"I come from haunts of coot and hern!"

EE 311 **February 22, 2019**
Lecture 13

Minimum Phase FIR filters

Geometric Interpretation of P/Z locations

Minimum Phase System - A system is said to be a minimum phase system if the deviation from zero in the phase curve is the smallest possible for the given magnitude plot.

Maximum Phase System - A system is said to be a maximum phase system if the deviation from zero in the phase curve is the maximum possible for the given magnitude plot.

Mixed Phase System - A system is said to be of mixed phase if the deviation from zero in the phase curve from 0 Hz and $f_s/2$ lies between that of the minimum phase and the maximum phase curves.

The definitions for the minimum and maximum phase systems above imply that there is more than one phase curve for a given magnitude curve. In general, this is true. Consider a system which has a zero at $z = \alpha$ where α is a real number. The magnitude of this zero term in a transfer function evaluated around the unit circle is

$$|z-\alpha| = \sqrt{(\cos(\Omega)-\alpha)^2 + \sin^2(\Omega)}$$

Expanding this equation give

$$|z-\alpha| = \sqrt{\cos^2(\Omega) - 2\alpha\cos(\Omega) + \alpha^2 + \sin^2(\Omega)} = \sqrt{1 - 2\alpha\cos(\Omega) + \alpha^2}$$

Now consider a second system which has a zero located at $z = 1/\alpha$. The magnitude of this zero term in a transfer function evaluated on the unit circle is

$$|z - 1/\alpha| = \sqrt{1 - (2/\alpha)\cos(\Omega) + 1/\alpha^2}$$

But this equation can be rewritten as

$$|z - 1/\alpha| = (1/\alpha)\sqrt{\alpha^2 - 2\alpha\cos(\Omega) + 1}$$

Thus we see that $|z - 1/\alpha| = (1/\alpha)|z - \alpha|$ which means that the magnitude curve for a system with a zero located at α will be the same shape as the magnitude curve with a zero located at $1/\alpha$. The two curves will differ only by a gain factor. However, the phase curves for the two systems are very different, as is shown in the following example. The same argument applies if the zero is complex with just a little more algebra.

Example 5.14
Plot the magnitude and phase plots for the two systems given by

$$H_1(z) = 0.3279\frac{z + (.6 \pm j.7)}{z^2} = 0.3279\frac{z^2 + 1.2z + .85}{z^2}$$

and

$$H_2(z) = 0.2787\frac{z + (.70588 \pm j.82353)}{z^2} = 0.2787\frac{z^2 + 1.4118z + 1.1765}{z^2}$$

Note that the reciprocal of 0.6 ± j0.7 is 0.70588 ± j0.82353.

Solution:

The MATLAB® code below produces the required plots which are shown in Figure 5.34.

```
fs = 2;
num1 = 0.3279*[1 1.2 .85];
num2 = 0.2787*[1 1.4118 1.1765];
den = [1 0 0];
[H1 F] = freqz(num1,den,512,fs);
subplot(2,3,1)
 plot(F,abs(H1))
subplot(2,3,2)
 Theta = unwrap(angle(H1))*(180/pi);
 plot(F,Theta)
subplot(2, 3, 3);
 zplane(num1, den);
%
[H2 F] = freqz(num2,den,512,fs);
subplot(2,3,4)
 plot(F,abs(H2))
subplot(2,3,5)
 Theta = unwrap(angle(H2))*(180/pi);
 plot(F,Theta)
subplot(2, 3, 6);
 zplane(num2, den);
```

