University of Washington
Department of Electrical Engineering
EE 235 Lab 3:
Unit Impulse and Continuous-Time LTI Systems

In this lab, we will investigate unit impulses and its significance and usage in LTI systems. In particular, we will revisit the time delay system from Lab 2 and implement it using LTI systems and impulse responses. We will then apply everything we learn to another audio signal problem.

Important Concepts From Lecture You Will Use In Lab 3

- Plotting unit impulse function
- Describing LTI systems using the impulse response **h(t)**
- Computing the output **y(t)** of an LTI system using the echo property of convolution

What Is Expected From You In Lab 3

- Completion of 3 pre-lab exercises (5 points)
- Completion of 2 in-lab check offs with TA (5 points)
- Completion of a lab report (10 points)

EXERCISE #1: Implementing a Unit Impulse in Matlab

In this exercise, we will practice creating a unit impulse in Matlab. If you recall from class, a unit impulse **δ(t)** is an ideal signal with very small duration and infinite height. In reality, any spiky signal of short duration and finite amplitude can be used to approximate a unit impulse. For this lab, we will approximate $\mathbf{K\delta(t - t_o)}$ as a vector that has the value K at the index where $\mathbf{t = t_o}$, and the values 0 at all other indices in the vector.

1) ***Pre-Lab Questions***:
   a) On a single plot, sketch **δ(t)**, **δ(t - 3)**, and **2δ(t - 4)** over the range $0 \le t \le 5$

   b) Suppose we wish to create a vector of zeros using a variable sampling rate **Fs**.

      i) If we want the vector to have a duration of 5 seconds (starting at time t = 0), what is the formula for the number of samples **nz** we need in terms of **Fs**?

      ii) At what index in that vector should we insert a 1 in order to implement **δ(t)**? That is, what is the formula for the index in terms of **Fs**?

      iii) How about for **δ(t - 3)**? That is, what is the formula for the index in terms of **Fs** at which we should insert a 1?

      iv) Where should we insert a 2 for **2δ(t - 4)**?

   c) **Read the Background Section** on these topics to prep for the upcoming lab exercise:

Matlab Concepts/Functions To Review	New Matlab Concepts/Functions You Will Learn
• Using the zeros function • Determining number of samples needed • Changing value(s) in a vector • Relationship between index and time • Basic plotting functions	• Graphing signals on one plot • Changing the color of a graph • Displaying a legend on the figure

2) ***Lab Exercise***: Let us now create the unit impulses from pre-lab.
   a) In your working folder, open up a script file and call it **Ex1.m**. Clear all variables and close all figures.

   b) Let us first implement the first unit impulse signal **δ(t)**:

      i) Using the **zeros** function, create a column vector of zeros called **d1** that lasts for 5 seconds using a sampling rate of **Fs** = 8000.

      ii) Using your formula from pre-lab, compute the index that corresponds to the location in **d1** that will be assigned a value 1 to represent **δ(t)**.

      iii) Using your computed index, assign a value of 1 at that index in vector **d1**.

   c) Create the corresponding time vector **t** using the length of **d1** and the sampling rate **Fs**. We will use this time vector for the other two unit impulse signals.

   d) Repeat (b) for **δ(t – 3)**. Call it **d2**, and insert the value 1 in the appropriate index.

   e) Repeat (b) for **2δ(t - 4)**. Call it **d3**, and insert the value 2 in the appropriate index.

   f) We will now graph all three signals on one plot, each with a different color:

      i) Load a new figure window.

      ii) Type in the command **hold on** to allow multiple graphs on one plot. **Refer to Background Section (3)**.

      iii) Plot **d1 vs. t**, and assign it the color blue. **Refer to Background Section (3)**. You may notice a strange artifact in the plot (almost an additional spike before the signal spikes to the desired value). This is an issue with Matlab when connecting points.

      iv) Plot **d2 vs. t**, and assign it the color red.

      v) Plot **d3 vs. t**, and assign it magenta.

      vi) Adjust the axes to that the x-axis is between -1 and 6, and the y-axis is between 0 and 3. No need to turn on the grid, but title and label your plot.

vii) Using the function **legend**, display a legend on the figure to label the signals as **d(t)**, **d(t - 3)**, and **2d(t - 4)**. **Refer to Background Section (3)**.

g) Make sure to comment your script, and then run it. Verify that the plots Matlab produces matches your plots from pre-lab.

