An Introduction to the Discrete Fourier Transform with Python

If you have ever:

1. opened a JPEG file on your computer or phone;
2. played an MP3 song;
3. used voice recognition capabilities of Amazon's Alexa;
4. used music applications (Shazam, synthesizers);

then you have used some variant of the **Discrete Fourier Transform** (DFT).

Its efficient implementation, the **Fast Fourier Transform** (FFT), is considered to be on of the most useful algorithms in computer science.

The goal of this video series is to:

1. understand the math behind the algorithm;
2. use Python to analyze and manipulate audio files using the DFT:
   - Analyze the frequency content of a note/chord played on various instruments.

Sampling an analog signal:

Given a signal $y(t),\ t \in [0, L]$, sample the signal at some sampling rate fs for a total of N samples: $\big[y[0], y[1], \ldots, y[N-1]\big]$.

The Discrete Fourier Transform:

The DFT converts: $\big[y[0], y[1], \ldots, y[N-1]\big] \rightarrow \big[Y[0], Y[1], \ldots, Y[N-1]\big]$ by:

$$Y[k] = \sum_{n=0}^{N-1} y[n] \exp\big\{-i2\pi kn/N\big\}, \quad k = 0, 1, \ldots, N-1$$

The magnitude of the Fourier coefficients $Y[k]$ measures the magnitude of the frequency $f_k = kf_1$ in the signal, where $f_1 = 1/L$ is the fundamental frequency.

Using Python to compute the DFT:

See `lecture1.ipynb`.