1. Administrivia
2. DSP review
3. Fun with Matlab
Course overview

- **Advanced** Digital Signal Theory
- Design, analysis, and implementation of audio effects and synthesizers
  - EQ, reverb, chorus, phase vocoder, sinusoidal modeling, FM synthesis, ...
- Emphasis on practical implementation, building complete systems.
whoami

- PhD, Electrical Engineering, Columbia University
- Research interests: Source separation, speech recognition, music information retrieval
- http://www.ee.columbia.edu/~ronw
Sound familiar?

- Periodic and aperiodic signals
- Discrete Fourier Transform
- Convolution, filtering
- Linear time-invariant systems
- Impulse response
- Frequency response
- z-transform
Digital signals

Discrete-time signal  sequence of samples
- e.g. $x[n] = [0, -2.3, -1.3, 20, 4.2, ...]$

Digital signal  discrete-time signal that has been quantized
- Discrete on both axes: samples can only take on a limited set of values (quantization levels)
- Quantization introduces noise
Sampling

\[ t = n \frac{1}{f_s} \]

- \( t \) time in seconds (real number)
- \( n \) time in samples (integer)
- \( f_s \) sampling rate (e.g. 44100 \( \text{samples/second} \))
Important signals: impulse

Think of all discrete time signals as a sequence of scaled and time-shifted impulses.

\[ x[n] = [0, -2.3, -1.3, 20, 4.2, \ldots] \]

\[ = 0 \delta[n] - 2.3 \delta[n - 1] - 1.3 \delta[n - 2] + 20 \delta[n - 3] + 4.2 \delta[n - 4] + \ldots \]
Important signals: sinusoid

\[ x[n] = \sin(2\pi fn) \]

- \( \sin(2\pi fn + \phi) \), frequency = \( f \) cycles/sample, phase = \( \phi \)
- Period: \( N = \frac{1}{f} \) samples
- But what will it sound like?

\( x[n] = \sin(2\pi fn) \)
Important signals: sinusoid

- \( x[n] = \sin(2\pi fn) \)

- \( \sin(2\pi fn + \phi) \), frequency = \( f \) cycles/sample, phase = \( \phi \)
- Period: \( N = \frac{1}{f} \) samples
- But what will it sound like?
  - Convert from samples: \( f \text{ cycles/sample} \times \frac{1}{f_s} \text{ samples/second} \)
  - How many samples in one period of a 440 Hz tone sampled at 44.1 kHz?
Discrete Fourier Transform

Decompose any **periodic** signal into sum of re-scaled sinusoids

**Inverse DFT**

\[
x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi nk/N} \quad k = 0, \ldots, N - 1
\]

**Forward DFT**

\[
X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi nk/N}
\]

Note that \(X[k]\) are complex: \(X[k] = X_R[k] + jX_I[k]\)
Magnitude and Phase spectra

$$X[k] = |X[k]| e^{j\angle X[k]}$$

- **Magnitude**: amount of energy at each frequency
  $$|X[k]| = \sqrt{X_R^2[k] + X_I^2[k]}$$

- **Phase**: delay at each frequency
  $$\angle X[k] = \arctan \frac{X_I[k]}{X_R[k]}$$

$$x[n] = \cos(2\pi fn)$$

Graphs showing magnitude and phase spectra for the given signal.
The spectrum of a discrete time signal is periodic with period $f_s$.

The spectrum of a real valued signal is symmetric around $f_s/2$.

Any energy at frequencies greater than $f_s/2$ will wrap around.
What if frequency content varies with time?

- Break signal up into short (optionally overlapping) segments
- Multiply by window function
- Take DFT of each segment
STFT example
Linear time-invariant systems

- Process input signal using delays, multiplications, additions
- Describe using a difference equation, e.g.
  \[ y[n] = b_0 x[n] + b_1 x[n - 1] \]
- Can also have feedback, e.g.
  \[ y[n] = b_0 x[n] + b_1 x[n - 1] + a_1 y[n - 1] + a_2 y[n - 2] \]
LTI systems – Block diagram
LTI systems – Properties

Given:

\[ x[n] \rightarrow h \rightarrow y[n] \]
\[ x_2[n] \rightarrow h \rightarrow y_2[n] \]

1. Linearity

\[ a \times x[n] \rightarrow h \rightarrow a \times y[n] \quad \text{Scaling} \]
\[ x[n] + x_2[n] \rightarrow h \rightarrow y[n] + y_2[n] \quad \text{Superposition} \]

