# Signals & Systems - EE 3TP3
# Lab #2

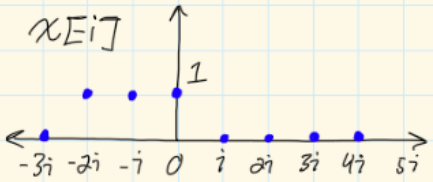
-

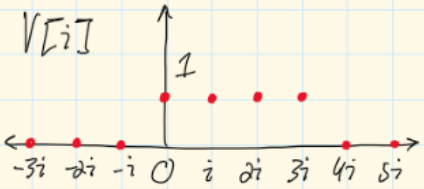
Stefan Tosti - Tostis - 400367761 - L08

2023 - 10 - 20

***Question 1***

a)




$y[\phi] = (1\times1) = 1$

$y[i] = \phi \; ; \; i < \phi$

$y[1] = (1\times1) + (1\times1) = 2$

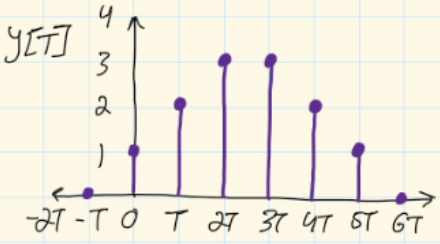
$y[2] = (1\times1) + (1\times1) + (1\times1) = 3$

$y[3] = (1\times1) + (1\times1) + (1\times1) = 3$

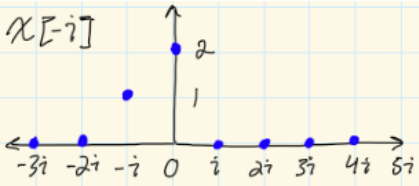
$y[4] = (1\times1) + (1\times1) = 2$

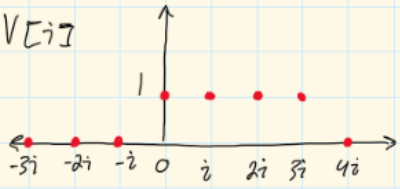
$y[5] = (1\times1) = 1$

$y[i] = \phi \; ; \; i > 5$




b)




$y[\phi] = (2\times1) = 2$

$y[i] = \phi \; ; \; i < \phi$

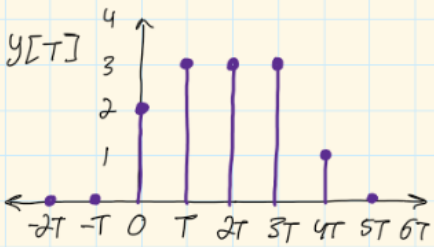
$y[1] = (1\times1) + (2\times1) = 3$

$y[2] = (1\times1) + (2\times1) = 3$

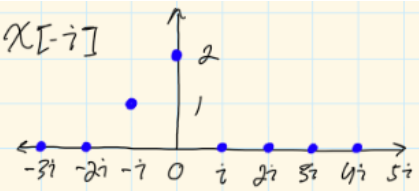
$y[3] = (1\times1) + (2\times1) = 3$

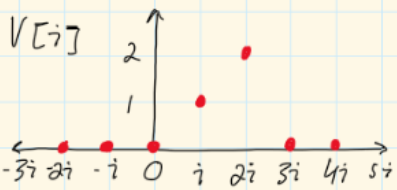
$y[4] = (1\times1) = 1$

$y[i] = \phi \; ; \; i > 4$




c)




$y[\phi] = (2\times0) = \phi$

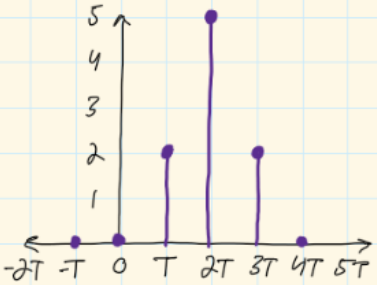
$y[i] = \phi \; ; \; i < \phi$

$y[1] = (2\times1) + (1\times0) = 2$

$y[2] = (2\times2) + (1\times1) = 5$

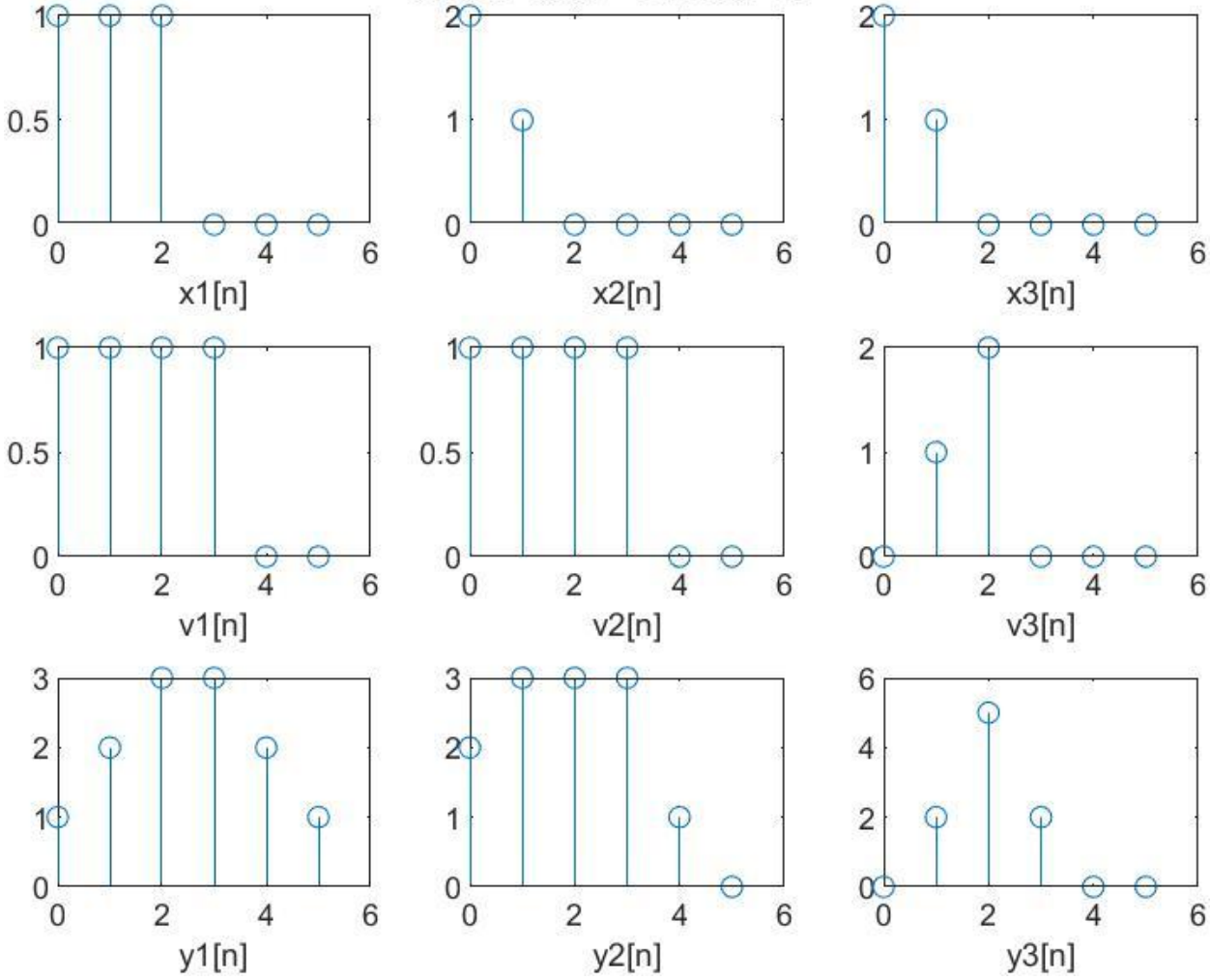
$y[3] = (1\times2) = 2$

$y[i] = \phi \; ; \; i > 3$



```
% Stefan Tosti - Tostis - 400367761
% ELECENG 3TP3 - Lab 2 - Question 1

% Prepare the Matlab enviornment
clear

% Create an instance of SimpleFunctions
f = SimpleFunctions;
t = 0:5;

% Define our functions to be used in parts a, b and c
% Similar to lab 1, we can define these as a weighted
% summation of unit step functions
x1 = f.unitstep(t) - f.unitstep(t-3);
x2 = 2.*f.unitstep(t) - f.unitstep(t-1) - f.unitstep(t-2);
x3 = 2.*f.unitstep(t) - f.unitstep(t-1) - f.unitstep(t-2);

v1 = f.unitstep(t) - f.unitstep(t-4);
v2 = f.unitstep(t) - f.unitstep(t-4);
v3 = f.unitstep(t-1) + f.unitstep(t-2) - 2.*f.unitstep(t-3);

% Use Matlabs Conv function to compute the convolution
y1 = conv(x1, v1);
y2 = conv(x2, v2);
y3 = conv(x3, v3);

% Prepare a 3x3 plot for each graph

% Plots for part A ================================================
subplot(3, 3, 1)
stem(t, x1(1:length(t)))
xlabel("x1[n]")

subplot(3, 3, 4)
stem(t, v1(1:length(t)))
xlabel("v1[n]")

subplot(3, 3, 7)
stem(t, y1(1:length(t)))
xlabel("y1[n]")
% ================================================================

% Plots for part B ================================================
subplot(3, 3, 2)
stem(t, x2(1:length(t)))
xlabel("x2[n]")

subplot(3, 3, 5)
stem(t, v2(1:length(t)))
xlabel("v2[n]")

subplot(3, 3, 8)
stem(t, y2(1:length(t)))
xlabel("y2[n]")
% ================================================================

