Fast Fourier Transform

Team *random-theory*
May 17, 2022

1 INTRODUCTION

A Fourier transform is a mathematical transform that decomposes functions depending on space or time into functions depending on spatial frequency or temporal frequency.

The central question that fourier transform is trying to solve is that if we have a signal, can we decompose it into the pure frequencies that make it up? such that adding up all the invidual decompositions gives us the original signal back. Its just like unmixing some colors that have been mixed together or finding the recipe of a given smoothie.

It is one of the most powerful tools in digital signal processing and digital image processing that is used for the the frequency analysis of signals. It is used in a wide range of applications, such as signal interpolation, signal smoothing, image filtering, image reconstruction and image compression. In fact any field of physical science that uses sinusoidal signals, will make use of Fourier series and Fourier transforms.

1.1 FOURIER TRANSFORM OF A FUNCTION

It gives us a unique way of viewing any function as the sum of simple sinusoids. The Fourier Transform of a function $g(t)$ is defined by:

$$G(f) = \int_{-\infty}^{\infty} g(t)\, e^{-2\pi i f t} \tag{1.1}$$

In addition, g can be obtained from G via the inverse Fourier Transform:

$$g(t) = \int_{-\infty}^{\infty} G(F)\, e^{2\pi i f t} \tag{1.2}$$

1.2 Why use Complex Numbers?

In the time domain, we have our horizontal and vertical axis. If we were to place our samples on the x-axis, then the y-axis would show the amplitude of those samples. Now, if we were to represent the samples in the frequency domain, our samples need to conserve two properties, the amplitude, and the phase; the latter is quite important for various applications of the transform such as data filtering. Complex numbers provide a nice way of storing the phase.

2 Discrete Fourier Transform

In Discrete Fourier Transform, we modify the above integral equations to Fourier transform the discrete finite samples. If we take a sample size of N, then there will be exactly N values generated after the transform, thus we can say that DFT maps N complex numbers from one domain, say time, to N complex numbers in another domain, say frequency. Finding the discrete sum of the continuous equations above gives us:

$$G_k = \sum_{n=0}^{N-1} g_n \cdot e^{\frac{-i2\pi}{N} \cdot kn} \tag{2.1}$$

2.1 Fourier Matrix

An N-point DFT is expressed as the multiplication $X = Wx$ where x is the original input signal and W is the N-by-N square DFT matrix.

$$\omega = e^{\frac{-2\pi i}{N}} \tag{2.2}$$

$$x = \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{pmatrix} \tag{2.3}$$

$$W = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \dots & \omega^{(N-1)(N-1)} \end{pmatrix} \tag{2.4}$$

$$DFT(x) = X = x \cdot W \tag{2.5}$$

```python
def compute_dft(x):
    N = len(x)
    X = [0 for _ in range(N)]
    for i in range(N):
        for j in range((N)):
            omega = np.exp(complex(0, -2) * np.pi / N)
            X[i] += x[j] * omega**(i * j)
    return X
```

Listing 1: A python program to compute DFT of a given signal

3 Fast Fourier Transform

The FFT employs a divide-and-conquer strategy to divide the DFT computation into odd and even parts. These parts are very similar to the original DFT, and we only need to perform half the computation. This would convert our O(n^2) algorithm to O(m^2) where m is n/2. As long as m is divisble by 2, we can recursively repeat this till our running time is O($nlogn$). The psuedocode is as follows:

Figure 3.1: Fourier transform psuedocode

```
def FFT(P) :
    # P - [p_0, p_1, ..., p_{n-1}] coeff representation
    n = len(P) # n is a power of 2
    if n == 1:
        return P
    ω = e^{2πi/n}
    P_e, P_o = [p_0, p_2, ..., p_{n-2}], [p_1, p_3, ..., p_{n-1}]
    y_e, y_o = FFT(P_e), FFT(P_o)
    y = [0] * n
    for j in range(n/2):
        y[j] = y_e[j] + ω^j y_o[j]
        y[j + n/2] = y_e[j] − ω^j y_o[j]
    return y
```

```python
def FFT(X):
    N = len(X)
    if N <= 1: return X
    omega = np.exp(complex(0, -2 * np.pi) / N)

    X_even = FFT(np.array(X[0:N:2]))
    X_odd = FFT(np.array(X[1:N:2]))

    X = [0 for _ in range(N)]

    for j in range(0, N // 2):
        X[j] = X_even[j] + omega**j * X_odd[j]
        X[j + N // 2] = X_even[j] - omega**j * X_odd[j]
    return X
```

Listing 2: A python program to compute DFT of a given signal using the fast fourier transform

After recursively dividing the input, P, we only need to run m, or n/2 times. This is what leads to the recurrence of T(n) = 2T(n/2) + O(n). This is only possible because of the symmetric nature of the DFT.

