EE 102: Signal Processing and Linear Systems

Instructor: Ayush Pandey

Homework #5: Applications of convolution and its properties

Name: ______________ **Submission Date:** ______________

Problem 1 Consider the RC circuit discussed in class with input voltage $v_{\text{in}}(t)$ and output voltage measured across the capacitor $v_{\text{out}}(t)$. You may directly use the impulse response of this circuit for this problem.

Compute the convolution and graphically visualize (by hand, no programming) the output for the following inputs:

(Note added on 04/10) For graphical visualization, you are not expected to compute the numerical / analytical equations using graphs. The goal is to visualize the process of convolution. You would use the integral to find the analytical equation for your final answer.

(a) [20 points] A pulse input of amplitude A lasting for T seconds, starting at $t = 0$:

$$x(t) = \begin{cases} A, & 0 \leq t < T, \\ 0, & \text{otherwise.} \end{cases}$$

(b) [20 points] A sinusoidal voltage input of frequency f Hz that is applied to the circuit after a delay of 5 seconds and taken away (set to 0 after 10 seconds):

$$x(t) = A \sin(2\pi f t) \left[u(t-5) - u(t-10) \right].$$

Problem 2 Prove that the convolution operation is associative. Although this holds true in both continuous-time and discrete-time, you should prove it for the discrete-time case only. Specifically, consider two systems with unit impulse responses: $h_1[n]$ and $h_2[n]$ given as:

$$h_1[n] = \alpha^n u[n]$$

for $\alpha = \frac{1}{4}$ and

$$h_2[n] = u[n-4] - 2u[n+1]$$

for $n \in \mathbb{Z}$. The two systems are cascaded as shown in the block diagram below (Figure 1).

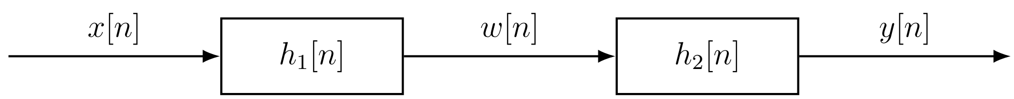

Figure 1: Cascaded discrete-time LTI systems.

The final output $y[n]$ can be computed in two different ways:

(a) [15 points] For the specific impulse responses given, compute $w[n] = x[n] * h_1[n]$ and then compute $y[n] = w[n] * h_2[n]$. That is, $y[n] = (x[n] * h_1[n]) * h_2[n]$. Obtain $y[n]$ in closed form.

(b) [15 points] Alternatively, we can find $y[n]$ by first computing the overall impulse response of the cascaded system $h[n] = h_1[n] * h_2[n]$ and then convolving it with the input: $y[n] = x[n] * (h_1[n] * h_2[n])$. Obtain $y[n]$ in closed form.

(c) [5 points] Show that the two results are equal, thus proving associativity of convolution. Write the associativity property of convolution (for three general signals) in your own words.

Problem 3

Let $x[n]$ be a 1-D signal that represents grayscale colors as pixel intensities (0 is black and 255 is white):

$$x[n] = [\,50,\ 100,\ 240,\ 255,\ 200,\ 120,\ 80,\ 80,\ 90,\ 150,\ 220,\ 240\,], \qquad 0 \le n \le 11.$$

Assume causal zero-padding outside the given range, that is, $x[n] = 0$ for $n < 0$ or $n > 11$. Your goal is to compute $y[n]$ by hand and also using a for loop implementation (in Python or MATLAB) of the convolution sum. **You are not allowed to use external libraries to compute the convolution. Of course, you should use external libraries for plotting.**

Note: Starter code is provided in the file `hw5_problem3.ipynb`. You should fill in the missing parts.

Consider the two physically meaningful impulse responses $h[\cdot]$ (both causal) of LTI systems:

(a) A blurring system:

$$h[n] = \tfrac{1}{3}\,[\,\delta[n] + \delta[n-1] + \delta[n-2]\,]$$

(b) A first-difference (edge detector):

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

For each of the parts above,

1. **4 points for each** h By hand, write the convolution sum for $y[n]$ and compute numerically $y[n]$ at $n = 0, 1, 2$.
2. **4 points for each** h Implement a for loop that computes $y[n]$ for all n for system above. Make sure to plot the original $x[n]$ and each $y[n]$ on the same axes.
3. **4 points for each** h Apply repeated convolution to intensify the effect of the system. You can choose one of the systems above and experiment with repeated convolutions.

Problem 4

(a) [1 point] How long did this assignment take you to complete (this does not include the time spent in lectures or in labs, but it does include the time spent programming).