What is 6.003?

What is a signal?
Abstractly, a signal is a function that conveys information.

Signal processing is about extracting meaningful information from signals, and/or manipulating information to produce new signals.

What is a transform?
Provide multiple views/perspectives on a signal.
Some information more clearly visible (and/or more easily manipulable) from one perspective than another.

Why Fourier?

Sinusoids are nice!
- prevalent in nature
- relevant to human perception
- mathematically convenient (particularly as complex exponentials)

Example: what are the following sounds, and how do they differ?
So Far

So far, we have talked about four transforms, useful for analyzing different kinds of signals:

<table>
<thead>
<tr>
<th>Transform</th>
<th>Time Domain</th>
<th>Frequency Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTFS</td>
<td>continuous, periodic</td>
<td>discrete, aperiodic</td>
</tr>
<tr>
<td>DTFS</td>
<td>discrete, periodic</td>
<td>discrete, periodic</td>
</tr>
<tr>
<td>CTFT</td>
<td>continuous, aperiodic</td>
<td>continuous, aperiodic</td>
</tr>
<tr>
<td>DTFT</td>
<td>discrete, aperiodic</td>
<td>continuous, periodic</td>
</tr>
</tbody>
</table>

Four transforms already; why introduce another?

Consider DTFT

DTFT: DT signal \(x[n]\), CT spectrum \(X(\Omega)\)

\[
x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega)e^{j\Omega n}d\Omega
\]

\[
X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}
\]

Easiest to appreciate when \(x[n]\) and \(X(\Omega)\) are defined as mathematical expressions.

However, it would be great to define the spectrum of a signal \(x[n]\) even when \(x[n]\) cannot be easily defined by a mathematical expression.

Indeed, it would be great to be able to write a computer program that could efficiently calculate the spectrum of an arbitrary signal from samples of that signal.

DFT

DFT (Discrete Fourier Transform) is discrete in both domains. Computationally feasible (opens doors to analyzing complicated signals).

Most modern signal processing is based on the DFT, and we’ll use the DFT almost exclusively moving forward in 6.003.
Why Bother with the Others?

In part, because they inform our understanding / interpretation of the results of the DFT.

Our common goal with other science / engineering endeavors is to

- model some aspect of the world
- analyze the model, and
- interpret results to gain better understanding.

![Diagram](human) → Model (computer) → Result (human)

World (human) → New Understanding (human)

The computer is useless if we can’t interpret the results!

Toward the DFT

Starting with the DTFT analysis equation:

\[ X(\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n} \]

If a goal is computational feasibility, two things stand in the way:

- infinite sum
- continuous function of frequency

Solutions:

- only consider a finite number of samples in time, and
- only consider a finite number of frequencies.

Toward the DFT

Starting with the DTFT analysis equation:

\[ X(\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n} \]

- only consider \( N \) samples
- take \( N \) uniformly spaced frequencies from the range \( 0 \leq \Omega \leq 2\pi \)

The DFT:

\[ X[k] = \frac{1}{N} X \left( \frac{2\pi k}{N} \right) = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi k n}{N}} \]
DFT: Definition

\[ X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi kn}{N}} \]

\[ x[n] = \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi kn}{N}} \]

Very similar in form to the other transforms we’ve seen (particularly the DTFS).

DFT: Comparison to Other Fourier Representations

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Synthesis</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DFT</strong></td>
<td>[ X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi kn}{N}} ]</td>
</tr>
<tr>
<td><strong>DTFS</strong></td>
<td>[ X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi kn}{N}} ]</td>
</tr>
<tr>
<td><strong>DTFT</strong></td>
<td>[ X(\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n} ]</td>
</tr>
</tbody>
</table>

DTFS: \( x[n] \) is presumed to be periodic in \( N \)
DTFT: \( x[n] \) is arbitrary
DFT: only a portion of an arbitrary \( x[n] \) is considered

DFT and DTFS

If a signal is periodic in the DFT analysis period \( N \), then the DFT coefficients are equal to the DTFS coefficients.

Consider \( x_1[n] = \cos \left( \frac{2\pi n}{64} \right) \). When analyzed with \( N = 64 \), the DFT coefficients are:

\[ X_1[k] = \frac{1}{N} \sum_{n=0}^{N-1} x_1[n] e^{-j\frac{2\pi kn}{N}} = \frac{1}{2} \delta[k - 1] + \frac{1}{2} \delta[k + 1] \]
**DFT and DTFS**

If a signal is **not** periodic in the DFT analysis period $N$, then there are no DTFS coefficients to compare.

