# DIGITAL SIGNAL PROCESSING: COSC390

Jordan Hanson

January 17, 2019

Whittier College Department of Physics and Astronomy

# UNIT 2.1 OUTLINE

1. **Introduction:** Types of filters (reading: ch. 3, ch. 5)
    - Butterworth
    - Bessel
    - Chebyshev
2. LTI systems and their properties (reading: ch. 5)
3. Convolution (reading: ch. 7)
    - Implementation with FFT
    - Impulse and step response

Future lectures will cover:

1. SNR of filtered signals: SNR
2. Common filter kernels (moving average, windows)
3. Recursive filters
4. FIR and IIR definitions

# INTRODUCTION: TYPES OF FILTERS

Chapter 3 lists three types of anti-aliasing filters: Butterworth, Bessel, and Chebyshev. Filters are examples of linear, time-invariant (LTI) devices

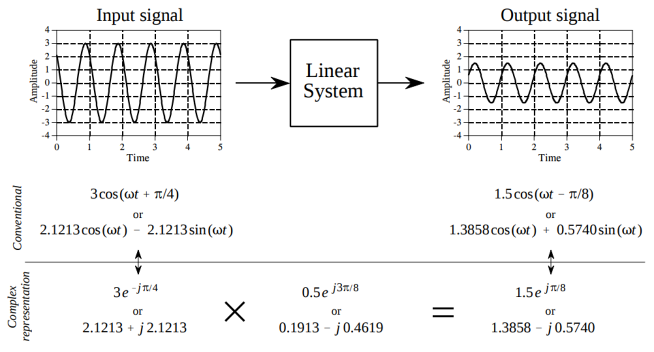


**Figure 1:** A linear, time-invariant system has special properties encapsulated by the *convolution operation*.

# INTRODUCTION: TYPES OF FILTERS

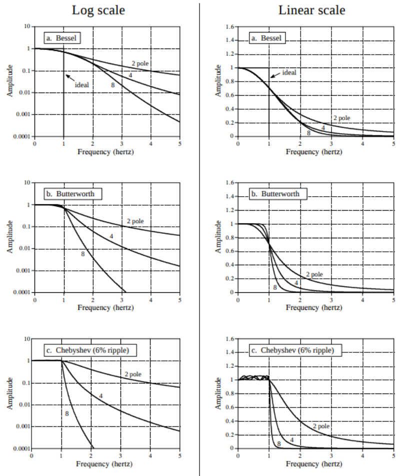


FIGURE 3-11
Frequency response of the three filters on a *logarithmic* scale. The Chebyshev filter has the sharpest roll-off.

FIGURE 3-12
Frequency response of the three filters on a *linear* scale. The Butterworth filter provides the flattest passband.

**Figure 2:** Comparison of transfer function magnitudes.

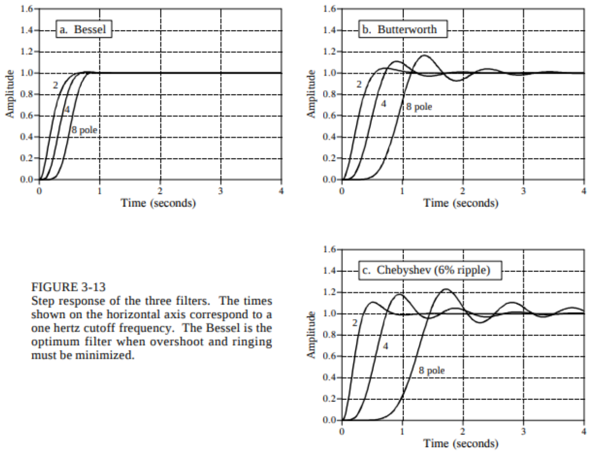


FIGURE 3-13
Step response of the three filters. The times shown on the horizontal axis correspond to a one hertz cutoff frequency. The Bessel is the optimum filter when overshoot and ringing must be minimized.

Figure 3: Comparison of transfer function step responses.