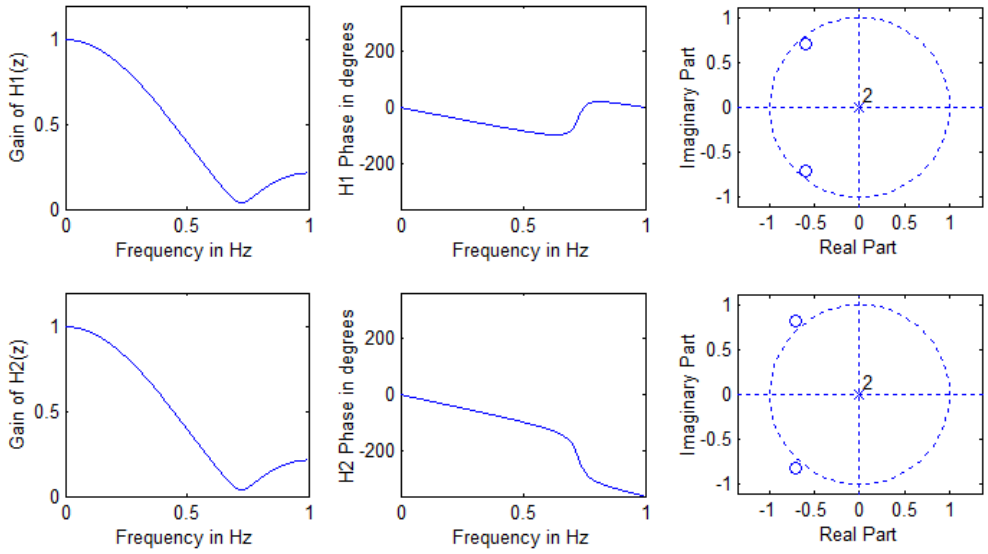
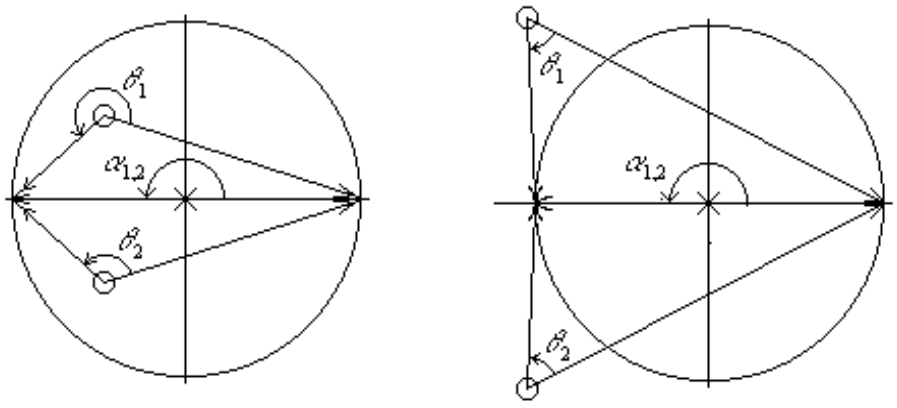


Figure 5.34

Magnitude and phase plots for systems with reciprocal complex zeros. The magnitude plots differ only by a gain factor while the phase curves are radically different. $H_1(z)$ at top, is referred to as a minimum phase system and $H_2(z)$ at bottom, is referred to as a maximum phase system.

Example 6.9 illustrates that two transfer functions can produce the same magnitude curve if there are zeros which are the reciprocal of one another. (The same argument could be made for poles except that reciprocal poles will produce an unstable system.) Thus if we have a system which has *N* complex zero pairs not on the unit circle and *M* real zeros not on the unit circle we can write 2^{N+M} transfer functions which have the same magnitude plot but different phase plots. Since phase shift corresponds to time delay it becomes important in some systems to know which of the transfer functions that are possible for given magnitude specification has the minimum phase deviation.

Consider the second order system consisting of a complex zero pair and two poles located the origin shown in Figure 5.35. Figure 5.35a shows two zeros located inside the unit circle and two poles at the origin. Using the geometric interpretation of the frequency response we find that between 0 Hz and $f_s/2$ the total angular change is $\theta_1 + \theta_2 = 360^o$ for the case where both zeros are inside the unit circle. The total angular change for the two poles is $\alpha_1 + \alpha_2 = 360^o$. Since the pole angles subtract from the zero angles, the net change in the phase response for case a is zero. Thus we expect the phase response to start at 0^o and end at 0^o. For case b where the two zeros are outside the unit circle (reciprocal locations of case a) we find that the total angular change for the zeros is $\theta_1 + \theta_2 = 0^o$ but the poles still produce a change of 360^o. The net change in the angular response for case b is -360^o.



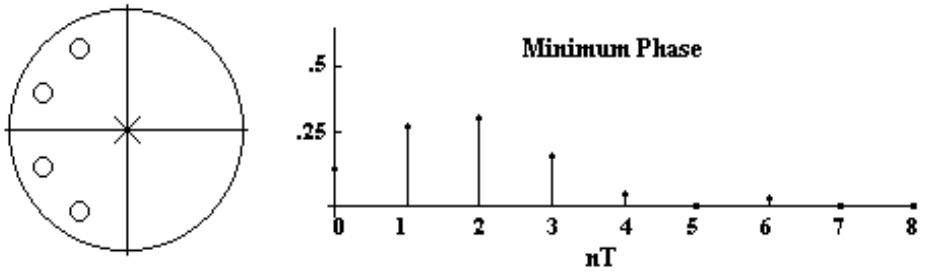
a) b)

Figure 5.35
This figure shows the total angular change produced by two zeros when a) they are both inside the unit circle and b) when they are both outside the unit circle. For case a, $\theta_1 + \theta_2 = 2\pi$ but for case b, $\theta_1 + \theta_2 = 0$. Case a is called minimum phase and case b is called maximum phase.

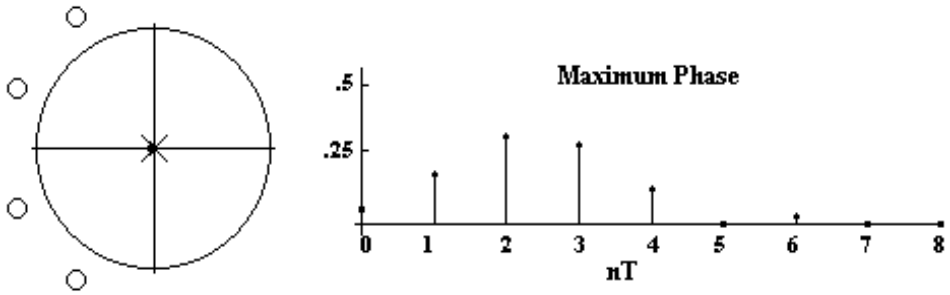
In general, when all of the zeros of a system are located inside the unit circle the system is said to be a *minimum phase system*. When all of the zeros are located outside the unit circle the system is said to be a *maximum phase system*. For those systems which have zeros both inside and outside the unit circle the term *mixed phase system* applies. Zeros located on the unit circle cause a discontinuity in the phase curve. Since zeros on the unit circle have reciprocals which are their conjugate counterparts, these zeros do not play a role in determining whether are not a system is minimum phase or maximum phase.

Since phase change translates into time delay in getting a particular frequency through a processing system, it is often important to design minimum phase systems. One other interesting characteristic of minimum phase systems comes from examining their impulse response characteristics. Figure 5.36 shows two systems which have identical magnitude plots but one of them, Figure 5.36(a), is a minimum phase system while the other is a maximum phase system. Notice that the impulse response of the minimum phase system has its larger terms first. For any minimum phase system we can write the following equation:

$\sum_{n=0}^{m} h^2[n]$ is maximized for all m from 0 to N.



(a)



(b)

Figure 5.36
(a) a minimum phase system and its corresponding impulse response. (b) a maximum phase system and its impulse response. Notice that (a) and (b) represent systems with the same magnitude response.

The maximum is with respect to other systems which have an identical magnitude response. Since $H(z) = H_1(z) \times H_2(z)$ we must convolve $h_1[n]*h_2[n]$ to form h[n]. We can therefore conclude that

$$E = \sum_{n=0}^{m} h^2[n]$$

is maximized when $H(z)$ represents a minimum phase system. E is sometimes referred to as the *partial energy* of the system. The practical result of this partial energy equation comes from the

step responses of minimum phase systems. Since the step response, S(n), of a system can be written in terms of its impulse response as S(n) = h(n) + S(n-1), we see that a system which maximizes the partial energy has the fastest time constant.

5.7 Applications

Moving Average FIR Filter

The average value of a finite sequence is given by

$$avg = \frac{1}{N}\sum_{i=1}^{N} x_i$$

If N is large, this equation is difficult to use since it requires that all values of x_i be summed before an output can be produced. The equation becomes difficult to implement when the value of N is unknown at the outset and there is no good way to establish an upper bound on the size of the sum. A practical alternative to the equation for the average value makes use of a *moving average*. The moving average is the average value of the last N input terms. In equation form this is given by

$$A_M[k] = \frac{1}{N}\sum_{i=k-N+1}^{k} x_i \qquad \textbf{5.16}$$

Example 5.15

Find the moving average sequence for the sequence given by
x = [0,1,1,1,2,2,2,4,1,5,7,4,6,2,3,7,1,...]
Take N = 3.

Solution:

Using equation 5.16 the answer is
x = [0, 1, 1, 1, 2, 2, 2, 4, 1, 5, 7, 4, 6, 2, 3, 7, 1,...]
$A_M[k]$ = [0, 1/3, 2/3, 1, 4/3, 5/3, 2, 8/3, 7/3, 10/3, 13/3, 16/3, 17/3, 4, 11/3, 4, 11/3,..]

Equation 5.16 is easily converted to a transfer function. For N = 3, equation 5.16 becomes

$$A_M[k] = \frac{1}{3}(x[k] + x[k-1] + x[k-2])$$

which has a transfer function given by

$$H_M(z) = \frac{z^2 + z + 1}{3z^2}$$

In general, for a moving average filter of length L

$$H_M(z) = \frac{z^{L-1} + z^{L-2} + \cdots + 1}{Lz^{L-1}} \qquad \textbf{5.17}$$

We will refer to equation 5.17 as the *exact nonrecursive form* for a moving average filter. The transfer function of the moving average filter is that of a low pass FIR filter which has all of its zeros located on the unit circle. Figure 5.37 shows a magnitude plot and pole/zero map for a moving average filter of different orders.

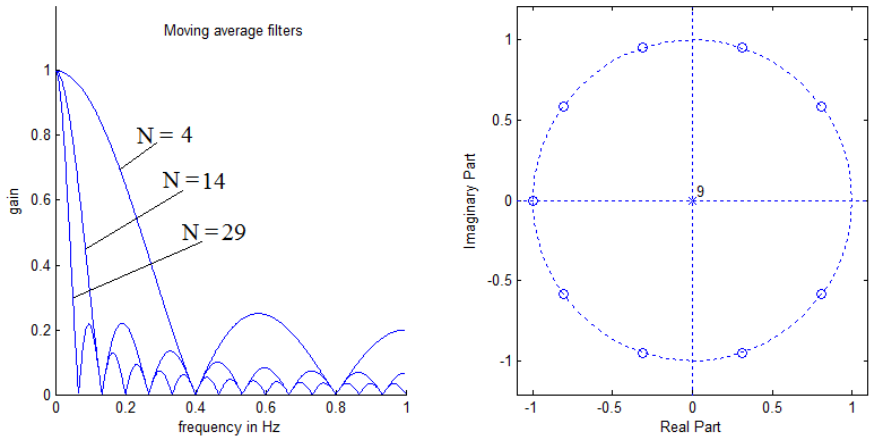


Figure 5.37
Left is a magnitude plot for moving average filters of order 4, 14, and 29. Right is pole/zero plot for a 9th order moving average filter.

An interesting characteristic of the moving average filter is what it becomes if the data is passed through such a filter multiple times. We refer to this as a *multi-pass moving average filter.* For example, noisy data may be passed through a moving average filter of length 3 and the output of that filter will be passed through a second identical moving average filter to create a moving average of the moving average of the data. In the frequency domain the transfer function is the same as the transfer function of the original moving average filter squared. For a length 3 filter for example, this would be

$$mp2Avg = \left(\frac{z^2+z+1}{3z^2}\right)^2 = \frac{z^4+2z^3+3z^2+2z+1}{9z^4}$$

Examination of this transfer function shows that it is identical to that of the triangular window function. If we continue this process we find that a four-pass moving average filter has a frequency response that is very similar to that of an equal length Blackman window function.

Example 5.16
Create a moving average filter of length 8. Use this filter as the basis for a four-pass moving average filter and compare its frequency response with that of a Blackman window function of equal length.

Solution
The MATLAB® code below produces a graph for the magnitude response of 32nd order Blackman window and a 32nd order 4-pass moving average filter. The results are shown in Figure 5.38.

```
fs = 2;                          %Sample frequency
N = 8;
ma1 = ones(1, N)/N;              %The original moving avg filter
ma2 = conv(ma1, ma1);            %Convolve the moving avg filter with
                                 %  itself to get a 2-pass MAF
ma4 = conv(ma2, ma2);            %Convolve the 2-pass with itself for
                                 %  a 4-pass MAF
[Hma4 f] = freqz(ma4, den, 1024, fs);
```

```
figure(1);clf;
Hma4 = abs(Hma4)/max(Hma4);  %normalize and change to db
Hma4DB = 20*log10(Hma4);
plot(f, Hma4DB);
hold on;
wBlk = blackman(length(ma4));%Create a Blackman window
[Hb f] = freqz(wBlk, den, 1024, fs);
Hb = abs(Hb)/max(Hb);
HbDB = 20*log10(Hb);         %Normalize and change to db
plot(f, HbDB, 'r');
```

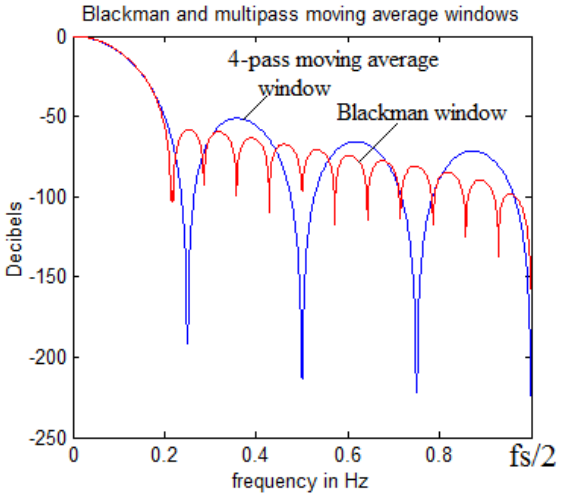


Figure 5.38
This figure compares the magnitude response of a length 32 Blackman window with the magnitude response of a 4-pass moving average window of equal length.

