The Short-Time Fourier Transform and Applications

Yi-Wen Liu

4 March 2013

Last time, we defined the discrete-time Fourier transform as a time-to-frequency mapping for any signal in time whose duration might run from $-\infty$ to ∞. This is simple definition, and it leads to several nice properties such as the convolution theorem. However, to apply DTFT in the analysis of speech or musical sounds, we face the reality that these audio signals are non-stationary. The resulting DTFT is affected by all samples in time and is therefore not easy to recognize — for example, we might be interested in looking at the signal and answer questions like the followings,

- Which notes are being played on the guitar?
- What words are spoken?
- When is a kick-drum hit upon and what is the beat pattern?

Answers to these questions involves determining short-time properties, rather than calculating long-term averages. Consequently, we often need to trade between time- and frequency-resolutions. This can be accomplished by choosing the right length of Fourier transform. So we shall study *short-time* Fourier transform this week. The basic idea is to process the samples in a block-by-block manner. In this course, the retrieved block of samples is called a *frame*.

1 The Discrete Fourier transform (DFT)

1.1 DFT as DTFT sampled in frequency

Let L be the length of a block of samples $\mathbf{s} = (s_0, s_2, ..., s_{L-1})^T$. Then, the *discrete Fourier transform*(DFT) of $\mathbf{s}$ is defined as a length-L vector $\vec{S}$ whose components are

$$S_k = \sum_{n=0}^{L-1} s_n \exp\left(-jk\frac{2\pi}{L}n\right), \quad k = 0, 1, 2, ..., L-1.$$

When the above definition is compared to the definition of DTFT, we find that DFT is DTFT sampled in frequency. This can be understood by recognizing that

$$S_k = S(\omega)\Big|_{\omega=k\frac{2\pi}{L}}, \tag{1}$$

where ω is the frequency variable in the DTFT of a discrete-time signal $s(n)$ whose values are $s_0, s_1, ..., s_{L-1}$ from $n = 0$ to $n = L - 1$, and $s(n) = 0$ elsewhere.

In other words, we can think of the DFT of a block of samples as the DTFT of that block calculated at L equally spaced frequencies $\omega_k = k\omega_0$, where $\omega_0 = 2\pi/L$, and $k = 0, 1, 2, ...L - 1$.

Conversely, we can obtain the DTFT of a signal that has a finite length L by interpolating the DFT of that signal. Interpolation in the frequency domain can be achieved via appending zeros in the end in time. The number of frequency bins increases as more zeros are appended in the time domain.

1.2 Matrix representations and inverse DFT

We can write the length-L Fourier transform as an $L \times L$ matrix $\mathbf{F}$,

$$\mathbf{F} = \begin{bmatrix} 1 & 1 & . \;.\; . & 1 \\ 1 & e^{-j\omega_0 \cdot 1} & . \;.\; . & e^{-j\omega_0 \cdot (L-1)} \\ 1 & e^{-j2\omega_0 \cdot 1} & . \;.\; . & e^{-j2\omega_0 \cdot (L-1)} \\ . & . & . \;.\; . & . \\ . & . & . \;.\; . & . \\ 1 & e^{-j(L-1)\omega_0 \cdot 1} & . \;.\; . & e^{-j(L-1)\omega_0 \cdot (L-1)} \end{bmatrix} \tag{2}$$

where $\omega_0 \equiv 2\pi/L$ is the frequency spacing between adjacent bins. $\mathbf{F}$ maps signal $\mathbf{s}$ of length L in the time domain to a spectrum of L bins in the frequency domain. It is straightforward to verify that $\mathbf{F}$ is essentially unitary:

$$\mathbf{F}^\dagger \mathbf{F} = L\mathbf{I}, \tag{3}$$

or equivalently, $\mathbf{F}^{-1} = \frac{1}{L}\mathbf{F}^\dagger$ — this gives a way to calculate inverse DFT:

THEOREM (Inverse DFT):

The length-L inverse DFT can be represented by the following $L \times L$ matrix,

$$\mathbf{F}^{-1} = \frac{1}{L}\begin{bmatrix} 1 & 1 & . \; . \; . & 1 \\ 1 & e^{j\omega_0 \cdot 1} & . \; . \; . & e^{j\omega_0 \cdot (L-1)} \\ 1 & e^{j2\omega_0 \cdot 1} & . \; . \; . & e^{j2\omega_0 \cdot (L-1)} \\ \cdot & \cdot & . \; . \; . & \cdot \\ \cdot & \cdot & . \; . \; . & \cdot \\ 1 & e^{j(L-1)\omega_0 \cdot 1} & . \; . \; . & e^{j(L-1)\omega_0 \cdot (L-1)} \end{bmatrix}. \tag{4}$$

$\mathbf{F}^{-1}$ maps a spectrum $\mathbf{Fs}$ in the frequency domain back to a signal $\mathbf{s}$ in the time domain.

Using the matrix notations, Parseval's theorem (DFT-version) can be derived as a result of Eq. (3):

THEOREM (Parseval's):

Let $\mathbf{s}$ be an arbitrary length-L signal, then its energy in the time-domain is the same as in the frequency domain; that is,

$$\mathbf{s}^{\dagger}\mathbf{s} = \frac{1}{L}(\mathbf{Fs})^{\dagger}(\mathbf{Fs}). \tag{5}$$

1.3 The fast Fourier transform (FFT)

The following paragraph obtained from wikipedia[1] explains what the FFT is and its importance in various fields.

A DFT decomposes a sequence of values into components of different frequencies. This operation is useful in many fields but computing it directly from the definition is often too slow to be practical. An FFT is a way to compute the same result more quickly: computing a DFT of N points in the naïve way, using the definition, takes $O(N^2)$ arithmetical operations, while an FFT can compute the same result in only $O(N \log N)$ operations. The difference in speed can be substantial, especially for long data sets where N may be in the thousands or millions — in practice, the computation time can be reduced by several orders of magnitude in such cases... This huge improvement made many DFT-based algorithms practical; FFTs are of great importance to a wide variety of applications, from digital signal processing and solving partial differential equations to algorithms for quick

[1] copied from http://en.wikipedia.org/wiki/Fast_ Fourier_ transform as of October 4, 2010.

multiplication of large integers.

For this course, we consider FFT as a fairly mature technique that can simply be used without knowing its details. More interested readers can search for the monumental paper by Cooley, James W., and John W. Tukey (1965). "An algorithm for the machine calculation of complex Fourier series," published in *Math. Comput.* 19: 297–301.

2 Short-time Fourier transform

To calculate short-time Fourier transforms (STFT) of a signal involves the following steps: (i) taking a block of the signal, (ii) optionally, multiplying the block by a window function, and then (iii) calculating the DTFT. These three steps combined are described by the following equation,

$$X(\omega, n_0) = \sum_{n=-N}^{N-1} w[n]x[n+n_0]\exp(-j\omega n), \tag{6}$$

where $(x[n_0-N], ..., x[n_0+N-1])^T$ is a block of length $2N$, $w[n]$ is a window function that helps shaping the spectrum, and X denotes its STFT.

Note that X is a function of both frequency ω and time n_0. What happens here is that we slide the window in time to look at a block of signal centering around a particular time n_0, and the spectral components of that block are calculated as a function of frequency.

2.1 Application: the spectrogram

In practice, we often make the window "hop" h samples in time and calculate the STFT accordingly. The resulting STFTs can be shown as a sequence of spectra like a movie, so we get a quick idea how spectral components vary in time.

Alternatively, we can convert the movie into a still image. This is called the *spectrogram* and an example is shown in Fig. 1. Brightness increases as a function of sound intensity. In this spectrogram, we can inspect with adequate frequency resolution where energy is concentrated.

2.2 Time-frequency resolution

When choosing the length of block, be aware of the tradeoff between time and frequency resolution:

- A longer window gives higher frequency resolution while sacrificing the ability to follow the signal's changes in time.
- A shorter window can capture the transient more accurately, but the frequency resolution is lost.