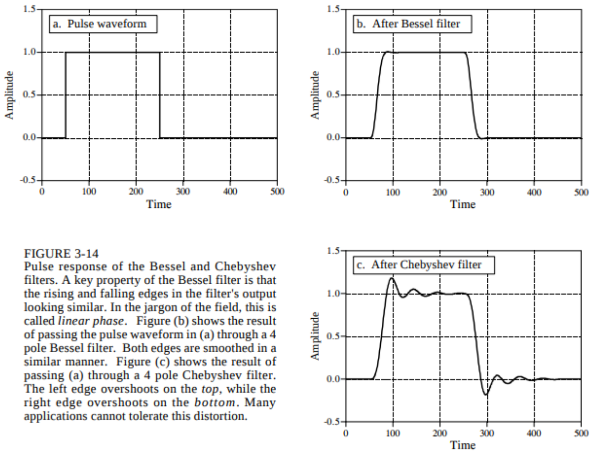


FIGURE 3-14
Pulse response of the Bessel and Chebyshev filters. A key property of the Bessel filter is that the rising and falling edges in the filter's output looking similar. In the jargon of the field, this is called *linear phase*. Figure (b) shows the result of passing the pulse waveform in (a) through a 4 pole Bessel filter. Both edges are smoothed in a similar manner. Figure (c) shows the result of passing (a) through a 4 pole Chebyshev filter. The left edge overshoots on the *top*, while the right edge overshoots on the *bottom*. Many applications cannot tolerate this distortion.

Figure 4: Comparison of transfer function pulse responses.

The single-pole Butterworth transfer functions are derived from the single-pole RC filter circuit:

$$H_{LP}(\omega) = \frac{\omega_0}{\omega_0 + j\omega} \tag{1}$$

$$H_{HP}(\omega) = \frac{\omega}{\omega - j\omega_0} \tag{2}$$

- What frequency causes a singularity in the transfer functions?
- What is the phase and group delay of this filter?

General expression for the transfer function of Butterworth filter (low-pass):

$$|H_{LP}(\omega)| = \frac{G_0}{\sqrt{1 + \left(\frac{j\omega}{\omega_0}\right)^{2n}}} \tag{3}$$

The integer $n$ is the number of poles. $G_0$ is the *gain*, and $\omega_0$ is the corner or cutoff frequency.

- Can we plot the poles in the complex plane?
- What is the phase and group delay of this filter?

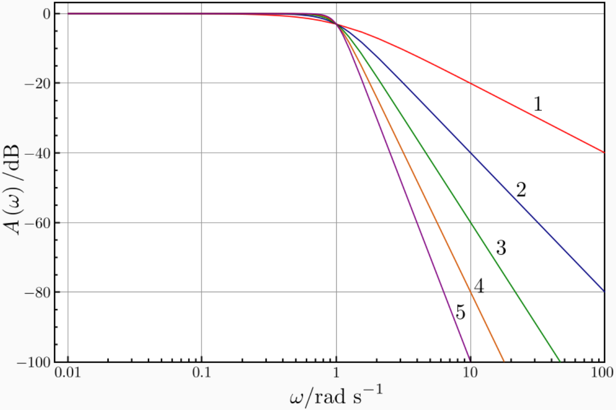


**Figure 5:** Gain of butterworth filters with $n$ poles.

The n-th order low-pass Bessel filter transfer function is a ratio of **reverse Bessel polynomials:**

$$H(\omega) = \frac{\theta_n(0)}{\theta_n(j\omega/\omega_0)} \tag{4}$$

where the reverse Bessel polynomials $\theta_n(x)$ are given by

$$\theta_n(x) = \sum_{k=0}^{n} \frac{(n+k)!}{(n-k)!k!} \frac{x^{n-k}}{2^k} \tag{5}$$

- What is $\theta_3$?
- How do we turn this into a high-pass filter?
- What are the pole locations of the 3rd-order Bessel filter?

The n-th order low-pass Chebyshev filter transfer function is

$$|H(\omega)| = \frac{1}{\sqrt{1 + \epsilon^2 T_n^2(\omega/\omega_0)}} \tag{6}$$

where the Chebyshev polynomials $T_n(x)$ are given by

$$T_n(x) = \cos(n \cos^{-1}(x)) \quad |x| < 1 \tag{7}$$

$$T_n(x) = \cosh(n \cosh^{-1}(x)) \quad x \geq 1 \tag{8}$$

$$T_n(x) = (-1)^n \cosh(n \cosh^{-1}(-x)) \quad x \leq 1 \tag{9}$$

- Can we plot $T_2(x)$ in Octave?
- Pole locations are interesting (next slide).

