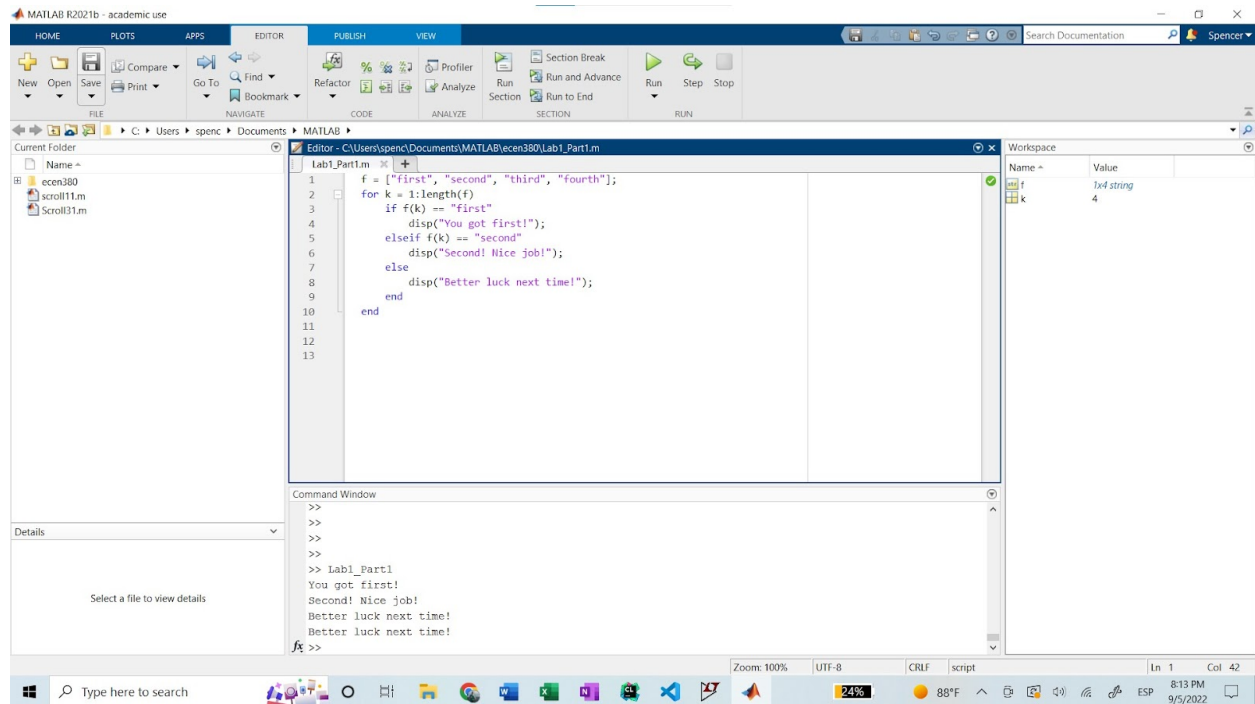


## Lab 1: Introduction to MatLab

### TASK 1:

#### MATLAB Exercises:

For the first part of this lab we went through the background part of the textbook and typed out and practiced all the example matlab practice problems, this helped me to get more familiar and use matlab. Next we had to write a simple for loop using if and else statements. I decided to have an array with some words or places after a race. Ex: First, second, third, fourth, etc. The for loop would loop through all the entries and output a phrase depending on the input. If you place below second place it printed out "Better luck next time!". See screenshots below!



See link to see all of the examples that were practiced in the Matlab command line interface.

[https://drive.google.com/file/d/1KKPEJLtWNVqzGs0e0\\_29RMjfal7xt24t/view?usp=sharing](https://drive.google.com/file/d/1KKPEJLtWNVqzGs0e0_29RMjfal7xt24t/view?usp=sharing)

