GEORGIA INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING

ECE 4271 SPRING 2016
MATLAB PROJECT #2

Binary Communication Transceiver Design with Alphabet Analysis

Assigned: Sunday, March 6, 2016 (t-square)

Due:

a) Report (hardcopy) Thursday, March 17, 2016 before the *beginning* of the lecture.

b) Code and Report, March 20, 2016-11:59 PM EST: You must also submit an electronic version of your report; however, only your hardcopy will be graded.

- This project is to be done *individually.* Collaboration on projects is *not* OK. Each student must work on the projects independently. Each student must develop his or her own analyses and computer code in its entirety. Any project-specific help needed should be sought from the TA and Dr. Marenco. Students are not to discuss the theory or approaches to coding the theory with one another, nor are they to assist in debugging each other's work. ***The Georgia Tech honor code and its policies apply.*** You may ask Dr. Marenco or the TA questions regarding theory and implementation of the project, including asking them at the beginning or end of class or during office hours, when others can benefit as well.
- Data required for this project are available for download from t-square, in Resources under Project #2. You will also find reference materials and code there. ***Whether you begin working right away or not, be sure you download the data and make sure you can load it into MATLAB as soon as possible to avoid last minute difficulties.***
- Reports and code will be graded primarily on completeness in addressing the assignment and quality of results. Reports must be typed, not handwritten; should be well-organized; and must clearly explain your design and the decisions and analysis that went into it. Code must be provided in the form specified below, and should be commented well-enough to be clearly understandable.
- Questions or clarifications should be directed to Dr. Marenco.[1] Errata, revisions and hints (if any) will be made available via e-mail, t-square, or during class.

[1]Office: 404-407-6303. Office hours: Thursday 1:45 – 2:45 PM, or by appointment, else e-mail: alvaro.marenco@gtri.gatech.edu

1. PROBLEM

We use Matlab to simulate a baseband binary communication system with line coding and matched filtering. Bit-error-rate and bandwidth efficiency will be calculated from the simulation. We will also perform line coding and statistical analysis on the English alphabet.

2. REQUIREMENTS

You must submit a report of your methods and findings that must include:

1. An overview description of how your processing algorithm works. This must include a block diagram of the major processing steps with a description of the rationale for each step. It should identify the major MATLAB functions or code blocks corresponding to each major function in your block diagram.

2. A brief description of the theory underlying the processing algorithm. Explain the analyses and design decisions that led to each of your parameter choices.

3. A complete listing of the MATLAB program used to process the signals. Code must be sufficiently modular and well-commented to be understood readily. Listings must also be included for any functions used, except for those which are built into either MATLAB or its Signal Processing Toolbox, or were provided in class.

In addition to the written report (submitted in class, hardcopy), you must submit your MATLAB code in t-square in a single `.zip` file containing MATLAB code and also the report by the two deadlines: a) Hardcopy report, **March 17, 2016-3:05 PM EST** and (b) code and report electronically, **March 19, 2016-11:59 PM EST**. See below for further instructions on naming the zip file and how your code must be organized. Your program will be tested by applying it to signals (or files) generated by Dr. Marenco (*not* the same provided for development of your algorithm.)

Start working early. If you encounter problems, please ask questions to clarify anything that is vague or incomplete.

3. TASKS

The project is divided in two major tasks: (a) Simulation of a baseband binary communication system with line coding and matched filtering; and (b) Simulation that encodes a text file and coverts it to a data stream after QPSK encoding the binary data. You must also write a decoder that converts from a QPSK data (symbols) to the original text file.

A. **Simulation of a binary communication system:** you will be asked to generate a binary sequence of specified length and SNR. The goal is to design an encoder and decoder using a matched filter that will minimize the probability of error. Details of functions and design considerations are provided below.

B. **QPSK encoding and decoding with statistical analysis:** in this task you will read a simple text file with the extension .txt (e.g. foo.txt). The file contains characters used in everyday writings. The text (ASCII) will be encoded into a binary in such a way that only 5 bits per character are needed. Once you have the binary stream (e.g. text to binary), you encode the bits into QPSK symbols. For this particular problem, QPSK takes successive pairs of bits and represents them as one of four "symbols", namely $\pm 1 \pm j$ according to

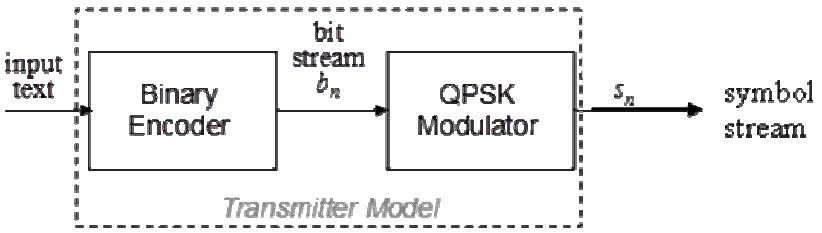
$$s_n = (-1)^{b_{2n}} + j(-1)^{b_{2n+1}}$$

where s_n is the n^{th} symbol and b_{2n} and b_{2n+1} are successive even- and odd-numbered bits in the bit stream. To recover the text, you must pass the signal s_n through a QPSK demodulator, and a binary-to-text decoder.

Additionally, you will analyze the text data as follows:

i) Similar to homework #3 (Exercises in Computer-Based Exercises for Signal Processing using MATLAB 5, chapter 6 Exercises 1.1, 1.2 on p. 163) you will create a histogram (relative frequencies of letters) of your text file. This histogram should provide information about the relative frequencies of letters (see Report Evaluation section). Do not count spaces or other character on your calculations. Estimate mean and standard deviation. For consistency, use numbers from your 5-bit domain.

ii) Generate a table with two columns (see Report Evaluation section): one column represents letters from a thru z (in that order), and the second column their relative frequency, in percentage.



4. MAJOR FUNCTIONS – BINARY COMMUNICATIONS TASK

1. Data generation

   function Bits=DataGeneration(Length)

Given a desired length of the data, generate the sequence with zeros and ones. You may need to use a Matlab function that generates integer bits like randi().

2. Line Coding

   function S=Pulse(Bits, SNR)

Given a binary bit sequence, generate the waveform with rectangular pulse shapers, i.e., when "1" is sent, use positive rectangular pulse with amplitude "A" (a parameter decided by the desired signal-to-noise ratio); when "0" is sent, use negative rectangular pulse with amplitude "-A";

3. Noise generation

   function Noise=NoiseGeneration(Length of S)

Given the transmitted sequence S, generate white Gaussian noise with zero mean and unit variance.

4. Receiver design

The received signal is "S + Noise". Design a receiver to decode the binary sequence from the received sequence.

   function est_Bits=receiver(S + Noise)

5. Bit-error-rate (BER) measurement

Compare the estimated sequence "est_Bits" with the original sequence "Bits" and count the number of errors. The bit-error-rate is calculated by the number of errors divided by the total number of bits.

5. MAJOR FUNCTIONS – QPSK ENCODING AND DECODING TASK

1. You may find MATLAB commands such as `char`, `dec2bin`, `bin2dec`, `num2str`, `str2num`, `double`, and `reshape` to be useful. (I am *not* saying you *have* to use all of these, or even any of them; but some may be helpful.)
2. Text to binary:

   function bn = text2bin(filename)

   The 'filename' should be the filename of simple text file.

3. Binary (bits) to QPSK symbols:

   function Sn = bin2QPSK(Bits stream)

4. QPSK symbols to binary (bits)

   function demodBn = QPSK2bin(Bits stream)

5. Bits to text:

   function T = bin2text(Bits stream)

5.1 DATA

Each signal encodes a non-trivial amount of text. Thus, a correctly decoded sequence, when displayed, should reproduce the original written text (except in lower case, as discussed below).

The text characters have to be encoded into binary in such a way that only 5 bits per character are needed. This is done by limiting the characters to the following:

- space, comma, period, apostrophe, line feed, return, and the 26 *lower case* characters ‘a’ through ‘z’

Because of the lower case limitation, your output text will be entirely in lower case characters. The specific encoding is as shown in the following table:

Character	ASCII Decimal Value	Binary Code in This Project
space	32	00000
,	44	00001
.	46	00010
′	39	00011
line feed	10	00100
carriage return	13	00101
′a′ through ′z′	97-122	00110-11111

After this encoding, original text data containing L characters will be represented by a binary sequence of M = 5L bits. For your symbols, L characters will require $5L/2$ QPSK symbols.

6. SOME CONSIDERATIONS

4.1 Theoretical Analysis: As we have shown in the class, the analytical results using the Q-function can be derived for this case. In your report, you should give the theoretical results as bench marks.

4.2 Bit sequence length: To verify your program, one way is to compare the BER results by your simulation with the theoretical analysis results. However, to calculate a proper BER, you need to make sure that the bit sequence length should be long enough. For example, if the theoretical BER is 10^{-3}, then you need about 10^5 bits to get a reliable estimate.

4.3 Signal-to-noise ratio: You need to be careful about the signal to noise ratio definition. Since a matched filter will be used here at the receiver, it will affect the noise variance. When you calculate the BER, make sure you find the corresponding SNR.

7. MATLAB CODE SUBMISSION AND REPORT EVALUATION

Your MATLAB code must be submitted in t-square in a single zip file ***by the deadline (*March 19, 2016-11:59 PM EST*)***. See below for further instructions on naming the zip file and how your code must be organized.

1. The file must contain all of the MATLAB code, including all functions and subfunctions, data files, *etc.* required to run your algorithm on a test signal. Place all of these files in a single flat (no subdirectories inside of it) directory with the name `FirstInitialLastName.zip`. Do not include any functions that are built into MATLAB or its Signal Processing Toolbox. Do not forget to include your report in the zip file.

A. Binary Communications Task:

2. In your report, you should contain one figure (BER vs. SNR) which includes two curves: one is the theoretical curve based on Q-function; the other is your simulated curve. X-axis is SNR in dB (0:2:16) and Y-axis is BER plotted in log-scale (use "semilogy" in Matlab).
3. In your report, you should calculate the bandwidth efficiency with a given BER. All the numbers, such as bandwidth must be obtained from simulation. Discuss the definition of bandwidth of your baseband waveform.
4. Your top level function *must* have as its first line

```
function BER = transceiver(length, SNR)
```

Furthermore, it must work when all of the files in your zip file have been unzipped and placed in the MATLAB `work` directory.

Your program should produce an output that is simply the BER given a length of signal and SNR.

5. I will test your program by simply executing the command

```
transceiver(length, SNR)
```

I will use different **SNR values and sequence lengths** to test your program.

B. QPSK Encoding and Decoding Task

6. You must download (from t-square) a text file that you will use to test your work. I refer to this file as 'test file'.

7. In your report, you should have one figure (histogram) which includes a relative frequencies of letters contained in the *test file* downloaded from t-square.

8. In your report, you should include one table that has two columns. The first column refers to the letters from a to z and the second column is the relative frequency of the letters (in percentage) in the *test file*.

9. Your top level function **must** have as its first line

```
function Qtext=QPSKnAlphabet(filename, …
          QPSKSymbols,letter)
```

filename: this is the name of the input text file (with extension .txt) that I am going to use to evaluate your program. For example, 'foo.txt'. This is a different text file than the *test file.*

QPSKSymbols: I may choose to enter the corresponding QPSKSymbols of 'foo.txt' and enter them as a vector (column vector). Therefore, your program should be able to detect whether filename or QPSKSymbols has been input so they can be processed accordingly. NOTE: you will have either filename or QPSKSymbols as an input for any given run. Two examples:

i) `Qtext= QPSKnAlphabet(filename,[],letter), or`

ii) `Qtext= QPSKnAlphabet([],QPSKSymbols,letter)`

letter: this is a single character (e.g. a letter in the file 'foo.txt' or in the QPSKSymbols vector).

Qtext: this is the output structure of your program with the following components:

Qtext.mean: mean value of your entire data.

Qtext.std: standard deviation of your entire data.

Qtext.outtext: the characters output (in a single vector) from your demodulator and decoder. They should be identically to the input text file ('foo.txt'), or to the text file I encoded and modulated to produce the QPSKSymbols. NOTE: A correctly decoded sequence, when displayed should reproduce the original written text (except in lower case, as discussed in the DATA Section). Characters in the text file (or QPSKSymbols) will have little or no noise.

Qtext.oneLetfreq: the relative frequency, in percentage, of the input character 'letter'.

Be absolutely certain that your program will run correctly when I unzip your files into the MATLAB `work` ***directory and type this command into MATLAB.***

8. GRADING

Your grade on this project will be allocated as follows:

- Code Performance: 70%
- Report: 30%
    - Algorithm and parameter description and rationale: 10%
    - Code listing and readability: 5%
    - General report quality (complete but succinct, figures, writing, *etc.*): 15%