NAME - SANTOSH - [CB.EN.U4CCE20053]
DEPARTMENT - COMPUTER AND COMMUNICATION ENGINEERING (CCE)

LAB TITLE AND CODE - SIGNAL PROCESSING LAB 19CCE281
EXPERIMENT NUMBER - 10
DATE - 28/12/2021

DESIGN OF FIR FILTER

* **AIM :**
For a given frequency response $H(e^{j\omega})$, design a Finite Impulse Response (FIR) low pass filter using rectangular and Hanning window techniques and determine the impulse and frequency response.

* **SOFTWARE REQUIRED :**
Spyder IDE - Anaconda 3 2021.11 (Python 3.9.7 64-bit)
Publisher - Anaconda, Inc.

* **THEORY :**
FIR filters are filters having a transfer function of a polynomial in z and is an all-zero filter in the sense that the zeroes in the z-plane determine the frequency response magnitude characteristic. The z transform of a N-point FIR filter is given by -

$$H(z) = \sum_{n=0}^{N-1} h(n)\, z^{-n}$$

FIR filters are particularly useful for applications where exact phase response is required. The FIR filter is generally implemented in a non-recursive way which guarantees a stable filter.

PTO

→ **RECTANGULAR WINDOW METHOD:-**

In this method, from the desired frequency response specification $H_d(w)$, corresponding unit sample response $h_d(n)$ is determined using the following relation -

$$h_d(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(w)\, e^{jwn}\, dw$$

where, $H_d(w) = \sum_{n=-\infty}^{\infty} h_d(n)\, e^{-jwn}$

In general, unit sample response $h_d(n)$ obtained from the above relation is infinite in duration, so it must be truncated at some point say $n = M-1$ to yield an FIR filter of length M (i.e. 0 to M-1). This truncation of $h_d(n)$ to length M is same as multiplying $h_d(n)$ by the rectangular window defined as -

$$w(n) = \begin{cases} 1, & 0 \le n \le M-1 \\ 0, & \text{otherwise} \end{cases}$$

Thus, the unit sample response of the FIR filter becomes -

$$h(n) = h_d(n)\, w(n) = \begin{cases} h_d(n), & 0 \le n \le M-1 \\ 0, & \text{otherwise} \end{cases}$$

Now, the multiplication of the window function $w(n)$ with $h_d(n)$ is equivalent to convolution of $H_d(w)$ with $W(w)$, where $W(w)$ is the frequency domain representation of the window function -

$$W_d(w) = \sum_{n=0}^{M-1} w(n)\, e^{-jwn}$$

Thus, the convolution of $H_d(w)$ with $W(w)$ yields the frequency response of the truncated FIR filter -

$$H(w) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(v)\, W(w-v)\, dw$$

The frequency response can also be obtained using the following relation -

$$H(w) = \sum_{n=0}^{M-1} h(n)\, e^{-jwn}$$

→ HANNING WINDOW METHOD:-

Similar to rectangular window,

$$w[n] = \sum_{k=0}^{K} (-1)^k a_k \cos\left(\frac{2\pi kn}{N}\right), \quad 0 \le n \le N.$$

In most cases, including the given question, all coefficients $a_k \ge 0$. These windows have only $2k+1$ non-zero N-point DFT coefficients.

The customary cosine-sin windows for case $K=1$ have the form -

$$w[n] = a_0 - \underbrace{(1-a_0)}_{a_1} \cdot \cos\left(\frac{2\pi n}{N}\right), \quad 0 \le n \le N$$

which is easily (and often) confused with its zero-phase version -

$$w_0(n) = w\left(n + \frac{N}{2}\right)$$

$$= a_0 + a_1 \cdot \cos\left(\frac{2\pi n}{N}\right), \quad -\frac{N}{2} \le n \le \frac{N}{2}$$

Setting $a_0 = 0.5$ produces a Hann window -

$$w[n] = 0.5\left[1 - \cos\left(\frac{2\pi n}{N}\right)\right] = \sin^2\left(\frac{\pi n}{N}\right)$$

* GRAPH PLOTTING ALGORITHM:

The following steps were followed -

1. Define the x-axis and corresponding y-axis 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.
6. To set the visibility of the grid inside the figure to on, we use the .grid() function.

* THEORETICAL CALCULATION:

Given -

Number of coefficients, $N = 9$ (also, indicated by M)

Frequency response, $H(e^{j\omega}) = \begin{cases} 1, & -2\pi/5 \le \omega \le 2\pi/5 \\ 0, & \text{otherwise} \end{cases}$

Desired filter coefficient ($h_d[n]$) is obtained by taking Inverse Discrete Time Fourier Transform (IDTFT) on $H(e^{j\omega})$ and yields $h_d[n] = \dfrac{\sin\left(\frac{2\pi n}{5}\right)}{\pi n}$