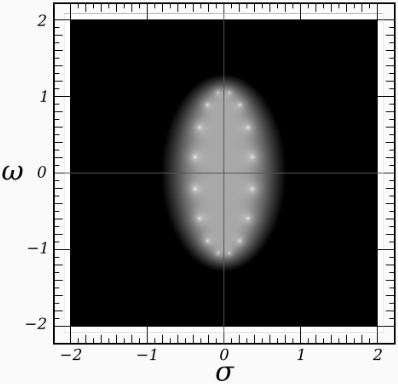


**Figure 6:** Eight-pole Chebyshev filter in the complex plane. The poles form an ellipse, due to the trigonometric nature of the definition of Chebyshev polynomials.

In the octave signal package, we can access the transfer functions of these filters:

```
pkg load signal;
[b1,a1] = butter(n,omega); %(e.g. include "high")
[b2,a2] = besself(n,omega);
[b3,a3] = cheby1(n,rp,omega); %rp pass-band ripple
x = (...); %data
y = filter(b1,a1,x);
```

Use **help** function on these for more information. The **filter** function is using the pole-zero information stored in the coefficients *a* and *b* to apply the transfer function to the data (more later).

# INTRODUCTION: TYPES OF FILTERS, OCTAVE PROGRAMMING EXAMPLE

If you have the **signal** package, you can specify a n-th order butterworth filter with **butter** as above. Otherwise, you can write a small function using Eq. 3 for the digital response.

1. Create a white-noise sample (**randn**) of $10^4$ samples. You can also specify a time vector of the same size.
2. Plot the spectrum of this noise using **fft**[1].).
3. **Filter** the noise with a 2nd-order low-pass butterworth filter, at a cutoff frequency of 0.2 $f_s$. Either use the **filter** function and fft, or multiply in the Fourier domain.
4. **Filter** the noise with a 2nd-order high-pass butterworth filter, at a cutoff frequency of 0.4 $f_s$.
5. Now do (3) then (4), and plot the spectrum. Repeat for (4) then (3). Do you see the same spectrum?

[1] See the **Aliasing.m** script in Unit 1

# LTI SYSTEMS

**Filters are LTI systems**. Let's review the properties of LTI systems:

1. Continuous vs. discrete
2. Scaling property
3. Distributive property
4. Time-invariance (also causality)
5. Commutative property
6. Combination of properties

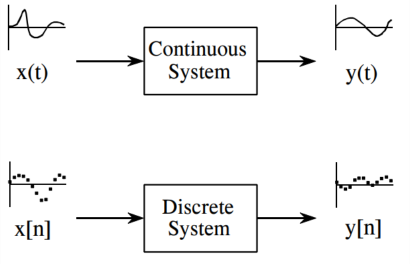


**Figure 7:** A continuous LTI system response to continuous data, and a discrete LTI system response to discrete data. All properties should hold both cases.

