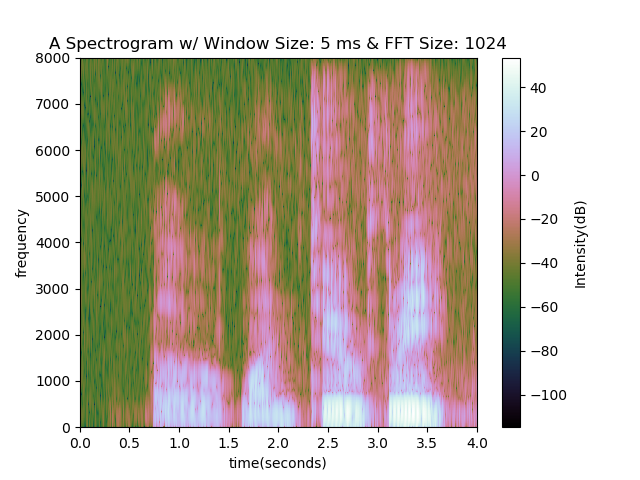
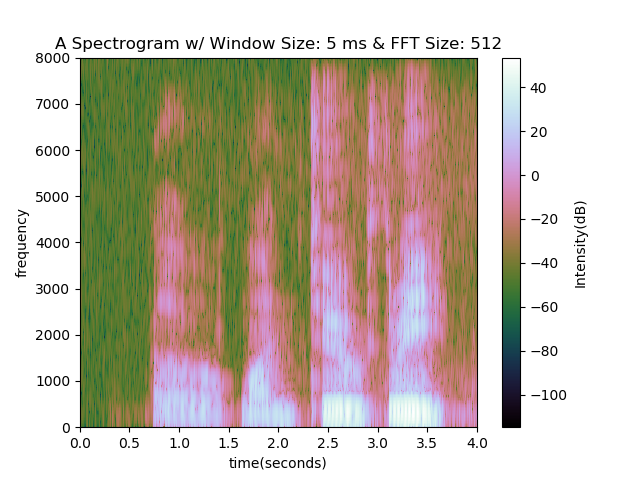
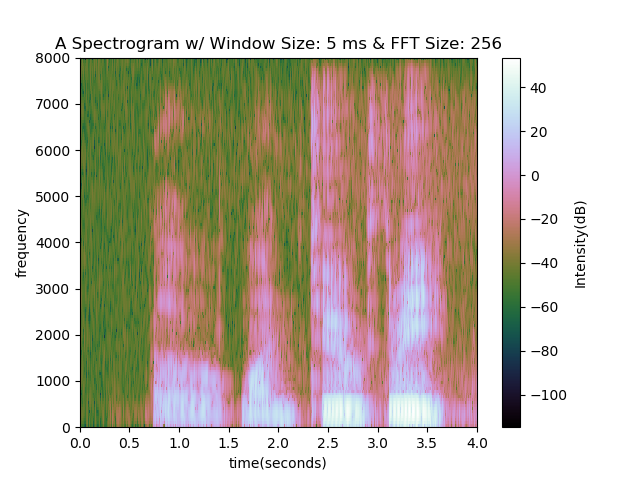
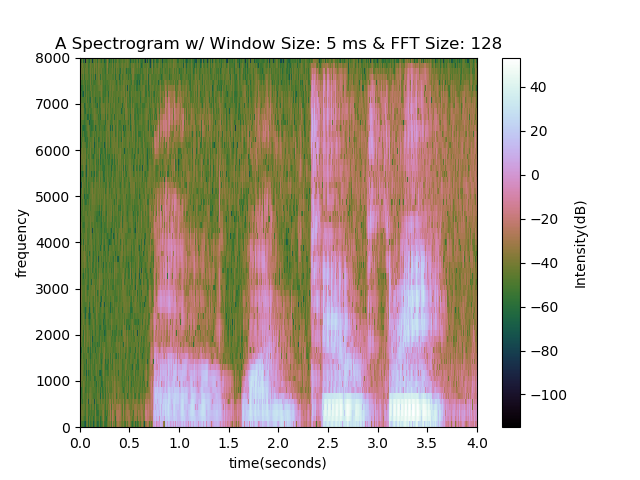
