ESE 351 Spring 2023 Homework #2 – Matlab supplement
Assigned Thursday, January 25
Due 5pm Friday, February 2 – Submit in gradescope
See syllabus for late submission policy

Complete the tasks below. Each student must write and submit their own work. You are welcome to discuss the assignment, course contents, and Matlab code with others, but you must write your own m-file and you must acknowledge any collaborators at the top of your m-file.

Submit your work as "published" pdf report (you do not need to submit your m-file). Include comments in your m-file so that a published pdf of your script output is a self-contained description of your work. See ESE351ExampleScript.m and ESE351ExampleScript.pdf on canvas for a Matlab example with general expectations. Pay close attention to the generation of figures that illustrate your results clearly. When helpful, use subplot and/or plot multiple items in a single graph. Include titles on all figures or subplots and label all axes. Use the **plot**( ) command for plotting CT functions and **stem**( ) for DT functions.

Introduction

In Matlab Homework 1 you used continuous-time and discrete-time models of two RC circuits to explore behavior of those circuits. You should have found that with the output as the voltage across the capacitor, the circuit passed low frequencies and attenuated high frequencies (a lowpass filter). With the output across the resistor, the frequency response was reversed, resulting in a highpass filter. (Note: the frequency of $1/\tau$ is roughly a transition frequency between the circuit's 'passband' and 'stopband').

This week, you will explore convolution and the system impulse response. As you experienced in Homework 1 when simulating lowpass and highpass RC circuits, computation is generally discrete time (DT), either for an inherently discrete-time system or as an approximation to a continuous-time system.

Task 1 involves DT convolution in Matlab. Task 2 explores solution of discrete-time difference equations, both directly from the difference equation and by using the impulse response, including simulation of the RC circuits as in Homework 1. In Task 3, you will examine system frequency response by computing the output for a chirp signal input. In Task 4, you will experiment again with processing an audio signal.

Background

Recall the differential equation for the lowpass RC circuit, $\frac{dy}{dt}+\frac{1}{\tau}y=\frac{1}{\tau}x$, with $\tau = RC$. As we will see later in this course, its impulse response is a right-sided decaying exponential function, $h(t)=\frac{1}{\tau}e^{-t/\tau}\ u(t)$. The highpass RC circuit, modeled as $\frac{dy}{dt}+\frac{1}{\tau}y=\frac{dx}{dt}$, has impulse response $h(t)=\delta(t)-\frac{1}{\tau}e^{-\frac{t}{\tau}}u(t)$, which includes a continuous-time impulse, making it difficult to model computationally. (If you are curious, you can try to derive the step and impulse responses yourself from the differential equations).

In addition to the impulse and step response, characterization of a system's response to sinusoids is very useful. Later, we will have simpler methods to compute a system's frequency response (its impact on inputs of different frequencies). A quick way to see that response is to find the output for a 'chirp' signal. A basic chirp signal is given by $x(t) = \cos\left(2\pi\frac{ft^2}{2}\right)$. Note that it has instantaneous frequency of ft, i.e., its frequency increases with time, which allows you to see the response of a system to multiple different frequencies using just a single input.

Parameters and definitions used in the tasks:

- Recall the discrete-time rectangular function, $rect[n] = \begin{cases} 1, & 0 \le n \le N \\ 0, & else \end{cases}$.

- For the RC circuits, use these values from Homework 1:
  - Use R and C for a time constant $\tau = \frac{1}{200} = 0.005$, e.g., $R = 1k\Omega, C = 5\mu F$.
  - Use sample frequency $f_s = 44.1kHz$ (common digital audio sampling rate).
  - Compute for a time range of 15 time constants, i.e., $0 \le t \le 15\tau$.
  - As in Homework 1, your results for the RC circuits represent CT functions, so use **plot** to display those results versus continuous time (note that you will need to compute the continuous-time values).

