Code Lab 4: Fourier Transforms, Spectra, and the FFT

Prof. Jordan C. Hanson

February 18, 2025

Abstract

In this activity, we create a digitized, sampled sinusoidal signal, plus noise, and compute its Fourier transform using the `fft` algorithm in `octave`. The signal should be visible above the *noise floor.* We will tune the various DSP parameters, and eliminate noise with low and high-pass filter algorithms.

1 The Full Code

For the full code, download the Code Lab 4 file from the course Moodle page. The title of the code is `FFT.m`. Run this code to create a graph of a *noisy sine wave* in the time and frequency domains. The Fourier transform magnitude is plotted in the lower graph. The `fft` algorithm produces positive and negative frequencies. Both are plotted, and the data should be similar.

2 Fine Tuning Parameters

Do not worry if you do not understand every aspect of the code. Our job here is to locate the following parameters in the code, and tune them to different values:

1. `fs`, the sampling frequency.
2. `f_sig`, the sine wave frequency.
3. `amplitude`, the number of modes in the signal spectrum.
4. `noise_sigma`, the standard deviation of our gaussian thermal noise.

3 Exercises

1. Change the sine wave frequency, and observe how the single frequency-mode moves around the spectrum.
2. Reset your parameters, and increase the noise so that the sine wave is not visible in the *time domain*, but remains visible in the *frequency domain.*
3. Reset your parameters, and copy the code from line 19 for the sine wave. Past it back in line 19 so that the signal `y` is equal to a sine-squared plus noise. Where is the frequency mode? Use trigonometric identities to show that the frequency of the sine-squared is the sum of the frequencies of the individual sine waves.
4. Reset your parameters, and activate the low and high-pass filters by uncommenting lines 22-25. Look up how the `butter` function works by running the `help butter` command in octave.
5. The arguments to the filters are how many "poles" the low and high-pass filters have, and where the cutoff frequency is located. The cutoff frequency is expressed as a fraction of the Nyquist frequency.
6. **Tuning the filters:** Tune the filter parameters to eliminate noise, except for the spectral region around the sine wave signal.
7. **Draw a graph** of your filtered signal output from `octave` below, including both the *time domain* and *frequency domain* data: