

NAME - SANTOSH - [CB.EN.V4CCE20053]

DEPARTMENT - COMPUTER AND COMMUNICATION ENGINEERING (CCE) A

LAB TITLE AND CODE : SIGNAL PROCESSING LAB 19CCE251

EXPERIMENT NUMBER : 8

DATE : 23/10/2024

FREQUENCY RESPONSE OF LTI SYSTEM

* AIM :

For a given LTI system characterized by an impulse response $h[n]$, determine the frequency response $H(\omega)$ and sketch the same.

* SOFTWARE REQUIRED :

Spyder IDE - Anaconda 3, 2021.05 (Python 3.8.8 64-bit)
Publisher - Anaconda, Inc.

* THEORY :

A system is characterized by its impulse $h[n]$ whose DTFT transform is -

$$H(\omega) = \sum_{n=-\infty}^{\infty} h(n) e^{-j\omega n}$$

And the inverse DTFT is -

$$h(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(\omega) e^{j\omega n} d\omega$$

$H(\omega)$ is called the frequency response or frequency characteristic of the system. It is the frequency characterization of the system whereas the impulse response is the time characterization.

FREQUENCY RESPONSE -

Now we use the time convolution property (or convolution theorem) to map the output $y(n)$ in time domain to its transform $Y(\omega)$ in the frequency domain -

$$y(n) = x(n) * h(n) \Leftrightarrow Y(\omega) = X(\omega) H(\omega)$$

②

$$\text{Or } H(\omega) = \frac{Y(\omega)}{X(\omega)}$$

The frequency response $H(\omega)$ is usually a complex quantity, so we write -

$$H(\omega) = H_R(\omega) + jH_I(\omega) = |H(\omega)| e^{j\phi(\omega)}$$

where

$$|H(\omega)| = \sqrt{H_R^2(\omega) + H_I^2(\omega)}$$

and

$$\phi(\omega) = \tan^{-1} \frac{H_I(\omega)}{H_R(\omega)}$$

$|H(\omega)|$ is the magnitude response and $\phi(\omega)$ is the phase response. If the impulse response $h(n)$ is real-valued, then -

$$|H(-\omega)| = |H(\omega)| \text{ and } \phi(-\omega) = -\phi(\omega)$$

The frequency response of a system exists if the system is BIBO (Bounded Input Bounded output) stable, i.e.

$$\text{Total energy } (\sum_{n=-\infty}^{\infty} |h(n)|^2) < \infty$$

* GRAPH PLOTTING ALGORITHM :

The following steps were followed -

- ① Define the x-axis and corresponding y-axis values as lists.
- ② Plot them on canvas using `plot()` function.
- ③ Give a name to x-axis and y-axis using `xlabel()` and `ylabel()` functions.
- ④ Give a title to your plot using the `title()` function.
- ⑤ Finally, to view your plot, we use the `.show()` function.

* PROGRAM WITH COMMENTS :

- ① Import numpy as np
- ② Import matplotlib.pyplot as plt
- ③
- ④ $n = np.arange(0, 10)$ # Get evenly spaced values within the interval [0, 10]

(3)

```

5 a = [] # u[n] Null List Declaration
6 for sample in range (len(n)):
7     if n[sample] >= 0:
8         temp = 1
9     else :
10        temp = 0
11    a.append (temp)
12 plt.subplot (5,1,1)
13 plt.xlabel ('n')
14 plt.ylabel ('u[n]')
15 plt.title ('Unit Step Discrete-Time Function for u[n]')
16 plt.stem (n,a)
17
18 b = [] # u[n-3] Null List Declaration
19 for sample in range (len(n)):
20     if n[sample] >= 3:
21         temp = 1
22     else :
23        temp = 0
24    b.append (temp)
25 plt.subplot (5,1,3)
26 plt.xlabel ('n')
27 plt.ylabel ('u[n-3]')
28 plt.title ('Unit Step Discrete-Time Function for u[n-3]')
29 plt.stem (n,b)
30
31
32 result = [] # u[n-3] - u[n] Null List Declaration
33 for sample in range (len(n)):
34     result.append (b[sample] - a[sample])
35 plt.subplot (5,1,5)
36 plt.xlabel ('n')
37 plt.ylabel ('u[n-3] - u[n]')

