Code Lab 1: Audio Signals and Complex Numbers

Prof. Jordan C. Hanson

January 21, 2025

Abstract

In this activity, we begin to draw a connection between complex numbers and sinusoidal signals. This connection will be further developed as the course proceeds. First, we will create a sinusoidal signal at a single frequency, use it to generate audio data, and play the sound. Second, we will create complex digital data and show that it produces equivalent sounds.

1 Generate Signals at a Single Frequency

Use the following code to create digitized sinusoidal data in Octave:

```
clear;
close;
home;

fs = 44000.0;
dt = 1/fs;
f = 440.0;

t = 0.0:dt:0.1;
S1 = sin(2.0*pi*f.*t);
S2 = sin(2.0*pi*f*2.*t);
```

The first three lines are standard housekeeping in Octave. We delete any persistent data or variables with `clear`, and we close any open figures. We also send the cursor to the top with `home` if we are working in an interactive Octave environment. Next, we define a *sampling frequency*, set to 44 kHz. A Hz is equivalent to one oscillation per second, and 1 Hz is s^{-1}. The variable `dt` or Δt is the time of each sample, 1/fs. **Show that Δt is about 23 μs.** Next, we define an audio frequency, set to 440 Hz. We then create sampled times between 0 s and 0.1 s, in intervals of Δt.

The variables `S1` and `S2` contain sampled, digitized sine waves with frequencies of f and $2f$, or 440 Hz and 880 Hz. The following code *concatenates* four copies of each signal, in an alternating sense, using the `mod` function to check if i is even or odd:

```
S_all = [];
for i=1:8
    if(mod(i,2)==0)
        S_all = [S_all S1];
    else
        S_all = [S_all S2];
    endif
endfor
```

Play the audio using the following code:

```
player = audioplayer(S_all,fs,8);
play(player)
```

2 Using Complex Exponentials

Replace the code that creates the signals `S1` and `S2` with complex exponentials like $\exp(j\phi(t))$. Play the *real part* of the signal, then play the *imaginary part*. What results do you hear?