ELECENG 3TP3 Signals and Systems

Lab 2 Report

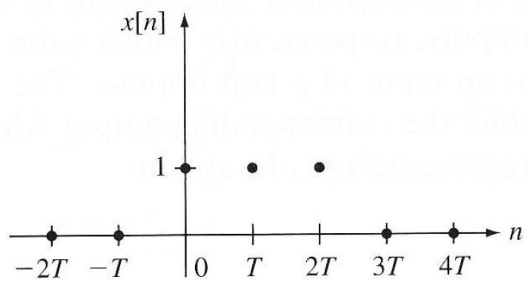
Section C01, Instructor: Prof. Jun Chen

Khaled Hassan
Hassak9
400203796

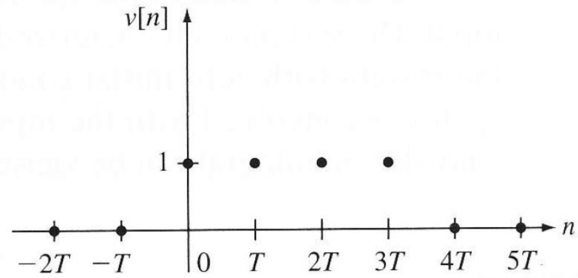
Note: I sincerely apologize for the late submission. As explained to Dr. Chen, I had a lot to go through these past couple of weeks, from projects to 4 midterms in 6 days, to falling ill. I hope that there would be no late penalty. If there is, please inform me so that I may use my MSAF to avert any late penalty. Thank you and have a wonderful day!

Experiments

Section 1: Textbook Question 2.7



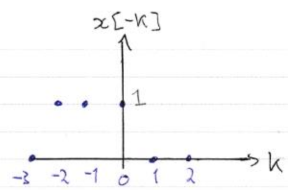
(a)

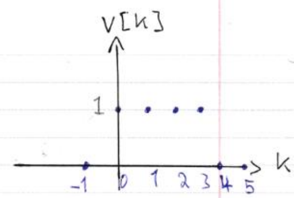


(a)

$y[n] = x[n] * v[n]$

(a)





$\tau_{L,0} = -2 \qquad \tau_{L,n} = n-2$

$\tau_{R,0} = 0 \qquad \tau_{R,n} = n$

$n < 0$ or $n > 5$: no overlap $\Rightarrow y[n] = 0$

$n=0$: overlap at $k=0$:
$y[0] = 1 \times 1 = 1$

$n=1$: overlap at $k=0,1$: $y[1] = 1\times1 + 1\times1 = 2$

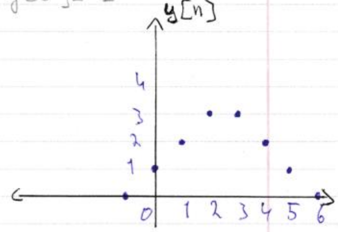
$n=2$: overlap at $k=0,1,2$: $y[2] = (1)(1)+(1)(1)+(1)(1) = 3$

