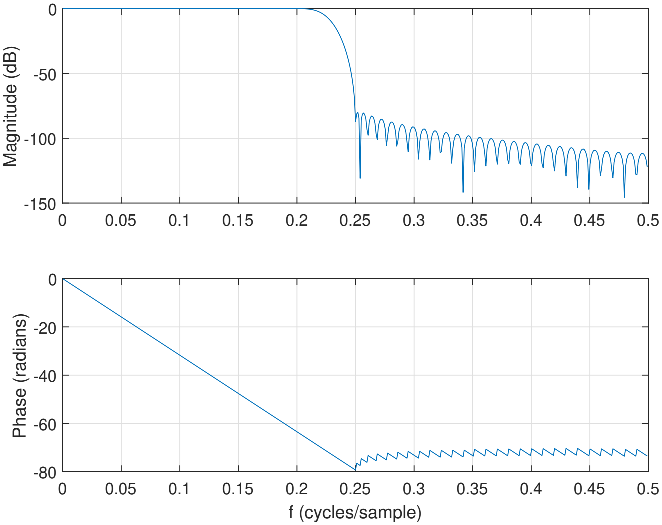
Section 1

Digital Filter Introduction

Subsection 1

Distortionless Filters

What is a Distortionless Filter?

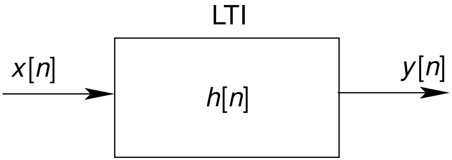


A Distortionless Filter Is

- A filter is distortionless in the passband, if the output signal of the filter has the same shape as the input signal.
- This type of filter can only amplitude scale and/or delay the signal. In terms of the frequency response of the filter, these conditions are
  - the magnitude response must have a constant gain across the passband,
  - the phase response must be linear (or equivalently, the group delay must be constant) in the passband.

Example 1: Linear Phase System

In this example, we want to examine the effect of a filter's phase response on the shape of a signal waveform. The filter of interest is a linear time-invariant (LTI) system, where $h[n]$ is the impulse response of the filter.



- Define the input $x[n]$ as the sum of two or more sinusoids

$$x[n] = \sum_{k=1}^{K} x_k[n]$$

where $x_k[n] = \sin(\omega_k n)$.

Example 1: Linear Phase System

- Assume there are only two sinusoids in the sum,

$$x[n] = x_1[n] + x_2[n] = \sin(\omega_1 n) + \sin(\omega_2 n),$$

  where $\omega_k = 2\pi f_k$, and $f_1 = 1/80$ cycles/sample and f_2 is the next odd harmonic, ie. $f_2 = 3/80$ cycles/sample.
- The output, $y[n]$, can be determined by recalling (see O&S 3rd ed, example 2.15, page 42) that if $h[n]$ is real, the output is

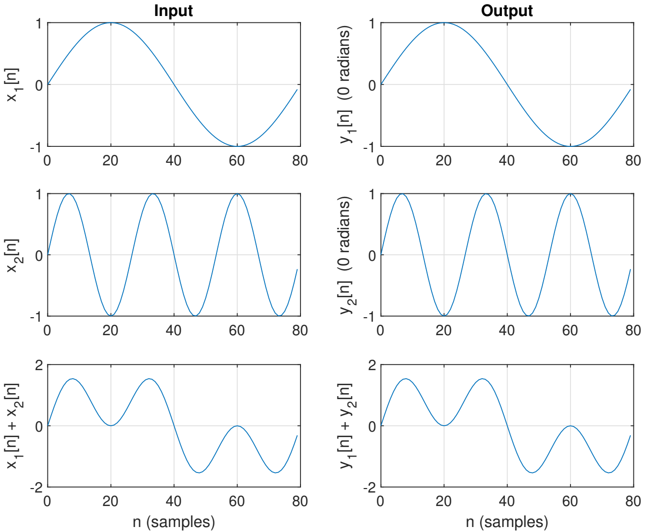
$$y[n] = |H(e^{j\omega_1})| \sin(\omega_1 n + \angle H(e^{j\omega_1}))$$

  if the input is

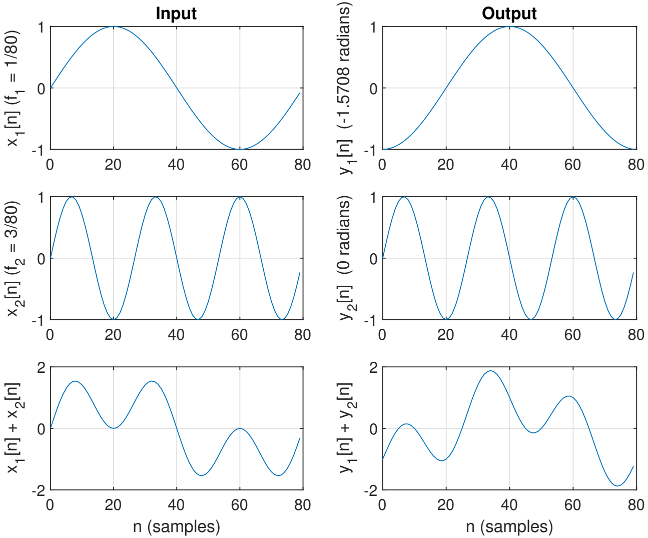
$$x[n] = \sin(\omega_1 n)$$

- Only interested in phase, so set $|H(e^{j\omega_1})| = 1$.
- User defined MATLAB function used: lp_demo(phase,num_sinusoids)

Example 1: Figure Generated with lp_demo([0,0],2)



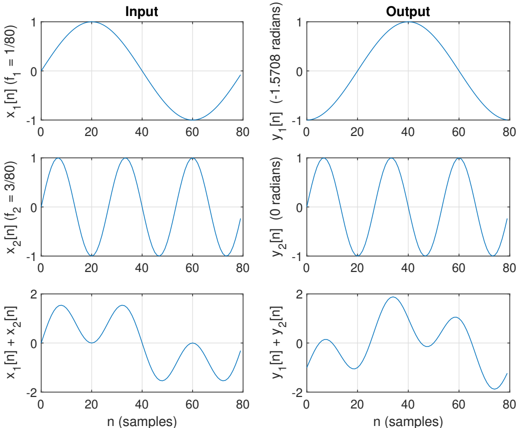
Example 1: Figure Generated with lp_demo([-$\pi/2$,0],2)



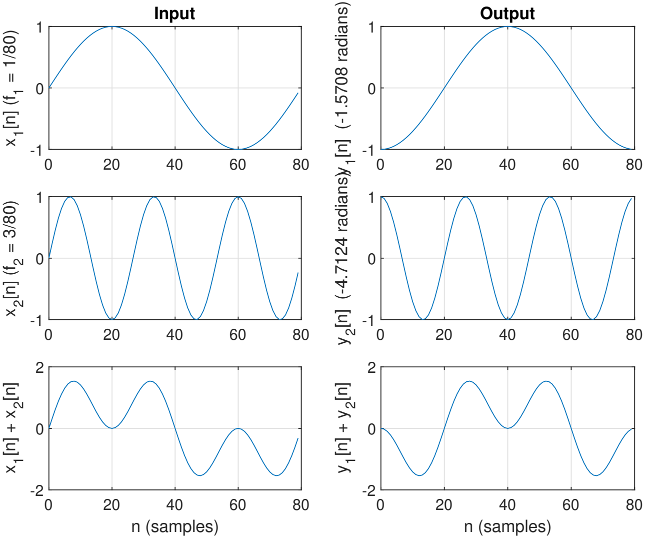
TopHat Question 1: Correct for Output Phase Distortion

What phase shift is needed for $x_2[n]$ to make the third row output waveform the same shape as the input waveform?

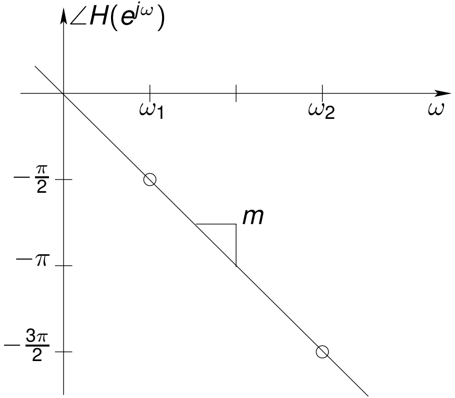
A $-\frac{\pi}{2}$, B $-\pi$, C $-\frac{3\pi}{2}$, D $\frac{3\pi}{2}$, E $\frac{\pi}{2}$



Example 1: Figure Generated with lp_demo([-$\pi/2$,-3$\pi/2$],2)



Example 1: Plotting the Phase Response



What is the group delay, τ_g, for the linear phase response?

Example 1: Constant Group Delay in the Passband

- In the previous slide, the group delay was calculated as constant value of 20 samples.
- A constant group delay is a pure delay of signals in the passband.
- For this example, the phase shifts of each sinusoid in the passband for a pure delay of 20 samples can be calculated as

$$\begin{aligned} y[n] &= x[n-20] \\ &= x_1[n-20] + x_2[n-20] \\ &= \sin(\omega_1(n-20)) + \sin(\omega_2(n-20)) \\ &= \sin(2\pi n/80 - \pi/2) + \sin(2\pi n3/80 - 3\pi/2) \end{aligned}$$

- This demonstrates that a 20 sample delay of the input signal requires $y_1[n]$ to have a phase shift of $-\pi/2$, and $y_2[n]$ to have a phase shift of $-3\pi/2$.

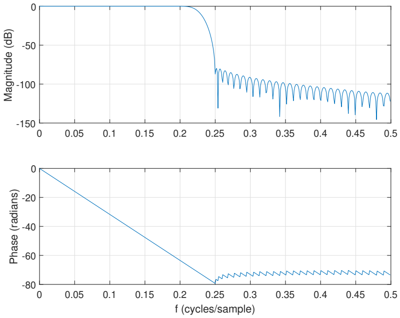
TopHat Question 2: Signal Delay Given Phase Response

The input to the filter defined in the figure is

$$x[n] = \sin(0.01\pi n) + \sin(0.03\pi n)$$

How many samples is the signal at the output of the filter delayed relative to the input (make sure the units are correct).

$$\text{A } 160, \quad \text{B } 160/\pi, \quad \text{C } 320, \quad \text{C } 320/\pi$$



TopHat Question 3: Signal Delay Given Input-Output Relationship

The passband input to a linear phase lowpass filter is

$$x[n] = \sin(0.01\pi n) + \sin(0.04\pi n + 0.1),$$

and the output is

$$y[n] = y_1[n] + \sin(0.04\pi n - 0.3),$$

What is the slope of the linear phase response?

$$\text{A } -10, \quad \text{B } -10/\pi, \quad \text{C } -30, \quad \text{D } -30/\pi$$

Example 1: Frequency Response

- Recall, for this example, $y[n] = x[n-20]$, and taking the z-transform gives

$$Y(x) = X(z)z^{-20}$$

- Rearrange to generate the transfer function

$$H(z) = \frac{Y(z)}{X(z)} = z^{-20}$$

- The frequency response is determined by substituting $z = e^{j\omega}$:

$$H(e^{j\omega}) = |H(e^{j\omega})|e^{j\theta_M(\omega)} = e^{-j20\omega}$$

- This can also be put in the form

$$H(e^{j\omega}) = e^{-j\tau_g\omega}$$

  where, for a linear phase response, τ_g is a constant, which is the negative of the slope of the phase response. τ_g is referred to as the group delay.

Example 1: Linear Phase System Summary

- A filter that has a linear phase response and constant gain results in a pure delay of the input signal. In this example, $y[n] = x[n-20]$.
- If the gain is one, the frequency response of a linear phase filter is

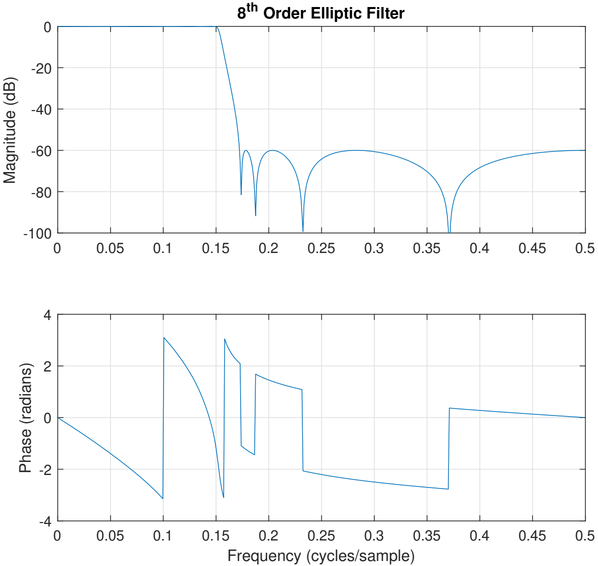
$$H(e^{j\omega}) = e^{-j\tau_g\omega}$$

  where τ_g is the negative of the slope of the phase response and is referred to as the group delay.
- In this example, $\tau_g = 20$ samples.
- Linear phase is important for applications that depend on the shape of the signal, such as communications systems.
- A linear phase response is relatively easy to interpret, but you probably can imagine that a non-linear phase response is more difficult to interpret. This will be explored in the next section.

Subsection 2

Generating a Continuous Phase Response

Frequency Response of an 8th Order Elliptic Filter



Pole Zero Plot for the 8th Order Elliptic Filter

Causes of Phase Response Discontinuities

- There are two types of discontinuities, one results in a jump of π radians and the other a jump of 2π radians.
- The π discontinuity results from the magnitude response being defined as non-negative, and it can be eliminated by using the amplitude response, $A(\omega)$, which can take on negative values, instead of the magnitude response, $|H(e^{j\omega})|$, which only can have non-negative values.
- The 2π discontinuity results from wrapping the phase, which is commonly done by software tools, such as MATLAB, to keep the phase in a range of $-\pi$ to π, and it can be eliminated by simply unwrapping the phase (in MATLAB the command is `unwrap`).
- The frequency response, specified in terms of the amplitude response and the continuous phase response is

$$H(e^{j\omega}) = A(\omega)e^{j\theta_A(\omega)}$$

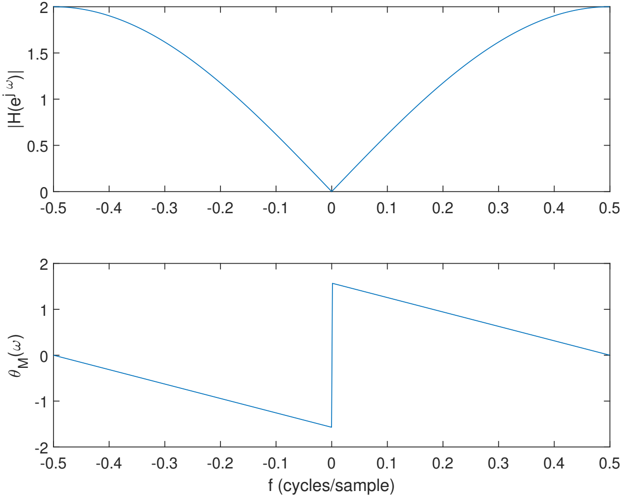
Example 2: System with a π Discontinuity

Given the system, $y[n] = x[n] - x[n-1]$, plot the linear magnitude response, $|H(e^{j\omega})|$, and phase response, $\theta_M(\omega)$, using MATLAB.

- **Solution**: Using the impulse response, $h[n] = \delta[n] - \delta[n-1]$, the MATLAB code that will generate a plot of the frequency response is:

```
h=[1 -1]; % system impulse response
[H,w]=freqz(h,1,[-pi:.01:pi]);
subplot(2,1,1)
plot(w/(2*pi),abs(H))
ylabel('|H(e^{j \omega})|')
subplot(2,1,2)
plot(w/(2*pi), angle(H))
ylabel('\theta_M(\omega)')
xlabel('f (cycles/sample)')
```

Example 2: System with a π Discontinuity



Example 2: System with a π Discontinuity

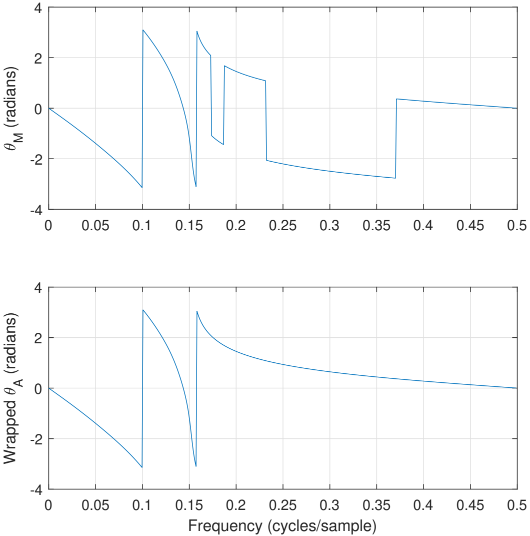
Given the system, $y[n] = x[n] - x[n-1]$, generate expressions for the two forms of the frequency response:

$$H(e^{j\omega}) = |H(e^{j\omega})|e^{j\theta_M(\omega)}$$

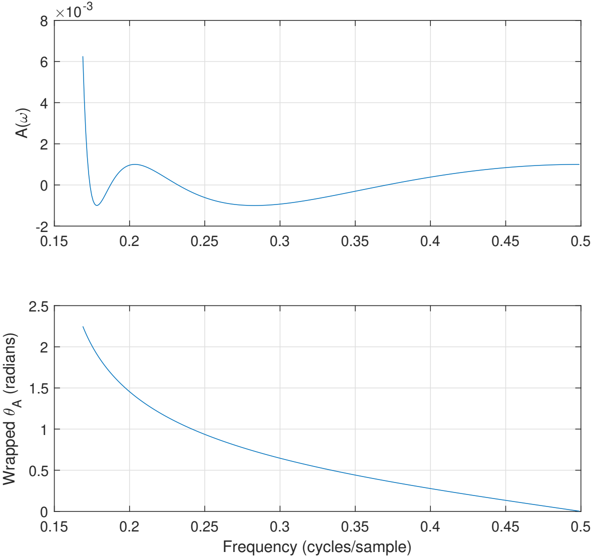
and

$$H(e^{j\omega}) = A(\omega)e^{j\theta_A(\omega)}$$

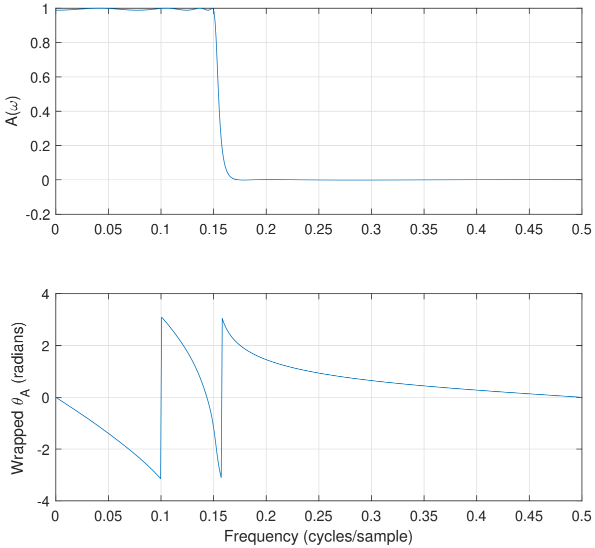
Phase Response with π Shifts Removed



Zoomed Amplitude and Phase Responses with π Shifts Removed



Amplitude and Phase Responses with π Shifts Removed



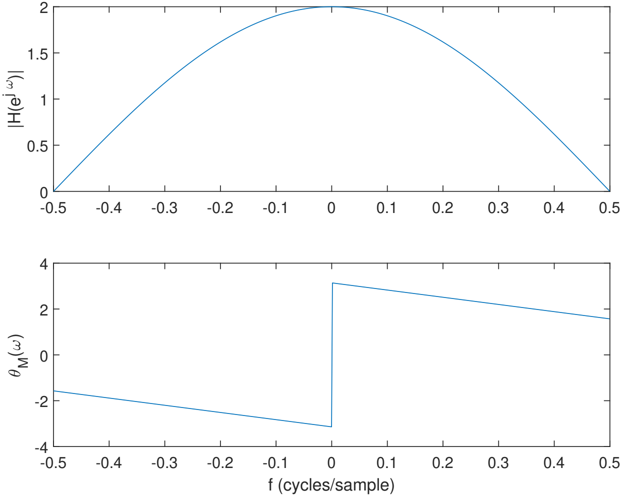
Example 3: System with a 2π Discontinuity

Given the system, $y[n] = -x[n] - x[n-1]$, plot the linear magnitude response, $|H(e^{j\omega})|$, and phase response, $\theta_M(\omega)$, using MATLAB.

