ECE 201 – Introduction to Signals and Systems

Project Part 1

Spring 2020

Due Date
The lab report for this part of the project must be submitted on Blackboard before the start of your next lab section. Late lab reports will be penalized as noted in the Lab Syllabus.

Description
This project has four main goals:

1. To learn about the *frequency response* of linear time-invariant (LTI) systems;
2. To learn how to use MATLAB to implement convolution numerically;
3. To explore different methods for measuring the frequency response of an LTI system;
4. To learn about the power of a signal and how to measure power.

The project is split into four parts. This document describes Part 1, which focuses on the definition of the frequency response of an LTI system and how to compute the frequency response from the impulse response.

Lab Report Your lab report for this lab will consist of answers and complete documentation of the questions and exercises in Section 2. Section 1, is designed to get you ready for the problems in Section 2. **Do not skip Section 1!** Guidelines for preparing the lab report are posted on the Blackboard site. If you have any questions, talk to your lab TA. Each student must do his or her own work on this lab. However, you may ask other students or any of the teaching staff for advice.

Honor Code
Any reasonable suspicion of an honor code violation will be reported. You are allowed to discuss lab exercises with other students, but the submitted work should be original and it should be your own work. For more information about the Honor Code, please see the ECE 201 Lecture Syllabus and the ECE 201 Lab Syllabus. If you have questions, ask a member of the teaching staff.

1 Background

Systems are used to process signals. As indicated in Figure 1, they accept an input signal $x[n]$, process that signal, and produce the output signal $y[n]$. Throughout this project, we focus on discrete-time system so that both signals $x[n]$ and $y[n]$ consist of sequences of samples. As a result, the frequencies that we will encounter are normalized (digital) frequencies $\hat{f}_d$. Recall that the digital frequencies relate to analog frequencies trough the relationship $\hat{f}_d = \frac{f}{f_s}$. We will assume that analog signals have been oversampled so that $0 \leq \hat{f}_d < \frac{1}{2}$.

Linear, time-invariant (LTI) systems are a large and very important subset of systems for signal processing. You have explored the concepts of linearity and time-invariance in a previous lab. For any LTI system, the output signal $y[n]$ is computed from the input signal $x[n]$ through a a *convolution* with the *impulse response* $h[n]$ of the LTI system,

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} h[k] \cdot x[n-k]. \tag{1}$$

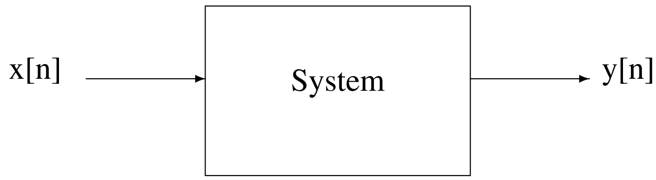

Figure 1: A system processes the input signal $x[n]$ to produce the output signal $y[n]$.

An important aspect of LTI systems is their response, i.e., output, when the input signal $x[n]$ is a complex exponential signal of frequency $\hat{f}_d$. When $x[n] = \exp(j2\pi \hat{f}_d n)$ is convolved with the filter's impulse response $h[n]$ according to equation (1) then the output signal $y[n]$ is

$$\begin{aligned}
y[n] &= \sum_{k=-\infty}^{\infty} h[k] \cdot x[n-k] \\
&= \sum_{k=-\infty}^{\infty} h[k] \cdot \exp(j2\pi \hat{f}_d (n-k)) \\
&= \sum_{k=-\infty}^{\infty} h[k] \cdot \exp(j2\pi \hat{f}_d n) \cdot \exp(-j2\pi \hat{f}_d k) \\
&= \exp(j2\pi \hat{f}_d n) \cdot \sum_{k=-\infty}^{\infty} h[k] \exp(-j2\pi \hat{f}_d k) \\
&= \exp(j2\pi \hat{f}_d n) \cdot H(e^{j2\pi \hat{f}_d}).
\end{aligned}$$

On the last line, we have defined

$$H(e^{j2\pi \hat{f}_d}) = \sum_{k=-\infty}^{\infty} h[k] \exp(-j2\pi \hat{f}_d k). \tag{2}$$

The quantity $H(e^{j2\pi \hat{f}_d})$ is called the *frequency response* of the system. Notice that $H(e^{j2\pi \hat{f}_d})$ is a complex-valued scalar that depends on the impulse response $h[n]$ and on the frequency f_d. For a given LTI system with impulse response $h[n]$ it is of great interest to know how the frequency response varies with $\hat{f}_d$.