$\Rightarrow$ for $n=0$, $h_d[0] = \frac{1}{2\pi}\int_{-\pi}^{\pi} 1 \times e^{j\omega n}\, d\omega = 0.4$

similarly, for $n=1$, $h_d[1] = 0.3027$

for $n=2$, $h_d[2] = 0.0935$

for $n=3$, $h_d[3] = -0.06236$

for $n=4$, $h_d[4] = -0.0757$

for $n=5$, $h_d[5] = -0.06236$

for $n=6$, $h_d[6] = 0.0935$

(5)

for $n=7$, $h_d[7] = 0.3027$
for $n=8$, $h_d[8] = 0.4$

① RECTANGULAR WINDOW TECHNIQUE:-

Expected filter coefficients ($h[n]$) on multiplying with rectangular window function ($w[n]$) yields the following -

$$w[n] = \begin{cases} 1, & 0 \le n \le M-1 = 0 \le n \le 8 \\ 0, & \text{otherwise} \end{cases}$$

$$h[n] = h_d[n] \times w[n]$$

$$= \begin{cases} h_d[n], & 0 \le n \le 8 \\ 0, & \text{otherwise} \end{cases}$$

Determine the frequency response $H(e^{j\omega})$ by direct substitution of the expression (even symmetry and length is odd integer) -

$$H(\omega) = H_R(\omega)\, e^{-j\omega\left(\frac{M-1}{2}\right)} = H_R(\omega)\, e^{-j\omega\left(\frac{8}{2}\right)}, \quad \left(\frac{8}{2}\right) = 4$$

where $H_R(\omega)$ is a real function of ω and can be expressed as -

$$H_R(\omega) = h\left(\frac{M-1}{2}\right) + 2 \sum_{n=0}^{(M-3)/2} h(n) \cos\left(\frac{M-1}{2} - n\right) ; \text{ when } M \text{ is odd}$$

and $$H_R(\omega) = 2 \sum_{n=0}^{(M/2)-1} h(n) \cos\omega\left(\frac{M-1}{2} - n\right) ; \text{ when } M \text{ is even}$$

Taking ω values from 0 to 180°, with a difference of 10° at regular intervals, we obtain 18 values corresponding for $H(\omega)$, which is converted to decibels by -

$$\text{Decibel Representation} = 20 \times \log_{10} |H(\omega)|$$

→ HANNING WINDOW TECHNIQUE:-

Expected filter coefficients ($h_d[n]$) on multiplying with Hanning window function ($w[n]$) yields the following -

$$w[n] = 0.5\left(1 - \cos\left(\frac{2\pi n}{m-1}\right)\right) = 0.5\left(1 - \cos\left(\frac{2\pi n}{8}\right)\right)$$

$$h[n] = h_d[n] \times w[n]$$

$$h[n] = \begin{cases} 0.00, & \text{for } n=0 \\ 0.0443, & \text{for } n=1 \\ 0.0468, & \text{for } n=2 \\ -0.0532, & \text{for } n=3 \\ -0.0757, & \text{for } n=4 \\ -0.0532, & \text{for } n=5 \\ 0.468, & \text{for } n=6 \\ 0.0443, & \text{for } n=7 \\ 0.00, & \text{for } n=8 \\ 0, & \text{otherwise} \end{cases}$$

Determine the frequency response, similar to rectangular window technique.

* PROGRAM WITH COMMENTS:

```
1 import matplotlib.pyplot as plt # Provides an implicit way of plotting
2 import numpy as np # Support for large, multi-dimensional arrays and matrices
3 from scipy import integrate # Compute a definite integral
4
5 N= int(input("\n Enter the number of coefficients for FIR filter"))
6 hd_w = int(input(" Enter the frequency response value : "))
```

```
7
8  # Inverse discrete-time Fourier transform:
9  hd_n = []
10 for n in range (N//2 + 1):
11     expression = lambda w: Hd_w * np.exp (1j * w * n)
   # Expression within the integral
12     temp= integrate.quad (expression, (-2 * np.pi)/5,
   (-2 * np.pi)/5)
13     integral = temp[0] / (2 * np.pi)  # quad() gives a tuple,
   integral_value[0] + constant[1]
14     hd_n.append (integral)
15
16 # Compute desired filter coefficients:
17 for n in reverse (range (N//2)): # Backward iteration will start
   occurring from the rear
18     symmetric = hd_n[n]
19     hd_n.append (symmetric)
20
21 print ("\n The desired filter coefficients h_d[n] is as follows: \n
   {}".format (hd_n))
22
23 # Design FIR filter using rectangular window technique
   (Impulse Response):
24
25 # Rectangular window function:
26 wn_rectangular = []
27 for n in range (N):
28     if n<N:
29         temp = 1
30     else:
31         temp=0
32     wn_rectangular.append (temp)
33
```