```

```

38 plt.title('Unit Step Discrete-Time Function for  $u[n-3] - u[n]$ ')
39 plt.stem(n, results)
40 plt.show()
41
42 temp = [] #  $(u[n-3] - u[n]) * 0.5^n$  Null List Declaration
43 for sample in range(len(n)):
44     temp.append((b[sample] - a[sample]) * 0.5 ** sample)
45 plt.xlabel('n')
46 plt.ylabel('( $u[n-3] - u[n]$ ) * 0.5^n')
47 plt.title('Impulse Response h[n]')
48 plt.stem(n, temp)
49 plt.show()
50
51 # Compute Frequency Response
52 frequency_response = []
53 def summation(f):
54     sum = 0
55     for k in range(len(temp)):
56         exponential = np.exp(-2j * np.pi * k * f)
57         sum = sum + float(temp[k]) * exponential
58     return sum
59
60 for w in range(len(temp)):
61     frequency_response.append(summation(w))
62
63 # Sketch Frequency Response
64 plt.xlabel('w')
65 plt.ylabel('h(w)')
66 plt.title("Frequency Response")
67 plt.plot(np.arange(0, len(frequency_response)),
68          np.real(frequency_response))
69 plt.show()

```

print("The Frequency Response of $h[n]$ is as follows :")
 format(frequency_response))

⑤

```

70
71 # Compute Magnitude and Phase spectrum
72 magnitude_spectrum = []
73 phase_spectrum = []
74
75 for sample in range(len(frequency_response)):
76     magnitude_spectrum.append([frequency_response[sample].real ** 2 + frequency_response[sample].imag ** 2] ** 0.5)
77     phase = np.arctan(frequency_response[sample].imag / frequency_response[sample].real)
78     phase_spectrum.append(phase)
79
80 # Sketch Magnitude Spectrum
81 plt.xlabel('w')
82 plt.ylabel('|h(w)|')
83 plt.title("Magnitude spectrum : ... {}... ".format(magnitude_spectrum))
84 plt.plot(np.arange(0, len(magnitude_spectrum)), magnitude_spectrum)
85 plt.show()
86 print("The Magnitude spectrum of h[n] is as follows : {}".
87       format(magnitude_spectrum))
88
89 # Sketch Phase Spectrum
90 plt.xlabel('w')
91 plt.ylabel('Angle (in radians)')
92 plt.title("Phase spectrum \phi(w)")
93 plt.plot(np.arange(0, len(phase_spectrum)), phase_spectrum)
94 plt.show()
95 print("The Phase spectrum of h[n] is as follows : {}".
96       format(phase_spectrum))

```

* INFEERENCE :