- **Solution**: Using the impulse response, $h[n] = -\delta[n] - \delta[n-1]$, the MATLAB code that will generate a plot of the frequency response is:

```
h=[-1 -1]; % system impulse response
[H,w]=freqz(h,1,[-pi:.01:pi]);
subplot(2,1,1)
plot(w/(2*pi),abs(H))
ylabel('|H(e^{j \omega})|')
subplot(2,1,2)
plot(w/(2*pi), angle(H))
ylabel('\theta_M(\omega)')
xlabel('f (cycles/sample)')
```

Example 3: System with a 2π Discontinuity



Example 3: System with a 2π Discontinuity

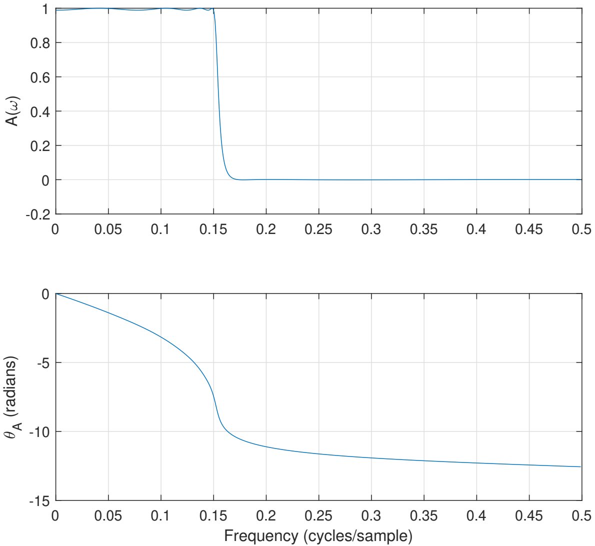
Given the system, $y[n] = -x[n] - x[n-1]$, generate expressions for the two forms of the frequency response:

$$H(e^{j\omega}) = |H(e^{j\omega})|e^{j\theta_M(\omega)}$$

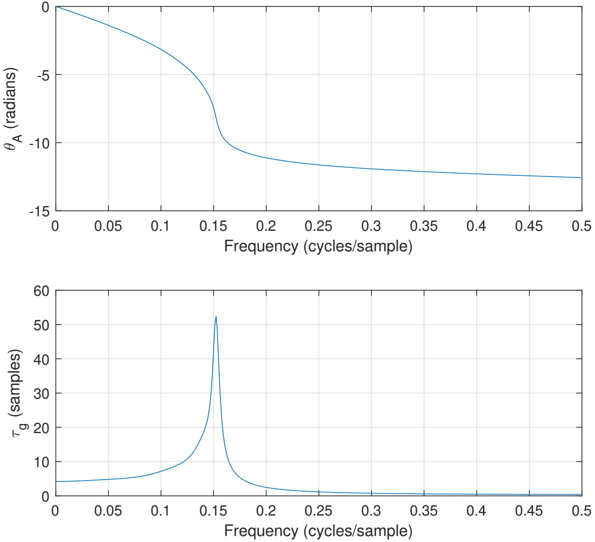
and

$$H(e^{j\omega}) = A(\omega)e^{j\theta_A(\omega)}$$

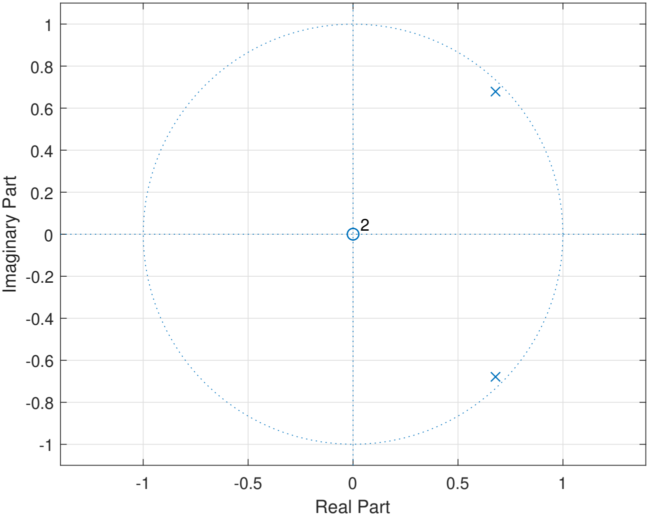
Continuous Phase Response



Group Delay of the 8th Order Elliptic Filter



Estimating Group Delay from a Pole-Zero Plot



Calculating the Amplitude Response

The DTFT of the $h[n]$ is

$$H(e^{j\omega}) = A(\omega)e^{j\theta_A(\omega)}.$$

Multiplying both sides by $e^{-j\theta_A(\omega)}$ gives

$$A(\omega) = H(e^{j\omega})e^{-j\theta_A(\omega)}.$$

Thus if $H(e^{j\omega})$ could be determined using MATLAB, $A(\omega)$ could then be calculated.
$H(e^{j\omega})$ could be evaluated at equally spaced frequencies around the unit circle using the DFT (Discrete Fourier Transform).

$$H(e^{j\omega})\Big|_{\omega=2\pi k/N} = \underbrace{\sum_{n=0}^{N-1} h[n]e^{-j\omega n}}_{\text{DTFT}}\Bigg|_{\omega=2\pi k/N} = \underbrace{\sum_{n=0}^{N-1} h[n]e^{-j2\pi kn/N}}_{\text{DFT}}$$

Note, the DFT is typically represented as

$$H[k] = \sum_{n=0}^{N-1} h[n]e^{-j2\pi kn/N}$$

Calculating the Amplitude Response

If $h[n]$ has a low number of elements (ie. N is small), then $H[k]$ will not have enough elements to produce a good representation of $H(e^{j\omega})$ in a plot. Thus, typically, $h[n]$ is zero padded to obtain a better plot for $H(e^{j\omega})$. Define the length, L, zero padded signal as

$$h_z[n] = \begin{cases} h[n]; & 0 \leq n \leq N-1 \\ 0; & N \leq n \leq L-1 \end{cases}$$

Thus

$$H_z(e^{j\omega})\Big|_{\omega=2\pi k/L} = \sum_{n=0}^{L-1} h_z[n] e^{-j2\pi kn/L} = \sum_{n=0}^{N-1} h_z[n] e^{-j2\pi kn/L}$$

Note that in MATLAB, the DFT is implemented as the fast Fourier transform, fft.
Using the the zero padded sequence, the amplitude response is

$$A(\omega)\big|_{\omega=2\pi k/L} = H_z(e^{j\omega})e^{-j\theta_A(\omega)}\Big|_{\omega=2\pi k/L} = \underbrace{H_z[k]}_{\text{DFT}}\, e^{-j\theta_A(\omega)}\Bigg|_{\omega=2\pi k/L}$$

Note that the fft calculates the response for $k = 0$ to $L-1$, which is from 0 to $2\pi(L-1)/L$ in radians/sample. Or from 0 to $(L-1)/L$ in cycles/sample. This is one period of the response.

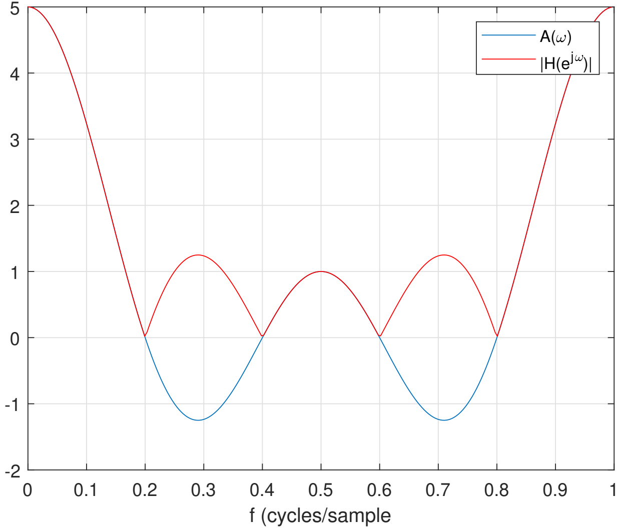
Example 4: Use MATLAB to Calculate $A(\omega)$

Calculate and plot $A(\omega)$ for the system with impulse response $h[n] = [1, 1, 1, 1, 1]$.

- **Solution:** Choose $L = 256$, a power of 2, for the length of the zero padded sequence. $h[n]$ is even symmetric, with a constant group delay of $\tau_g = M/2$, where $M = 4$ is the order of the filter (one less than the length of the filter). The phase response is $\theta_A(\omega) = -\tau_g\omega = -\frac{M}{2}\omega = -2\omega$. The MATLAB solution is

```
h=ones(1,5); L=256; hz=[h, zeros(1,L-length(h))];
Hz=fft(hz); k=0:L-1;
theta_A=-2*2*pi*k/L; W=exp(-j*theta_A);
A=Hz.*W;
A=real(A); %imaginary part is very close to zero
f=k/L; % cycles/sample
clf, plot(f,A); hold
plot(f, abs(Hz),'r')
legend('A(\omega)','|H(e^{j\omega})|')
xlabel('f (cycles/sample'), grid
```

Example 4: Use MATLAB to Calculate $A(\omega)$



Summary: Continuous Phase Response

- There are two types of discontinuities, one results in a jump of π radians and the other a jump of 2π radians.
- The two causes of discontinuities in a phase response are:
  1. The magnitude response is defined as non-negative.
  2. The phase response is typically defined for the range $-\pi$ to π (thus tools, such as MATLAB, wrap the phase).
- These discontinuities can be eliminated by using the amplitude, $A(\omega)$, and associated continuous phase response, $\theta_A(\omega)$, instead of the magnitude, $|H(e^{j\omega})|$, and associated phase response (which is usually discontinuous), $\theta_M(\omega)$, as given in the following frequency response expressions:

$$\begin{aligned} H(e^{j\omega}) &= |H(e^{j\omega})| e^{j\theta_M(\omega)} \\ &= A(\omega) e^{j\theta_A(\omega)} \end{aligned}$$

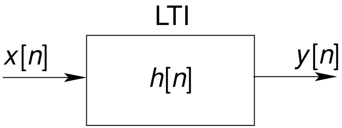
- The difference between these two is that the magnitude, $|H(e^{j\omega})|$, is non-negative and the amplitude, $A(\omega)$, can be negative.

Subsection 3

Types of Linear Phase Systems

Linear Phase Systems

The frequency response of the output of this system is $Y(e^{j\omega}) = H(e^{j\omega})X(e^{j\omega})$.



Using the continuous phase notation,

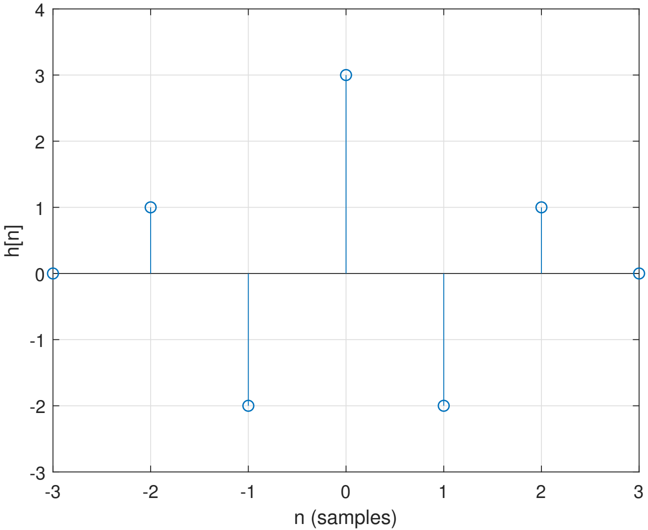
$$\begin{aligned} Y(e^{j\omega}) &= A_Y(\omega)e^{j\theta_Y(\omega)} \\ H(e^{j\omega}) &= A(\omega)e^{j\theta_A(\omega)} \\ X(e^{j\omega}) &= A_X(\omega)e^{j\theta_X(\omega)} \end{aligned}$$

Thus

$$\begin{aligned} A_Y(\omega) &= A(\omega)A_X(\omega) \\ \theta_Y(\omega) &= \theta_A(\omega) + \theta_X(\omega) \end{aligned}$$

Focusing on the phase, if $\theta_A(\omega)$ is nonzero, it will distort the input signal. Thus, ideally a zero phase is desired, $\theta_A(\omega) = 0$. But, practically, this is not possible, as explored in the next slides.

System with a Zero Phase Response



System with a Zero Phase Response

- The impulse response of the system defined in the previous figure is

$$h[n] = \delta[n+2] - 2\delta[n+1] + 3\delta[n] - 2\delta[n-1] + \delta[n-2]$$

- $h[n]$ is symmetrical about $n = 0$ and thus it is an even function.
- An even function is defined as

$$h[n] = h[-n]$$

- Recall that the Fourier transform of an even function is real, thus there is no imaginary term in the frequency response, resulting in the phase response being 0.
- This can be demonstrated by evaluating the frequency response

$$H(e^{j\omega}) = 3 - 4\cos(\omega) + 0.5\cos(2\omega) = A(\omega)e^{j\theta_A(\omega)}$$

- Thus $\theta_A(\omega) = 0$

Example 5: Zero Phase System

Derive the frequency response of

$$h[n] = \delta[n+2] + \delta[n+1] + \delta[n] + \delta[n-1] + \delta[n-2]$$

Plot $h[n]$ and use the DTFT to determine the frequency response, $H(e^{j\omega}) = A(\omega)e^{j\theta_A}$.

TopHat Question 4: Group Delay of System $h[n]$

Given a system with an impulse response,

$$h[n] = \delta[n+2] + \delta[n-2],$$

what is the associated group delay for this system?

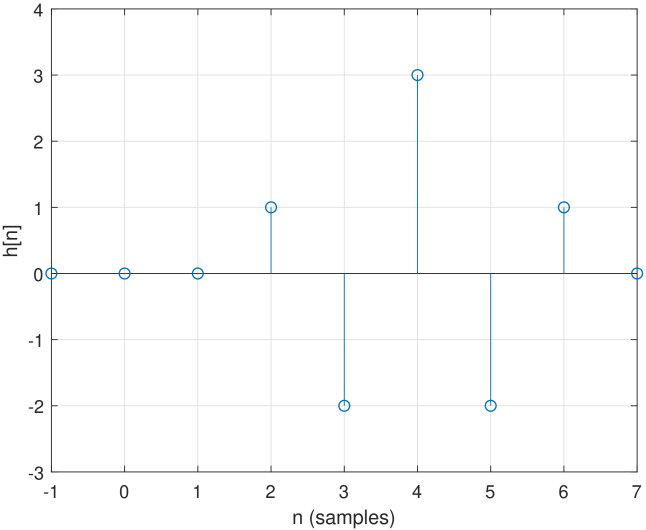
A $\tau_g = 0$

B $\tau_g = 1/2$

C $\tau_g = 1$

D $\tau_g = 2$

Even Symmetric Impulse Response



Even Symmetric Impulse Response

- The impulse response of the system defined in the previous figure is

$$h[n] = \delta[n-2] - 2\delta[n-3] + 3\delta[n-4] - 2\delta[n-5] + \delta[n-6]$$

- $h[n]$ is even about $n = 4$ and thus it is referred to as an even symmetric function, which is defined as

$$h[n] = h[M-n]$$

  where M is the order of the system ($M = N - 1$, where N is the length of $h[n]$). Here M has a value 8.
- An even symmetric $h[n]$ has a linear phase response and an even amplitude response:

$$\theta_A(\omega) = -\frac{M}{2}\omega \text{ and } A(\omega) = A(-\omega)$$

- This can be demonstrated by evaluating the frequency response,

$$H(e^{j\omega}) = (3 + 4\cos(\omega) + 0.5\cos(2\omega))e^{-j4\omega} = A(\omega)e^{j\theta_A(\omega)}$$

- Thus $\theta_A(\omega) = -4\omega$ and $A(\omega) = 3 + 4\cos(\omega) + 0.5\cos(2\omega)$

Example 6: Even Symmetric System

Determine the frequency response of $h[n] = \delta[n] + \delta[n-1] + \delta[n-2] + \delta[n-3] + \delta[n-4]$.

Example 7: Even Symmetric Impulse Response with Odd Order

Given a system $h[n] = \delta[n-2] + \delta[n-3] + \delta[n-4] + \delta[n-5]$, determine

1. what value is $h[n]$ is even symmetric about,
2. the system frequency response in the form of an amplitude response, $A(\omega)$ and phase response, $\theta_A(\omega)$.

TopHat Question 5: Group Delay of a System with an Even Symmetric $h[n]$

Given a system with an impulse response,

$$h[n] = \delta[n] + \delta[n-1],$$

what is the associated group delay for this system?

A $\tau_g = 0$

B $\tau_g = 1/2$

C $\tau_g = -1/2$

D $\tau_g = 1$

E $\tau_g = -1$

Summary: Even Symmetric $h[n]$

A system that has an impulse response that is even symmetric about $M/2$ (M could be even or odd), which is defined by the relationship

$$h[n] = h[M - n],$$

is a linear phase system with a phase response given by

$$\theta_A = -\frac{M}{2}\omega,$$

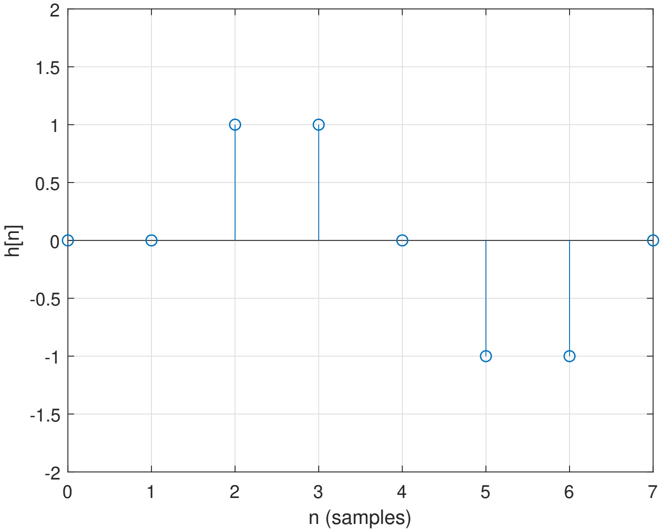
and group delay

$$\tau_g = \frac{M}{2}.$$

The amplitude response for the $h[n]$ is an even function,

$$A(\omega) = A(-\omega).$$

Odd Symmetric Impulse Response



Odd Symmetric Impulse Response

- The impulse response of the system defined in the previous figure is

$$h[n] = \delta[n-2] + \delta[n-3] - \delta[n-5] - \delta[n-6]$$

- $h[n]$ is an odd about $n = 4$ and thus it is referred to as an odd symmetric function, which is defined as

$$h[n] = -h[M-n]$$

  where M is the order of the system ($M = N - 1$, where N is the length of $h[n]$). M is even with value 8.
- An odd symmetric $h[n]$ has a **generalized** linear phase response and an odd amplitude response:

$$\theta_A(\omega) = \frac{\pi}{2} - \frac{M}{2}\omega \text{ and } A(\omega) = -A(-\omega)$$

- This can be demonstrated by evaluating the frequency response,

$$H(e^{j\omega}) = (2sin(2\omega) + 2sin(\omega))e^{j(\pi/2-4\omega)}$$

- Thus $\theta_A(\omega) = \pi/2 - 4\omega$ and $A(\omega) = 2sin(2\omega) + 2sin(\omega)$

Example: Odd Symmetric Impulse Response with Odd Order

Given a system $h[n] = \delta[n] + \delta[n-1] - \delta[n-2] - \delta[n-3]$, determine the frequency response.