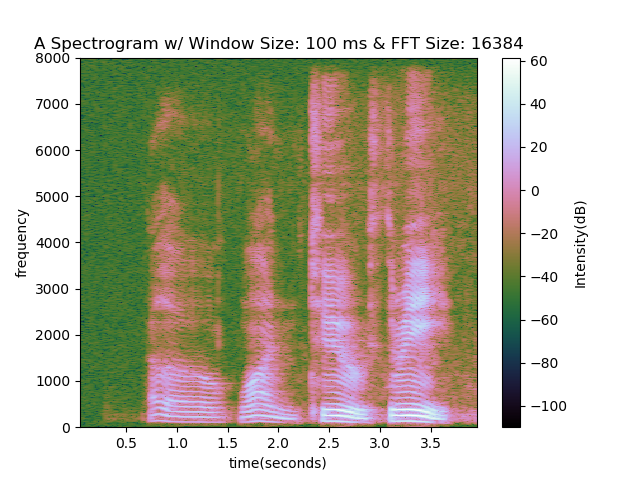
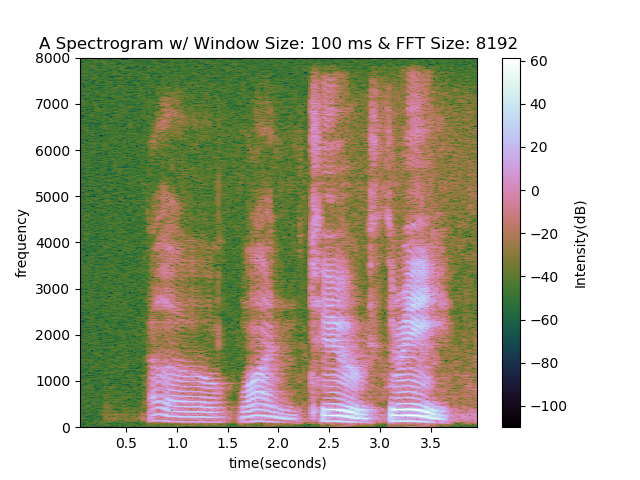
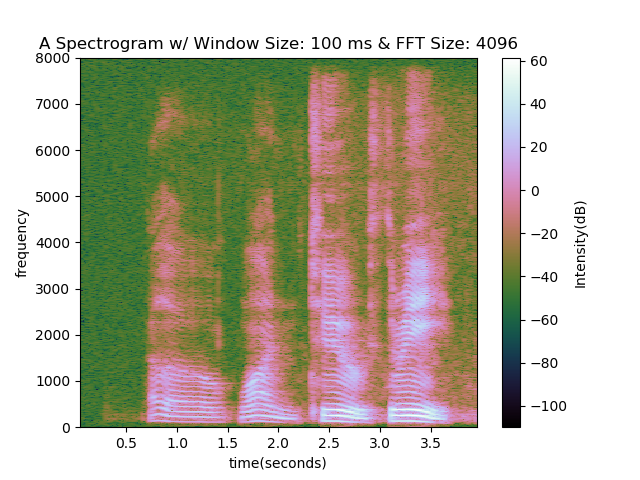
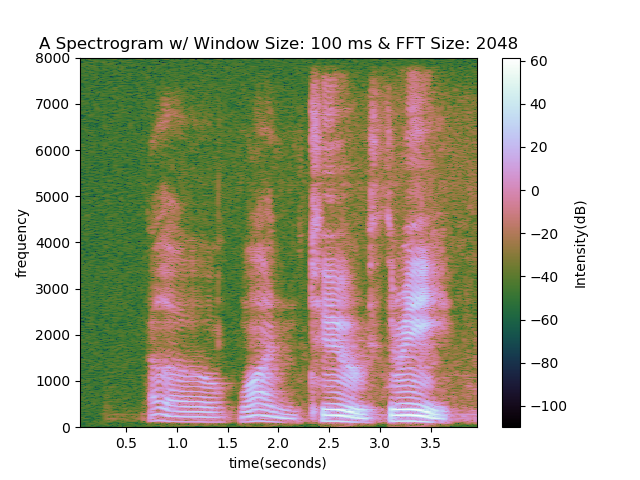
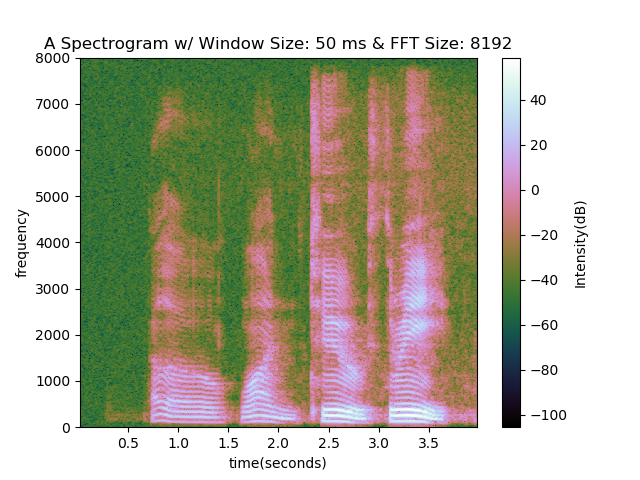