The Blackman window has better stop band performance and even as the length is increased the moving average peak ripple remains higher than the same length Blackman window.

A multi-pass moving average window can approximate the frequency response of a Blackman window. For instances where a Blackman window is applied directly to data in real time, the moving average approximation will require much less computational complexity and may be suitable for small real time applications.

Comb Filters
A comb filter may be created from a moving average filter in two different ways. In the first method, we will add a zero to the transfer function for the FIR moving average filter at $z = +1$. For a filter of order N, this will create $N/2$ equally spaced main lobes that make up a comb filter. In method 2 we will use the existing magnitude plot of a moving average filter and frequency scale it so that the main lobe is duplicated M times in frequency space to make up M "teeth" in the comb filter. This second method results in an IIR filter and is taken up in detail in section 6.7 on IIR Applications.

The transfer function for the FIR moving average filter was given in equation 5.17 as

$$H_M(z) = \frac{z^{L-1} + z^{L-2} + \cdots + 1}{Lz^{L-1}}$$

Adding a zero at z = +1 (and a pole at the origin to stay causal) to this transfer function gives

$$H_M(z) = \frac{z^{L-1} + z^{L-2} + \cdots + 1}{Lz^{L-1}} \cdot \frac{z+1}{z} = \frac{z^L - 1}{Lz^L}$$

Thus the transfer function for a comb filter can be written as

$$H_C(z) = K\frac{z^N - 1}{z^N} \qquad \textbf{5.18}$$

A typical comb filter of order 10 is shown in Figure 5.39. If the filter order is odd there will not be a zero at $z = -1$ and the filter will have a gain of 1 at $f_s/2$. The $N/2$ lobes are centered at kf_s/N, k = 0, 1, … N/2 for N even.

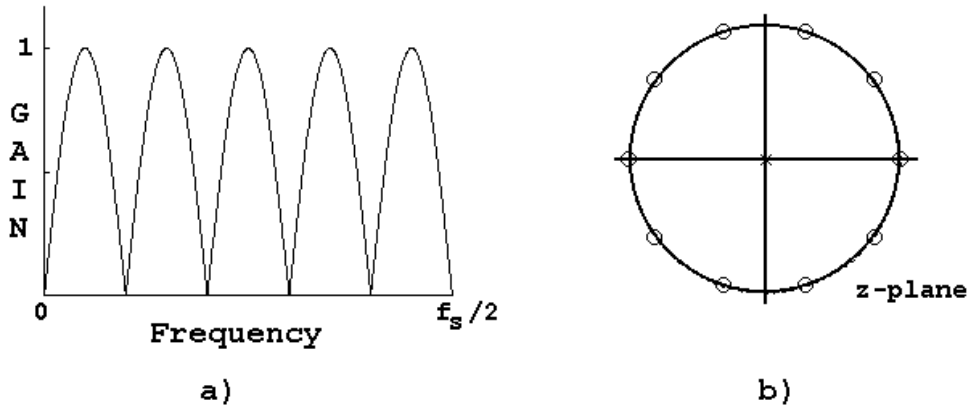


Figure 5.39
A comb filter created by adding a zero to a moving average filter. The order is 10.

The lobes of the comb filter can be sharpened by moving the poles at the origin radially out to the vicinity of the unit circle and spaced between the zeros. This of course, changes the filter to an IIR filter and greatly increases the computational load.

Differentiators

A differentiator takes the time derivative of an incoming signal. Taking the derivative in the time domain corresponds to multiplication by *s* in the Laplace domain. To get the frequency response for a differentiator we replace *s* by $j\omega$ to get

$$H_{diff}(\omega) = j\omega$$

or

$$\left|H_{diff}(\omega)\right| = \omega$$
$$\angle[H_{diff}(\omega)] = \pi/2 \qquad \textbf{5.19}$$

The frequency characteristics of an ideal differentiator are shown in Figure 5.40.

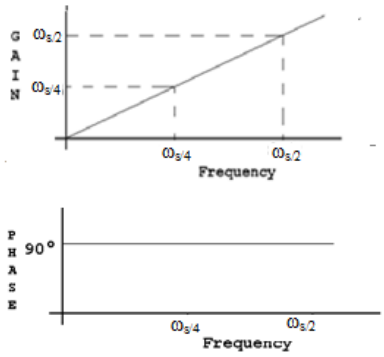


Figure 5.40
Magnitude and phase plot for the ideal differentiator.

