ECE 4750/6750 + CS 4501/6501: Digital Signal Processing **Due Wed 2/15**

Homework 1: Signal and System Basics

Instructions: Submit a single Jupyter notebook (.ipynb) of your work to Collab by 11:59pm on the due date. All code should be written in Python. **Be sure to show all the work involved in deriving your answers! If you just give a final answer without explanation, you may not receive credit for that question.**

You may discuss the concepts with your classmates, but write up the answers entirely on your own. Do not look at another student's answers, do not use answers from the internet or other sources, and do not show your answers to anyone. **Cite any sources you used outside of the class material (webpages, etc.), and list any fellow students with whom you discussed the homework concepts.**

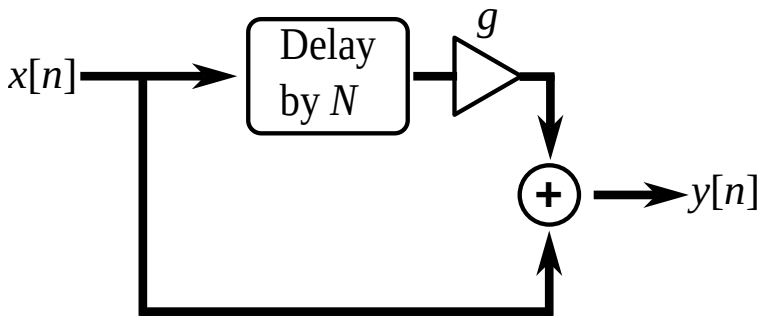
1. Consider the signal
$$x[n] = 2\delta[n] - \delta[n-1] + 3\delta[n-2],$$
and its moving average
$$y[n] = \frac{1}{2}(x[n] + x[n-1]).$$
Write $y[n]$ in terms of scaled and shifted impulses ($\delta[n]$).

2. What is the period of the signal
$$x[n] = \sin\left(\frac{2\pi M n}{N}\right),$$
for the following values:
   (a) $M = 15, N = 35$
   (b) $M = 7, N = 13$
   (c) $M = 21, N = 3$

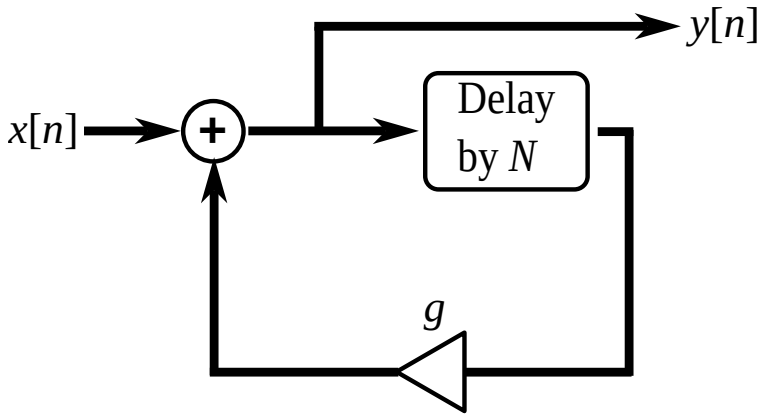
3. Compute the following complex numbers, and write them both in standard form and Euler notation:
   (a) $e^{\frac{i\pi}{3}}\left(1 + \frac{1}{2}i\right)$
   (b) $\left(e^{\frac{i\pi}{2}}\right)^{-1}$
   (c) $(3 + 2i)^{\frac{1}{2}}$

Fun with Audio Signals

4. Consider the following system for a *feedforward comb filter* (FFCF):



Also consider the following system for a *feedback comb filter* (FBCF):



(a) Write out equations for both the FFCF and FBCF systems.

(b) Are these both LTI systems? Briefly explain why or why not.

(c) Implement both the FFCF and FBCF systems as Python functions. Each of your functions should take three parameters: (1) the input signal $x[n]$ (i.e., a `numpy` array of any length), (2) the delay $N > 0$, and (3) the *gain*, $g \in \mathbb{R}$. Each should return the output signal for that system, $y[n]$.

(d) Run both of your systems on the signal `woodchuck.wav` with a delay of 100ms and a gain of 0.75. Note: you will have to convert the delay from milliseconds to number of samples, N. Play the resulting output audio signal and describe what you hear. What are the differences between the resulting sounds from the FFCF and FBCF outputs? (Just looking for your qualitative observations.)

(e) Now run both systems on the same `woodchuck.wav` with the same delay, but now set the gain to 1.0. What changed? Explain what is different about the FFCF and the FBCF that causes them to behave so differently in this case.

(f) Now apply four FBCF filters in parallel and sum the resulting four outputs together to get the final output. That is, build the following system:

$$y[n] = \text{FBCF}_1\{x[n]\} + \text{FBCF}_2\{x[n]\} + \text{FBCF}_3\{x[n]\} + \text{FBCF}_4\{x[n]\}.$$

The individual FFCF systems should have the following parameters:

FBCF_1 : delay = 100ms, gain = 0.75
FBCF_2 : delay = 90ms, gain = 0.8
FBCF_3 : delay = 120ms, gain = 0.77
FBCF_4 : delay = 110ms, gain = 0.73

Describe what you are hearing. How is it different from the single FBCF output from part (d)? (again, qualitative!)

(g) Construct a signal in Python representing the unit impulse function, $\delta[n]$. Your signal should range from $n = -10, \ldots, 10$. Plot this signal as a stem plot.

(h) Run your unit impulse function through each of your two systems, FFCF and FBCF, to get the impulse response $h[n]$ for each system. Use the same parameters, delay = 100ms and gain = 0.75. Plot $h[n]$ for each system. Now repeat these plots of $h[n]$ for both systems, but changing the gain to 1.0. What properties of these two impulse responses might lead to the different behavior of the FFCF and FBCF systems that you observed above? **Update:** Consider the sampling rate of your impulse from (g) to be 10 Hz. Alternatively, you can use an impulse with same sampling rate and signal length as the `woodchuck.wav` signal.

(i) Write a Python function to convolve two signals. Your function should take two input signals $x[n]$ and $h[n]$ and return the convolution signal $(x * h)[n]$. Do not use `numpy.convolve` or other Python library that implements convolution—do it yourself with basic operations. You should zero pad, that is, when your index into an array goes out of bounds during the computation, just consider the value for that array to be zero. (**Hint:** You will want to avoid using two nested `for` loops to do this. You may consider using a numpy function, for example, `sum` or `dot`, to do the summation in the convolution equation.)

(j) Create another unit impulse, now with length equal to the `woodchuck.wav` signal. Compute the corresponding impulse response, $h[n]$, for the FBCF system with delay = 100ms, gain = 0.75. Now use your convolve function to convolve `woodchuck.wav` with $h[n]$. Play the audio—does it match what you heard in part (d)?

For Grads Only (or Extra Credit for Undergrads)

5. Consider the causal LTI system

$$y[n] = gy[n-1] + x[n] - g^N x[n-N]$$

(a) Write the equation for the impulse response function, $h[n]$, for this system.

(b) For what values of g is this system stable? Explain.

(c) Implement this system as a Python function. Similar to the procedure used above, create a unit impulse and pass it to your Python function. Plot the resulting impulse response. Overlay a plot of the equation for $h[n]$ that you got in part (a). Verify that they match.