Shah Mahdi Hasan Follow
Apr 15, 2020 · 7 min read · Listen

Save

Breaking down confusions over Fast Fourier Transform (FFT)

Fourier Transform is undoubtedly one of the most valuable weapons you can have in your arsenal to attack a wide range of problems. FFT is literally the bread and butter for many signal processing engineers. At the same time, there are examples of application of FFT in data science realm. For example, time-series data with dense interval (for example: instead of monthly sales data, you might be dealing with hourly sales data) exhibits complex seasonality. Fourier Transform is a handy tool to analyze the seasonalities of those kinds. Unfortunately, in an inexperienced hand, it can lead to disastrous and frustrating conclusions sometimes.

I was surfing through hundreds of discussions and Q&A in stack-exchange, Mathworks and other forums. What I have learned is: people who intend to use Fourier Transform do know why they want to use it. But the common confusion is: **What's next?** How to "post-process" the sequence in hand and get a meaningful and accurate insight? To be more specific, there are mainly following questions:

1. What is Single Side Band (SSB)? Why do we care?
2. What is the relationship between my FFT sequences and physical frequencies? How these two can be correctly aligned?

4. Why there is a normalization factor?

The source of confusions are buried beneath the mathematical fundamentals which built the foundation of Fourier Transform. In this article, I will try to breakdown several common confusions which arise while working with Fourier Transform. I will try to explain the reasoning using as little mathematics as possible. Hopefully this will help you to work with this amazing tool without going through the trouble of picking up the mathematics behind it.

For explanation, I will be using an example code for Fast Fourier Transform (FFT). FFT is essentially a super fast algorithm that computes Discrete Fourier Transform (DFT). The example code is written in MATLAB (or OCTAVE) and it is a quite well known example to the people who are trying to understand Fourier Transform. Similar example can be found in here.

```
clear;
clc;
Fs = 1000;          %sampling rate
T = 1/Fs;           %sampling interval
L = 500;            %Number of time points
t = 0:T:(L-1)*T;    %time vector
N = 1024;           %FFT points
% Define frequencies
f1 = 20;
f2 = 220;
f3 = 138;
% Create the signal
x  = 5 + 12*sin(2*pi*f1*t-0.3) + 5*sin(2*pi*f2*t-0.5) + 4*sin(2*pi*f3*t - 1.5);
figure(1)
plot(t*1000,x)
xlabel('time (in ms)')
ylabel('Amplitude')
title('Original Signal')
% Add some noise
figure(2)
x = awgn(x,5,'measured')
plot(t*1000,x)
xlabel('time (in ms)')
ylabel('Amplitude')
title('Noisy Signal')

% FFT
X = fft(x,N);
figure(3)
plot(abs(X))
SSB = X(1:N/2);
SSB(2:end) = 2*SSB(2:end);
f = (0:N/2-1)*(Fs/N);
% Amplitude
figure(4)
plot(f,abs(SSB/L))
xlabel('f (in Hz)')
ylabel('|X(f)|')
title('Corrected Frequency Spectrum')
```

