332.501 Final Project

Fall 2021

Name: Lohitanvita Rompicharla NetID: 211009876

Question 1: Verify the convolution relationship

Let us first generate a row vector a = [1 1 1 1 0 0 0 0] and make another vector b = a. Then
1.) Please plot the vector a in MATLAB.
2.) Use the definition of the convolution in the textbook to compute the output c = a * b and plot the vector c in MATLAB. That is, please derive the expression of the output c = a*b in your project document and plot your derived results.
3.) Use the MATLAB command d = conv (a; b); to get the convolution output d and plot d in MATLAB.
4.) Compare with c and d in MATLAB plots and make your comments.

MATLAB Script:

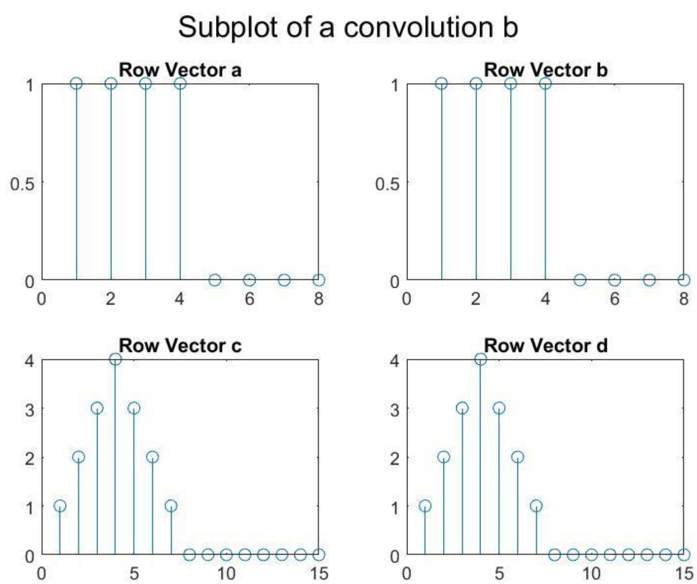
```
%1. (Verify the convolution relationship)

% Consider a row vector a = [1 1 1 1 0 0 0 0] and given its plot
 a = [1 1 1 1 0 0 0 0];
 subplot(2, 2, 1)
 stem(a)
 title ('Row Vector a')

% Consider another row vector b = [1 1 1 1 0 0 0 0] and given its plot
 b = a;
 subplot(2, 2, 2)
 stem(b)
 title ('Row Vector b')

% Manually convoluting a and b vectors, we get vector as shown in the plot:
 c = [1 2 3 4 3 2 1 0 0 0 0 0 0 0 0]
 subplot(2, 2, 3)
 stem(c)
 title ('Row Vector c')

% Convoluting a and b vectors in the MATLAB we get vector d and its plot as follows:
 d = conv(a,b);
 subplot(2, 2, 4)
 stem(d)
 title ('Row Vector d')

% A subplot including all the vectors a, b, c, d for comparison
 sgtitle ('Subplot of a convolution b')
```

Comments: refer to Question1_output.pdf for detailed output

Given two row vectors a = b = [1 1 1 1 0 0 0 0] and need to find their convolution through MATLAB and manually to compare the difference. We can find the convolution for two row vectors using linear convolution formula $x(n) * h(n) = \int_{-\infty}^{\infty} x(k).h(n-k)$.
We can assume vector ‘a’ as x(n) and vector ‘b’ as h(n), since both the vectors are starting at index zero we can manually convolute them as follows: that is 1st row = b(0)* all elements of a, 2nd row = b(1)* all elements of a and so on. The resulting single row convoluted matrix is obtained as below:

```
Vector a                 1 1 1 1 0 0 0 0
Vector b                 1 1 1 1 0 0 0 0
                    ___________________________________
                         1 1 1 1 0 0 0 0
                           1 1 1 1 0 0 0 0
                             1 1 1 1 0 0 0 0
                               1 1 1 1 0 0 0 0
                                 0 0 0 0 0 0 0 0
                                   0 0 0 0 0 0 0 0
                                     0 0 0 0 0 0 0 0
                                       0 0 0 0 0 0 0 0
                    ___________________________________
                         1 2 3 4 3 2 1 0 0 0 0 0 0 0 0
                    ___________________________________
```

Adding all the elements in the columns we derived the convoluted output as c = a*b = [1,2,3,4,3,2,1,0,0,0,0,0,0,0,0] with length 15 (i.e., a+b-1 = 8+8-1 =15)



The convolution of two vectors is the integral over the product of both functions, where one function is time-shifted and flipped in time. Now plotting c = a*b and d = conv(a, b) we observe that there is no difference in the values, since in the linear convolution of two row vectors we overlap and shift that always gives a linear time-invariant output. Also since a = b and there is no shift or deviation in the vectors, we expect to get same outputs for c and d.

