Spectral Cross-correlation

So far, we have done cross-correlation in the spatial domain. Cross-correlation can also be done in the spectral domain by completing a Fourier transform, multiplying signals, and doing an inverse Fourier transform. We're not going to go through the derivation of a Fourier transform but there are some great videos on the topic linked at the end of this document.

Convolution using FFTs

It can be shown that the discrete convolution of signal u and v as defined by,

$$(u * v)(\tau) = \sum_{m=1}^{N} u(m)v(\tau - m)$$

can also be expressed in terms of the Fourier transform

$$(u * v)(\tau) = \mathcal{F}^{-1}\{\mathcal{F}\{u\} \cdot \mathcal{F}\{v\}\}$$

where $\mathcal{F}\{u\}$ is the Fourier transform of u, $\mathcal{F}\{v\}$ is the Fourier transform of v, and $\mathcal{F}^{-1}$ is the inverse Fourier transform. You can check this for yourself very easily

Correlation using FFTs

So cross-correlation can be calculated through summation of a product

$$(u \star v)(\tau) = \sum_{m=1}^{N} u^*(m)v(m + \tau)$$

or using FFTs,

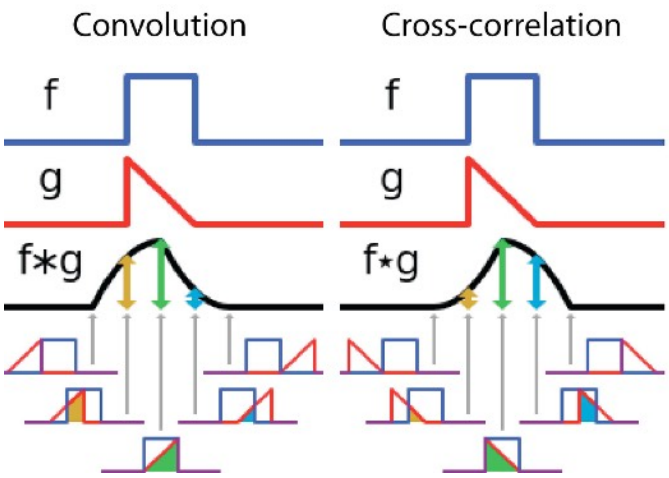
$$(u \star v)(\tau) = \mathcal{F}^{-1}\{(\mathcal{F}\{u\})^* \cdot \mathcal{F}\{v\}\}$$

Where * refers to the complex conjugate.

You will see that convolution, as well as correlation are being mentioned above. Convolution is well explored in signal analysis, and cross-correlation is what we have been discussing thus far.

Correlation is measurement of the similarity between two signals/sequences.
Convolution is measurement of effect of one signal on the other signal.

Turns out, the output of a **convolution** between two sequences is the same as the correlation when one of the signals is reversed.



This allows us to use convolution functions for correlation. We are specifically interested in this function:

$$(u \star v)(\tau) = \mathcal{F}^{-1}\left\{(\mathcal{F}\{u\})^* \cdot \mathcal{F}\{v\}\right\}$$

This implies that we can do correlation following these steps:

1. Fourier transform each input signal (2D matrix is the 'signal')
2. For one of the Fourier transformed inputs, take the complex conjugate
3. Multiply
4. Inverse Fourier transform the result

This will provide an output which shows their similarity when they are shifted relative to one another.

The best *lag* corresponds to the location with highest value in the returned output.

Question 1: write a function which accepts an input matrix, and calculates the fourier transform (use numpy fourier transform functions – np.fft for example)

Question 2: write a function which accepts a fourier space input, and returns the complex conjugate.

Question 3: write a function which accepts two inputs in fourier space and performs elementwise multiplication

Question 4: write a function which accepts an input in fourier space and returns the inverse fourier transform (use numpy fourier transform functions – np.fft for example)

Question 5: write a function which uses the above functions to perform spectral cross-correlation.

Question 6: visualise your results! Does it all make sense?

https://www.youtube.com/watch?v=spUNpyF58BY&ab_channel=3Blue1Brown

https://www.youtube.com/watch?v=r18Gi8lSkfM&ab_channel=PhysicsVideosbyEugeneKhutoryansky