TRIBHUVAN UNIVERSITY

Institute of Engineering

Central Campus, Pulchowk

Lab Report On:

Signal Analysis

Experiment No. 4

Submitted To:

Department of Electronics and Computer Engineering

Submitted By:

Rajil Bajracharya (073BEX430)

Submitted On:

01-08-019

Convolution, Fourier Transform and Fourier Series

1. Objectives

- To understand convolution of two discrete signals.
- To understand the gate-sinc Fourier transform pair.
- To express the triangular wave in terms of Fourier Series.

2. Theory

2.1 Fourier Series Representation

The Fourier Series Representation of any signal was developed by Joseph Fourier. According to him, 'Any periodic signal can be rewritten as the weighted sum of sines and cosines of different frequencies. Although his theory was not accepted in the mathematical community during his time, it finds widespread usage in modern signal analysis as well as a lot many fields such as image processing, file compression, cryptography, etc. The Fourier series is important because it allows one to model periodic signals as a sum of distinct harmonic components. This allows us to analyse the frequencies present in the harmonics present in the given signal, thereby allowing us to filter/manipulate particular frequency components. Any signal $x(t)$ can be represented as the sum of weighted sines and cosines of varying frequencies as follows:

$$x(t) = \sum_{k=-\infty}^{\infty} a_k e^{jk\omega t} \ldots (eqn\ 2.1.1)$$

$$where, a_k = \frac{1}{T}\int_0^T x(t)e^{-jk\omega t}dt \ldots (eqn\ 2.1.2)$$

$$Here, \omega = \frac{2\pi}{T}$$

Eqn (2.2.1) is known as the synthesis equation of the Fourier-series while the eqn (2.2.2) is known as the analysis equation.

From eqn (2.2.2), we see that when k = 0, a_k yields a value that is free of the frequency components. Thus, a_o represents the dc component value or simply, the average value of the signal over the given time period.

Eqn(2.1.1) can also be expressed in terms of sine and cosine components of the complex exponential as follows:

$$x(t) = a_o + \sum_{k=1}^{\infty} a_k \cos(k\omega t) + b_k \sin(k\omega t) \ldots eqn(2.1.3)$$

2.2 Fourier Transform:

The Fourier Transform, derived from the Fourier Series explained above, is a powerful mathematical tool used for the spectral analysis of a signal. It is used to convert a time-domain signal into a frequency-domain signal and thus, helps us understand the frequency, phase and magnitude variations of the sinusoidal components present in the given signal. The Fourier transform of a continuous-time signal, $x(t)$ can be obtained by using the analysis equation:

$$X(j\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt \ldots eqn(2.2.1)$$

Similarly, the inverse Fourier transform can be obtained from the synthesis equation as follows:

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\omega)e^{j\omega t}dt \ldots eqn(2.2.2)$$

However, the signal $x(t)$ has to satisfy the following convergence conditions for the Fourier transform to be applicable:

- The function should be single-valued in any finite interval, T.
- It should have a finite number of discontinuities in any finite interval, T.
- It should have a finite number of maxima and minima in any finite interval of time, T.
- It should be absolutely integrable i.e.,

$$\int_{-\infty}^{\infty} |x(t)|dt < \infty$$

2.3 Convolution Sum of Discrete Signal

A discrete signal is simply a sequence representing the value of the signal at discrete intervals of time. Convolution is vital tool in understanding Linear Time-Invariant (LTI) systems. Convolution is the process by which an input interacts with an LTI system to produce an output. Convolution between an input signal, $x[n]$ and an LTI system with a system response, $h[n]$ is given as:

$$x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k] \times h[n-k] \ldots eqn(2.3.1)$$

In MATLAB, the library function `conv()` is used to obtain the convolution of two discrete-time signals. Convolution may also be evaluated by hand using any of the following techniques:

