# Acoustic Impedance Laboratory - Background

## Impedance

Impedance (Z), in general, is how hard you have to "push" to get a "unit response" from any system.

In electrical system, impedance is voltage divided by current. So (for example), for a resistor, impedance = resistance.

What is the impedance of a capacitor? For direct current, it's infinity… you can't run DC current through a capacitor, of course. But an AC voltage placed across a capacitor will result in some current flow… less current at lower frequencies, of course. In addition, the current will not be in phase with the AC voltage, because the voltage is proportional to the charge (the integral of the current). To keep track of the phase of the response, Z is complex number.

$$Z_{cap} = \frac{1}{i\omega C}$$

This makes sense if we use complex notation to represent the voltage and the current:

$$I(t) = I_0 e^{i\omega t}$$
$$V(t) = V_0 e^{i\omega t}$$

When using complex notation, it is understood that the actual voltage and current are the real parts! Writing

$$e^{i\omega t} = \cos \omega t + i \sin \omega t$$

and using the above impedance of a capacitor, you should be able to show that the voltage across a capacitor *lags* the current by 90°: if the current is a cosine, the voltage is a sine.

## Acoustic Impedance

In a sound wave, pressure does the pushing and the "current" is air flow. In a pipe, the impedance is the ratio of the pressure amplitude to the volumetric flow. Just as in the case of electrical circuits, these are not necessarily in phase (in fact they are usually out of phase) and thus Z is complex.

However, for a sound wave moving down a pipe (in the +x direction, say), the pressure and the flow *are* in phase! In a sinusoidal wave, for example, the peaks of the pressure are also where the air is "flowing" most, in the +x direction.

However, in musical instruments (and in general), a pipe will have waves going in both directions. For a wave going in the –x direction, the flow is 180° out of phase with the pressure.

(At the pressure peaks, the air flows in the –x direction.) So the impedance measured at a point in a pipe will in general be complex.

**Acoustic Impedance Lab Outline**

The first day, we will use a GUI program to explore sounds made by pipes and how sound reflects off pipe ends. The second week, you will make measurements that will allow you to find acoustic impedance.

To find acoustic impedance, you'll measure the sound with two microphones separated by 4 cm. Your job is to use those measured signals to find the left-going wave and the right-going wave. In our case, we don't need to worry about the absolute volumetric flow… we are just interested in finding something proportional to the flow. For the right-going wave (assuming right = +x), the flow is proportional to the pressure; for the left-going wave, the flow is proportional to –pressure.

The calculation is best done using MatLab. There's a starter "stub" of a program, "TMAnalyze", that includes matlab commands to read in the save waveforms from each microphone. The rest is up to you! There is also some "fake" data, "t-mics" and "t-waves" that you can use to test your analysis program. (t-mics is the recorded microphone signals, and t-waves is the correct decomposition into left and right-going waves.)

Of course, impedance depends on frequency, so you need to Fourier transform the waves and consider each Fourier component separately. Eventually, you'll get a plot of "relative impedance" vs "frequency". (The best thing to plot is the magnitude of the impedance, since impedance is complex.)

**Signals, Fourier Transforms, Matlab**

Audio recording will be done at 44100 Hz (samples per second), 16 bits, two channels (one for each microphone.)

The typical signal record you will use is 16384 points.
The matlab command g = fft(signal) will fourier transform the signal into frequency space.

The fft implicitly assumes that the signal repeats after 16384 points. Thus, the signal can contain only sinusoids that repeat exactly in 16384/44100 = 0.3715 s. Another way of saying the same thing is that the signal contains only *multiples* of frequency 44100/16384 = 2.6917 Hz (include DC, frequency = 0.) The vector g will be complex; the first value, g(1), is the DC component, the 2nd value, g(2), is the fourier component for frequency 2.6917 Hz, g(3) is the component for frequency 5.3834 Hz, etc, *up to the 8196th element.* **Then things change.**

Since the signal is real, it must be the case that $g(\omega) = g^*(-\omega)$, where the * means complex conjugate. (It may help to write out the complex exponentials to see what you need to do to make the imaginary parts cancel!) Matlab stores the fourier coefficient corresponding to a frequency of -2.6917 Hz in the very last element; the component for frequency -5.3834 Hz is in the 2nd to last element, etc. (If you were worried about things lining up, the bin containing the highest frequency (called the Nyquist frequency), g(8196), is always real. So it doesn't need a complex conjugate partner. )

When you write your analysis program, you may want to modify phases of fourier components (for example, to shift a wave left or right, you add or subtract a phase that varies linearly with frequency). My recommendation is to only worry about phases for points 1:8196. Then, zero out the top half of the fourier transform and rewrite it as the conjugate symmetric reflection of the lower half. Like this:

```
g([nFFT/2 + 1:end]) = 0;
g = g + conj(g([1, end:-1:2]);
```

(nFFT = 16384). A good test is the inverse fourier transform, ifft. If matlab's ifft gives you a complex result, then you fed it input that wasn't conjugate symmetric. (All real waves are real!)

## Batteries

Please try to remember to turn off the microphones at the end of lab. Otherwise you'll drain the batteries!