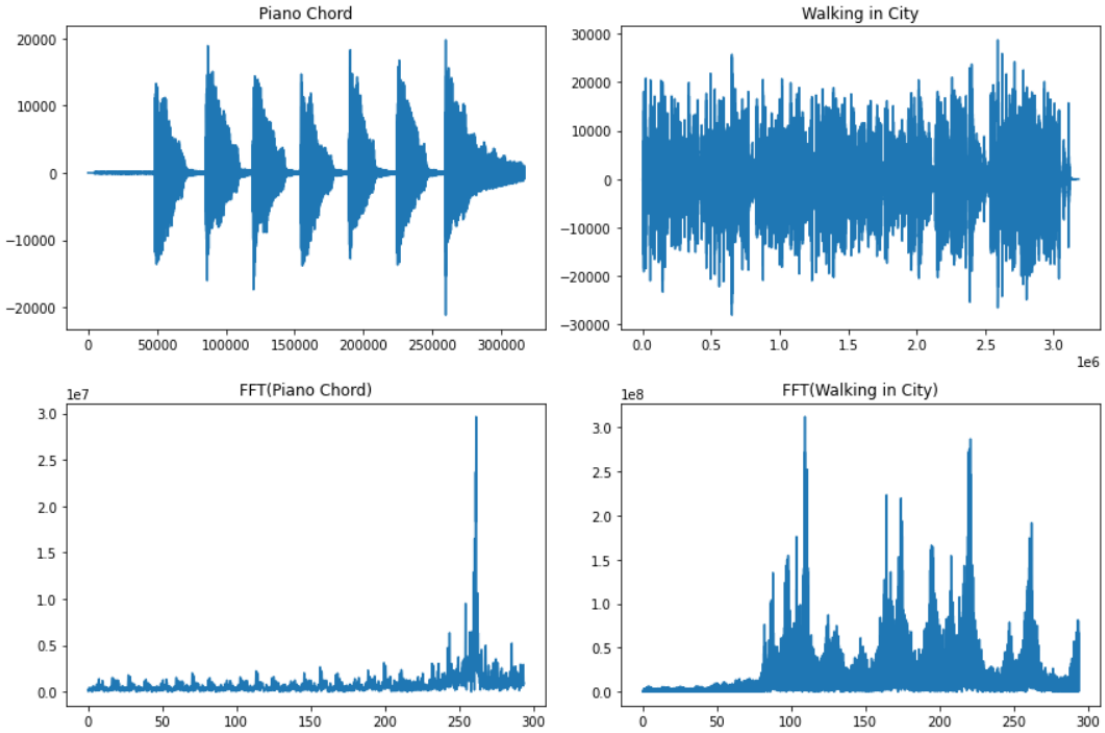
4 Visualization

To show the workings of the FFT, we applied it in one of it's most popular applications: signal processing. We use 2 audio files to show the main functionality the FFT allows.

- Piano chord being played repeatedly.
- Background noise while walking in a city.

Looking at the first graph below, we see the signal for the piano chord being played repeatedly. We can physically see the chord being repeated in this signal. In the second graph, we see the signal for the noises in the city. It is more noisy and there are alot more noises mixed in.

Essentially, the FFT converts our time domain signal into the frequency domain. It unmixes the frequencies mixed in to the final sound to seperate distinct sounds. In signal processing, this would allow us to identify background noises, or noises to isolate. The FFT identifies the piano chord and we can see this as a peak on the graph. For the second FFT graph, there are alot more noises in the final audio track. These different noises are identified and the most frequent ones are displayed as the highest peaks.



5 Theoretical runtime Analysis

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus:

Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies

6 Experimental runtime Analysis

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus:

Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies

7 Proof of FFT supremacy

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus:

Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies

8 Conclusion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus:

Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies