ECE3311 – Principles of Communication Systems

Project 05

Due: 9:00 AM, Tuesday 8 December 2020

1 Project Objective & Learning Outcomes

The objective of this project is to have you master digital modulation schemes employed in passband communication systems and operating environments. From this project, it is expected that the following learning outcomes are achieved:

- Understand one-dimensional digital modulation schemes such as Amplitude Shift Keying (ASK).
- Obtain an understanding of two-dimensional digital modulation schemes such as Phase Shift Keying (PSK) and Quadrature Amplitude Modulation (QAM).
- Learn about the Orthogonal Frequency Division Multiplexing (OFDM) concept by leveraging the Inverse Discrete Fourier Transform (IDFT) and DFT.
- Learn about modulation detection using Error Vector Magnitude (EVM) analysis.

For this project, you will need to download the Jupyter Notebook "*Project 5 ECE3311.ipynb*" from the ECE3311 Canvas website and open it in VSCode, which will provide you with all the example source code and outputs of subsequent topics discussed in this project.

2 Preparations

The purpose of this project is to provide a solid foundation in simulating and evaluating communication systems using digital modulation techniques, including both one-dimensional and two-dimensional modulation schemes, as well as both single-carrier and multi-carrier formats. Via the implementation of all of these tasks, you will learn these fundamentals and ultimately be capable of employing these techniques to other courses as well as future tasks in both graduate school and industry.

The starting point for all of the source code you are about to write is the passband inphase/quadrature (I/Q) representation. In Table 4.1 of the course textbook, the mapping of the complex baseband signal $g(t)$ to passband signal $s(t)$ is expressed using several representations, but in this project we will focus on the representation $g(t) = x(t) + jy(t)$ that will yield $s(t) = x(t)\cos(2\pi f_c t) - y(t)\sin(2\pi f_c t)$. Throughtout the rest of this project, **everything**, both this report and the Python source code, will be represented in I/Q, including the channel noise added to the transmitted signal.

Let us begin with the one-dimensional digital modulation scheme, Amplitude Shift Keying, specifically: 2-ASK and 4-ASK. As with any communication system, there are three stages: Transmitter, Channel, and Receiver. The first step in simulating a digital communication system is the generation of the binary data, its mapping to the corresponding complex symbol representation, and its pulse shaping using something like a raised cosine filter. Once all the digital processing is performed on the digital signal, it is modulated up to passband and takes on the I/Q representation.

Next, we assume the transmission passes through an Additive White Gaussian Noise (AWGN) channel, which introduces distortion to the signal. This noise needs to also be modulated to the same carrier frequency, which we achieve using both cosine and sine functions. Since cosine and sine modulation functions are employed, the noise energy needs to be reduced by half, hence the division of each noise term by the square root of 2.

At the receiver, we use coherent detection assuming that we know the carrier frequency at this end of the communication system. Using both cosine and sine functions to demodulate the signal down to baseband and applying a lowpass filter to remove its double frequency term. Once the baseband signal has been extracted for both inphase and quadrature components, we need to sample the correct time instances to extract the obtain the I/Q symbol values. Note that since both the raised cosine filters and the lowpass filters introduce something called *group delay*, we need to start our sampling after we have taken into consideration the delay introduced by these Finite Impulse Response (FIR) filters. Even though the ASK waveforms transmitted did not have a quadrature component, we still use the sine functions in the demodulation process (this will help set things up for the two-dimensional modulation schemes later on in this project).

Finally, to show how these modulation schemes work, we can plot the outputs of the sampling process on something called a scatter plot. Examples of the 2-ASK and 4-ASK scatter plots are shown in Figure 1 and Figure 2. The scatter plots provide us with imformation about the demodulated inphase and quandrature amplitude information. If things work out with the communication system design, the scatter plots should be concentrated around defined I/Q locations. On the other hand, if there is a problem, our scatter plots will not be neatly confined to these locations. Since there will always be noise present in our system, you will notice that the scatter plots produce a small cloud of points rather than an exact point. This is due to how the noise displaces the I/Q amplitude values.

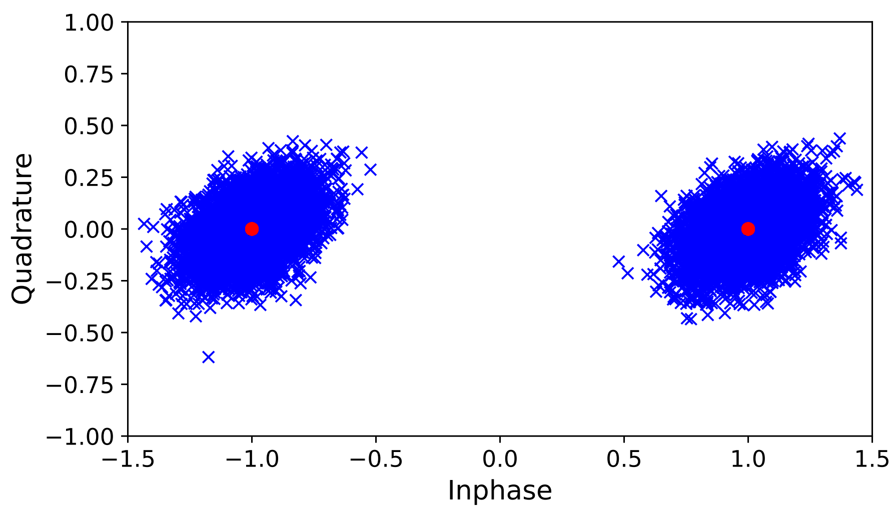

Figure 1: Scatter plot of a 2-ASK signal constellation.

3 Two-Dimensional Digital Transmission

From Section 2, the process of converting binary data into digital waveforms that get sent across a noisy channel and then converted down to a baseband signal at the receiver was presented for a couple

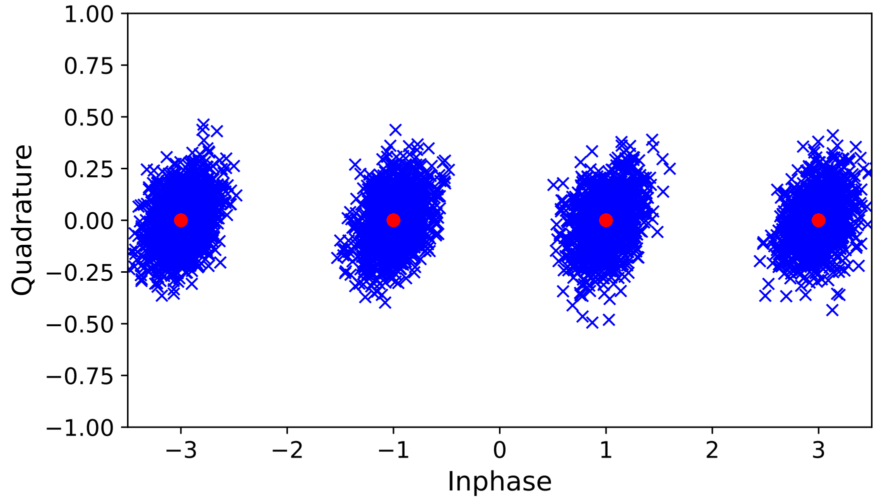

Figure 2: Scatter plot of a 4-ASK signal constellation.

of one-dimensional modulation schemes (2-ASK and 4-ASK). The reason for calling these two modulation schemes “one-dimensional” is due to the fact the signaling information is contained within one axis of the signal constellation diagram. However, we did account for the noise contribution from the AWGN channel to be two-dimensional, hence the scatter plots having signal contellation points defined in two dimensions in Figure 1 and Figure 2. Given this example code, the next step is to implement the two-dimensional modulation schemes $\pi/4$-offset Quadrature Phase Shift Keying (QPSK) and square Quadrature Amplitude Modulation (QAM). Example code in the Jupyter Notebook illustrates the mapping of binary information into a 4-QAM signal constellation (see Figure 3).

Question 1 (5 points): Plot the scatter plot of a square 4-QAM transmission prior to transmission across a channel and after sampling at the receiver. Use the sample code from Section 2 as well as the example code for this section from the Jupyter Notebook to get started.

Question 2 (5 points): Plot the scatter plot of a $\pi/4$-offset QPSK transmission prior to transmission across a channel and after sampling at the receiver. Use the sample code from Section 2 as well as the example code for this section from the Jupyter Notebook to get started.

Question 3 (5 points): Plot the scatter plot of a square 16-QAM transmission prior to transmission across a channel and after sampling at the receiver. Use the sample code from Section 2 as well as the example code for this section from the Jupyter Notebook to get started.

4 Orthogonal Frequency Division Multiplexing

As we have seen thus far, signals can be transmitted on a single carrier frequency, *e.g.*, signals are modulated to a carrier frequency f_c. However, it turns out that a single transmitted signal may only use up a small portion of the total available bandwidth. To increase bandwidth efficiency and

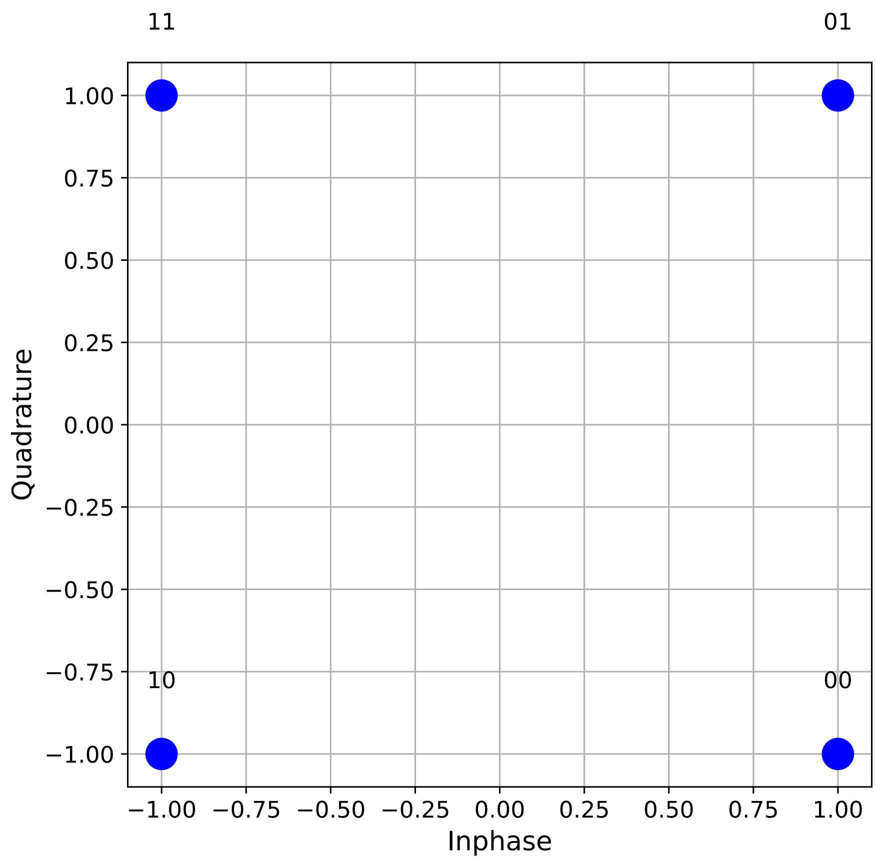

Figure 3: Square 4-QAM signal constellation.

overall throughput across a frequency band, it is possible to send additional individual signals in other portions of the unused spectrum simultaneously, each with a different carrier frequency. Based on this approach, it is possible to take a single high-speed data transmission, demultiplex it into N parallel slower data streams, modulate them individually and transmit them across these unique carrier frequencies only to be reconstructed at the receiver and multiplexed back into an estimate of the original high-speed data transmission. This technique of translating a single high-speed transmission into parallel low-speed transmissions back into a single high-speed transmission defines the concept of *multicarrier modulation*.

One might ask, "Why do something this complicated instead of just sending the single high-speed transmission across the same bandwidth?" Although it is true that the overall throughput is equivalent whether transmitting the data via single carrier of multicarrier, there are significant advantages with respect to the multicarrier approach enabling for the correction of channel-based distortion such as multipath propagation. In fact, the benefits of using multicarrier modulation are considerable relative to the costs that it is used in 4G and 5G cellular systems, WiFi, and many other wireless technologies involving signficant data rates.

To reduce the complexity of the multicarrier modulation approach, Orthogonal Frequency Division Multiplexing (OFDM) was proposed, which efficiently maps the low-speed data streams (called *subcarriers*) onto the different unique carrier frequencies via the Discrete Fourier Transform (DFT) and Inverse DFT (IDFT). The process at the transmitter works as follows:

1. Take a high-speed binary data stream and demultiplex it into M subcarriers.
2. On each subcarrier, map b bits to a complex baseband symbol.
3. Take a symbol from each subcarrier and apply each one to an input of an M-point IDFT.
4. The output of the IDFT is your OFDM symbol. This output is converted from a set of parallel points to a sequence of serial points.
5. The OFDM symbol then has a cyclic extension added to it, is pulse shaped, and upconverted to a carrier frequency f_c before being transmitted across the channel.

Once the OFDM symbol is intercepted at the receiver, the reverse set of operations are generally performed, namely:

1. The OFDM symbol is downconverted to baseband, equalized, and the cyclic extension removed.
2. The OFDM symbol is transformed from a serial format to a parallel format and an M-point DFT is applied to it.
3. The outputs of the DFT correspond to the individual symbols from each subcarrier.
4. The subcarrier symbols are mapped to b bits per symbol, yielding individual sequences of bits per subcarrier.
5. The subcarrier bit streams are multiplexed together to form an estimate of the orignal high-speed bit stream.

To provide some insight on how to implement an OFDM transceiver in Python, some sample code has been included in this project's Jupyter Notebook. The Python OFDM example implements the conversion of a high-speed binary data stream into several parallel subcarriers, which are then each converted into a single 4-QAM symbol per subcarrier (there are only enough bits generated to produce exactly one 4-QAM symbol per subcarrier). An IDFT is applied to the subcarriers at once, generating an OFDM symbol. This OFDM symbol is then corrupted by an AWGN channel, and this corrupted signal is then converted back to a collection of individual subcarriers via DFT operation.

Note the subcarrier symbols have potentially been corrupted by the channel and thus might not correspond to the original 4-QAM symbols transmitted (see Figure 4 for a scatter plot of all subcarrier symbols retrieved from intercepted OFDM symbol). Consequently, these corrupted subcarrier symbols need to be mapped to the nearest possible 4-QAM representation, demodulated, and an estimate of the original high-speed binary sequence is recreated. The Bit Error Rate (BER) is calculated, which is the number of errors introduced to the estimated high-speed binary sequence relative to the original sequence.

Question 4 (5 points): Modify the OFDM Python code such that $\pi/4$-offset QPSK symbols are used instead of 4-QAM symbols per subcarrier. Generate the scatter plot of the received corrupted subcarrier symbols.

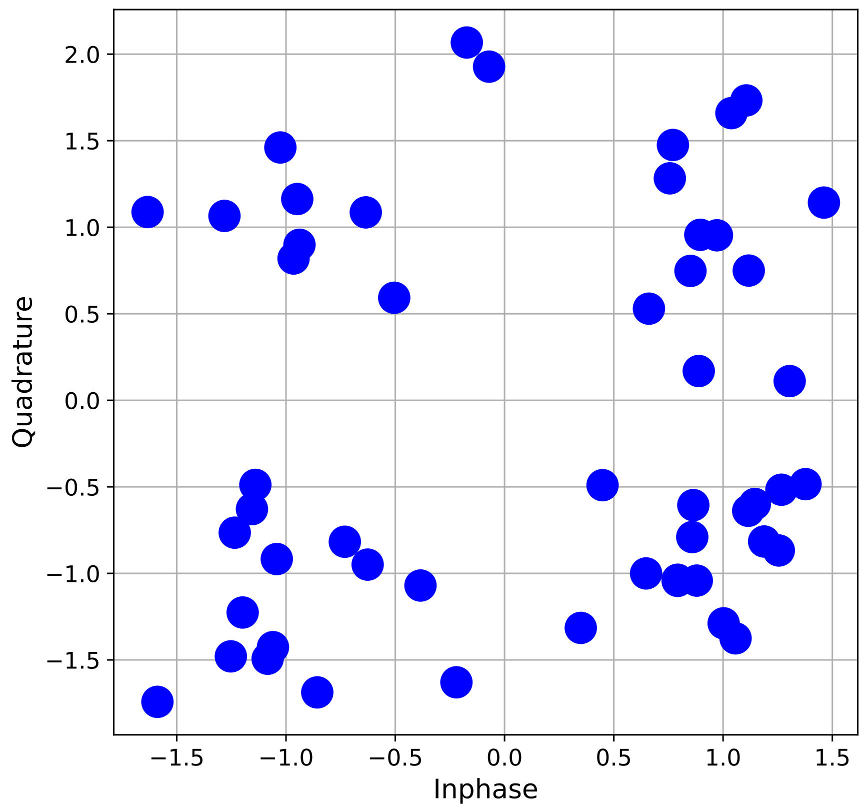

Figure 4: Scatter plot of received subcarrier symbols that were initially transmitted as 4-QAM symbols but were corrupted by the AWGN channel.

> Question 5 (20 points): Instead of just transmitting and receiving just one OFDM symbol, modify the sample code such that it can transmit/receive L OFDM symbols ($L > 1$). Demonstrate functionality via a cumulative scatter plot of the received corrupted subcarrier symbols across all OFDM symbols.

5 Error Vector Magnitude (EVM) Modulation Detector

A common practice in communications where transmissions are received without prior knowledge of the transmitted waveform's contents or even structure is to firstly decide the modulation scheme. Immediately prior to demodulation, the scheme used must be decided. The classical method of determining this is compare each data point to a reference constellation. The reference constellation with the lowest Root Mean Square (RMS) euclidean distance to these reference points is chosen as the most likely modulation scheme. An illustration of this concept is shown in Figure 5 for the mapping of subcarrier symbols to known 4-QAM symbols.

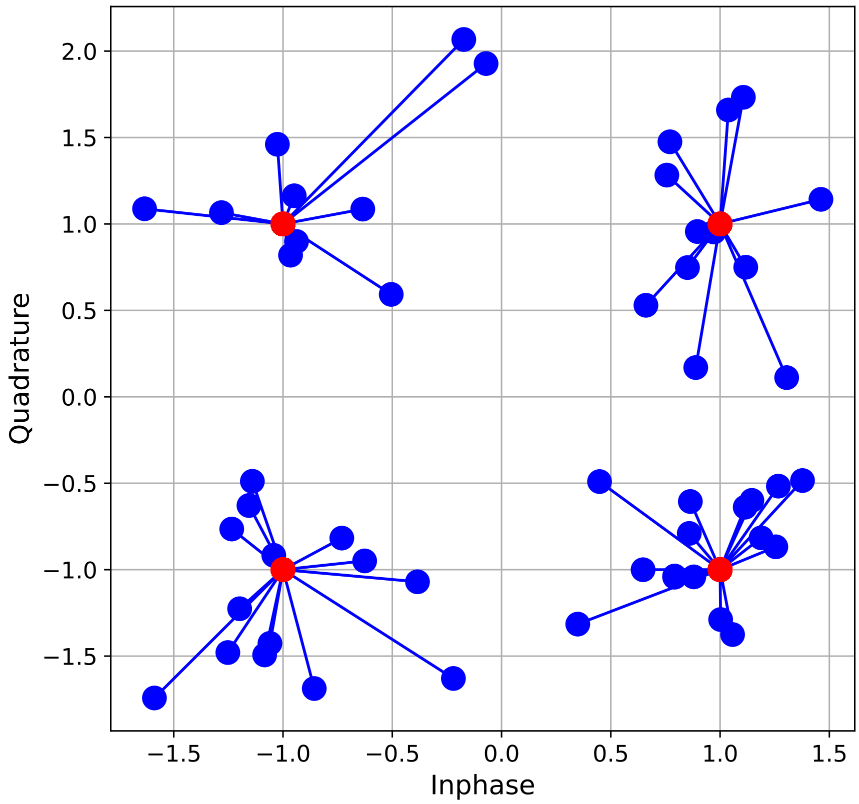

Figure 5: Mapping of received subcarrier symbols to known 4-QAM symbols.

Question 6 (20 points): For single carrier transmission of 4-ASK symbols, calculate the RMS EVM values given 4-ASK, 4-QAM, and $\pi/4$-offset QPSK reference signal constellations. Show that the RMS EVM value is lower for 4-ASK relative to when either a 4-QAM or a $\pi/4$-offset QPSK reference signal constellation is used. Repeat experiment when the transmission consists of 4-QAM and $\pi/4$-offset QPSK symbols.

Question 7 (10 points): For an OFDM implementation employing 4-QAM per subcarrier at the transmitter and sending L OFDM symbols ($L > 1$), use RMS EVM to map the received subcarrier symbols to the nearest 4-QAM signal constellation point. Repeat when employing $\pi/4$-offset QPSK modulation. Produce plots similar to Figure 5 in both cases. Indicate the RMS EVM values for each implementation.

Question 8 (10 points): Find the value of SNR for the OFDM transceiver that would yield a Bit Error Rate (BER) approximately ($\pm 10\%$) equal to 0.01, 0.001, and 0.0001. Provide the associated RMS EVM values for these SNRs. HINT: Transmit enough bits such that 100 errors are obtained in order to achieve a BER with 95% confidence.

6 Report Submission

For each project in this course, the report submission to be uploaded to the ECE3311 Canvas website will consist of a single comprehensive Jupyter Notebook (.ipynb) file and nothing else unless it is requested. For this project, only the Jupyter Notebook is to be submitted electronically by the due date. Failure to submit the Jupyter Notebook by the specified due date and time will result in a grade of "0%" for the project.

Several important items to keep in mind when preparing your submission:

- Include the course number, project team number, names of team members, and submission date at the top of the Jupyter Notebook.
- Make sure your source code is thoroughly documented such that it is made clear what exact the program is doing line-by-line. Adequate commenting is worth **5 points** of the total score for this project.
- Responses to all questions indicated in the project handout. Please make sure that the responses are of sufficient detail.