SANTOSH - [CB.EN.U4ECE20053]

```
34 # Expected filter coefficients obtained using rectangular
   window technique:
35 hn_rectangular = []
36 for n in range(N):
37     hn_rectangular.append(hd_n[n] * wn_rectangular[n])
38
39 # sketch expected filter coefficients obtained using
   rectangular window technique:
40 plt.xlabel('n')
41 plt.ylabel('magnitude of h[n]')
42 plt.title("Impulse Response Using Rectangular Window
   Technique")
43 plt.stem(np.arange(0, len(hn_rectangular)), hn_rectangul
44 plt.grid(True)
45 plt.show()
46 print("\n The expected filter coefficients h[n] obtained using
   rectangular window technique is as follows: \n {}".
   format(hn_rectangular))
47
48 # Design FIR filter using rectangular window technique
   (Frequency Response):
49
50 hd_w_rectangular = [] # Real function of ω
51 h_w_rectangular = [] # Frequency response (magnitude)
52 h_w_rectangular_decibels = [] # Frequency response (decibel
53 index_rectangular = 0 # Index variable to iterate through
   "hd_w_rectangular" list
54 x_axis_rectangular = [] # Define the angle axis
55
56 for degrees in range(0, 190, 10):
57
58     w_rectangular = degrees * (np.pi/180)
```

```
    sum_rectangular = 0

    if (N % 2 == 1):

        for n in range ((N-3)//2) + 1):
            summation = hn_rectangular[n] *
np.cos(w_rectangular) * ((N-1)//2) - n)
            sum_rectangular = sum_rectangular + summation
        hr_w_rectangular.append(hn_rectangular[(N-1)//2] +
(2 * sum_rectangular))

    else:

        for n in range((N//2) - 1):
            summation = hn_rectangular[n] *
np.cos(w_rectangular) * ((N-1)//2) - n)
            sum_rectangular = sum_rectangular + summation
        hr_w_rectangular.append(2 * sum_rectangular)

    h_w_rectangular.append(hr_w_rectangular[index_
rectangular] * np.exp(-1j * w_rectangular * ((N-1)//2)))

    h_w_rectangular_decibels.append(20 * np.log10(abs(
h_w_rectangular[index_rectangular])))

    index_rectangular = index_rectangular + 1

    x_axis_rectangular.append(degrees)

# sketch frequency response obtained using rectangular
window technique:
plt.xlabel('w in degrees')
plt.ylabel('magnitude of H[w]')
```

```
plt.title("Frequency Response Using Rectangular Window Technique")
plt.plot(x_axis, h_w_rectangular_decibels)
plt.grid(True)
plt.show()
print("\n The frequency response H[w] obtained using rectangular window technique is as follows: \n\n",
format(h_w_rectangular_decibels)

# Design FIR filter using Hanning window technique (Impulse Response):

# Hanning window function:
wn_hanning = []
for n in range(N):
    wn_hanning.append(0.5 * (1 - np.cos((2 * np.pi * n)/(N-1))))

# expected filter coefficients obtained using Hanning window technique:
hn_hanning = []
for n in range(N):
    hn_hanning.append(hd_n[n] * wn_hanning[n])

# Sketch expected filter coefficients obtained using Hanning window technique:
plt.xlabel('n')
plt.ylabel('magnitude of h[n]')
plt.title("Impulse Response using Hanning Window Technique")
plt.stem(np.arange(0, len(hn_hanning)), hn_hanning)
plt.grid(True)
plt.show()
```

```
print("\n The expected filter coefficients h[n] obtained using
Hanning window technique is as follows: \n {}".
format(hn_hanning))

# Design FIR filter using Hanning window technique
(Frequency Response):

hn_w_hanning = [] # Real function of w
h_w_hanning = [] # Frequency response in magnitude
h_w_hanning_decibels = [] # Frequency response (decibels)
index_hanning=0 # Index variable to iterate through
'hn_w_hanning' list
x_axis_hanning = [] # Define the angle axis

for degrees in range(0,190,10):

    w_hanning = degrees * (np.pi/180)
    sum_hanning =0

    if (N%2 ==1):

        for n in range(((N-3)/2) + 1):
            summation = hn_hanning[n] * np.cos(w_hanning) *
((N-1)//2) -n)
            sum_hanning = sum_hanning + summation
        hn_w_hanning.append(hn_hanning[(N-1)//2] + (2 *
sum_hanning))

    else:

        for n in range((N-2)-1):
            summation = hn_hanning[n] * np.cos(w_hanning) *
((N-1)//2) -n)
```

```
137         sum_hanning = sum_hanning + summation
138         hr_w_hanning.append(2 * sum_hanning)
139
140     h_w_hanning.append(hr_w_hanning[index_hanning] *
    np.exp(-1j * w_hanning * ((N-1)//2)))
141
142     h_w_hanning_decibels.append(20 * np.log10(abs
    (h_w_hanning[index_hanning])))
143
144     index_hanning = index_hanning + 1
145
146     x_axis_hanning.append(degrees)
147
148 # sketch frequency response obtained using Hanning window
    technique:
149 plt.xlabel('w in Degrees')
150 plt.ylabel('H[w] in Decibels')
151 plt.title("Frequency Response using Rectangular Window
    Technique")
152 plt.plot(x_axis_hanning, h_w_hanning_decibels)
153 plt.grid(True)
154 plt.show()
155 print("\n The frequency response H[w] obtained using
    Hanning window technique is as follows: \n {}".
    format(h_w_hanning_decibels))
156
```