fft_sample.m hosted with ❤ by GitHub   view raw

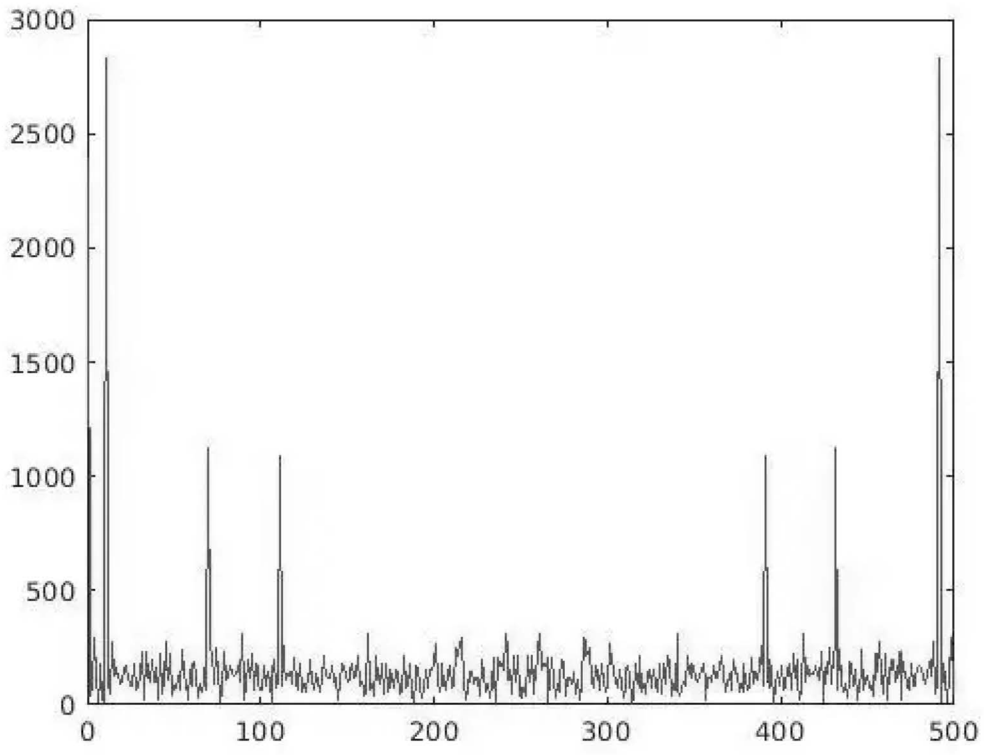
In what follows next, we will be taking parts from this code and analyze it. First, let us set up the context. We have a sampler which can sample 1000 samples in a second. We have gathered 500 time points using that sampler.

```
Fs = 1000;          %sampling rate
T = 1/Fs;           %sampling interval
L = 500;            %Number of time points
t = 0:T:(L-1)*T;    %time vector
```

Next, let us create a signal with 3 frequencies and a DC component (component with 0 Hz) which will be sampled. Let us add some noise as well. If you want, you can plot your signal for an initial visual inspection. That's a good practice.

```
% Define frequencies
f1 = 20;
f2 = 220;
f3 = 138;
% Create the signal
x  = 5 + 12*sin(2*pi*f1*t-0.3) + 5*sin(2*pi*f2*t-0.5) +
4*sin(2*pi*f3*t - 1.5);
x = awgn(x,5,'measured') %noise with 5 dB Signal to Noise ratio
```

Now, we are all set for performing FFT. We have a real world signal in our hand which is a mixture of sinusoids corrupted by additive noise. Let us take the FFT and plot it. Bummer. Oh yes, FFT sequences are complex sequences. Why? Because they contain information about amplitude and phase for a given frequency component and complex numbers are the best way to describe it. Let us take the absolute value (magnitude) of those sequences then. What do we get?



The magnitude of the FFT sequences FFT(x)

This do not make much sense at all. First of all, there are 7 peaks (including the one at zero). But we were expecting 4 peaks, (3 for frequencies f1,f2,f3 and 1 for the DC component). Secondly, what has happened to the amplitude? It looks too high! The peaks are in wrong place as well. It is pretty evident that to work with this thing, we have works to do.

Let us break the first confusion: why there are more peaks than we were expecting? This is where Single Side Band (SSB) spectrum comes to play. If you look closely, one side of the plot above is the mirror reflection of another side, dropping the peak at 0. This is because, for real signals, the coefficients of positive and negative frequencies become complex conjugates. That means, we do not need both sides of spectrum to represent the signal, a single side will do. This is known as Single Side Band (SSB) spectrum. To get the SSB spectrum, we need to our time domain signal into the analytic signal which is $x_a = x + iH(x)$. *H(x)* is the well known Hilbert Transform. But, the good news is- we do not need to learn a new transform for that. It turns out, if you take the spectrum associated with positive frequencies only and multiply it by 2, you will get the SSB spectrum without explicitly applying Hilbert Transform on converted

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.