Question 2: a) (Use convolution concept to estimate the relative time delay)
1.) Let vectors a = [1 1 1 1 0 0 0 0] and b = [0 0 0 1 1 1 1 0]. Use subplot to plot them in one figure. What is the relationship between a and b with your observation?
2.) Let vector c = conv(a; b) and plot it in MATLAB.
3.) Use MATLAB command [max val; ind] = max(c) to find the maximum output. What does ind mean? (i.e., is there any relationship between ind and the relative delay of a and b?)
4.) Change b to b = [0 0 1 1 1 1 0 0] and repeat the steps 2.) and 3.). What is the new ind value?
5.) Change b to b = [1 1 0 0 0 0 1 1] and repeat the steps 2.) and 3.). What is the new ind value?
6.) From the observation of 4.) and 5.), what is your conclusion of the meaning for ind?

MATLAB Script:

```
% 2. (Use convolution concept to estimate the relative time delay)
% Consider a row vector a = [1 1 1 1 0 0 0 0]
% Subplot of vectors a and b are plotted
a = [1 1 1 1 0 0 0 0]
subplot(2, 1, 1)
stem(a)
title ('Plot of Vector "a"')

% Now changing value of row vector b = [0 0 0 1 1 1 1 0], b = [0 0 1 1 1 1 0 0 ], b =
[1 1 0 0 0 0 1 1 ]

b = [0 0 0 1 1 1 1 0]
% b = [0 0 1 1 1 1 0 0]
% b = [1 1 0 0 0 0 1 1]
subplot(2, 1, 2)
stem(b)
title ('Plot of Vector "b"')

% Convoluting a and b, we get vector c which is plotted as below
c = conv(a,b)
figure()
stem(c)
title ('Plot of vector "c"')
[max_val, ind] = max(c)
```

Comments: refer Question2a_ouputs for detailed outputs.

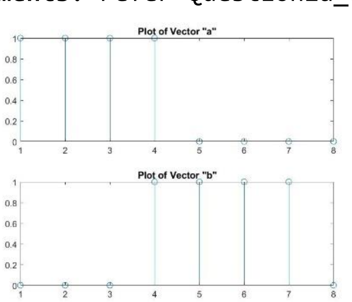


Fig.1

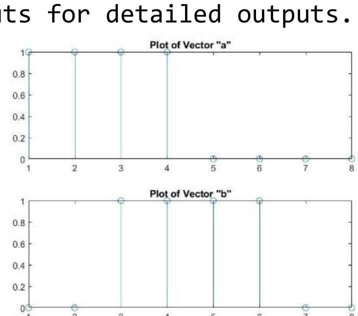


fig.2

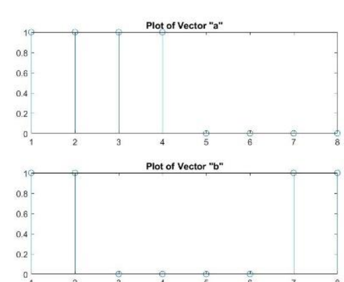


fig.3

```
Above are the three subplots with a = [1 1 1 1 0 0 0 0] and b = [0 0 0 1 1 1 1 0], b
= [0 0 1 1 1 1 0 0], b = [1 1 0 0 0 0 1 1] respectively.
From fig.1(refer Q2a1.jpg) we see that vector b is delayed by 3 samples (right
shift), fig.2(refer Q2a2.jpg) we see that vector b is delayed by 2 samples (right
shift), fig.3(refer Q2a3.jpg) we see that vector b is a circular shift of vector a by
2 samples.Therefore, all the values of vector b are the shifted versions of vector a.
```

OUTPUT: 2a 1.)
a =1 1 1 1 0 0 0 0
b = 0 0 0 1 1 1 1 0
c =0 0 0 1 2 3 4 3 2 1 0 0 0 0 0
max_val =4
ind =7

Comment: The 'ind' is the index of the maximum value present in the vector c, that is the maximum value in vector c is 4 at index 7. This means there is a delay in the signal. The delay of the signal can be found by ind – max_value = 7- 4 = 3 samples delay. Therefore 'ind' helps in representing the shift or delay between a, b vectors.

OUTPUT: 2a 4.)
a =1 1 1 1 0 0 0 0
b =0 0 1 1 1 1 0 0
c =0 0 1 2 3 4 3 2 1 0 0 0 0 0 0
max_val =4
ind = 6

Comment: The new ind value is 6. ind is the index of the maximum value present in the vector c, that is the maximum value in vector c is 4 at index 6. This means there is a delay in the signal. The delay of the signal is given by ind – max_value = 6 - 4 = 2 samples delay.

OUTPUT: 2a 5.)
a =1 1 1 1 0 0 0 0
b =1 1 0 0 0 0 1 1
c =1 2 2 2 1 0 1 2 2 2 1 0 0 0 0
max_val =2
ind =2

Comment: The new ind value is 2. ind is the index of the maximum value present in the vector c, that is the maximum value in vector c is 2 at index 2. The delay of the signal is given by ind – max_value = 2 - 2 = 0 samples delay.

2a 6.) ind = sub2ind(sz , row , col) returns the linear indices ind corresponding to the row and column subscripts in row and col for a matrix of size sz . Therefore, from the above outputs we can conclude that depending on the delay of a signal, the ind (index) value also changes.