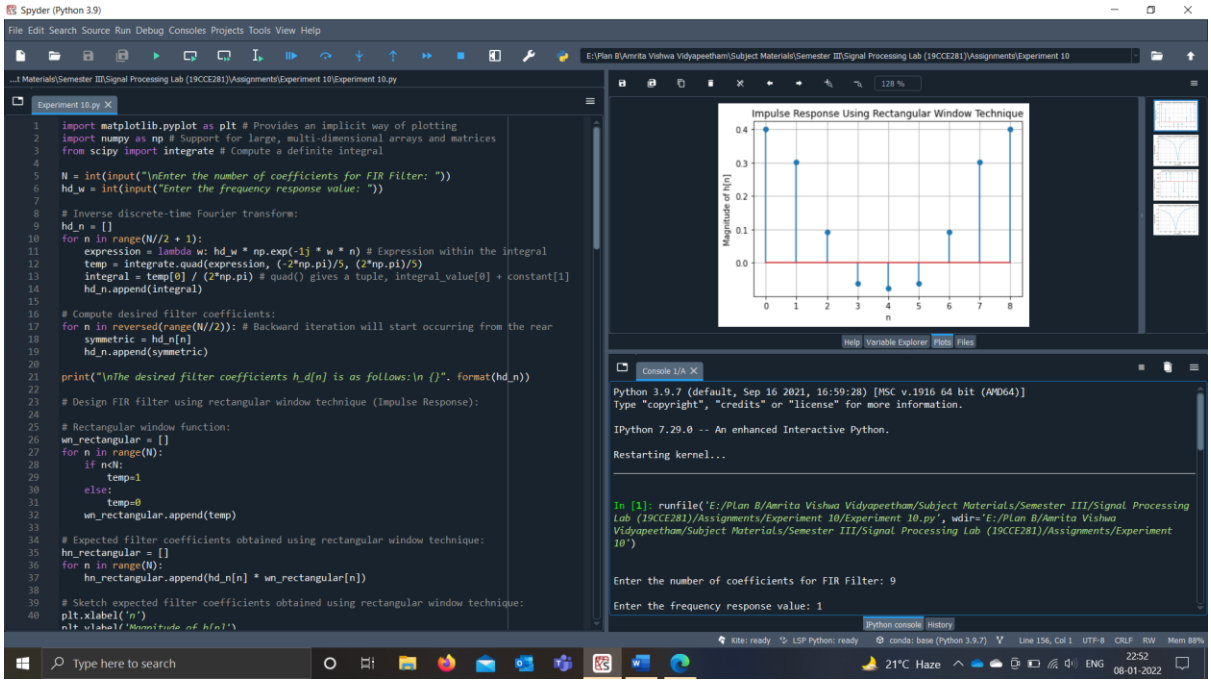
* INFERENCE:

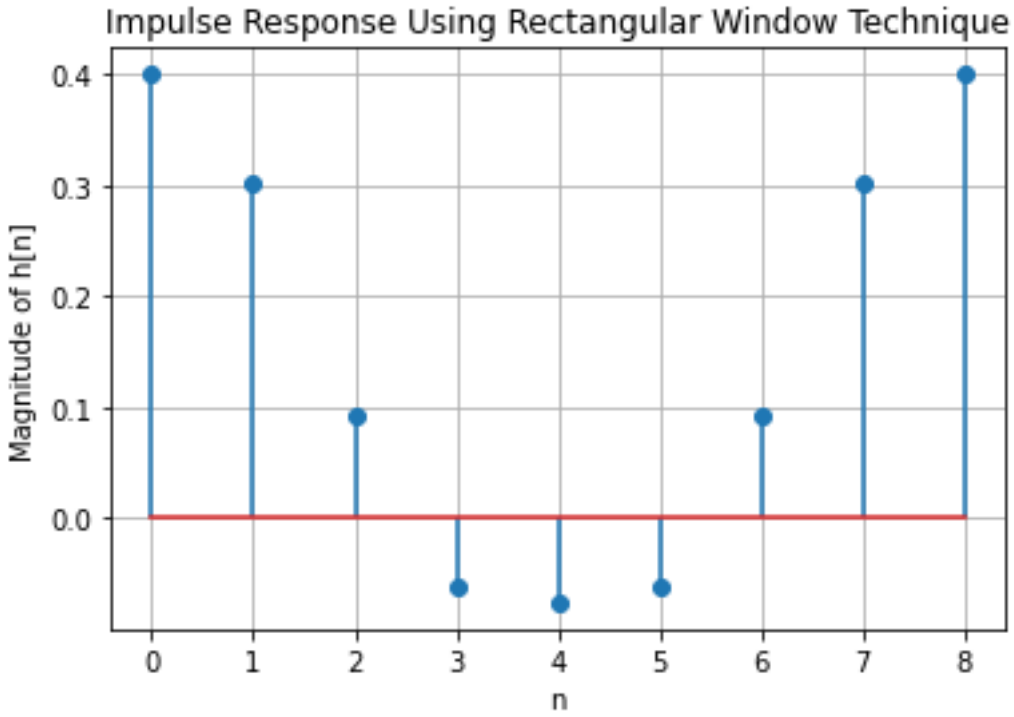
Compute the impulse and frequency response for a FIR low pass filter using rectangular and Hanning window techniques.

