CS : 215
Signal & Data Communication Laboratory

Experiment: VI-B

Mritunjay Aman (1912170)

07.04.2021

Total Harmonic Distortion(THD)

Aim

- To plot ideal signal, actual signal and error in the first subplot along with numerical value of signal power of ideal signal and actual signal.
- To plot fundamental signal and sum of other harmonics in the second subplot along with numerical value of signal power of the fundamental signal, sum of other harmonics and THD.

Theoretical Background

Any Periodic Signal in itself can be broken up into number of sinusoidal components. The more the periodic signal departs from pure sinusoidal nature, the stronger the harmonic components are. The Addition of these higher frequencies or harmonic components is called as Harmonic Distortion.

Total Harmonic Distortion(THD) is the measure of harmonic distortion present in the signal. The higher the THD, the higher is the harmonic disruption.

Methodology

1. The ideal signal is generated and all values > 7 and < -7 are replaced for real signal.
2. The ideal signal, real signal and error between them is plotted.
3. The Fundamental frequency of the signal is found using Fourier Transform.
4. Using Fourier transform, the fundamental signal and other harmonics are seperated as two.
5. Their inverse Fourier transform is taken and the resulting time domain signals are plotted.
6. Power of the signal is plotted are computed using $NORM$ function, i.e., $P_x = \frac{NORM(x)^2}{LENGTH(x)}$

Code

```
freq = (1/s_ rate);
freq_ range = (-len+1: len-1)*freq/(2*len);
freq_ range = round(double(freq_ range), 2,
'significant');


subplot(1,1,1);
plot(freq_ range, abs(fftshift(ft)));
freq_ data = zeros(len,2);
freq_ data(:, 1) = abs(ft(1:len))';
freq_ data(:, 2) = freq_ range(len:end)';
```

freq_ data = sortrows(freq_ data, 'descend');
freq_ data(freq_ data(:,1)¡freq_ data(4,1),:)=[];

text(freq_ data(:,2), freq_ data(:,1), ...
    num2cell(freq_ data(:,2)'),...
    'VerticalAlignment', 'bottom');
set(gca, ...
    'Box', 'off', ...
    'TickDir', 'out', ...
    'YGrid', 'on', ...
    'FontSize', 15);
axis([0, freq_ range(end)/6 , -500, 4500]);
pbaspect([2,1,1]);

```
1. clc;
2. clear all;
3.
4. s_ rate = 1e-6;
5. t = [0:s_ rate:1e-3];
6. ideal_ sgl = 10*sin(8000*pi*t);
7. subplot(2,1,1);
8. hold on;
9. plot(t, ideal_ sgl, 'b', 'linewidth', 1);
10. set(gca, ...
11.       'Box', 'off', ...
12.       'Ytick', [-10:5:10], ...
13.       'TickDir', 'out', ...
14.       'YGrid', 'on', ...
15.       'FontSize', 15);
16.
17. real_ sgl = ideal_ sgl;
```

```
18. real_ sgl(real_ sgl> 7) = 7;
19. real_ sgl(real_ sgl< -7) = -7;
20. plot(t, real_ sgl, 'r', 'linewidth', 1);
21.
22. error = ideal_ sgl-real_ sgl;
23. plot(t, error, 'm', 'linewidth', 1);
24.
25. legend('Ideal Response', 'Real Response', 'Error',...
26.       'Location','northoutside', ...
27.       'Orientation', 'horizontal');
28. legend('boxoff');
29. pbaspect([4,1,1]);
30. hold off;
31.
32. ft = fft(real_ sgl);
33. len = int32(length(ft)/2);
34.
35. fndmtl = ft;
36. fndmtl(fndmtl < max(ft)) = 0;
37. fndharm = ifft(fndmtl,'symmetric');
38. subplot(2,1,2);
39. hold on;
40. plot(t, fndharm, 'b', 'linewidth', 1);
41. set(gca, ...
42.       'Box', 'off', ...
43.       'Ytick', [-10:5:10], ...
44.       'TickDir', 'out', ...
45.       'YGrid', 'on', ...
```

```
46.       'FontSize', 15);
47.
48. highft = ft;
49. highft(highft >= max(ft)) = 0;
50. highharm = ifft(highft, 'symmetric');
51. plot(t, highharm, 'r', 'linewidth', 1);
52.
53. legend('Fundamental Harmonic', 'Higher Harmonic',...
54.       'Location','northoutside', ...
55.       'Orientation', 'horizontal');
56. legend('boxoff');
57. pbaspect([4,1,1]);
58. hold off;
59. freq = (1/s_ rate);
60. freq_ range = (-len+1:  len-1)*freq/(2*len);
61. freq_ range = round(double(freq_ range), 2,
62. 'significant');
63.
64. subplot(1,1,1);
65. plot(freq_ range, abs(fftshift(ft)));
66. freq_ data = zeros(len,2);
67. freq_ data(:, 1) = abs(ft(1:len))';
68. freq_ data(:, 2) = freq_ range(len:end)';
69. freq_ data = sortrows(freq_ data, 'descend');
70. freq_ data(freq_ data(:,1)<freq_ data(4,1),:)=[];
71.
72. text(freq_ data(:,2), freq_ data(:,1), ...
73.       num2cell(freq_ data(:,2)'),...
```

```
74.       'VerticalAlignment', 'bottom');
75. set(gca, ...
76.       'Box', 'off', ...
77.       'TickDir', 'out', ...
78.       'YGrid', 'on', ...
79.       'FontSize', 15);
80. axis([0, freq_ range(end)/6 , -500, 4500]);
81. pbaspect([2,1,1]);
```