3) ***Lab Check-Off #1 of 2***: Demonstrate your script **Ex1.m** to your lab TA and show that you know how to implement a unit impulse signal in Matlab.

4) ***Lab Report Question #1 of 7***: Suppose a student misreads step (e) and enters the value -2 instead of 2. Explain why only two impulses instead of three will show up in the final plot. What would you have to modify to see all three impulses?

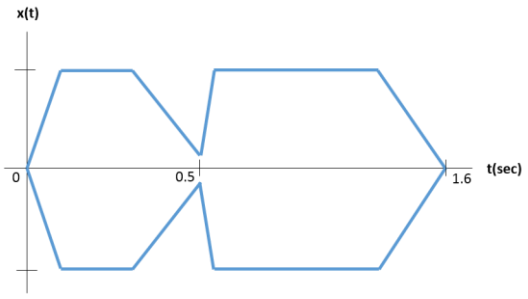
5) ***Lab Report Question #1 of 7***: A student claims that if someone misreads step (b) part (i) and enters Fs = 4000 instead of Fs = 8000, then the plot in part (g) will change. Explain why this student is incorrect.

6) ***Lab Report Question #2 of 7***: A student in class forgets to use the **hold on** command, as well as the line of code to produce a **legend**. Describe the plot that you expect Matlab to produce. Explain your answers.

EXERCISE #2: Revisiting Time Delay Transformation

In this exercise, we will revisit the time delay transformation, but this time we will be using convolution and the impulse response of an LTI system to implement it.

1) ***Pre-Lab Questions***: Recall the envelope (which we will call **x(t)**) for the **train.mat** signal:



We want to implement a time delay system using an LTI system, as shown below:

a) To design an LTI system, one simply needs to specify the impulse response **h(t)**. One way to determine the impulse response is to derive it from the input-output relationship of the desired system. The input-output relationship for a time delay system is given by **y(t) = x(t – to)**. Using the definition of the impulse response, what is the equation for the corresponding impulse response **h(t)**?

b) Using your answer from (a), sketch the impulse response **h(t)** for the case when **to = 1** over the range $0 \leq t \leq 4$.

c) Suppose the input **x(t)** is the envelope of **train.mat** above, and the impulse response is given by **h(t)** in (b). Using the echo property of convolution, find and sketch the output **y(t)** over the range $0 \leq t \leq 6$.

d) Repeat (b) and (c) for the case when **to = 3**.

e) **Read the Background Section** on these topics to prep for the upcoming lab exercise:

Matlab Concepts/Functions to Review	New Matlab Concepts/Functions
• Loading and playing sound files • Creating time samples vector • Determining number of samples needed • Relationship between time and index • Using subplot	• Using conv function

2) ***Lab Exercise***: We will now implement the LTI systems you analyzed in pre-lab and compute and plot the output **y(t)** using the **conv** function.

a) Open up a script file and call it **Ex2.m**. Clear all values and close all figures.

b) Let us first load the data file we will as the input to our LTI system

   i) Load the data file **train.mat**. This file will again have the variables: **y** and **Fs**.

   ii) To designate this sound vector as the input signal to our LTI system, we need to change name of the sound vector. Change the name by assigning the sound variable **y** to a variable called **x**.

   iii) Create the corresponding time samples vector **t_x** for vector **x**.

c) We will first create and implement the LTI system that delays a signal by **to** = 1 sec

   i) Create a column vector of zeros called **h1** that lasts a duration of 4 seconds using the same sampling rate **Fs** from **train.mat**.

   ii) Implement the impulse response you analyzed in pre-lab by inserting a 1 in the correct index in **h1**.

   iii) Create the corresponding time samples vector for **h1** and call it **t_h1**.

   iv) Using the **conv** function, compute the output **y1** by convolving **x** and **h1**. **Refer to Background Section (4) if needed**. Also, create the corresponding time vector for **y1** and call it **t_y1**.

   v) To see the relationship between the input, impulse response, and output signals, we will use subplots. On a 3 x 1 subplot, plot the following:
   - **x vs. t_x**
   - **h1 vs. t_h1**
   - **y1 vs. t_y1**

   On all plots, adjust the x-axis to be between 0 and 6 and the y-axis to be between -2 and 2. Do not forget to title and label all three of your plots.

d) Comment your script, and run what you have so far to verify your results from pre-lab. Feel free to play the output sound **y1** on the COMMAND window to verify the signal does indeed get delayed by 1 second.

e) Repeat (c), but this time create a separate LTI system to delay the sound file by **to** = 3 sec. Use the variables **h2** and **t_h2** for your impulse response and the variables **y2** and **t_y2** for the output signal. Plot everything on a separate figure window.