Tasks

1. Compute discrete-time convolution for a finite-length function.
   a. Compute and plot $x[n] = rect[n] * rect[n]$ for $N = 3, 10$ and 21. Use Matlab's **conv** for the computation. Be careful about the indexing and labeling of n.
   b. For $x[n]$ of part (a) with $N = 21$, compute and plot $y[n] = x[n] * \sum_{k=-\infty}^{\infty} \delta[n - kM]$ for $M = 10, 25,$ and 50. Plot for $-200 \le n \le 200$. Note that you only need to model that portion of the impulse train required to accurately compute $y[n]$ over this range.
2. Use **filter** in Matlab to simulate DT systems described by difference equations (recall syntax **filter(b,a,x))**. You'll model finite-length DT filters as in Task 1 as well as the lowpass and highpass RC circuits of Homework 1.
   a. Use the difference equation representation for the lowpass and highpass RC circuits to compute (using **filter()**) the impulse response (DT model) for each circuit. Plot and compare to the analytic forms given above in the background section for the CT models. Note that these impulse responses are infinite-length; compute and plot only for the time range given above. Also, the DT and CT impulse responses should be comparable but not identical (as we discussed in class, the DT and CT impulse functions are quite different).
   b. Compute and plot the response of each RC circuit to a step input, $u(t - 2\tau)$, using the methods below and compare the results:
      i. Compute directly from the difference equation using **filter**,
      ii. Compute as a convolution of the input and the impulse response found in the previous step,
      iii. Compute using **lsim**. Note: recall that the coefficients (**b**,**a**) used for the DT model in **filter()** are different from those used for the CT model in **lsim**().

Recall that depending on your version of Matlab, you might need to use a newer syntax: **lsim(sys, u, t),** for time **t**, input **u,** and **sys=tf(b,a)**.

c. Use **filter** to compute the convolution in Task 1.a.

3. Use the chirp function (frequency sweep) to approximate and plot the system frequency response for the systems below. Use either function (**filter** or **conv**) to compute.
   a. Lowpass and highpass RC circuits (DT approximation),
   b. DT rectangular function, with variable length $N = 10, 25, 50, and\ 100$.
4. Filter an audio signal with the systems in Task 3a and 3b. You do not need to include these results in your m-file or published pdf, but include a comment with your observations.

Note: Matlab has two functions that compute a DT convolution result:

- **conv(**a,b) computes the DT convolution between vectors a and b. There are no time arguments, and the output has length of length(a) + length(b) – 1 as expected for a convolution of two finite-length sequences.
- **filter**(b,a,x) computes the output of a DT system with difference equation coefficient vectors **b** and **a**, for input **x**. The output is computed for the same length as the input.

This snippet of Matlab code computes a chirp, then filters with two short DT filters, then plots the output as an approximate system frequency response.

$$y[n] = \frac{1}{2}(x[n] + x[n-1])\ \ (\textit{2-point causal moving average filter})$$

$$y[n] = x[n] - x[n-1]\ \ (\textit{causal first difference, e.g., a DT approximation to the CT derivative})$$

```
% sample code to define chirp signal and filter it to show approximate
% system frequency response. Shown here for simple FIR HPF and LPF
fs = 44.1e3; % sampling frequency
dT = 1/fs; % sampling period
t = 0:dT:3; % time vector
fmin = 10; fmax = 10e3; % 10000; % min and max frequencies for chirp
fchirp = (fmax-fmin).*t/max(t)+fmin; % chirp instantaneous frequency
xchirp = cos(2*pi*fchirp/2.*t); % chirp signal
% xchirp = chirp(t,fmin,max(t),fmax,'logarithmic'); % use of Matlab chirp
with logarithmic frequency variation
sound(xchirp,fs)
ychirpHPF = filter(.5*[1 -1],1,xchirp);
ychirpMA = filter(ones(2,1),1,xchirp);
% visualization - log-frequency linear-amplitude
figure, subplot(3,1,1), plot(fchirp,xchirp); title('Chirp input')
subplot(3,1,2), plot(fchirp,(ychirpMA)); title('DT Moving Average')
subplot(3,1,3), plot(fchirp,(ychirpHPF)); title('DT Highpass Filter'),
xlabel('Frequency (Hz)')

% to visualize with log or linear scale for frequency or amplitude, use
this code accordingly
for i = 1:3, subplot(3,1,i), set(gca,'XScale','log'),
set(gca,'YScale','linear'), axis tight, end
```