Consider $x_2[n] = \cos\left(\frac{3\pi}{64} n\right)$. When analyzed with $N = 64$, the DFT coefficients are:

$$X_2[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi k n}{N}}$$

Even though $x_2[n]$ contains a single frequency $\Omega = \frac{3\pi}{64}$, there are large components at every component $k$.

**DFT and DTFS**

Although $x_2[n] = \cos\left(\frac{3\pi}{64} n\right)$ was not periodic in $N = 64$, we can define a signal $x_3[n]$ that is equal to $x_2[n]$ for $0 \leq n < 64$ and periodic in $N = 64$.

DFT coefficients of $x_2[n]$ equal the DTFS coefficients of $x_3[n]$. The large number of non-zero coefficients are necessary to produce the step discontinuity at $n = 64$.

**DFT: Relation to DTFT**

$X[n]$ is the DFT of $x[n]$.

$$X_N(\frac{2\pi k}{N})$$

sample: $\Omega \rightarrow \frac{2\pi k}{N}$

scale: $1/N$
Two Ways to Think About DFT

We can think about the DFT in a few different ways:

1. The DFT is equal to the DTFS of the periodic extension of the first $N$ samples of a signal.

2. The DFT is equal to scaled samples of the DTFT of a “windowed” version of the original signal.

These views are equivalent, but they highlight different phenomena.

DFT: Properties

Very similar in form to the other transforms we've seen (particularly the DTFS), so lots of the usual properties hold (with some slight tweaks).

<table>
<thead>
<tr>
<th>Property</th>
<th>$y[n]$</th>
<th>$Y[k]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linearity</td>
<td>$Ax_1[n] + Bx_2[n]$</td>
<td>$AX_1[k] + BX_2[k]$</td>
</tr>
<tr>
<td>Time Reversal</td>
<td>$x[N - n]$</td>
<td>$X[-k]$</td>
</tr>
<tr>
<td>Time Shift</td>
<td>$x[(n - n_0) \mod N]$</td>
<td>$e^{-j\frac{2\pi k n_0}{N}} X[k]$</td>
</tr>
<tr>
<td>Frequency Shift</td>
<td>$e^{j\frac{2\pi k n}{N}} x[n]$</td>
<td>$X[k - k_0]$</td>
</tr>
<tr>
<td>Conjugation</td>
<td>$x^*[n]$</td>
<td>$X^*[-k]$</td>
</tr>
</tbody>
</table>

Check Yourself!

Consider a waveform containing a single, pure sinusoid. This waveform was recorded with a sampling rate of 8kHz, and we have 60 samples of the waveform. Computing the DFT magnitudes, we find:

What note is being played?
How accurately can we tell?
**Frequency Resolution**

We only have $N$ distinct samples of the DTFT.

| $-N/2$ | 0 | $N/2$ | $k$ (cycles/$N$) |
| $-\pi$ | 0 | $\pi$ | $\Omega$ (rad/sample) |
| $-f_s/2$ | 0 | $f_s/2$ | $f$ (cycles/sec) |

We’re uniformly breaking up a range of $2\pi$ into $N$ discrete samples: the spacing between samples is $\frac{2\pi}{N}$. The $k^{th}$ coefficient is associated with $\Omega = 2\pi k / N$.

In Hz, the spacing between samples is $\frac{f_s}{N}$. Thus, the $k^{th}$ coefficient is associated with a frequency of $f = kf_s / N$.

Fundamental trade-off: increasing frequency resolution necessarily requires considering more samples of the signal (i.e., increasing $N$).

---

**Check Yourself!**

For the previous example (pure sinusoid, $f_s = 8\text{kHz}$), how many samples do we need to consider in order to be able to determine the frequency of the tone to within 1Hz? Within 0.1Hz?

---

**Check Yourself!**

For a portion of the Chopin song containing only one chord, 250332 samples long, and recorded with a sampling rate of 48kHz, the DFT magnitudes look like:

Peaks in magnitude around $k = 1021, 1282, 1715, 2037, 2576, 3062, \ldots$

What are the frequencies (in Hz) of the notes being played? What chord does this correspond to?
Summary

Today: introduced a new transform: the DFT.
- closely related to both DTFS and DTFT
- enables computational analysis of complicated signals

Recitation: practice with DFT, analyzing sounds

PSet: practice with DFT, implement FFT