Convolution Operation

`Input` 2 arrays/vectors representing a time domain signals

`Output` Convolution of 2 vector

$$Conv = x_1 * x_2$$

$$Conv[n] = \sum_{m=0}^{N-1} x_1[m]x_2[(n-m) \mod N]$$

Polynomial is same operation

if we have polynomial P1 of degree d and is represented by array of d+1 coefficients and Polynomial P2 of degree e and is represented by array of e+1 coefficients.

Now if we pad both arrays by 0 till d+e+1. The polynomials will remain same.

Their multiplication will be of degree d+e+1 say M then M can be written as

$$M(n) = \sum_{m=0}^{d+e} P1[m]P2[(n-m) \mod d+e+1]$$

$$M(n) = P1 * P2(\text{with padded 0s})$$

Solving using FFT

From the fact that convolution in time domain is equal to simple multiplication in frequency domain

say we X_1 and X_2 to be frequency domain(DFT of) of x_1 and x_2. Let X_3 be the convolution in frequency domain(DFT) and its domain be x3.

Then,

$$X_3[k] = X_1[k].X_2[k]$$

$$x_3 = IDFT(X3)$$

Since we already discussed very fast algorithm for converting time domain into frequency domain which is fast Fourier transform

So,

$$x_1 * x_2 = IFFT(FFT(x_1).FFT(x2))$$

So multiplication of can also be calculated by same after padding with appropriate no. of zeroes.

Algorithm

1. Pad P1 and P2 with zeroes until their length becomes degree(P1) + degree(P2) +1
2. Call FFT of P1 store in Q1
3. Call FFT of P2 store in Q2

4. For i = 0,..... degree(P1) + degree(P2)
    1. Q3[i] = Q2[i].Q1[i]
5. P3 = InvFFT(Q3)
6. P3 is the answer.

Time complexity analysis

FFT/IFFT takes nlogn time

We did 3 FFT/IFFT operations

So final time complexity = O(nlogn) + O(nlogn) + O(nlogn) + O(n) = O(nlogn)