We showed above that when the input signal is a complex exponential signal of frequency $\hat{f}_d$, i.e., $x[n] = \exp(j2\pi \hat{f}_d n)$, then the output signal is

$$y[n] = \exp(j2\pi \hat{f}_d n) \cdot H(e^{j2\pi \hat{f}_d}). \tag{3}$$

This equation provides three important insights:

1. When the inputs signal to an LTI system is a complex exponential signal of frequency $\hat{f}_d$, then the output signal is also a complex exponential signal of the same frequency.

2. The frequency response $H(e^{j2\pi \hat{f}_d})$ determines the amplitude and phase of the output signal. From the above, we see that for an input signal with amplitude $A_x = 1$ and phase $\phi_x = 0$, $H(e^{j2\pi \hat{f}_d})$ is the phasor of the output signal. Therefore, the amplitude of the output signal is $A_y = |H(e^{j2\pi \hat{f}_d})|$ and its phase is $\angle H(e^{j2\pi \hat{f}_d})$. More generally, when the input signal has amplitude A_x and phase ϕ_x, then the output signal has amplitude $A_y = A_x \cdot |H(e^{j2\pi \hat{f}_d})|$ and phase $\phi_y = \phi_x + \angle H(e^{j2\pi \hat{f}_d})$.

3. From these two considerations, it follows that for a complex exponential input signal $x[n] = \exp(j2\pi \hat{f}_d n)$, we do not need to perform a convolution to find the output signal $y[n]$. The frequency of $y[n]$ is the same as the input frequency and the amplitude and phase can be determined from the frequency response $H(e^{j2\pi \hat{f}_d})$. This observation is easily extended to sums of complex exponentials because of linearity.

Because of these observations, the frequency response often provides the most useful characterization of a LTI system. In particular, the magnitude of the frequency response is usually the first thing to look at to understand what a LTI system does.

In this lab, we focus on computing and visualizing the frequency response of an LTI system from it's impulse response.

2 Compute the Frequency Response

Equation (2) shows how the frequency response $H(e^{j2\pi \hat{f}_d})$ is computed from the impulse response $h[n]$. In that expression, the summation ranges from $-\infty$ to ∞ and is therefore not directly computable. Fortunately, for many LTI systems the impulse response is of finite duration so that the summation can be truncated. The impulse response is strictly true for *finite impulse response* (FIR) filters with filter coefficients b_n for $n = 0, 1 \ldots, M-1$

$$h[n] = \begin{cases} b_n & \text{for } 0, 1 \ldots, M-1, \\ 0 & \text{else.} \end{cases} \tag{4}$$

Thus, for FIR filters the expression (2) for computing the frequency response reduces to a finite summation

$$H(e^{j2\pi \hat{f}_d}) = \sum_{k=0}^{M-1} h[k] \exp(-j2\pi \hat{f}_d k) \tag{5}$$

which can be computed without problems.

There exist LTI system with infinitely long impulse responses; these are referred to as *infinite impulse response* (IIR) filters. We can approximate the frequency response of IIR filters by truncating the impulse response when it's samples are close to zero. All (useful) IIR filters' impulse response will converge to to zero as n approaches infinity.

2.1 Assignment: Compute Frequency Response for a Single Frequency

Write a function `FIR_freq_resp_single` that computes the frequency response Hfd for a given impulse response $h[n]$ and a given frequency $\hat{f}_d$. Start your function implementation as follows

```
function H = FIR_freq_resp_single(h, fd)
% FIR_freq_resp_single - compute frequency response from impulse
%                        response at single frequency
%
% H = FIR_freq_resp_single(h, fd)
%
% Input Variables:
% h  - vector, samples of the impulse response of the filter
% fd - scalar, normalized frequency to compute frequency
%      response at (0 <= fd <= 0.5)
```

```
%
% Output Variable:
% H - complex scalar, frequency response at frequency fd
```

Verify that your function works correctly using the following test script. The total error you observe must be very, very close to zero.

```
%% Test case: 3-pt averager
% expected values at four frequencies
f = [0, 1, 2, 3]/6;
expected = [1, 2/3*exp(-1j*pi/3), 0, 1/3];

% coefficients for 3-py averager
M = 3;
b = ones(1, 3)/3;

% compute sum of absolute errors - should be very close to zero
sum_errors = 0;
for n = 1:4
    sum_errors += abs(expected(n) - FIR_freq_resp_single(b, f(n)));
end

printf("Total absolute error: %g", sum_errors)
```

2.2 Assignment: Compute Complete Frequency Response

Write a function that computes the frequency response over a grid of N evenly spaced frequencies dft. Your function should wrap a for loop around `FIR_freq_resp_single` to accomplish that.