Summary: Odd Symmetric $h[n]$ (Generalized Linear Phase)

A system that has an impulse response that is odd symmetric about $M/2$ (M could be even or odd), which is defined by the relationship

$$h[n] = -h[M-n],$$

is referred to as a generalized linear phase system with a phase response given by

$$\theta_A(\omega) = \frac{\pi}{2} - \frac{M}{2}\omega,$$

and group delay

$$\tau_g = \frac{M}{2}.$$

The amplitude response for the $h[n]$ is an odd function,

$$A(\omega) = -A(-\omega).$$

Section 2

Designing FIR Filters

Subsection 1

FIR Filter Introduction

FIR Filter Characteristics

FIR filters have a number of desirable characteristics when compared to IIR filter,

- FIR filters have linear phase or generalized linear phase, and thus they do not distort the input signal shape,
- FIR filters are always stable,
- There are a number of versatile methods for designing FIR filters.

On the other hand,

- FIR filters are higher order and thus have a more complex implementation compared to IIR, thus if a linear phase response is not required, IIR filters are more efficient.
- The group delay for FIR filters can be large, resulting in a large delay between input and output.

FIR Filter Types

Practical FIR filters are designed to have linear phase and there are four types of linear phase filters:

$$H(e^{j\omega}) = A(\omega)e^{j(\beta - \frac{M}{2}\omega)}$$

Type	**1**	**2**	**3**	**4**
Order (M)	even	odd	even	odd
$h[n]$	even symmetric	even symmetric	odd symmetric	odd symmetric
$A(\omega)$	even (about $\omega = 0$)	even	odd (about $\omega = 0$)	odd
$A(\omega) period$	2π	4π	2π	4π
β	0	0	$\pi/2$	$\pi/2$
$H(e^{j0})$	arbitrary	arbitrary	0	0
$H(e^{j\pi})$	arbitrary	0	0	arbitrary
Common Uses	LP, HP, BP, BS, Mulitband	LP, BP	Differentiator, Hilbert Transform	Differentiator, Hilbert Transform

Note: Type 3 could be used for bandpass filters and Type 4 could be used for highpass and bandpass, but they are not commonly used for these types of filters, since they have generalized linear phase, not linear phase.

Type 1 Filter Frequency Response

Type 1 filters have even order, M, phase term $\beta = 0$, and a even symmetric impulse response

$$h[n] = h[M-n]$$

The frequency response is derived using the DTFT

$$\begin{aligned} H(e^{j\omega}) &= \sum_{n=0}^{M} h[n]e^{-j\omega n} \\ &= e^{-j\omega M/2} \sum_{n=0}^{M} h[n]e^{j\omega(M/2-n)} \\ &= e^{-j\omega M/2} \left[h[\tfrac{M}{2}] + \sum_{n=0}^{M/2-1} h[n]e^{j\omega(M/2-n)} + \sum_{n=M/2+1}^{M} h[n]e^{j\omega(M/2-n)} \right] \end{aligned}$$

Take the third term in the above equation and substitute $l = M - n$. Replace n for l. Substitute $h[n] = h[M-n]$. Replace the third term with the new expression, to give,

$$H(e^{j\omega}) = e^{-j\omega M/2} \left[h[\frac{M}{2}] + \sum_{n=0}^{M/2-1} h[n]e^{j\omega(M/2-n)} + \sum_{n=0}^{M/2-1} h[n]e^{-j\omega(M/2-n)} \right]$$

Type 1 Filter Frequency Response

Using Euler's and substituting $l = M/2 - n$

$$H(e^{j\omega}) = e^{-j\omega M/2}\left[h[\frac{M}{2}] + 2\sum_{n=M/2}^{1} h[M/2 - l]\cos(\omega l)\right]$$

This can be put in the form

$$H(e^{j\omega}) = e^{-j\omega M/2}\sum_{k=0}^{M/2} a[k]\cos(\omega k)$$

where

$$a[k] = \begin{cases} h[M/2]; & k = 0 \\ 2h[M/2 - k]; & k = 1, 2, \cdots M/2 \end{cases}$$

Type 1 Filter Frequency Response

- The frequency response of a type 1 FIR filter is

$$A(\omega) = \sum_{k=0}^{M/2} a[k] \cos(\omega k) \quad \text{and} \quad \theta_A(\omega) = -\omega \frac{M}{2}$$

where

$$a[k] = \begin{cases} h[M/2]; & k = 0 \\ 2h[M/2 - k]; & k = 1, 2, \cdots M/2 \end{cases}$$

- $A(\omega)$ is even about $\omega = 0$ and $\omega = \pi$, since the cosine terms are even at these ω locations.
- $A(\omega)$ is also periodic with a period of 2π, since the fundamental period of the cosine terms is 2π.

Example 8: Type 1 System

Given

$$H(z) = 1 + 0.4z^{-1} - 0.4z^{-2} + 1.2z^{-3} - 0.4z^{-4} + 0.4z^{-5} + z^{-6},$$

is this filter a Type 1 filter, and if yes, what is the frequency response?

Type 2 Filter Frequency Response

Type 2 filters have odd order. They have a symmetric impulse response,

$$h[n] = h[M - n],$$

and thus the phase term $\beta = 0$. The frequency response can derived using an approach similar to the Type 1 derivation, to give

$$H(e^{j\omega}) = e^{-j\omega M/2} \sum_{k=1}^{(M+1)/2} b[k] \cos(\omega(k - 1/2)$$

where

$$b[k] = 2h[(M+1)/2 - k]; \quad k = 1, 2, \cdots (M+1)/2$$

Type 2 Filter Frequency Response

Using the results of the previous slide, the frequency response of a Type 2 FIR filter is

$$A(\omega) = \sum_{k=1}^{(M+1)/2} b[k] \cos(\omega(k - 1/2))$$

$$\theta_A(\omega) = -\omega \frac{M}{2}$$

where

$$b[k] = 2h[(M+1)/2 - k]; \;\; k = 1, 2, \cdots (M+1)/2$$

- $A(\omega)$ is even about $\omega = 0$ and odd about $\omega = \pi$, since the $\cos(\omega(k - 1/2))$ terms are even about $\omega = 0$ and they are odd about $\omega = \pi$.
- $A(\omega)$ is also periodic with a period of 4π, since the fundamental period of $\cos(\omega/2)$ $(k = 1)$ is 4π.

Type 3 Filter Frequency Response

Type 3 filters have even order. They have a odd symmetric impulse response,

$$h[n] = -h[M-n],$$

and thus the phase term $\beta = \pi/2$. The frequency response can derived using an approach similar to the Type 1 derivation, to give

$$H(e^{j\omega}) = e^{\pi/2 - j\omega M/2} \sum_{k=1}^{M/2} c[k]\,\sin(\omega k)$$

where

$$c[k] = 2h[M/2-k]; \quad k = 1, 2, \cdots M/2$$

Type 3 Filter Frequency Response

Using the results of the previous slide, the frequency response of a Type 3 FIR filter is

$$A(\omega) = \sum_{k=1}^{M/2} c[k] \sin(\omega k)$$

$$\theta_A(\omega) = \frac{\pi}{2} - \frac{M}{2}\omega$$

where

$$c[k] = 2h[M/2 - k]; \quad k = 1, 2, \cdots M/2$$

- $A(\omega)$ is odd about $\omega = 0$ and $\omega = \pi$, since $\sin(\omega k)$ is odd about these ω locations.
- $A(\omega)$ is also periodic with a period of 2π, since the fundamental period of $\sin(\omega k)$ $(k = 1)$ is 2π.

Type 4 Filter Frequency Response

Type 4 filters have odd order. They have a odd symmetric impulse response,

$$h[n] = -h[M-n],$$

and thus the phase term $\beta = \pi/2$. The frequency response can derived using an approach similar to the Type 1 derivation, to give

$$H(e^{j\omega}) = e^{j(\pi/2-\omega M/2)} \sum_{k=1}^{(M+1)/2} d[k] \sin(\omega(k-1/2))$$

where

$$d[k] = 2h[(M+1)/2-k]; \;\; k = 1, 2, \cdots (M+1)/2$$

Type 4 Filter Frequency Response

Using the results of the previous slide, the frequency response of a Type 4 FIR filter is

$$\begin{aligned} A(\omega) &= \sum_{k=1}^{(M+1)/2} d[k] \sin(\omega(k - 1/2)) \\ \theta_A(\omega) &= \frac{pi}{2} - \omega \frac{M}{2} \end{aligned}$$

where

$$d[k] = 2h[(M+1)/2 - k]; \;\; k = 1, 2, \cdots (M+1)/2$$

- $A(\omega)$ is odd about $\omega = 0$ and even about $\omega = \pi$, since the $\sin(\omega(k - 1/2))$ terms are even about $\omega = 0$ and they are odd about $\omega = \pi$.
- $A(\omega)$ is also periodic with a period of 4π, since the fundamental period of $\sin(\omega/2)$ $(k = 1)$ is 4π.

Filter Type Defined Zero Locations

The frequency resonse of a Type 2 FIR filter always has a zero at $\omega = \pi$. This can be demonstrated by example.

First note that a Type 2 filter is even symmetric with odd order. One such system is $h[n] = [h_0, h_1, h_2, h_2, h_1, h_0]$. Taking the z-transform of $h[n]$ gives

$$H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + h_2 z^{-3} + h_1 z^{-4} + h_0 z^{-5}$$

Evaluating $H(z)$ at $z = -1$ gives

$$H(-1) = h_0 - h_1 + h_2 - h_2 + h_1 - h_0 = 0$$

Thus

$$H(e^{j\pi}) = H(-1) = 0$$

Similarly, rules can be derived for Type 3 and 4 filters

Type	**Defined Zeros**
1	none
2	$\omega = \pi$
3	$\omega = 0, \pi$
4	$\omega = 0$

TopHat Question 6: Zero Location for H(z)

If $H(z) = h_0 + h_1 z^{-1} - h_1 z^{-2} - h_0 z^{-3}$, is there a zero at $H(e^{j0})$?

A Yes

B No

Summary of FIR Filter Types

$$H(e^{j\omega}) = A(\omega)e^{j(\beta - \frac{M}{2}\omega)}$$

Type	**1**	**2**	**3**	**4**
Order (M)	even	odd	even	odd
$h[n]$	even symmetric	even symmetric	odd symmetric	odd symmetric
$A(\omega)$	even (about $\omega = 0$)	even	odd (about $\omega = 0$)	odd
$A(\omega) period$	2π	4π	2π	4π
β	0	0	$\pi/2$	$\pi/2$
$H(e^{j0})$	arbitrary	arbitrary	0	0
$H(e^{j\pi})$	arbitrary	0	0	arbitrary
Common Uses	LP, HP, BP, BS, Mulitband	LP, BP	Differentiator, Hilbert Transform	Differentiator, Hilbert Transform

FIR Filter Zero Locations

The zero locations for the all-zero linear phase FIR filters are not arbitrary. The zero locations are constrained and possible locations are derived as follows:
The symmetry condition for linear phase filters is

$$h[n] = \pm h[M-n]$$

and in the z-domain

$$H(z) = \pm z^{-M} H(\frac{1}{z}).$$

If z_0 is a zero of $H(z)$, $H(z_0) = 0$ and $H(z_0^*) = 0$, since roots of real coefficient polynomials are complex conjugate pairs, then

$$\begin{aligned} H(z_0) &= \pm z_0^{-M} H(\tfrac{1}{z_0}) = 0 \\ H(z_0^*) &= \pm (z_0^*)^{-M} H(\tfrac{1}{z_0^*}) = 0 \end{aligned}$$

and thus

$$H(\frac{1}{z_0}) = H(\frac{1}{z_0^*}) = 0$$

FIR Filter Zero Locations

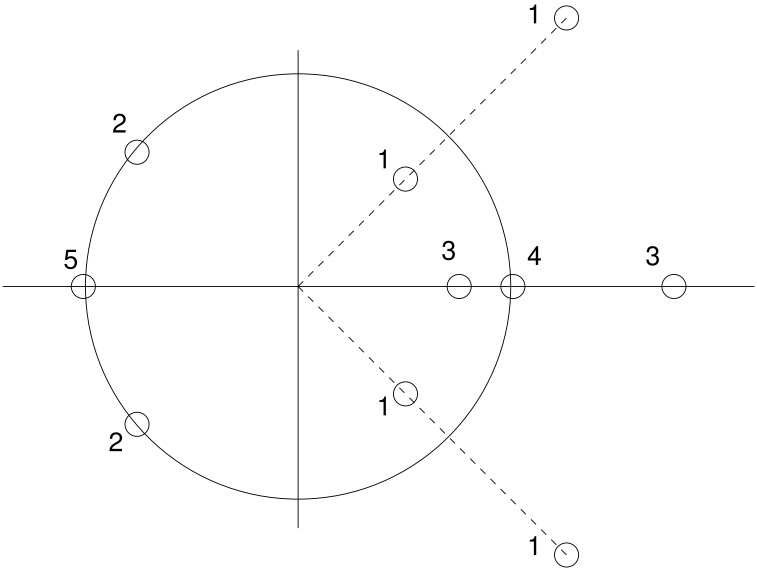
$$H(\frac{1}{z_0}) = H(\frac{1}{z_0^*}) = 0$$

indicates that if z_0 is a zero of a real valued linear phase filter, then so are z_0^*, $1/z_0$ and $1/z_0^*$. It follows that

1. generic zeros of a linear phase filter exist in sets of 4 ($z_0 = re^{j\phi}$,$z_0^* = re^{-j\phi}$, $1/z_0 = r^{-1}e^{-j\phi}$ and $1/z_0^* = r^{-1}e^{j\phi}$)
2. zeros on the unit circle exist in sets of 2 ($z_0 = e^{\pm j\phi}$, $z_0 \neq \pm 1$).
3. zeros on the real line exist in sets of 2 ($z_0 = r, 1/r$).
4. a zero at $z_0 = 1$ has no other conditional zero locations.
5. a zero at $z_0 = -1$ has no other conditional zero locations.

The figure on the following slide is a plot of the above 5 possible zero locations.

FIR Filter Zero Locations



Subsection 2

Impulse Response Truncation

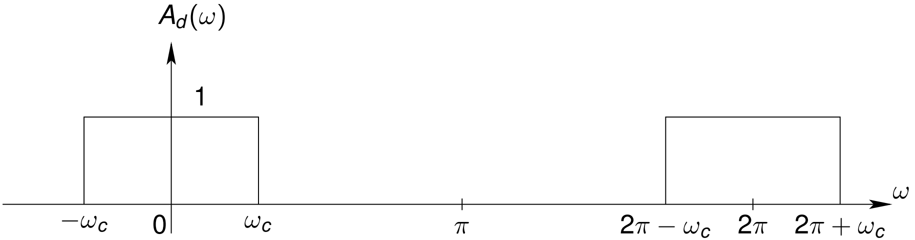
Introduction to the IRT Design Method

- FIR filter design usually involves calculating values for $h[n]$ that meet a desired $H(e^{j\omega})$ specification.
- There are a number of different FIR design methods that will be explored, starting with the simplest method, called impulse response truncation (IRT).
- The IRT method is introduced by first considering a version that involves delaying the impulse response.
- This delay IRT method involves the following steps:
  - Specify the desired ideal filter (with zero phase response),
  - Generate the infinite impulse response using the Inverse DTFT (IDTFT),
  - Truncate the impulse response.
  - Delay (shift) the impulse response to make it causal.
- The delay IRT method is demonstrated in the example on the following slide.

Example 9: LPF Design using the Delay IRT Method

Design a lowpass filter with cutoff frequency, ω_c, and order M.

- **Solution:** The ideal frequency response for this lowpass filter is (assume a zero phase response)



- Recall from the table summarizing the four filter types on slide 71, only types 1 and 2 can be lowpass filters. Choose Type 1 for this example.
- Take the IDTFT of $A_d(e^{j\omega})$ to calculate the non-causal impulse response, $h_{nc}[n]$ (recall the frequency response is zero phase).

Example 9: LPF Design using the Simplified IRT Method

$$h_{nc}[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} A_d(e^{j\omega})e^{j\omega n}d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi} \underbrace{A_d(e^{j\omega})}_{even}(\cos(\omega n) + j\underbrace{\sin(\omega n)}_{odd})d\omega$$

- The above expression can be simplified by noting that an even function times an odd function is odd and the integral of an odd function is 0.

$$h_{nc}[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} A_d(e^{j\omega})\cos(\omega n)d\omega = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} \cos(\omega n)d\omega$$

- Evaluating the integral gives

$$h_{nc}[n] = \begin{cases} \frac{1}{2\pi}2\omega_c; & n = 0 \\ \frac{1}{2\pi n}\sin(\omega n)\Big|_{-\omega_c}^{\omega_c}; & \text{otherwise} \end{cases}$$

- and simplifying

$$h_{nc}[n] = \begin{cases} \frac{\omega_c}{\pi}; & n = 0 \\ \frac{\omega_c}{\pi}\frac{\sin(\omega_c n)}{\pi\omega_c}; & \text{otherwise} \end{cases}$$

Example 9: LPF Design using the Delay IRT Method

- Delay $h_{nc}[n]$ to make it causal, call this new sequence $h[n]$.

$$h[n] = h_{nc}\left[n - \frac{M}{2}\right] = \begin{cases} \frac{\omega_c}{\pi}; & n = 0 \\ \frac{\omega_c}{\pi}\frac{\sin(\omega_c(n-\frac{M}{2}))}{\omega_c(n-\frac{M}{2})}; & n = 1, 2, \cdots, M \end{cases}$$

- Using $\omega_c = 2\pi f_c$,

$$h[n] = \begin{cases} 2f_c; & n = 0 \\ 2f_c\frac{\sin(2\pi f_c(n-\frac{M}{2}))}{2\pi f_c(n-\frac{M}{2})}; & n = 1, 2, \cdots, M \end{cases}$$

- Using $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$

$$h[n] = 2f_c \mathrm{sinc}\left(2f_c\left(n - \frac{M}{2}\right)\right); \quad n = 0, 1, 2, \cdots, M$$

IRT Procedure

- In the previous example, the impulse response for the non-causal system, $h_{nc}[n]$, was generated and then it was shifted by $\frac{M}{2}$ to make it causal ($h[n] = h_{nc}\left[n - \frac{M}{2}\right]$). This restricts M to an even number, since a discrete-time sequence can only be shifted a integer number of samples.
- An alternative approach, which avoids the even number constraint for M, is to incorporate a linear phase term into the frequency response,

$$A(\omega)e^{-j\omega M/2}$$

- After taking the IDTFT, the resulting impulse response, once truncated to length $N = M + 1$, will be causal.
- This is implemented in the impulse response truncation filter design procedure listed on the next slide.

IRT Procedure

1. Specify the desired ideal amplitude response $A_d(\omega)$, ie. LP, HP, etc.
2. Choose the phase characteristics: integer delay (even M) or fractional delay (odd M) and initial phase $\beta = 0$ (for even symmetric $h[n]$) or $\pi/2$ (for odd symmetric $h[n]$).
3. Choose the filter order M (even for Type 1 or 3, odd for Type 2 or 4).
4. The resulting ideal desired frequency response is

$$H_d(e^{j\omega}) = A_d(\omega)e^{j(\mu\frac{\pi}{2} - \frac{M}{2}\omega)}$$

   where $\mu = 0$ for even symmetric $h[n]$ and $\mu = 1$ for odd symmetric $h[n]$.
5. Compute the impulse response of the ideal filter using the IDTFT,

$$h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} A_d(e^{j\omega})e^{j(\mu\frac{\pi}{2} - \omega M/2)}e^{j\omega n}d\omega.$$

6. Truncate the impulse response

$$h[n] = \begin{cases} h_d[n]; & 0 \leq n \leq M \\ 0; & \text{otherwise} \end{cases}$$

IRT: Lowpass, Highpass and Bandpass Filters

- A common form for the impulse response can be derived for the three types of filters: lowpass, highpass and bandpass.
- Define the desired amplitude response

$$A_d(\omega) = \begin{cases} 1; & \omega_1 \le |\omega| \le \omega_2 \\ 0; & \text{otherwise} \end{cases}$$

- For $A_d(\omega)$,

$$\begin{array}{lcl} \text{if } \omega_1 = 0 & \rightarrow & \text{a LPF} \\ \text{if } \omega_2 = \pi & \rightarrow & \text{a HPF} \\ \text{otherwise} & \rightarrow & \text{a BPF} \end{array}$$

- Referring to the table on page 71, only the Type 1 filter can be a LP, HP or BP.
- A Type 1 filter has $\beta = 0$, thus $\theta_A(\omega) = -\omega M/2$, where M is the filter order. Thus the desire impulse response is

$$H_d(\omega) = \begin{cases} e^{-j\omega M/2}; & \omega_1 \le |\omega| \le \omega_2 \\ 0; & \text{otherwise} \end{cases}$$

IRT: Lowpass, Highpass and Bandpass Filters

- Taking the IDTFT

$$
\begin{aligned}
h_d[n] &= \tfrac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega n}d\omega \\
&= \tfrac{1}{2\pi}\int_{-\omega_2}^{-\omega_1} e^{-j\omega M/2}e^{j\omega n}d\omega + \tfrac{1}{2\pi}\int_{\omega_1}^{\omega_2} e^{-j\omega M/2}e^{j\omega n}d\omega \\
&= \tfrac{1}{2\pi}\int_{-\omega_2}^{-\omega_1} e^{j\omega(n-M/2)}d\omega + \tfrac{1}{2\pi}\int_{\omega_1}^{\omega_2} e^{j\omega(n-M/2)}d\omega \\
&= \tfrac{1}{2\pi j(n-M/2)}\left(e^{-j\omega_1(n-M/2)} - e^{-j\omega_2(n-M/2)}\right) + \tfrac{1}{2\pi j(n-M/2)}\left(e^{j\omega_2(n-M/2)} - e^{j\omega_1(n-M/2)}\right) \\
&= \tfrac{1}{2\pi j(n-M/2)}\left(e^{j\omega_2(n-M/2)} - e^{-j\omega_2(n-M/2)}\right) - \tfrac{1}{2\pi j(n-M/2)}\left(e^{j\omega_1(n-M/2)} - e^{-j\omega_1(n-M/2)}\right)
\end{aligned}
$$

- Using Euler's

$$h_d[n] = \frac{\sin(\omega_2(n-M/2))}{\pi(n-M/2)} - \frac{\sin(\omega_1(n-M/2))}{\pi(n-M/2)}$$

- Putting this in terms of f_1 and f_2, and using $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$ gives,

$$h_d[n] = 2f_2\mathrm{sinc}(2f_2(n-M/2)) - 2f_1\mathrm{sinc}(2f_1(n-M/2))$$

IRT: Lowpass, Highpass and Bandpass Filters

- Truncate to a length of $N = M + 1$,

$$h_d[n] = 2f_2 \mathrm{sinc}(2f_2(n - M/2)) - 2f_1 \mathrm{sinc}(2f_1(n - M/2)) \quad n = 0, 1, \cdots, M$$

- For a LPF, with $f_2 = f_c$ and $f_1 = 0$,

$$h_d[n] = 2f_c \mathrm{sinc}(2f_c(n - M/2))$$

- For a HPF, with $f_2 = f_c$ and $f_1 = 1/2$,

$$h_d[n] = \mathrm{sinc}(n - M/2) - 2f_c \mathrm{sinc}(2f_c(n - M/2)) \quad n = 0, 1, \cdots, M$$

- For the HPF, what is $\mathrm{sinc}(n - M/2) \quad n = 0, 1, \cdots, M$ equivalent to? (note M is even for a Type 1 filter)

Example 10: IRT: Bandpass Filter

Design a bandpass filter, using the impulse response truncation technique, with cutoff frequencies of

$$\omega_1 = 0.2\pi, \omega_2 = 0.6\pi$$

and an order of 40. Plot the impulse response on one figure window, the amplitude response (linear) and phase response on a second figure window and the magnitude response (linear) and associated phase response (with phase in the range $-\pi$ to π) on a third figure window. Plot the frequency response plots as a function of frequency in cycles/samples for the range -0.5 to 0.5.

Multiband Filters

- A multiband filter is a superposition of bandpass filters. If the desired amplitude response has K bands, where the cutoff frequencies for the k^{th} band are ω_{1k} and ω_{2k} and the band gain is C_k, then

$$A_d(\omega) = \sum_{k=1}^{K} A_{d,k}(\omega)$$

  where

$$A_{d,k}(\omega) = \begin{cases} C_k; & \omega_{1k} \leq \omega \leq \omega_{2k} \\ 0; & \text{otherwise} \end{cases}$$

- Using the impulse response truncation technique and the bandpass impulse response expression on slide 84, the impulse response for the general multiband filter (K bands) is

$$h[n] = \sum_{k=1}^{K} C_k[2f_{2k}\mathrm{sinc}(2f_{2k}(n - M/2)) - 2f_{1k}\mathrm{sinc}(2f_{1k}(n - M/2))]$$

- A bandstop filter is a special case of a multiband filter with two bands, having $C_1 = C_2 = 1$, $f_{11} = 0$, and $f_{22} = 1/2$.

Example 11: IRT: Two Band Filter

Design a two band filter, using the impulse response truncation technique, with $M = 80, f_{11} = 0.1, f_{21} = 0.2, f_{12} = 0.35, f_{22} = 0.4, C_1 = 1, C_2 = 0.5$. Plot the impulse response on one figure window and the amplitude response on a second figure window for f from -0.5 to 0.5 cycles/sample. Use the MATLAB function fft to generate the amplitude response.

How Good is the IRT Method

- IRT filters are optimal in the sense of minimizing the integral of the square error,

$$\epsilon = \frac{1}{2\pi}\int_{-\pi}^{\pi} |H_d(e^{j\omega}) - H(e^{j\omega})|^2 d\omega$$

- If $h[n], 0 \leq n \leq M$ are the coefficients of a causal FIR filter of order M, then using Parseval's theorem,

$$\epsilon = \sum_{n=-\infty}^{\infty} (h_d[n] - h[n])^2 = \sum_{n=-\infty}^{-1} h_d^2[n] + \sum_{n=M+1}^{\infty} h_d^2[n] + \sum_{n=0}^{M} (h_d[n] - h[n])^2$$

- Only the third term is a function of $h[n]$. This term is non-negative and ϵ is minimized if this term is 0, which occurs if

$$h[n] = h_d[n] \quad 0 \leq n \leq M$$

  which is the criterion for the impulse response truncation method.

How Good is the IRT Method

- But, even though this technique is optimal in the sense of the integral of the square error, it is not considered a good technique, because of the ripple in the frequency response near discontinuity locations in $H_d(e^{j\omega})$.
- As M increases, ϵ decreases, but the magnitude of the ripple does not decrease to 0, it approaches a finite number. This is known as Gibbs phenomenon.

Example 12: Gibbs Phenomenon

Compare the amplitude responses for two impulse response truncation lowpass filters. One with the order $M = 10$ and the other with $M = 40$.Set the cutoff to be 0.25 cycles/sample.
Use the MATLAB function fft in the process of generating the amplitude response. Plot both responses on one plot as a function of frequency in cycles/samples for the range -0.5 to 0.5.

IRT Method Limitations

- In the previous demonstration, the passband and stopband tolerance parameters, δ_p and δ_s are approximately the same for $M = 10$ and $M = 40$. Their value is close to 0.09.
- For the IRT method, this value remains approximately constant, regardless of the filter order.
- This phenomenon is named after J.W. Gibbs.
- Practically, the IRT method is suitable only for filters whose tolerances are not less than 0.09 or

$$A_p = 20 \log_{10}(1 + 0.09) = 0.75dB$$

$$A_s = -20 \log_{10}(0.09) = 20.9dB$$

- Other design techniques can be use to mitigate the Gibbs phenomenon. For example, the windows FIR design approach, which will be covered later.

Subsection 3

Filter Transformations

Lowpass to Highpass Transformation

- A lowpass filter can be converted to a highpass filter using simple techniques that can be implemented in real time.
- The two techniques that will be examined are:
  - Technique 1: Subtract a lowpass filter from an allpass filter.
  - Technique 2: Modulation theorem.

Technique 1: Lowpass Subtracted from a Allpass Filter

Allpass Filter

- An allpass filter has a constant gain (usually 1) magnitude response at all frequencies, ie.

$$|H(e^{j\omega})| = 1$$

- There are no restrictions on the phase response.
- An allpass filter implemented as an FIR filter is simply

$$H(e^{j\omega}) = A(\omega)e^{j\theta_A(\omega)}$$

where $A(\omega) = 1$ and $\theta_A(\omega) = -\alpha\omega$.

Technique 1: Lowpass Subtracted from a Allpass Filter

- A highpass filter, with frequency response $H_{HP}(e^{j\omega})$, can be generated from a lowpass filter, with frequency response $H_{LP}(e^{j\omega})$, by subtracting the lowpass response from and allpass filter frequency response that has the same phase response as $H_{LP}(e^{j\omega})$,

$$H_{HP}(e^{j\omega}) = \underbrace{e^{j\theta_A(\omega)}}_{\text{allpass}} - \underbrace{A(\omega)e^{j\theta_A(\omega)}}_{H_{LP}(e^{j\omega})} = (1 - A(\omega))e^{j\theta_A(\omega)}$$

- For a Type 1 FIR linear phase filter, $\theta_A(\omega) = -\frac{M}{2}\omega$, and

$$H_{HP}(e^{j\omega}) = (1 - A(\omega))e^{-j\frac{M}{2}\omega}$$

- Converting this expression to its z-transform gives,

$$H_{HP}(z) = z^{-M/2} - H_{LP}(z)$$

- Taking the inverse z-transform gives

$$h_{HP}[n] = \delta[n - M/2] - h_{LP}[n]$$

- The cutoff frequency of the lowpass filter occurs at approximately $A(\omega) = 0.5$ (ie. the 6 dB magnitude squared point), thus the highpass filter will have the same cutoff frequency as the lowpass filter.

Technique 2: Modulation Theorem

Modulation Theorem

- The modulation theorem is defined as

$$h[n]x[n] \xrightarrow{\text{DTFT}} \frac{1}{2\pi} H(e^{j\omega}) \circledast X(e^{j\omega})$$

  where $\circledast$ represents circular (periodic) convolution.
- If $x[n] = \cos(\omega_0 n)$, with $\omega_0 = \pi$, specify

$$h_{HP}[n] = h_{LP}[n]\cos(\pi n)$$

- In the frequency domain, the circular convolution between the lowpass frequency response and the impulses due to the cosine will result in a highpass filter response.
- $h_{HP}[n]$ can be simplified,

$$h_{HP}[n] = h_{LP}[n]0.5(e^{j\pi n} + e^{-j\pi n}) = h_{LP}[n]e^{j\pi n} = h_{LP}[n](-1)^n$$

Technique 2: Modulation Theorem

- Taking the DTFT

$$H_{HP}(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h_{LP}[n]e^{j\pi n}e^{-j\omega n} = \sum_{n=-\infty}^{\infty} h_{LP}[n]e^{-j(\omega-\pi)n} = H_{LP}(e^{j(\omega-\pi)})$$

- Thus $H_{HP}(e^{j\omega})$ is generated by shifting $H_{LP}(e^{j\omega})$ by π.
- The cutoff frequency of the lowpass filter is ω_c^{LP} and the cutoff frequency of the high pass filter will be

$$\omega_c^{HP} = \pi - \omega_c^{LP}$$

- In the frequency domain, an example of the relationship is

$$H_{HP}(e^{j\omega_c^{HP}}) = H_{LP}(e^{j(\omega_c^{HP}-\pi)}) = H_{LP}(e^{j(\pi-\omega_c^{LP}-\pi)}) = H_{LP}(e^{-j\omega_c^{LP}})$$

Example 13: Convert LPF to HPF

Design a lowpass filter, using the impulse response truncation technique, with a cutoff frequency of $f_c = 0.1$ and an order of 30. Convert this filter to a highpass filter with a cutoff frequency of $f_c = 0.1$.

Section 3

Analysis of Finite Precision Effects

Subsection 1

Review of Signal Averages

Signal Averages

Total Power

The total power in a signal, $x[n]$, is the average power, given by

$$\overline{P} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=-N/2}^{N/2-1} x^2[n] = \underbrace{\overline{P}_{AC} + \overline{P}_{DC} = \overline{\sigma^2} + \overline{\mu}^2}_{\text{to be defined}}$$

DC Component

The DC component of signal, $x[n]$, is the average value, given as

$$\overline{\mu} = \underbrace{\overline{x[n]}}_{\text{time average}} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=-N/2}^{N/2-1} x[n]$$

DC Power

The DC power in signal, $x[n]$, is

$$\overline{P}_{DC} = \mu^2 = (\overline{x[n]})^2$$

Signal Averages

AC Component

The AC component of signal, $x[n]$, is

$$x_{AC} = x[n] - \overline{\mu}$$

AC Power

The AC power in a signal, $x[n]$, is

$$\overline{P}_{AC} = \overline{\sigma^2} = (\overline{x_{AC}^2[n]}) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=-N/2}^{N/2-1} (\underbrace{x[n] - \overline{\mu}}_{x_{AC}[n]})^2$$

Example 14: Signal Averages

- The signal, $x[n]$, is a periodic sawtooth given by

$$x[n] = n; \quad 1 \leq n \leq 6.$$

Determine the DC component, the AC component, the AC power, the DC power and the total power of the signal $x[n]$.

Subsection 2

Random Signal Processing

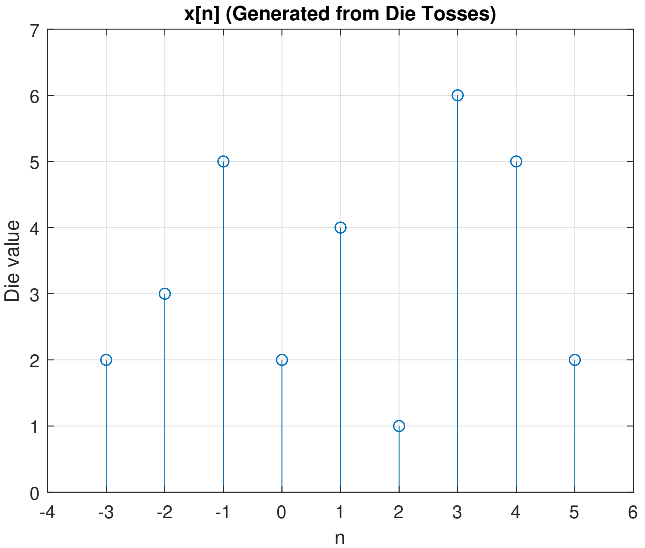
Random Signals

- A signal $x[n]$ is deterministic if $x[n]$ can be determined from knowledge of a finite number of samples of $x[n]$. An example of a deterministic signal is

$$x[n] = \sum_{k=1}^{K} A_k \cos(\omega_k n + \phi_k).$$

- A signal $x[n]$ is random if any sample $x[n_0]$ cannot be determined from knowledge other samples in $x[n]$, except $x[n_0]$.
- A random signal could be generated using a six-sided die. Let the n^{th} sample of $x[n]$ be given by the value of the die (number of dots) on the n^{th} roll of the die.

Random Signals



Random Signals

- $x[n]$ generated using a die is a random signal that cannot be characterized using deterministic techniques.
- Instead statistical averages are typically used.
- Estimates of the statistical averages can be calculated using the time averages specified in slides 221 and 102. But, a large number of samples are needed to get a good estimate. Obtaining a large number of samples may not be possible in some cases.
- A different approach is needed.
- If the amplitude distribution of $x[n]$ is known, this can be used in the calculation of statistical averages.

Example 15: Introduction to Statistical Averages

- A random sequence is generated using the value on a single tossed die. Assume the die is tossed *N* times, where *N* approaches infinity. Determine μ and σ^2.

Amplitude Probability Distribution

- The amplitude probability distribution of a signal, $x[n]$, is a function of amplitude that specifies the proportion of each amplitude that is expected to occur in $x[n]$.
- The amplitude probability distribution is represented as $p(A)$. $p(A)$ is also referred to as a probability density function.
- For example, a signal, $x[n]$, generated from the value on a roll of a die has an amplitude probability distribution of

$$p(A) = \begin{cases} \frac{1}{6}; & A = 1, 2, 3, 4, 5, 6 \\ 0; & \text{otherwise} \end{cases}$$

- Using $p(A)$, the statistical averages can be calculated as

$$\mu = \sum_A A p(A) \quad \text{(mean)}$$

$$\sigma^2 = \sum_A (A - \mu)^2 p(A) \quad \text{(variance)}$$

$$P = \sigma^2 + \mu^2 = \sum_A A^2 p(A) \quad \text{(mean square value)}$$

Example 16: Statistical Averages for a Fair Die

- A signal, $x[n]$, is generated using a single tossed fair die. Determine μ and σ^2, using the amplitude probability distribution,

$$p(A) = \begin{cases} \frac{1}{6}; & A = 1, 2, 3, 4, 5, 6 \\ 0; & \text{otherwise} \end{cases}$$

Example 17: Statistical Averages for a Non-Fair Die

- A fair die is one where each of the six sides is equally likely to occur. It is possible to modify a die, such that it is not fair. Given such a die, with amplitude probability distribution,

$$p(A) = \begin{cases} \frac{1}{8}; & A = 2, 3, 4, 5 \\ \frac{1}{4}; & A = 1, 6 \\ 0; & \text{otherwise} \end{cases}$$

  Determine μ, the total power, P, and use these two to calculate σ^2.

TopHat Question 7: Mean Value

The signal $x[n]$ is generated by repeatedly tossing a fair coin and mapping a head to A volts and a tail to $-A$ volts. What is the mean value of $x[n]$?

$$\text{A } A \text{ volts}, \quad \text{B } A/2 \text{ volts}, \quad \text{C } 0 \text{ volts}$$

Subsection 3

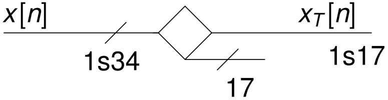
FIR Filter Finite Wordlength Effects

Finite Wordlength Effects

- When implementing a filter on an FPGA, the number of bits used to represent input data, filter coefficients and for performing arithmetic operations should be as small as possible to improve efficiency and reduce the cost of the filter.
- Generally, using a finite number of bits lowers the performance of the filter. This is also referred to as finite wordlength effects.
- Finite wordlength effects can be divided into four categories:
  1. Analog to Digital Converter (ADC) noise: sampling a continuous time signal results in quantization noise. This noise limits the obtainable SNR.
  2. Coefficient quantization errors: Results from representing coefficients with a finite number of bits. It can adversely effect the desired frequency response. For example, in the stopband, it limits the maximum attenuation possible.
  3. Round-off errors due to scaling and arithmetic operations: For example, this results from discarding lower order bits after a multiplication.
  4. Arithmetic overflow: Results when the output of arithmetic operations exceed the wordlength. Handled by scaling the signals and/or coefficients.
- Here, the focus is on round-off errors, which for FPGA implementations typically involves truncation.

Truncation Errors

- A truncated signal is generated by discarding one or more least significant bits (LSBs).
- For example, a trimmer is used to truncate a signal.



- Consider a signal, $x[n]$, with a wordlength of N bits. Let the binary number representing $x[n]$ and $x_T[n]$ be defined as,

$$\underbrace{\underbrace{\overbrace{b_0 b_1 b_2 \cdots b_{N_T-1}}^{N_T \text{ bits retained}}}_{x_T[n]} \; \underbrace{\overbrace{b_{N_T} \cdots b_{N-1}}^{N-N_T \text{ bits truncated}}}_{\text{trimmed bits}}}_{x[n]}$$

Truncation Errors

- The decimal worth of the truncated bits removed is always positive, regardless of whether $x[n]$ is signed or unsigned.
- The truncated bits removed can be expressed as the decimal worth with respect to the LSB of $x_T[n]$ (ie. b_{N_T-1}).
- The worth of the LSB is defined as s. Here s is the decimal worth of b_{N_T-1}.
- The decimal worth of the truncated bits removed, defined as $-q[n]$, is

$$-q[n] = s \times [\underbrace{b_{N_T}[n]2^{-1} + b_{N_T+1}[n]2^{-2} + \cdots + b_{N-1}[n]2^{-(N-N_T)}}_{\text{unsigned fraction}}]$$

- Clearly, $x[n] = x_T[n] + (-q[n])$, thus

$$x_T[n] = x[n] + q[n]$$

- $q[n]$ is called the quantization noise and $q[n]$ is negative, since $-q[n]$ is positive.

Statistics of Quantization Noise

- Here we are interested in the statistics of $q[n]$, specifically the mean, μ, (DC component) and the variance, σ^2 (AC power).
- The quantization noise, $q[n]$, can be viewed as random noise.
- Thus to calculate the statistical averages, μ and σ^2, knowledge of the amplitude probability distribution is required.
- It is assumed that each of the possible values of $q[n]$ are equally likely to occur.
- There are 2^{N-N_T} possible values for $q[n]$ and these values have a uniform amplitude probability distribution between

$$-s \times \left(\frac{2^{N-N_T} - 1}{2^{N-N_T}} \right) \text{ and } 0$$

Example 18: Quantization Noise Statistical Averages

$x[n]$ is a 1s10 number that is truncated to a 1s8 number. Determine the statistical averages μ, σ^2 and the mean square value for the resulting quantization noise.

Quantization Noise Formula

- The following is a summary of the statistical averages for quantization noise due to truncation.
- If L bits are removed from $x[n]$ by truncation to form the quantization noise, $q[n]$, then the following are the statistical averages for $q[n]$ (see the derivation of this expression in the EE365 notes):

$$\mu = \overline{q[n]} = -\frac{s}{2}(1 - 2^{-L})$$

$$\sigma^2 = \overline{(z[n] - \mu)^2} = \frac{s^2}{12}(1 - 2^{-2L})$$

$$P = \overline{q[n]^2} = \sigma^2 + \mu^2$$

Example 19: Quantization Noise Formulae

Repeat the example on slide 118 using the quantization noise formulae on slide 133,
Restating the problem: $x[n]$ is a 1s10 number that is truncated to a 1s8 number. Determine the statistical averages μ, σ^2 and the mean square value for the resulting quantization noise.

Modelling Truncation Quantization Noise

- Recall, that for a truncated signal is

$$x_T[n] = x[n] + q[n]$$

where $q[n]$ is the quantization noise. Thus the truncation is modelled as additive noise.

- Mathematically, it is much easier to work with a statistical model for $q[n]$. Here a statistical model is proposed, which involves replacing $q[n]$ with a source that has the same DC component (μ) and AC power (σ^2), but it is a different signal.
- The statistical model is

$$\tilde{q}[n] = \underbrace{\overline{q[n]}}_{\text{DC component}} + \underbrace{q_{AC}[n]}_{\text{AC statistical replacement}}$$

Modelling Truncation Quantization Noise

- In the statistical model, $\tilde{q}[n]$ must have
  1. The same DC component as $q[n]$,
  $$\overline{\tilde{q}[n]} = \overline{q[n]} = \mu$$
  2. The same AC power as $q[n]$,
  $$\overline{(\tilde{q}[n] - \mu)^2} = \overline{q^2_{AC}[n]} = \sigma^2$$
- In the statistical model $\tilde{q}[n]$ is independent of $x[n]$ and is not influenced by $x[n]$. But this is not the case for the actual system, since $q[n]$ depends on the $x[n]$.

Generating Quantization Noise using MATLAB

- In MATLAB the statistical model noise, $\tilde{q}[n]$, could be generated using

```
q_tilde = s * floor(rand(1)*2^L)*2^(-L);
```

  where `L` is the number of bits truncated.
- If $L \geq 4$, $\tilde{q}[n]$ is approximately equal to

```
q_tilde = s * floor(rand(1));
```

- The actual quantization noise can be generated using

```
xT = floor(x*2^L*2^(-L));
q = x - xT;
```

Power Spectral Density of Quantization Noise

- The AC power in $q[n]$ is defined as σ_q^2, and since $q[n]$ is random, this power is spread uniformly across all frequencies (ie. it is white).
- The one-sided power spectral density (PSD) of the AC component of the quantization noise, $q[n]$, is denoted as $S_{qq}(e^{j2\pi f})$.
- $S_{qq}(e^{j2\pi f})$ is defined as the power in the infinitesimal bandwidth Δf divided by Δf.
- The power in the infinitesimal bandwidth Δf is the proportion of the infinitesimal bandwidth with respect to all the possible frequencies times the AC power, σ_q^2, and is given as

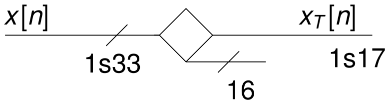
$$\sigma_q^2 \times \frac{\Delta f \text{ cycles/sample}}{1/2 \text{ cycles/sample}} = 2\sigma_q^2 \times \Delta f$$

- Dividing this by Δf, gives the one-sided PSD

$$S_{qq}(e^{j2\pi f}) = \frac{2\sigma_q^2 \times \Delta f}{\Delta f} = 2\sigma_q^2 \text{ V}^2\text{/cycles/sample}$$

Example 20: Quantization Noise PSD

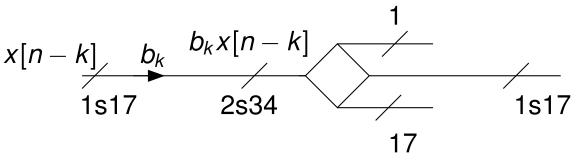
Given the trimmer:



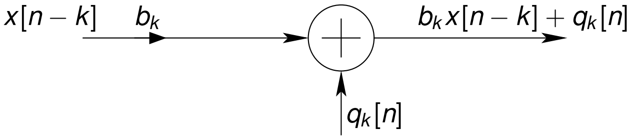
What is the power spectral density of the AC component of the quantization noise produced by the trimmer in units $\mathrm{dB_{mV}/cycles/sample}$.

FIR Filter Quantization Noise

- An FIR filter is defined by the difference equation $y[n] = \sum_{k=0}^{N-1} b_k x[n-k]$ which when implemented in an FPGA, must include a trimmer. For example, the term $b_k x[n-k]$ is implemented as



is modelled as



FIR Filter Quantization Noise

- Thus, for the model

$$y[n] = \sum_{k=0}^{N-1} b_k x[n-k] + q_k[n] = \sum_{k=0}^{N-1} b_k x[n-k] + \underbrace{\sum_{k=0}^{N-1} q_k[n]}_{q[n]}$$

Here $q[n]$ is simply the sum of each of the quantization noise terms for the N trimmers.

- These N trimmer noise sources are uncorrelated and thus the total AC power is N times the noise power in $q_k[n]$ (ie. they add on a power basis).
- The total DC value is the sum of the N DC values (which are all the same).

$$\sigma_q^2 = \sum_{k=0}^{N-1} \overline{(q_k[n] - \mu_k)^2} = N\frac{s^2}{12} \quad \text{for large } L$$

$$\mu_q = \sum_{k=0}^{N-1} \overline{q_k[n]} = -N\frac{s}{2} \quad \text{for large } L$$

Power in the Sum of Two Signals

- If two zero mean signals, $x_1[n]$ and $x_2[n]$, are uncorrelated then the power of the sum of the two signals is

$$
\begin{aligned}
P_{x_1+x_2} &= \overline{(x_1[n] + x_2[n])^2} \\
&= \overline{x_1^2[n] + 2x_1[n]x_2[n] + x_2^2 2[n]} \\
&= \overline{x_1^2[n]} + \underbrace{\overline{2x_1[n]x_2[n]}}_{=0 \text{ if uncorrelated}} + \overline{x_2^2[n]} \\
&= \overline{x_1^2[n]} + \overline{x_2^2[n]} \\
&= P_{x_1} + P_{x_2}
\end{aligned}
$$

  This is referred to as adding on a power basis.
- Define $R_{x_1 x_2}$ as

$$
R_{x_1 x_2} = \frac{1}{N} \sum_{n=0}^{N-1} x_1[n] x_2[n].
$$

  $R_{x_1 x_2}$ is a measure of correlation that gives an indication of how similar x_1 and x_2 are.
- If $x_1[n]$ and $x_2[n]$ are zero mean then they are uncorrelated if $R_{x_1 x_2} = 0$.

Example 21: Power of the Sum of Two Signals

Given two signals,

$$x_1[n] = [1, 2, 3, 2, 1, 0, -1, -2, -3, -2, -1, 0]$$

and

$$x_2[n] = [2, 1, 0, -1, -2, -3, -2, -1, 0, 1, 2, 3],$$

calculate $R_{x_1 x_2}$. Compare $P_{x_1} + P_{x_2}$ with $P_{x_1+x_2}$.

Example 22: Power of the Sum of Two Uncorrelated Signals

1. Use MATLAB to generate two uncorrelated random signals with a uniform amplitude probability distribution.
   1.1 Compare the power of the individual signals with the power of the sum of the two signals.
   1.2 Calculate $R_{x_1 x_2}$ for the two signals.

Example 23: Power of the Sum of Two Correlated Signals

1. Use MATLAB to generate two correlated random signals with a uniform amplitude probability distribution.
   1.1 Compare the power of the individual signals with the power of the sum of the two signals.
   1.2 Calculate $R_{x_1 x_2}$ for the two signals.

Power Spectrum

- Previously, the one-side PSD for quantization noise was defined as

$$S_{qq}(e^{j2\pi f}) = \frac{2\sigma_q^2 \times \Delta f}{\Delta f} = 2\sigma_q^2 \ \mathrm{V}^2/\mathrm{cycles}/\mathrm{sample}$$

- $S_{qq}(e^{j2\pi f})$ is sometimes referred to by engineers as the power spectrum. But, this is not technically correct.
- The power spectrum is a plot of power versus the center frequency of a bandpass filter.
- For example, spectrum analyzers plot the power spectrum, which is the power at the output of the resolution bandwidth (RBW) filter, as a function of frequency.
- The PSD could be estimated from the power spectrum by dividing by the bandwidth of the RBW filter

$$\hat{\mathrm{PSD}} = \frac{\text{Power Spectrum}}{\text{Bandwidth of RBW Filter}} \ \ \mathrm{watts}/\mathrm{Hz}$$

Forms of Quantization Noise

- Quantization noise can be
  - random with no DC component
  - random with a DC component
  - periodic
  - or a mixture of all three
- Here periodic quantization noise will be investigated further.
- Quantization noise is periodic iff the signal being quantized is periodic. Perhaps the best way to describe periodic noise is through an example.

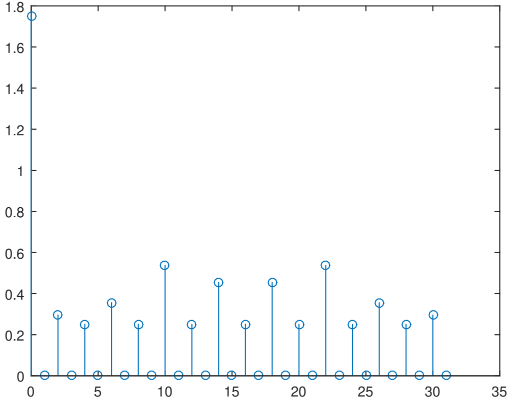
Example 24: Periodic Quantization Noise

Generate two periods of a 1/32 cycles/sample sinusoid. Quantize the sinusoid to 3 bits, using truncation and rounding. Plot the quantization noise and the spectral content of the quantization noise.

TopHat Question 8: Fundamental Period of Quantization Noise 1

The figure shows the fft of the quantization noise from a quantized sinusoidal signal, where the fft is a multiple of the fundamental period. What is the fundamental period of the quantization noise?

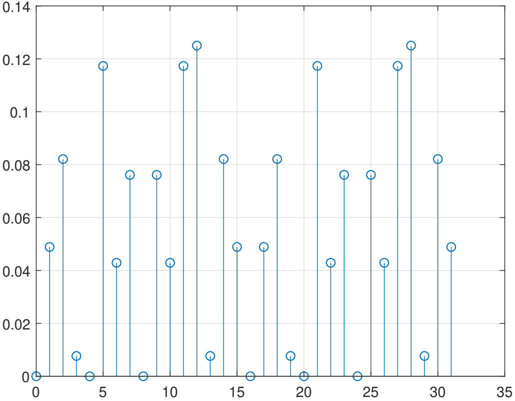
A 32, B 16, C 64, C 8



TopHat Question 9: Fundamental Period of Quantization Noise 2

The figure shows the quantization noise from a quantized sinusoidal signal. What is the fundamental period of the quantization noise?

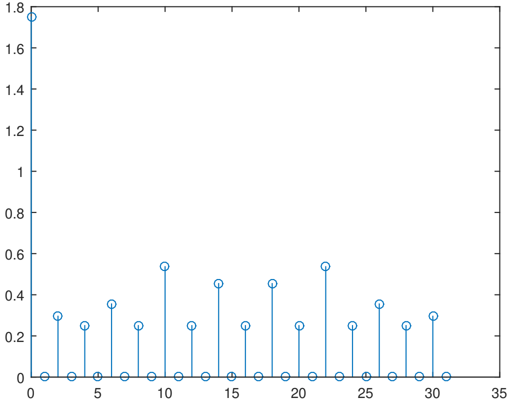
A 32, B 16, C 64, C 8



TopHat Question 10: Fundamental Frequency of x[n]

The figure shows the fft of the quantization noise from a quantized sinusoidal signal, where the original sinusoid signal is $x[n]$, and the fft is a multiple of the fundamental period. What is the fundamental frequency of $x[n]$?

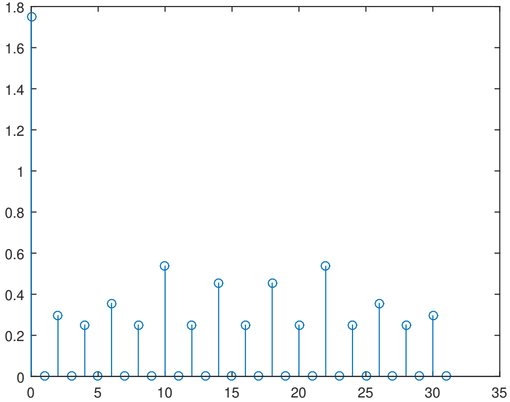
A $1/32$, B $1/16$, C $1/8$, C Cannot determine



TopHat Question 11: Type of Periodic Quantization Noise

The figure shows the fft of the quantization noise from a quantized sinusoidal signal, where the fft is a multiple of the fundamental period. Was this quantization noise produce by:

A Rounding, B Truncation



Section 4

FIR Filter Design

Subsection 1

Window FIR Design Introduction

Introduction

- Recall the section on the impulse response truncation (IRT) technique, which is used to design FIR filters.
- This technique had a problem, which is related to Gibbs oscillations. These oscillations limited the stopband attenuation to 21 dB and the passband ripple to 0.75 dB.
- These values can be improved upon using a technique involving windows.
- The window approach will be introduced using the IRT technique, since the IRT technique is a windows design approach that uses a rectangular window.
- But first, the Gibbs phenomenon will be reviewed in a MATLAB demonstration.

Example 25: Gibbs Oscillations

This demo generates and plots the amplitude response for an impulse response truncation technique lowpass filter with cutoff frequency, $f_c = 0.2$ cycles/sample, for filter orders $M = 20, 40, 80$ and 160, as subplots on a figure window. It also plots the magnitude of the amplitude response squared (which is equivalent to the squared magnitude response) in dB on a separate figure window.

- See gibb.m

Rectangular Window

- First define the following notation:
  - $h_{ideal}[n]$ - not shifted and infinite, where
$$A_{ideal}(e^{j\omega}) = H_{ideal}(e^{j\omega})$$
  - $h_d[n]$ - shifted and infinite, where
$$H_d(e^{j\omega}) = A_{ideal}(e^{j\omega})e^{j(\beta-\alpha\omega)}$$
  - $h[n]$ - shifted and truncated, where
$$H(e^{j\omega}) = A(e^{j\omega})e^{j(\beta-\alpha\omega)}$$

and
$$A(e^{j\omega}) = A_{ideal}(e^{j\omega}) \circledast A_W(e^{j\omega})$$

Note: $A_W(e^{j\omega})$ is the amplitude response from the DTFT of window $w[n]$ and $A_d(e^{j\omega}) = A_{ideal}(e^{j\omega})$

Rectangular Window

- The impulse response of the ideal filter is

$$h_{ideal}[n] = IDTFT\{A_{ideal}(e^{j\omega})\} = \frac{1}{2\pi}\int_{-\pi}^{\pi} A_{ideal}(e^{j\omega})e^{j\omega n}d\omega$$

- For this ideal filter
  1. $h_{ideal}[n]$ is of infinite duration,
  2. $h_{ideal}[n]$ is noncausal
- The impulse response truncation method truncates $h_{ideal}[n]$ to a finite length and shifts the result to give a causal filter.
- Take these two operations and reverse them,

$$h_d[n] = h_{ideal}[n-\alpha];\ \text{(shift)}$$

$$h_[n] = h_d[n]w[n]\ \text{(truncate)}$$

where, for a linear phase filter, $\alpha = M/2$, M is the order of the filter, and $w[n]$ is a rectangular function, referred to as a rectangular window.

Rectangular Window

- The rectangular window is defined as

$$w[n] = \begin{cases} 1; & 0 \leq n \leq M \\ 0; & \text{otherwise} \end{cases}$$

Taking the DTFT of $h[n]$, using the modulation property,

$$H(e^{j\omega}) = \frac{1}{2\pi}(A_{ideal}(e^{j\omega}) \circledast W(e^{j\omega}))(e^{-j(\beta-\alpha\omega)})$$

Thus

$$A(e^{j\omega}) = A_{ideal}(e^{j\omega}) \circledast A_W(e^{j\omega})$$

and

$$W(e^{j\omega}) = DTFT\{w[n]\} = A_W(e^{j\omega})e^{j\theta_{A_W}(\omega)} = \underbrace{\frac{\sin(\frac{\omega(M+1)}{2})}{\sin(\omega/2)}}_{\text{diric in MATLAB}} e^{-j\omega M/2}$$

Example 26: Rectangular Window FIR LPF

Demonstration of rectangular window in FIR lowpass filter design.

- See irt.m

Subsection 2

Window FIR Design

Introduction

- The purpose of windowing is to reduce the ripple in the passband and stopband.
- The reduction in ripple is at the expense of a wider transition band.
- There are a number of well known windows that can be selected. The correct choice depends on the application.

FIR Window Design Method

1. Select a filter type.
2. Specify $A_d(e^{j\omega})$ for the chosen filter type.
3. Determine the impulse response that minimizes the integral of the squared error of the magnitude response for a filter of length $M+1$ (impulse response truncation technique).
4. Multiply the impulse response with a window that is symmetric $w[n] = w[M-n]$.

Windowing in the Frequency Domain

- Windowing can be explained using the DTFT modulation (multiplication) property:

$$x_1[n]x_2[n] \xrightarrow{\text{DTFT}} \frac{1}{2\pi} X_1(e^{j\omega}) \circledast X_2(e^{j\omega})$$

  where $\circledast$ represents circular (periodic) convolution.
- Let $h[n] = h_d[n]w[n]$, then

$$H(e^{j\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\theta})W(e^{j(\omega-\theta)})d\theta = \frac{1}{2\pi}\int_{-\pi}^{\pi} A_d(e^{j\theta})e^{j(\beta-\alpha\theta)}A_W(e^{j(\omega-\theta)})e^{-j\alpha(\omega-\theta)}d\theta$$

  and simplifying

$$H(e^{j\omega}) = A(e^{j\omega})e^{j\theta_A} = \underbrace{\frac{1}{2\pi}\int_{-\pi}^{\pi} A_d(e^{j\theta})A_W(e^{j(\omega-\theta)})d\theta}_{\text{Circular Convolution}}\, e^{j(\beta-\alpha\omega)}$$

  where $\alpha = M/2$, and this is valid for M even or odd.

Properties of Window Functions

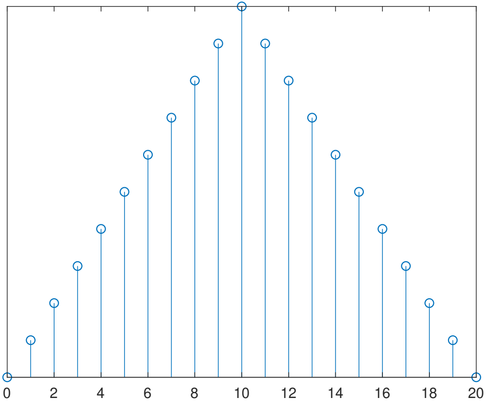
1. $w[n] = 0$ for $n < 0$ and $n > M$.
2. $w[n]$ is symmetric, $w[n] = w[M-n]$.
3. $A_W(e^{j\omega}) = A_W(e^{-j\omega})$, ie. even about $\omega = 0$.
4. $A_W(e^{j(\pi+\omega)}) = A_W(e^{-j(\pi-\omega)})$ for M even; ie. even about π, periodic with a period of 2π.
5. $A_W(e^{j(\pi+\omega)}) = -A_W(e^{-j(\pi-\omega)})$ for M odd; ie. odd about π, periodic with a period of 4π.
6. The window, $w[n]$, is chosen, such that

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} A_W(e^{j\omega})d\omega = w[M/2] = 1 \text{ for } M \text{ even, or}$$

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} A_W(e^{j\omega})d\omega = 1 \text{ for } M \text{ odd}$$

TopHat Question 12: Peak Value of a Window Function

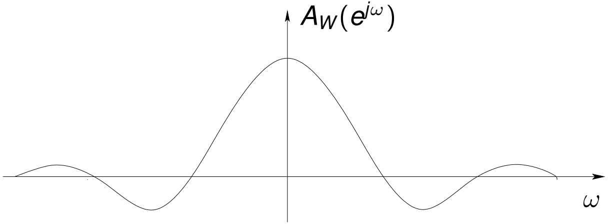
What is the peak value of the $M = 20$ Bartlett window function shown in the figure?



A 20, B $1/20$, C 1,

Form of Window Amplitude Response

- $A_W(e^{j\omega})$, for most windows, has the form



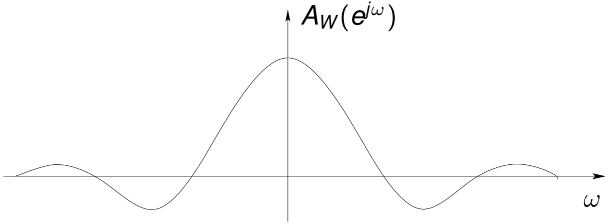
- The area under the lobes define the ripple in the passband and stopband, though the area under the lobes does not depend on M.
- Also, the width of the lobes are proportional to $1/M$ and the height of the lobes are proportional to $M+1$.

Example 27: Rectangular Window Parameters

For a rectangular window of length $M+1$, determine from $W(e^{j\omega})$, the height of the main lobe, the locations of the nulls between lobes and the area of the main lobe.

Example 28: Relationship Between Window Lobe Area and Ripple

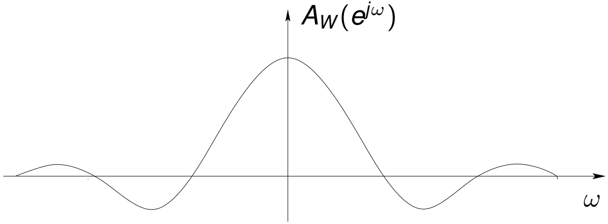
Given the window amplitude response, $A_W(e^{j\omega})$, where the area of the main lobe is 2.4π radians, the area of the adjacent negative lobe is -0.3π radians, and the area of the outside lobe is 0.1π radians,



Determine the total area under $A_W(e^{j\omega})$ and find the values of the points $\frac{2\pi k}{M+1}$ from the edge of the passband, for a $M = 23$ lowpass filter with $\omega_c = \pi/2$, using circular convolution. What is the peak ripple, δ.

TopHat Question 13: Determine the Peak Ripple, δ

What is the peak ripple, δ, for an $M = 40$ lowpass filter designed with a window that has an amplitude response shown in the figure and a mainlobe area of 1.1.



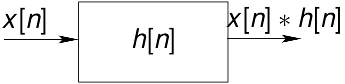
A 0.1, B .05, C 0.025, C 0.2

Units for the Area of the Amplitude Response

- The units of$A(e^{j\omega})$ can be determined from the frequency response of the system, which is

$$H(e^{j\omega}) = \sum_{n=0}^{M} h[n]e^{-j\omega} = A(e^{j\omega})e^{-j\omega M/2}; \text{ for a Type 1 or 2ystem}$$

- Start by determining the units of $H(e^{j\omega})$, referred to as $\mathrm{H}_{\mathrm{units}}$.
- $H(e^{j\omega})$ is the IDTFT of the impulse response, $h[n]$, of a system. $h[n]$ is unitless.



Units for the Area of the Amplitude Response

- Take the IDTFT,

$$h[n] = \underbrace{\underbrace{\frac{1}{2\pi}}_{1/\text{radians}/\text{cycle}} \underbrace{\int_{-\pi}^{\pi} \overbrace{H(e^{j\omega})}^{\text{H}_{\text{units}}} e^{j\omega} d\omega}_{\text{H}_{\text{units}}*\text{radians}/\text{sample}}}_{\text{H}_{\text{units}}*\text{cycles}/\text{sample}}$$

- Thus $\text{H}_{\text{units}} = \text{samples}$, since $h[n]$ is unitless, and since $H(e^{j\omega})$ has units samples, $A(e^{j\omega})$, must also have units samples.
- Finally, the area under $A(e^{j\omega})$ (converted to cycles) is

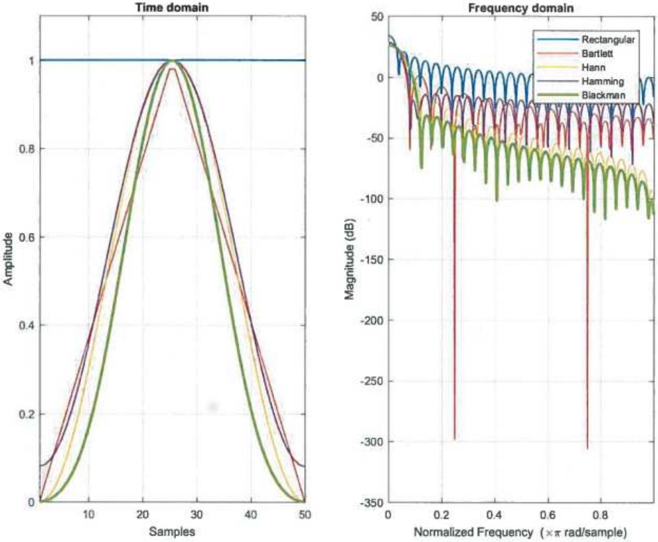
$$\underbrace{\underbrace{\frac{1}{2\pi}}_{1/\text{radians}/\text{cycle}} \int_{-\pi}^{\pi} \underbrace{A(e^{j\omega})}_{\text{samples}} \underbrace{d\omega}_{\text{radians}/\text{sample}}}_{\text{cycles}}$$

- Thus the area of $A(e^{j\omega})$ is unitless.

Window Functions

- Windows are used in digital filter design to reduce the effects of the Gibbs phenomenon.
- There are a large number of possible window functions. Initially five will be consider in detail (the same ones that Oppenheim and Schafer focus on): rectangular, Bartlett, Hann, Hamming and Blackman.
- The windows can be examined using the MATLAB app, windowDesigner, (wintool is used in older versions of MATLAB).
- Using windowDesigner, plot the time domain and frequency domain representations of the five windows, with $M = 49$.

windowDesigner Plots for $M = 49$



MATLAB windowDesigner

- There are a number of parameters that can be viewed or set in windowDesigner:
  - Leakage factor - ratio of the power in the side lobes to the total window power.
  - Relative side lobe attenuation - the maximum side lobe amplitude relative to the maximum main lobe amplitude.
  - Sampling - Symmetric vs periodic reflects how the window is calculated. Use symmetric for filter design. Periodic is used for spectral analysis.

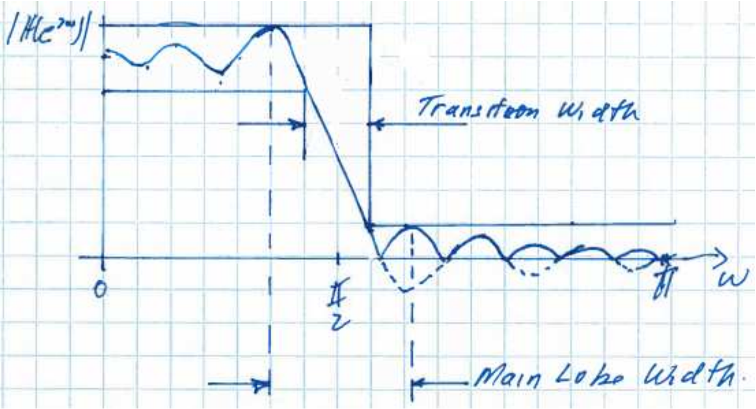
Window Specifications

The specifications for the five windows are summarized in this table (similar to Table 7.2 in Oppenheim and Schafer, 3rd Ed.):

Window	**Relative Side Lobe Attenuation**	**$\approx$Main Lobe Width**	A_s **(dB)**	β **(Equiv. Kaiser Window)**	$(\omega_s - \omega_p)$ **Transition Width (Equiv. Kaiser Window)**
Rectangular	-13 dB	$4\pi/(M+1)$	21 dB	0	$1.81\pi/M$
Bartlett	-25 dB	$8\pi/M$	25 dB	1.33	$2.37\pi/M$
Hann	-31 dB	$8\pi/M$	44 dB	3.86	$5.01\pi/M$
Hamming	-41 dB	$8\pi/M$	53 dB	4.86	$6.27\pi/M$
Blackman	-57 dB	$12\pi/M$	74 dB	7.04	$9.19\pi/M$

Note: In the above table, the transition widths are specified in terms of an equivalent Kaiser window. For a given shape factor, β, the Kaiser window is approximately the same as the given window, so the Kaiser window transition width is used for the given window.

Mainlobe Width vs Transition Width



Note: The cutoff frequency is $\omega_c = \pi/2$, this is defined as the half gain point, or in terms of the magnitude squared, the 6 dB point (not the 3 dB point). This notation is used in MATLAB and Oppenheim and Schafer.

Window Definitions

- **Rectangular**

$$w[n] = \begin{cases} 1; & 0 \leq n \leq M \\ 0; & \text{otherwise} \end{cases}$$

- **Bartlett**

$$w[n] = \begin{cases} \frac{2n}{M}; & 0 \leq n \leq \frac{M}{2} \\ 2 - \frac{2n}{M}; & \frac{M}{2} \leq n \leq M \\ 0; & \text{otherwise} \end{cases}$$

The Bartlett window is a triangular function for even order and a triangular function with a flat top for odd order.

- **Hann(Hanning), Hamming, Blackman (generalized cosine windows)**

$$w[m] = a_1 - a_2 \cos\left(\frac{2\pi n}{M}\right) - a_3 \cos\left(\frac{4\pi n}{M}\right) \quad 0 \leq n \leq M$$

where for Hann, $a_1 = 0.5, a_2 = 0.5, a_3 = 0$, for Hamming, $a_1 = 0.54, a_2 = 0.46, a_3 = 0$, and for Blackman, $a_1 = 0.42, a_2 = 0.5, a_3 = 0.08$.

Example 29: Lowpass Filter Design Using the Window Method

Design a lowpass filter using the window method, where the

- passband gain is 1,
- peak ripple is 0.01 (passband and stopband),
- passband corner frequency is $\omega_p = \pi/8$,
- stopband corner frequency is $\omega_s = \pi/4$.

See lpf_hann_example.m

TopHat Question 14: Filter Cutoff Frequency

A lowpass filter has a transition width of $f_s - f_p = 1/16$, where the passband corner frequency is $f_p = 1/16$ and the stopband corner frequency is $f_s = 1/8$. What is the cutoff frequency for this filter?

$$\text{A } 1/16, \quad \text{B } 1/32, \quad \text{C } 3/32, \quad \text{C } 5/64$$

Example 30: Highpass Filter with Odd Order

Design a HPF, with odd order M and a phase response, $\theta(\omega) = -\omega M/2$, using a windows design. The filter specifications are

$$\omega_p = \frac{\pi}{2}, \omega_s = \frac{\pi}{4}, \delta = \delta_p = \delta_s \leq 0.005$$

Note that this is not a practical filter, since it is a Type 2 filter. A Type 2 filter has a frequency response that is odd about π and must have $H(e^{j\pi}) = 0$.

- See hpf_odd_order_example.m

TopHat Question 15: Filter Gain at Cutoff Frequency

An ideal lowpass filter has a cutoff frequency of $f_c = 1/8$. If the amplitude response of the ideal LPF is convolved with the amplitude response of an even symmetric window to produce the windowed filter amplitude response $A(e^{j\omega})$, what is the gain of the windowed filter at $f_c = 1/8$?

$$\text{A } 1/2, \quad \text{B } 1, \quad \text{C } 0.707\ (1/\sqrt{2}), \quad \text{C } 1/4$$

Filter Design Using a Kaiser Window

- A Kaiser window is defined as

$$w[n] = \begin{cases} I_0(\beta\sqrt{1-(\frac{2n}{M}-1)^2}); & 0 \le n \le M \\ 0; & \text{otherwise} \end{cases}$$

  where $I_0(\cdot)$ is the modified Bessel function of the first kind.
- The Kaiser window parameter, β, controls the shape of the window.
- For example, if $\beta = 0$, the Kaiser window is rectangular. If $\beta = 4.86$, the Kaiser window is very similar to the Hamming window.
- Kaiser, the namesake of this window, developed expressions for β and M.

$$\beta = \begin{cases} 0.1102(A-8.7);; & A > 50 \\ 0.5842(A-21)^{0.4} + 0.07886(A-21);; & 21 \le A \le 50 \\ 0; & A < 21 \end{cases}$$

$$M = \frac{A-8}{2.285(\omega_s - \omega_p)}$$

  where $A = -20\log_{10}\delta$.

Example 31: Lowpass Filter Design Using a Kaiser Window

Design a LPF using a Kaiser window, given $\delta = 0.001, \omega_p = 0.4\pi$ and $\omega_s = 0.5\pi$.

- See kaiser_example.m for the MATLAB implementation.

Example 32: Design a Type 1 HPF

Design a Type 1 highpass filter with $\delta_s = 0.0002, \delta_p = 0.001, \omega_p = 0.8\pi$ and $\omega_s = 0.7\pi$, using a windows design and a Kaiser design.

- See blackman_kaiser_example.m for the MATLAB implementation.

Filter Design if $G_p \neq 1$ and/or $G_s \neq 0$

- In a filter design, if $G_p \neq 1$ and/or $G_s \neq 0$, then the ripple specification must be modified to account for G_p or G_s changes before the filter is designed.
- For example, if $G_p = 2$, then if the ripple specification is not modified, the ripple will be larger in the final design by a factor of 2. To account for the increased gain the ripple would have to be scaled by a factor of 2.

Scaling the Ripple

- If the desired passband gain is G_p and the desired stopband gain is G_s, then the value for δ is

$$\delta = \frac{\delta_{\text{desired}}}{G_p - G_s}$$

where δ_{desired} is the desired peak ripple for the filter and δ is the normalized peak ripple.

Example 33: Lowpass Filter with a Gain of 2

Design a lowpass filter using a Kaiser window, with $G_p = 2, G_s = 0, \delta = 0.01, \omega_p = 0.3\pi$ and $\omega_s = 0.4\pi$.

- See lowpass_filter_kaiser_example_gp2.m for the MATLAB implementation.

Subsection 3

Least Squares Design of FIR Filters

Introduction

- One of the major disadvantages of the window method for FIR filter design is that δ_s and δ_p must be equal.
- Though, often the specification of δ_s is more stringent that δ_p (ie. $\delta_s < \delta_p$), resulting in unnecessarily high accuracy in the passband. This results in a higher order filter than is actually needed.
- In the least squares design technique, δ_s and δ_p can be indirectly specified separately.

Least Squares FIR Design

- The least squares design involves choosing the coefficients that minimize the integral of the weighted square frequency response error,

$$\epsilon^2 = \int_0^{\pi} E^2(\omega)d\omega$$

where

$$E(\omega) = W_t(\omega)(A_d(e^{j\omega}) - A(e^{j\omega}))$$

and $W_t(\omega)$ is a weight function, which reflects the relative importance of the error between $A_d(e^{j\omega})$ and $A(e^{j\omega})$ at a given ω.

- For example, for a weight function for a lowpass filter, one possibility is to assign δ_p^{-1} to all frequencies in the passband and assign δ_s^{-1} to all frequencies in the stopband and a zero weight for the transition band.

Least Squares FIR Design

- In the least squares filter design, the order M of the filter is assumed known and so is the type of linear phase filter.
- For example, for the type 1 filter, from slide 61,

$$A(e^{j\omega}) = \sum_{k=0}^{\frac{M}{2}} a[k] \cos \omega k$$

where

$$a[k] = \begin{cases} h[\frac{M}{2}]; & k = 0, \\ 2h[\frac{M}{2} - k]; & k = 1, 2, \cdots, \frac{M}{2} \end{cases}$$

Thus

$$E(\omega) = W_t(\omega)(A_d(e^{j\omega}) - \sum_{k=0}^{\frac{M}{2}} a[k] \cos \omega k)$$

- Designing the FIR filter involves determining the coefficients $a[k]$ (ie. the filter impulse response coefficients) that minimizes the expression for ϵ^2.

Least Squares FIR Design in MATLAB

The basic form of the least squares filter design function, from the MATLAB documentation, is `b = firls(n,f,a,w)`, Using the notation in this class or notation that does not conflict with this class, this function can be written as

```
b = firls(M,fb,a,wght)
```

where

- `b` is a vector containing the `M+1` calculated impulse response coefficients.
- `M` is the order of the desired FIR filter
- `fb` is a vector of the corner frequencies of the stop/pass bands. `fb` has two enteries for each band. The band frequencies specfied for ω from 0 to π. The units of `fb` are radians/sample divided by π (or cycles/sample multiplied by 2). This results in `fb` having values between 0 and 1 (this is because MATLAB normalizes the analog frequencies by $\frac{F_s}{2}$ instead of F_s), and the first element must be a 0 and the last element must be a 1.
- For example, for a pass band from 0 to 0.2π and a stop band from 0.5π to π, the band frequency vector is `fb = [0, 0.2, 0.5, 1];`.

Least Squares FIR Design in MATLAB

Continuing the definition of

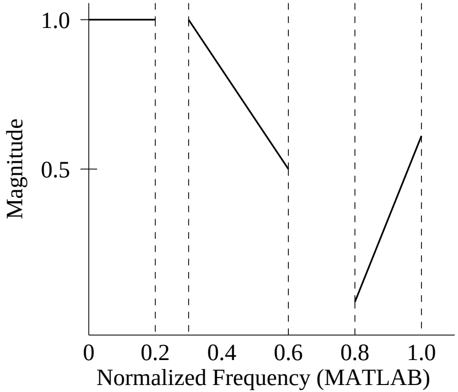
```
b = firls(M,fb,a,wght)
```

- `a` is a vector specifying the desired gain at the frequency points defined by `fb`. The gain does ont have to be flat for each band. The gain is specified at the corners of each band and the gain across the band is a straight line joining the gains at the corners. `fb` and `a` must have the same length and this length must be an even number.
- For example, `a = [1, 0.8, 0, 0];`, results in the first band gain starting at 1 and ending at 0.8 and the gain of the second band being zero.
- `wght` specifies the relative ripple in the bands. It is a vector having one element for each band, thus it is half the length of `fb`. `wght` weights the error.
- For example, a relative weight of 10 will cause the ripple to be 1/10 that of the reference band (`wght = [1, 10];`) Or you can assign the inverse of the peak ripple desired for each band, for example, for a lowpass filter `wght=[1/deltap, 1/deltas]`.

Least Squares FIR Design in MATLAB

For example, the desired frequency magnitude response is given in the figure for the vectors

```
fb = [0, 0.2, 0.3, 0.6, 0.8, 1.0];
a = [1, 1.0, 1.0, 0.5, 0.1, 0.6];
wght = [1/delta1, 1/delta2, 1/delta3];
```



Example 34: Least Squares FIR Filter Design

Design a lowpass filter using a least squares design and compare it with a Kaiser window design. The filter specifications are $\delta_p = 0.1, \delta_s = 0.01, \omega_p = 0.2\pi$ and $\omega_s = 0.3\pi$.

- Determining the order of the filter is a trial an error process involving selecting the order to obtain the desired stop band attuenuation. The resulting chosen filter has an order of M=32.
- See firls_design_example.m for the MATLAB implementation.
- Note: The least squares designed filter is compared with a filter designed using a Kaiser window. The Kaiser filter has an order of M = 45. This is a significant difference in length of filters, which primarily results from allowing the pass band ripple to be different and larger than the stop band ripple.

Subsection 4

Equiripple (Parks-McClellan) Design of FIR Filters

Introduction

- The least squares design approach focused on minimizing the integral of the square of the weighted frequency response error over all frequency domain bands.
- The equiripple design approach uses a better criterion, which involves the minimization of the maximum error in each band.
- This criterion results in an equiripple filter, one whose amplitude response oscillates uniformly between the ripple bounds for each band.
- This technique is optimum in the sense of minimizing the maximum magnitude of the ripple in each of the bands for a given filter order. The bands between transitions are not considered in the optimization.
- The equiripple design approach is also referred to as the Parks-McClellan approach or the optimum approximation method.
- This type of filter is also called a minmax filter, since the algorithm tries to minimize the maximum error between the desired and actual frequency responses.

Equiripple (Parks-McClellan) Filter Design

- Similar to the least squares technique, the weighted frequence response error is defined as

$$E(\omega) = W_t(\omega)(A_d(e^{j\omega}) - A(e^{j\omega}))$$

- The goal of the equiripple design algorithm involves determining the impulse response that minimizes the maximum absolute weighted error $|E(\omega)|$ of the entire set of bands.
- This can be expressed as minimizing ϵ over all coefficients of the impulse response, where

$$\epsilon = \max_{\text{all bands}} |E(\omega)|$$

Equiripple (Parks-McClellan) Filter Design

- For example, for a type 1 filter, the actual frequency response (from slide xx) is

$$A(e^{j\omega}) = \sum_{k=0}^{\frac{M}{2}} a[k] \cos \omega k$$

where

$$a[k] = \begin{cases} h[\frac{M}{2}]; & k = 0, \\ 2h[\frac{M}{2} - k]; & k = 1, 2, \cdots, \frac{M}{2} \end{cases}$$

to give

$$E(\omega) = W_t(\omega)(A_d(e^{j\omega}) - \sum_{k=0}^{\frac{M}{2}} a[k] \cos \omega k)$$

- Thus, designing the equipple FIR filter involves determining the coefficients, $a[k]$ (ie. the filter impulse response coefficients) that minimizes ϵ.
- An advantage of this approach, similar to the least squares design, is that the pass band ripple does not have to be the same as the stop band ripple.

Estimating the Filter Order for the Parks-McClellan Method

The order of the filter is needed to use this technique and the order can be estimated using emperically derived expressions developed by Bellanger, Kaiser and Harris. The approximation derived by Bellanger is

$$M \approx \frac{2}{3}\left(\log_{10}\left(\frac{1}{10\delta_p\delta_s}\right)\right)\left(\frac{2\pi}{|\omega_w - \omega_p|}\right)$$

The Kaiser approximation is

$$M \approx \frac{-20\log_{10}(\sqrt{\delta_p\delta_s}) - 13}{2.324|\omega_s - \omega_p|}$$

The Harris approximation is

$$M \approx \frac{-20\log_{10}(\delta_s)}{22|f_s - f_p|}$$

Equiripple (Parks-McClellan) Design in MATLAB

- The equiripple design algorithm is implemented in MATLAB using the `firpm` function, where `pm` stands for Parks-McClellan.
- The `firpm` function implements the Parks-McClellan algorithm, which uses Chebyshev approximation theory to develop a transfer function with equiripple passband and stopband magnitude responses.
- The basic form of the function from the MATLAB documentation is `b = firpm(n,f,a,w)`, Using the notation in this class or notation that does not conflict with this class, this function can be written as `b = firpm(M,fb,a,wght)`.
- The variables in `b = firpm(M,fb,a,wght)` are the same as those used for the least squares method as describe in slides 178 to 180.

`firpmord` Generates the Filter Order and Parameters

- One of the inputs for `firpm` is the filter order `M`. The order can be estimated using Bellanger's, Kaiser's or Harris's equation specified earlier. Though, there is also a MATLAB function that can be used to estimate the order for the equiripple filter, along with the parameters for `firpm`.
- MATLAB's basic form of the function is `[n,fo,ao,w] = firpmord(f,a,dev,fs)`.
- Using the notation in this class (or notation that does not conflict with this class), this function is written as

```
[Mf,fo,ao,wo]=firpmord(ford,aord,dev,Fsord)
```

  where
  - `ford` is `fb` with the first and last elements removed. `firpmord` assumes they are 0 and 1.
  - `aord` is a vector giving the amplitudes of each band in `ford`. It has a length of one half of the (length of `ford` - 2).
  - `dev` is the peak ripple in linear units. It is a vector with one element for each frequency band. It has the same number of elements as `oard`.
  - `Fsord` is the sampling frequency in Hz. If not included it has a default value of 2. Though, if specified as 1 the `ford` can use the non-MATLAB normalized frequency notation. The actual sampling frequency can also be specified.

Example 35: LPF Design Using the Equiripple (Parks-McClellan) Method

Design a lowpass filter using a equiripple design. The filter specifications are $\delta_p = 0.02, \delta_s = 0.01, \omega_p = \pi/4$ and $\omega_s = 3\pi/8$, $G_p = 1$, $G_s = 0$.
Divide the example into two parts:

1. Generate the filter order using 4 techniques: Bellanger's, Kaiser's and Harris's equations and `firpmord`. Select the highest order and then design the filter by specifying `f,a` and `wght` and then using `firpm`.
2. Design the filter by first using `firpmord` to generate `f,a,wght` and the filter order `M`, then use `firpm`.

- See lpf_equiripple_example.m for the MATLAB implementation.

Example 36: Multiband Filter Design Using the Equiripple (Parks-McClellan) Method

Design a multiband filter using a equiripple design. The filter specifications are
$\omega_{p1} = \pi/4, \omega_{s1} = 3\pi/8, \omega_{s2} = 5\pi/8, \omega_{p2} = 11\pi/16, \delta_{p1} = 0.02, \delta_s = 0.01,$
$\delta_{p2} = 0.02, G_{p1} = 1, G_s = 0, G_{p2} = 0.3.$
Divide the example into two parts:

1. Generate the filter order using 4 techniques: Bellanger's, Kaiser's and Harris's equations and `firpmord`. Select the highest order and then design the filter by specifying `f,a` and `wght` and then using `firpm`.
2. Design the filter by first using `firpmord` to generate `f,a,wght` and the filter order `M`, then use `firpm`.

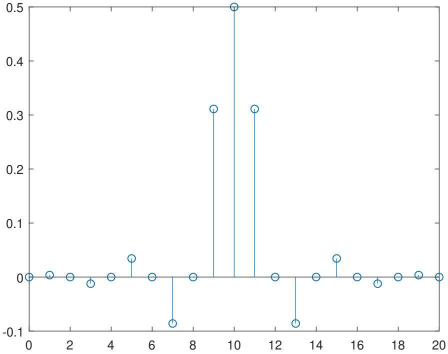
- See multiband_equiripple_example.m for the MATLAB implementation.

Subsection 5

FIR Nyquist Filter Design

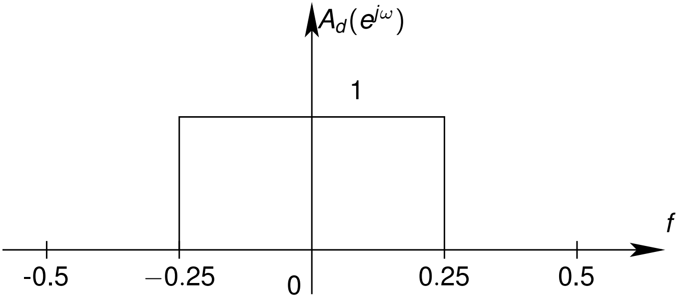
Introduction

- Nyquist filters (also called K^{th} band filters) have periodic zero values for every K^{th} coefficient, except for the center coefficient.
- These filters are used in multirate applications, such as as interpolation and decimation.
- They also find applications in pulse shaping filters for communications systems.
- An example of the impulse response of a K^{th} band filter, with $K = 2$ is



Halfband Filters

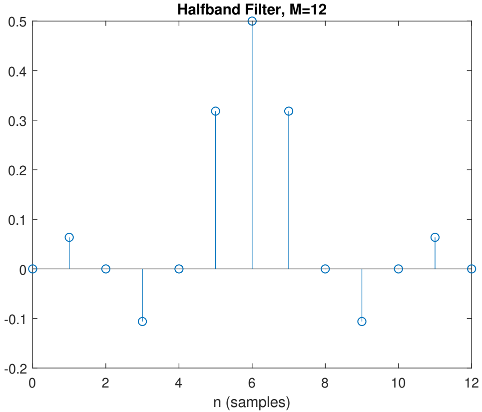
- A halfband filter is a K^{th} band filter where $K = 2$. Every second coefficient in these filters is zero, making for a more efficient implementation.
- But, note that every second coefficient will be zero only if the order of the halfband filter is even.
- An ideal halfband filter is a lowpass filter with a cutoff frequency of $\omega_c = \pi/2$ ($f_c = 1/4$).



Basic Halfband Filter

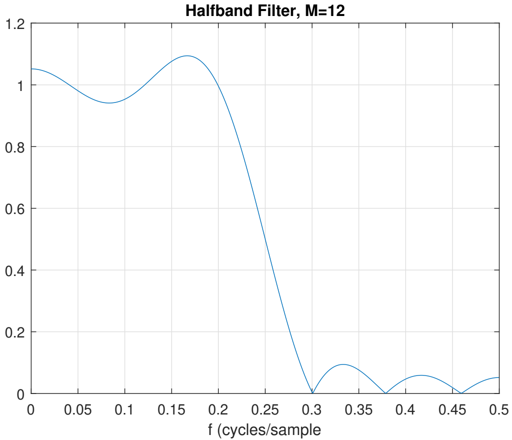
- A basic halfband filter can be designed using the IRT technique (rectangular window).
- The impulse response for such a filter is

$$h[n] = 2f_c \mathrm{sinc}(2f_c(n - M/2)) = 0.5\mathrm{sinc}(0.5(n - M/2)) \quad n = 0, 1, \cdots, M$$



Basic Halfband Filter Frequency Response

- The frequency response for the basic halfband filter of the previous slide is shown below.
- It has a cutoff frequency of 0.25 cycles/sample, as indicated by the half gain location.



Example 37: Window Design of a Halfband Filter

Design a halfband filter using a Kaiser window, with a ripple of 0.001 and a transition width of 0.1 cycles/sample.

- See halfband_kaiser.m for the MATLAB implementation.

Example 38: Parks-McClellan Design of a Halfband Filter

Design a halfband filter using the Parks-McClellan method, with a ripple of 0.001 and a transition width of 0.1 cycles/sample.

- See halfband_equiripple.m for the MATLAB implementation.

Ideal Continuous-Time Raised Cosine

- Raised cosine filters are Nyquist (K^{th} band) filters. Every K^{th} coefficient of these filters is zero.
- They are used extensively in communications as pulse shaping filters.
- They are called raised cosine because their frequency response has a raised cosine shape.
- In continuous time, the frequency response is

$$H_{rc}(F) = \begin{cases} 1; & 0 \le |F| \le (1-\beta)F_c \\ \frac{1}{2}\left(1 + \cos\left[\frac{\pi}{2\beta}\left(\frac{|F|}{F_c} - (1-\beta)\right)\right]\right); & (1-\beta)F_c \le |F| \le (1+\beta)F_c \\ 0; & |F| \ge (1+\beta)F_c \end{cases}$$

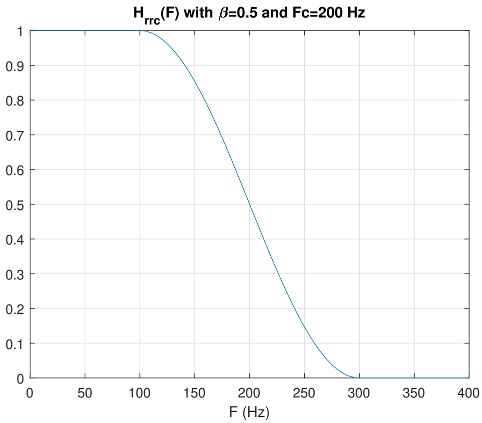
  where the cutoff frequency is $F_c = 1/(2T_s)$, T_s is the symbol time and $0 \le \beta \le 1$ is the roll-off factor.
- In the time domain,

$$h_{rc}(t) = \begin{cases} \frac{\pi}{4T_s}\mathrm{sinc}\left(\frac{1}{2\beta}\right); & t = \pm\frac{T_s}{2\beta} \\ \frac{1}{T_s}\mathrm{sinc}\left(\frac{t}{T_s}\right)\frac{\cos\left(\frac{\pi\beta t}{T_s}\right)}{1-\left(\frac{2\beta t}{T_s}\right)^2}; & \text{otherwise} \end{cases}$$

- Note that the time between sidelobe zero crossing in $h_{rc}(t)$ is T_s.

Ideal Raised Cosine Frequency Response

- The following figure is a plot of the ideal frequency response for a raised cosine pulse. Note the raised cosine has a gain of 1/2 at the cutoff frequency.



Excess Bandwidth

- As indicated in the earlier $H_{rc}(F)$ expression, the absolute bandwidth of a raised cosine pulse is

$$B_{abs} = (1+\beta)F_c = \frac{1+\beta}{2T_s}$$

  where $0 \leq \beta \leq 1$.
- Using this expression, the minimum bandwidth is $1/T_s$, when $\beta = 0$ (spectrum is rectangular), and the maximum bandwidth is $1/T_s$, when $\beta = 1$.
- One convention is to call β the excess bandwidth.
- For example, a 0 % excess bandwidth indicates the raised cosine has the minimum bandwidth allowed, $1/2T_s$, and a 100 % excess bandwidth indicates the raised cosine has the maximum bandwidth allowed, $1/T_s$.

Ideal Discrete-Time Raised Cosine

- The discrete-time version of the raised cosine impulse response can be obtained exactly using the impulse invariance technique, since $H_{rc}(F)$ is bandlimited.
- Using the impulse invariance technique, the discrete-time impulse response is

$$h_{rc}[n] = Th_{rc}(t)|_{t=nT}$$

where T is the sampling time. Thus

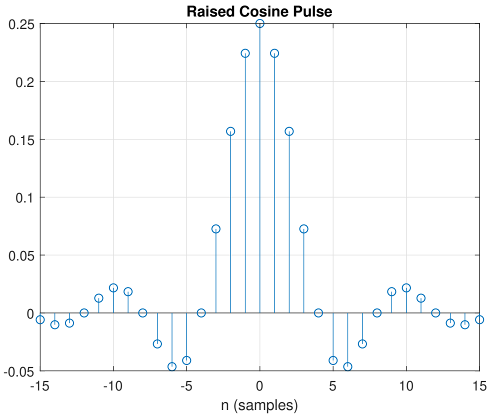
$$h_{rc}[n] = \begin{cases} \frac{\pi T}{4T_s}\mathrm{sinc}\left(\frac{1}{2\beta}\right); & n = \pm\frac{T_s}{2\beta T} \\ \frac{T}{T_s}\mathrm{sinc}\left(\frac{nT}{T_s}\right)\frac{\cos\left(\frac{\pi\beta nT}{T_s}\right)}{1-\left(\frac{2\beta nT}{T_s}\right)^2}; & \text{otherwise} \end{cases}$$

- In communications, it is common for the ratio T_s/T to be defined as $N_{sps} = T_s/T$, where N_{sps} is the number of samples per symbol (typically N_{sps} is 4 or 8). Substituting N_{sps} gives

$$h_{rc}[n] = \begin{cases} \frac{\pi}{4N_{sps}}\mathrm{sinc}\left(\frac{1}{2\beta}\right); & n = \pm\frac{N_{sps}}{2\beta} \\ \frac{1}{N_{sps}}\mathrm{sinc}\left(\frac{n}{N_{sps}}\right)\frac{\cos\left(\frac{\pi\beta n}{N_{sps}}\right)}{1-\left(\frac{2\beta n}{N_{sps}}\right)^2}; & \text{otherwise} \end{cases}$$

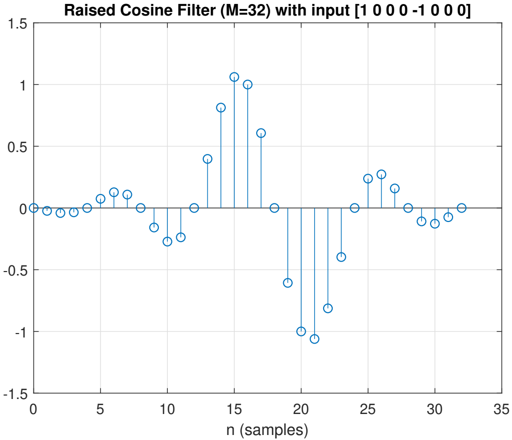
Ideal Discrete-Time Raised Cosine Pulse

- The following figure is a plot of a raised cosine pulse. Here $\beta = 0.25$ and $Nsps = 4$ have been used.



Raised Cosine Filters and Intersymbol Interference

- Raised cosine filters are used in communications to avoid intersymbol interference (ISI).
- For example, a raised cosine filter ($Nsps = 4$) input of [1 0 0 0 -1 0 0 0], produces output



Ideal Continuous-Time Square Root Raised Cosine

- Raised cosine filters are typically split into two filters, one at the transmitter and one at the receiver, ie. $H_{rc}(F) = \sqrt{H_{rc}(F)}\sqrt{H_{rc}(F)}$.
- $H_{rrc}(F) = \sqrt{H_{rc}(F)}$ is called the square root raised cosine filter or the root raised cosine filter.
- In continuous time, the frequency response is

$$H_{rrc}(F) = \begin{cases} 1; & 0 \le |F| \le (1-\beta)F_c \\ \cos\left[\frac{\pi}{4\beta}\left(\frac{|F|}{F_c} - (1-\beta)\right)\right]; & (1-\beta)F_c \le |F| \le (1+\beta)F_c \\ 0; & |F| \ge (1+\beta)F_c \end{cases}$$

where the cutoff frequency is $F_c = 1/(2T_s)$, T_s is the symbol time and $0 \le \beta \le 1$ is the roll-off.
- In the time domain,

$$h_{rrc}(t) = \begin{cases} \frac{1}{T_s} + \frac{\beta}{T_s}\left(\frac{4}{\pi} - 1\right); & t = 0 \\ -\frac{\beta}{T_s}\left[\frac{2}{\pi}\cos\left(\frac{\pi}{4\beta}(1+\beta)\right) - \cos\left(\frac{\pi}{4\beta}(1-\beta)\right)\right]; & t = \pm\frac{T_s}{4\beta} \\ \frac{1}{T_s}\frac{\frac{4\beta t}{T_s}\cos\left(\frac{\pi(1+\beta)t}{T_s}\right)+\sin\left(\pi(1-\beta)\frac{t}{T_s}\right)}{\frac{\pi t}{T_s}\left[1-\left(\frac{4\beta t}{T_s}\right)^2\right]}; & \text{otherwise} \end{cases}$$

Practical Square Root Raised Cosine

- The square root raised cosine pulse, defined on the previous slide, is of infinite length, thus it's frequency response has a finite length, with an absolute bandwidth of $(1+\beta)/2T_s$.
- In a practical implementation, the pulse is truncated to $-(N_{symb}-1)T_s/2 \le t \le (N_{symb}-1)T_s/2$, where N_{symb} is the number of symbols in the pulse. In MATLAB N_{symb} is typically referred to as the span.
- The practical square root raised cosine pulse has finite length, thus the frequency response of the pulse must have infinite length and this increased length takes the form of sidelobes.
- A larger value for N_{symb} results in a greater attenuation of these sidelobes, but the practical stopband attenuation is limited to about 46 dB, depending on β and the length of the pulse.

Ideal Discrete-Time Square Root Raised Cosine

- The discrete-time version of the square root raised cosine impulse response can be obtained exactly using the impulse invariance technique, since $H_{rrc}(F)$ is bandlimited.
- Using the impulse invariance technique, the discrete-time impulse response is

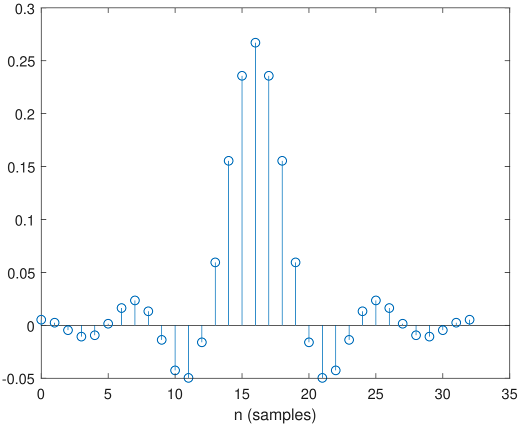
$$h_{rrc}[n] = Th_{rrc}(t)|_{t=nT}$$

where T is the sampling time. Then substituting $N_{sps} = T_s/T$ gives

$$h_{rrc}[n] = \begin{cases} \frac{1}{N_{sps}} + \frac{\beta}{N_{sps}}\left(\frac{4}{\pi} - 1\right); & n = 0 \\ -\frac{\beta}{N_{sps}}\left[\frac{2}{\pi}\cos\left(\frac{\pi}{4\beta}(1+\beta)\right) - \cos\left(\frac{\pi}{4\beta}(1-\beta)\right)\right]; & n = \pm\frac{N_{sps}}{4\beta} \\ \frac{1}{N_{sps}} \frac{\frac{4\beta n}{N_{sps}}\cos\left(\frac{\pi(1+\beta)n}{N_{sps}}\right) + \sin\left(\pi(1-\beta)\frac{n}{N_{sps}}\right)}{\frac{\pi n}{N_{sps}}\left[1 - \left(\frac{4\beta n}{N_{sps}}\right)^2\right]}; & \text{otherwise} \end{cases}$$

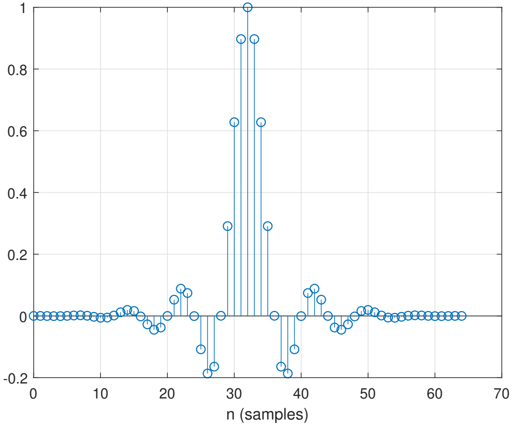
Square Root Raised Cosine Filter Impulse Response

- The following is a plot of a length 33, truncated and shifted raised cosine pulse for $\beta = 0.25$, $N_{sps} = 4$ and $M = 32$. Note that this is not a unit energy pulse.



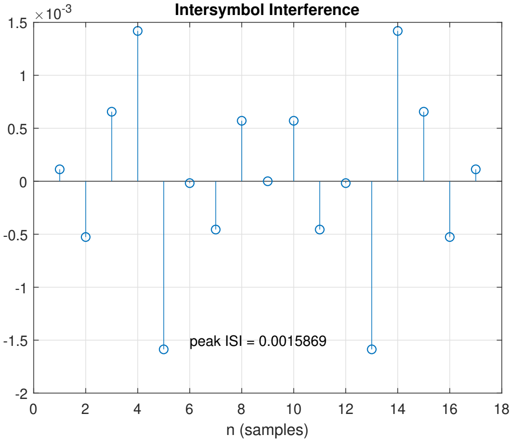
Convolution of Square Root Raised Cosine Filters

- The convolution of two square root raised cosine filters results in a raised cosine response.
- The square root raised cosine pulses that were used had unit energy, this results in a peak value of 1 for the raised cosine response.



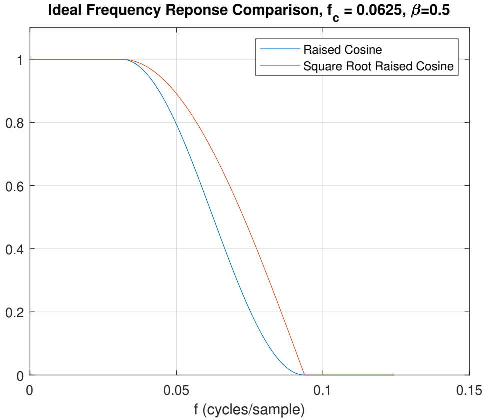
Peak ISI of $h_{rc}[n] = h_{rrc}[n] * h_{rrc}[n]$

- The following is a plot of every $N_{sps} = 4^{th}$ sample of the raised cosine resulting from the convolution $h_{rrc}[n] * h_{rrc}[n]$ (not including the peak value), shown in the previous slide.
- The ISI is a result of the truncation of the ideal square root raised cosine impulse response.



Ideal Frequency Response Comparison

- The following figure is a comparison of the ideal frequency response for a raised cosine and a square root raised cosine filter.
- Note the raised cosine has a gain of 1/2 at the cutoff frequency and the square root raised cosine has a gain of $1/\sqrt{2}$ at f_c.



Transition Width for Ideal Frequency Response

- The transition width for the square root raised cosine can be determined from the ideal frequency response equation on slide 204.
- The transition width is $f_s - f_p$, where the continuous time frequency is normalized by the sampling period T

$$f_p = (1-\beta)F_c T = (1-\beta)f_c$$

and

$$f_s = (1+\beta)F_c T = (1+\beta)f_c$$

where the cutoff frequency, $F_c = 1/(2T_s)$ is scaled by T to give

$$f_c = TF_c = T/(2T_s) = 1/(2N_{sps})$$

- For the figure on the previous page, you can see that both the raised cosine and square root raised cosine have the same f_p and f_s, where $f_p = (1-0.5)0.0625 = 0.03125$ and $f_s = (1+0.5)0.0625 = 0.09375$.

Improving the Stopband Attenuation of the Square Root Raised Cosine Filter

- The square root raised cosine filter has sidelobes that are rather high, approximately 24 to 46 dB below the passband gain depending on β and the length of the filter.
- Typically, the desired out of band attenuation is on the order of 60 to 80 dB.
- Another window could be used to improve the stopband attenuation, but this leads to significant increases in the ISI levels, which would negatively impact a communications system .
- The increase in ISI levels can be traced to the $1/\sqrt{2}$ gain point moving away from f_c.
- The important characteristics of a square root Nyquist filter are the transition bandwidth, which is defined by β and the f_c location of the $1/\sqrt{2}$ gain point.
- One technique that can produce these characteristics is an iterative algorithm that uses the MATLAB firpm function.

Square Root Nyquist Filter Using an Iterative Algorithm

- This algorithm starts with a Parks-McClellan design of a LPF with $f_p = (1-\beta)f_c$ and $f_s = (1+\beta)f_c$, where $f_c = 1/(2N_{sps})$.
- This LPF has a gain of 1/2 at f_c. This gain can be increased to $1/\sqrt{2}$ by increasing the frequency of the passband corner frequency, f_p.
- The algorithm minimizes the error in the gain

$$e = |H(f_c)| - \frac{1}{\sqrt{2}}$$

- The gain is increased using a gradient descent method based on the equations:

$$e(n_i) - \frac{1}{\sqrt{2}} - |H(f_c)|$$

$$f_i(n_i + 1) = f_i(n_i)(1 + \mu e(n_i))$$

where n_i indexes the iteration steps and μ is a positive constant that scales the step size.

MATLAB Iterative Algorithm

- Iterative algorithm using firpm to determine a square root nyquist filter with $A_s > 65$ dB.

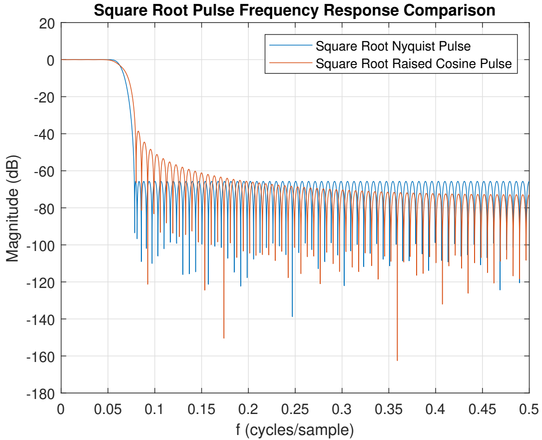
```
beta=0.25; Nsps=8; % number of samples/symbol
Nsymb=18; % number of symbols
M=Nsymb*Nsps; % order of the filter
fc=1/(2*Nsps); % cutoff frequency
fp=(1-beta)*fc; % passband corner frequency
fs=(1+beta)*fc; % stopband corner frequency
f=[0:.0001:.5];
fi=fp; e=1/sqrt(2); % initialize variables for iterations
mu=.01; % step size, controls convergence rate
while abs(e)>.0001
    h=firpm(M,[0 2*fi 2*fs 1],[1 1 0 0],[1 1]);
    [H,w]=freqz(h,1,f*2*pi);
    ifc=find(f==fc); % find the frequency index for fc
    e=1/sqrt(2)-abs(H(ifc));
    fi=fi*(1+mu*e);
end
```

Choosing an Order for the Square Root Nyquist Filter

- Choosing the order for the algorithm on the previous slide is an iterative process.
- One of the order estimation techniques for firpm could be used to choose an initial estimate.
- For example, using Harris's expression
  `M_Harris=(As)/(22*abs(fp-fs))`,
  with $A_s = 65$, gives 94.55. But, since f_p and f_s change as the algorithm converges the required order will likely be much larger.
- In the previous MATLAB code an order of $M = N_{symb} * N_{sps} = 18 * 8 = 144$ is used.

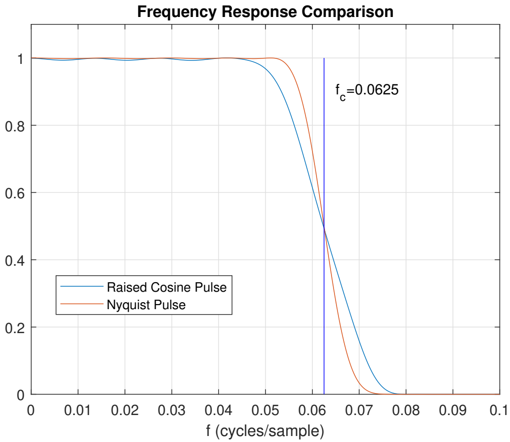
Square Root Pulse Frequency Response Comparison

- The following figure is a comparison of the frequency response for the square root raised cosine pulse with the frequency response for the square root Nyquist pulse determined using the iterative algorithm. The order is the same for both cases.



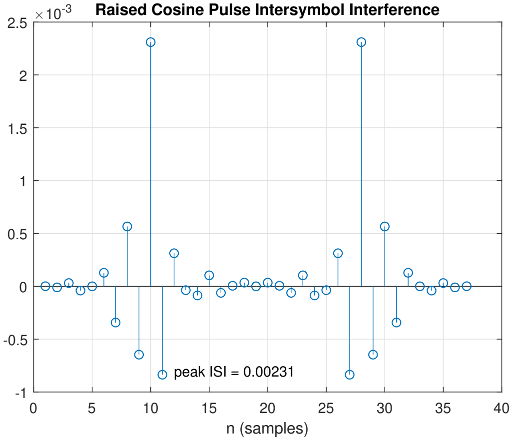
Raised Cosine - Nyquist Pulse Frequency Response Comparison

- The following figure is a comparison of the frequency response for the raised cosine pulse with the frequency response for the Nyquist pulse determined using the iterative algorithm. The order is the same for both cases.



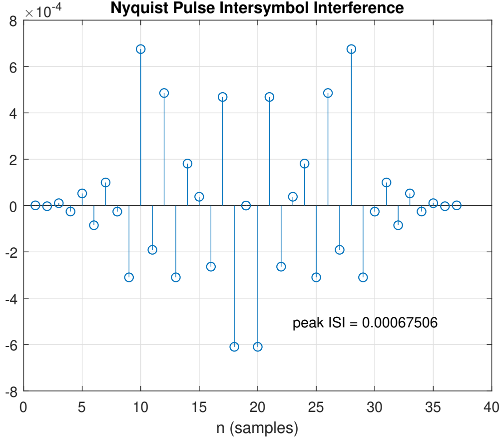
Raised Cosine Pulse Intersymbol Interference

- The following is a plot of every $N_{sps} = 8^{th}$ sample of the raised cosine resulting from the convolution $h_{rrc}[n] * h_{rrc}[n]$ (not including the peak value) ($\beta = 0.25, M = 144$)



Nyquist Raised Cosine Pulse Intersymbol Interference

- The following is a plot of every $N_{sps} = 8^{th}$ sample of the Nyquist pulse resulting from the convolution square root Nyquist pulse with itself (not including the peak value), which was generated using the iterative algorithm.



MATLAB rcosdesign

- Raised cosine and square root raised cosine filters can be designed in MATLAB using `rcosdesign` (it replaces `firrcos`).
- A raised cosine impulse response is generated using `hrc = rcosdesign(beta,Nsymb,Nsps,'normal')`
- A square root raised cosine impulse response is generated using `hrrc = rcosdesign(beta,Nsymb,Nsps)`
- The length of the filter is `Nsymb*Nsps + 1` and thus the order of the filter is $M = N_{symb}N_{sps}$.
- Both `hrc` and `hrrc` are generated as unit energy pulses (ie. $\sum h_{rc}^2 = 1$).
- Thus the nominal passband gain will not be 1, as was defined earlier for the ideal frequency response expressions for the two pulses.
- These pulses need to be scaled to produce a nominal gain of 1.
- If you want to have a filter with a nominal passband gain of 1, what should the scaling factor be?

Scaling the Raised Cosine or Square Root Raised Cosine Pulse

- `rcosdesign` produces a unit energy raised cosine,$h_{rc}[n]$, or square root raised cosine, $h_{rrc}[n]$.
- For a square root raised cosine pulse, the convolution, $h_{rrc}[n] * h_{rrc}[n]$ results in raised cosine pulse with a peak value of 1, since $\sum(h_{rrc}[n] * h_{rrc}[n])^2 = 1$.
- It a nominal passband gain of 1 is desired, then the pulse can be scaled. To produce the same pulse that would have been produced by the truncated ideal pulse, use
  1. For a raised cosine pulse, scale using
  $$h_{rcs} = \frac{h_{rc}[n]}{max(h_{rc}[n])N_{sps}}$$
  2. For a square root raised pulse, scale using
  $$h_{rrcs} = \frac{h_{rrc}[n]}{max(h_{rrc}[n])}\left(\frac{1}{N_{sps}} + \frac{\beta}{N_{sps}}(\frac{4}{\pi} - 1)\right)$$
- You could also scale by the DC value of the pulse, though this will be slightly different than the above technique, because the pulse is truncated. The scaling factor is the same for the raised cosine and square root raised cosine.

$$h_s[n] = \frac{h[n]}{\sum h[n]}$$