The purpose of this assignment is to decode the message hidden in an audio signal. The audio signal contains a series of tones, each of duration 50ms, and each representing one ASCII character. Each 50ms tone will contain one or more cosines at eight different frequencies: 400, 450, 500, 550, 600, 650, 700, and 750Hz. The presence or absence of each frequency can be thought of as a bit. Since there are eight frequencies, the tone therefore represents an 8-bit number; the value of that number will map to an ASCII character.

Confused? Read on: Start by loading the data file called `message_1.mat`. It contains an audio signal named `tone` and a sampling frequency named `fs`.

In order to decode the message, follow these steps:

1. Isolate the first 50ms of the signal
2. Use the `myFFT` function to determine and plot the frequency content of the tone
3. Determine the presence or absence of each of the eight target frequencies
4. Look up the corresponding 8-bit number on an ASCII table to determine the character.
5. Repeat for the remaining characters.

ASCII is a standard code so any table will be fine, but this one seems nice: `https://www.asciitable.xyz/`

Again, the eight frequencies used are: 400, 450, 500, 550, 600, 650, 700, and 750Hz. These frequencies correspond in order from bit seven (most significant bit) to bit zero (least significant bit). For example, the character `M` has ASCII value `77` which corresponds to bit sequence `[ 0 1 0 0 1 1 0 1 ]`. Therefore we would expect its tone to contain energy at 450, 600, 650, and 750Hz.

Extra Credit / Honors Students Although this assignment can be done by manually inspecting the Fourier Transform of each 50ms window, it would be better (and way cooler) if you had an automated function that could do it all for you. Write a function whose only input is the tone. You can assume the window duration will always be 50ms and the sampling frequency will always be 16,000 samples per second. Your function should display the message to the Matlab command line using the `disp` command. Test your function on `message_1.mat` to make sure you get the same answer as you did before, and then test with `message_2.mat` which is too long of a tone to realistically decode by hand. Does it work?!

Questions - Is this an efficient way to encode text? How many characters per second are we currently encoding? What is the maximum number of characters per second we could achieve with this technique? Why?

What to hand in - Using the provided template, you should hand in a single page report written in MS Word with the following sections: Intro, Methods, Results, Discussion. Your report must answer all of the above questions as well as any other interesting observations you make along the way. You should decide for yourself what information is important to present in your report. The

purpose of the report is to convince me that you understand the assignment, so think about the most efficient way of getting your point across. This is a technical report, so avoid humor and colloquial jargon!

Teams - You may work by yourself or in groups of two. If you work in groups, put both your names on the paper, but only one of you needs to do the submission in Canvas. In all cases, you and/or your group may work with other individuals or groups, but you/your group's code and paper must be uniquely yours.

You should also turn in your code in one or more `.m` files that I can run directly in Matlab. The code should use meaningful variable names and be well commented and otherwise easy for me to read. No need to include the `.mat` files - I already have those.

Zip together the MS Word report (not a PDF) and the Matlab code and submit *a single zip file* through Canvas. You may work in teams of two or by yourself. Submissions should arrive by Monday 2/17/2020 at 11pm.