The start of your function should look like this:

```
function [H,f] = FIR_freq_resp(h, N)
% FIR_freq_resp - frequency response of FIR filter computed
%
% compute the frequency response corresponding to impulse response h
% for N evenly spaced frequency points between 0 and 0.5.
%
% [H, f] = FIR_freq_resp(h, N)
%
% Input Variables:
% h - vector, samples of the impulse response of the filter
% N - number of frequency points to compute
%
% Output variables:
% H - length N complex vector, frequency response at frequencies in f
% f - length N vector, evenly spaced frequencies between 0 and 0.5

f = linspace(0, 0.5, N);
```

```
H = zeros(size(f));
```

Verify that your function works correctly using the following test script. The total error you observe must be very, very close to zero.

```
%% Test case: 3-pt averager
% expected values at four frequencies
expected_f = [0, 1, 2, 3]/6;
expected = [1, 2/3*exp(-1j*pi/3), 0, 1/3];

% coefficients for 3-py averager
M = 3;
b = ones(1, 3)/3;

% compute frequency response at 4 points
[H, f] = FIR_freq_resp(b, 4);

% compute sum of absolute errors - should be very close to zero
sum_errors = 0;
sum_errors_f = 0;
for n = 1:length(f)
   sum_errors += abs(expected(n) - H(n));
   sum_errors += abs(expected_f(n) - f(n));
end

printf("Total absolute error H: %g\n", sum_errors)
printf("Total absolute error f: %g\n", sum_errors_f)
```

3 Visualize the Frequency Response

To understand what a system does, it is customary to make a plot of the frequency response. Recall that the frequency response is complex-valued. We saw above that the magnitude of the frequency response indicates the *gain* the system provides at a given frequency. Similarly, the phase of the frequency response indicates the phase shift that the system provides at a given frequency. Consequently, it is customary to plot the frequency in polar coordinates.

Moreover, typical systems have frequency response with gains that range from near 1 in the *pass-band* to gains that are very close to 0 in the *stop-band*. To see differences in the frequency response when gains are close to 0 (e.g., the difference between 0.01 and 0.001) it is useful to plot the $\log$ of the gain. Specifically, the gain of the frequency response is usually plotted in decibel (dB). To convert the gain of the frequency response from the linear scale to decibel, the following conversion is needed

$$\text{Gain in dB} = 10 \cdot \log_{10}(|H(e^{j2\pi \hat{f}_d})|^2) = 20 \cdot \log_{10}(|H(e^{j2\pi \hat{f}_d})|).$$

You may be wondering about the square inside the logarithm. It is convention to specify the gain of the frequency response in terms of the change in power (not amplitude).

3.1 Assignment: plot the gain of the frequency response in dB

Write a function `plot_freq_resp_gain`that accepts equal lengths vectors for the frequency `f` and complex frequency response `H` and plots the gain of the frequency response in dB. Your function should not return anything.

Your function should start like this

```
function plot_freq_resp_gain(f, H)
% plot_freq_resp_gain - plot gain of frequency response (in dB)
%
% plot_freq_resp_gain(f, H)
%
% Input variables:
% f - length N vector, evenly spaced frequencies between 0 and 0.5
% H - length N complex vector, frequency response at frequencies in f
```

Make sure that your plot:

- includes a grid
- the x-axis is labeled “Normalized Frequency”
- the y-axis is labeled “Gain (dB)”
- the range of the y-axis goes from -100 to 10

3.2 Assignment: plot the phase of the frequency response in degrees

Write a function `plot_freq_resp_phase`that accepts equal lengths vectors for the frequency `f` and complex frequency response `H` and plots the phase of the frequency response in degrees. Your function should not return anything.

Your function should start like this

```
function plot_freq_resp_phase(f, H)
% plot_freq_resp_phase - plot phase of frequency response (in degrees)
%
% plot_freq_resp_phase(f, H)
%
% Input variables:
% f - length N vector, evenly spaced frequencies between 0 and 0.5
% H - length N complex vector, frequency response at frequencies in f
```

Make sure that your plot:

- includes a grid
- the x-axis is labeled “Normalized Frequency”
- the y-axis is labeled “Phase (degrees)”

Test your code by running the following script

```
% plot frequency response of 3-point averager
M = 3;
b_M = ones(M, 1)/M;

[H, f] = FIR_freq_resp(b_M, 512);

subplot(2,1,1)
plot_freq_resp_gain(f, H)
subplot(2,1,2)
plot_freq_resp_phase(f, H)
```

The resulting plot should look as shown in Figure 2.

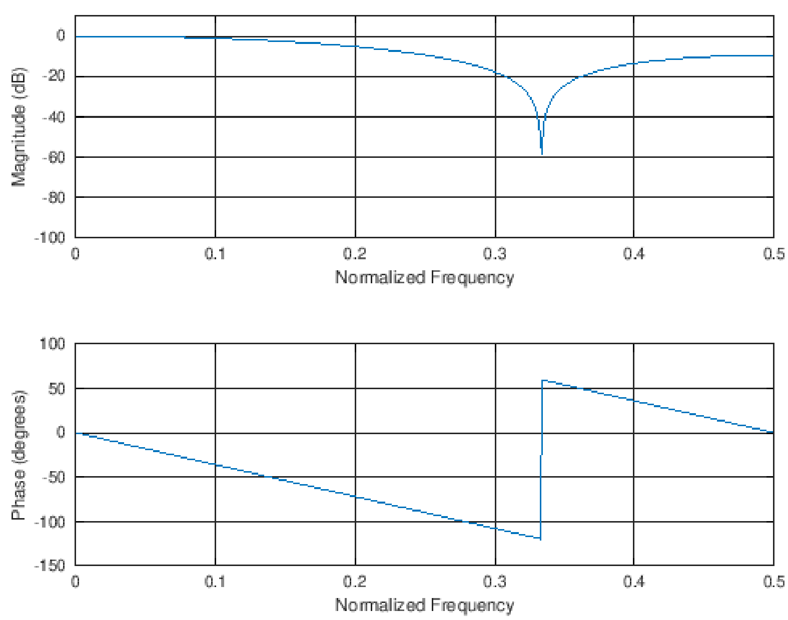

Figure 2: Frequency Response of 3-Point Averager.

4 Frequency Response of Black-Box Systems

In this final section of the lab, use the tools that you developed above to analyze three unknown LTI systems. These systems are implemented in the file `lab8systems.m`. To pass a signal `x` through one of the systems, use

```
y = lab8systems(x, sysnum);
```

The input `sysnum` select which of the three systems to use; it must be `1`, `2`, or `3`.

4.1 Assignment: find the frequency response for each system

To begin, measure the impulse response for each of the three systems. To do so, proceed as follows:

- Construct a unit-impulse signal; make the length of the signal 127 samples. The first sample has the value 1 all other samples are zero.
- Pass the impulse signal to each of the three systems and record the three impulse responses, e.g., like this for system 1.

  ```
  h1 = lab8systems(delta, 1);
  ```

  where `delta` is the unit impulse signal.
- Plot all three impulse responses in the same plot. Make sure to label the axes and to include a legend that identifies the systems.

From the impulse responses, compute the frequency response of each system. Proceed as follows for each of the three systems

- Pass the impulse response to your function `FIR_freq_resp`; use `N=512` frequency points.
- Plot the gain (in dB) and the phase (in degrees) using the functions from Section 3.

4.2 Questions

In addition to documenting the programming exercises and the plots above, in your report also answer the following questions:

1. Each of the three systems is a low-pass filter. Explain how you can tell that they are lowpass filters.
2. The cut-off frequency of the three filters is the frequency where the pass-band of the filter ends. The pass-band is the range of frequencies where the gain is approximately 1 (or 0 dB). What is the cut-off frequency for each of the filters?
3. A figure of merit for a filter is the attenuation (gain) in the stop-band. The stop-band begins where the gain drops below that attenuation for the first time. The stop-band attenuation is the largest gain in the stop-band. Measure the stop-band attenuation for each of the filters.
4. Also, measure the frequency where the stop-band begins for each of the filters.
5. Can any of the above questions be answered from the impulse response?
6. Which of the three filters would you use as a low-pass filter. Explain.

4.3 Lab Check-Off

For this lab, you will demonstrate to a TA in lab or office hours that you have completed all the lab exercises in this part of the project. Please print and fill out the lab check-off sheet and have the TA sign it after you answer questions.

Extra Credit Opportunity: If you complete all the exercises during the lab period in which they are assigned, you will receive 5 points extra credit (5% extra credit) added to your score for Project Part 1.

ECE 201 Lab Check-Off Sheet

Name: ____

Lab Section: ______________________

Lab Project Part 1

Problem 1: Compute the frequency response

Show and demonstrate that your function `FIR_freq_resp` passes the test script.

Problem 2: Visualize the frequency response

Show and demonstrate that your functions `plot_freq_resp_gain` and `plot_freq_resp_phase` produce the expected plot for the 3-point averager.

Problem 3: Black-box system

Show the plot with the three impulse responses and the three plots of the frequency responses.

Problem 4: Questions

Provide written answers to the questions in section 4.2.

TA signature and date: ____

Part 1 qualifies for extra credit: YES/NO
TA comments: