GAUSSIANWAVES

Signal Processing Simplified

LATEST ARTICLES, MATLAB CODES, SIGNAL PROCESSING, TIPS & TRICKS, TUTORIALS

How to interpret FFT results – obtaining magnitude and phase information

Posted on November 19, 2015 by Mathuranathan in Latest Articles, Matlab Codes, Signal Processing, Tips & Tricks, Tutorials

(**74** votes, average: **4.36** out of 5)

This post is a part of the ebook : Digital Modulations using Matlab – Build Simulation Models from Scratch -by Mathuranathan Viswanathan

In the previous post, Interpretation of frequency bins, frequency axis arrangement (fftshift/ifftshift) for complex DFT were discussed. In this post, I intend to show you how to obtain magnitude and phase information from the FFT results.

Outline

In this discussion, I will take an arbitrary cosine function of the form $x(t) = Acos\left(2\pi f_c t + \phi\right)$ and proceed step by step as follows

1. Represent the signal $x(t)$ in computer (discrete-time) and plot the signal (time domain)
2. Represent the signal in frequency domain using FFT ($X[k]$)
3. Extract amplitude and phase information from the FFT result
4. Reconstruct the time domain signal from the frequency domain samples

For more such examples check this ebook : Digital Modulations using Matlab : build simulation models from scratch – by Mathuranathan Viswanathan

1. Discrete-time domain representation

Consider a cosine signal of amplitude $A = 0.5$, frequency $f_c = 10Hz$ and phase $\phi = \pi/6$ radians (or 30°)

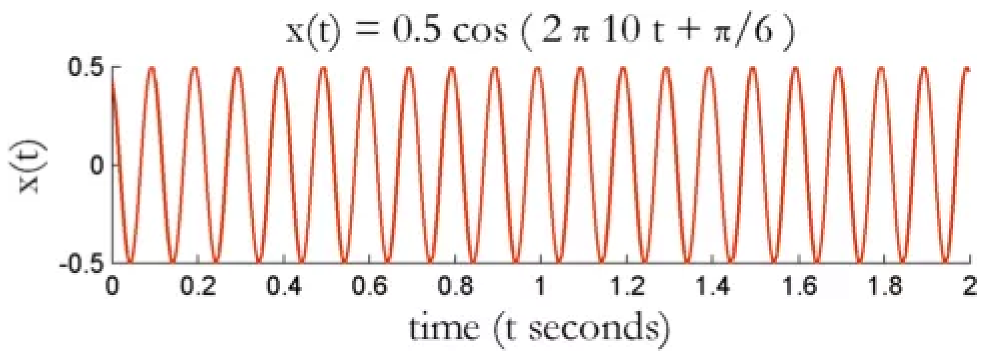
$$x(t) = 0.5cos\left(2\pi 10t + \pi/6\right)$$

In order to represent the continuous time signal $x(t)$ in computer memory, we need to sample the signal at sufficiently high rate (according to Nyquist sampling theorem). I have chosen a oversampling factor of 32 so that the sampling frequency will be $f_s = 32 \times f_c$, and that gives 640 samples in a 2 seconds duration of the waveform record.

```
A = 0.5; %amplitude of the cosine wave
fc=10;%frequency of the cosine wave
phase=30; %desired phase shift of the cosine in degrees
fs=32*fc;%sampling frequency with oversampling factor 32
t=0:1/fs:2-1/fs;%2 seconds duration

phi = phase*pi/180; %convert phase shift in degrees in radians
x=A*cos(2*pi*fc*t+phi);%time domain signal with phase shift

figure; plot(t,x); %plot the signal
```



Represent the signal in frequency domain using FFT

Lets represent the signal in frequency domain using the FFT function. The FFT function computes N-point complex DFT. The length of the transformation N should cover the signal of interest otherwise we will some loose valuable information in the conversion process to frequency domain. However, we can choose a reasonable length if we know about the nature of the signal.