Input Data Description

Given Input functions:

$$x[n] = u[n-1] - u[n-5]$$

$$h[n] = tri\left(\frac{n-6}{4}\right)$$

where

$$tri[m] = \begin{cases} 1-|m|, & |m|<1 \\ 0, & \text{otherwise} \end{cases}$$

A suitable range is taken to ensure no loss of data during convolution.

Result

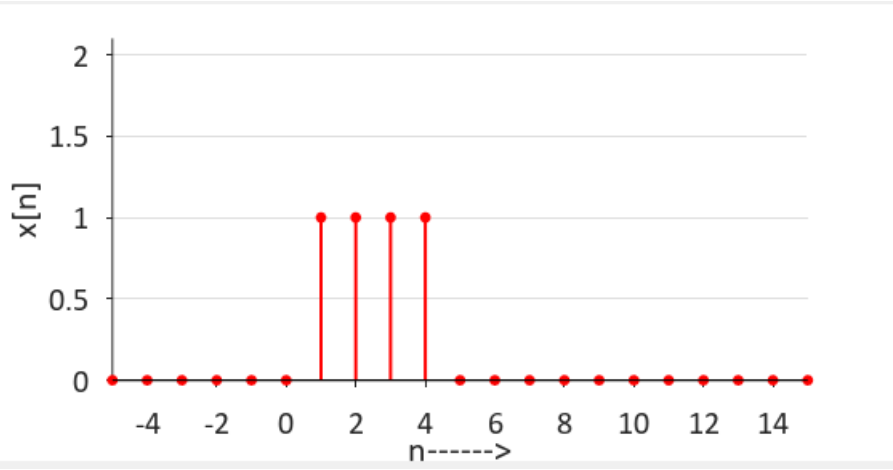

Figure 1: Input Signal, x[n] = u[n - 1] - u[n - 5]

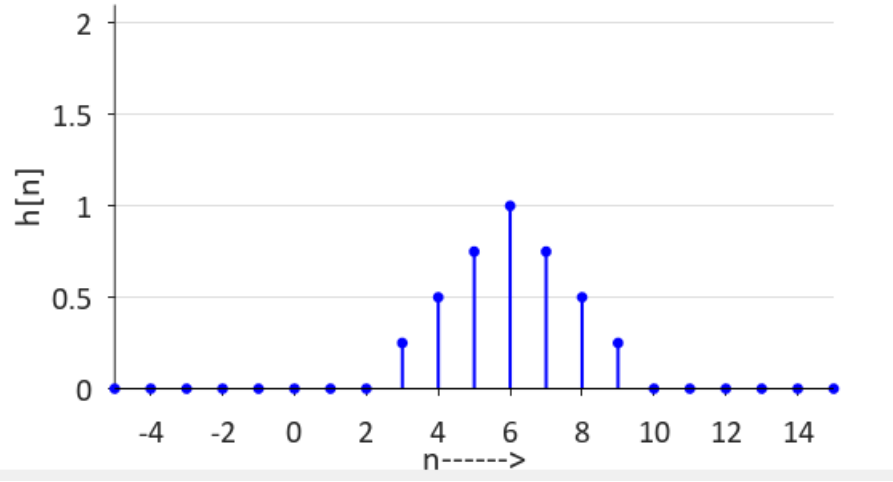

Figure 2: Impulse Response, $h[n] = tri\left(\frac{n-6}{4}\right)$

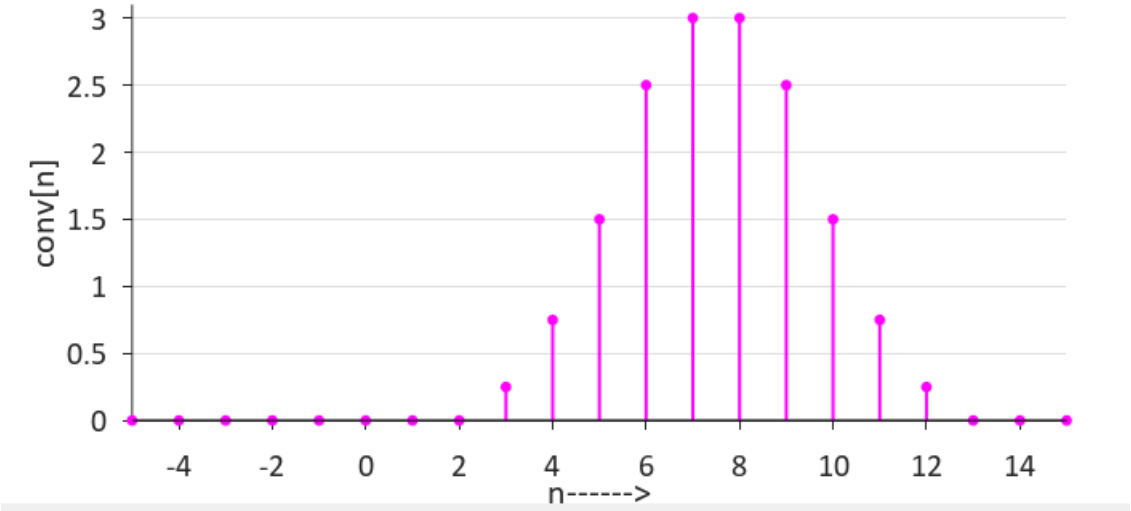

Figure 3: Output Signal

Conclusion

```
The resulting output is a pulse shifted left from both the signals.
```

Filtering and Superposition

Aim

1. Find impulse response when unit step response is

$$s[n] = 2\left[\left(\frac{-1}{2}\right)^n - 1\right] u[n]$$

2. Compute response of system to $x[n] = n.u[n]$ and filter it.

Theoretical Background

The unit step function is the sum of the infinite shifted unit impulse function. Hence, impulse response can be computed by subtracting a shifted unit step response.

Filtering a Signal refers to rectifying its output into a continuous signal that reasonably approximates the original signal. One such method is the Running Average, where the values are substituted by the average value in the window of certain width.

Methodology

1. The impulse response is computed from the given system response.
2. The Convolution Sum is then found using the impulse response.
3. A filtered output is generated using filter and superimpose on original signal.