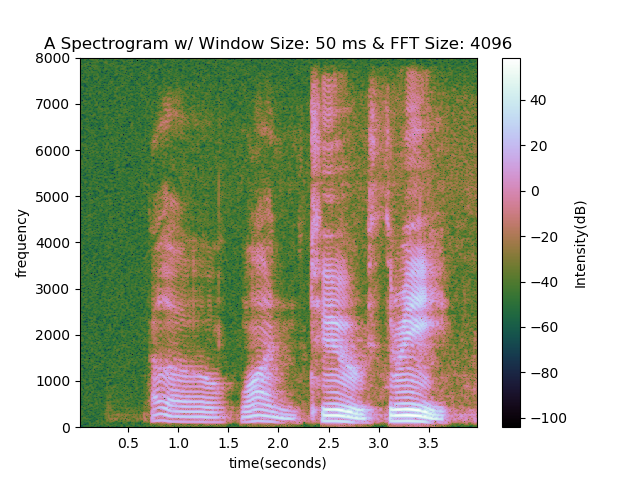
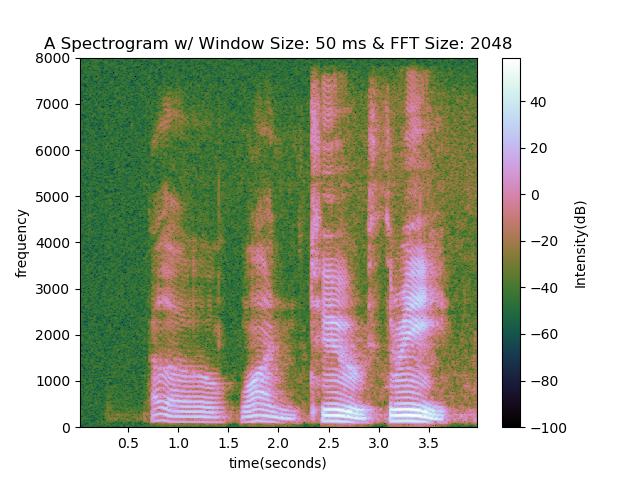
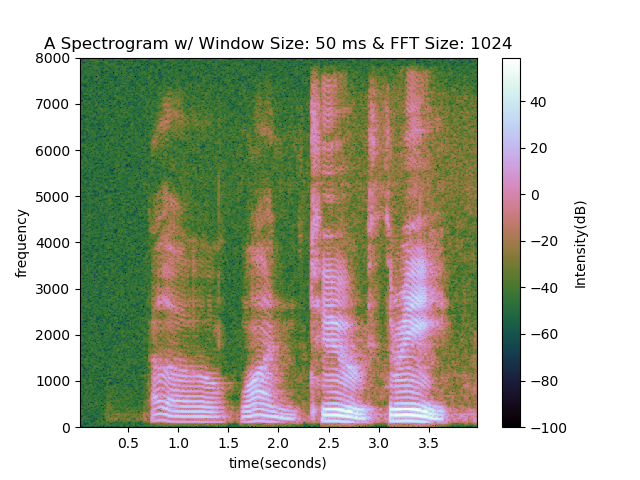
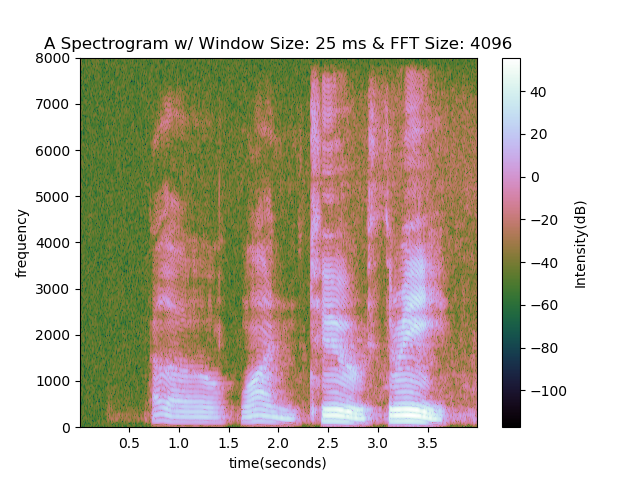