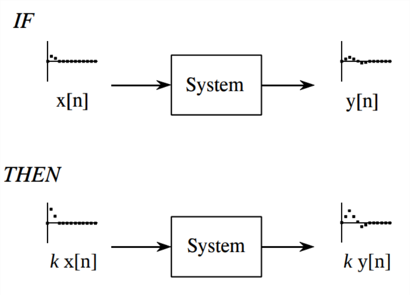


**Figure 8:** Scaling: If the data is scaled by a real constant, the output should be scaled by a real constant.

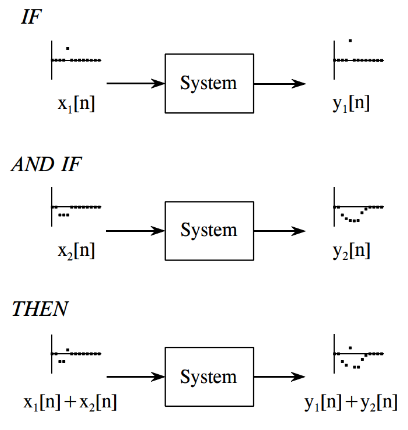


**Figure 9:** Distributive property: the LTI system should respond to each signal separately.

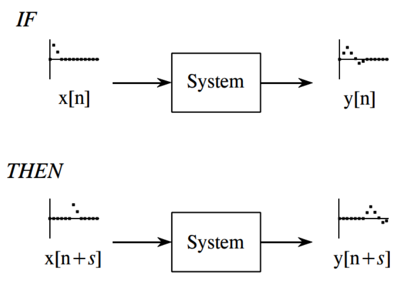


**Figure 10:** Time-invariance: the LTI system should respond when the signal arrives, independent of global time. Also, the filter should not respond *before* the signal arrives (causality).

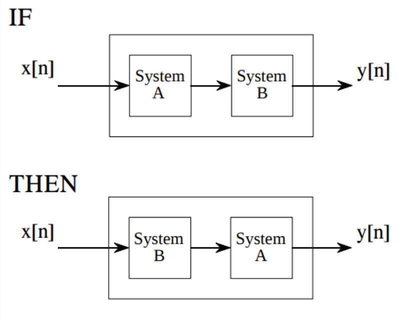


**Figure 11:** Commutative property: the LTI system should not depend on the existence of previous systems.

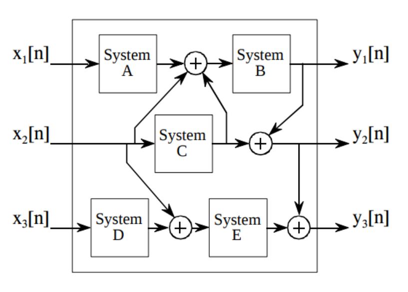


**Figure 12:** Combination of properties: an LTI system may be built from a combination of LTI systems.

Suppose **A(x)**, **B(x)**, **C(x)**, **D(x)**, and **E(x)** represent operators of LTI systems. What is the formula for a) $y_1$, b) $y_2$ and c) $y_3$? What is the the Fourier transform of the simplest output?

The **convolution** operator has all of the necessary properties of the LTI system operator. The convolution of data streams $x_1$ and $x_2$:

```
x_1 = (...); %data1
x_2 = (...); %data2
y_1 = fft(x_1);
y_2 = fft(x_2);
Z_1 = y_1.*y_2;
result = real(ifft(Z_1));
```

Or, in one line:

```
result = real(ifft(fft(x_1).*fft(x_2)));
```

Or,

```
result = conv(x_1,x_2); %Pay attention to size
```

# LTI SYSTEMS: PROGRAMMING WITH OCTAVE

The built-in Octave function **conv** for convolution produces two sizes, the full and the half-size.

```
x_1 = (...); %data
x_2 = (...); %data
y = conv(x_1,x_2);
size(y)
size(x_1)
size(x_2)
```

Special case of **conv**:

```
y = conv(x_1,x_2,”same”); %default; ”full”
size(y)
size(x_1)
size(x_2)
```

The unit-impulse function $\delta$ is given by the identity:

$$x[n] \circ \delta[n] = x[n] \tag{10}$$

Try the following in Octave:

```
clear; home; close;
function out = gauspulse(t,amp,mu,sigma)
    out = amp*exp(-0.5*((t-mu)/sigma).^2);
endfunction
t = linspace(0,10,10000);
r = zeros(size(t));
r(1000) = 1/sqrt(2); r(2500) = -1/sqrt(2);
x = gauspulse(t,1.0,5.0,0.2);
y = conv(r,x,"same");
plot(t,x,'color','black'); hold on;
plot(t,y,'color','blue');
sum(x.^2)/sum(y.^2)
```

1. Does the output make sense? Why?
2. **Exercise**: Write Octave code that produces a sine wave. Now write a transfer function that causes total destructive interferance. When the sine wave is convolved with the transfer function, the output should be a DC level.
3. **Exercise**: Write a transfer function that doubles the amplitude.
4. **Thought experiment**: Could you write a transfer function that would double the frequency? Why or why not?