RESULTS

VERIFIED.

Impulse Response Using Rectangular Window Technique





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

Step 2: Enter the number of coefficients for the FIR filter along with the frequency response value.

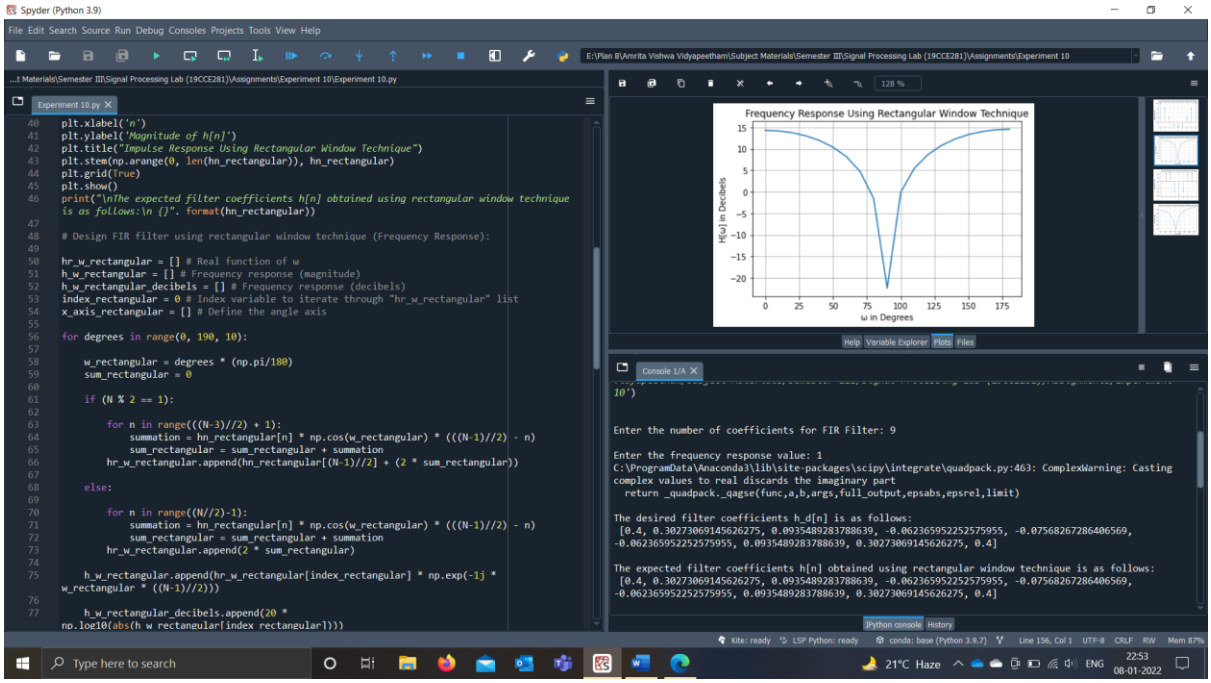
Step 3: Calculate inverse discrete-time Fourier transform and compute desired filter coefficients.

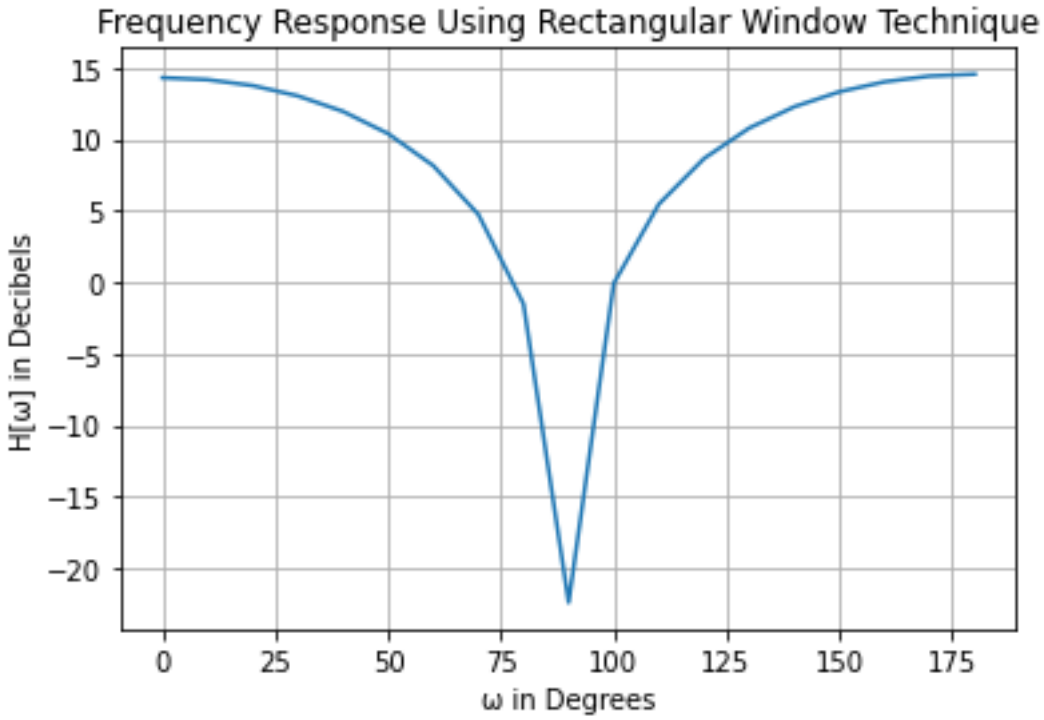
Step 4: Design FIR filter using rectangular window technique (Impulse Response).

