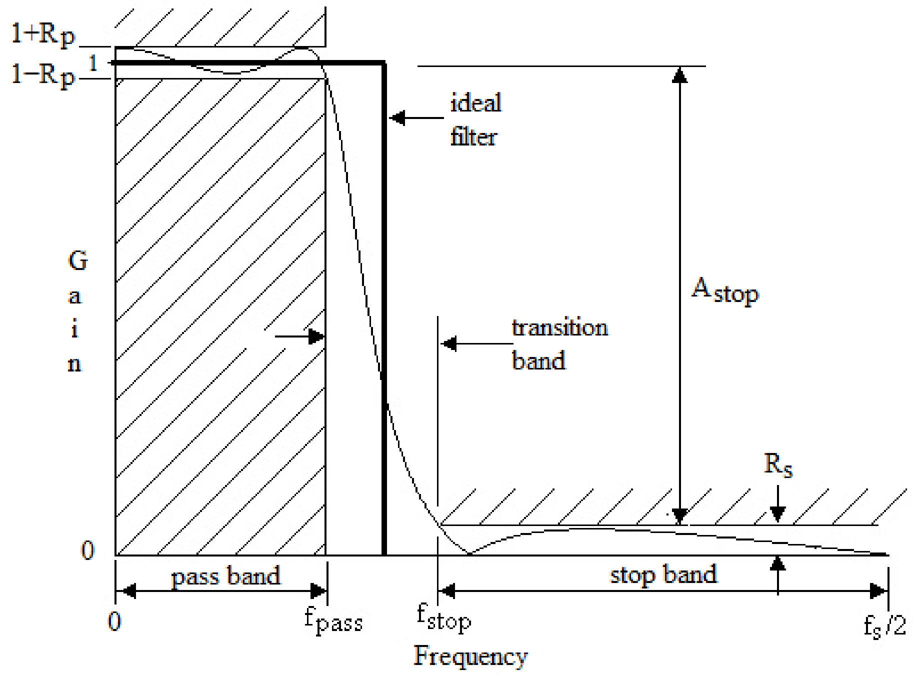


Figure 4.33
When the frequency spectrum of a sampled signal is passed through an ideal low pass filter we recover the original spectrum.

From Equation 4.6 we see that we can interpret the interpolation equation as a weighted sum of delayed sinc functions. The sum, of course, is infinite in extent and noncausal so that an ideal interpolation function is impossible to implement. We can however approximate the ideal interpolation function with a low pass filter which has a fast transition band. For applications, such as audio, where phase can make a difference, an FIR filter is appropriate. For other applications, such as driving a motor in a control application, an IIR filter may provide better computational efficiency. We take up the subject of oversampling and interpolation filters in more detail in Chapter 8.

Chapter 5

The graph in Figure 5.1 is a *linear plot* but it is traditional to specify ripple and attenuation specification in decibels. The decibel is a unit for measuring the loudness of sound or sound pressure. In signal processing and communications applications, it is a measurement of signal power. The decibel is defined by the equation $db = 10\log_{10}(P_{out}/P_{in})$. Since power is proportional to voltage squared the decibel can be written as $db = 20\log_{10}(V_{out}/V_{in})$. The stop band attenuation is defined in decibels as $-20\log_{10}(R_s)$ which is also referred to as the decibel ripple specification for the stop band. In the pass band the filter gain is normalized to one and the pass band ripple can go both above and below 1 as shown. The decibel pass band ripple is defined as $-20\log_{10}(1-R_p)$.



Design using the Fourier series.

If the Fourier series is limited to $2N + 1$ terms, we will call this approximation $F_N(t)$ where

$$F_N(t) = \frac{a_0}{2} + \sum_{k=1}^{N} a_k \cdot \cos(\omega_0 kt) + \sum_{k=1}^{N} b_k \cdot \sin(\omega_0 kt) = \sum_{k=-N}^{N} C_k \cdot e^{jk\omega_0 t}$$

From equations 3.5 and 3.14 we know that if a periodic function is approximated by a Fourier series for a finite number of terms, the *integral mean square error* is minimized for that number of terms.

