LAB 1

VATTIKUTI SAI ANVITH – IMT2018528

13 August 2020

CONTENTS

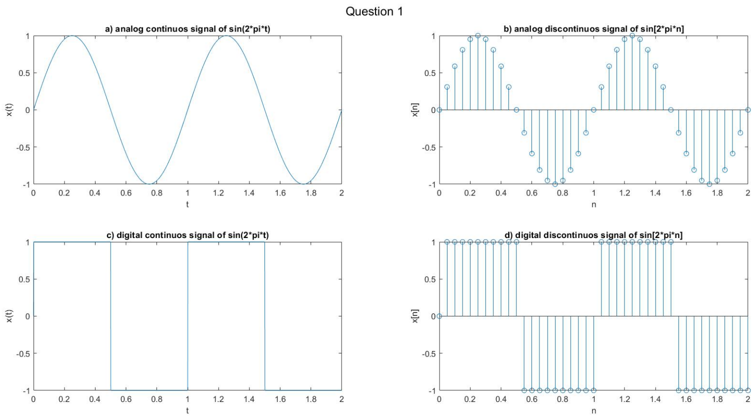
1 QUESTION 1

Consider a sinusoidal signal x(t)=sin(2*π*t) for two complete cycles. Using this signal, plot to represent the following:
a) Analog continuous signal
b) Analog discrete-time signal
c) Digital discrete-time signal
d) Digital continuous signal
(Take a large no. of samples to get a smooth curve. Plot all the sub-parts in the same plot using subplot)

1.1 Code

```
close all, clc

sgtitle('Question 1')

%analog continuos
subplot(2,2,1);
t = 0:0.001:2;
x = sin(2*pi*t);
plot(t, x)
title('a) analog continuos signal of sin(2*pi*t)')
xlabel('t')
ylabel('x(t)')

%analog discontinuos
subplot(2,2,2);
n = 0:0.05:2;
x = sin(2*pi*n);
stem(n, x)
title('b) analog discontinuos signal of sin[2*pi*n]')
xlabel('n')
ylabel('x[n]')

%digital continuos
subplot(2,2,3);
t = 0:0.001:2;
x = sin(2*pi*t);
y = sign(x);
plot(t, y)
title('c) digital continuos signal of sin(2*pi*t)')
xlabel('t')
ylabel('x(t)')

%digital discontinuos
subplot(2,2,4);
n = 0:0.05:2;
x = sin(2*pi*n);
y = sign(x);
stem(n, y)
title('d) digital discontinuos signal of sin[2*pi*n]')
xlabel('n')
ylabel('x[n]')
```

1.2 Plot



1.3 Description & Observations

1.3.1 *Section A*

An analog continuos sine wave is required. Hence I have used plot() function to generate a continuos signal over the large number of samples created. Since the number of samples taken is large we can comfortably observe that the signal output is without much loss of data.

1.3.2 *Section B*

An analog discontinuos sine wave is required. Hence I have used stem() function to generate a discontinuos signal over the samples created. Since the number of samples taken is small we can comfortably observe that the signal output is discrete.

1.3.3 *Section C*

A digital continuos sine wave is required. Hence I have used plot() function to generate a continuos signal over the large number of samples created. As it is digital signal, sign is taken for output through sign() function. We can observe that the output looks like a periodic square wave signal.

1.3.4 *Section D*

A digital discontinuos sine wave is required. Hence I have used stem() function to generate a discontinuos signal over the large number of samples created. As it is digital signal, sign is taken for output through sign() function. Since the number of samples taken is small we can comfortably observe that the signal output is discrete.

2 QUESTION 2

Spectrum plot of sinusoidal signal

a) Generate a sinusoidal signal of amplitude 2 V and frequency 100 Hz. (Take a large no. of samples to get a smooth curve)

b) Plot the amplitude spectrum for the sinusoidal signal generated in part (a).

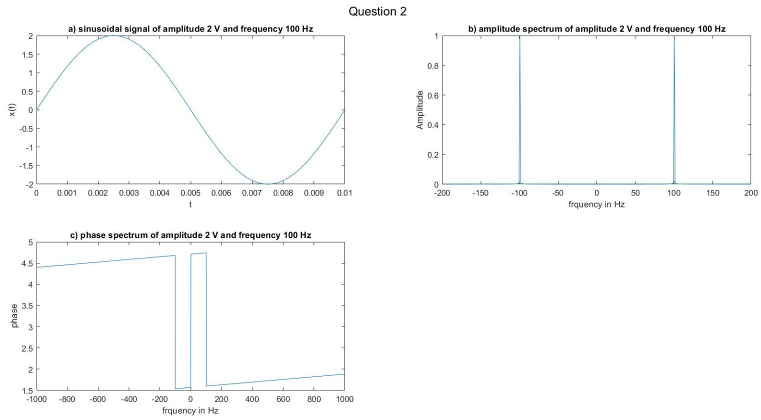
c) Plot the phase spectrum for the sinusoidal signal generated in part (a).

(Take a large no. of samples to get a smooth curve. Plot all the sub-parts in the same plot using subplot)

2.1 Code

```
close all, clc

sgtitle('Question 2')

fs = 10000;
t = linspace(0,1,fs);
f = 100;
y = 2*sin(2*pi*t*f);
dft_ys = fftshift(fft(y))/length(fft(y));
freq = linspace(-fs/2,fs/2,fs);

%analog continuos
subplot(2,2,1)
plot(t,y)
title('a) sinusoidal signal of amplitude 2 V and frequency 100 Hz')
xlabel('t')
ylabel('x(t)')
xlim([0,0.01])

%amplitude spectrum
subplot(2,2,2)
plot(freq,abs(dft_ys))
title('b) amplitude spectrum of amplitude 2 V and frequency 100 Hz')
xlabel('frquency in Hz')
ylabel('Amplitude')
xlim([-200,200])

%phase spectrum
subplot(2,2,3)
plot(freq,unwrap(angle(dft_ys)))
title('c) phase spectrum of amplitude 2 V and frequency 100 Hz')
xlabel('frquency in Hz')
ylabel('phase')
xlim([-1000,1000])
```

2.2 Plots



2.3 Description & Observations

2.3.1 *Section A*

An analog continuos sine wave of amplitude 2 V and frequency 100 Hz is required. Hence I have used plot() function to generate a continuos signal over the large number of samples created. As seen, we have a sinusoidal wave of amplitude 2 V and time period of 0.01 seconds. Therefore frequency being, 100Hz

2.3.2 *Section B*

An amplitude spectrum for earlier signal is required. Since it's in frequency domain fourier transform is applied using fft() function. For better visual, fftshift() is applied to bring zero frequency component to center. Output obtained from abs() function is plotted against sampling frequency space. In output, 2 peaks are observed at -100 Hz and +100 Hz.

2.3.3 *Section C*

A phase spectrum for earlier signal is required. Since it's in frequency domain fourier transform is applied using fft() function. For better visual, fftshift() is applied to bring zero frequency component to center. Output obtained from angle() function is plotted against sampling frequency space.

I have observed in both spectrum plots that as sampling frequency is increased the plots get thinner and appear more narrow. Hence I've scaled the x-axis for better representation.

3 QUESTION 3

Power Spectrum of Noise

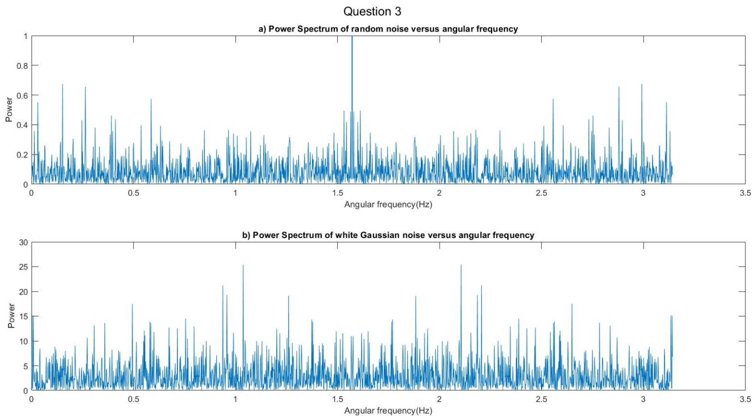
a) Generate a random noise element, N=2048 and plot the power spectrum of random noise versus angular frequency. (Take N equal spaced vector between 0 to pi for angular frequency & limit y-axis range between 0 to 1).

b) Generate a white Gaussian noise element, N=2048 of power 5dBW. Plot the power spectrum of random noise versus angular frequency. Also confirm that the power is approximately 3 watts, which is 5 dBW.

3.1 Code

```
close all, clc

sgtitle('Question 3')

N = 2048;

%Power Spectrum of random noise
x = rand(N,1);
w = linspace(0,pi,N);
x_fts = fftshift(fft(x));
Gn=(abs(x_fts).^2)/N;

subplot(2,1,1)
plot(w, Gn)
ylim([0,1]) %limit y-axis range between 0 to 1
title('a) Power Spectrum of random noise versus angular frequency')
xlabel('Angular frequency(Hz)')
ylabel('Power')

%Power Spectrum of white Gaussian noise
wGn = wgn(N,1,5);
wGn_fts = fftshift(fft(wGn));
y = (abs(wGn_fts).^2)/N;

subplot(2,1,2)
plot(w, y)
title('b) Power Spectrum of white Gaussian noise versus angular frequency')
xlabel('Angular frequency(Hz)')
ylabel('Power')

%Power display
display(var(wGn))
```

3.2 Plots



3.3 Description & Observations

3.3.1 *Section A*

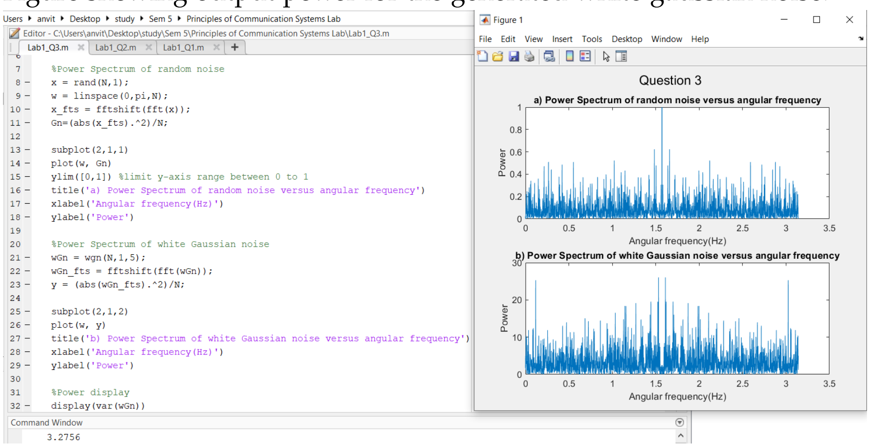
Random noise is generated by rand() function. For it's power spectrum the output is shown in code as Gn. w in the code is ω/ angular frequency.

3.3.2 *Section B*

White Gaussian noise for given power is generated using wgn() add-on function. y in the code is required spectrum. It is plotted against w as asked in the question.

The confirmation for output power's variance is shown in the terminal of matlab by using display() function. Since it is random noise, it is not same all time but varies near 3 watts.

Figure showing output power for the generated white gaussian noise:



4 REFERENCES

1)https://in.mathworks.com/matlabcentral/answers/100459-how-can-i-insert-a-title-over-a-group-of-subplots
2)https://in.mathworks.com/matlabcentral/answers/143588-convert-analog-to-digital
3)https://www.youtube.com/watch?v=1UFm75h65F4
4)https://in.mathworks.com/matlabcentral/answers/244720-vectors-must-be-the-same-length
5)https://www.youtube.com/watch?v=2IkdNsGQgEM
6)https://www.youtube.com/watch?v=LXw1uTE6Zv4
7)https://in.mathworks.com/help/matlab/ref/rand.htmld120e1015100
8)https://in.mathworks.com/help/signal/ug/power-spectral-density-estimates-using-fft.html
9)http://www.ece.northwestern.edu/local-apps/matlabhelp/techdoc/ref/linspace.html
10)https://www.youtube.com/watch?v=ety4nea5lvofeature=youtu.be
11)https://in.mathworks.com/help/matlab/ref/var.html
12)https://www.overleaf.com/latex/templates/highlighting-matlab-code-in-latex-with-mcode/nhtksndnsmmx