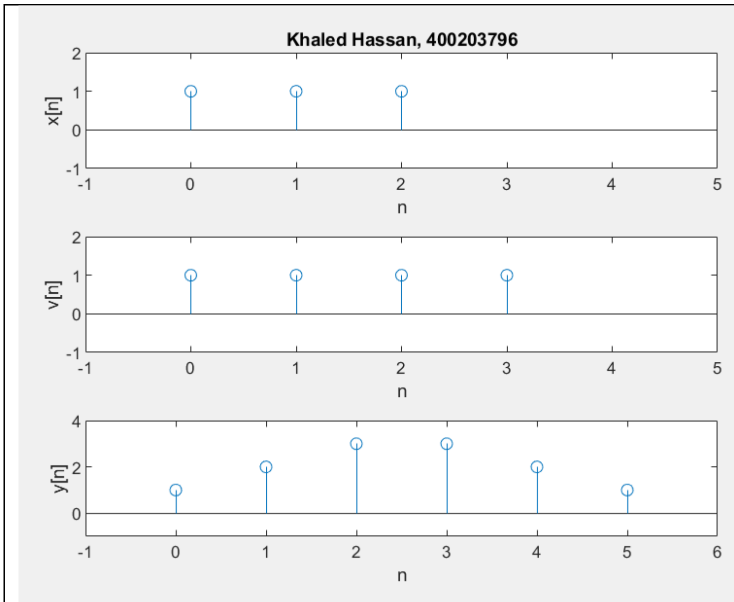
$n=3$: overlap at $k=1,2,3$: $y[3] = 3$

$n=4$: overlap at $k=2,3$: $y[4] = 2$

$n=5$: overlap at $k=3$: $y[5] = 1$

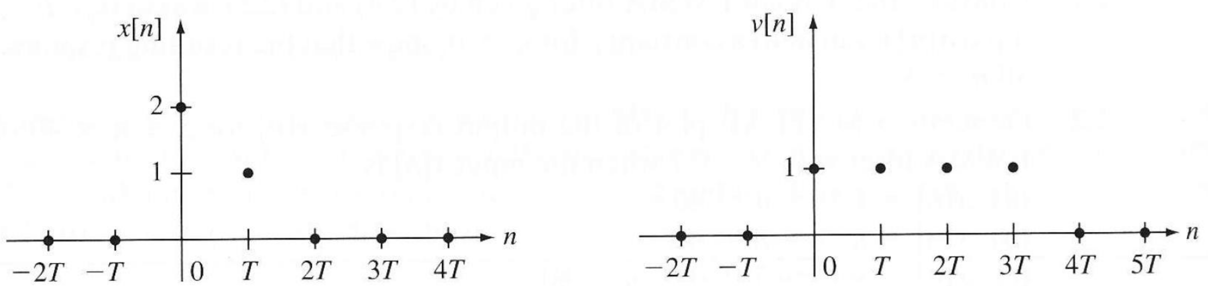
$$\Rightarrow y[n] = \begin{cases} 1, & n = 0, 5 \\ 2, & n = 1, 4 \\ 3, & n = 2, 3 \\ 0 & \text{otherwise} \end{cases}$$





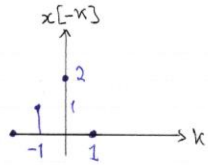

```
%% Defining x[n], v[n] and their index values
x = [1 1 1];
 v = [1 1 1 1];
 Ind_x = 0:length(x) - 1;
 Ind_v = 0:length(v) - 1;

 y = conv(x, v); %% y[n] = x[n] * v[n]
 Ind_y = 0:length(y) - 1; %% index array for y[n]
%% plotting on a subplot
 subplot(3, 1, 1);
 stem(Ind_x, x); %% Plotting x[n]
%% changing the axes to better view function values
axis([-1 5 -1 2]);
 ylabel("x[n]");
 xlabel("n");
 title("Khaled Hassan, 400203796");

subplot(3, 1, 2);
 stem(Ind_v, v); %% plotting v[n]
 axis([-1 5 -1 2]);
 ylabel("v[n]");
 xlabel("n");

 subplot(3, 1, 3);
 stem(Ind_y, y); %% plotting y[n]
 axis([-1 6 -1 4]);
 ylabel("y[n]");
 xlabel("n");
```

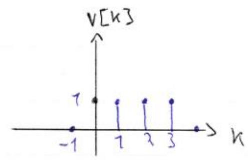
(b)



(b)

(b)





$\tau_{L,0} = -1$, $\tau_{L,n} = n-1$

$\tau_{R,0} = 0$, $\tau_{R,n} = n$

$n<0$ or $n>4$: no overlap $\Rightarrow y[n] = 0$

$n=0$: overlap @ $k=0$: $y[0] = 2(1) = 2$

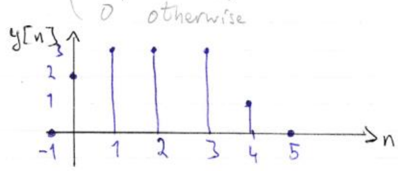
$n=1$: overlap @ $k=0,1$: $y[1] = (2)(1)+(1)(1) = 3$

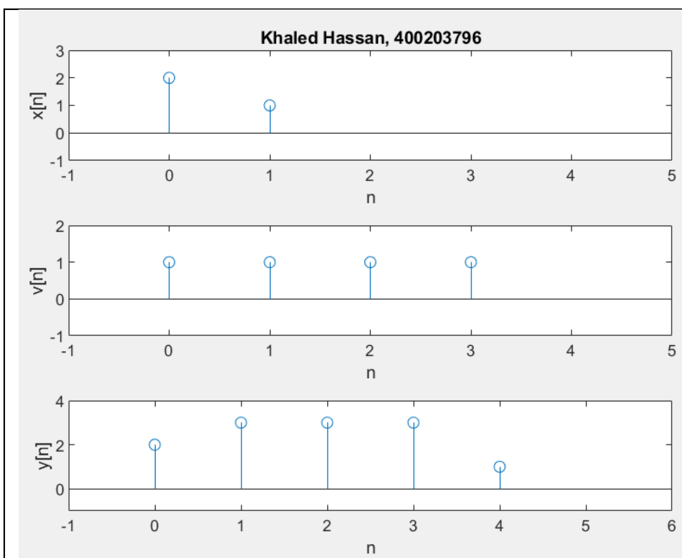
$n=2$: overlap @ $k=1,2$: $y[2] = 3$

$n=3$: overlap @ $k=2,3$: $y[3] = 3$

$n=4$: overlap @ $k=3$: $y[4] = 1$

$$y[n] = \begin{cases} 1, & n=4 \\ 2, & n=0 \\ 3, & n=1,2,3 \\ 0 & \text{otherwise} \end{cases}$$

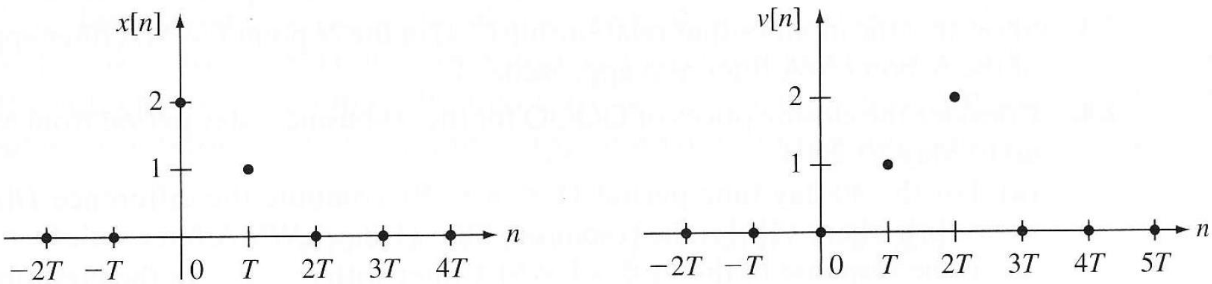




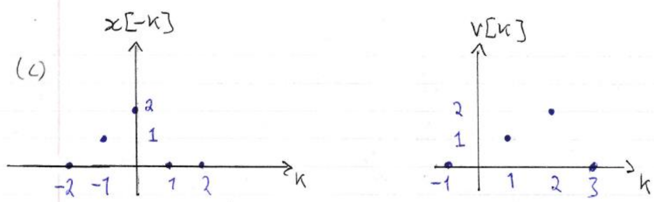

```
x = [2 1];
 v = [1 1 1 1];
 Ind_x = 0:length(x) - 1;
 Ind_v = 0:length(v) - 1;

 y = conv(x, v); %% y[n] = x[n] * v[n]
 Ind_y = 0:length(y) - 1; %% index array for y[n]
%% plotting on a subplot
 subplot(3, 1, 1);
 stem(Ind_x, x); %% Plotting x[n]
%% changing the axes to better view function
values
axis([-1 5 -1 3]);
 ylabel("x[n]");
 xlabel("n");
 title("Khaled Hassan, 400203796");

subplot(3, 1, 2);
 stem(Ind_v, v); %% plotting v[n]
 axis([-1 5 -1 2]);
```

	ylabel("v[n]");
	xlabel("n");
	
	subplot(3, 1, 3);
	stem(Ind_y, y); %% plotting y[n]
	axis([-1 6 -1 4]);
	ylabel("y[n]");
	xlabel("n");

(c)



(c)



$\tau_{L,0} = -1 \quad , \tau_{L,n} = n-1$

$\tau_{R,0} = 0 \quad , \tau_{R,n} = n$

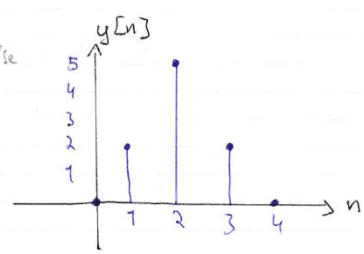
$n < 1$ or $n > 3$: no overlap $\Rightarrow y[n] = 0$

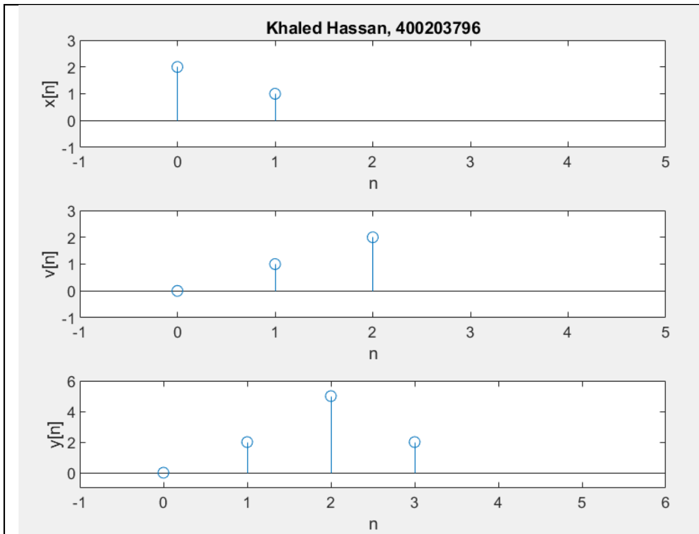
$n = 1$: overlap @ $k = 1 \Rightarrow y[1] = 2 \times 1 = 2$

$n = 2$: overlap @ $k = 1, 2 \Rightarrow y[2] = (1)(1) + (2)(2) = 5$

$n = 3$: overlap @ $k = 2 \Rightarrow y[3] = 2 \times 1 = 2$

$$y[n] = \begin{cases} 2, & n = 1, 3 \\ 5, & n = 2 \\ 0, & \text{otherwise} \end{cases}$$






```
x = [2 1];
 v = [0 1 2];
 Ind_x = 0:length(x) - 1;
 Ind_v = 0:length(v) - 1;

 y = conv(x, v); %% y[n] = x[n] * v[n]
 Ind_y = 0:length(y) - 1; %% index array for y[n]
%% plotting on a subplot
 subplot(3, 1, 1);
 stem(Ind_x, x); %% Plotting x[n]
%% changing the axes to better view function
values
axis([-1 5 -1 3]);
 ylabel("x[n]");
 xlabel("n");
 title("Khaled Hassan, 400203796");

subplot(3, 1, 2);
 stem(Ind_v, v); %% plotting v[n]
 axis([-1 5 -1 3]);
 ylabel("v[n]");
 xlabel("n");

 subplot(3, 1, 3);
 stem(Ind_y, y); %% plotting y[n]
 axis([-1 6 -1 6]);
 ylabel("y[n]");
 xlabel("n");
```

Discussion:

The procedure to generate plots of the three different y[n]'s was fairly simple and repetitive. First, 2 arrays were created to reflect the DT signals x[n] and v[n], with another 2 to hold the respective index values. The conv() function is called to perform convolution y[n] = x[n] * v[n], and all 3 signals are displayed next to each other. The results of the hand calculation match up with the resulting graph from the called MATLAB conv() function.

Section 3:

```
[signal, Fs] = audioread("speech.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
```

Section 4:

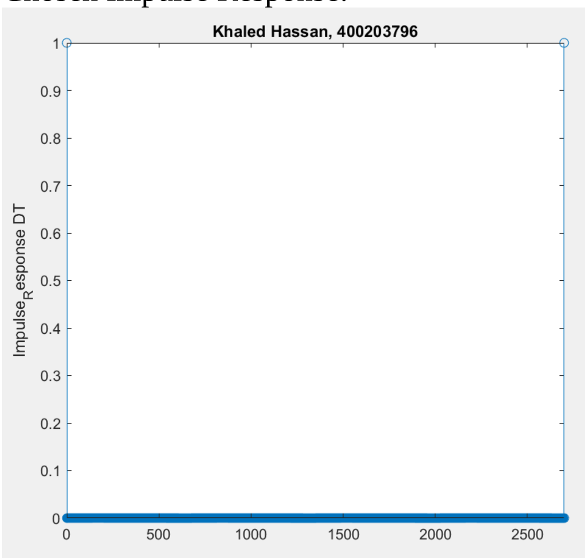
Original signal $f_s(t)$ is the signal without echo. Received signal $f_r(t)$ is the signal with echo, where $f_r(t) = f_s(t) + \alpha f_s(t - t_o)$ and α is the reducing amplitude factor. t_e is the echo delay in seconds and Te is the echo delay in milliseconds.

To be able to adequately hear the echo, a Te and α of 3000 and 0.8 were arbitrarily chosen. The new signal, the one with an echo, is made by adding zeros to the beginning of the signal vector to simulate a delay. The new signal is then truncated and added to the original signal vector.

```
[signal, Fs] = audioread("speech.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

 Te = 2700; %% chosen echo delay, in msec
 a = 0.8; %% chosen reducing amplitude factor

 delayed_signal = [zeros(Te, 1); signal]; %% adding zeros to the beginning of the signal to create a
delayed signal
 delayed_signal = delayed_signal(1:L, 1); %% truncating the array
 echo_signal = signal + a .* delayed_signal; %% applying given equation to create a signal with echo

 echo_signal = echo_signal /max(abs(echo_signal)); %% rescaling to avoid errors
 audiowrite("speechwithecho.wav", echo_signal, Fs); %% creating new audio file of echo signal
```