3) ***Lab Check-Off #2 of 2***: Demonstrate your script **Ex2.m** to your lab TA and show that you know how to implement an LTI system and can properly use the **conv** function.

4) ***Lab Report Question #3 of 7***: Suppose we modify (2e) slighty and insert a 0.25 in the impulse response vector instead of a 1, so that we now implement $\mathbf{h(t) = K\delta(t - 3)}$, where **K** = 0.25.

   a) Describe (in words) what would happen to the output signal **y(t)** if you used this **h(t)** instead. That is, how will the graph of **y(t)** differ from (2e)? How will the sound differ from (2e) if you play the audio signal?

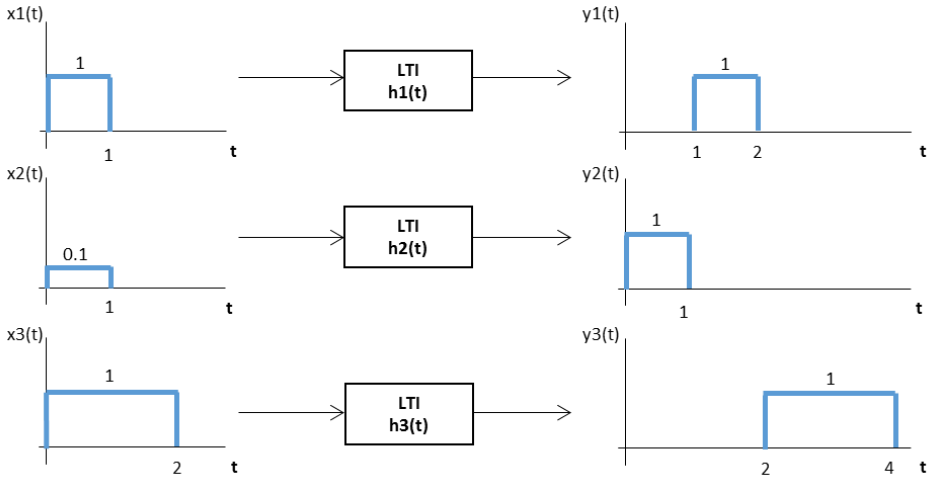
   b) What is the purpose of value **K** in **h(t)**? What does it do to the output **y(t)** if K < 1? What if K > 1?

EXERCISE #3: Audio File Realignment

In this experiment, we will perform a simple speech realignment task from a set of three signals putting together what we know about unit impulses, impulse responses, and LTI systems.

The evil former EE235 student is at it again. Using transformations in time apparently was only one of several tricks s/he had up her/his sleeve. This time, the student has again corrupted some of the TA's signals by splitting them up into three pieces. The TA started resolving the split signals last Friday, but at the last minute, the TA decided to take a trip to Hawaii and hang out at the beach (because the TA's Mom said that Seattle weather is starting to make the TA look pale). When the TA flew back from Hawaii, the TA was so tired and went straight to sleep. Your TA has been hibernating ever since and needs your help to finish this task and defeat the evil former EE235 student once and for all!

Before flying to Hawaii, the TA deduced that the sound file was split into a 1-sec piece, another 1-sec piece, and an approximately 2-sec piece. The second 1-sec piece was attenuated by 10, while the other two pieces remained unchanged. The instructor also drew out the following systems that need to be implemented to realign the audio file.