To use this information in the design of an FIR filter recall that sampling in the time domain causes periodicity in the frequency domain and vice versa. An ideal low pass filter which is discrete in time will have a frequency characteristic similar to that shown in Figure 5.2.

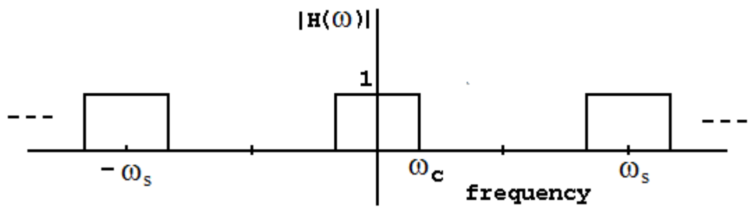


Figure 5.2
An ideal low pass filter that is sampled in the time domain.

If the signal in Figure 5.2 was in the time domain, we could find its Fourier series in the frequency domain. Since it already in the frequency domain we can do a change of variables: Let $t \leftarrow f$ so that $T_0 \leftarrow f_s$

The exponential form of the Fourier series is:

$x(t) = \sum_{k=-\infty}^{\infty} C_k e^{jk\omega_0 t}$ where $C_k = \frac{1}{T_0}\int_{T_0} x(t)e^{-jk\omega_0 t}dt$ and $\omega_0 = \frac{2\pi}{T_0}$

When we do our variable transformation we get

$$H(\Omega) = \sum_{k=-\infty}^{\infty} C_k e^{jk2\pi f/f_s}$$

where

$$C_k = \frac{1}{f_s}\int_{f_s} H(\Omega)e^{-j2\pi kf/f_s}df.$$

Applying these equations to the signal of Figure 5.2 we get the following:

$$C_k = \frac{1}{f_s}\int_{-f_c}^{f_c} (1)e^{-j2\pi kf/f_s}df = \frac{1}{-2\pi kj}e^{-j2\pi kf/f_s}\Big|_{-f_c}^{f_c} = \frac{1}{k\pi}\sin(2\pi kf_c/f_s) \qquad \textbf{(5.1)}$$

If the equation for $H(\Omega)$ is regarded as an evaluation of a transfer function $H(z)$ on the unit circle, we can replace $e^{j2\pi f/f_s} \leftarrow z$. This gives a transfer function in z with the problem that it is infinite in extent and noncausal.

$$H(z) = \sum_{k=-\infty}^{\infty} C_k z^{-k}$$

If we truncate this infinite series to $L = N+1$ terms and shift it in time by $N/2$ places we get the following causal, finite, transfer function in z. (N is the filter order and L is the filter length).

$$H(z) = \sum_{k=0}^{N} C_{k-N/2} z^{-k} \qquad \textbf{5.2}$$

Example 5.2
Choose a sample frequency f_s = 11,025Hz, cutoff frequency f_c = 2,000Hz and order N = 10 (length = 11) find the coefficients for a transfer function in z using the Fourier series method and the ideal filter characteristic of Figure 5.2.

Solution

From equation 5.1, $C_k = \frac{1}{k\pi} Sin(2\pi k f_c / f_s)$. Letting *k* go from -5 to +5 gives the following values for C$_k$

k	***C$_k$***
-5	-0.0351090
-4	-0.0786459
-3	-0.0291006
-2	0.1208196
-1	0.2892013
0	0.3628118
1	0.2892013
2	0.1208196
3	-0.0291006
4	-0.0786459
5	-0.0351090

Table 5.1
Tabulated values of C_k using equation 5.1.

From equation 5.2 the transfer function for this FIR filter becomes:

$$H(z) = \frac{-0.03511z^{10} - 0.07865z^{9} - 0.02910z^{8} + \cdots - 0.07865z - .03511}{z^{10}}$$

The corresponding magnitude and phase plots are shown in Figure 5.3.

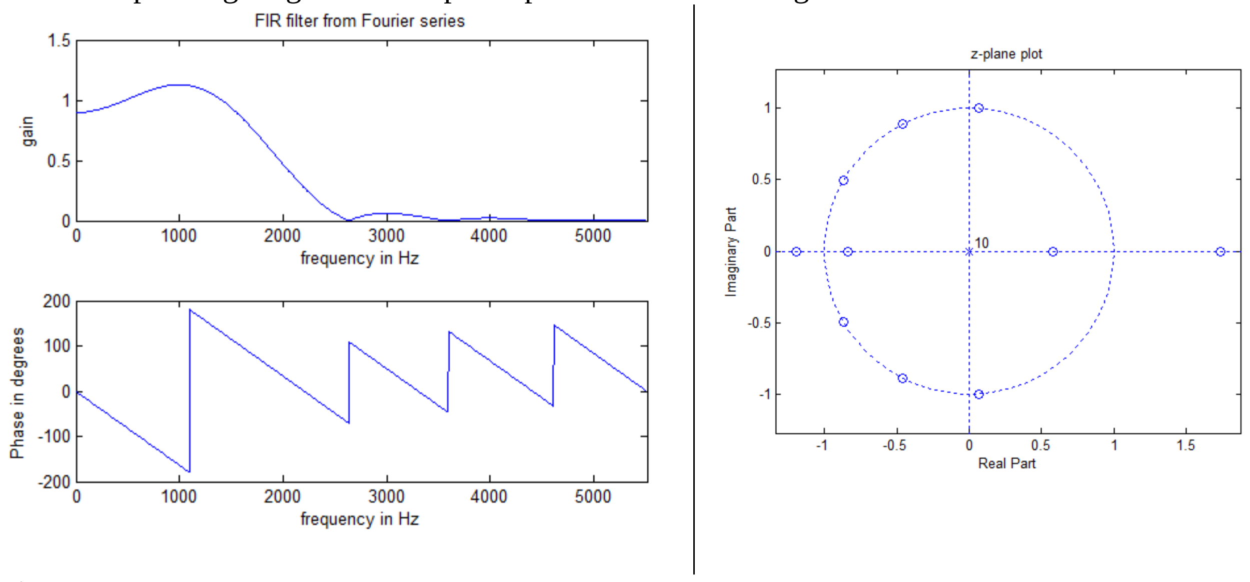


Figure 5.3
Magnitude and phase plots for the low pass FIR filter with a cutoff frequency at 2,000Hz.

The phase curve is said to be piecewise linear with the discontinuities after the first caused by zeros located directly on the unit circle.

The Fourier series design technique is implemented in MATLAB® as the function *firls*. The following lines of MATLAB® code produce a frequency response identical to that shown in Figure 5.3.

```
N = 10;                              %Order
fs = 11025;
F = [0 2000/(fs/2) 2000/(fs/2) 1];   %Frequency vector normalized to fs/2
M = [1 1 0 0];                       %Corresponding magnitude vector
num = firls(N, F, M);
[H f] = freqz(num, 1, 1024, fs);
figure(1);
subplot(2, 1, 1)
plot(f, abs(H));
axis([0 fs/2 0 1.5]);
xlabel('frequency in Hz');
ylabel('gain');
title('FIR filter from Fourier series');
subplot(2, 1, 2);
plot(f, angle(H)*180/pi);
axis([0 fs/2 -200 200]);
xlabel('frequency in Hz');
ylabel('Phase in degrees');
figure(2);clf;
zplane(num, 1);
title('z-plane plot');
```

Linear phase and FIR filters

In general, FIR filters are computationally less efficient than IIR filters that meet the same specifications. However, FIR filters can be designed to have linear phase whereas IIR filters can only approximate linear phase and often, that is over a small band. Hence, for applications where information is encoded as part of the phase of a signal, having a filter that does not distort the phase curve is important.

We define two new terms related to phase:

$$\tau_p = -\frac{\theta(\omega)}{\omega} \equiv \textit{Phase Delay} \qquad \textbf{(5.3)}$$

$$\tau_g = -\frac{d\theta(\omega)}{d\omega} \equiv \textit{Group Delay} \qquad \textbf{(5.4)}$$

The phase delay is simply the phase function normalized to the frequency. It represents the time delay in seconds that a particular sinusoid experiences as it passes through a system. For example, a single unit delay will have a transfer function of z^{-1} and a phase function of

$\theta(\omega) = -\omega T$

.

The phase delay is

$\tau_p = \omega T / \omega = T$

and has units of seconds. In this case the phase delay is constant and all frequencies are delayed by *T* seconds.

Conditions for linear phase in FIR filters

For a filter with a constant phase delay and a constant group delay the phase response must be of the form:

$\theta(\Omega) = -m\Omega$ where $\Omega = \omega T$ **(5.6)**

This is the equation for a straight line with a slope of *-m* giving us a linear phase response.

To determine what conditions produce a constant phase delay, constant group delay, and a linear response, we write the frequency response for an FIR filter from the transfer function.

$$H(z) = \sum_{n=0}^{L-1} h[n] z^{-n}$$

$$H(z)\big|_{z=e^{j\Omega}} = \sum_{n=0}^{L-1} h[n] e^{-j\Omega n} = \sum_{n=0}^{L-1} h[n][\cos(\Omega n) - j\sin(\Omega n)]$$

From the frequency response we can write the phase response as the following:

$$\theta(\Omega) = \angle[H(e^{j\Omega})] = \tan^{-1}\left[\frac{-\sum_{n=0}^{L-1} h[n]\sin(\Omega n)}{\sum_{n=0}^{L-1} h[n]\cos(\Omega n)}\right] = -m\Omega \quad \textbf{(5.7)}$$

Since the tangent is the sine over the cosine we can rewrite this equation as:

$$\tan(\theta(\Omega)) = \frac{-\sum_{n=0}^{L-1} h[n]\sin(\Omega n)}{\sum_{n=0}^{L-1} h[n]\cos(\Omega n)} = \frac{\sin(-m\Omega)}{\cos(-m\Omega)}$$

If we cross multiply in this equation and put everything on the left side of the equals sign we get an equation which can be written as the sine of the sum of two angles.

$$\sum_{n=0}^{L-1} h[n]\sin(m\Omega)\cos\Omega n) - \sum_{n=0}^{L-1} h[n]\sin\Omega n)\cos(m\Omega) = 0$$

or, by applying the angle-difference relation of sines,

$$\sum_{n=0}^{L-1} h[n]\sin(m\Omega - \Omega n) = 0$$

The solution to this equation becomes more apparent if we expand it.

$$h[0]\sin(m\Omega) + h[1]\sin(m\Omega - \Omega) + h[2]\sin(m\Omega - 2\Omega) + \cdots$$
$$+ \quad h[L-2]\sin\left(m\Omega - \Omega[L-2]\right) + h[L-1]\sin\left(m\Omega - \Omega[L-1]\right)$$

We see that we may satisfy this equation under the following conditions:

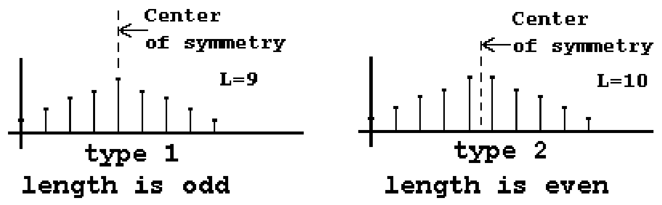
$$\sin(m\Omega) = -\sin\left(m\Omega - \Omega[L-1]\right) \quad \textit{and} \quad h[0] = h[L-1]$$
$$\sin(m\Omega - \Omega) = -\sin\left(m\Omega - \Omega[L-2]\right) \quad \textit{and} \quad h[1] = h[L-2]$$
$$\cdots$$
$$\sin(m\Omega - n\Omega) = -\sin\left(m\Omega - \Omega[L-n-1]\right) \quad \textit{and} \quad h[n] = h[L-n-1]$$

or

$$m\Omega - n\Omega = -m\Omega + \Omega(L-n-1)$$

$$m = \frac{(L-1)}{2} \quad \text{and} \quad h[n] = h[L-n-1] \tag{5.8}$$

Equation 5.8 gives conditions for having a linear phase response for an FIR filter. The equation for $h(n)$ stipulates that the impulse response must be symmetrical. There are two ways of achieving this symmetry depending on whether L is even or odd. Figure 5.5 shows typical impulse response functions of even and odd lengths with the two kinds of symmetry.



(a) **(b)**

Figure 5.5

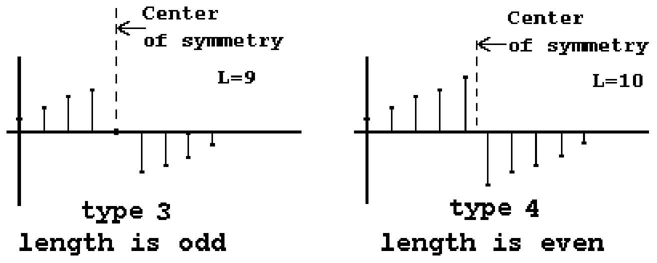
Typical impulse response functions with (a) odd and (b) even lengths. Both functions are symmetrical and therefore produce phase responses which are linear.

An FIR filter whose impulse response is symmetrical with an odd length (even order) as shown in Figure 5.5(a) is referred to as a *type 1* filter. If the length is even (odd order) the filter is referred to as a *type 2* filter.

In determining the conditions necessary for linear phase we had assumed that the phase response was of the form $\theta(\Omega) = -m\Omega$. This is the equation for a straight line that passes through the origin giving a constant phase delay, constant group delay, and a linear phase response. We could also write a straight line as $\theta(\Omega) = -m\Omega + \theta_0$ where θ_0 is a constant. If we use this equation the phase delay will no longer be a constant but will be a function of ω. The group delay however, will be constant. If we use this condition in equation 5.7 the solution can be calculated as the following (taking $\theta_0 = \pm\pi/2$):

$$m = \frac{(L-1)}{2} \quad \text{and} \quad h[n] = -h[L-n-1] \tag{5.9}$$

The impulse response is said to be anti-symmetric. Typical impulse response functions are shown in Figure 5.6 for the cases where the length is odd (type 3) and even (type 4).



(a) **(b)**

Figure 5.6

Typical impulse response functions with (a) odd and (b) even lengths. Both functions are anti-symmetrical and therefore produce phase responses which are linear.

For type 2 and type 4 FIR filters where the length of the filter is even, it is necessary to shift the filter in time by a fraction of a sampling period in order to get a causal filter. For example, In Figure 5.6(b), the center of symmetry falls between two samples so that the filter had to be shifted four and half samples to make it causal. It can be shown that the frequency response of type 1 and type 2 filters can be written as a sum of cosine functions while the response of type 3 and type 4 filters can be written as a sum of sine functions. The frequency response of the four filters therefore, places some restrictions on their usefulness. These restrictions are summarized in Table 5.2.

FIR filters of even length can be created by resampling the impulse response of an odd length filter in such a way as to achieve even symmetry. This is illustrated in Example 5.3.

Type	Symmetry	Restrictions	Suitability
1	Odd length Symmetrical	None	Lowpass Highpass Bandpass Bandstop
2	Even length Symmetrical	$H(f_s/2)=0$	Lowpass Bandpass
3	Odd length AntiSymmetric	$H(0)=0$ $H(f_s/2) = 0$	Highpass Differentiator
4	Even length antiSymmetric	$H(0)=0$	Highpass Differentiator

Table 5.2
Restrictions and suitability for the various types of linear phase FIR filters.