Code

```
clc;
clear all;

n=-10:1:20;
u = (n>=0);
s = 2*[(-1/2).^(n)-1].*u;

subplot(3,1,1);
stem(n,s,'filled', 'r', 'MarkerSize', 3, 'linewidth', 1);
xlabel('n------>');
```

```
11. ylabel('s[n]');
12. set(gca, ...
13.     'Box', 'off', ...
14.     'TickDir', 'out', ...
15.     'YGrid', 'on', ...
16.     'Xtick', [-6:2:16], ...
17.     'FontSize', 20, ...
18.     'FontName', 'Calibri');
19. axis([-2 10 -3.1 0]);
20.
21. for i=2:31
22.     s2(i) = s(i)-s(i-1);
23. endfor
24. subplot(3,1,2);
25. stem(n,s2,'filled', 'b', 'MarkerSize', 3, 'linewidth', 1);
26. xlabel('n------>');
27. ylabel('h[n]');
28. set(gca, ...
29.     'Box', 'off', ...
30.     'TickDir', 'out', ...
31.     'YGrid', 'on', ...
32.     'Xtick', [-6:2:16], ...
33.     'FontSize', 20, ...
34.     'FontName', 'Calibri');
35. axis([-2 10 -3.1 2]);
36.
37. n1 = -19:1:41;
38. subplot(3,1,3);
```

```
39. x = n.*u;
40. y = conv(x,s2);
41. stem(n1,y,'filled', 'm', 'MarkerSize', 3, 'linewidth', 1);
42. xlabel('n------>');
43. ylabel('conv[n] for x[n] = nu[n] ');
44. set(gca, ...
45.     'Box', 'off', ...
46.     'TickDir', 'out', ...
47.     'YGrid', 'on', ...
48.     'Xtick', [-6:2:16], ...
49.     'FontSize', 20, ...
50.     'FontName', 'Calibri');
51. axis([-2 10 -20 0]);
52. hold on;
53.
54. windowsize = 2;
55. b = (1/windowsize)*ones(1,windowsize);
56. yfiltered = filter(b, 1, y);
57. plot(n1, yfiltered, 'r', 'linewidth', 0.5);
```

Input Data Description

Unit Step Response

$$s[n] = 2\left[\left(\frac{-1}{2}\right)^n - 1\right]u[n]$$

The input to the system is the ramp function

$$x[n] = n.u[n]$$

The windowsize for the filter function is taken to be 2.

Result

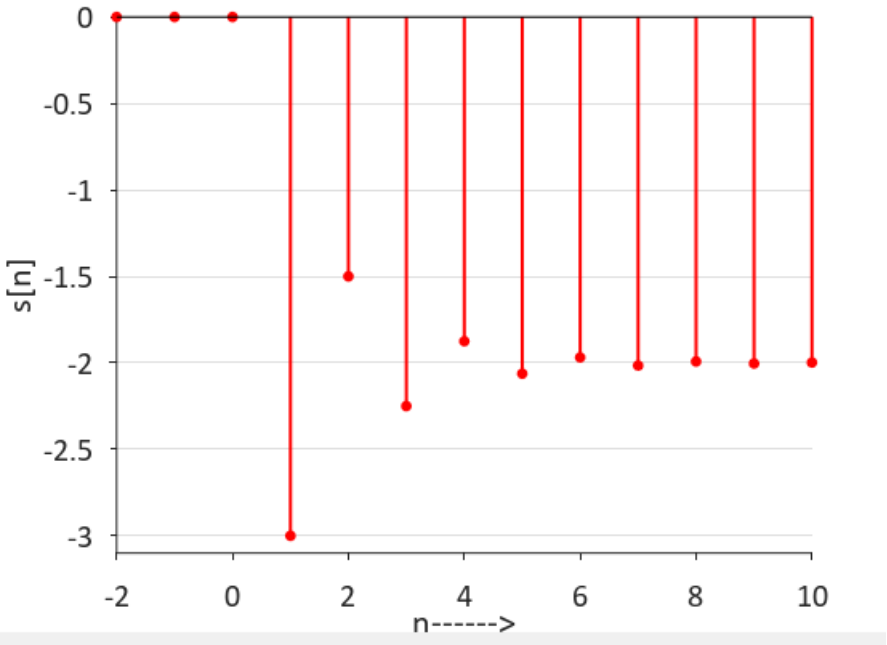

Figure 4: Unit Response, s[n]

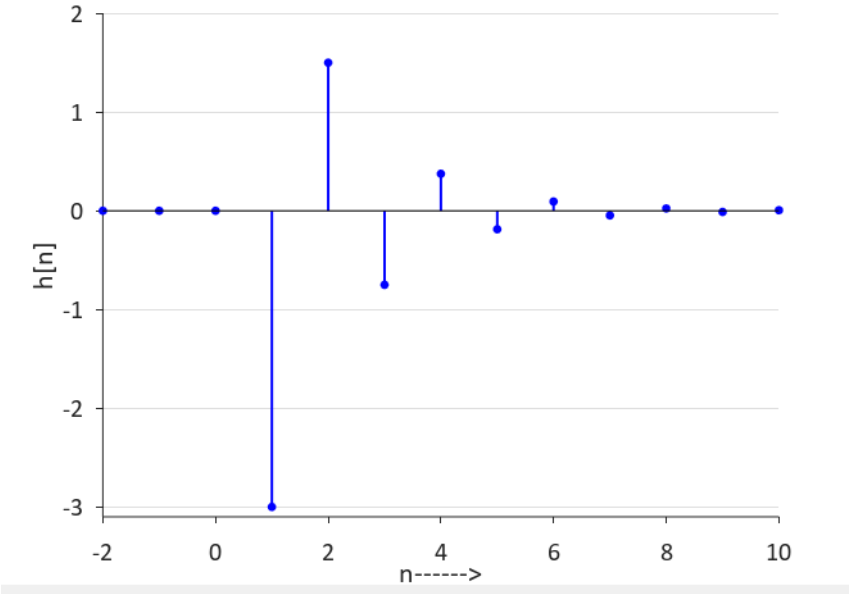

Figure 5: Impulse Response, s[n]-s[n-1]

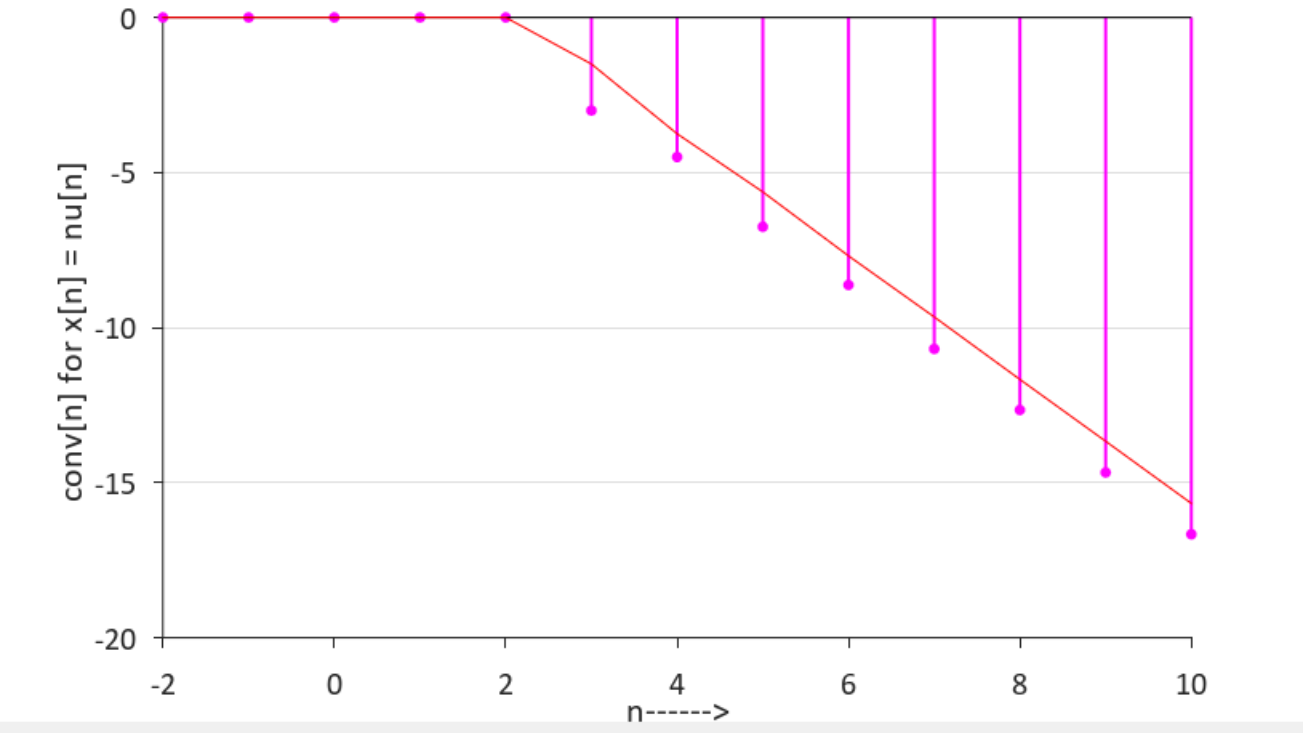

Figure 6: Convoluted Signal and Filtered Counterpart(red)

Conclusion

The filtered output is a Scaled version of the input. The largest(here negative) value of the impulse function dominates the convolution sum.