As a simple approximation to the derivative we consider using the slope between sample *n* and sample *n*-1 as the value of the derivative at time n. In equation form:
$y = \left.\frac{du}{dt}\right|_{t=nT} \approx \frac{u[n]-u[n-1]}{T}$, or taking the *z*-transform $\frac{y(z)}{u(z)} = \frac{z-1}{Tz}$. Figure 5.41 shows the frequency plot for this transfer function and compares it to the plot for an ideal differentiator. From the figure it is clear that the approximate differentiator works well for very low frequencies and its error increases significantly as the frequency approaches $f_s/2$. This is to be expected since higher frequencies have more signal change between samples. This method has a very light computation load so it could be suitable for say a microcontroller measuring a low frequency signal such as a temperature or moisture sensor.

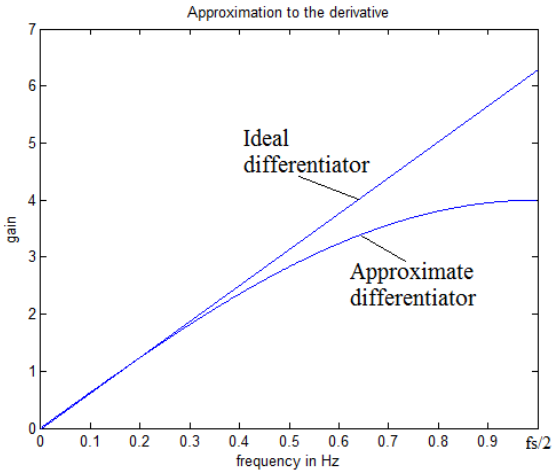


Figure 5.41
The frequency response of an differentiator approximated as the slope of the line from the point at *nT* to the point at (*n* -1)*T*.

To get a better approximation to the derivative we can use the Fourier series method, and if necessary add a window function. MATLAB® does this with the `firls` function. If we add the word '`differentiator`' as an argument to this function, it creates a differentiator which places a

heavier weight on the error at low frequencies. Likewise, the Parks-McClellan method and frequency sampling can be used to create successful approximations to a differentiator.[5]

Example 5.17
Design a digital filter to approximate an ideal differentiator using the Fourier series method. Use a sample frequency to 11,025 Hz.

Solution:
The following MATLAB® code shows how to create a differentiator using the `firls` function.

```
fs = 2;
N = 11;      %Order
F = [0 1];   %Frequency points
M = [0 1];   %Corresponding magnitude points
num = 2*pi*firls(N, F, M, 'differentiator');
den = 1;
[Hdf f] = freqz(num, den, 1024, fs);
plot(f, abs(Hdf));
```

Figure 5.42 shows the results of using `firls` for various order filters. Note that when the filter order is odd this method produces a zero at $f_s/2$.

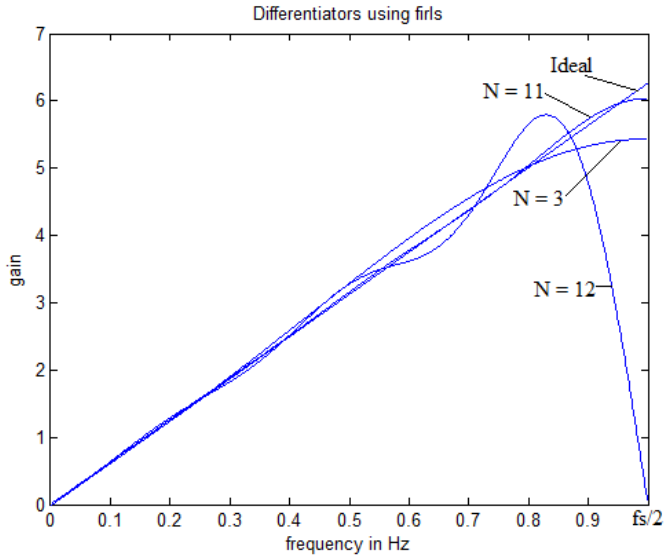


Figure 5.42
Three differentiators designed using the firls function in MATLAB®. The straight line labeled "Ideal" is the frequency response of the ideal differentiator for comparison.

Differentiators find use in many different systems but especially so in digital control systems where they serve as compensation networks and as part of PID controllers.

Example 5.18
In the Fourier series section of Chapter 3, we show that a square wave can be approximated by the series:

$$x(t) = \sum_{\substack{k=1 \\ k\ odd}}^{N} \frac{4}{k\pi} \sin(k\omega_0 t)$$

Create $x(t)$ in MATLAB® with $\omega_0 = 200\pi$, $f_s = 22050$ Hz, and $N = 21$ using 1024 samples. Filter this signal with the 11th order differentiator of Example 5.17 using MATLAB® filter function `firls` and plot the results.

