EE 521: Instrumentation and Measurements

Aly El-Osery

Electrical Engineering Department, New Mexico Tech
Socorro, New Mexico, USA

October 26, 2009

Rect Window

Figure: Fourier transform pair - rectangular window and sinc function

- Fourier transform of a rectangular window is a sinc.
- Inverse Fourier transform of a rectangular window is also a sinc.
- We can only have either timelimited or bandlimited but not both.

Transform Equations

DTFT

$$x(n) = \frac{1}{2\pi} \int_{2\pi} X(\omega) e^{j\omega n} d\omega \qquad (1)$$

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n) e^{-j\omega n} \qquad (2)$$

Transform Equations

DFT

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}, \qquad k = 0, 1, 2, \ldots, N-1 \tag{3}$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi kn/N}, \qquad n = 0, 1, 2, \ldots, N-1 \tag{4}$$

CTFT

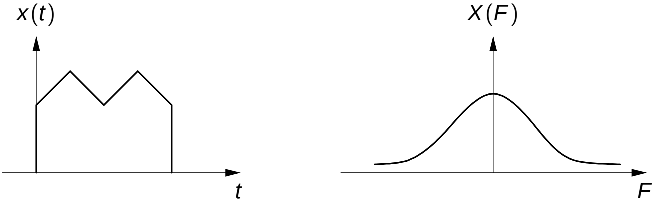

Figure: Continuous Fourier transform of continuous signal

DTFT

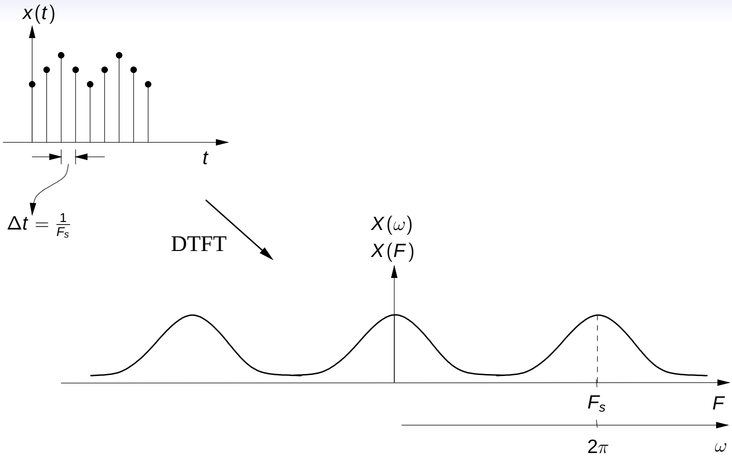

Figure: Discrete time Fourier transform

DFT

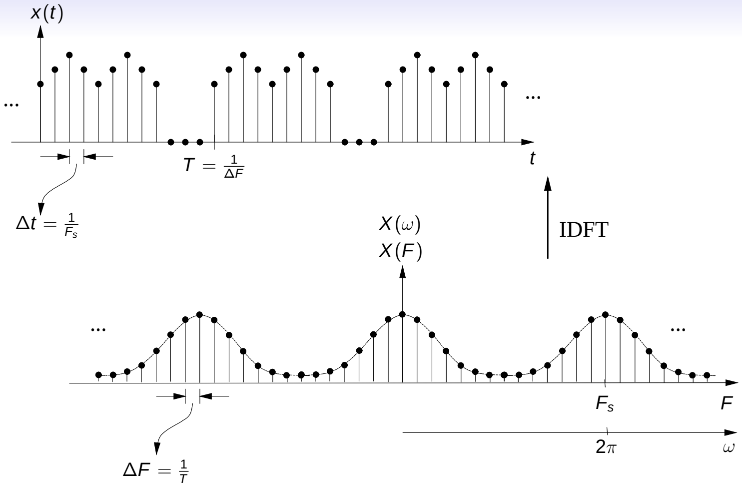

Figure: Discrete Fourier transform

Limitations

1. The number of data points must be finite.
2. The computation time required increase as the number of data points increase.
3. Frequency resolution is important in determining the signal content.
4. Limiting the number of points of a continuous time signal results in *spectral leakage*.

Number of Points

Assume the number of point in the time domain is N_t and the number of points in the frequency domain is N_F.

$$N_t = \frac{T}{\Delta t} = \frac{1/\Delta F}{1/F_s} = \frac{F_s}{\Delta F} = N_F = N \tag{5}$$

Zero Padding

We can increase the time series sequence by adding zeros and that would not affect it. By doing so the number of points in the time domain, N_t is increased, and consequently, also is the number of points in the frequency domain, N_F. Referring to Eq. 5, this means that ΔF is decreased.

Zero padding shows more details but not more information

Window Size

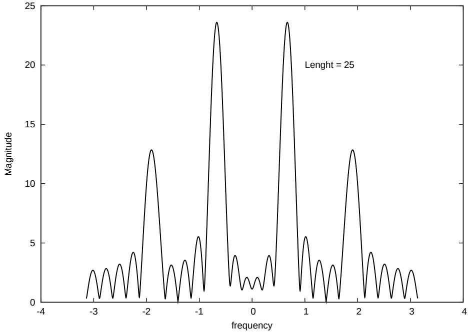

Figure: $N = 4096$

Window Size

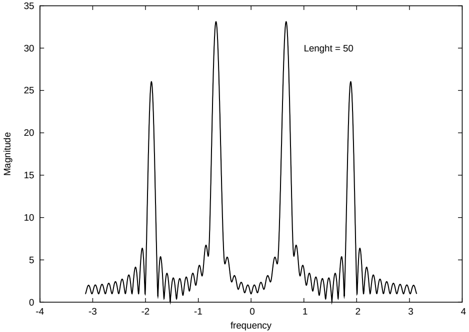

Figure: $N = 4096$

Window Size

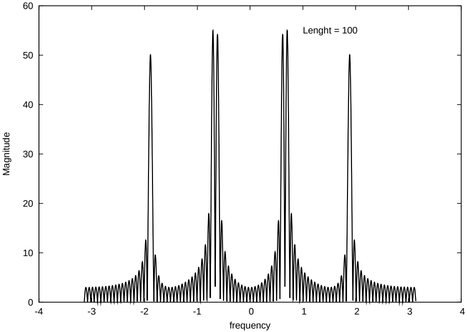

Figure: $N = 4096$

Min. Resolvable Resolution

The DTFT of a rectangular window of length L is given by

$$W(\omega) = \frac{\sin(\omega L/2)}{\sin(\omega/2)} e^{-j\omega(L-1)/2} \tag{6}$$

To avoid main lobes of overlapping

$$|\omega_1 - \omega_2| > 2\pi/L \tag{7}$$

Different Windows

Spectral leakage is due to the sharp cut-off rectangular window. To reduce this effect different windows with smoother roll-off are used. This is at the cost of wider main lobe which may be undesirable.

Hamming Window

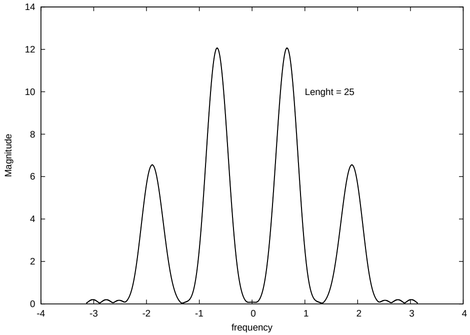

Figure: $N = 4096$

Hamming Window

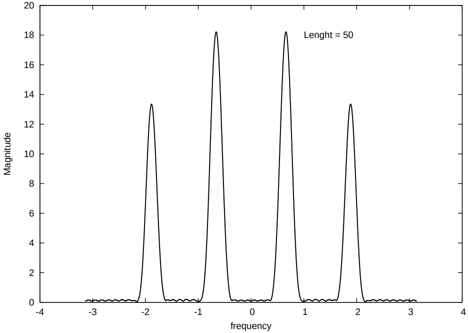

Figure: $N = 4096$

Hamming Window

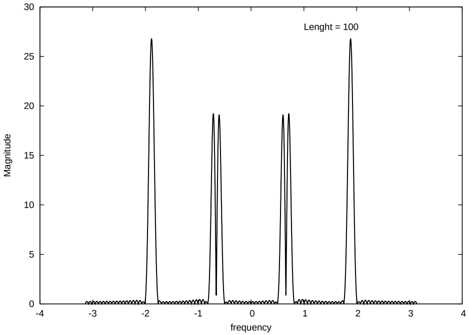

Figure: $N = 4096$

- An efficient way to compute DFT.
- Direct computation of DFT requires approx. N^2 complex multiplications and N^2 complex additions.
- FFT algorithm requires approximately $N/2\log_2 N$.
- For a 1024 point signal direct computation requires 1,048,576 complex computation versus 5,120 of the FFT.

Example

Best way to explain that is using an example. Assume that we have a sinusoidal signal that we want to determine its Fourier transform.

```
>> fs=100;
>> t=0:1/fs:1;
>> y=cos(2*pi*t)
>> plot(t,y)
>> xlabel('t')
>> ylabel('y(t)'
```

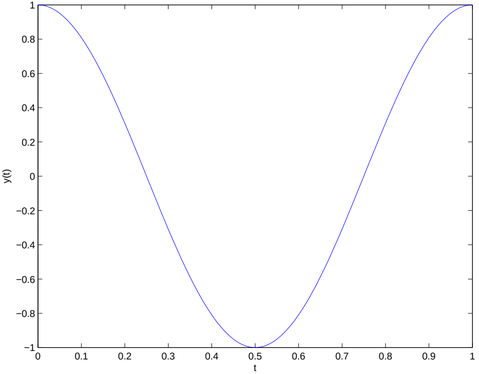

Figure: 1sec sinwave of 1Hz

Straight FFT

Use the following commands to compute the FFT, find its length and plot the magnitude of Y.

```
>> Y=fft(y);
>> length(Y)
ans =
   101
>> plot(abs(Y))
```

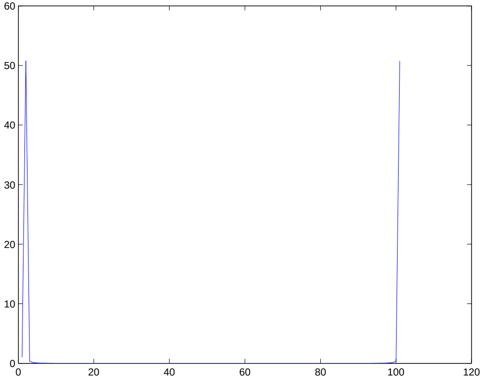

Figure: 101-point FFT

More Details

We can specify the length of the FFT to be longer by

```
>> N=1024;
>> Y=fft(y,N);
>> plot(abs(Y))
```

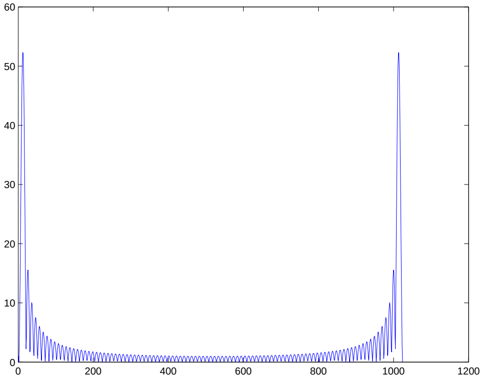

Figure: 1024-point FFT

fftshift

```
>> plot(fftshift(abs(Y)))
```

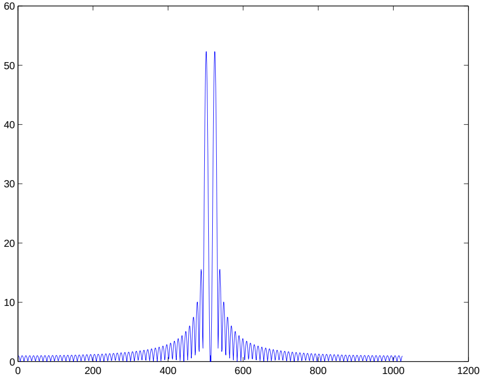

Figure: `fftshift`

Axes Mapping

```
>> k=-N/2:N/2-1;
>> plot(k,fftshift(abs(Y)))
```

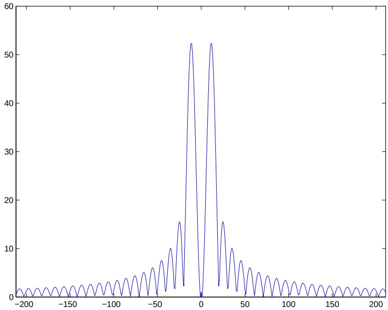

Figure: `fftshift` with $k = -N/2 : N/2 - 1$

Axes Mapping

Now we get the axis containing positive and negative values. All we have left to do is to map it to actual frequencies.

```
>> plot(k*fs/N,fftshift(abs(Y)))
```

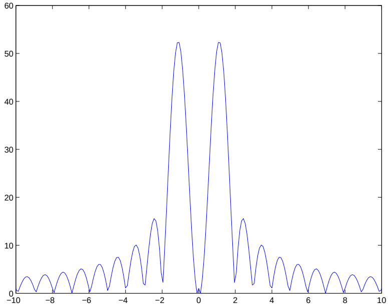

Figure: Frequency mapping

Window Length

```
>> fs=100;
>> t=0:1/fs:10;
>> y=cos(2*pi*t);
>> N=4096;
>> Y=fft(y,N);
>> k=-N/2:N/2-1;
>> plot(k*fs/N,...
   fftshift(abs(Y))
```

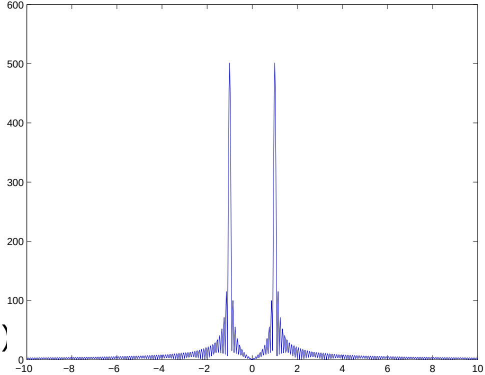

Figure: FFT of a 10sec 1Hz sinwave