For example, the cosine signal of our interest is periodic in nature and is of length 640 samples (for 2 seconds duration signal). We can simply use a lower number $N = 256$ for computing the FFT. In this case, only the first 256 time domain samples will be considered for taking FFT. No need to worry about loss of information in this case, as the 256 samples will have sufficient number of cycles using which we can calculate the frequency information.

```
N=256; %FFT size
X = 1/N*fftshift(fft(x,N));%N-point complex DFT
```

In the code above, $fftshift$ is used only for obtaining a nice double-sided frequency spectrum that delineates negative frequencies and positive frequencies in order. This transformation is not necessary. A scaling factor $1/N$ was used to account for the difference between the FFT implementation in Matlab and the text definition of complex DFT.

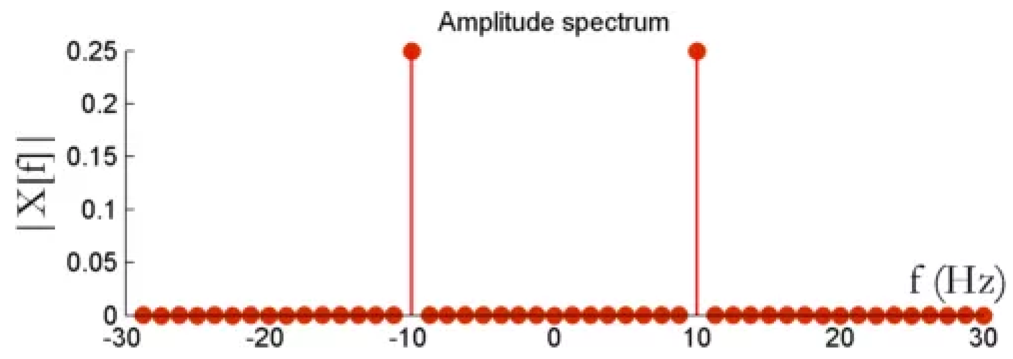
3a. Extract amplitude of frequency components (amplitude spectrum)

The FFT function computes the complex DFT and the hence the results in a sequence of complex numbers of form $X_{re} + jX_{im}$. The amplitude spectrum is obtained

$$|X[k]| = \sqrt{X_{re}^2 + X_{im}^2}$$

For obtaining a double-sided plot, the ordered frequency axis (result of fftshift) is computed based on the sampling frequency and the amplitude spectrum is plotted.

```
df=fs/N; %frequency resolution
sampleIndex = -N/2:N/2-1; %ordered index for FFT plot
f=sampleIndex*df; %x-axis index converted to ordered frequencies
stem(f,abs(X)); %magnitudes vs frequencies
xlabel('f (Hz)'); ylabel('|X(k)|');
```



3b. Extract phase of frequency components (phase spectrum)

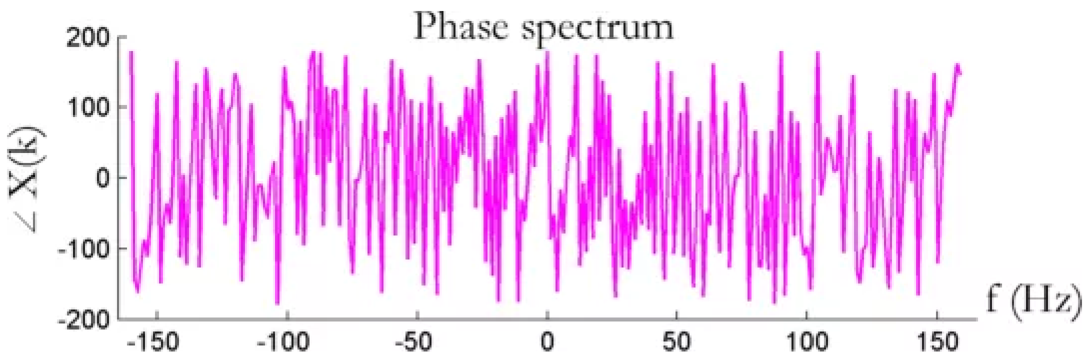
Extracting the correct phase spectrum is a tricky business. I will show you why it is so. The phase of the spectral components are computed as

$$\angle X[k] = tan^{-1}\left(\frac{X_{im}}{X_{re}}\right)$$

That equation looks naive, but one should be careful when computing the inverse tangents using computers. The obvious choice for implementation seems to be the $atan$ function in Matlab. However, usage of $atan$ function will prove disastrous unless additional precautions are taken. The $atan$ function computes the inverse tangent over two quadrants only, i.e, it will return values only in the $[-\pi/2, \pi/2]$ interval. Therefore, the phase need to be unwrapped properly. We can simply fix this issue by computing the inverse tangent over all the four quadrants using the $atan2(X_{img}, X_{re})$ function.

Lets compute and plot the phase information using $atan2$ function and see how the phase spectrum looks

```
phase=atan2(imag(X),real(X))*180/pi; %phase information
plot(f,phase); %phase vs frequencies
```



The phase spectrum is completely noisy. Unexpected !!!. The phase spectrum is noisy due to fact that the inverse tangents are computed from the $ratio$ of imaginary part to real part of the FFT result. Even a small floating rounding off error will amplify the result and manifest incorrectly as useful phase information (read how a computer program approximates very small numbers).

To understand, print the first few samples from the FFT result and observe that they are not absolute zeros (they are very small numbers in the order 10^{-16}. Computing inverse tangent will result in incorrect results

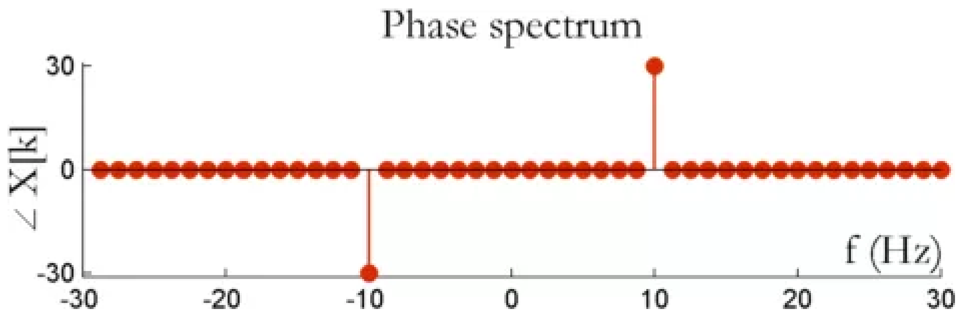
```
>> X(1:5)
ans =
   1.0e-16 *
  -0.7286            -0.3637 - 0.2501i   -0.4809 - 0.1579i   -0.3602 - 0.5579i   0.026
>> atan2(imag(X(1:5)),real(X(1:5)))
ans =
    3.1416   -2.5391   -2.8244   -2.1441   -1.5181
```

The solution is to define a tolerance threshold and ignore all the computed phase values that are below the threshold.

```
X2=X;%store the FFT results in another array
%detect noise (very small numbers (eps)) and ignore them
```

```
threshold = max(abs(X))/10000; %tolerance threshold
X2(abs(X)<threshold) = 0; %maskout values that are below the threshold
phase=atan2(imag(X2),real(X2))*180/pi; %phase information
plot(f,phase); %phase vs frequencies
```

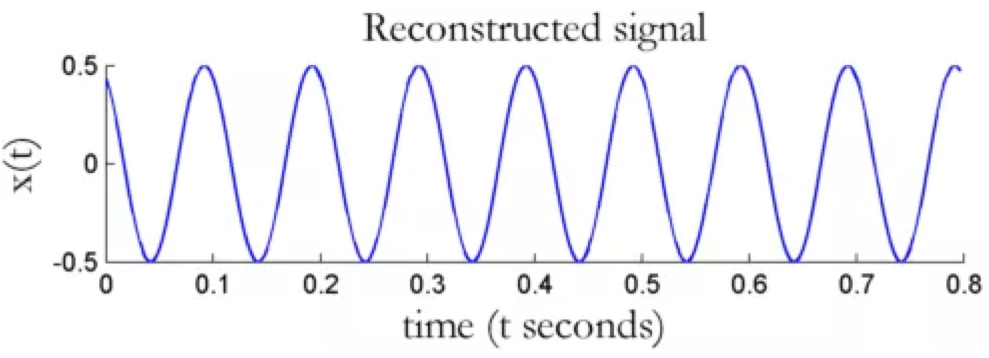
The recomputed phase spectrum is plotted below. The phase spectrum has correctly registered the 30° phase shift at the frequency $f = 10Hz$. The phase spectrum is anti-symmetric ($\phi = -30^\circ$ at $f = -10Hz$), which is expected for real-valued signals.



4. Reconstruct the time domain signal from the frequency domain samples

Reconstruction of the time domain signal from the frequency domain sample is pretty straightforward

```
x_recon = N*ifft(ifftshift(X),N); %reconstructed signal
t = [0:1:length(x_recon)-1]/fs; %recompute time index
plot(t,x_recon);%reconstructed signal
```



The reconstructed signal has preserved the same initial phase shift and the frequency of the original signal. Note: The length of the reconstructed signal is only 256 sample long (~ 0.8 seconds duration), this is because the size of FFT is considered as $N = 256$. Since the signal is periodic it is not a concern. For more complicated signals, appropriate FFT length (better to use a value that is larger than the length of the signal) need to be used.

Rate this article: ★★★★☆ **(74 votes, average: 4.36 out of 5)**

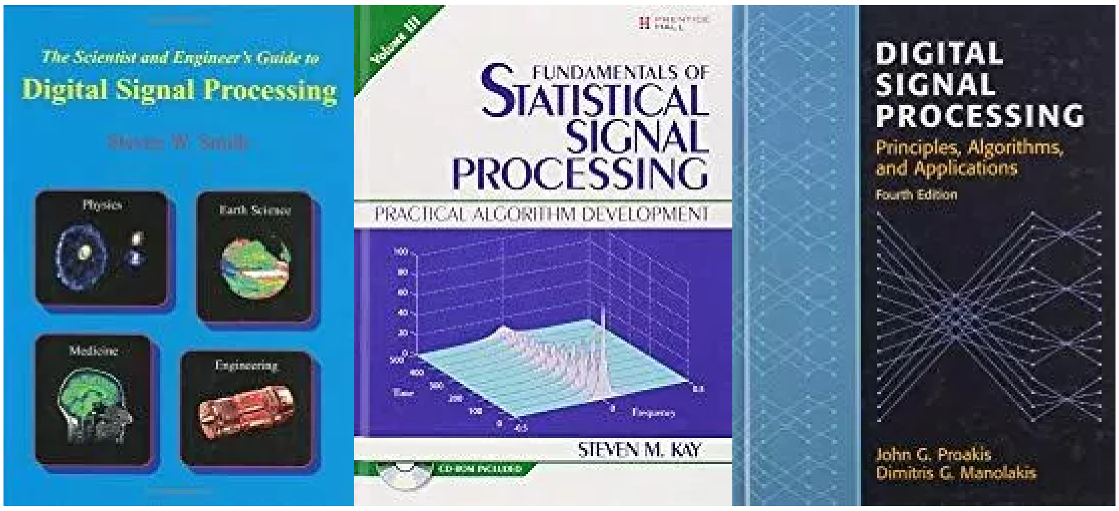
Articles in this series:

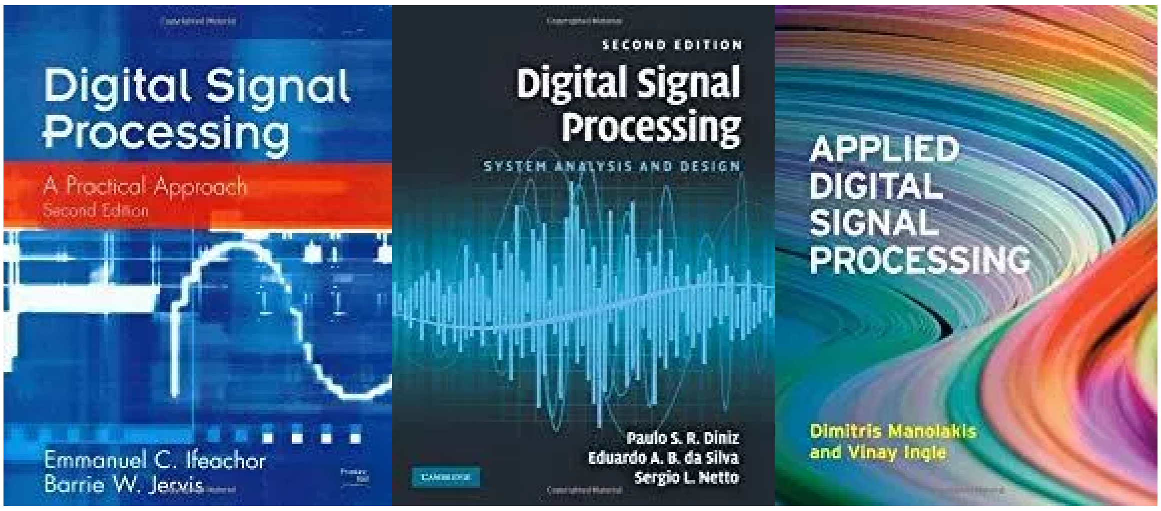
How to Interpret FFT results – complex DFT, frequency bins and FFTShift

How to Interpret FFT results – obtaining Magnitude and Phase information (this article)
FFT and Spectral Leakage
How to plot FFT using Matlab – FFT of basic signals : Sine and Cosine waves
Generating Basic signals – Square Wave and Power Spectral Density using FFT
Generating Basic signals – Rectangular Pulse and Power Spectral Density using FFT
Generating Basic Signals – Gaussian Pulse and Power Spectral Density using FFT
Chirp Signal – Frequency Sweeping – FFT and power spectral density
Constructing the Auto Correlation Matrix using FFT

For more such examples check this ebook : Digital Modulations using Matlab: build simulation models from scratch – by Mathuranathan Viswanathan

Recommended Signal Processing Books


The Scientist and Engineer's Guide to Digital Signal Processing
Steven W. Smith
Fundamentals of Statistical Signal Processing
Practical Algorithm Development
Steven M. Kay
Digital Signal Processing
Principles, Algorithms, and Applications
Fourth Edition
John G. Proakis
Dimitris G. Manolakis


Digital Signal Processing
A Practical Approach
Second Edition
Emmanuel C. Ifeachor
Barrie W. Jervis
Second Edition
Digital Signal Processing
System Analysis and Design
Paulo S. R. Diniz
Eduardo A. B. da Silva
Sergio L. Netto
Applied Digital Signal Processing
Dimitris Manolakis and Vinay Ingle

Share this:

Share 2 Tweet G+ 공유 Share submit More

COMPLEX DFT DFT FFT FFTSHIFT REAL DFT

PREVIOUS POST
How to Interpret FFT results – complex DFT, frequency bins and FFTShift

NEXT POST
Quiz on Logic gates

Gaussianwaves Comment Policy
We welcome relevant and respectful comment. Any deviating comment will be deleted.
Please read our Comment Policy before commenting.

17 Comments **Gaussianwaves** 1 **Login**

Recommend 5 Share Sort by Best

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

afshin aghayan • a year ago
look at the following Matlab function, it can calculate phase spectrum as well as amplitude spectrum with a perfect accuracy:

https://www.mathworks.com/m...
1 • Reply • Share ›

Mathuranathan Mod → afshin aghayan • a year ago
Great work !! it will definitely help everyone.
1 • Reply • Share ›

Dhanya E • a year ago
Hi Mathuranathan, thank you for this article.
I have a doubt regarding calculation of DFT in matlab using DFT formulae and using inbulit MATLAB function FFT.
The
phase of DFT computed using DFT formula and FFT (inbuilt MATLAB function) is different at N/2. Why is it so? Attaching sample code

```
clc;
close all;
clear all;

x = [2 3 -1 4];
N = length(x);
X = zeros(4,1)

%DFT formulae
for k = 0:N-1
for n = 0:N-1
X(k+1) = X(k+1) + x(n+1)*exp(-j*pi/2*n*k)
```

see more

Reply • Share ›

Rohan Kotwani • a year ago

Thanks for this! I was wondering my my phase plot was not working. It turns out it was because of the atan function

Reply • Share ›

Benjamin Reboui • a year ago

Hello Mathuranathan
Thank you very much for this article, I've a naive question, why when I try with a sine wave
x=A*sin(2*pi*fc*t+phi);
I get phase spectrum peak at 60 degree (pi/3) instead of 30 degree.
I'm not sure why this is happening do you have any clue?
It's probably because cos(x)=-sin(x+pi/2) but I do not see how the phase origin is taken.
Thank you very much

Reply • Share ›

Mathuranathan Mod → Benjamin Reboui • a year ago

FFT does not know if the input signal is cosine or sine. We just compute the angle as the tan^-1(imaginary part/real part). The angle is always measured between the vector and the real axis.

To give you clue, investigate the FFT output of a pure cosine wave and pure sine wave. Check the real part and imaginary part in each output, you will begin to understand

• Reply • Share ›

Nitin • a year ago

Thanks for the explanation. If I increase N to 1024 or change fs to 20*fc, the magnitude and phase plots changes. Why does this happen?

• Reply • Share ›

Mathuranathan Mod Nitin • a year ago

It is due to the spectral leakage. If the input signal is not periodic with respect to the length of FFT, it creates artifacts in the input signal that creates all these types of issues.

Examples here: http://www.gaussianwaves.co...

• Reply • Share ›

Nitin • a year ago

--

• Reply • Share ›

orial • a year ago

This tutorial was amazing for me. Thank you for typing up the extremely clear presentation. I'm trying to understand the phase I'm getting - my data is an almost pure cosine from an RK4 algorithm but the phase has a discontinuity right at the frequency of the solution!

Reply • Share ›

Mathuranathan Mod ➦ orial • a year ago

Thanks !!!. Is it really a pure cosine function ? is it possible to share the input data ?

Reply • Share ›

orial ➦ Mathuranathan • a year ago

Thank you for the response. I found the answer - my solution had constant amplitude and frequency but was varying in phase over time, so the FFT accurately described my data.

Reply • Share ›

Giovanni Piccinni • a year ago

Hi Mathuranathan, thank you for this article, it's very helpful. I'd like to extract the phase information from an OFDM signal. How can I do this? Thanks in advance

Reply • Share ›

Mathuranathan Mod ➦ Giovanni Piccinni • a year ago

In OFDM, the phase estimation should be done after the FFT block in the receiver. Several algorithms/technique available for phase estimation in an OFDM system.

Consider a coherent detection system.
1. For each transmitted OFDM symbol, the FFT output contains say N modulated values (say QAM modulation is used)
2. These values may contain random phase shifts and amplitude variations caused by local oscillator drift, jitter, channel response and other factors.
3. We usually use a channel estimator (can be a pilot/preamble based one) to estimate the reference phases and amplitudes.
4. Once the reference phases and amplitudes are learnt, we can apply them to correct phase/amplitude errors in the actual data transmission.

You can take a look at this patent. I hope it offers more clue on how to use it for your system.
https://www.google.com/pate...

Reply • Share ›

Giovanni Piccinni → Mathuranathan • a year ago

Thank you for your answer. In my system, an OFDM signal is used for extracting the distance between a TX and RX. The signal is composed solely by zadoff-chu pilots. I have extracted a coarse distance measure from the correlation function between the received signal and a refrence signal. Now I'd like to perform a fine estimation exploiting the signal phase estimated in frequency domain but I'm not able to do this. Can you help me please?

Reply • Share ›

Marivn • 2 years ago

how can I realize this in java?

Reply • Share ›

Mathuranathan Mod → Marivn • 2 years ago

I do not have the code in Java

Reply • Share ›

ALSO ON **GAUSSIANWAVES**

Chirp Signal – Frequency Sweeping – FFT and power spectral density

Modeling a Frequency Selective Multipath Fading channel using TDL

Proudly powered by WordPress
Theme: Satellite by WordPress.com.