## Transfer Function

The convolution of a **transfer function** with an input signal to an LTI system produces the output signal. Mathematically, if $h(t)$ is the transfer function, $i(t)$ is the input signal, and $o(t)$ is the output:

$$o(t) = h(t) \circ i(t) = \int_{-\infty}^{\infty} h(\tau) i(t - \tau) d\tau \tag{11}$$

The *convolution theorem* also states that

$$o(t) = F^{-1}\{H(\omega) I(\omega)\} \tag{12}$$

Consider the following function:

$$h(t) = \omega_0 \exp(-\omega_0 t) \quad t \geq 0 \tag{13}$$

$$h(t) = 0 \quad t < 0 \tag{14}$$

1. What is the Fourier transform of $h(t)$?
2. Do you recognize this transfer function?
3. Write an Octave function that digitizes the time-dependent version of this transfer function.
4. Convolve with a sine wave of frequency $2\omega_0$. What happens?

Let's repeat the exercise, but for *high-pass* filtering. Without knowing the time-domain equation for the response, how can we do this with transfer functions?

1. Consider taking signal *minus* the low-pass component. How do you add two transfer functions to get this to happen?
2. Convolve with a sine wave of a frequency less than the cutoff of the high-pass filter. Do you see attenuation?

**Normalization** of filters: (1) high-pass filters should have no gain at DC ($f = 0$ Hz), and (2) Filters should conserve energy (amplitude squared) in the pass-band. Let's ensure this by requiring

```
sum(h) = 1.0; %High-pass case
sym(h.^2) = 1.0; %Think about this one more...
```

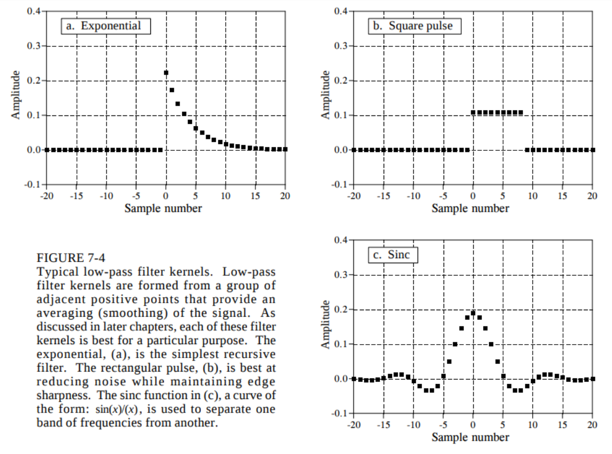


FIGURE 7-4
Typical low-pass filter kernels. Low-pass filter kernels are formed from a group of adjacent positive points that provide an averaging (smoothing) of the signal. As discussed in later chapters, each of these filter kernels is best for a particular purpose. The exponential, (a), is the simplest recursive filter. The rectangular pulse, (b), is best at reducing noise while maintaining edge sharpness. The sinc function in (c), a curve of the form: $\sin(x)/(x)$, is used to separate one band of frequencies from another.

**Figure 13:** Chapter 7: low-pass transfer functions, digitized. Notice that they are wide, and positive.

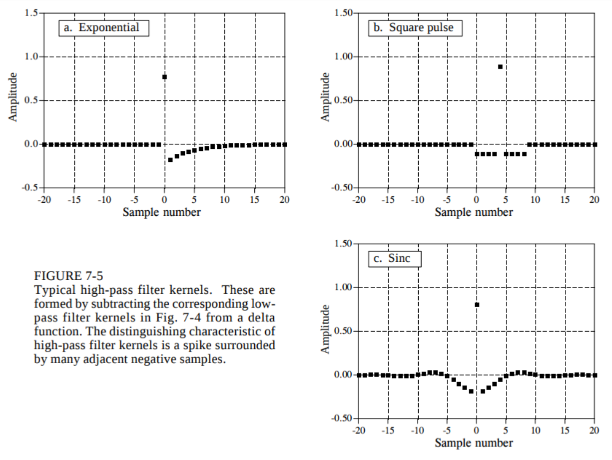


**Figure 14:** Chapter 7: high-pass transfer functions, digitized. Notice that they resemble the unit impluse minus a low-pass filter transfer function.

# LTI SYSTEMS: PROGRAMMING WITH OCTAVE

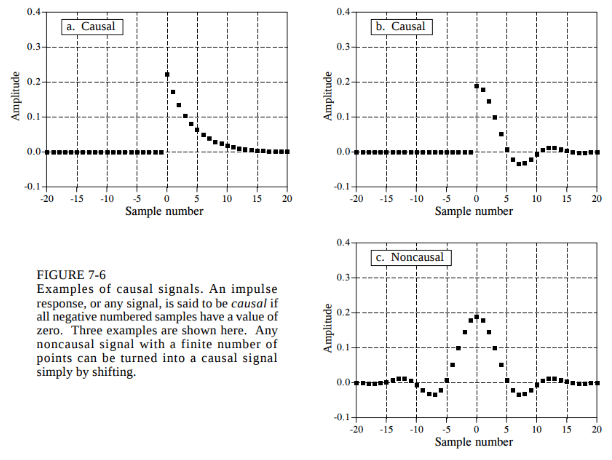


**Figure 15:** Chapter 7: causal filters do not *anticipate* the signal.

# LTI SYSTEMS: PROGRAMMING WITH OCTAVE

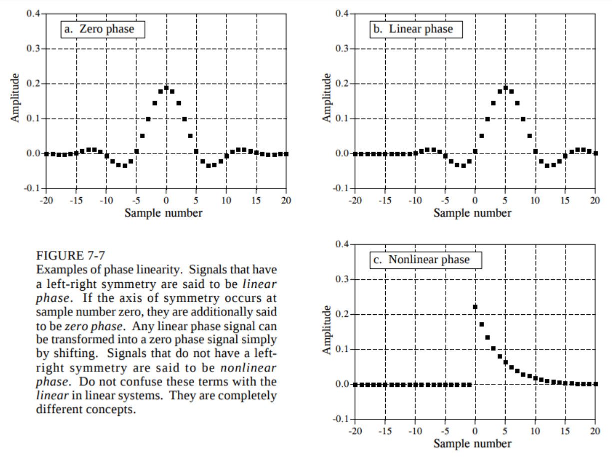


FIGURE 7-7
Examples of phase linearity. Signals that have a left-right symmetry are said to be *linear phase*. If the axis of symmetry occurs at sample number zero, they are additionally said to be *zero phase*. Any linear phase signal can be transformed into a zero phase signal simply by shifting. Signals that do not have a left-right symmetry are said to be *nonlinear phase*. Do not confuse these terms with the *linear* in linear systems. They are completely different concepts.

**Figure 16:** Chapter 7: the phase of the transfer function can be related to the group delay. What is meant by *linear* phase, versus zero phase, in terms of group delay?

# LTI SYSTEMS APPLICATION: RF FILTERS AND JOHNSON-NYQUIST NOISE

**Johnson-Nyquist noise** is the presence of random voltages across a circuit due to electron thermal fluctuations. Let $k_B$ be Boltzmann's constant[2], $\Delta f$ be some bandwidth $f_{max} - f_{min}$ in frequency, $R$ be the resistance in Ohms at a temperature $T$ in Kelvin. The variance of the noise is

$$v_{rms}^2 = 4k_B TR\Delta f \tag{15}$$

Let's write an Octave script to generate normally distributed noise with this *variance*. Make sure to define $\Delta f = f_{max} - f_{min}$. Next, define a cosine or sine with frequency that is inside this bandwidth, and add it to the noise. $SNR \approx 10$.

1. What happens to the SNR when you begin to filter the total signal, such that we keep the sinusoid intact?
2. Plot the SNR vs. cutoff frequency in your filter as you vary it.

[2] $k_B = 1.38 \times 10^{-23}$ m$^2$ kg s$^{-2}$ K$^{-1}$.

# SPECIAL TOPIC: IMPULSE RESPONSE OF RF ANTENNAS

A paper on RF antenna response for UHE neutrino research:
`https://doi.org/10.1016/j.astropartphys.2014.09.002`