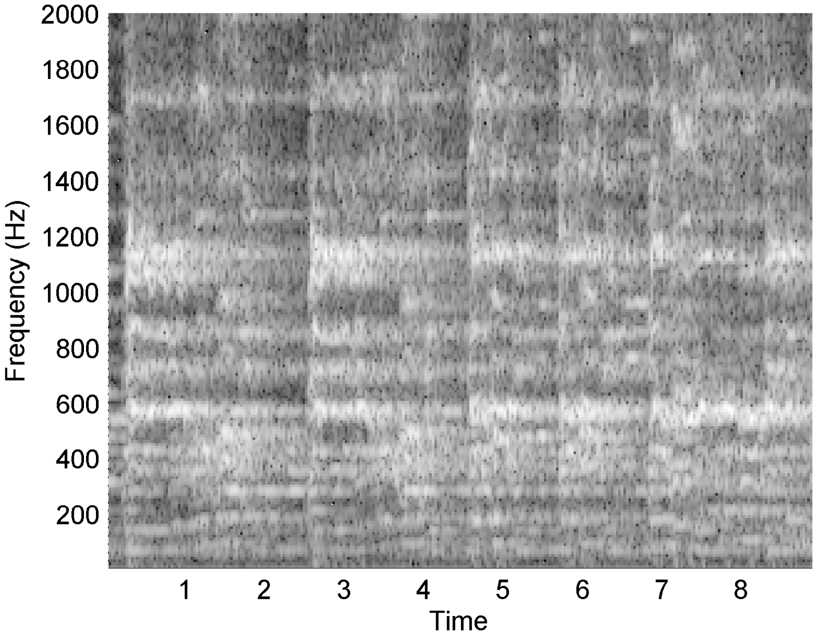

Figure 1: Example of a spectrogram. The signal is MATLAB's default Hallelujah chorus sampled at 8192 Hz. The window type is Hann, and the window length is 512.

For instance, in Fig. 2, a spectrogram is computed for the same signal as in Fig. 1 but the window length is reduced from 512 to 64. We can see that the spectral resolution is lost while temporal resolution is overly high.

3 The cyclic convolution theorems

Roughly speaking, it is still true with DFT that multiplication in one domain (time or frequency) is equivalent to convolution in the other domain. However, because of the finite length of transformation, the convolution needs to be calculated in a wrapped-around manner. This is called the *cyclic* convolution and the corresponding theorems are called the cyclic convolution theorems.

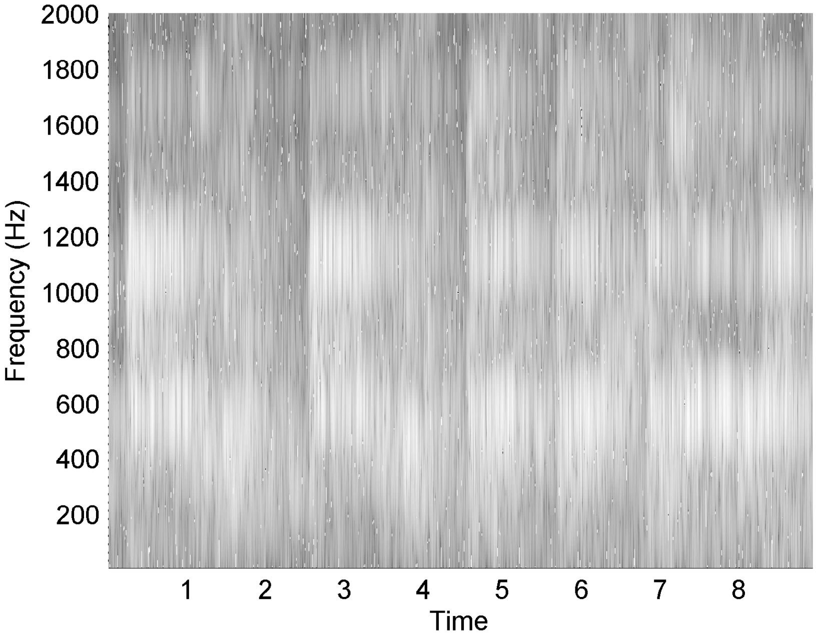

Figure 2: Another spectrogram of the same signal as in Fig. 1, but the window is much shorter ($L = 64$).