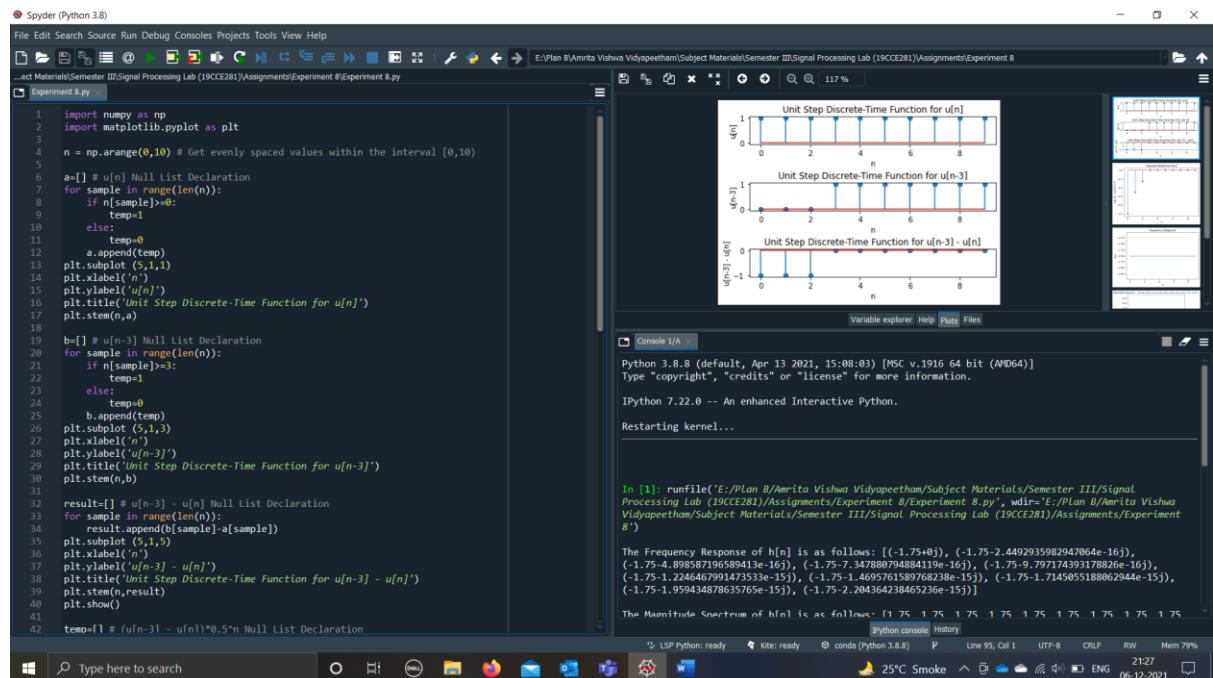
For the given LTI system characterized by the Impulse response $h[n]$, compute frequency response $H(e^{j\omega})$ and plot the same along with magnitude and phase spectrums.

RESULTS

VERIFIED

(i) Frequency response magnitude and phase plots
 (ii) Magnitude spectrum and phase spectrum

Unit Step Discrete-Time Functions



The screenshot shows the Spyder Python IDE interface. The code in the editor window is as follows:

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 n = np.arange(0, 10) # Get evenly spaced values within the interval [0,10)
5
6 a=[ ] # u[n] Null List Declaration
7 for sample in range(len(n)):
8     if n[sample]>=0:
9         temp=1
10    else:
11        temp=0
12    a.append(temp)
13 plt.subplot (5,1,1)
14 plt.xlabel('n')
15 plt.ylabel('u[n]')
16 plt.title('Unit Step Discrete-Time Function for u[n]')
17 plt.stem(n,a)
18
19 b=[ ] # u[n-3] Null List Declaration
20 for sample in range(len(n)):
21     if n[sample]>=3:
22         temp=1
23     else:
24         temp=0
25     b.append(temp)
26 plt.subplot (5,1,3)
27 plt.xlabel('n')
28 plt.ylabel('u[n-3]')
29 plt.title('Unit Step Discrete-Time Function for u[n-3]')
30 plt.stem(n,b)
31
32 result=[ ] # u[n-3] - u[n] Null List Declaration
33 for sample in range(len(n)):
34     result.append(b[sample]-a[sample])
35 plt.subplot (5,1,5)
36 plt.xlabel('n')
37 plt.ylabel('u[n-3] - u[n]')
38 plt.title('Unit Step Discrete-Time Function for u[n-3] - u[n]')
39 plt.stem(n,result)
40 plt.show()
41
42 temp=[ ] # (u[n-3] - u[n])*0.5^n Null List Declaration

