EE20N: Structure and Interpretation of Systems and Signals **Fall 2013**

Lecture 17: October 29

Lecturer: Prof. Kannan Ramchandran *Scribe: TJ Tsai*

17.1 Announcements

- Midterm 2 next Thursday in class – one cheat sheet (like midterm 1)
- No homework next week

17.2 Agenda

- DFT Applications
- DFT Examples
- DFT Perspectives: matrix view & sampling
- Clarification on Guitar lab

17.3 Recap

Recall that the equations for the DFT are:

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{ik\omega_0 n}, n = 0, 1, ..., N-1$$

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-ik\omega_0 n}, k = 0, 1, ..., N-1$$

where N is the length of the sequence and $\omega_0 = \frac{2\pi}{N}$.

	Periodic (in time)	Aperiodic (in time)
Continuous Time	CTFS	CTFT
Discrete Time	DTFS/DFT	DTFT

Figure 17.1: Relationship between various transforms.

17.4 DFT Applications

1. Spectral Analysis. Any applications that involves doing an analysis of the spectrum of a signal will use the DFT. These include the fields of speech, audio, video, images, radar, finding Russian submarines, etc.

2. Frequency Response of LTI Systems. Given the impulse response $h[n]$ of an LTI system, we can calculate the frequency response of the system, $H(e^{i\omega})$, by computing the DTFT of $h[n]$. However, since the DTFT of a signal requires doing a computation over an infinite number of samples, this is not feasible. In practice, we approximate the DTFT of the impulse response using the computer friendly DFT.

3. Convolution using the frequency domain. Suppose that we would like to determine the output of a system to a given input, where the input is represented as a time-domain signal. We could do this directly by computing a time-domain convolution. However, in practice this calculation is done in the frequency domain using the DFT in order to avoid doing a computationally expensive time-domain convolution. Using the DFT in this manner requires a new kind of convolution called circular convolution, which we will talk about soon. By exploiting the structure in the DFT equations, we can derive a very efficient algorithm to compute the DFT. This algorithm is referred to as the Fast Fourier Transform (FFT).

17.5 DFT Examples

Example 1: Compute the DFT of the signal

$$x[n] = \begin{cases} 1, & n = 0 \\ 0, & n = 0, 1, ..., N-1 \end{cases}$$

You should work out this example to confirm that the DFT is

$$X[k] = 1, \; k = 0, 1, ..., N-1$$

See figure 17.2.

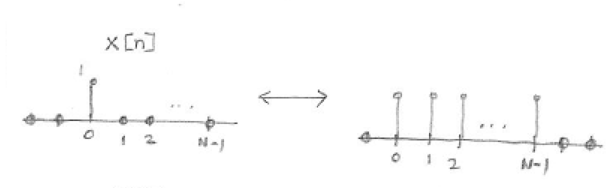

Figure 17.2: DFT transform pair: example 1

Example 2: Compute the DFT of the signal

$$x[n] = 1, \; n = 0, 1, ..., N-1$$

Let's work out the DFT. For $k = 0, 1, ..., N-1$,

$$
\begin{aligned}
X[k] &= \sum_{n=0}^{N-1} x[n] e^{-i\frac{2\pi}{N}nk} \\
&= \sum_{n=0}^{N-1} e^{-i\frac{2\pi}{N}nk} \\
&= \begin{cases} N, & k = 0 \\ \frac{1-e^{-i\frac{2\pi}{N}kN}}{1-e^{-i\frac{2\pi}{N}k}}, & k = 1, ..., N-1 \end{cases} \\
&= \begin{cases} N, & k = 0 \\ 0, & k = 1, ..., N-1 \end{cases}
\end{aligned}
$$

The third line is derived using the geometric series formula, and the fourth line is derived by noting that the numerator is always 0. See figure 17.3. Observe the duality between the time and frequency "spread". When a signal is skinny in the time-domain, it is fat in the frequency domain. Likewise, when a signal is fat in the time-domain, it is skinny in the frequency domain.

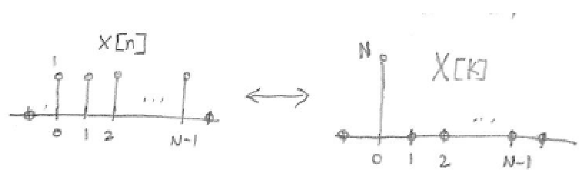

Figure 17.3: DFT transform pair: example 2

17.6 DFT Perspectives

17.6.1 Matrix View

We can look at the DFT from a matrix perspective. Let's express $x[n]$ as a vector containing the entries $x[0], x[1], ..., x[N-1]$ and $X[k]$ as a vector containing the entries $X[0], X[1], ..., X[N-1]$. Example 1 from the previous section demonstrated that:

$$
\underline{x} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \xrightarrow{DFT} \underline{X} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}
$$

More generally, we can express the relationship between $\underline{x}$ and $\underline{X}$ as:

$$
\underline{X} = F\underline{x}
$$

where $\underline{X}$ and $\underline{x}$ are N-length vectors and F is an $N \times N$ matrix. Let's look at a simple case when $N = 2$. In this case, the matrix form of the DFT equations are:

$$\begin{pmatrix} X[0] \\ X[1] \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} x[0] \\ x[1] \end{pmatrix}$$

You should work through the equations to verify that this is correct. Let's consider as slightly more complicated case when $N = 4$. To reduce the clutter, we will let $W = e^{-i\frac{2\pi}{N}}$. Then the 4×4 matrix F is:

$$F = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & W & W^2 & W^3 \\ 1 & W^2 & W^4 & W^6 \\ 1 & W^3 & W^6 & W^9 \end{pmatrix}$$

Note that the "naive" computational complexity of the matrix form of the DFT is $O(N^2)$. However, it turns out that there is a fast algorithm called the Fast Fourier Transform (FFT) that computes an N-length DFT in $O(N \log N)$. Note that the savings factor is $\frac{N}{\log N}$. For large N, this is a huge savings. For example, when $N = 1,000,000$, the savings factor is about $50,000$!

17.6.2 Sampling

We will get to this next time.

17.7 Clarification on Guitar Lab

We wanted to clarify a concept that has been confusing to a lot of students in the guitar lab. For those who did the guitar lab last week, you should recognize this equation:

$$\Omega_d = \frac{2\pi f_c}{f_s}$$

We would like to explain and clarify what this equation means. But let's first take a step back and give some background knowledge that will help us to understand this equation. Consider the system shown in figure 17.4, which is called a sampler. A sampler takes a continuous-time signal as input and returns a discrete-time signal as output.

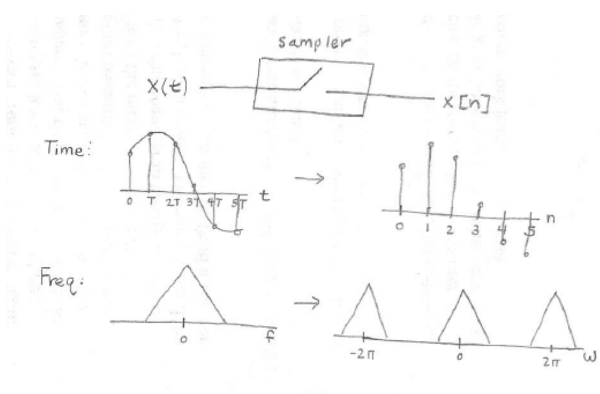

Figure 17.4: Picture book explanation of what a sampler does

We will not delve into mathematical equations, but instead give you a picture book explanation of what a sampler does. Just like with LTI systems, we can describe what the system does from two different perspectives: the time-domain perspective and the frequency-domain perspective. Let's first consider the time-domain perspective. As you can see from the figure above, the sampler does two things in the time-domain: (1) it samples the continuous-time signal every T seconds, and (2) it reindexes time. Now let's consider the frequency-domain perspective. The sampler does two things in the frequency-domain: (1) it scales the axes (both the magnitude axis as well as the frequency axis), and (2) it repeats the scaled version every 2π. Note that if the spectrum of the continuous-time signal (represented as a triangle) extends over too large of a frequency range, the spectral "islands" of $X_d(\omega)$ at $0, 2\pi, 4\pi, \ldots$ may overlap. This phenomenon is known as aliasing, and it is generally a bad thing since information is mixed up or lost.

Now we're ready to talk about the equation of interest:

$$\Omega_d = \frac{2\pi f_c}{f_s}$$

This equation tells us how sampling scales the frequency axis. Note that the variable f_s is describing the system (the sampler), not the signals. No matter what the input is, the sampler will always take $f_s = \frac{1}{T_s}$ samples per second. The other two variables f_c and Ω_d represent a frequency component of the continuous-time signal $x(t)$ and the discrete-time signal $x[n]$, respectively.

To tie everything together, let's look at a simple, concrete example. Consider a sampler that samples at 4 Hz, and let's assume the input is a sinusoid with frequency 1 Hz. The signal and it's sampled counterpart is shown in figure 17.5. The continuous-time signal is

$$x(t) = \sin(2\pi f_c t)$$

and the corresponding sampled signal is

$$\begin{aligned} x[n] &= x(nT_s) \\ &= \sin(2\pi f_c n T_s) \\ &= \sin(2\pi f_c n \frac{1}{f_s}) \\ &= \sin(\frac{\pi}{2} n) \end{aligned}$$

Here we see that a continuous-time frequency of 1 Hz maps to a discrete-time frequency of $\frac{\pi}{2}$ radians per second. We could have calculated this directly by simply using the equation above with $f_s = 4$ and $f_c = 1$.

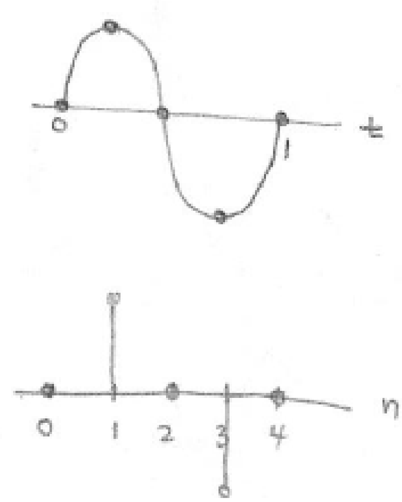

Figure 17.5: Sampling: simple example