ECE 310 Fall 2023

Lecture 25

Short-time Fourier transform and spectrograms

Corey Snyder

Learning Objectives

After this lecture, you should be able to:

- Understand how we can use multiple short-window DFTs to obtain the frequency spectrum of a long digital signal as it changes with time.
- Explain the design tradeoff between frequency and temporal resolution when computing the STFT.
- Interpret the STFT of a signal using spectrogram plots.

Note: this lecture is bonus content on spectral analysis. Short-time Fourier transforms and spectrograms will not be tested on exams or seen on homeworks.

Recap from previous lecture

We have explored spectral analysis in the previous two lectures and developed useful tools for separating nearby frequencies and accounting for multiple spectral components of varying amplitudes. In this lecture, we consider how the spectrum of a digital signal may not be fixed for all time. In other words, we may have spectral components appear and disappear as a function of time. We will develop simple extensions of the DFT to account for this phenomenon.

1 Short-time Fourier transform

Thus far, we have considered performing spectral analysis on discrete-time signals of the form

$$x[n] = \sum_{m=1}^{M} A_m \cos(\omega_m n),\ 0 \leq n \leq N-1. \tag{1}$$

Equation 1 implicitly assumes that all M spectral components begin and end at the same time. Consider instead if we analyze an audio recording a musical scale played on a piano. A standard scale has eight musical notes and thus eight spectral components. However, each note is played sequentially after the previous note is complete. Looking at the DFT of this musical scale recording will capture each spectral component, but we cannot know the order in which they were played because the DFT takes us completely out of the time-domain.

A simple solution to this problem is to take several shorter DFTs using a class of algorithms known as the *short-time Fourier transform* (STFT). Let $x[n]$ be a length-N discrete-time signal and $w[n]$ be a length-L window function where $L < N$. The STFT of $x[n]$ will be a two-dimensional signal with time and frequency indices denoted by $X[n,k]$:

$$X[n,k] = \sum_{m=0}^{L-1} x[n+m]w[m]e^{-j\frac{2\pi k}{L}m},\ 0 \leq k \leq L-1,\ 0 \leq n \leq N-1. \tag{2}$$

The STFT given in Eqn. 2 computes the DFT of a length-L window of $x[n]$ starting at each time-step n. Thus, we compute N length-L DFTs to capture the time-frequency behavior of $x[n]$. In practice, computing a separate DFT for each starting point in $x[n]$ is unnecessarily expensive as adjacent spectra will be nearly identical. It is common practice to pick a *step-size* or *stride* to separate each DFT. Let $1 \leq S \leq L$ denote our step-size. We may then describe a more efficient STFT as follows:

$$X[t,k] = \sum_{m=0}^{L-1} x[St+m]w[m]e^{-j\frac{2\pi k}{L}m},\ 0 \leq k \leq N,\ 0 \leq t \leq \frac{N-L}{S} - 1. \tag{3}$$

We now have a more efficient STFT that computes $\frac{N-L}{S}$ length-L DFTs. Of course, there is a tradeoff here in terms of the amount of computation and the time resolution. A very small S, e.g. $S = 1$ like in Eqn. 2, incurs the greatest computational cost with the smallest change in time between windows, while $L = S = N$ requires only one DFT but has no temporal information. There is also a tradeoff to consider when choosing the window-length L.

1.1 Tradeoffs in computing the STFT

1.1.1 Frequency resolution

First, we know that our frequency resolution will depend on the length of our window, L. Recall from lecture 24 that the main lobe of each spectral component has width $\frac{4\pi}{L}$ with a rectangular window and width $\frac{8\pi}{L}$ for the Hamming window. Thus, increasing the size of the window improves our ability to precisely resolve the frequencies of the spectral components. However, increasing L degrades our temporal resolution. Consider the extreme case where $L = N$ and we are back where we started! We would lack any temporal information in this case.

1.1.2 Temporal resolution

There are a couple ways to consider temporal resolution. We may look at the amount of time covered by each window or the amount of time separated by each window. Let f_s be the sampling rate used to acquire the digital signal $x[n]$. We may then say:

$$\text{Time-width of each window} = \frac{L}{f_s}$$
$$\text{Time-separation between windows} = \frac{S}{f_s}.$$

The inherent tradeoff here is that frequency resolution improves as L increases while temporal resolution degrades as L increases. In practice, we may fix a worst-case main lobe width or time-width for the window in order to make the other resolution as good as possible. Consider, for example, our previous scenario of a musical scale played on piano. Suppose we know each note is played for 0.5 seconds and the audio is recorded at 10kHz. We would like more than one window to cover each note, so suppose we decide that ten windows per note will be sufficient. Thus, we want each window to cover 0.05 seconds, which corresponds to a window-length of $L = 500$ for each DFT.