% Plots for part C ================================================
subplot(3, 3, 3)
stem(t, x3(1:length(t)))
xlabel("x3[n]")

subplot(3, 3, 6)
stem(t, v3(1:length(t)))
xlabel("v3[n]")

subplot(3, 3, 9)
stem(t, y3(1:length(t)))
xlabel("y3[n]")
% ================================================================
```

Stefan Tosti - 400367761



## *Question 2*

Audio file recorded in this step named *"my_speech_clip.wav"* can be found in the file submission

## *Question 3*

```
% Stefan Tosti - Tostis - 400367761
% EE3TP3 - Lab 2 - Question 2
[signal, Fs] = audioread('my_speech_clip.wav');
L = length(signal); % Number of samples in the signal.
T = 1/Fs; % Sampling period in seconds.
t = [0:L-1]*T; % Time vector in seconds.
```

## *Question 4*

```
% Stefan Tosti - Tostis - 400367761
% EE3TP3 - Lab 2 - Question 4

% Read in the audio file (from Q3)
[signal, Fs] = audioread('my_speech_clip.wav');
L = length(signal);
T = 1/Fs;
t = [0:L-1]*T;

% Calculate the echo delay in seconds
% Set Td to be 500ms, then transform into seconds
Te = 300;
Te_seconds = 300/1000;

% Next we have to compute the number of delays/sample
% Round it just to make it a whole number
Delay = round(Te_seconds / T);

% setup up alpha, adjustment to the amplitude
alpha = 0.5;

% create the echo signal next
% initialize a matrix with zeros, size is equal to original signal
signalplusecho = zeros(size(signal));
signalplusecho (1:Delay) = signal (1:Delay);
% Implement the function: fr(t) = fs(t) + alpha*fs(t - te)
for i=Delay+1 : length(signal)
    signalplusecho(i) = signal(i) + alpha*signal(i-Delay);
end

%output the signal and save it
signalplusecho = signalplusecho/max(abs(signalplusecho));
audiowrite('Q4_Echo.wav', signalplusecho, Fs);
```

## *Question 5*

```
% Stefan Tosti - Tostis - 400367761
% EE3TP3 - Lab 2 - Question 5

% Read in the audio file (from Q3)
[signal, Fs] = audioread('my_speech_clip.wav');
L = length(signal);
T = 1/Fs;
t = [0:L-1]*T;

% Calculate the echo delay in seconds
% Set Td to be 500ms, then transform into seconds
Te = 300;
Te_seconds = 300/1000;

% Next we have to compute the number of delays/sample
% Round it just to make it a whole number
Delay = round(Te_seconds / T);
% setup up alpha, adjustment to the amplitude
alpha = 0.5;

% Create an empty matrix to hold the impulse response vector
IR = zeros(Delay+1, 1);

% Set value in the IR matrix - the first value in the matrix is set
% to the value 1 as this will transfer the original audio signal
% to the output, and the rest of the IR matrix is filled with alpha
% as this is used to generate the noise
IR(1) = 1;
IR(Delay+1) = alpha;

% generate the new output signal as the convolution of IR and signal
signalplusecho = conv(signal, IR);
signalplusecho = signalplusecho/max(abs(signalplusecho));
audiowrite('Q5_Echo_IR.wav', signalplusecho, Fs);
```

Just as we did in the previous question, we begin by reading in the audio file, and defining some basic information such as… *The number of samples in the signal, L, The sampling period in second, T, The time vector in seconds, t*

I then define the echo delay, *Te,* in milliseconds, and then convert that into seconds to match the previously defined units. The number of delays per sample is computed as the time delay in seconds (*Te*) divided by the sampling period in seconds (*T*). Alpha is also arbitrarily defined as 0.5.

A vector is then created, of length Delay+1, that will hold the contents of the impulse response. The first index of the impulse response vector (*IR(1)*) is set to the value 1, this lets us copy over the original audio signal. All subsequent indices are set to the value of *Alpha* which lets us implement the noise into the signal.

Once this is finished, the audio signal is re-scaled and saved as a new file.

## *Question 6*

With the value of alpha fixed at 1, it appears that when Te ≈ 15ms the effects of the echo completely go away. At this value of Te, the audio is acceptable, though it sounds as if the volume has been amplified, this is likely because the delay of the distortion is so small that the two signals are constructive rather than destructive, resulting in an increase in volume and no (or very little) distortion.

When the value of alpha is decreased from 1, it seems as though there may have been a slight increase in the range of Te values that result in acceptable audio quality. As a test, the audio quality was compared for two samples. The first sample had alpha = 1 and Te = 50, and the second sample had alpha = 0.5 and Te = 50. It was apparent that the sample with alpha = 1 had much more distortion and echoing present, this is likely because we are increasing the amplitude of the waveform, making it more prominent. That being said, the effects of this were negligible to the range of Te that results in acceptable audio quality, as 15msec still resulted in ideal audio quality.

# *Question 7*

```
% Stefan Tosti - Tostis - 400367761
% EE3TP3 - Lab 2 - Question 5

% Read in the audio file (from Q3)
[signal, Fs] = audioread('my_speech_clip.wav');
L = length(signal);
T = 1/Fs;
t = [0:L-1]*T;

% Calculate the echo delay in seconds
% Set Td to be 500ms, then transform into seconds
Te = 300;
Te_seconds = 300/1000;

% Next we have to compute the number of delays/sample
% Round it just to make it a whole number
Delay = round(Te_seconds / T);
 % setup up alpha, adjustment to the amplitude
 alpha = 1;

 % setup the number of echos
 Ne = 3;

 % Create an empty matrix to hold the impulse response vector
 % We not have to take into account the number of echos in the size
 IR = zeros(Ne*Delay+1, 1);

 % set the first value of IR
 IR(1) = 1;

 % instead of just multiplying by alpha, we sum over all the echos
 % and implement the function from the lab manual
 for x=1:Ne
     IR(x*Delay+1) = IR(x*Delay + 1) + alpha^x;
 end
% generate the new output signal as the convolution of IR and signal
signalplusecho = conv(signal, IR);
signalplusecho = signalplusecho/max(abs(signalplusecho));
audiowrite('Q7_reverb.wav', signalplusecho, Fs);
```