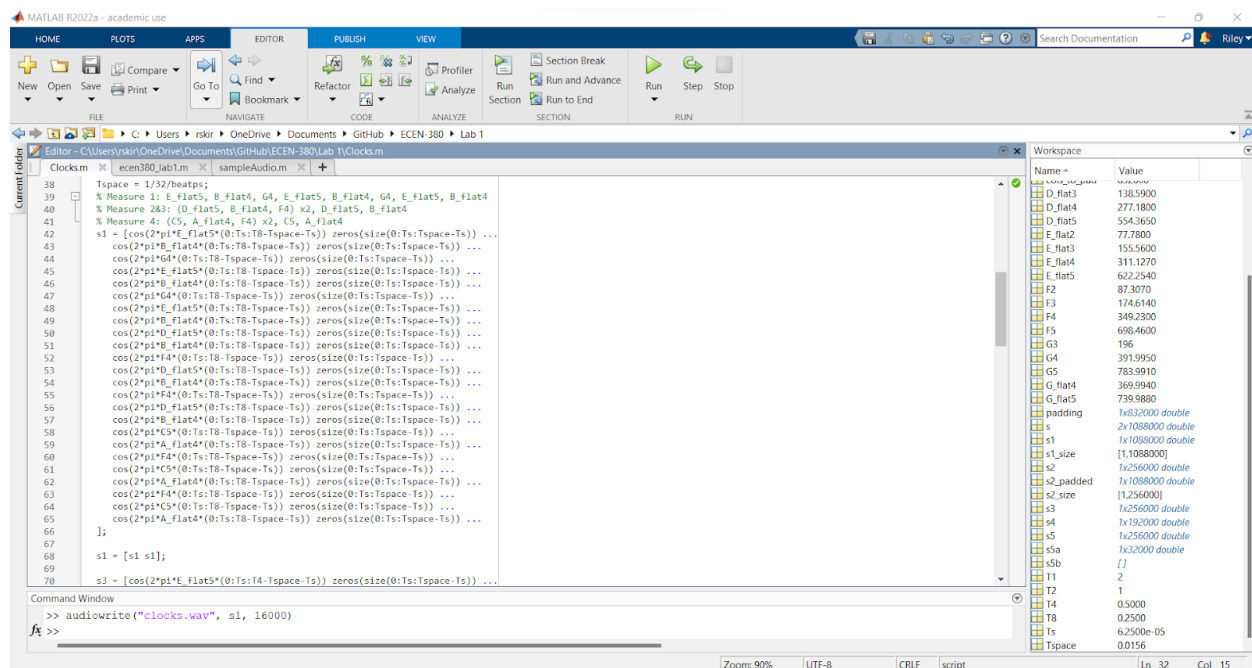
After using the audiowrite command to create sound1 at 16000 Hz sample rate. A sample rate is how often it takes the piece of data from the signal. For sound2 we had

a sample rate of 48000 Hz. Sound2 had a significantly higher pitch and a shorter sound clip. This is because it was being sampled 3 times as often.

## **TASK 2:**

### **Sound Production**

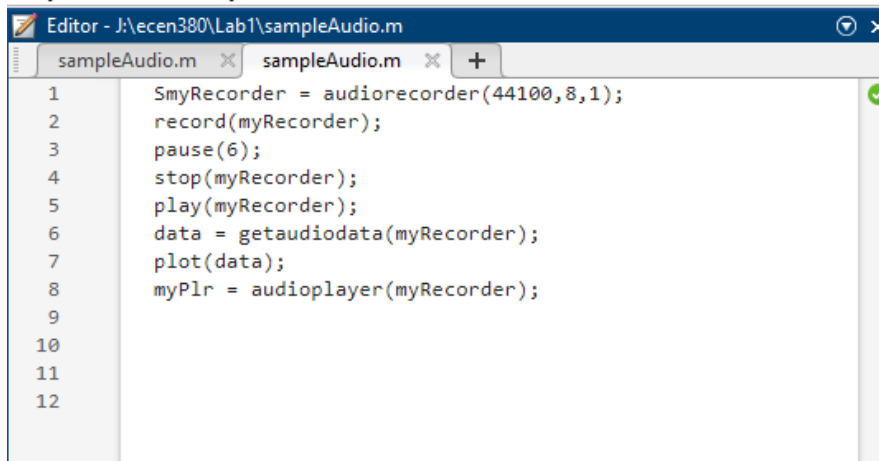
For part 2, we chose to create the song Clocks by Coldplay. In order to do so, we created variables for each frequency. Using those variables, we could create an array of sinusoids that represent the song. Then using the MATLAB soundsc() function we were able to play the song and using the audiowrite() function we saved the audio to a .wav file. The code for all of this part of the lab and the song can be found in this github repository: <https://github.com/rskirkwood/ECEN-380/tree/main/Lab%201>. We were able to add multiple notes on top of each other by concatenating the 2 different sound arrays. This meant the arrays needed to be of equal sizes, so we were able to pad one of the arrays with zeros because one array contained mostly whole notes and the other was eighth notes so it was much longer.



## **Audio Sampling and Playback**

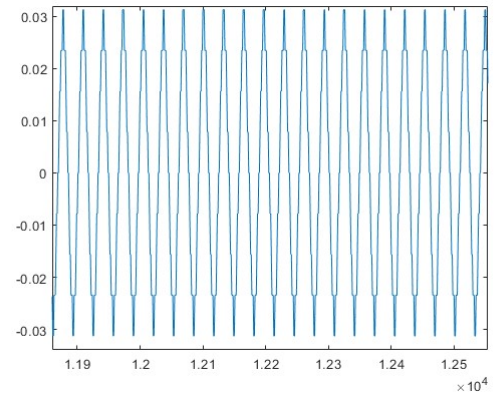
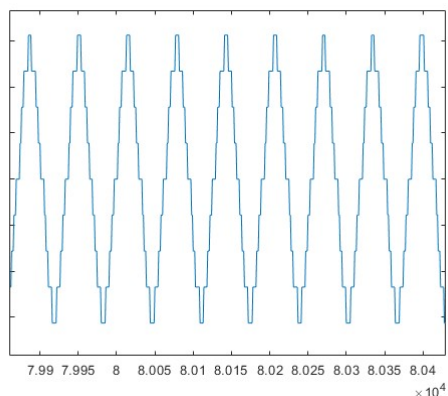
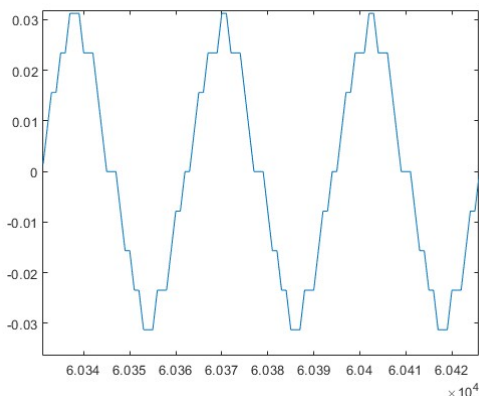
We used the built in sound card of the PC and Matlab to process and record different sound signals. We connected the function generator to the oscilloscope to the black breakout box using the BNC T adapter. The black box was connected to the lab

machine to plug in headphones as well as our input from the function generator and oscilloscope. We made sure that our peak to peak voltage was under 1 Volt so that we didn't damage the sound card or the motherboard. We then used the audiorecorder and audioplayer functions in matlab to create a sound at a specific frequency and amount of time. We were able to play around with the frequency and make the sound higher and lower. We learned how to play the sound, pause it for a certain amount of time and then record and analyze the data.

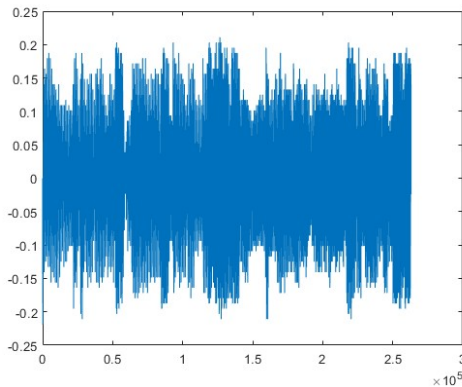


```
Editor - J:\ecen380\Lab1\sampleAudio.m
sampleAudio.m x sampleAudio.m x +
1  SmyRecorder = audiorecorder(44100,8,1);
2  record(myRecorder);
3  pause(6);
4  stop(myRecorder);
5  play(myRecorder);
6  data = getaudiodata(myRecorder);
7  plot(data);
8  myPlr = audioplayer(myRecorder);
9
10
11
12
```

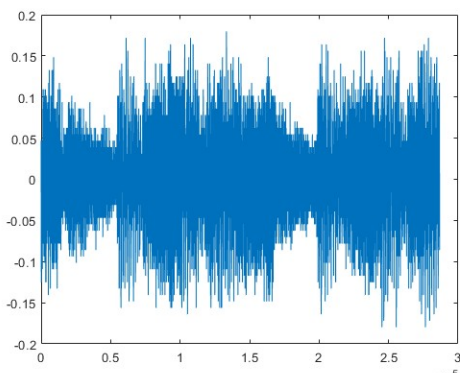
Next we began to change the frequency of the wave. We noticed that as we increased the frequency on the wave generator the waves got closer and closer together, while maintaining the amplitude. As we increased the frequency the pitch of the sound got higher and higher.



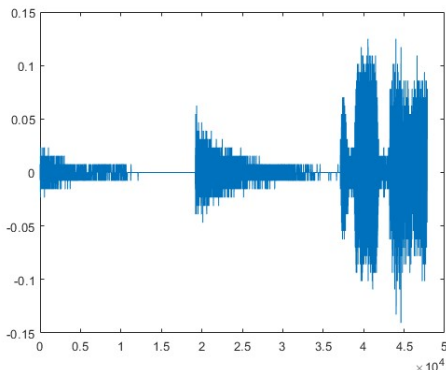
Next we used an aux cord with our phone, we played the “Can’t Hold Us” by Macklemore and tested the sample rate. Shown below a few of the sample rates that we tested and what their graphs look like.



High Quality (44k Hz) - The highest quality was 44k Hz and we can see that the graph is very high quality. Song was very clear



Common Audio Sampling Rate (44.1k Hz) - This is the common CD and MP3 quality sampling rate. The quality is good, but not as good as the 44k Hz. The song was good, but not great quality.



Low Quality (8k Hz) - This was the lowest quality, and it was very apparent in the graph. The graph was fuzzy and very unclear. The sound was also not recognizable.

## **Conclusion**

In conclusion, we learned many of the important and useful functions for Matlab. This will be very helpful for the rest of the semester as we use Matlab for doing the math and creating, graphing and manipulating sound vectors. We also learned the value in picking a proper sampling frequency in order to keep the quality of the signal. As we saw from the final part of this lab, a low sampling frequency can miss a lot of signals and not properly portray the signal that was transmitted.