For this assignment, you will be uncompressing some audio clips that I have compressed using a compression algorithm of my own invention. The compression algorithm works by breaking the audio clip into windows, and then representing each window as a sum of cosines. You will be given data that correspond to a list of frequencies and their associated complex A values. Using these data, you will uncompress the audio by rebuilding the signal out of cosines.

The specifics of the compression algorithm are as follows (you need to know this in order to work backwards to decompress the data. Note that in all cases, the sampling frequency is $F_s = 44.1$kHz.

1. Divide the signal into 40ms windows (which corresponds to $0.04 \times 44100 = 1764$ samples)
2. In each window, determine the frequencies that contain the most energy. Store the frequency in Hertz and the `A` value for each of the strongest 25 harmonics in each window.

Your goal will be to reconstruct three audio clips (all three are compressed in the exact same way). Load the file named `ca1Data.mat` using the `load` command. This data file contains the variables:

`fs` the sampling rate, in Hertz

`freqs` a matrix of dimension `[nWindows,25]` that stores the frequencies of the strongest 25 harmonics in each of `nWindows` time windows

`A` a matrix of dimension `[nWindows,25]` that stores the complex coefficient of the various cosines indicated in `freqs`.

(note that there are actually three values of `freqs` and `A` which correspond to three different audio clips of different lenghts)

To rebuild the signal, you will have to create a vector that is $1764 \times$ `nWindows` samples long. In each window, you will need to create and sum the 25 cosine waves corresponding to the frequencies, amplitudes, and phases stored in `freqs` and `A`. Note that when you are done, your audio clip won't sound great, but you should be able to understand or even recognize what you're hearing.

Once you have reconstructed the audio from `A1` and `freqs1`, run your code on the other two examples. These are three very different audio samples (all compressed in the exact same way), so you can think about why our compression algorithm works better in some circumstances than others.

Finally, alter your code to vary how many of the frequencies you include in your audio reconstruction. For example, instead of adding in all 25 cosines, what happens if you only use the first five? The first 10? How many cosines does it take for the sound to become recognizeable? Does that number vary according to what type of audio has been compressed?

Honors Students Get the file `ca1DataHonors.mat`, which has three audio signals compressed using a slightly different compression method. In this case, the audio clip was broken down into *overlapping* 40ms windows. For example, the first window covered samples 1-1764 of the audio clip.

The second window covered samples 883-2646, the third window covered samples, 1765-3528, etc. Each window shifts by 882 samples. Each window is processed exactly as before (e.g. represented as the sum of 25 cosines). Use this knowledge to decompress the data in file `ca1DataHonors.mat`. Note that cosines from overlapping windows should be summed together.

What to hand in - Using the provided template, you should hand in a single page report 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!

You *must* work in teams of two, or by yourself, and you may consult with other teams for ideas. However, your methods, analysis, and write-up should be uniquely your own. You do not need to hand in any hard copies of your paper. Instead, you should submit a single zip file containing your paper (MSWord format ONLY; PDFs will be returned with no grade) as well as any code you wrote (well documented) via Canvas. Submissions should arrive by Friday 2/2/2018 at 11pm.

Finally: Each team will be asked to sign up for a very short (5 minute) meeting with the teaching staff to demonstrate their working code. During that meeting, you will be asked quesitons about your code and how you would make implement various changes (eg. "What would you change your code if the sampling rate changed?) as well as other comprehension questions ("How much data compression does this method achieve"?) Our goal is to determine how well you understand what you did.

Useful Matlab Plotting Commands

- `subplot`
- `xlabel`
- `ylabel`
- `title`
- `xlim`
- `ylim`
- `legend`
- `figure`
- `clf`
- `close`

You can read about all of these by typing `help` and then the command name at the Matlab command prompt.