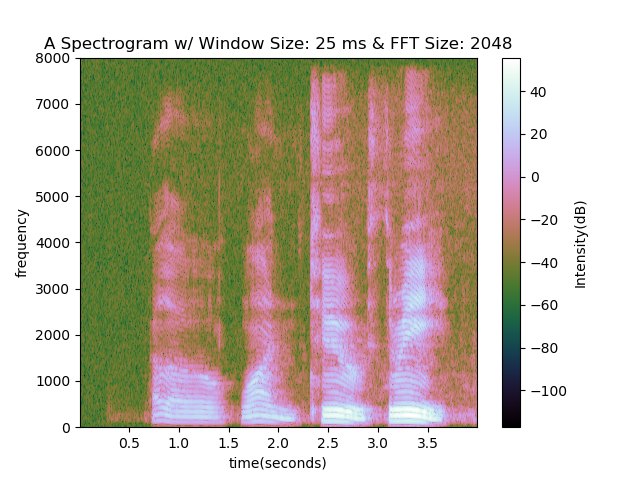
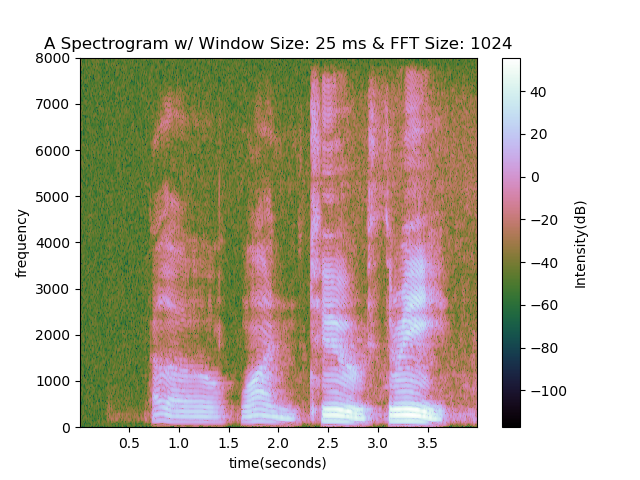
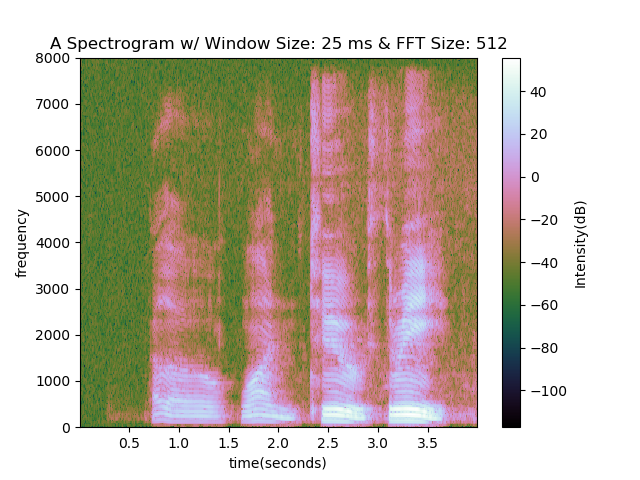
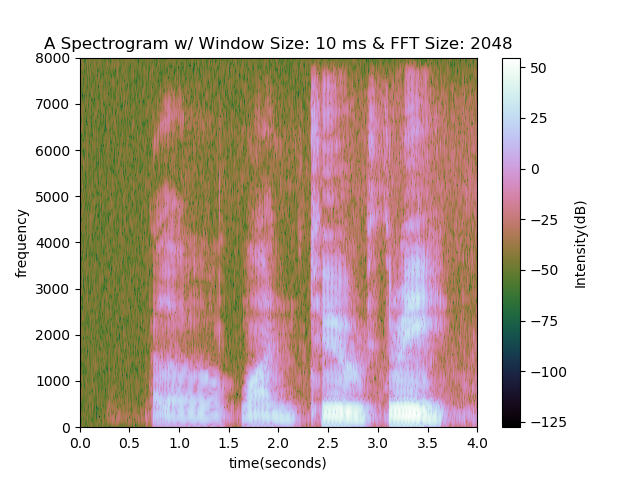