2 Spectrograms

To interpret the STFT of a digital signal, we typically plot the time-varying spectrum using a diagram known as a *spectrogram*. A spectrogram typically has an x-axis that indicates time while the y-axis indicates the frequency. The intensity of the plot at each time-frequency location expresses the amount of energy for that particular frequency at that moment in time. Typically, we plot the magnitude of each spectrum in the STFT on a log-scale (db-scale) when plotting spectrograms. Figure 1 shows the DFT spectrum and spectrogram for an audio recording of a C major scale played on guitar. The audio here is recorded at 48 kHz and each note is held for roughly 0.5 seconds. We use a window-length of $L = 2^{12} = 4096$ and stride of $S = \frac{1}{3}L \approx 1,365$ samples. Thus, each window spans a little less than 0.1 seconds and each window overlaps

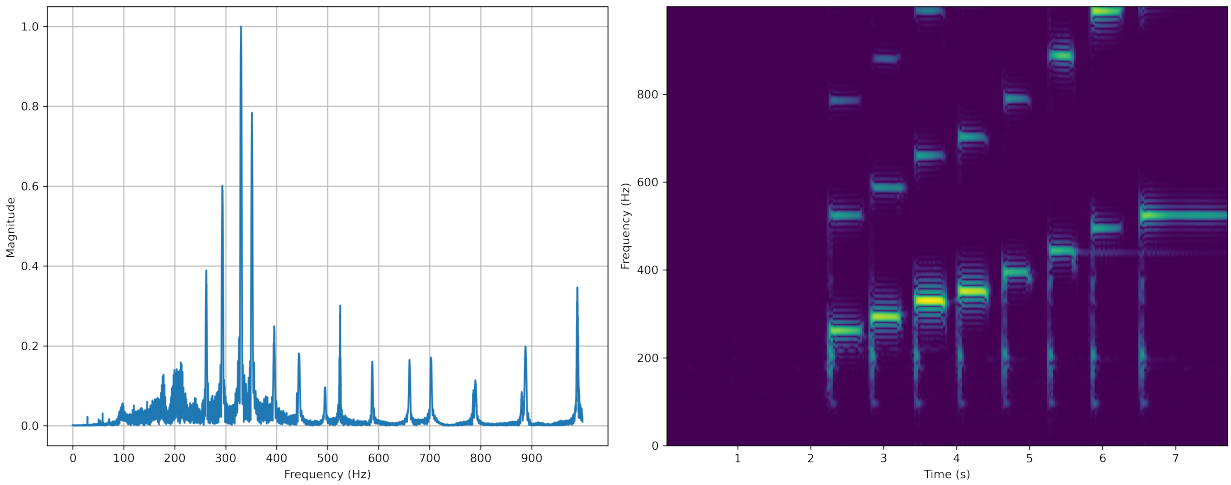

Figure 1: Frequency spectrum (left) and spectrogram (right) of a C major scale being played on guitar.

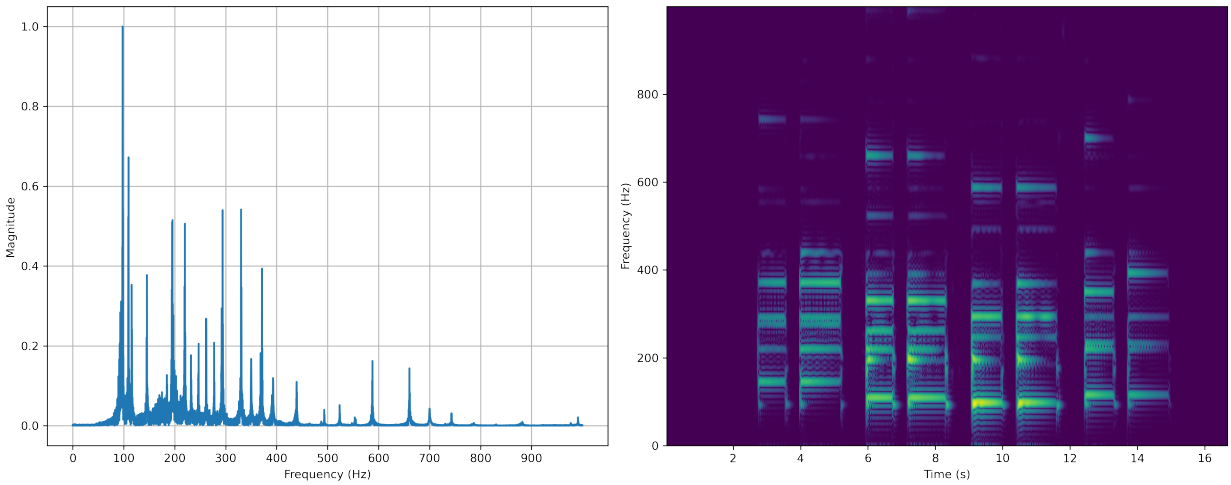

Figure 2: Frequency spectrum (left) and spectrogram (right) of the main chord loop from Best Part by Daniel Caesar (feat. H.E.R.) played on guitar.

for about 0.06 seconds. We also have limited the y-axis from 0 to 1kHz since none of the played notes exceed this range. We see a horizontal bar for each note that spans about 0.5 seconds as all eight notes are played in the scale. The additional frequencies seen about each played note are known as *overtones* and represent other resonant frequencies, e.g. integer multiple of the fundamental played frequency. There is also additional low-frequency information due to the contact of the guitar pick with the guitar strings.

We have also included a second set of plots in Fig. 2 to show the an example of multiple distinct sets of spectral components played simultaneously. This figure illustrates the spectrogram for the main chord loop of the song "Best Part" by Daniel Caesar (feat. H.E.R.) played on guitar. This chord loop contains four distinct chords each played twice. We can clearly see all four chords in the diagram and how each is played two times successively.