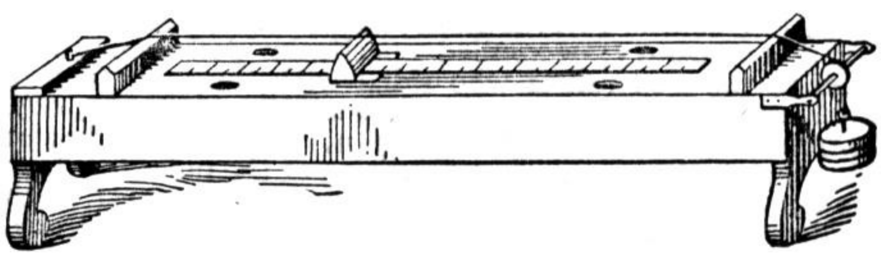
Lecture 2

# FOURIER TRANSFORMS

## Bases of Exponentials

The bases we chose for Fourier transforms on $l^2(\mathbf{Z}), L^2(\mathbf{Z_K}), L^2(\mathbf{R}), L^2[0,\ 1]$ are all bases of exponentials. We want to discuss why complex exponentials might be reasonable choices, and we'll do this by looking at special characteristics the exponentials have for the three distinct function spaces.

The first space we'll look at is $L^2[0,\ 1]$. The exponential $e^{-2\pi inx}$ can be written as $e^{-2\pi inx} = \cos(2\pi nx) + i\sin(2\pi nx)$, so that we are really dealing with sines and cosines of different frequencies on $[0,\ 1]$. You can see why both sines and cosines are needed to represent a function on $[0,\ 1]$; $\sin(2\pi nx)$ is zero at $x = 0,\ 1$, so sines alone can't represent general periodic functions. However, sines are important historically in the development of the theory of music in ancient Greek philosophy. The early experiments by Pythagorean philosophers were based on the sound given off by a *monochord*. The monochord consisted of a single string, held down at either end, and stretched taut so that it could vibrate. If we think of the string having length one, and $f(x)$ as representing the height of the string, then the condition that the string be help down at either end is equivalent to requiring that $f(0) = f(1)$, which sine functions are perfectly adapted to do.



Now, imagine the string plucked at time $t = 0$. Think of the plucking as an initial height of the string at each point $x$: *height* $= f(x)$ As we let go, the string will change height over time, leading to other heights, $u(x,t)$, over time. Of course we need to have $u(x,0) = f(x)$, and $u(0,t) = u(1,t) = 0$. If the string motion is relatively small, it's easy to analyze what that motion has to be over time:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}$$

<u>Homework Problem</u> Show that $u(x,t) = \cos(2\pi nt)\ \sin(2\pi nx)$ solves this equation. What must $c$ be in this case?

Notice that $u(x,0) = \sin(2\pi nx)$ is a sine wave. As time varies, $u$ is called a *standing wave*, and it produces what is called a *pure tone* for the monochord. The case $n = 1$ is called the *fundamental* frequency (as no standing waves of lower frequencies can exist on the monochord), and the higher $n$ are called *harmonics*. The Fourier series representation of a function then has a musical interpretation: every sound on the monochord is a sum of pure tones.

The Fourier representation of sounds produced by a monochord is nice all by itself, but it turns out that the human ear uses vibrations of (biological analogues of) strings as its basic mechanism for hearing. Thus, for humans, the Fourier basis is the correct biological basis for understanding hearing – an idea that we'll return to. For now, we remark that this fact is at the basis of the mp3 method of music compression.

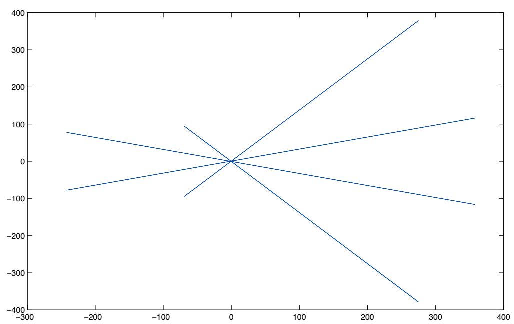
Let's see the Fourier transform in action. The computations we'll be doing are based on real signals, which means they've all been sampled at some sampling rate.  This means we'll be using the Discrete Fourier Transform. As discussed in Lecture, the fastest implementation of this is the FFT. Here's what it looks like in Matlab:

> x=linspace(0,1,1000);
% I created a 'sample space'; this samples the interval [0, 1] a thousand times. If you think of [0, 1]
% as one second, then you have a thousand samples a second -- a kiloHertz, kHz.

