EE2025 Independent Project (2019-20)
Programming Assignment - 1

See Google Classroom for submission deadline

Submission Instructions: You may form teams of size 1-2 students (only among students crediting the course). Exactly one of the team members must upload the simulation result online in Google Classrooms. You must upload the following, as two separate files (do not 'zip' them):

1. simulation results in '.pdf' format. Other formats are not allowed. This file must include the details of all the team members (name and roll number).
2. your program/script file(s).

Cheating, in any form, will be taken seriously. Please write the program on your own. You could be asked to present the script file and explain the algorithm orally at short notice.

Programming Language: You can use Matlab, Python or any other tool for this programming assignment.

The Problem: You will implement digital modulation and demodulation to communicate a binary image file across an additive white Gaussian noise channel. The image file `binary_image` is included with this assignment in numpy and Matlab formats. This is a 110×100 array, where each element is a 0 or a 1. Here, a 0 represents a black pixel and 1 is a white pixel. You can load and render the image in python and Matlab as follows. Do not copy-paste these commands, but manually type them in the command window or terminal.

Python:
```
import numpy as np
from matplotlib import pyplot as plt
MonaLisa = np.load('binary_image.npy')
plt.imshow(MonaLisa,'gray'), plt.show()
```

Matlab:
```
load binary_image.mat
imshow(MonaLisa, [0 1])
```

The image, in all, contains $110 \times 100 = 11000$ information bits. You will modulate and transmit them using 4-QAM modulation scheme with carrier frequency 2 MHz and symbol duration 1 micro sec, i.e., 2 bits are transmitted per micro second. The receiver will use the optimal demodulator, i.e., the maximum-likelihood detector or the minimum distance detector.

You will simulate the communication for 4 values of $\mathcal{E}_b/N_o$: $-10, -5, 0, 5$ dB. For each of these scenarios, you will plot the received image (which will be noisy) and report the number of pixels that were wrongly demodulated. These four figures and the corresponding number of wrong pixels must be reported in a single pdf file.

Channel model and modulation scheme: The carrier frequency $f_c = 2$ MHz and symbol duration $T = 1\,\mu$ sec. Since each symbol carries 2 bits, the overall communication duration is $11000 \times T/2 = 5.5$ m sec. The transmitted waveform $s(t)$ for duration $0 \le t < 0.0055$ is determined as follows. We first order the pixels or information bits in a sequence $b_1, \ldots, b_{11000}$. For each $i = 1, \ldots, 5500$, we transmit one 4-QAM symbol in the time window $(i-1)T \le t < iT$ that modulates the bits b_{2i-1} and b_{2i}. The transmitted waveform in this interval is given by

$$s(t) = x_{2i-1} \cos(2\pi f_c t) \; + \; x_{2i} \sin(2\pi f_c t), \text{ for } (i-1)T \le t < iT,$$

where $x_j = +1$ if $b_j = 0$ and $x_j = -1$ if $b_j = 1$. Given this modulation scheme, you must calculate the energy per information bit $\mathcal{E}_b$. This will be required to determine the value of noise power spectral density $N_o/2$.

The noise $w(t)$ is modelled as a Gaussian random process with power spectral density equal to $N_o/2$ across all frequencies. For a given value of $\mathcal{E}_b/N_o$, you can calculate N_o since $\mathcal{E}_b$ is already known. The received waveform is

$$r(t) = s(t) + w(t) \text{ for } 0 \le t < 0.0055.$$

The Discrete-Time Model: Since we need to simulate the communication scheme using Matlab/python, we need to discretize the channel model. We will use a sampling rate of $f_s = 50$ MHz. We will assume that the received waveform $r(t)$ is first passed through an ideal low-pass filter with bandwidth $-f_s/2 < f < f_s/2$, and then sampled at rate f_s samples per second. The total number of samples of $r(t)$ will be $f_s \times 0.0055 = 275000$, since the communication duration is 5.5 m sec. We will make the simplifying assumption that the low pass filtering operation does not distort the transmitted waveform $s(t)$. Under this assumption, the n^{th} sample is

$$r[n] = s[n] + w[n] = r(nT_s) = s(nT_s) + w(nT_s), \text{ for } n = 0, 1, \ldots, 274999,$$

where $T_s = 1/f_s$. The transmitted signal is captured by the sequence $s[n] = s(nT_s)$ and noise by $w[n]$. The noise signal $w[n]$ is modelled as composed of independent and identically distributed Gaussian random variables with mean zero and variance $f_s \times N_o/2$.

Note that the number of samples per QAM symbol is $f_s \times T = 50$, i.e., samples $r[0], \ldots, r[49]$ correspond to bits b_1 and b_2; the next 50 samples correspond to bits b_3 and b_4, and so on. The receiver uses minimum distance decoding on $r[0], \ldots, r[49]$ to demodulate b_1 and b_2, and uses the next 50 samples to demodulate b_3 and b_4, and so on.

Verifying your simulation results: The bit error rate (BER) of 4-QAM modulation scheme is $Q(\sqrt{2\mathcal{E}_b/N_o})$, where Q is the Gaussian tail function. Thus, the number of pixels that are wrongly demodulated will be approximately equal to $11000 \times$ BER. You can use this rule-of-thumb to verify if your program is correct.