Once we compute each output, we can recover the overall audio signal simply by summing all three outputs: **y(t) = y1(t) + y2(t) + y3(t)**

1) ***Pre-Lab Questions***:
   a) Using the system and input-output diagrams above, as well as your knowledge of the echo property, determine the equations for the impulse responses **h1(t)**, **h2(t)**, and **h3(t)**?

   b) **Read Background Section (6)** and then answer the following questions. Consider the following column vectors **y1** and **y2**:

$$y1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \qquad y2 = \begin{bmatrix} 2 \\ 1 \\ 3 \\ 2 \end{bmatrix}$$

i) Suppose we wish to perform the following operation to obtain the following result:

$$y1 + y2 = \begin{bmatrix}1\\2\end{bmatrix} + \begin{bmatrix}2\\1\\3\\2\end{bmatrix} = \begin{bmatrix}1+2\\2+1\\3\\2\end{bmatrix} = \begin{bmatrix}3\\3\\3\\2\end{bmatrix}$$

Explain why the operation **y1 + y2** will not work in Matlab.

ii) We can perform the operation **y1 + y2** only if we append zeros to the end of matrix **y1**. How many zeros **nz** do we need to append to **y1** in order for the operation **y1 + y2** to work?

iii) Suppose we did not know the number of elements in either **y1** or **y2**, but only knew that the length of **y2** was bigger than the length of **y1**. In terms of **length(y1)** and **length(y2)**, what is the formula for the number of zeros **nz** we need to append to **y1** in order for the operation **y1 + y2** to work?

c) **Read Background Section (5)**, and then answer the following question. Suppose we are given three zero vectors **z1**, **z2**, and **z3**. Write the lines of code to implement the following decision statement:

- If the length of vector **y1** is greater than length of vector **y2** and the length of vector **y3**, then append **z2** to **y2** and also append **z3** to **y3**
- Else, if the length of vector **y2** is greater than the lengths of both **y1** and **y3**, then append **z1** to **y1** and also append **z3** to **y3**
- Else, if the length of vector **y3** is greater than the lengths of both **y1** and **y2**, then append **z1** to **y1** and also append **z2** to **y2**

d) **Read the Background Section** on these topics to prep for the upcoming lab exercise:

Matlab Concepts/Functions to Review	New Matlab Concepts/Functions
• Loading and playing sound files • Determining number of samples needed • Relationship between index and time • Basic plotting functions	• Using conv function • Appending to a vector

2) ***Lab Exercise***: We will now implement your design and recover the audio file.

a) Select and download your choice of one of the three top-secret audio files s1, s2, and s3 from the class website. Then, open up a script file and call it **Ex3.m**. Clear all variables and close all figures first.

b) Load the Matlab data file. It will contain the following variables: the sampling rate **Fs** and the three sound file pieces **x1**, **x2**, and **x3**.

c) Create the impulse responses you designed in pre-lab using the same sampling rate **Fs** and that each last a duration of 4 seconds. Use the variable names **h1**, **h2**, and **h3**.

d) Using convolution, compute the outputs of each LTI system. You can use the variable names **y1**, **y2**, and **y3**.

e) Now we need to add the three outputs together to form the recovered signal **y**. The outputs are of different lengths though. As discussed in pre-lab, we need to zero pad the end of each vector so that all vectors are the same length as the longest vector. Because we are not sure which vector is the longest, we will use a decision statement to handle zero padding for all the possibilities. Here is the pseudocode for the decision statement, which expands on what you already wrote in pre-lab:

```
% If the length of y1 is greater than both the lengths of y2 and y3

    % Create a column of vector zeros called z2 so that y2 will have the same length as y1
    % Use your equation from pre-lab to determine number the length of z2 based on the
    % on the lengths of y1 and y2

    % Append z2 to y2

    % Create a column of vector zeros called z3 so that y3 will have the same length as y1

    % Append z3 to y3


% Else, if the length of y2 is greater than both the lengths of y1 and y3

    % Create a column of vector zeros called z1 so that y1 will have the same length as y2

    % Append z1 to y1

    % Create a column of vector zeros called z3 so that y3 will have the same length as y2

    % Append z3 to y3


% Else, if the length of y3 is greater than both the lengths of y1 and y2

    % Create a column of vector zeros called z1 so that y1 will have the same length as y3

    % Append z1 to y1

    % Create a column of vector zeros called z2 so that y2 will have the same length as y3

    % Append z2 to y2
```

f) Now that the vectors are all of the same length, add the three signals to produce the recovered signal **y**.

g) Compute the corresponding time samples vector **t_y** for the recovered audio signal **y** by using the sampling rate **Fs**.

h) Plot overall recovered sound file **y vs. t_y**. Adjust the x-axis to be between 0 and 5, and the y-axis to be between -2 and 2. Title and label your plot.

i) Play your recovered audio file **y**.

j) Comment your script, and then run it to hear the recovered audio file.