> y=sin(2*pi*x*200)+sin(2*pi*x*400);
% create a sum of two frequencies: 200Hx and 400Hz.

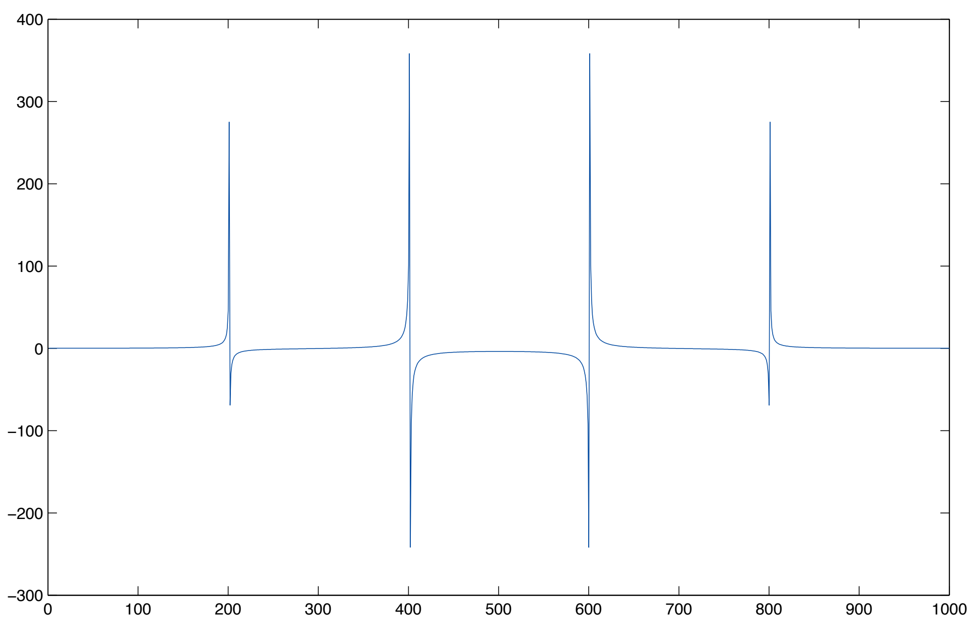
> z=fft(y,1000);
%tells Matlab to do the FFT on 1000 points (which it will write as $2^3*5^3$)

> plot(z)



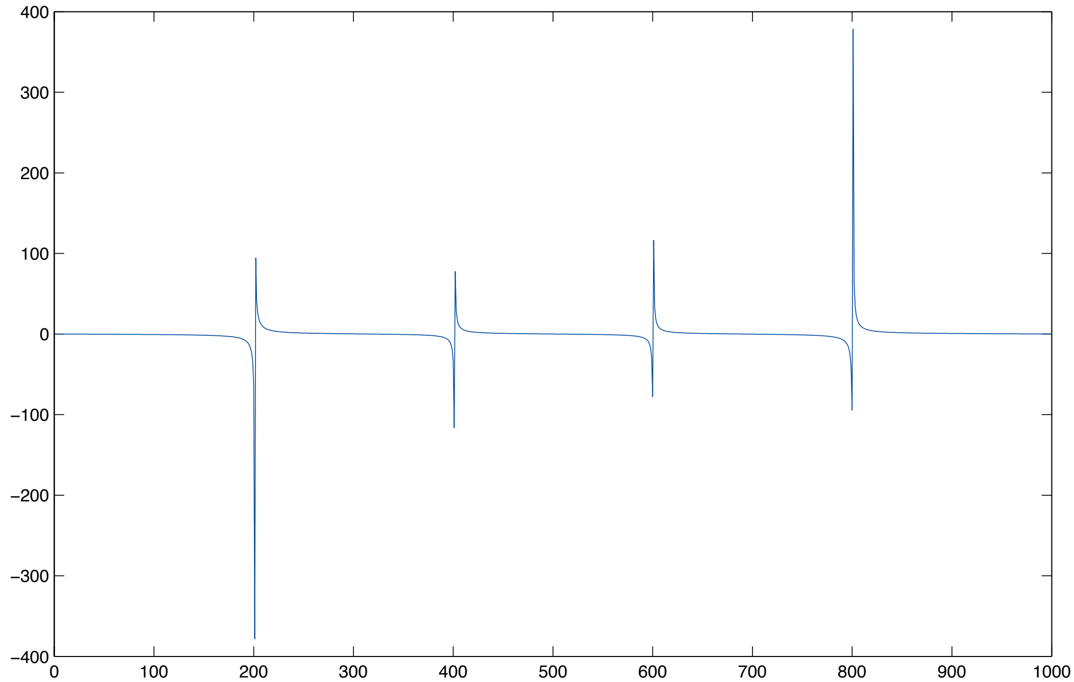

Whoa, that's weird! The problem is that z is a complex vector, with real and imaginary parts. Let's try just the real part:

> plot(real(z))



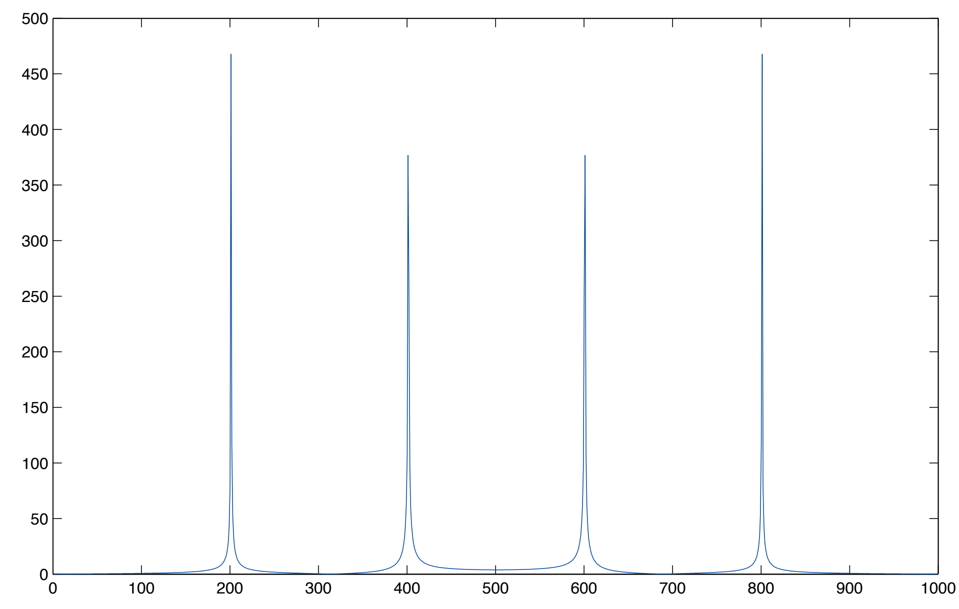

A bit better: there are peaks at 200 and at 400, but symmetric peaks at 600 and 800 as well. As there's no 600 or 800 cycle wave in my signal y, something peculiar is going on. Let's look at the imaginary part:

> plot(imag(z))




The same kind of strangeness,  but this time anti-symmetric peaks at 600 and 800 as well. Put it all together by plotting the absolute value:

```
> plot(abs(z))
```




This, of course, removes the negative components of the transform completely. This kind of graph is called a *power spectrum* -- with the understanding that it's only a part of the Fourier transform, the magnitude. Whether one has negative or positive values, real or imaginary parts, that's called the *phase* of the transform.

Time to stop being mysterious. First of all, we're sampling at 1000Hz. By Nyquist, the highest frequency we can "see" is 500hz. This explains, in a way, why the frequencies from 500hz to 1000hz are duplicates of the information from 0Hz to 500Hz.

Homework Problem We can be more precise than that. If $f$ is in $l^2(\mathbf{Z_K})$, then $\hat{f}$ is also defined on $0,\ 1, \ldots, K-1$. Prove that

$$\begin{aligned} \Re(\hat{f})(j) &= \Re(\hat{f})(K-j) \\ \Im(\hat{f})(j) &= -\Im(\hat{f})(K-j) \\ \left|\hat{f}(j)\right| &= \left|\hat{f}(K-j)\right| \end{aligned}$$

Because of this, it's traditional to plot only half the FFT values; typically

```
> plot(abs(z(1:500)))
```

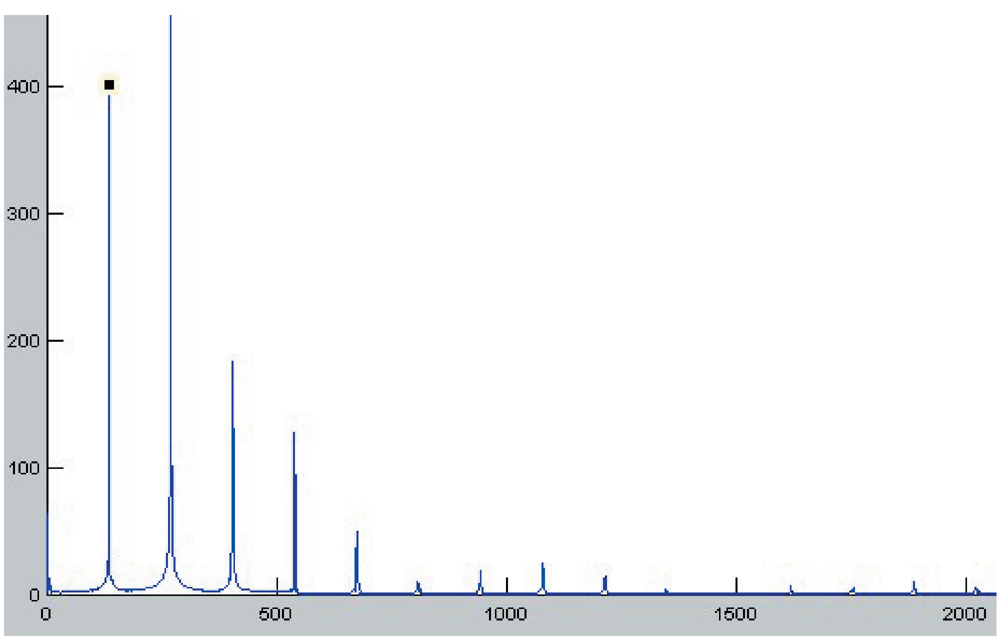
Lab Project Now that we have the FFT, let's start using it. Download the files one.wav through seven.wav, and compute FFT's. These are the tones used in touch-tone phones. What does each represent, and what do you learn about touch-tone encoding?

Lab Project Download the clip from Madonna's version of American Pie. It's sampled at 44100; load it into Matlab. Do the following computations:

```
> z=fft(x, 325240);
> good=ifft(z);
> badreal= ifft(real(z));
> badabs= ifft(abs(z));
```

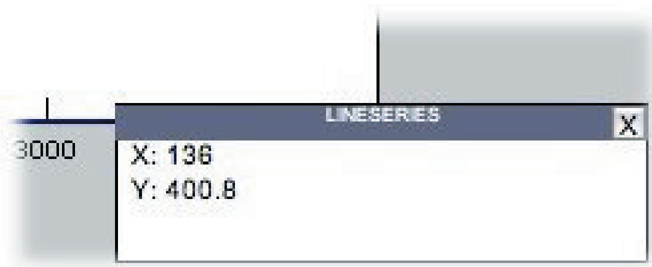
Save these files, then play them. Conclusion?

Lab Project Download the clip of a violin playing the note A4. It's sampled at 22050; load it into Matlab. Compute the FFT, and plot the power spectrum. You'll get something like the following:



Find the frequency of each of the first five or six peaks, and show that they are all (approximately) muliples of the lowest frequency. Thus, the violin is playing a fundamental frequency and its harmonics.

Remark: if you can’t find the peaks using matlab and trickery, you can do it by hand as follows. Click the circled button below to get yourself into ‘point selecting’ mode. Then right-click and follow the pop-up menu to ‘display characteristics’ and select ‘in a box’. Then, when you click on a peak, the frequency and height of each peak is shown in a small box.


Lab Project Repeat the process with the linked file of a *didgeridoo,* a musical instrument of the Aboriginal peoples of Australia (sampled at the rather unusual frequency of 16000 Hz). Is it still true that the frequencies are a multiple of the fundamental? This is a tube with one end closed (?)

Lab Project Repeat with the linked file of a chime (sampled at 48000 Hz). The chime doesn’t have fixed ends. It vibrates by moving up and down along its length, like a string, but the ends aren’t held down. So there’s no reason to suppose that the frequencies it can support have to be multiples of a fundamental.