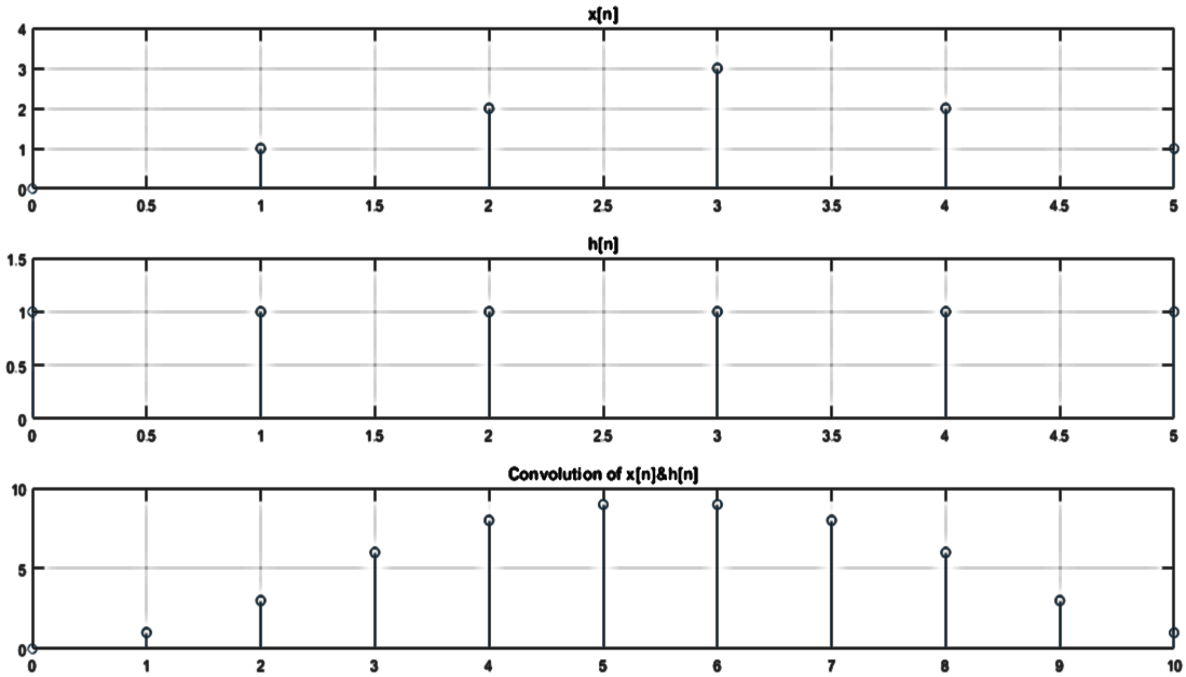
- S^3A Graphical Method
  - Scale, shift, stack, add stack
- FSMA/Table Method
  - Flip, Shift, Multiply, Add
- Analytical Method

3. Code and Outputs:

3.1

```
%convolution
x = [0 1 2 3 2 1];
h = [1 1 1 1 1 1];
convolution = conv(x, h);
nx = numel(x);
nh = numel(h);
nconv = numel(convolution);
argx = 0:nx-1;
argh = 0:nh-1;
argconv = 0:nconv-1;

subplot(3,1,1)
stem(argx, x);
axis([0 5 0 4]);
title('x[n]');
grid on;
subplot(3,1,2)
stem(argh, h);
axis([0 5 0 1.5]);
title('h[n]');
grid on;
subplot(3,1,3)
stem(argconv, convolution);
axis([0 10 0 10]);
title('Convolution of x[n]&h[n]');
grid on;
```

OUTPUT:

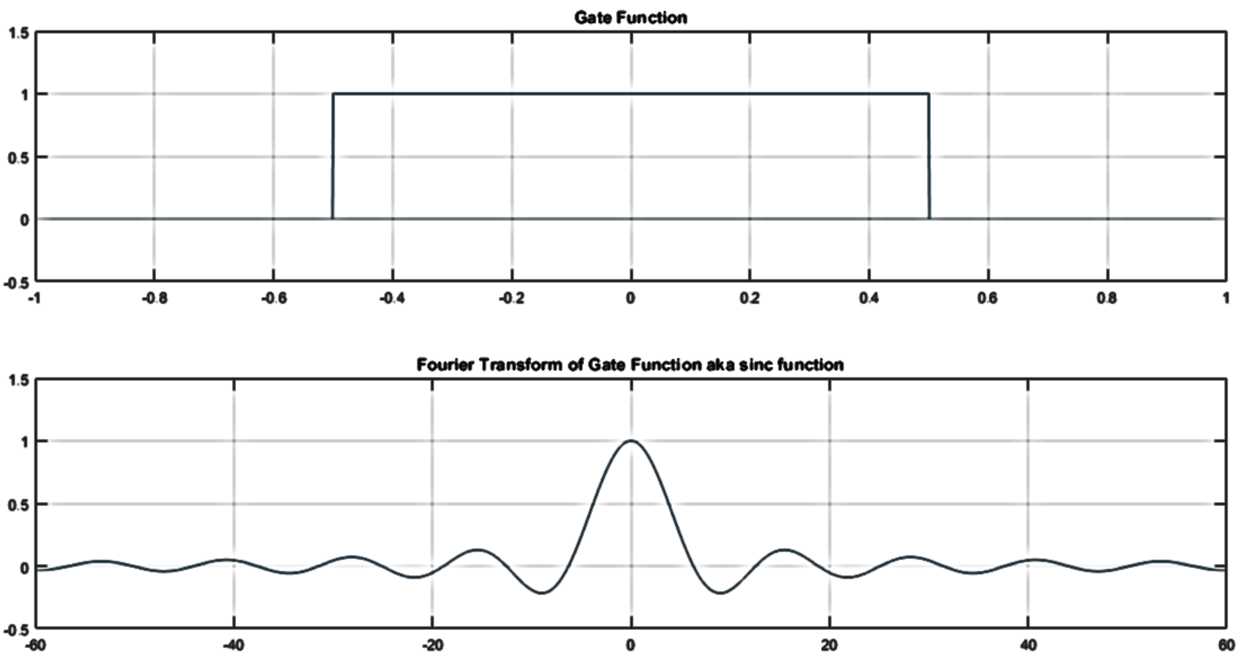


3.2

```
Fs = 1000;
dt = 1/Fs;
t = -1 : dt : 1;
n = numel(t);
```

```
%define gate function
rect = zeros(n);
for i = 1:n
    if t(i) >= -0.5 && t(i) <= 0.5
        rect(i) = 1;
    end
end

%get Fourier Tranform
%X = int(exp(-1j * w * t), t, -0.5, 0.5)
tF = -20*pi:dt:20*pi;

X = 2 * sin(tF/2)./tF;

subplot(2,1,1)
plot(t, rect);
axis([-1 1 -0.5 1.5])
title('Gate Function');
grid on
subplot(2,1,2);
plot(tF, X);
axis([-60 60 -0.5 1.5]);
title('Fourier Transform of Gate Function aka sinc function');
grid on
```

OUTPUT:

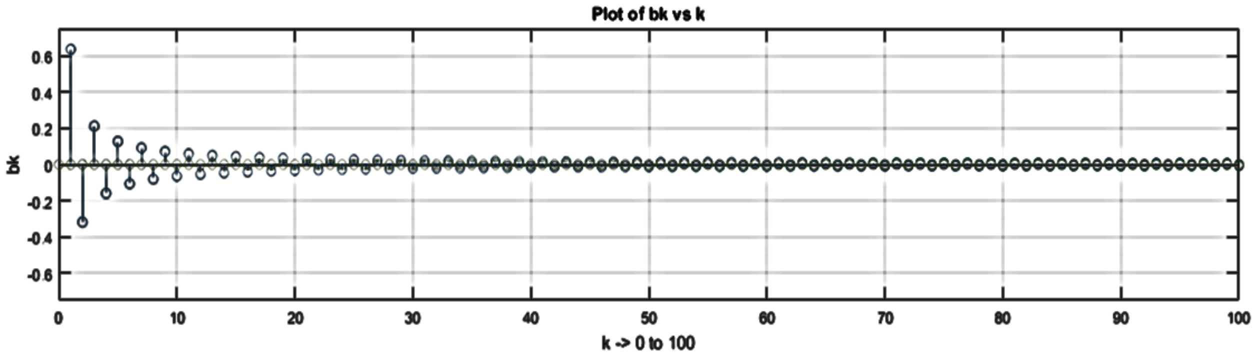


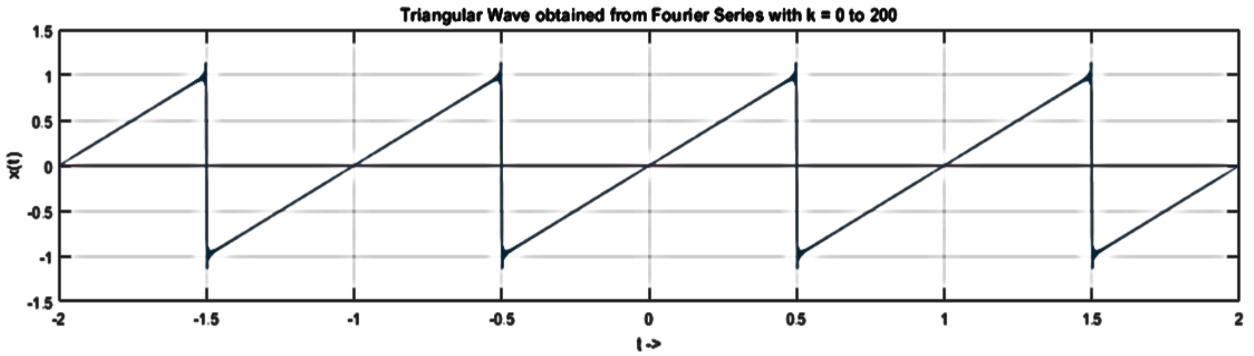
3.3

```
% %triangular wave ko a0, an, bn nikaalne also, original signal plot
garne y
% %= 0.5t
% syms t w k;
% a0 = (1/1)*int(2*t, t, -0.5, 0.5)
% ak = (2/1)*int(2*t*cos(k*w*t), t, -0.5, 0.5)
% bk = (4/1)*int(2*t*sin(k*w*t), t, 0, 0.5)
% %result:
```

```matlab
% a0 = 0;
% ak = 0;
% bk = (8*(sin((k*w)/2) - (k*w*cos((k*w)/2))/2))/(k^2*w^2)
Fs = 1000;
dt = 1/Fs;
T = 1; %-0.5 to 0.5
t = -2 : dt: 2;
s = 200; %no. of samples
k = 0: s; %values of k: 0 - 500
nk = numel(k);
nt = numel(t);
bk = zeros(nk);
x = zeros(nt);
w = 2*pi/T;
for i = 2:nk
    bk(i) = (8*(sin((k(i)*w)/2) -
(k(i)*w*cos((k(i)*w)/2))/2))/(k(i)^2*w^2);
end

for it = 1:nt
    sum = 0;
    for ik = 1:nk
        sum = sum + bk(ik) * sin(k(ik) * w * t(it));
    end
    x(it) = sum;
end

subplot(2, 1, 1);
stem(k, bk);
axis([0 100 -0.75 0.75]);
grid on;
ylabel('bk');
xlabel('k -> 0 to 100');
title('Plot of bk vs k');

subplot(2, 1, 2);
plot(t, x);
axis([-2 2 -1.5 1.5]);
grid on;
ylabel('x(t)');
xlabel('t ->');
title('Triangular Wave obtained from Fourier Series with k = 0 to
200');
```

OUTPUT:





4. Discussion and Conclusion

In this way, we used MATLAB to find the convolution of two discrete signals, plot the Fourier Transform of a gate function and plot a triangular wave using Fourier Series.

While plotting the convolution of the two discrete signals using MATLAB's in-built function, it was seen that the result had 11 values, compared to 6 for the discrete signals themselves. This is because the integral in eqn (2.3.1) is evaluated from k = -5 to k = 5. This, in turn, is due to the fact that while evaluating the convolution sum, $h[n]$ is inverted i.e., we evaluate $h[-n]$ and then shift it from k = 0 to k = 5. Since $h[n]$ runs from $n = 0 \; to \; 5$, $h[-n]$ runs from $h[-5] \; to \; h[0]$. Thus, the convolution yields 11 different values.

In the second exercise, the gate-sinc Fourier Transform pair was verified using MATLAB. It was seen that on taking the Fourier transform of a gate function yielded a sinc function.

In the third exercise, a triangular wave was represented using a Fourier series. For this purpose, expressions for the coefficients in the Trigonometric Fourier Series Representation were obtained using the `int()` function in the command-line (for integration). The functions so obtained were then used to obtain the Fourier Series and then, plotted in MATLAB. Slight distortions were seen at the peaks of the triangular wave so formed due to the discontinuity at the peak [Gibbs' Phenomenon].

In this way, various basic manipulations [convolution, Fourier Transform, Fourier Series representation] were performed on various signals.