Section 5:

Chosen Impulse Response:




```
[signal, Fs] = audioread("speech.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

 echo_delay = 2700;
%% creating the DT Imuplse Response
 Impulse_response = zeros(L, 1);
 Impulse_response(1) = 1;
 Impulse_response(echo_delay) = 1;
%% Convolution of the signal with the Impulse Response
 echo_signal = conv(signal, Impulse_response);
 echo_signal = echo_signal/max(abs(echo_signal));
 audiowrite("speechwithecho_conv.wav", echo_signal, Fs);

%% Plotting the impulse response
Indices = 0:L-1;
stem(Indices, Impulse_response);
axis([0 2700 0 1]);
title("Khaled Hassan, 400203796");
ylabel("Imuplse_Response DT")
```

The impulse response created contains two impulse values: at index 1 and at index 2700, the chosen value of the delay. It was created by first creating an array of zeros, then changing the values at the aforementioned indices to 1. The length of the impulse response is the same as the length of the original signal. When convoluted with the original signal, that yields the same signal with echo as we got in the previous section.

Section 6:

```
[signal, Fs] = audioread("speech.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

 Te = 550; %% chosen echo delay, in msec
 a = 1; %% chosen reducing amplitude factor

 delayed_signal = [zeros(Te, 1); signal]; %% adding zeros to the beginning of the signal to
create a delayed signal
 delayed_signal = delayed_signal(1:L, 1); %% truncating the array
```

```
 echo_signal = signal + a .* delayed_signal; %% applying given equation to create a signal with
echo

 echo_signal = echo_signal /max(abs(echo_signal)); %% rescaling to avoid errors
 audiowrite("speechwithecho_lowTe.wav", echo_signal, Fs); %% creating new audio file of
echo signal
```

The code used here was more or less the same code used in section 4, but with changing the variables Te and α.Through trial and error (holding α = 1 and reducing Te), I found that the quality of the audio becomes much more acceptable when Te is at around 550 msec.

Since α is a reducing amplitude factor, and as can be deduced from the given equation, decreasing it decreases the delay in the received signal compared to the sent signal. Therefore, under lower α values, we can allow for a higher Te while still maintaining an acceptable audio quality.

Section 7:

```
%% To create Reverberated Signal
[signal, Fs] = audioread("speech.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

Te = 2700; %% chosen echo delay, in msec
 a = 1; %% chosen reducing amplitude factor
Ne = 3; %% number of echoes
for I = 1 : Ne %% FOR LOOP used to iterate through the following code
    delayed_signal = [zeros(i.*Te, 1); signal]; %% adding zeros to the beginning of the signal to create a delayed signal
    delayed_signal = delayed_signal(1:L, 1); %% truncating the array
    echo_signal = signal + a^i .* delayed_signal; %% applying given equation to create a signal with echo
end
echo_signal = echo_signal /max(abs(echo_signal)); %% rescaling to avoid errors
 audiowrite("speechwithecho_Rev.wav", echo_signal, Fs); %% creating new audio file of echo signal

%% Repetition of part 6
[signal, Fs] = audioread("speech.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

Te = 400; %% chosen echo delay, in msec
```

```
 a = 1; %% chosen reducing amplitude factor
Ne = 3; %% number of echoes
for i = 1 : Ne %% FOR LOOP used to iterate through the following code
    delayed_signal = [zeros(i.*Te, 1); signal]; %% adding zeros to the beginning of the signal to create a delayed signal
    delayed_signal = delayed_signal(1:L, 1); %% truncating the array
    echo_signal = signal + a^i .* delayed_signal; %% applying given equation to create a signal with echo
end
echo_signal = echo_signal /max(abs(echo_signal)); %% rescaling to avoid errors
 audiowrite("speechwithecho_Rev6.wav", echo_signal, Fs); %% creating new audio file of echo signal
```

Reverberation was achieved by using a for loop to implement the summation equation given:

$$\sum_{i=1}^{N_e} \alpha^i fs(t - it_e),$$

where N_e is the number of echos.

The above code yields a repeated echo that can be heard in the newly created audio file. To repeat part 6, I kept $\alpha = 1$ and kept decreasing the values of Te until I found that at around Te = 400, the signal becomes acceptable once more, with minimal echo. Once again, decreasing α reduces the extent to which the delay affects the signal, which allows us to use higher manual values of Te and still experiencing a lower effect of the delay.