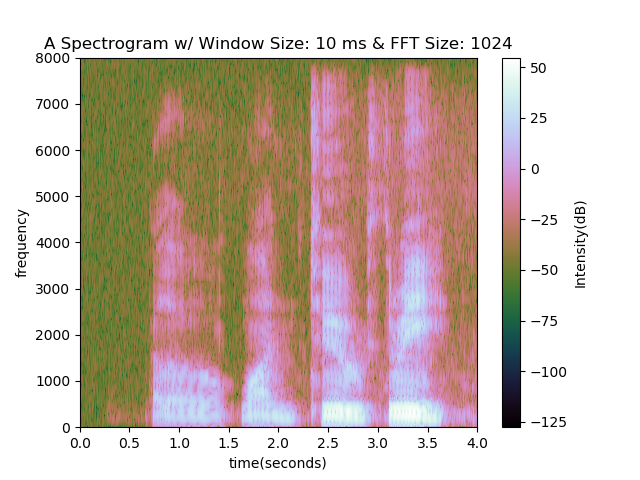
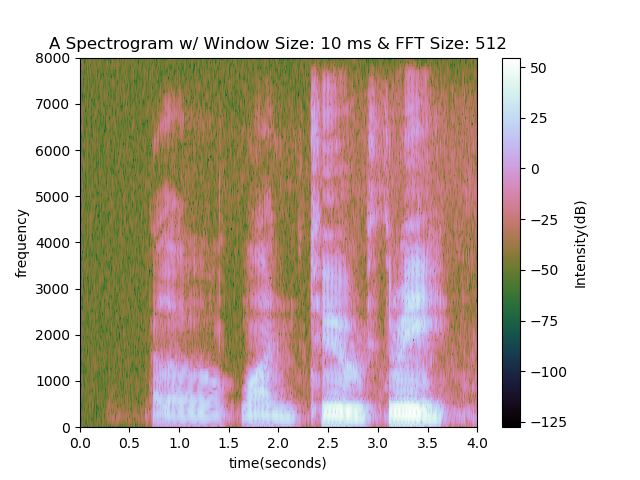
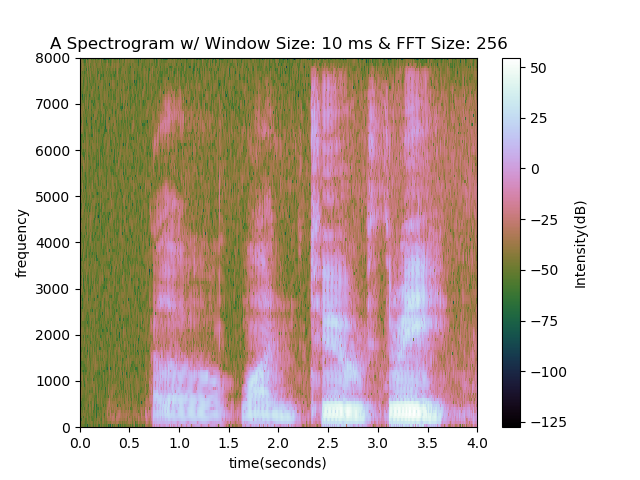
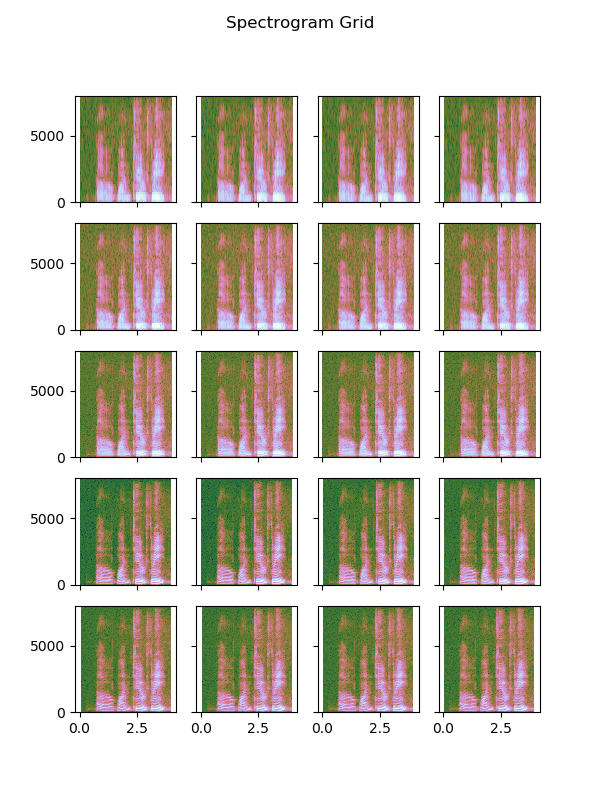
Problem 1

1. Each of the following 20 spectrograms represents an audio clip for a recording of me saying “testing one two three”. The spectrograms can be separated into groups each with a different window size increasing from 5ms, to 10ms, 25ms, 50ms, and 100ms. Within each group, four different FFT sizes were used by padding the windowed data to match a data size matching an increasing power of two.





1. The following is a grid of all the spectrograms in a single figure. Each row represents a different window size with the top row using a 5ms window and the bottom row using a 100ms window. Each column represents a different FFT size with the left most column using the smallest power of two that is also bigger than the windowed sample size.



1. When using small window sizes, the frequency resolution seems to be lower. Many of the harmonic features that can be seen with the larger window sizes can’t be seen with the smaller window size. The features seem to blur together.
2. The differences between the graphs that used the same window size but different FFT sizes seemed to be very small. There were small areas occasionally that seemed to have some slight differences in how the intensity varies, but there were no major differences that would prevent me from visually differentiating one spectrogram from another as long as the window sizes were the same.

Problem 2

1. A
2. B
3. C

Problem 3

Can use python fft

Create own: speech windowing, calling fft, taking magnitudes, mel-warping, avging fft magnitude outputs, log transforms, calling ifft transform (same as dct), compare delta features.

1. A
2. I used a hamming window function:

Where N = the width of the window which in this case is 25ms or 400 sample points. With the windowed audio sample, I then ran it through the MFCC warping and filtering algorithms.

1. The full matrix is of size 26 x (T – 2M) because of the parameters that were chosen. When creating the MFCC, the first 13 coefficients were chosen to be kept for further analysis. This means when the delta features are computed, there are 13 sets of features to analyze, one for each coefficient. Also, the length of the matrix is determined by the size of the recorded audio data and the window size used when calculating the delta features. With a window size of 2, there are two data points at the beginning and end of the data that the algorithm can’t compute since the algorithm would require reaching out of bounds in those areas. That leaves the overall length of the feature matrix to be T – 2M.