Solution

We begin by creating the differentiator from Example 5.17 with $f_s = 22050$ Hz.

```
fs = 22050;
N = 11;      %Order
F = [0 1];   %Frequency points
M = [0 1];   %Corresponding magnitude points
num = 2*pi*firls(N, F, M, 'differentiator');
den = 1;
```

Next we create the approximate square wave

```
T = 1/fs;
n = 1:1024;
nT = n*T;
w = 200*pi;
sq = zeros(1, length(nT));
for k = 1:2:21
    sq = sq + (4/(k*pi))*sin(k*w*nT);
end
figure(1);clf;
plot(nT, sq);
```

Finally, we use the filter function to differentiate the approximate square wave.

```
sqDiff = filter(num, den, sq);
figure(2);clf;
plot(nT, sqDiff);
```

The results are shown in Figure 5.43.

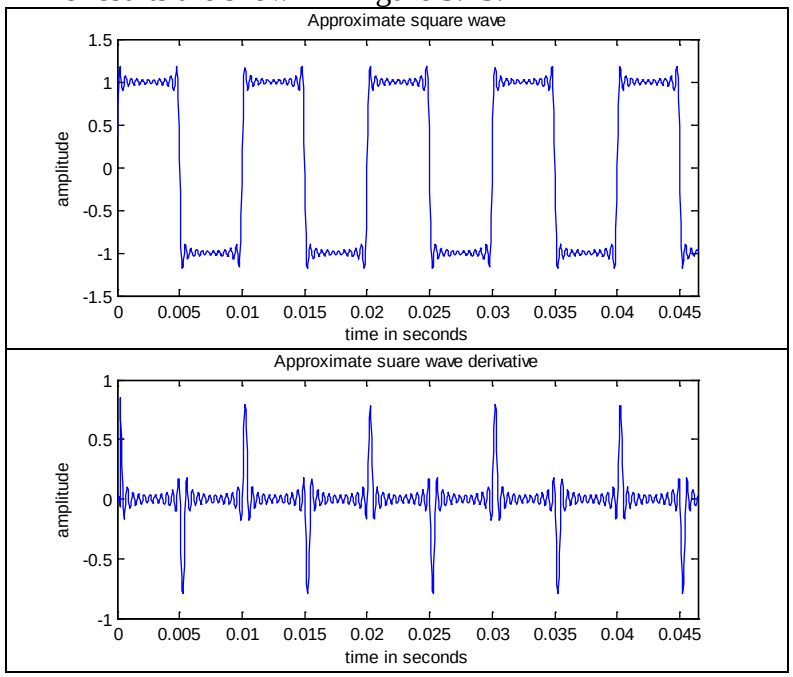


Figure 5.43

Top, a square wave approximated using a truncated Fourier series with 11 terms. Bottom, the result of passing the square wave through an 11th order diffentiator.

Skip Hilbert Transformers

5.8 Summary of FIR Characteristics

IIR filters have feedback while FIR filters do not. This distinction provides unique advantages and disadvantages resulting in design tradeoffs. The more important characteristic differences and design tradeoffs are summarized below:

- *Impulse response* – The impulse response of an FIR filters goes to exactly zero after a finite amount of time regardless of the precision of their implementation.
- *Stability* – FIR filters have no feedback terms and the output can never by unbounded since it must be implemented with finite coefficients.
- *Computational efficiency* – poles located inside by near the unit circle cause the gain to rise in the range of frequencies in their vicinity. Zeros on or near the unit circle cause the gain to fall or go to zero. With FIR filters the gain is pushed up uniformly and selectively lowered by zero placement.
- *Phase linearity* – FIR filters are almost always implemented with linear phase in mind and achieving linear phase with an FIR design is relatively easy. Phase linearity specifications in an application are often the main factor in dictating an FIR filter.
- *Quantization and round- off error* – This is much less of a problem with FIR filters than it is for IIR filters because in FIR filters any error that is introduced cannot be fed back to remain in the system.
- *VLSI implementation* – FIR structures lend themselves more readily to implementation in VLSI structures because the can be constructed with a sequence of delays and multipliers.
- *Design simplicity* – FIR filters tend to slightly more complicated to design than IIR filters. Simple IIR filters such as resonators and the notch filters can be done by hand.

EE 311
FIR Application Examples

Moving Average

```
%FIRMovingAvg1.m
%Plots moving average filters for various orders.
fs = 2;                                %Sample frequency
num = (1/8)*[1 1 1 1 1 1 1 1];
den = 1;
[Hm f] = freqz(num,den, 1024, fs);     %Find the frequency
response
figure(1); clf;
subplot(1,3,1)
plot(f, abs(Hm))                       %Plot the magnitude
axis([0 fs/2 0 1.2]);
title('FIR Moving Average Magnitude');
xlabel('frequency in Hz');
ylabel('gain');
subplot(1,3,2)
hold on;
HmAngle = unwrap(angle(Hm))*180/pi;   %Put the angle in degrees
plot(f, HmAngle)                %plot the phase curve
axis([0 fs/2 -180 180]);
title('FIR Moving Avergage Phase');
xlabel('frequency in Hz');
ylabel('Degrees');
subplot(1, 3, 3);
zplane(num, den);
figure(2);clf;
subplot(1, 2, 1);
hold on;
N = [5 15 30];
for i=1:3
    num = ones(1, N(i))/N(i);
    den = 1;
    [Hmavg f] = freqz(ones(1, N(i))/N(i), den, 1024, 2);
    plot(f, abs(Hmavg));
end
xlabel('frequency in Hz');
ylabel('gain');
title('Moving average filters');
axis([0 fs/2 0 1.2]);
subplot(1, 2, 2);
num = ones(1, 10)/10;
den = 1;
zplane(num, den);
```

Moving average with random noise

```
%FIRMovingAvg2.m
%Plots random noise and filters it with moving avg
%  filter
randSeq = (1:500)/500;
a = 0; b = 10;
randSeq = randSeq.*(a + (b-a).*rand(1, 500));
randSeq = [randSeq (a + (b-a).*rand(1, 500))];
%
L = 20;                          %Length
numFIR = (1/L)*ones(1, L);     %Create an FIR filter of
length L
denFIR = 1;
yFIR = filter(numFIR, denFIR, randSeq); %Filter the
sequence
figure(1);clf;
subplot(1, 2, 1);
f = 1:1000;
plot(f, randSeq);
xlabel('nT');
ylabel('random sequence)');
title('Pseudorandom sequence');
subplot(1, 2, 2);
plot(f, yFIR, 'k');            %Plot the result
axis([0 1000 0 7]);
xlabel('nT');
ylabel('y(nT)');
title('FIR Moving Average Filter');
```

Moving Average and Blackman

```
%FIRMovingAvg4.m
%Compares moving average to blackman
fs = 2;                              %Sample frequency
N = 8;
ma1 = ones(1, N)/N;
den = 1;
ma2 = conv(ma1, ma1);
ma4 = conv(ma2, ma2);
disp(ma4/max(ma4));
[Hma4 f] = freqz(ma4, den, 1024, fs);
figure(1);clf;
subplot(1, 2, 1);
Hma4 = abs(Hma4)/max(Hma4);
plot(f, abs(Hma4));
hold on;
wBlk = blackman(length(ma4));
disp(wBlk');
[Hb f] = freqz(wBlk, den, 1024, fs);
Hb = abs(Hb)/max(Hb);
plot(f, abs(Hb), 'r');
subplot(1, 2, 2);
Hma4DB = 20*log10(Hma4);
plot(f, Hma4DB);
hold on;
HbDB = 20*log10(Hb);
plot(f, HbDB, 'r');
xlabel('frequency in Hz');
ylabel('Decibels');
title('Blackman and multipass moving average windows');
figure(2);clf
subplot(1, 2, 1);
ma4 = ma4/max(ma4);
stem(ma4);
subplot(1, 2, 2);
stem(wBlk);
figure(3);clf;
subplot(1, 2, 1);
zplane(wBlk);
subplot(1, 2, 2);
zplane(ma4);
```

Differentiator

```
%FIRLSDiff.m
%Plots differentiators of various orders
fs = 2;
figure(1);clf;
for N = 3:3:12
    F = [0 1];
    M = [0 1];
    num = 2*pi*firls(N, F, M, 'differentiator');
    den = 1;
    [Hd f] = freqz(num, den, 1024, fs);
    hold on;
    plot(f, abs(Hd));
end
```

Differentiator Application

```
%DiffExmp1.m
%Differentiates a square wave
fs = 22050;
N = 11;      %Order
F = [0 1];   %Frequency points
M = [0 1];   %Corresponding magnitude points
num = 2*pi*firls(N, F, M, 'differentiator');
den = 1;
%Next we create the approximate square wave
T = 1/fs;
n = 1:1024;
nT = n*T;
w = 200*pi;
sq = zeros(1, length(nT));
for k = 1:2:21
    sq = sq + (4/(k*pi))*sin(k*w*nT);
end
figure(1);clf;
plot(nT, sq);
axis([0 1024*T -1.5 1.5]);
xlabel('time in seconds');
ylabel('amplitude');
title('Approximate square wave');
%Finally, we use the filter function to differentiate the
approximate square wave.
sqDiff = filter(num, den, sq);
figure(2);clf;
plot(nT, sqDiff);
axis([0 1024*T -1 1]);
xlabel('time in seconds');
ylabel('amplitude');
title('Approximate square wave derivative');
```