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 in signals 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.

2D signals

So far, our signals have been a function of time: $f(t)$, $f[n]$

Now, start to consider functions of space: $f(x,y)$, $f[r,c]$

Our goal is still the same: extract meaningful information from a signal, or manipulate information in a signal. Fourier representations will still be useful!
Fourier Representations

One dimensional CTFT:
\[ F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt \]
\[ f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{j\omega t} d\omega \]

Two dimensional CTFT:
\[ F(\omega_x, \omega_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j(\omega_x x + \omega_y y)} dx dy \]
\[ f(x,y) = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(\omega_x, \omega_y) e^{j(\omega_x x + \omega_y y)} d\omega_x d\omega_y \]

\(x \text{ and } y\) are continuous spatial variables (units: cm, m, etc.)
\(\omega_x \text{ and } \omega_y\) are spatial frequencies (units: radians / length)

Python Representation

We will represent our images (both in space and in frequency) as NumPy arrays. NumPy arrays implement a number of common operations very efficiently, compared to the techniques we have used in the past in 6.003. Some of the key differences from lists are:

- NumPy arrays are homogenous (all objects in a numpy array are the same type)
- Operators like +, -, abs, x.imag, x.real work element-wise on numpy arrays, and much more efficiently than writing a loop or list comprehension. For example, \(x + y\) computes the element-wise sum of the arrays \(x\) and \(y\), and produces a new NumPy array. \(x + 2\) produces a new Numpy array, adding 2 to each element of \(x\).
- 2-D NumPy arrays can be indexed using tuples, specifying first a row and then a column. For example, to grab the element at row 3 and column 4 from an array \(x\), we can use the following notation: \(x[3, 4]\). NumPy arrays also support negative indices.

Python Representation

We provide methods in a file called image_utils.py for saving/loading PNG images:

- \(\text{png_read(filename)}\) loads an image from a file into a NumPy array
- \(\text{png_write(array, filename)}\) saves image data to a file

These arrays are indexed in row, column order, and the values are brightnesses, usually in the range \([0, 1]\). In this representation, index \(0, 0\) corresponds to row 0, column 0.

We also provide functions for computing 2D DFTs: \(\text{fft2}\) and \(\text{ifft2}\)
In the frequency domain, index \(0, 0\) into a numpy array corresponds to \(k_x = 0, k_y = 0\) (the DC component).
Fourier Representations

One dimensional DTFT:

\[ F(\Omega) = \sum_{n=-\infty}^{\infty} f[n] e^{-jn\Omega} \]

\[ f[n] = \frac{1}{2\pi} \int_{2\pi}^{2\pi} F(\Omega) e^{jn\Omega} d\Omega \]

Two dimensional DTFT:

\[ F(\Omega_r, \Omega_c) = \sum_{r=-\infty}^{\infty} \sum_{c=-\infty}^{\infty} f[r,c] e^{-j(\Omega_r r + \Omega_c c)} \]

\[ f[r,c] = \frac{1}{4\pi^2} \int_{2\pi}^{2\pi} \int_{2\pi}^{2\pi} F(\Omega_r, \Omega_c) e^{j(\Omega_r r + \Omega_c c)} d\Omega_r d\Omega_c \]

\( r \) and \( c \) are discrete spatial variables (units: pixels)
\( \Omega_r \) and \( \Omega_c \) are spatial frequencies (units: radians / pixel)

Fourier Representations

One dimensional DFT:

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

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

Two dimensional DFT:

\[ F[k_r, k_c] = \frac{1}{RC} \sum_{r=0}^{R-1} \sum_{c=0}^{C-1} f[r,c] e^{-j\left(\frac{2\pi r}{R} k_r + \frac{2\pi c}{C} k_c\right)} \]

\[ f[r,c] = \sum_{k_r=0}^{R-1} \sum_{k_c=0}^{C-1} F[k_r, k_c] e^{j\left(\frac{2\pi r}{R} k_r + \frac{2\pi c}{C} k_c\right)} \]
Check Yourself

The 2D DFT basis functions have the form
\[ \phi_{k_r,k_c}[r,c] = e^{-j \frac{2\pi k_r}{R} r} e^{-j \frac{2\pi k_c}{C} c} \]

Which (if any) of the following images show the real part of one of the basis functions \( \phi_{k_r,k_c}[r,c] \)?

A B C D

What values of \( k_r \) and \( k_c \) correspond to each basis function?

Fourier Transform Pairs

In 1D, we found that it was useful to know how the transforms of simple shapes looked (for example \( \delta \rightarrow \text{constant} \)), in part because it was often possible to use that understanding to simplify thinking about bigger problems.

The same will be true in 2D! The rest of today: 2D Fourier analysis of simple shapes.

2D Discrete Fourier Transform

Example: Find the DFT of a 2D unit sample.
\[ f_0[r,c] = \delta[r] \delta[c] = \begin{cases} 1 & \text{if } r = 0 \text{ and } c = 0 \\ 0 & \text{otherwise} \end{cases} \]
\[ F_0[k_r,k_c] = \frac{1}{RC} \sum_{r=0}^{R-1} \sum_{c=0}^{C-1} \delta[r] \delta[c] e^{-j\left( \frac{2\pi k_r}{R} r + \frac{2\pi k_c}{C} c \right)} \]
\[ = \frac{1}{RC} \sum_{r=0}^{R-1} \sum_{c=0}^{C-1} e^{-j\left( \frac{2\pi k_r}{R} 0 + \frac{2\pi k_c}{C} 0 \right)} \]
\[ = \frac{1}{RC} \]
\[ \delta[r] \delta[c] \xleftarrow{\text{dft}} \frac{1}{RC} \]
2D Discrete Fourier Transform

Alternatively, implement a 2D DFT as a sequence of 1D DFTs.

$$F[k_r, k_c] = \frac{1}{RC} \sum_{r=0}^{R-1} \sum_{c=0}^{C-1} f[r, c] e^{-j \frac{2\pi kr \cdot r}{R} + \frac{2\pi kc \cdot c}{C}}$$

Could just as well start with columns and then do rows.
2D Discrete Fourier Transform

Example: Find the DFT of a horizontal line.

Example: Find the DFT of a constant.

Example: Find the DFT of a shifted 2D unit sample.
Translating (Shifting) an Image

Effect of image translation (shifting) on its Fourier transform.

Assume that $f_0[r,c] \Leftrightarrow F_0[k_r, k_c]$.

Find the 2D DFT of $f_1[r,c] = f_0[r-r_0, c-c_0]$

$$F_1[k_r, k_c] = \sum_r \sum_c f_1[r,c] e^{-j \frac{2\pi k_r r}{R}} e^{-j \frac{2\pi k_c c}{C}}$$

Let $l_r = r - r_0$ and $l_c = c - c_0$. Then

$$F_1[k_r, k_c] = e^{-j \frac{2\pi k_r r_0}{R}} e^{-j \frac{2\pi k_c c_0}{C}} \sum_{l_r} \sum_{l_c} f_0[l_r,l_c] e^{-j \frac{2\pi k_r l_r}{R}} e^{-j \frac{2\pi k_c l_c}{C}}$$

2D Discrete Fourier Transform

Example: Find the DFT of a horizontal cosine.

$f[r,c]$
2D Discrete Fourier Transform

Example: Find the DFT of a horizontal cosine.

$\mathbf{f[r,c]}$

Magnitude

Angle

Example: Find the DFT of a square.

$\mathbf{f[r,c]}$

Magnitude

Angle
Summary

Image Processing
- Introduction to 2D signal processing
- 2D Fourier Representations

Recitation
- More simple shapes,
- Construction Project