Step 5: Compute the rectangular window function.

Step 6: Evaluate expected filter coefficients obtained using the rectangular window technique and sketch the same.

Frequency Response Using Rectangular Window Technique



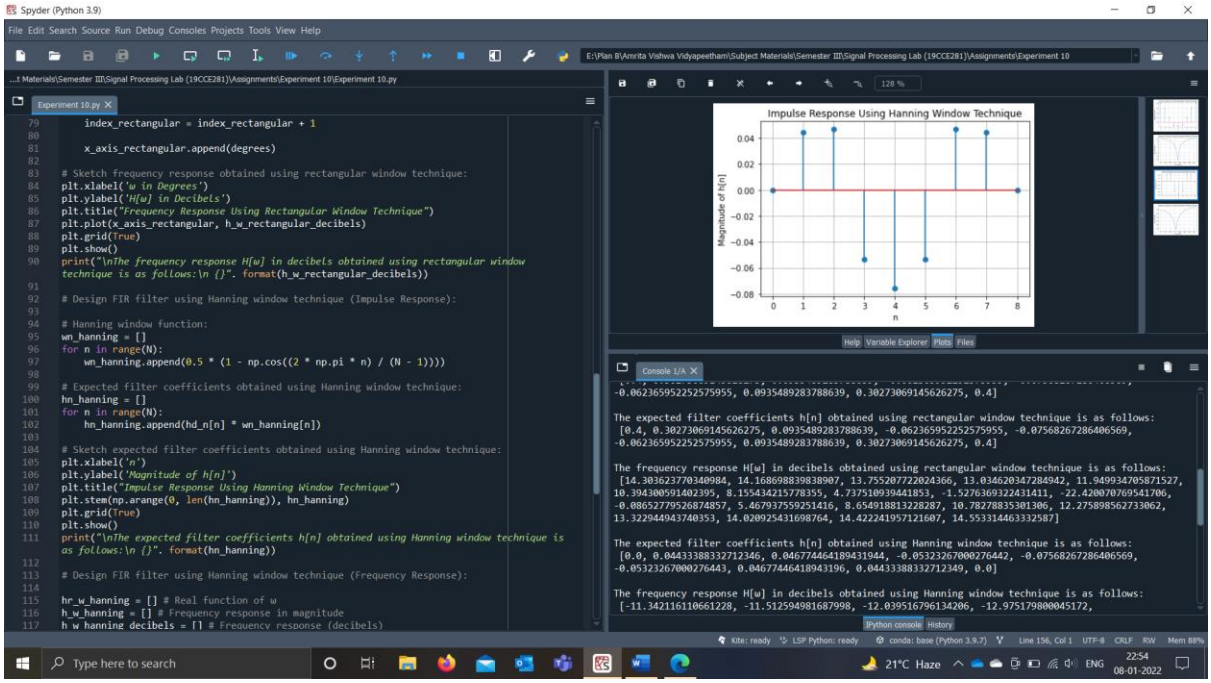


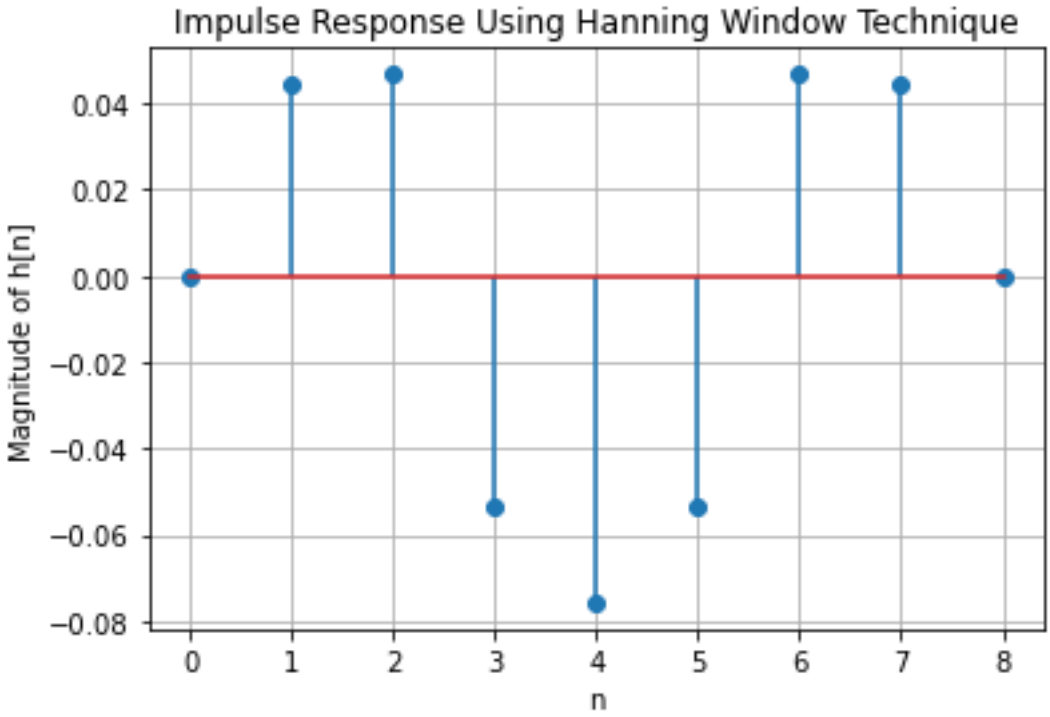
Step 1: Design FIR filter using rectangular window technique (Frequency Response).

Step 2: Initialize the arrays for the real function of ω, frequency response (magnitude), frequency response (decibels) and define the angle axis.

Step 3: Sketch frequency response obtained using rectangular window technique.

Impulse Response Using Hanning Window Technique



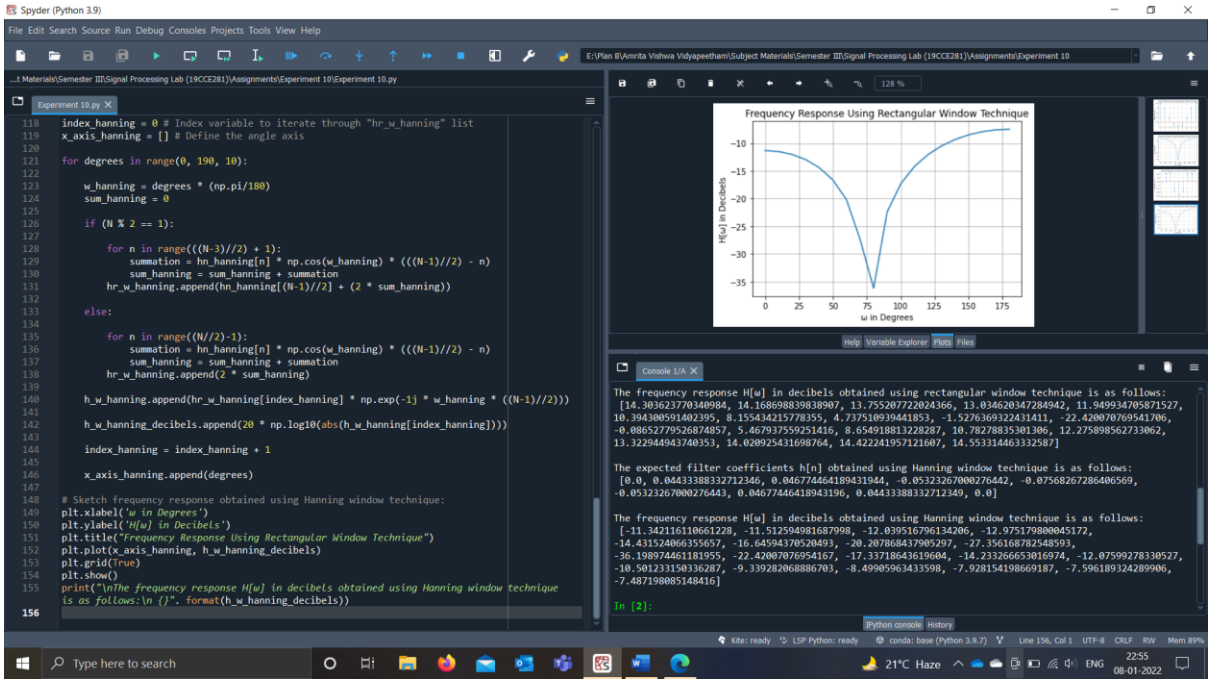


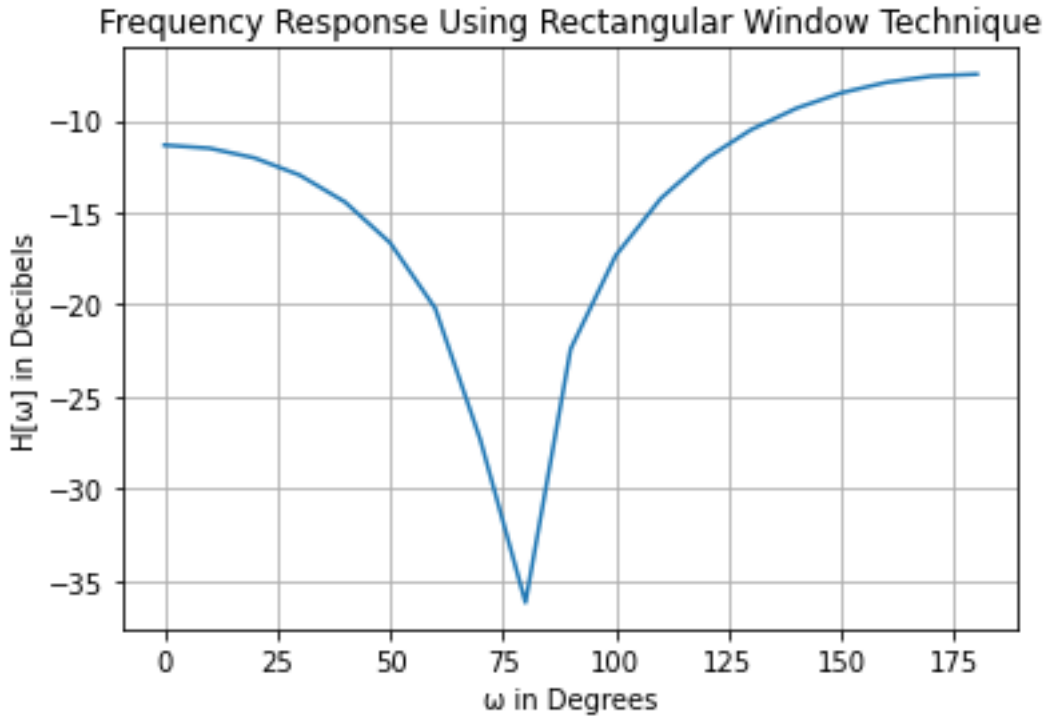
Step 1: Design FIR filter using Hanning window technique (Impulse Response).

Step 2: Compute the Hanning window function.

Step 3: Evaluate expected filter coefficients obtained using the rectangular window technique and sketch the same.

Frequency Response Using Rectangular Window Technique





Step 1: Design FIR filter using Hanning window technique (Frequency Response).

Step 2: Initialize the arrays for the real function of ω, frequency response (magnitude), frequency response (decibels) and define the angle axis.

Step 3: Sketch frequency response obtained using Hanning window technique.

```
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 10/Experiment 10.py'      ='E:/
Plan B/Amrita Vishwa Vidyapeetham/Subject Materials/Semester III/Signal Processing Lab
(19CCE281)/Assignments/Experiment 10'


Enter the number of coefficients for FIR Filter: 9

Enter the frequency response value: 1
C:\ProgramData\Anaconda3\lib\site-packages\scipy\integrate\quadpack.py:463:
ComplexWarning: Casting complex values to real discards the imaginary part
  return _quadpack._qagse(func,a,b,args,full_output,epsabs,epsrel,limit)

The desired filter coefficients h_d[n] is as follows:
 [0.4, 0.30273069145626275, 0.0935489283788639, -0.062365952252575955,
-0.07568267286406569, -0.062365952252575955, 0.0935489283788639, 0.30273069145626275, 0.4]

The expected filter coefficients h[n] obtained using rectangular window technique is as
follows:
 [0.4, 0.30273069145626275, 0.0935489283788639, -0.062365952252575955,
-0.07568267286406569, -0.062365952252575955, 0.0935489283788639, 0.30273069145626275, 0.4]

The frequency response H[ω] in decibels obtained using rectangular window technique is as
follows:
 [14.303623770340984, 14.168698839838907, 13.755207722024366, 13.034620347284942,
11.949934705871527, 10.394300591402395, 8.155434215778355, 4.737510939441853,
-1.5276369322431411, -22.420070769541706, -0.08652779526874857, 5.467937559251416,
8.654918813228287, 10.78278835301306, 12.275898562733062, 13.322944943740353,
14.020925431698764, 14.422241957121607, 14.553314463332587]

The expected filter coefficients h[n] obtained using Hanning window technique is as
follows:
 [0.0, 0.04433388332712346, 0.046774464189431944, -0.05323267000276442,
-0.07568267286406569, -0.05323267000276443, 0.04677446418943196, 0.04433388332712349, 0.0]

The frequency response H[ω] in decibels obtained using Hanning window technique is as
follows:
 [-11.342116110661228, -11.512594981687998, -12.039516796134206, -12.975179800045172,
-14.431524066355657, -16.64594370520493, -20.207868437905297, -27.356168782548593,
-36.198974461181955, -22.42007076954167, -17.33718643619604, -14.233266653016974,
-12.07599278330527, -10.501233150336287, -9.339282068886703, -8.49905963433598,
-7.928154198669187, -7.596189324289906, -7.487198085148416]

In [2]:
```