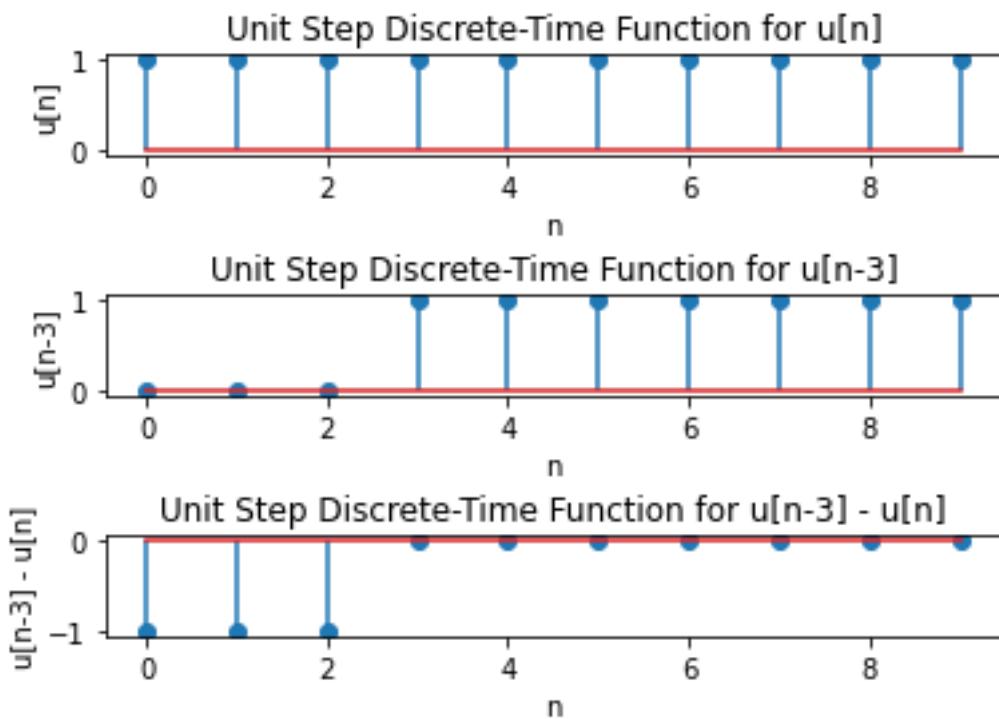
```

The console window shows the Python environment and the magnitude spectrum of $h[n]$:

```

In [1]: runfile('E:/Plan B/Amrita Vishwa Vidyapeetham/Subject Materials/Semester III/Signal Processing Lab (19CCE281)/Assignments/Experiment 8/Experiment 8.py', wdir='E:/Plan B/Amrita Vishwa Vidyapeetham/Subject Materials/Semester III/Signal Processing Lab (19CCE281)/Assignments/Experiment 8')
The Frequency Response of h[n] is as follows: [(-1.75+0j), (-1.75-2.4492935982947064e-16j),
(-1.75-4.898587196589413e-16j), (-1.75-7.347880794884119e-16j), (-1.75-9.797174393178826e-15j),
(-1.75-1.2246467991473533e-15j), (-1.75-1.4695761989768238e-15j), (-1.75-1.7145055188062944e-15j),
(-1.75-1.959434678635765e-15j), (-1.75-2.204364238465526e-15j)]
The Magnitude Spectrum of h[n] is as follows: [1 2.5 1.75 1.75 1.75 1.75 1.75 1.75 1.75]

```

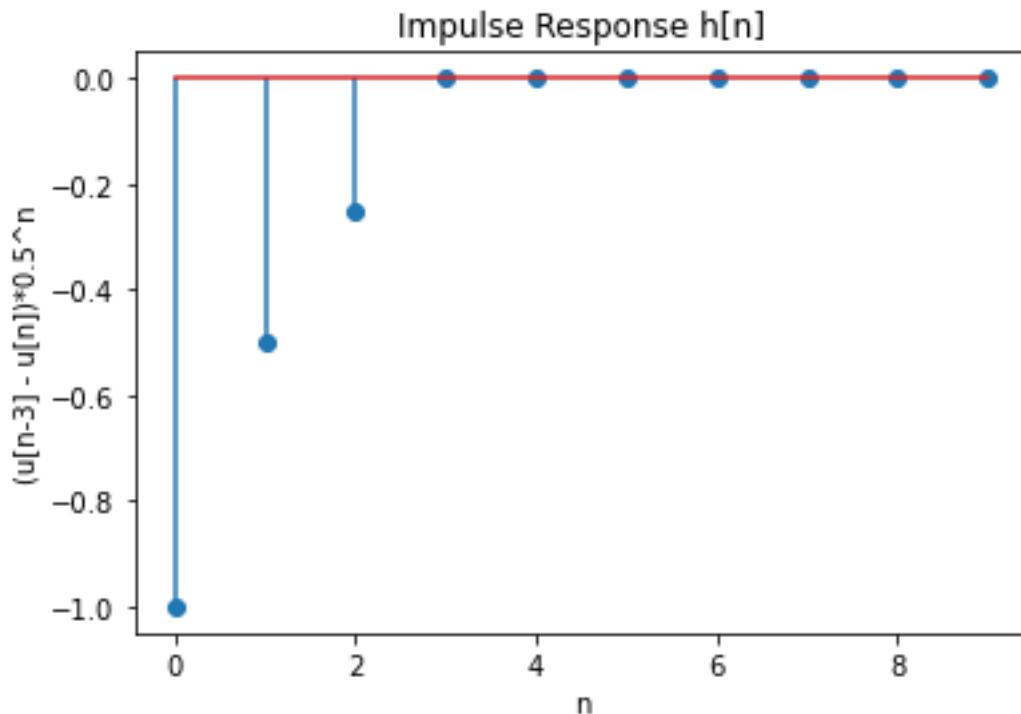
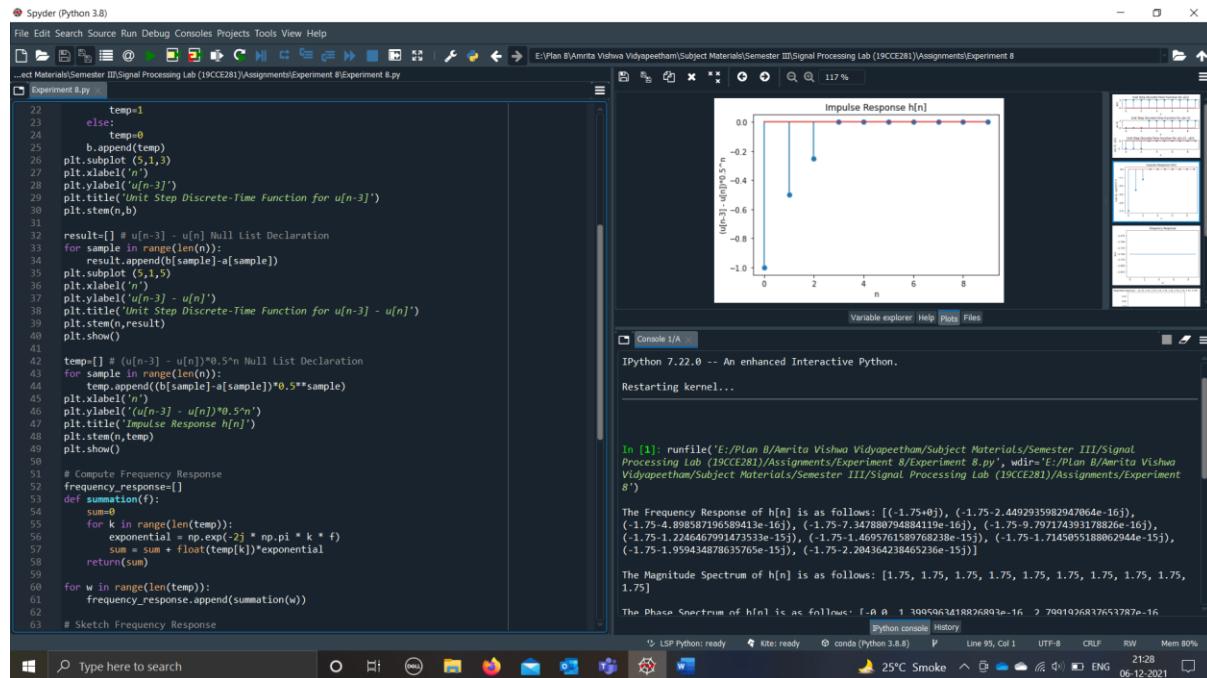


Step 1: Import library source files, numpy and matplotlib.pyplot.

Step 2: Get evenly spaced values within the interval [0,10].

Step 3: Compute $u[n]$ and $u[n-3]$ followed by $(u[n-3] - u[n])$ and finally, $(u[n-3] - u[n]) * 0.5^n$

Step 4: Sketch the plot obtained for each function in Step 3.

Given Impulse Response $h[n]$ 

Frequency Response – $h(\omega)$

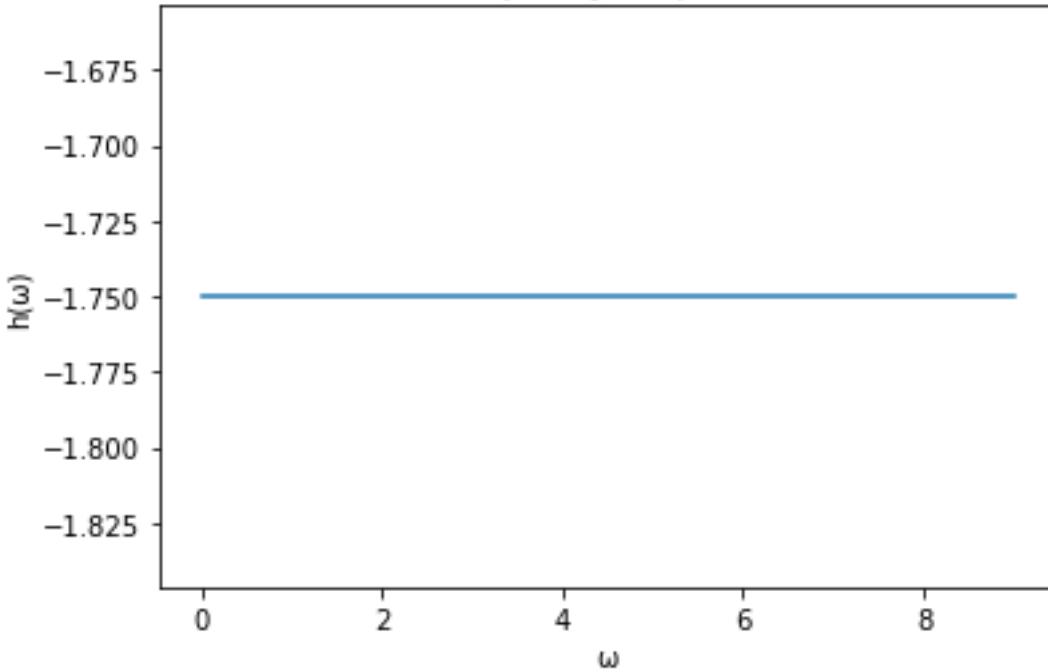
The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays a script named 'Experiment 8.py' containing Python code for signal processing tasks. The code includes functions for calculating impulse responses, frequency responses, magnitude spectra, and phase spectra. On the right, the execution results are shown in several windows: a plot titled 'Frequency Response' showing a constant value of -1.750 across the range of omega from 0 to 8; a 'Variable explorer' window; a 'Console 1/A' window showing the output of IPython 7.22.0 and the results of the calculations; and a 'Python console' window at the bottom.

```

40 plt.show()
41
42 temp=[] # (u[n-3] - u[n])*0.5*n Null List Declaration
43 for sample in range(len(n)):
44     temp.append((b[sample]-a[sample])*0.5**sample)
45 plt.xlabel('n')
46 plt.ylabel('(u[n-3] - u[n])*0.5^n')
47 plt.title('Impulse Response h[n]')
48 plt.stem(n,temp)
49 plt.show()
50
51 # Compute Frequency Response
52 frequency_response=[]
53 def summation(f):
54     sum=0
55     for k in range(len(temp)):
56         exponential = np.exp(-2j * np.pi * k * f)
57         sum = sum + float(temp[k])*exponential
58     return(sum)
59
60 for w in range(len(temp)):
61     frequency_response.append(summation(w))
62
63 # Sketch Frequency Response
64 plt.xlabel('w')
65 plt.ylabel('h(w)')
66 plt.title("Frequency Response")
67 plt.plot(np.arange(0, len(frequency_response)), np.real(frequency_response))
68 plt.show()
69 print("The Frequency Response of h[n] is as follows: {}".format(frequency_response))
70
71 # Compute Magnitude and Phase Spectrum
72 magnitude_spectrum = []
73 phase_spectrum = []
74
75 for sample in range(len(frequency_response)):
76     magnitude_spectrum.append((frequency_response[sample].real)**2 +
77     frequency_response[sample].imag**2)**0.5
78     phase = np.arctan(frequency_response[sample].imag/frequency_response[sample].real)
79     phase_spectrum.append(phase)
80
81 # Sketch Magnitude Spectrum

```

Frequency Response



Step 1: Compute frequency response for $h[n]$.

Step 2: Print and sketch the frequency response, $h(\omega)$.

Step 3: Compute magnitude and phase spectrum for $h[n]$.

Step 4: Print and sketch magnitude spectrum, $|h(\omega)|$.

Step 5: Print and sketch phase spectrum, $\Phi(\omega)$.

Magnitude Spectrum – $|h(\omega)|$ & Phase Spectrum – $\Phi(\omega)$

The screenshot shows the Spyder Python IDE interface. The code in the editor window is as follows:

```

56     exponential = np.exp(-2j * np.pi * k * f)
57     sum = sum + float(temp[k])*exponential
58   return(sum)
59
60 for w in range(len(temp)):
61   frequency_response.append(summation(w))
62
63 # Sketch Frequency Response
64 plt.xlabel('w')
65 plt.ylabel('h(w)')
66 plt.title("Frequency Response")
67 plt.plot(np.arange(0,len(frequency_response)),np.real(frequency_response))
68 plt.show()
69 print("\nThe Frequency Response of h[n] is as follows: {}".format(frequency_response))
70
71 # Compute Magnitude and Phase Spectrum
72 magnitude_spectrum = []
73 phase_spectrum = []
74
75 for sample in range(len(frequency_response)):
76   magnitude_spectrum.append(np.abs(frequency_response[sample].real)**2 +
77     frequency_response[sample].imag**2)**0.5
78   phase = np.arctan(frequency_response[sample].imag/frequency_response[sample].real)
79   phase_spectrum.append(phase)
80
81 # Sketch Magnitude Spectrum
82 plt.xlabel('w')
83 plt.ylabel('Angle (in radians)')
84 plt.title("Magnitude Spectrum: ...".format(magnitude_spectrum))
85 plt.plot(np.arange(0,len(magnitude_spectrum)),magnitude_spectrum)
86 plt.show()
87 print("\n\nThe Magnitude Spectrum of h[n] is as follows: {}".format(magnitude_spectrum))
88
89 # Sketch Phase Spectrum
90 plt.xlabel('w')
91 plt.ylabel('Angle (in radians)')
92 plt.title("Phase Spectrum Φ(w)")
93 plt.plot(np.arange(0,len(phase_spectrum)),phase_spectrum)
94 plt.show()
95 print("\n\nThe Phase Spectrum of h[n] is as follows: {}".format(phase_spectrum))

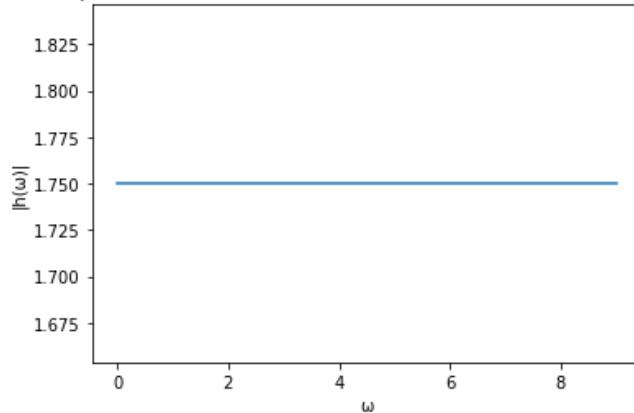
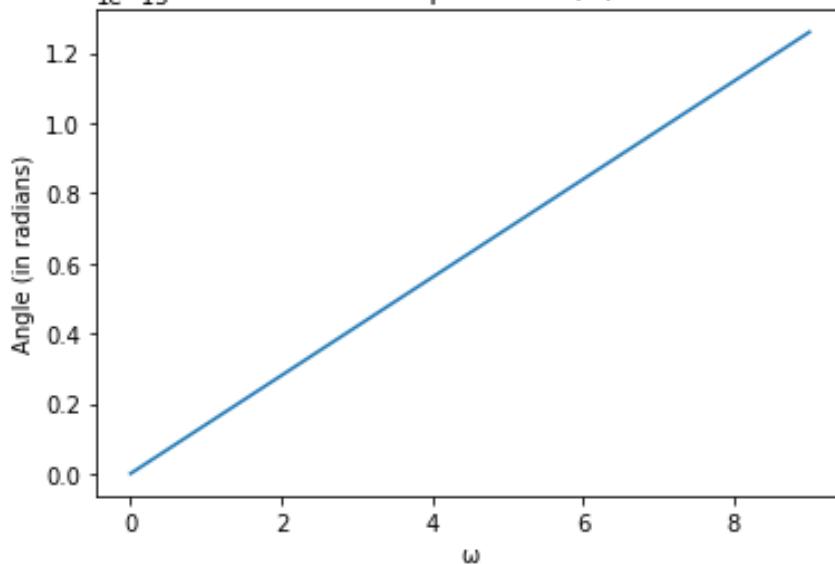
```

The plot titled "Frequency Response" shows a constant value of approximately 1.75 across the range of omega from 0 to 8.

The plot titled "Phase Spectrum $\Phi(\omega)$ " shows a linear increase from 0 to approximately 1.25 radians over the same range of omega.

The console output provides the numerical values for the Frequency Response, Magnitude Spectrum, and Phase Spectrum.

Magnitude Spectrum: ...[1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75]...

Phase Spectrum $\Phi(\omega)$ 

```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]  
Type "copyright", "credits" or "license" for more information.
```

```
IPython 7.29.0 -- An enhanced Interactive Python.
```

```
Restarting kernel...
```

```
In [1]:      'E:/Plan B/Amrita Vishwa Vidyapeetham/Subject Materials/Semester III/  
Signal Processing Lab (19CCE281)/Assignments/Experiment 8/Experiment 8.py'      = 'E:/Plan  
B/Amrita Vishwa Vidyapeetham/Subject Materials/Semester III/Signal Processing Lab  
(19CCE281)/Assignments/Experiment 8'
```

The Frequency Response of $h[n]$ is as follows: $[-1.75+0j, -1.75-2.4492935982947064e-16j, -1.75-4.898587196589413e-16j, -1.75-7.347880794884119e-16j, -1.75-9.797174393178826e-16j, -1.75-1.2246467991473533e-15j, -1.75-1.4695761589768238e-15j, -1.75-1.7145055188062944e-15j, -1.75-1.959434878635765e-15j, -1.75-2.204364238465236e-15j]$

The Magnitude Spectrum of $h[n]$ is as follows: $[1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75, 1.75]$

The Phase Spectrum of $h[n]$ is as follows: $[-0.0, 1.3995963418826893e-16, 2.7991926837653787e-16, 4.198789025648068e-16, 5.598385367530757e-16, 6.997981709413447e-16, 8.397578051296136e-16, 9.797174393178826e-16, 1.1196770735061515e-15, 1.2596367076944204e-15]$

```
In [2]:
```