2. Time-invariance

\[ x[n-k] \rightarrow h \rightarrow y[n-k] \]
Impulse response and convolution

- Characterize LTI system in time-domain by its impulse response \( h[n] \)
  - An LTI system is just another signal
  - Given input signal \( x[n] \), output is convolution with \( h[n] \)

**Convolution**

\[
y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k] h[n-k]
\]

- But how do we compute the impulse response from a difference equation?
Maps discrete-time signal to a continuous function of a complex variable

Incredibly useful for analyzing LTI systems
- Turns difference equations into polynomials:
  \[ x[n - m] \leftrightarrow z^{-M} X(z) \]

Convolution becomes multiplication:
\[ x[n] * h[n] \leftrightarrow X(z) H(z) \]

If \( z = e^{j\Omega} \), get discrete-time Fourier transform (DTFT)
Transfer function

- Transfer function $H(z)$ is the z-transform of the impulse response
- Can read off z-transform from difference equation

$$y[n] = b_0 x[n] + b_1 x[n - 1] + a_1 y[n - 1] + a_2 y[n - 2]$$
Transfer function

- Transfer function $H(z)$ is the z-transform of the impulse response
- Can read off z-transform from difference equation

\[ y[n] = b_0 x[n] + b_1 x[n - 1] + a_1 y[n - 1] + a_2 y[n - 2] \]

\[ \zrightarrow Y(z) = (b_0 + b_1 z^{-1}) X(z) + (a_1 z^{-1} + a_2 z^{-2}) Y(z) \]
Transfer function

- Transfer function $H(z)$ is the z-transform of the impulse response
- Can read off z-transform from difference equation

\[ y[n] = b_0 x[n] + b_1 x[n - 1] + a_1 y[n - 1] + a_2 y[n - 2] \]

\[ \overset{z}{\leftarrow} Y(z) = (b_0 + b_1 z^{-1}) X(z) + (a_1 z^{-1} + a_2 z^{-2}) Y(z) \]

\[ H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1}}{1 - a_1 z^{-1} - a_2 z^{-2}} \]
Transfer function

- Transfer function $H(z)$ is the z-transform of the impulse response
- Can read off z-transform from difference equation

$$y[n] = b_0 x[n] + b_1 x[n - 1] + a_1 y[n - 1] + a_2 y[n - 2]$$

$$\overset{z}{\rightarrow} Y(z) = (b_0 + b_1 z^{-1}) X(z) + (a_1 z^{-1} + a_2 z^{-2}) Y(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1}}{1 - a_1 z^{-1} - a_2 z^{-2}}$$

- But why? $\delta[n] \overset{z}{\rightarrow} 1$
- Compute impulse response analytically by finding inverse z-transform of $H(z)$ (lots of algebra)
Often more intuitive to analyze system in the frequency-domain

- $H(e^{j\Omega})$ DTFT of impulse response
  - Slice of z-transform corresponding to the unit circle
  - DFT (discrete freq) is just sampled DTFT (continuous freq)
Zeros: roots of numerator of $H(z)$
  - Correspond to valleys in frequency response
Poles: roots of denominator of $H(z)$
  - Correspond to peaks in frequency response
System is **unstable** if it has poles outside of (or on) the unit circle
  - Impulse response goes to infinity
Finite Impulse Response
- No feedback ⇒ all zeroes ⇒ always stable*
  * if coefficients are finite
- Easy to design by drawing frequency response by hand, then using inverse DFT to get impulse response
- Often needs a long impulse response ⇒ expensive to implement

Infinite Impulse Response
- Feedback ⇒ has poles ⇒ can be unstable
- Can implement complex filters using fewer delays than FIR
- But harder to design
<table>
<thead>
<tr>
<th>Topic</th>
<th>Function(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal generation</td>
<td>linspace, rand, sin</td>
</tr>
<tr>
<td>I/O</td>
<td>wavread, wavwrite, soundsc</td>
</tr>
<tr>
<td>Plotting</td>
<td>plot, imagesc</td>
</tr>
<tr>
<td>Transforms</td>
<td>fft, ifft</td>
</tr>
<tr>
<td>Filtering</td>
<td>conv, filter</td>
</tr>
<tr>
<td>Analyzing filters</td>
<td>freqz, zplane, iztrans</td>
</tr>
</tbody>
</table>
• Review your DST notes
• Skim *Introduction to Digital Filters*
  • Linear Time-Invariant Filters
  • Transfer Function Analysis
  • Frequency Response Analysis
• *DAFX*, Chapter 1 (if you have it)