3) ***Lab Report Question #4 of 6***: Play your recovered sound file. Can you name the character or TV show for your selected sound file? Here are some hints:

- **S1**: This space ranger action figure is one of the main characters in the first computer animated film released by Pixar Studios. This character has appeared in multiple feature films, along with his own spinoff film and TV show. The inspiration for this character is said to be astronaut Buzz Aldrin.
- **S2**: This quote comes from a 1980 space opera film, often cited as the best film in the entire franchise. One of the most well-known quotes of that film comes from this pivotal antagonist and warrior of the dark side.
- **S3**: This animated TV character has been on the air for a very long time, probably longer than most of you have been alive! This family patriarch has been described by some as the "the greatest comic creation of all time" and one of the greatest cartoon characters "of the last 20 years."

4) ***Lab Report Question #5 of 7***: All sound pieces for this exercise were column vectors. What would happen if **y2** was a row vector? Explain why step (2f) would not work. How would you resolve the issue so that you can add all the outputs **y1**, **y2**, and **y3**? Note: there are several answers to this question.

5) ***Lab Report Question #6 of 7***: In the last two exercises, we implemented a time delay as an LTI system by finding the corresponding impulse response **h(t)**. Can the same be done for the time scaling transformation from Lab 2? Explain why an impulse response **h(t)** does not exist for the time scaling system **y(t) = x(at)**.

FOR NEXT WEEK: LAB REPORT DUE

1) Use the lab report format on the class website
2) Turn in a PDF of your lab report (one per team) online via link on class website
3) Remember to include your M-files with your submission
4) Report is due in a week prior to the start of your next lab section.

BACKGROUND MATERIAL

1) **Matlab Review**

Concepts/Functions	Sample Code
Creating a variable	x = 1; % Scalar y1 = [2; 4 ; 6]; % Column vector y2 = [1 3 5]; % Row vector z = [3 6 9; 1 1 1]; % Matrix
Create vector with colon operator j:i:k	m = 0:2:10; % [0 2 4 6 8 10] n = 1:5; % [1 2 3 4 5]
Extract elements in a vector/matrix	a = y1(1); % Extract one element b = y1(1:2); % Extract multiple elements b1 = y1(3:end); % Extract until end of vector c = z(1,2); % Extract one element d = z(2, 1:2); % Extract multiple elements
Changing elements or storing values in a vector/matrix	y1(1) = 3; % Change one element y1(1:2) = -1; % Change multiple elements z(1,2) = 0; % Change one element z(2, 1:2) = 2; % Change multiple elements
Dimensions of a vector or matrix	length(y1) % Num elements in vector size(y1) % Dimensions (rows, columns) size(z, 1) % Num of rows size(z, 2) % Num of columns
Transposing a vector/matrix	a = y1';
Scalar multiplication	B = 2*A; % Perform scalar operation 2A
Creating time samples vector	% Given start and end time t = 0:1/Fs:5 % $0 \le t \le 5$ with Fs = 2 % Using vector length t_x = (0:length(x)-1) * (1/Fs) % Vector is x

Relationship between index and time:
$i = t \times F_S + 1$
	% Extracting t = 5 and storing in y
	index = 5 * Fs + 1;
	y = x(index);
	
	% Accessing t = 5 and changing value to 1
	index = 5 * Fs + 1;
	x(index) = 1;
	
	% Extracting $0 \le t \le 5$ and storing in y
	start_index = 0 * Fs + 1;
	end_index = 5 * Fs + 1;
	y = x(start_index:end_index);

Using zeros function	x = zeros(10, 1); % Column vector with 10 elements
Opening a new figure window	figure;
Using a 2 x 1 subplot and plotting on 1st figure	subplot(2, 1, 1); plot(……..);
Plotting a signal x vs. t	plot(t, x);
Turn on grid lines	grid on;
Changing axes limits	xlim([0 10]); ylim([-5 5]);

Labeling axis and adding plot title	xlabel('Time');
	ylabel('x(t)');
	title('Signal x(t)');

Loading and play sound file	load file.mat; % Contains variables y and Fs sound(y, Fs);

2) **Review: On Sampling Rate and Determining Number of Samples Needed**
   - Recall what sampling rate **Fs** means and how we are using it in Matlab. It represents the number of data samples per second. Suppose we create a signal using a sampling rate of Fs = 4000. This means that, per second, we have 4,000 samples of data.

   - Given a particular sampling rate **Fs**, suppose we want to create a vector with duration of **t** seconds. How many samples **m** do we need? We can create a formula by performing a simple unit conversion

$$samples = seconds \times \frac{samples}{second} + 1 \rightarrow \boxed{m = t \times F_S + 1}$$

   - The additional + 1 is needed since we are using discrete-time signals and need to count the first element.

   - Ex: Suppose Fs = 4000. How many samples **m** do we need for a signal to last 3 seconds?

$$m = t \times F_S + 1 = 3 \times 4000 + 1 = \boxed{12{,}001\ samples}$$

3) **More Plotting Commands**
   - To **plot multiple graphs** on one axis, use the **hold on** command

```
figure;
hold on;
plot(t, x);
plot(t, y);
```

   - To **change the color of your graph**, you can add a third parameter to the function call to **plot** specifying the color you want

plot(t, x, 'r'); % Plot in red

Color Specifier	Color
r	Red
g	Green
b	Blue
c	Cyan
m	Magenta
y	Yellow
k	Black
w	White

- To **add a label for each graph**, you can use the function **legend**:
  Syntax: legend(string1, string2, string3, …);
  Description: Puts a legend on the current plot using the specified strings as labels
  Usage: % Suppose you plotted x1(t) and x2(t)
  legend('x1(t)', 'x2(t)');

4) **On *conv* Function**
   - Matlab has a function called **conv** that you can use two convolve two signals x and h
     Syntax: y = conv(x, h)
     Description: Convolves x and h

   - It assumes that the sampling rate (and hence, sampling period) are the same for both signals. For this lab and the rest of this class, this will always be the case

   - In theory, the length of the output of the convolution for continuous-time signals should be: **length(x) + length(h)**. This is not the case in Matlab.

   - In Matlab, the length of output **y** is actually **length(x) + length(h) – 1**. The reason for this is because again Matlab deals discrete-time signals and not continuous-time signals. In EE 341, you will learn why the length of the output is slightly different for the case of discrete-time signals

5) **Matlab Decision Statement**
   - The format for the if-else decision statement in Matlab is similar to other languages:

```
if conditionalstatement1
        code1
        code2
        … all other code …
elseif conditionalstatement2
        code1
        … all other code …
else
        code1
        … all other code …
end
```

   - The main differences from other languages are as follows:
     - No need for curly braces to include all lines of code in a decision statement.
     - No need to enclose conditional statement in parentheses
     - A decision statement must include the statement *end*
     - There is no space between else and if in the *elseif* statement.

- An example decision statement is given below:

```
if a == b
    a = a + 1;
    c = 2;
elseif a > b && a > 1
    b = b – 1;
    c = -1;
elseif a <= b
    c = 0;
else
    c = 1
end
```

6) **On Appending to a Vector**

- Consider the following column vectors **y1** and **y2**

$$y1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \qquad y2 = \begin{bmatrix} 2 \\ 1 \\ 3 \\ 2 \end{bmatrix}$$

- We can **append** vector **y2** to vector **y1** and create a new vector called **y3** by using the semicolon delimiter as such:

y3 = [y1; y2]; → $y3 = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \\ 3 \\ 2 \end{bmatrix}$

- Appending to a vector is useful when we need to zero pad a vector with extra zeros. Zero padding can be achieved with the **zeros** function to create a vector with the desired number of zeros and appending that to a vector.

- Suppose we wish to zero pad y1 with two extra zeros. We can do this Matlab with the following code:

```
z1 = zeros(2, 1);
y1 = [y1; z1];
```

→ $y1 = \begin{bmatrix} 1 \\ 2 \\ 0 \\ 0 \end{bmatrix}$