THEOREM (Cyclic Convolution):

Let $\mathbf{x} \leftrightarrow \mathbf{X}$ and $\mathbf{y} \leftrightarrow \mathbf{Y}$ be DFT-pairs, and the product spectrum $\mathbf{Z}$ is $\mathbf{X}$ multiplied by $\mathbf{Y}$ at every frequency, i.e., $Z_k = X_k Y_k$, for $k = 0, 1, ..., L-1$. Then, the inverse DFT of $\mathbf{Z}$ is the cyclic convolution of $\mathbf{x}$ and $\mathbf{y}$; that is,

$$z[n] = \sum_{l=0}^{L-1} x[l] \cdot y[n - l|L], \; n = 0, 1, ..., L-1.$$

Proof:

$$\begin{aligned} Z_k = X_k Y_k &= \sum_{l=0}^{L-1} \sum_{m=0}^{L-1} x[l] y[m] e^{-j\omega_k l} e^{-j\omega_k m} \\ \therefore z[n] &= \frac{1}{L} \sum_{k=0}^{L-1} \left(\sum_{l=0}^{L-1} \sum_{m=0}^{L-1} x[l] y[m] e^{-j\omega_k l} e^{-j\omega_k m} \right) e^{j\omega_k n} \\ &= \frac{1}{L} \sum_{l=0}^{L-1} \sum_{m=0}^{L-1} x[l] y[m] \Big(\sum_{k=0}^{L-1} e^{j\omega_k (n-l-m)} \Big), \end{aligned}$$

where $\omega_k = k \cdot (2\pi/L)$. Note that $\sum_{k=0}^{L-1} e^{j\omega_k (n-l-m)} = 0$ unless $m = n - l$ or $m = L + n - l$.

Continuing from above, we have

$$z[n] = \frac{1}{L}\sum_{l=0}^{L-1}\sum_{m=0}^{L-1} x[l]\cdot y[m](L\delta_{m,n-l|L}) = \sum_{l=0}^{L-1} x(l)y(n-l|L). \blacksquare$$

3.1 Block-wise FIR filtering

We've mentioned last time that the convolution theorem is useful for fast computation of FIR filtering; we first compute the DFT of two signals via FFT, multiply their spectra, and then use the inverse FFT to get back to the time domain. However, in musical applications, we often do not want to wait for the entire signal to end before calculating convolutions. It is usually necessary to compute convolution in a block-by-block manner. In this case, note that multiplication in the frequency domain no longer corresponds exactly to convolution in the time domain; it corresponds to the *cyclic* convolution. In our next lecture, we will explain that the exact convolution can still be calculated via FFT as long as zeros are padded around a signal before taking its DFT.

3.2 Time-frequency dualities

The dual of the cyclic convolution theorem is also true: if two block of signals are multiplied in the time domain, then the resulting spectrum is the cyclic convolution of two spectra. We denote this fact as the following,

$$x[n]y[n] \leftrightarrow \frac{1}{2\pi}X(\omega)\otimes Y(\omega), \tag{7}$$

where the symbol $\otimes$ denotes the cyclic convolution.

Exercises

1. Conceptually, spectral interpolation by zero-padding in time is similar to temporal interpolation by band-limited filtering in frequency. Discuss these two concepts in terms of time-frequency dualities.

2. Compare Eq. 5 to Parseval's theorem of DTFT. In what senses are they similar?

3. To visualize the spectrogram of a signal, discuss what range of window lengths (how many ms?) would be good for audio applications.

4. Derive the cyclic convolution theorem in the frequency domain (Eq. 7).