Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020

DIGITAL COMMUNICATIONS LAB

Experiment 2

Handling noisy channels: Matched filters

Introduction

This is the second experiment in the Digital Communications Lab. In this experiment, we start studying the effects of non-ideal channels; we focus on Additive White Gaussian Noise (AWGN) channels in this experiment. We investigate the effect of AWGN channels on signals, and we see how we can design the best receiver to detect signals in the presence of AWGN.

About the lab

The digital communications lab consists of a set of experiments which introduce essential digital communication concepts to students. The lab consists mainly of MATLAB-based experiments in which the students complete the required tasks of the experiment by writing MATLAB codes. Some experiments of the lab may consist of hardware-based experiments which are performed in the digital communication laboratory on campus.

Location of the digital communication laboratory on campus:

2nd floor, Electrical Engineering Building.

Instructions for completing the experiment

The following describes the procedure for completing MATLAB-based experiments in the digital communication lab. Other experiments that are not MATLAB-based may have different instructions which will be described in the DOCX documents of those experiments.

- An experiment consists of 1) a DOCX document explaining the experiment, 2) a link to a video explaining the experiment, and 3) the MATLAB files required to complete the experiment
- Each experiment typically consists of several parts.
- The DOCX document is your starting point. It explains in detail the requirement of each experiment part.
- The accompanying video explains the experiment. The explanation is mainly based on the experiment DOCX.
- he MATLAB files are used by the student to complete the experiment. The main MATLAB file for the experiment is usually named LabX_script.m with X being the experiment number. For example, the main file for Experiment 1 would be called Lab1_script.m. This is the first file that you should open and start working on. Inside that file, there can be referrals to other MATLAB files that you need to complete.
- Follow the instructions in the DOCX file and the explanation video to fill the provided MATLAB files. The experiment is completed once you generate the outcomes required in all parts of the experiment.

Lab submission and discussion

- This experiment is due on the weeks **from 21st of November to 5th of December.**
- Please bring a **printed copy of your codes** and **the submission pages (the Submission Submission Pages below) of this lab DOCX with your answers in it** (your answers can be handwritten, but it is preferable to type them).
- Please submit a copy of your codes in the lab submission email shown below. The time to send the codes is **one minute before the start of your lab.**

  Email: eee.digitalcomm.fall20@gmail.com
- There will be individual discussions with students during lab hours. Discussions will be based on the files submitted by the students before the deadline.

Lab policy regarding grading and cheating

Grading

- Each experiment is graded out of 20 marks. If you sum the total grades of the lab, it would be larger than 20. This means that you have a chance to miss some parts of the lab and still get a full mark.
  - The grading is very lenient. Typically, it would be **very hard to grade that is less than 16**.
  - Any student who does the whole lab by themselves are expected to get a grade of 18 or more, without even getting everything right.

Cheating

- In exchange for this grading policy, cheating will not be tolerated.
- If a student cheats in **any part in an experiment**, he/she will get **a zero for that entire lab**.
- If a student cheats in **two or more experiments**, he/she will get **a zero in the entire lab**.
- The reason behind this policy towards cheating is the following: detecting cheating cases requires an enormous amount of work by the instructors. Given how lenient the grading of the lab is expected to be, there is no justification for a student to cheat (this is in addition to the fact that there is no justification for cheating in general).

Note: it is okay to talk with your friends about the experiment if you feel that you are stuck somewhere. But please be aware that doing any of the following **is considered cheating, and all involved students in these cases are treated as cheaters:**

- A group of students write MATLAB codes together.
- A group of students discuss the experiment together, one student of the group writes the code, and all students use the same code.

Lab inventory

File name	Description
Lab video	**Link:** TBD
Lab2_BER.docx	Contains the details of the experiment.
Lab2_script.m	The main m-file containing the MATLAB script for the experiment.
AWGNChannel.m	Additional MATLAB function files used by the experiment script file.
GenerateSquarePulses.m	
GetFreqResponse.m	
MatchedFilter.m	

Background

Communication between transmitters and receivers typically happen over channels which has detrimental effects on the transmitted signals and therefore affects the performance of communication systems. One of the most common channels is the Additive White Gaussian (AWGN) channel. In this channel, the input signal is affected by a noise component, which is modeled as a Gaussian random variable that is added to the input signal. This is described by the following equation

$$y(t) = x(t) + n(t)$$

where $x(t)$ is the transmitted signal passing through the AWGN channel, $y(t)$ is the output of the channel which models the input signal after being changed by the noise component $n(t)$. When dealing with the sampled version of this equation, it can be written as

$$y(k) = x(k) + n(k)$$

where the only change is the discrete time index k instead of the continuous time index t. The term $n(k)$ is modeled as a Gaussian random variable with zero mean and variance equal to N_o.

Let's examine how this noise term affects the input signal. Consider the signal in Figure 1 and the effect of added noise on it. As you can see from the output signal, the signal samples are changed significantly due to the added noise samples.

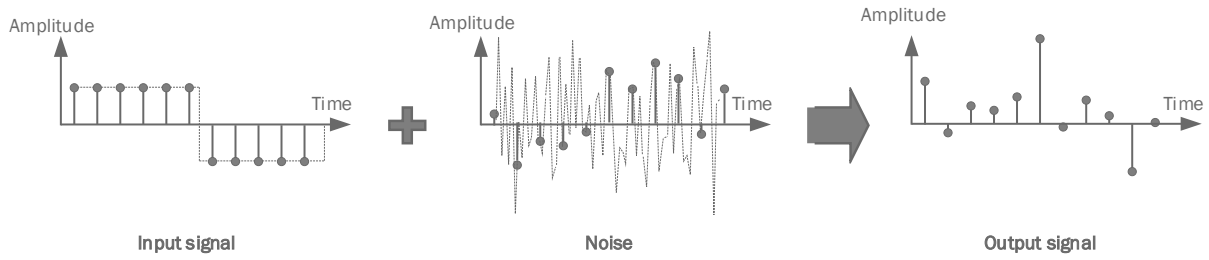


Figure 1 AWGN effect on a signal with small energy.

On the other hand, consider the signal in Figure 2 with a larger amplitude. The effect of the noise samples is smaller in comparison to the previous case: for example, all signal samples maintain their original sign.

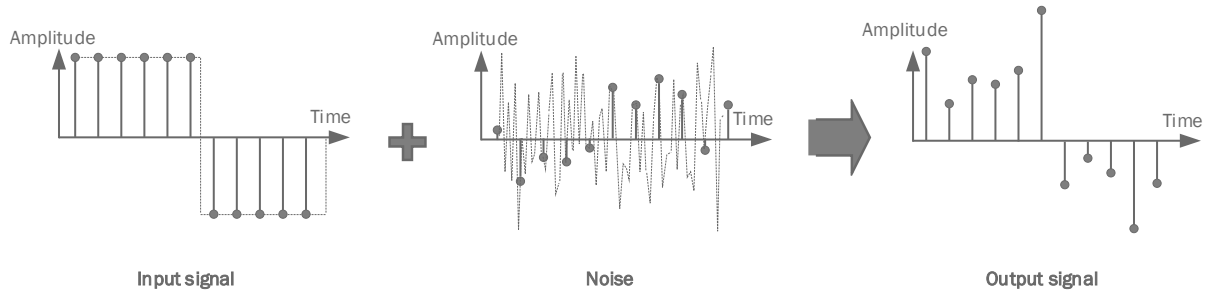


Figure 2 AWGN effect on a signal with large energy.

The previous examples show that the severity of the added noise component is dependent on the noise level compared to the signal level. Due to the fact that a signal amplitude is related to how much energy/power the signal contains, we therefore talk about the relation between the signal and noise power and/or energy levels through the concept of *Signal-to-Noise-Ratio (SNR)*.

Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020

Signal-to-Noise-Ratio (SNR)

When we talk about the effect of the AWGN channel on the signal, we usually relate that to the ratio between how much energy is in the signal compared to how much power the noise has. In a more technical definition, we define the SNR as

$$SNR = \frac{E_b}{N_o}$$

Here, $\boldsymbol{E_b}$ **is the Energy-per-bit: it is equal to the energy of the signal used to represent one bit.** On the other hand, the formal definition of N_o is that it is the Power Spectral Density (PSD) of the noise random process. This mathematically translates **to $\boldsymbol{N_o/2}$ being the variance of the noise component added to each input sample through the AWGN channel**.

The units of E_b and N_o are both Joules, and therefore SNR is dimensionless. Therefore, it is common to represent SNR in dB, which is computed as follows

$$SNR_{dB} = 10\log_{10} E_b/N_o$$

The effect of E_b on a square signal amplitude

Recall the example mentioned in Figure 1. The input signal shows a sequence of two square signals with opposite signs, where each square represents an encoding for an input bit. The process of encoding an input bit with a signal of a particular shape is referred to as *pulse shaping.* Pulse shaping is important to allow the transmitted signal to combat some of the channel impairments such as noise in the case of AWGN as we will see in this experiment.

In **Error! Reference source not found.**, each bit is pulse-shaped using a square signal. A bit of value 0 is pulse-shaped with a square signal of magnitude A, while a bit of value 1 is pulse-shaped with a square of magnitude $-A$. This form of pulse-shaping or encoding is referred to as bipolar encoding. In contrast, unipolar encoding uses a positive pulse shape to encode one bit, and encodes the opposite value of the bit with no signal/no transmission. In a sampled communication system, each square is sampled at a particular sampling rate as shown. The energy contained in one such square is given by $A^2 \cdot T_{sq}$ where T_{sq} is the duration of the square. This can be alternatively given by $A^2 \cdot T_s \cdot N_s$, where T_s is the sampling time and N_s is the number of samples taken within one square. Given that the sampling time T_s is a constant, we usually compute the energy of one square as $A^2 N_s$. Setting this value equal to the energy-per-bit E_b thus gives the amplitude of the sequence of square samples to maintain the target energy value per bit.

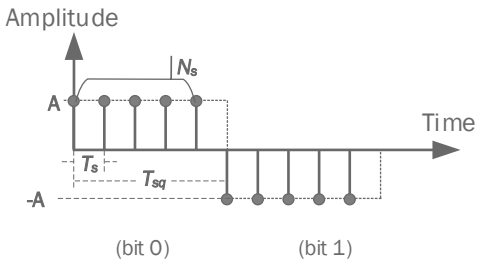


Figure 3 Bipolar pulse shaping using a square signal.

The calculations presented above is for bipolar encoding. However, for unipolar encoding, the signal used for one bit value is set to zero, consequently with zero energy. In such cases, the value of E_b refers to the *average* energy-per-bit, with the averaging being over the signals used for the two bits. Because one signal contains zero energy as we just mentioned, this leaves the energy of the other signal equal to $2E_b$ to maintain an average energy-per-bit equal to E_b. Therefore, in the case of unipoloar encoding, the amplitude of the non-zero square signal is given by setting $A^2 N_s$ equal to $2E_b$.

The effect of N_o on the noise component

As we mentioned above, $N_o/2$ represents the variance of the noise samples that are added to the signal samples in the AWGN channel. Consider the equation

$$y(k) = x(k) + n(k)$$

The value of $n(k)$ at each sample k is modeled as a Gaussian random variable with zero mean and variance $N_o/2$. As the value of N_o increases, the range of values that $n(k)$ is likely to have increases as well. Therefore, the variance of $n(k)$ plays a similar role to the increase of the signal power or energy, although not as deterministically as E_b effect on the signal amplitude. Note also that, due to the "white" assumption in the AWGN channel, noise samples $n(k)$ are independent of each other.

Before starting with the experiment

There are a few things that are important to highlight during the experiment

- This experiment deals with sampled signals. There are some parameters related to the sampling of signals which are set at the beginning of the experiment. **These parameters should not be changed throughout the experiment** since they will require corresponding changes in some later parts of the code, and changing their values does not contribute much to the demonstration of the experiment. These parameters are
    - fs: the sampling frequency, and Ts: sampling time
    - N: total number of samples used for the generation of the whole signal representing the encoding of all bits
    - t_axis: the time axis of the whole signal
    - f_axis: the frequency axis
- While the parameters fs and Ts are naturally defined and fixed for simulations dealing with sampled signals, the other parameters are fixed in this experiment because the experiment involves a lot of visual demonstrations (i.e., plots) of signals in both time and frequency domains. These visual demonstrations are already coded in the MATLAB script, and therefore the parameters N, t_axis and f_axis have to be fixed to certain values in order not to break these parts of the code.

- Regarding visual demonstrations, the experiment plots several signals in the frequency domain, which requires that you obtain the Fourier transform of these signals. The operation of applying Fourier transform to the signal and getting the frequency response can be done using the function `GetFreqResponse()` provided to you in the experiment files. This function simply applies Fast Fourier Transform (FFT) to the signal which is the typical way of getting the frequency transform of a signal in MATLAB. The experiment provides this utility function instead of using FFT right away in order to avoid any issues that may come from not knowing how FFT works which is something not exactly in the scope of the course.

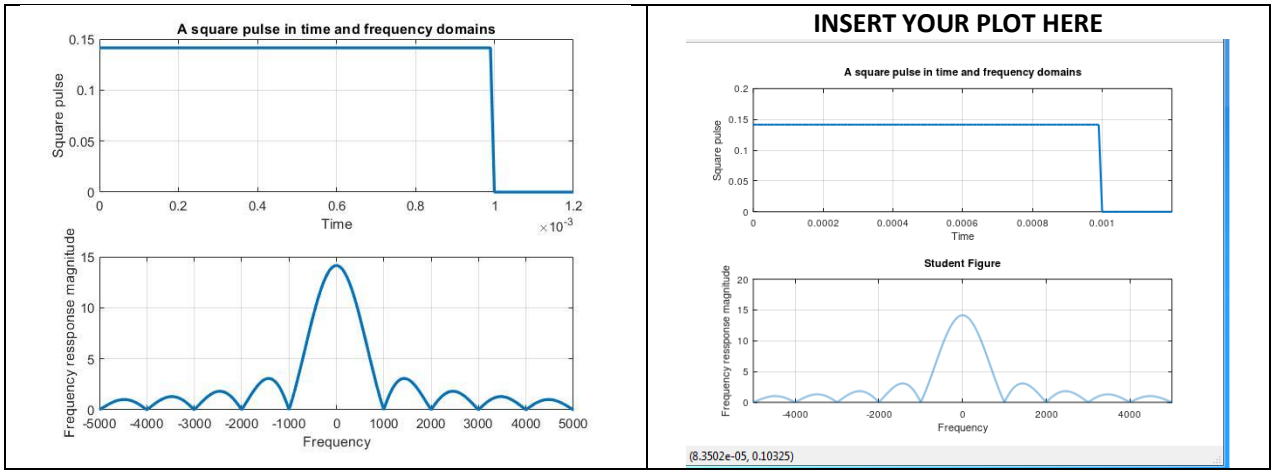
Part 1: Generate a square pulse and the AWGN channel effect

In this part, you will investigate the idea of pulse shaping using square waves, and the effect of AWGN channel on it.

Part 1-a: Generate a square wave pulse

In this part, you need to generate one square wave corresponding to one bit of value 1. The outcome of this part should be plots of how this signal looks in time and frequency domain.

EXP. (2 Marks) ***Complete PART 1-a in the experiment M-file*** **Lab2_script.m** ***and the missing implementation of the functions*** `GenerateSquarePulses`***. You should implement only the part corresponding to the unipolar case. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***



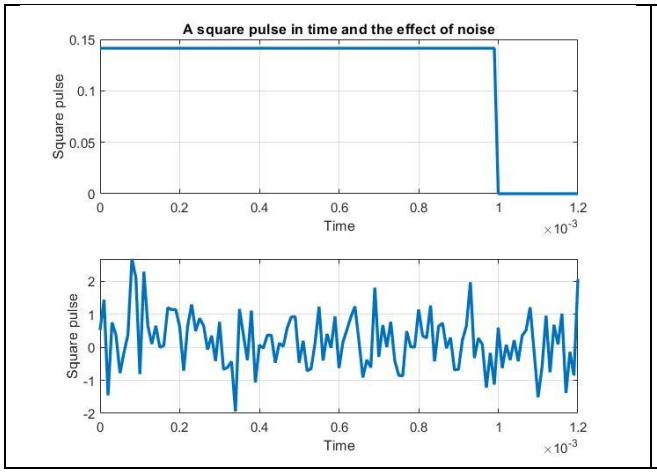
Hint: Take note of the inputs provided in the function `GenerateSquarePulses`. The vector t_axis specifies the time axis for the signal to be generated by the function. This time axis corresponds to the total

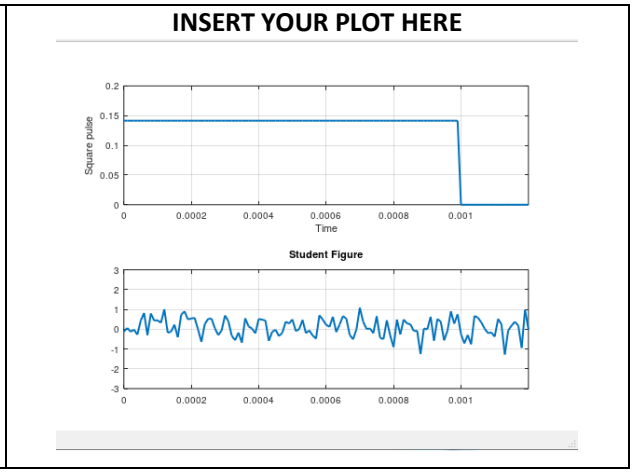
duration of the signal. However, if you examine the parameters of the square signal you want to generate, it is much shorter. This square wave is only the pulse shape of *one* input bit given in the variable x_bits. So the total signal generated by the function will always be of the same dimensions as t_axis, but the square pulse you want to generate will be only at the beginning of the signal.

Part 1-b: See the effect of the noise on the signal

Now that you've created a square pulse signal, let's pass the signal through an AWGN channel. In this part, you will pass the signal you created previously through an AWGN channel with a particular noise level N_o. The noise level N_o is specified by a particular $SNR = E_b/N_o$ value and the value of energy-per-bit E_b.

EXP. (2 Marks) ***Complete PART 1-b in the experiment M-file*** **Lab2_script.m** ***and the missing implementation of the functions*** `AWGNChannel`***. After completing this part, insert the plots that were generated in the following table; it should be similar to the provided plot.***





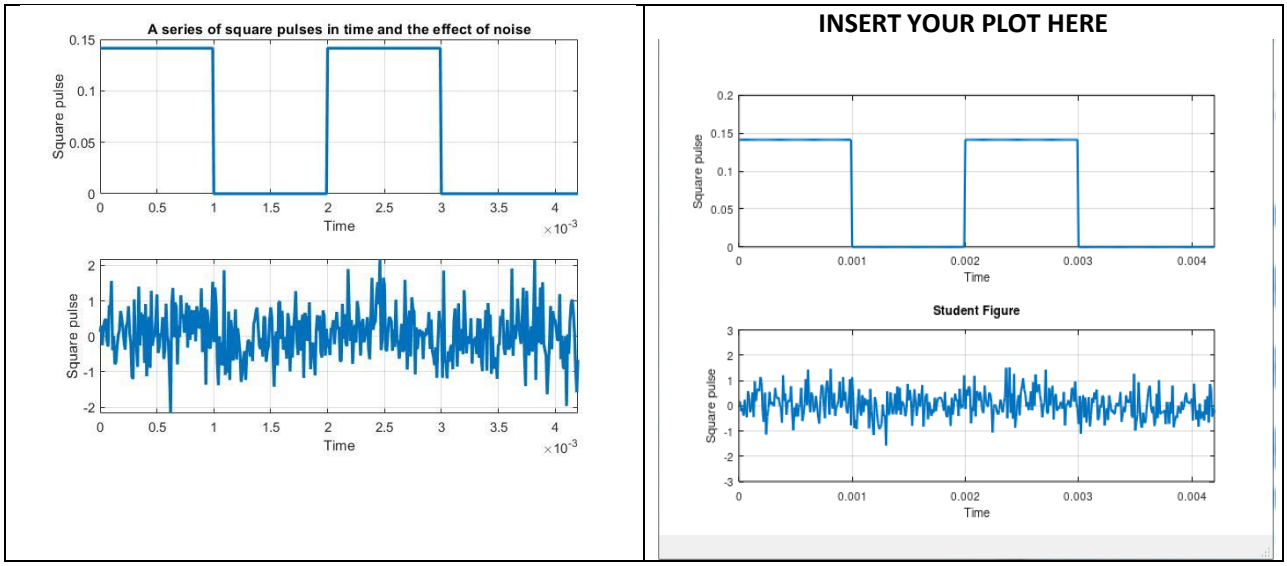
Part 1-c: See the effect of the noise on a sequence of square pulses

Now let's see the effect of noise on a sequence of squares corresponding to a bit sequence 1010. You will use the functions you completed in the previous parts to complete this part. You will also use the same energy-per-bit and SNR values.

EXP. (2 Marks) ***Complete PART 1-c in the experiment M-file*** **Lab2_script.m.** ***After completing this part, insert the plots that were generated in the following table; it should be similar to the provided plot.***

Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020



Part 2: Combatting the noise: Matched filters

In the previous part, we generated square pulses that are used as pulse shapes to encode data bits. We also investigated the effect of the noise due to AWGN channels on the signal. From looking at the plots in the previous part, does it seem possible to recover the encoded data bits from the noisy version of the square pulse signals? If you believe that it is not possible, then you'd be glad to know you're wrong! Using the right receiver for the noisy signal can still recover the data bits; the right receiver in this case is referred to as the *Matched Filter (MF)*.

The design of Matched Filters

The design of a MF is based on the signals used to represent a 0 and a 1. Consider the case of unipolar representation of a bit using square waves. The signals used for bit representation along with the resultant impulse response of the MF are shown in Figure 4. The signals used for representing a 0 and a 1 are respectively labeled as $x_0(k)$ and $x_1(k)$. The design rule for the impulse response of the corresponding MF is

$$h(k) = x_0(k) - x_1(k)$$

which is also shown below (think about the MF corresponding to bipoloar square pulse shaping).

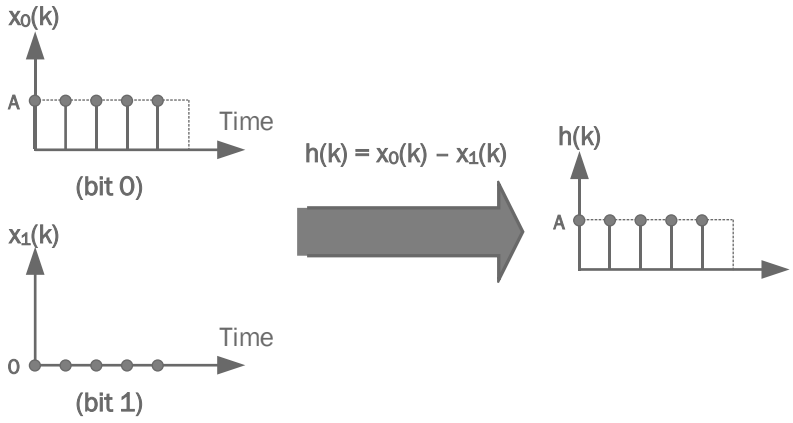


Figure 4 Unipolar square pulse shaping and the resultant MF.

The operation of the receiver based on the MF is shown in Figure 5. The channel output signal is fed into the MF filter with an impulse response $h(k)$ (what is the mathematical operation to give the output of the MF with an impulse response $h(k)$?). Then, a decision module makes a decision on the value of the input bits using the output signal from the MF.

Figure 5 The receiver operation using a MF.

In this part, we will investigate the design of the MF and how a decision at the receiver using the MF output.

Part 2-a: Design a matched filter for unipolar encoding

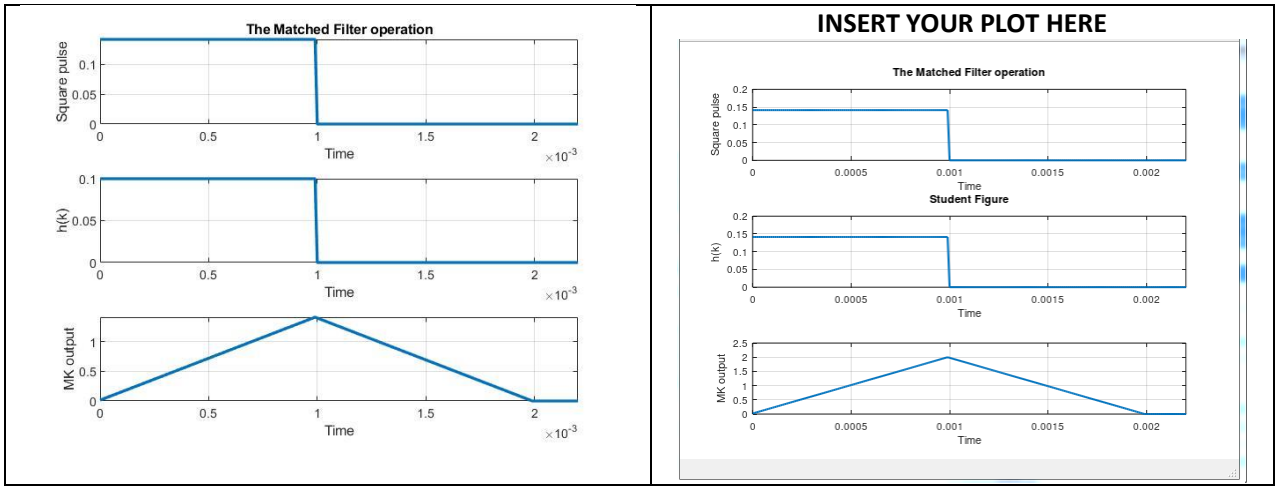
In this part, you will complete the function `MatchedFilter` which applies the impulse response of the MF to the input signal. Part 2-a of the experiment MATLAB script will then pass exactly one bit of value 1 to the function which will provide

- the impulse response of the MF,
- the output of the MF corresponding to the input signal.

EXP. (2 Marks) ***Complete PART 2-a in the experiment M-file*** **Lab2_script.m** ***and the function*** `MatchedFilter`***. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***

Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020



Part 2-b: Design the decision making module for the matched filter of unipolar encoding

For one bit of value 1, the MF designed in Part 2-a generates the signal shown in the last plot above. The decision making module of the MF receiver would then take this signal, takes **one sample** of the filter output, and makes a decision based on this sample. This sample should be the **the sample that is most resilient against added noise. Which sample would that be?** Complete the function `MatchedFilter` with the implementation of the decision module.

EXP. (2 Marks) ***Complete PART 2-b in the experiment M-file*** **Lab2_script.m** ***by completing the function*** `MatchedFilter`***.***

Part 2-c: Check the BER performance of the MF

Now that we've completed the MF implementation, let's answer the question we had at the end of Part 1: can we recover the encoded bits from the noisy signal? We test that by computing the BER performance of the MF receiver we implemented. We will do that by computing Part 2-c of the experiment (you may need to remember how we compute the BER from Experiment 1, you can even use some of the functions you used in that experiment). After completing the experiment, take note of the BER value you computed.

EXP. (2 Marks) ***Complete PART 2-c in the experiment M-file*** **Lab2_script.m.** ***After completing the experiment, write the BER value you computed in the following table.***

For E_b/N_o equal to 0 dB, the BER value is equal to:	**0.07**

Part 3: Implementing a Matched Filter for bipolar encoding

In this part, we wish to redo Part 2 but for a bipolar representation of the signal. The bipolar square pulse shaping of the input bit is shown in Figure 6.

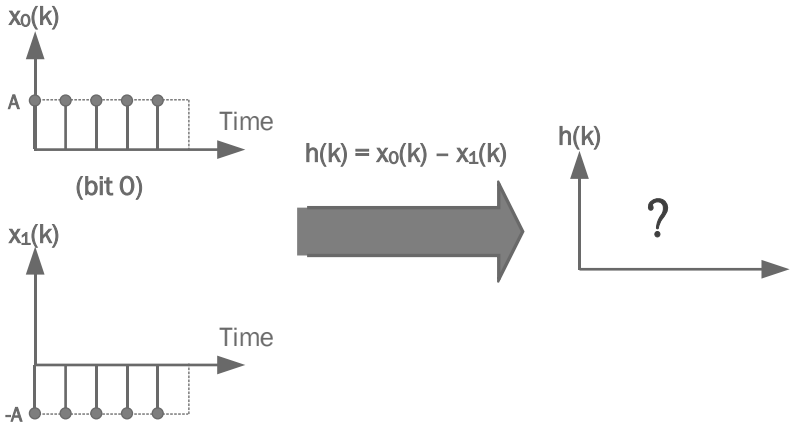
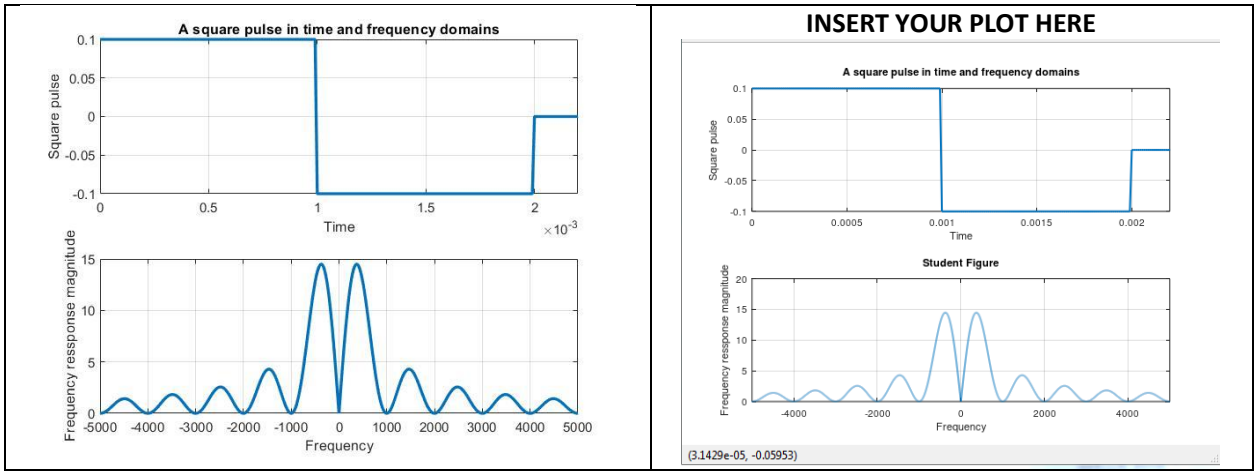


Figure 6 Bipolar square pulse shaping and the resultant MF.

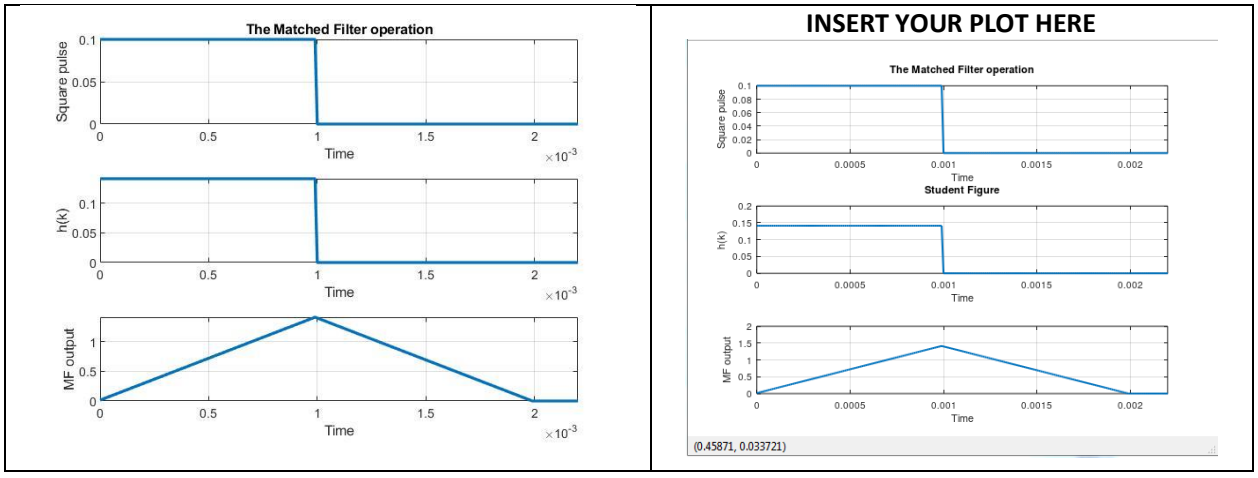
We first begin by completing Part 3-a of the experiment, where you will complete a part of the function `GenerateSquarePulses` for the bipolar encoding case. After you complete the function, the script will generate a signal corresponding to two bits 10, which you will insert in the box below.

EXP. (1 Mark) ***Complete PART 3-a in the experiment M-file*** **Lab2_script.m** ***and the function*** `GenerateSquarePulses`***. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***



Next, you will complete Part 3-b, Part 3-c and Part 3-d similar to how you completed Part 2-a, Part 2-b and Part 2-c respectively. Only you will do these parts for the bipolar encoding cases.

EXP. (2 Marks) ***Complete PART 3-b in the experiment M-file*** **Lab2_script.m** ***and the function*** `MatchedFilter`***. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***



EXP. (1 Mark) ***Complete PART 3-c in the experiment M-file*** **Lab2_script.m** ***by completing the function*** `MatchedFilter`***.***

EXP. (2 Marks) ***Complete PART 3-d in the experiment M-file*** **Lab2_script.m.** ***After completing the experiment, write the BER value you computed in the following table.***

For E_b/N_o equal to 0 dB, the BER value is equal to:	**0.13**

Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

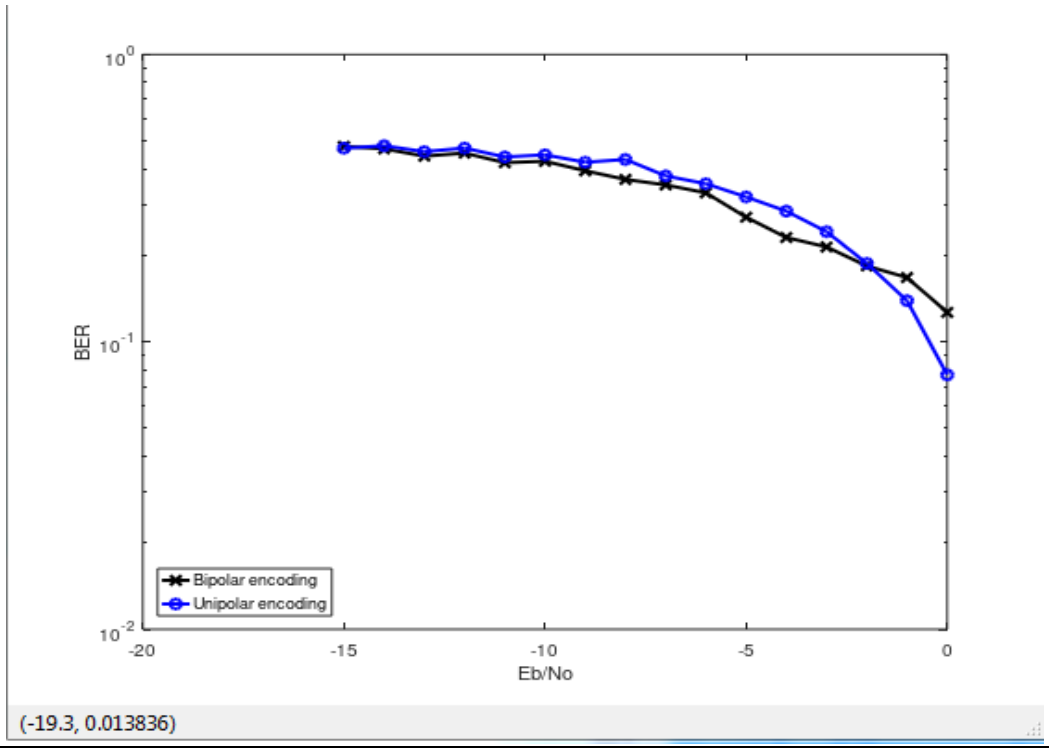
جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020

Part 4: The performance of MF of bipolar and unipolar encoding at different SNR values

For the final part of the experiment, you will now test the BER performance of both MF receivers you implemented at different SNR values. In Part 4 you are provided with a vector of different SNR values (all in dB). For each value, you need to compute the corresponding BER value for both MF receivers and store in the corresponding vectors. The MATLAB script will then generate a plot of the BER versus SNR values for both MF receivers. Which of the two MF receivers do you think will outperform the other?

EXP. (6 Marks) ***Complete PART 4 in the experiment M-file*** **Lab2_script.m.** ***After completing the experiment, write the BER value you computed in the following table. After completing this part, insert the one plot that was generated in the following table.***

INSERT YOUR PLOT HERE



Contact us if you have questions

	Contact Information
Eng. Ahmed Elshazly	**Email:** Ahmedmelshazly@alexu.edu.eg
Eng. Hossam Mohammed	**Email:**

	h.m.hassan@alexu.edu.eg **Office hours:** Saturday 3rd period
Dr. Mohammed Karmoose	**Email:** m_h_karmoose@alexu.edu.eg **Office hours:** Sunday 2nd period Monday 2nd period Thursday 3rd period **Office Location: 2nd floor, EE building**

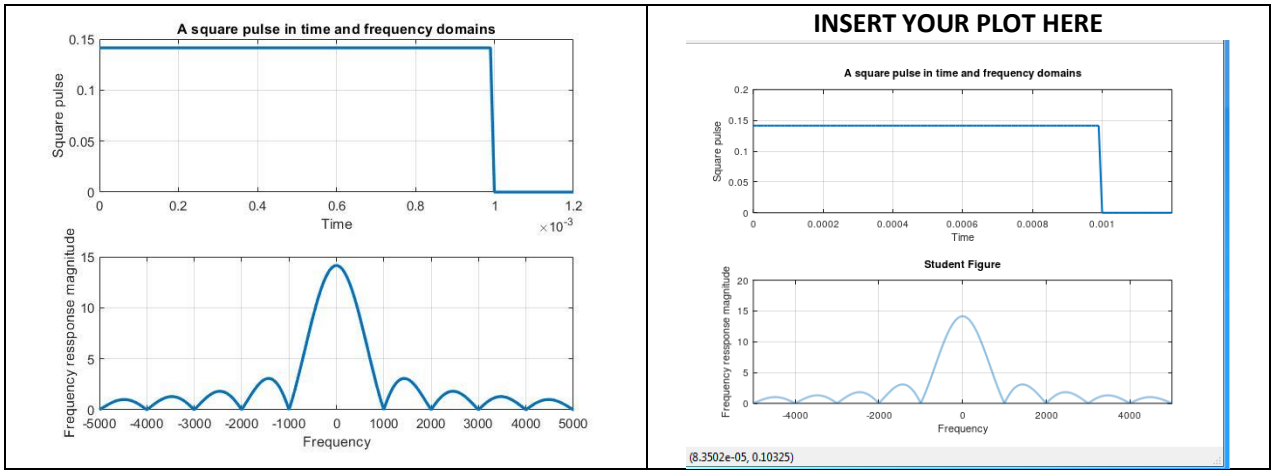
Student Submission Pages

Name:	Omar abdelazeem Mohamed abdelrahman
ID:	155
Section:	4

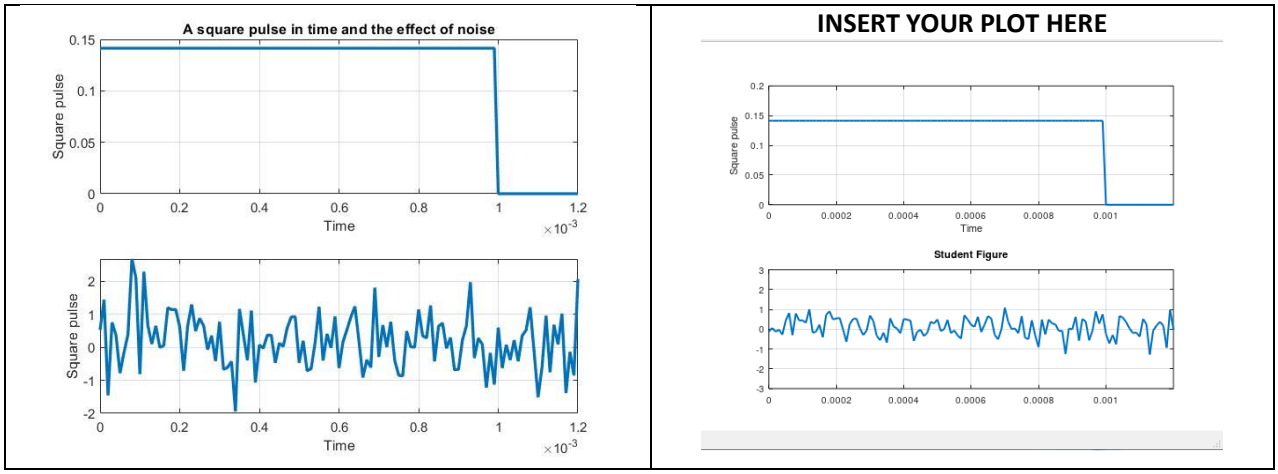
Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020

EXP. ***Complete PART 1-a in the experiment M-file*** **Lab2_script.m** ***and the missing implementation of the functions*** `GenerateSquarePulses`***. You should implement only the part corresponding to the unipolar case. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***



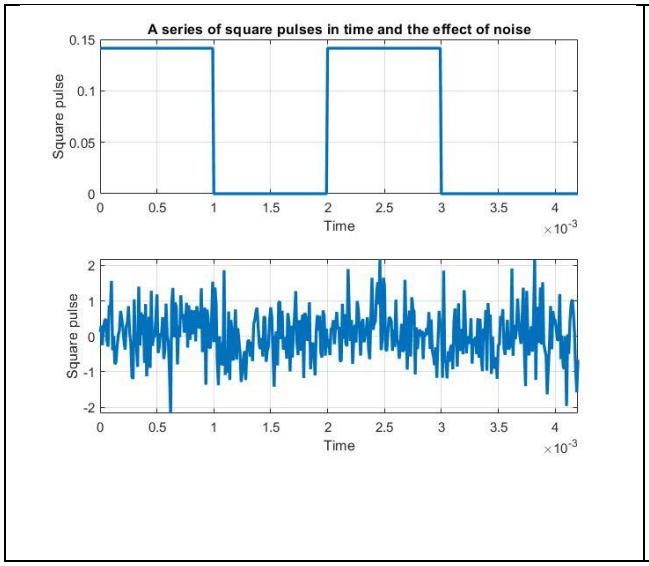
EXP. ***Complete PART 1-b in the experiment M-file*** **Lab2_script.m** ***and the missing implementation of the functions*** `AWGNChannel`***. After completing this part, insert the plots that were generated in the following table; it should be similar to the provided plot.***

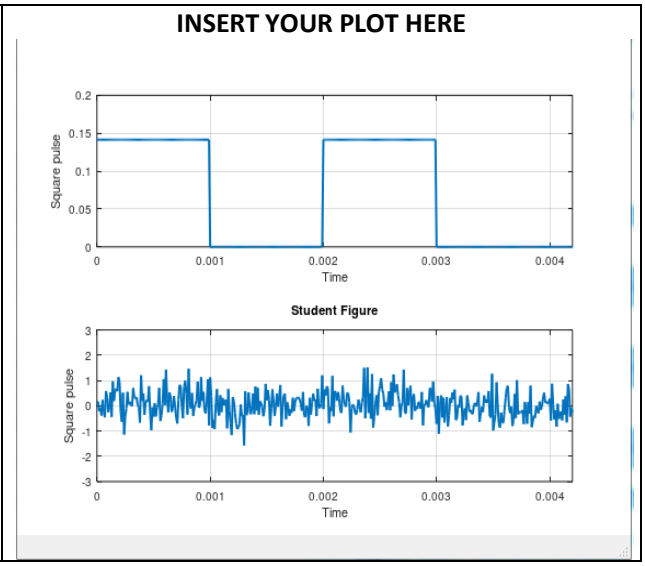


EXP. ***Complete PART 1-c in the experiment M-file*** **Lab2_script.m.** ***After completing this part, insert the plots that were generated in the following table; it should be similar to the provided plot.***

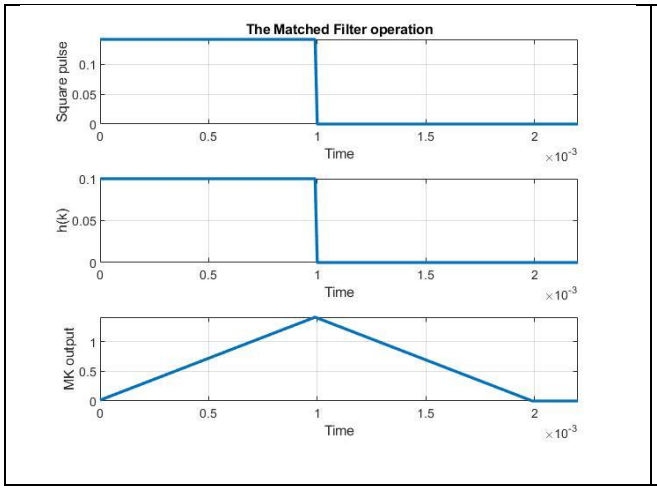
Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

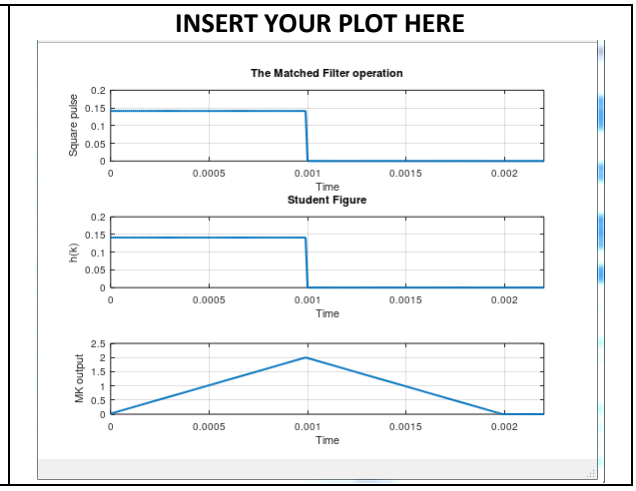
جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020





EXP. ***Complete PART 2-a in the experiment M-file*** **Lab2_script.m** ***and the function*** `MatchedFilter`***. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***





EXP. ***Complete PART 2-b in the experiment M-file*** **Lab2_script.m** ***by completing the function*** `MatchedFilter`***.***

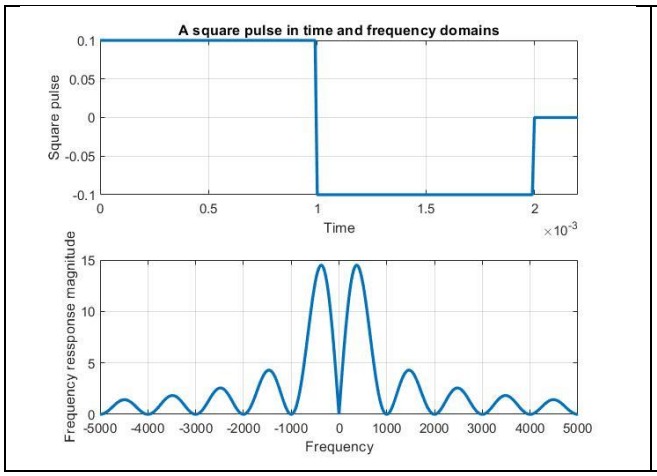
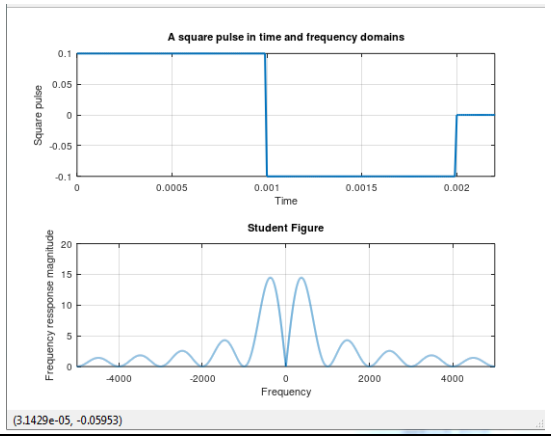
EXP. ***Complete PART 2-c in the experiment M-file*** **Lab2_script.m.** ***After completing the experiment, write the BER value you computed in the following table.***

For E_b/N_o equal to 0 dB, the BER value is equal to:	**0.07**

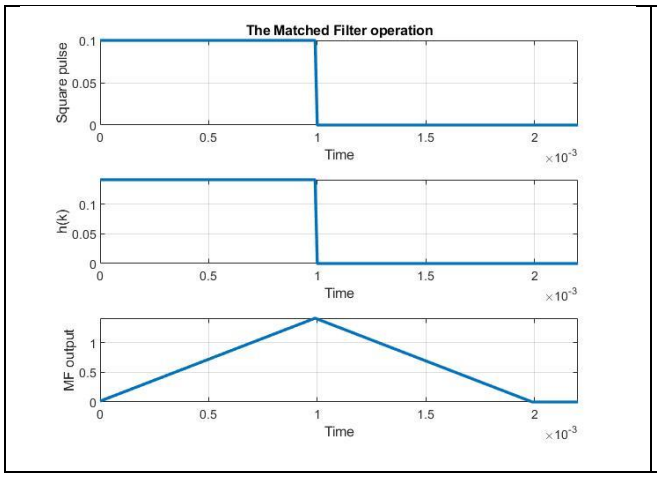
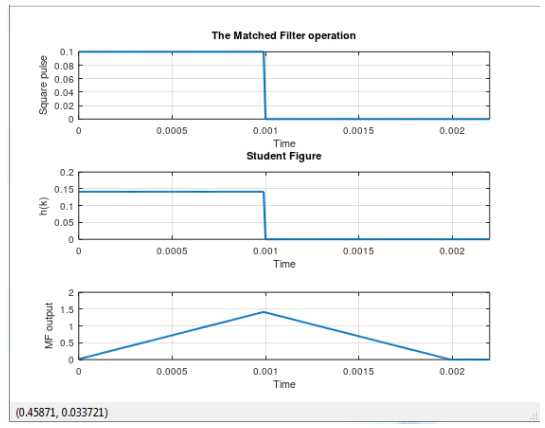
Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2020/2021

EXP. ***Complete PART 3-a in the experiment M-file*** **Lab2_script.m** ***and the function*** `GenerateSquarePulses`***. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***

	**INSERT YOUR PLOT HERE** 

EXP. ***Complete PART 3-b in the experiment M-file*** **Lab2_script.m** ***and the function*** `MatchedFilter`***. After completing this part, insert the plots that were generated in the following table; it should be identical to the provided plot.***

	**INSERT YOUR PLOT HERE** 

EXP. ***Complete PART 3-c in the experiment M-file*** **Lab2_script.m** ***by completing the function*** `MatchedFilter`***.***

EXP. ***Complete PART 3-d in the experiment M-file*** **Lab2_script.m.** ***After completing the experiment, write the BER value you computed in the following table.***

For E_b/N_o equal to 0 dB, the BER value is equal to:	**0.13**

Alexandria University
Faculty of Engineering
Electrical and Electronics Engineering Department
Fall semester, 2020/2021

جامعة الإسكندرية
كلية الهندسة
قسم الهندسة الكهربية
الفصل الدراسى الأول, 2021/2020

EXP. ***Complete PART 4 in the experiment M-file*** **Lab2_script.m.** ***After completing the experiment, write the BER value you computed in the following table. After completing this part, insert the one plot that was generated in the following table.***

INSERT YOUR PLOT HERE