In the name of God

University of Tehran

Faculty of Engineering

ECE

Digital Communication Systems

CA2

Reza Jahani
810198377

Spring 14001

Index

Abstract	**3**
Raised Cosine Pulse Generation	**4**
Transmit Signal Generation	**6**
AWGN Channel Simulation	**7**
Symbol Detection	**8**
Error Probability Computation	**9**
Repetition of Sections for different β	**10**
Conclusions	**16**

Abstract

This project is divided into 5 sections, each following a certain goal. Our overall aim is to transmit a digital signal through AWGN channel employing raised cosine pulse and to detect the symbols at the receiver's end alongside the computation of error probability.

I. Raise Cosine Pulse Generation

In this section we want to generate three Raise Cosine Pulses. Each of these 3 scenarios represent an error caused by sampling in the wrong time. First pulse with no error. Second pulse with error of 0.1T and last pulse with error of 0.2T.

As we already know:

$$p_R(t) = sinc\left(\frac{t}{T}\right)\frac{\cos(\frac{\pi\beta t}{T})}{1-\left(\frac{2\beta t}{T}\right)^2}$$

In this part we assume T=1 and $\beta = 0.5$

To present the error in sampling time we employ different time interval as instructed in the project manual.

Employing a simple code 3 pulses are plotted as below shown in figures 1.1, 1.2, 1.3

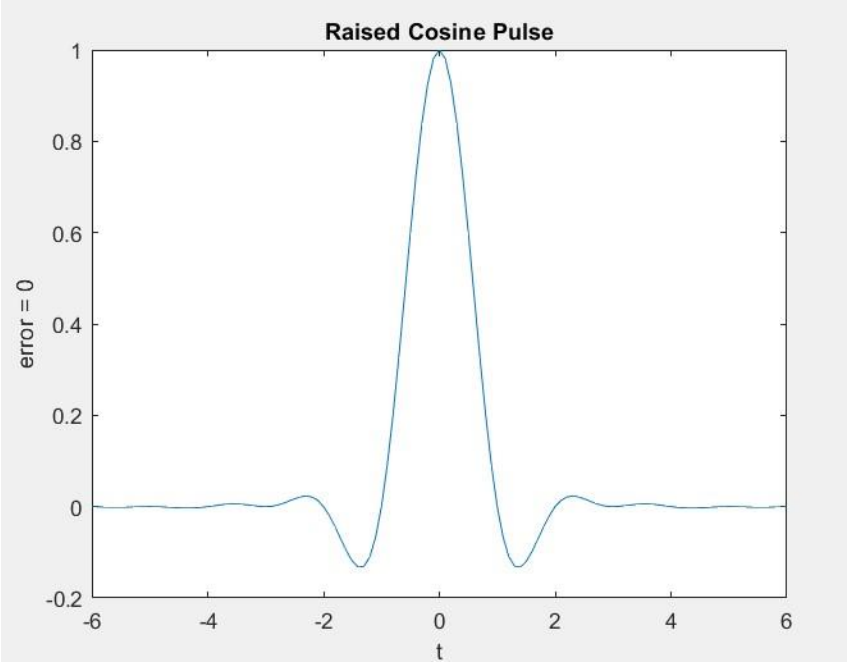

Figure 1.1: Error = 0

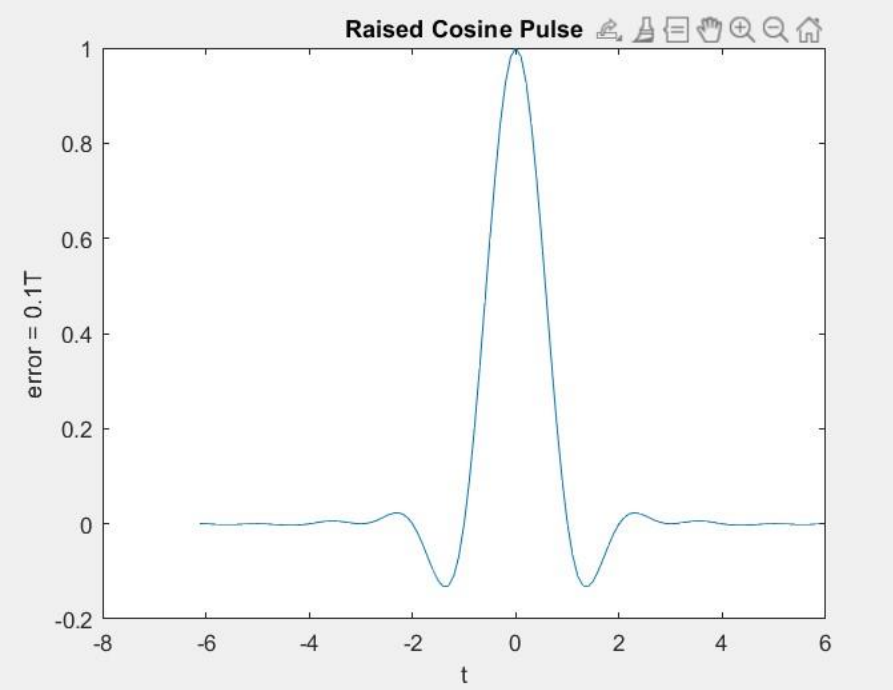

Figure 1.2: Error = 0.1T

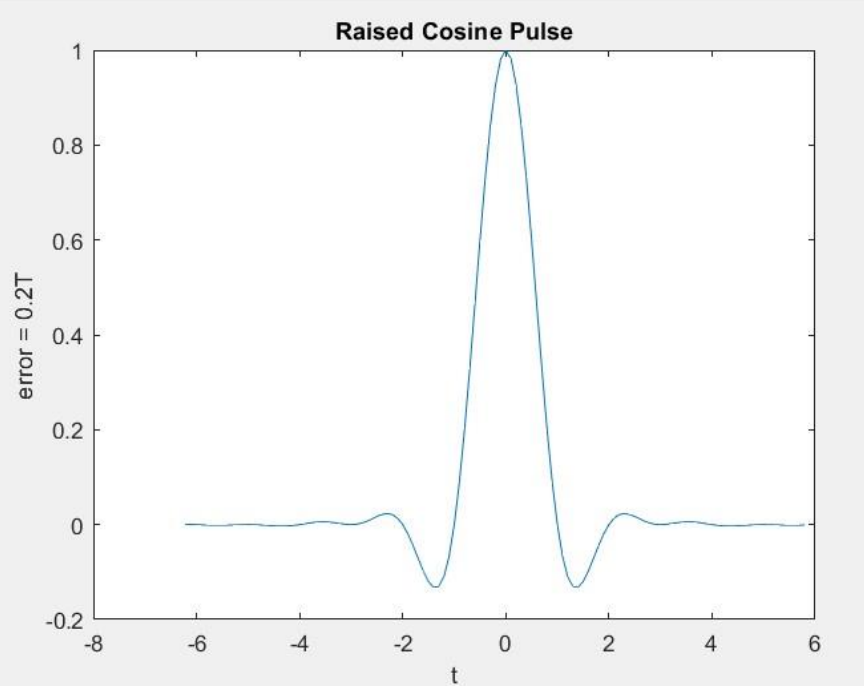

Figure 1.3: Error = 0.2T

- This section uses the file **P1.m**
- The function to generate these pulses is **RCP.m**

II. Transmitted Signal Generation

In this section, based on the instruction given in the project manual, we aim to generate the transmitted signal. After the bit chain generation and modulating it into 1 and -1, zeros are implemented among our symbols using the concept of up sampling.
After this procedure, the signal to be sent is prepared. Next step which should be conducted is convolving it with $p_r(t)$

Using the function RCP we have already written for the first section, we generate the pulse we want to employ to transmit our signal.

By convolving signals, $P_R(t)$ and Digital Symbols we will approach the prepared transmitted signal.

Three transmitted signals are generated since we are using 3 sampling errors in generation of $p_r(t)$

- This section uses the file **P2.m**

III. AWGN Channel Simulation

For this section which employs the second part of P2.m file, using the SNR terms we have, noise power is calculated first. After calculation of η using the power obtained based on each SNR, following noise related to each η is computed and added to the transmitted signal we already prepared.

Since we used 3 sampling errors in simulating $P_R(t)$ we have 3 transmitted signal. Generating 11 noisy terms we will have 33 received signals.

All these explanations are brought in the code in format of comments and titles.

- Implementation of this part is also in second part of **P2.m**

IV. Symbol Detection

This section is divided into 2 parts.

1. Sampling
2. Decision Making

Sampling

To pick samples from the received signals we use the syntax in the manual as instructed and obtain the time that samples should be taken. Using an iteration which iterates for 11 times, each time picking samples from received signals we will have 3 matrix which contain the samples taken from the signals at the receiver end. First matrix refers to the one with 0 error sampling, each row refers to the i'th noisy additive term. Second matrix refers to the one with 0.1T sampling error and the last is for the one with 0.2T sampling error. More explanation is available throughout the code with comments.

Decision Making

After gathering all the samples it's time to make decision and decide each sample refers to what symbol. To follow this aim we need a threshold so that we can compare the samples with and finally decide each sample refers to which symbol.

Since ISI is zero and noise is AWG and the probability of generated symbols at the transmitter's end is equal the threshold is 0.

$$\Delta = 0$$

This means:

SAMPLE	SYMBOL	BIT
$\boldsymbol{Y(t_m) > 0}$	1	1
$\boldsymbol{Y(t_m) < 0}$	-1	0

Same algorithm and process is implanted on all the taken samples and finally detected symbols vector is generated.

- This section is implemented in third part of **P2.m**

V. Error Probability Computation

In this section, we aim to check and find the error probability of our system. To conduct this process, we use an iteration to find the number of errors using a simple comparison between detected symbols and original modulated symbols. After computation of numbers of error, by performing a simple division to number of transmitted symbols which was N, we can obtain error probability. This term is computed for all sampling errors and SNRs.

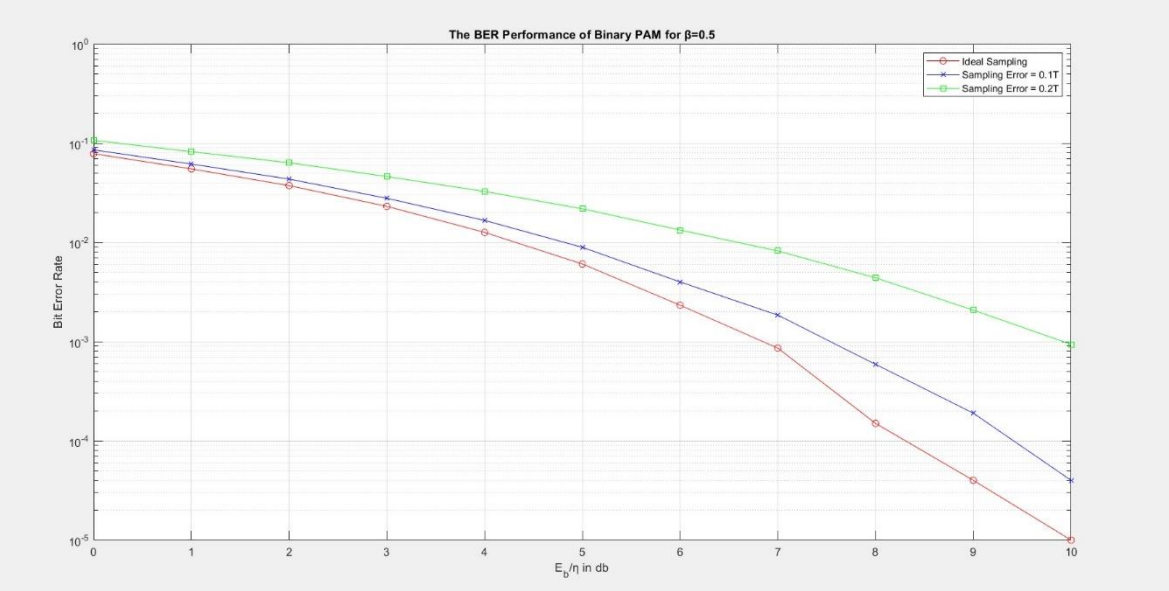

Figure 5.1: System Error Probability for β=0.5

As it can be observed in the obtained result by increasing the SNR term, our error probability continuously decreases. What we want to analyze now, is the impact of ISI caused by wrong sampling on error probability.

$$Y(t_m) = A_m + \sum_{k \neq m} A_k p_r\big((m-k)T\big) + n(t_m)$$

As far as we already know, the taken sample is computed as above. Second term stands for ISI. As we can analyze in figure 5.1, in ideal sampling error probability is always, at all SNR values, lower than the error probability in the scenarios in which wrong sampling is done. In addition, P_E in the second scenario in which sampling error is 0.1T is lower than P_E in third scenario in which sampling error is 0.2T at all SNR values. What we can deduce is:

$$\forall\, SNR; \quad P_{E1} < P_{E2} < P_{E3}$$

So it is obvious that more sampling error results in more ISI which contributes to higher error probability.

- This section uses **P2.m**

Repetition of Sections for β = 0 and β =1

Here in this section, based on what we have been instructed in the project manual, we want to repeat all the sections for 2 values of 0 and 1 for β. As the code we have been using to obtain our results uses the parameter β as an input of the code we can change this variable at the beginning and then obtain the result by running the code. It is notable that RCP function which is written to generate the Raised Cosine Pulses also uses beta as an input parameter to generate this pulses. Therefor by changing the beta value at the beginning of the files P1.m and P2.m, we can get our hands on the pulses shape and error probability respectively. Shown below is the obtained results.

β=0

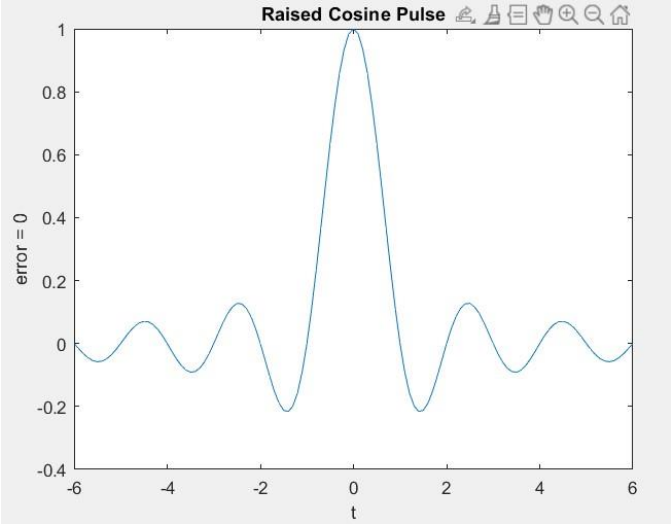

Figure 5.2: Raised Cosine Pulse for β=0

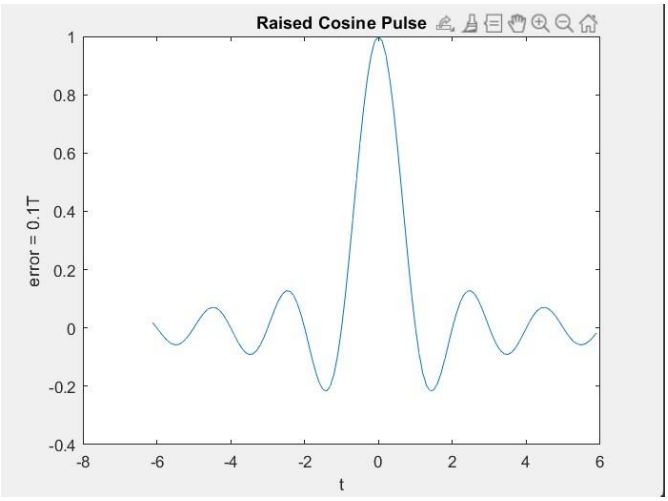

Figure 5.3: Raised Cosine Pulse for β=0

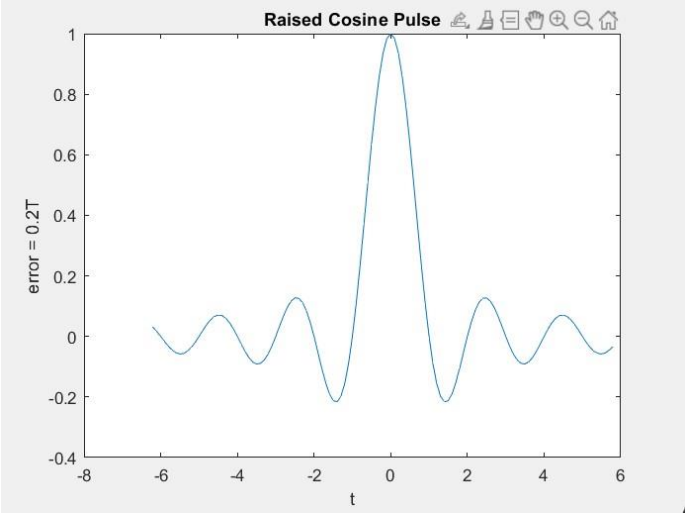

Figure 5.4: Raised Cosine Pulse for β=0

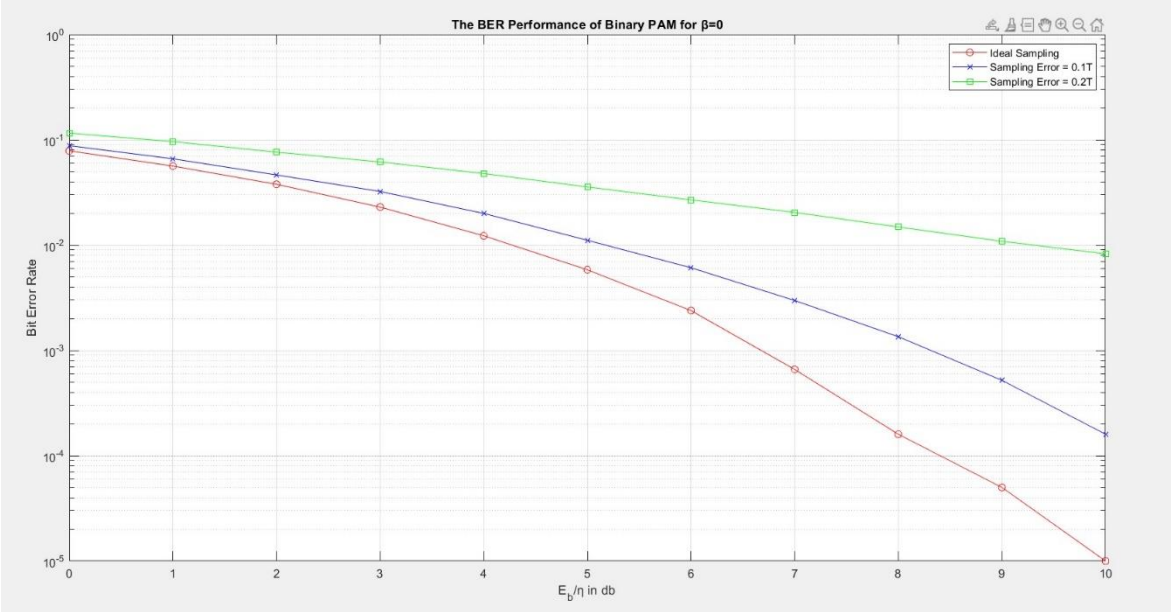

Figure 5.5: Error Probability for β=0

Achieved result is exactly like the presented graph in the project manual.
Raised Cosine Pulses are also the exact signal of sinc(t)

β=1

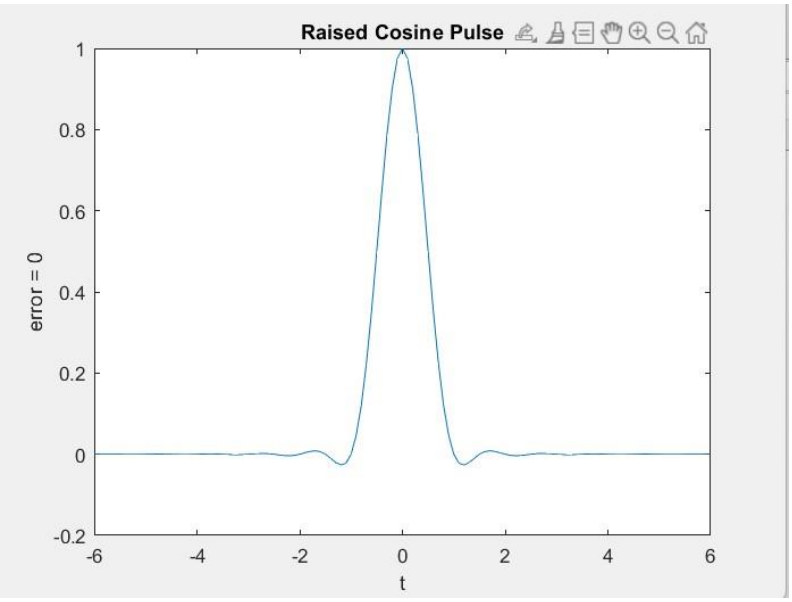

Figure 5.6: Raised Cosine Pulse for β=1

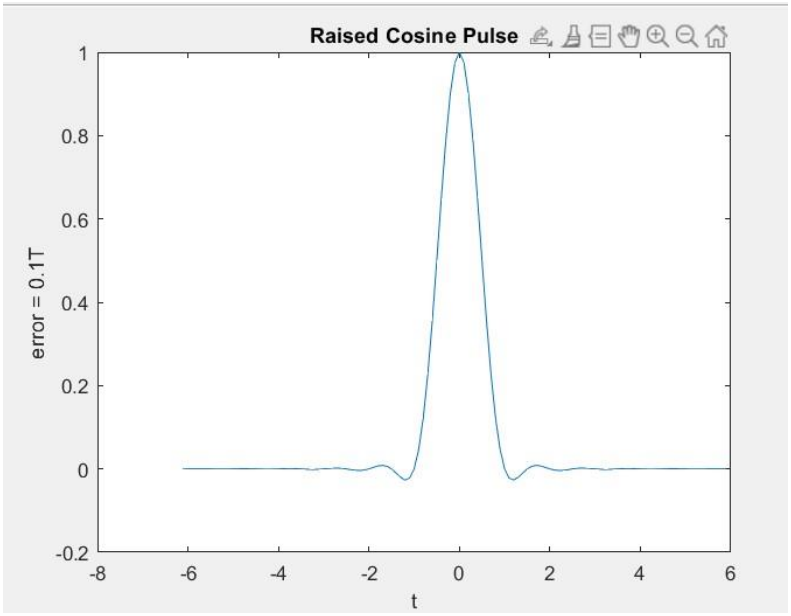

Figure 5.7: Raised Cosine Pulse for β=1

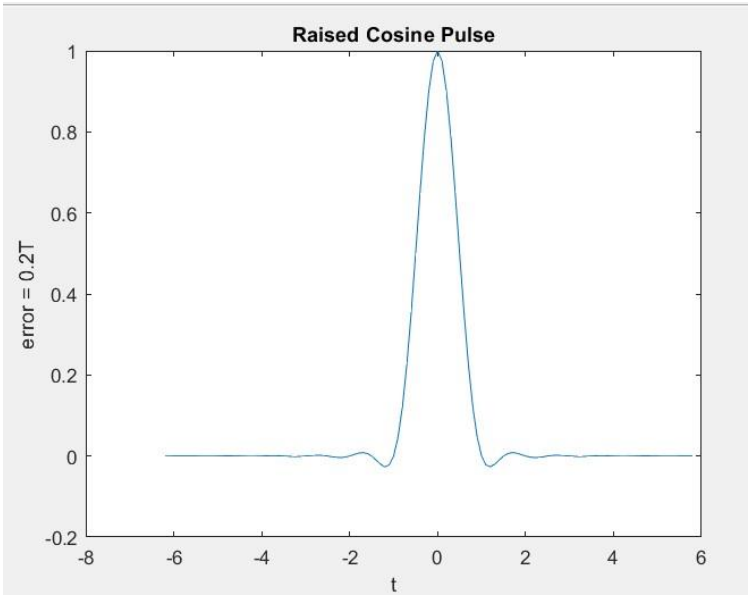

Figure 5.8: Raised Cosine Pulse for β=1

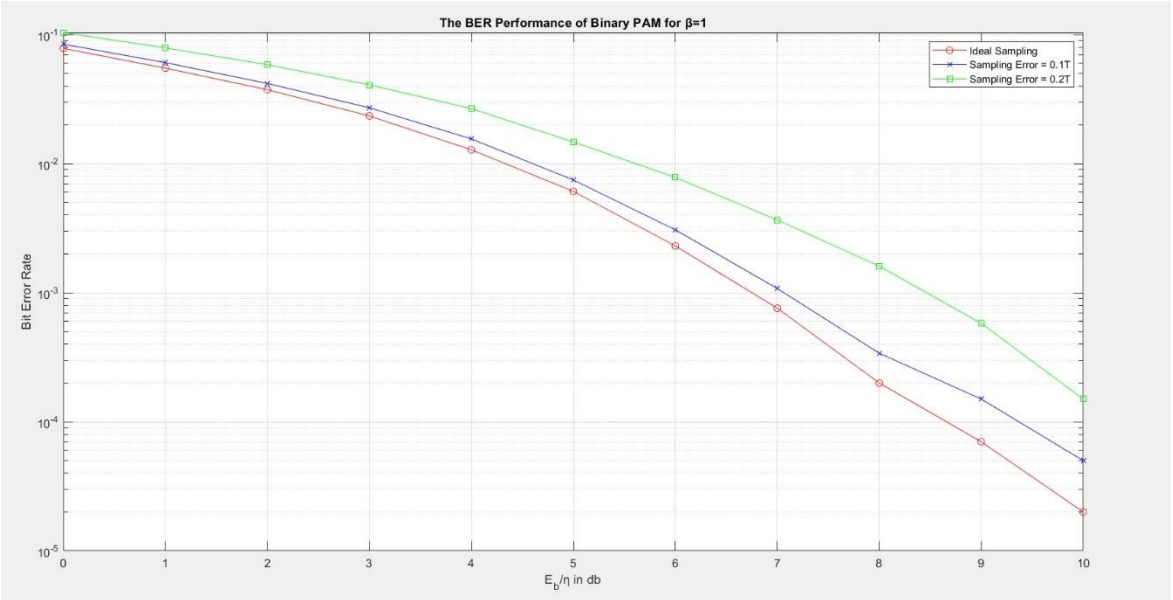

Figure 5.9: Error Probability for β=1

As it can be observed in the achieved results above, error probability in all cases where β changes, decreases with increment of SNR and increases with increment of sampling error. In all cases the graph which refers to sampling error of 0, is below other graphs which means lower P_E and ….

Sensitivity of P_E to sampling error with different β

Now we aim to perform an analysis about sensitivity of error probability to sampling error with different βs and figure out in which β, this sensitivity is lower.

Taking another glance at the graphs we already plotted for P_E in different βs in figure 5.10,

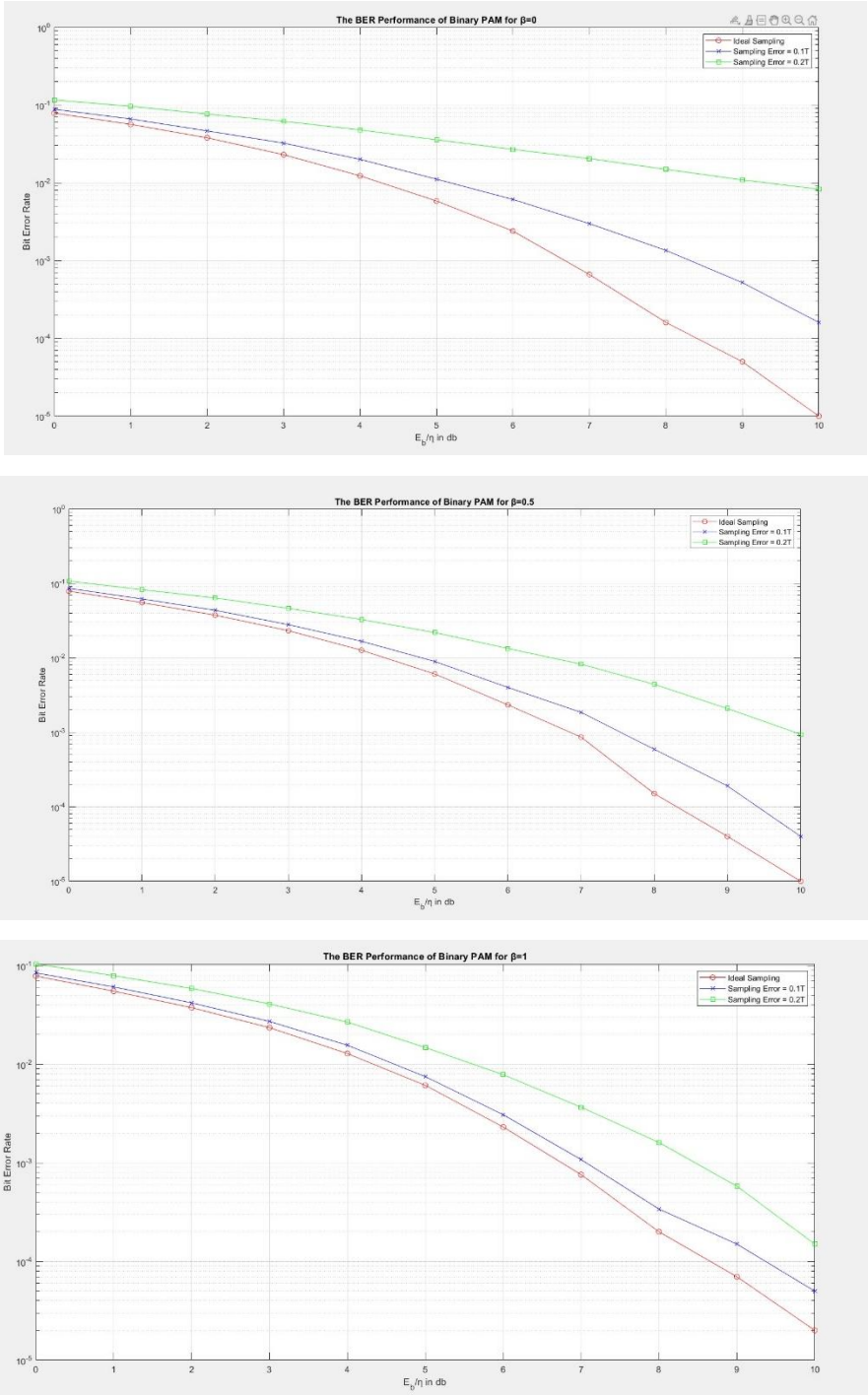

Figure 5.10: P_E in different β values

The distances that each point of the following graphs have in figure above is obviously due to the ISI term which is caused by sampling error as already discussed. As it is notable as β increases from 0 to 1, this distance decreases and graphs get closer together.
This means that by increasing β the ISI term difference in different sampling errors decreases which let us conclude the fact that increment of β decreases sensitivity of P_E to ISI.

In fact, when β=1 the sensitivity to sampling error is lower than the case when β=0.5 and it is lower the scenario in which β=0

- When is higher the sensitivity is lower.
- By sensitivity we mean sensitivity of error probability to sampling error which contributes to ISI.
- Value of β=1 provides us with the least sensitivity.

Conclusions

What was deduced throughout the project can be listed as below:

1. In transmission of a digital signal the pulse which is used to generate the analog transmitted signal plays an important role.
2. More SNR results in lower error probability.
3. Less sampling error results in lower ISI and lower error probability.
4. Parameters of the pulse can change the sensitivity of error probability to ISI and sampling error.

Project Files

- All MATLAB files are available in the folder **Matlab files**
- **RCP.m** function generates raised cosine pulses
- **P1.m** plots these pulses.
- **P2.m** plots error probability graphs.