Introduction

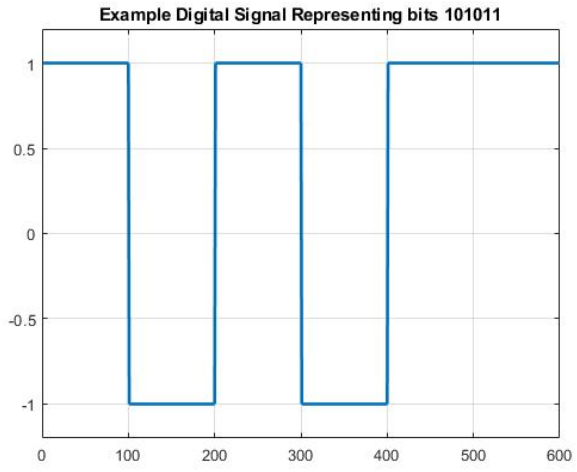
In this project, you are going to implement the receiver for an acoustic modem. An acoustic modem is a device which sends digital signals using sound waves. This approach was used for internet access for many years. For underwater communications, it's still the method of choice since acoustic signals propagate better than electromagnetic signals in salt water!

Your goal is to decode a message contained in an acoustic signal.

The approach

The basic idea here is to use amplitude modulation to transmit signals through an acoustic channel.

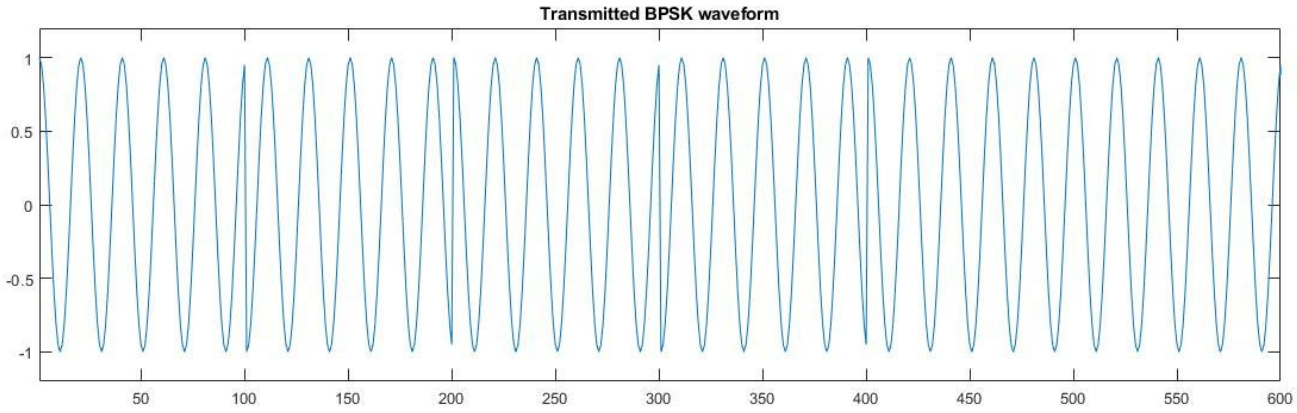
Digital information can be transmitted using a sequence of pulses, with positive pulses representing ones and negative pulses representing zeros. For example, a continuous-time (CT) waveform $m(t)$ can be used to represent a bit sequence '101011', as shown below:



In this example, a pulse of width 100 time units is used. The time between pulses is known as the Symbol Period.

The message signal $m(t)$ cannot be transmitted through the air. Instead, $m(t)$ will have to be translated to higher frequencies by multiplying it with a high-frequency cosine. This is the same approach used in Amplitude Modulation that you saw earlier in Signals.

If given the digital message signal $m(t)$, then the transmitted signal is $x(t) = m(t)\cos(2\pi f_c t)$, where fc is known as the carrier frequency. The $x(t)$ signal would look something like this:



The type of data modulation shown above is called binary-phase-shift keying: "binary" because there are one of 2 possible phase offsets to the cosine and "phase" because the information is carried in the phase of the cosine. Each transition from a 0 to a 1 and vice-versa results in a phase shift of the cosine.

Note that in practice, the frequency of the cosine will be considerably higher than what's illustrated in the graph, but we're using a lower frequency cosine to make the graph easier to read.

The waveform $x(t)$ is what should be transmitted from the speaker of an acoustic modem.

At the receiver end, the received signal $y(t)$ is multiplied by a cosine and lowpass filtered, thus producing an approximation to $m(t)$ which can be used to decode the transmitted bits. You'll find it helpful to refer to Homework 2 for how Amplitude Modulation (AM) works.

Your goal

Minimum viable product

The Acoustic Modem project uses the following files which you'll find on canvas:

- modem_tx.m – script to generate a transmit signal vector and store it into a .wav file. This is provided as a reference only. The signals have already been transmitted and recorded for you.
- modem_rx_starter.m – starter script to build your receiver
- short_modem_rx.mat and long_modem_rx.mat are two files containing received samples of a short and long transmission of data, respectively.
- find_start_of_signal.m – function used to detect when the signal is present in the received data

- BitsToString.m
- StringToBits.m

Your job is to decode the messages contained in the two .mat files. Start with the short file. The message in it is "Hello." The transmit code is given to you for reference in modem_tx.m

Possible directions for further exploration

1. Generate new data and transmit it over the air. Please see section below on practical considerations.
2. Reduce the symbol period. The symbol period directly impacts how quickly your data is transmitted. The smaller you can make it, the higher your data rate. (Note that you'll have to do the previous part first).
3. Transmit Quadrature signals. You can transmit two independent bit-streams through the channel, one modulated by cosine and another modulated by sine and have the signals added together. To decode, you simply multiply by the corresponding cosine/sine signal and lowpass filter the result.
4. Use multi-level pulses, e.g., 4 levels of pulses with possible heights of -1, -1/3, 1/3, 1.

Practical considerations

Lining up the signals

The data in the .mat files was collected as follows:

1. Run modem_tx.m
2. Transfer the resulting .wav file to a mobile phone
3. Create a recorder object in Matlab and start recording. Matlab has documentation on how to do this.
4. Play the .wav file on the mobile phone about 2 feet away from the laptop that is recording on Matlab.
5. Store the recorded data in the vector y_r.

Since the receiver was started first (to ensure we don't miss the start of the transmission), the signal lives somewhere in the middle of y_r. You can plot y_r to see this. So we need to truncate y_r to start at the correct point. The standard way to do this is to prepend a known, but noise-like signal before the actual data-bearing signal. Then an operation called cross correlation can be done to determine the start of the signal. The cross correlation operation has already been done for you in the modem_tx.m and modem_rx_starter.m files, so you can use what's already there.

Flipping signals

Due to propagation through the channel between the speaker on the phone and the mic on the computer, there is a good chance that the signal could arrive with its sign flipped at the receiver. You could flip it by hand or implement some automatic way to check if a sign flip is necessary. Note that in the two data files we provided for you, it is not necessary to flip the sign. You will only need to do this if you choose to make your own transmissions.

Deliverables

1. A brief report that includes an introduction, block diagram, equations used, graphs of signals in the time domain, and graphs of signals in the frequency domain.

2. Link to audio recordings if you produced new ones.

3. Your code.

4. Short demo video (2 minutes or less)