Question 2: b) (Use convolution concept to estimate the relative time delay)

1.) Let vectors a = [1; 1; 1; 1; 0; 0; 0; 0;] (Note this a is a column vector but not a row vector) and b = circshift(a; 3)+0:1* randn(8; 1). (Note herein it is the function randn but not rand) Use subplot to plot them in one figure. Here b is essentially a shifted version of a with some noise distortion.

2.) Repeat steps 2.) and 3.) in the previous section a) multiple times (say, 10 times). What is your observation of the output value ind? Does ind change during your 10 trials?

3.) Keep a the same and change b = circshift(a; 3) + 10 *randn(8; 1).

4.) Repeat steps 2.) and 3.) in the previous section a) multiple times (say, 20 times). What is your observation of the output value ind? Does ind change during your 20 trials? What are the values you get in your test?

5.) Keep a the same and change b = circshift(a,-2) + 10 *randn(8; 1).

6.) Repeat steps 2.) and 3.) in the previous section a) multiple times (say, 20 times). What is your observation of the output value ind? Does ind change during your 20 trials? What are the values you get in your test?

MATLAB Script:

```
% Consider a column vector a = [1; 1; 1; 1; 0; 0; 0; 0;]
a = [1; 1; 1; 1; 0; 0; 0; 0;]
subplot(2, 1, 1)
stem(a)
title('Plot of vector "a"')

% Now taking values of b and plotting them respectively
b = circshift(a, 3)+0:1 * randn(8, 1)
% b = circshift(a, 3) + 10 * randn(8, 1)
% b = circshift(a,-2) + 10 * randn(8, 1)
subplot(2, 1, 2)
stem(b)
title('Plot of vector "b"')

% Now using for loop make 10 iterations.
l = 10
for k = 1:l
    a = [1; 1; 1; 1; 0; 0; 0; 0;]
    b = circshift(a, 3)+0:1 * randn(8, 1)
    c = conv(a,b)
    stem(c)
    title('Plot of vector"c"')
    [max_val, ind] = max(c)
end

% Now using for loop make 20 iterations.
i = 20
for k = 1:i
    a = [1; 1; 1; 1; 0; 0; 0; 0;]
    % b = circshift(a,-2) + 10 * randn(8, 1)
    b = circshift(a, 3) + 10 * randn(8, 1)
```

```
    c = conv(a,b)
    stem(c)
    title('Plot of vector"c"')
    [max_val, ind] = max(c)
end
```

OUTPUT 2b.2) (refer Quesiton2b_1outputs.pdf for complete set of outputs and 2b1_1.jpg, 2b_1result.jpg for plots)

a =
1
1
1
1
0
0
0
0

b =
0 1

c =
0
1
1
1
1
0
0
0
0
max_val =1

ind = 2

Comment: Here the vector b is the circular shift of vector a by 3 samples but there is an addition of noise function multiplied by 0.1, giving minute noise signal to the original signal. Since this noise signal is very minor and gives value randomly, we are getting similar 'ind' and 'max_val'. That is nearly max_value = 1, ind = 2, providing a delay of ind- max_val = 2 – 1 = 1 sample. Therefore, the SNR ratio is high providing good signal strength with less noise.

OUTPUT: 2b.4) (refer Quesiton2b_2outputs.pdf for complete set of outputs 2b3_1.jpg, 2b_3result.jpg for plots)

a =
1
1
1
1
0
0

0
0

b =
-3.4043
-7.4847
-2.4037
6.0293
-15.1625
-0.6835
8.8239
-13.2073

c =
-3.4043
-10.8890
-13.2927
-7.2634
-19.0216
-12.2204
-0.9928
-20.2294
-5.0668
-4.3833
-13.2073
0
0
0
0

max_val = 0
ind = 12

Comment: Here the vector b is the circular shift of vector a by 3 samples but there is an addition of noise function multiplied by 10, giving high noise signal to the original signal. Since this noise signal is higher than previous and gives value randomly, we are getting little deviations in the 'ind' and 'max_val'. Example: max_value = 0, ind = 12, providing a delay of ind- max_val = 12 – 0 = 12 samples. Therefore, the SNR ratio is low providing bad signal strength with less noise.

OUTPUT: 2b.6) `(refer Quesiton2b_3outputs.pdf for complete set of outputs 2b5_1.jpg, 2b_5result.jpg for plots)`

a =
1
1
1
1
0
0
0
0

b =
19.5120
-17.9766
-17.7868
-9.2264
-19.9793
-3.5710
-2.3640
3.5043

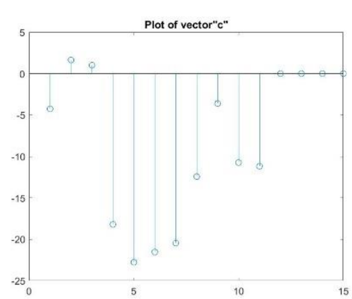
c =
19.5120
1.5354
-16.2514
-25.4779
-64.9692
-50.5635
-35.1407
-22.4100
-2.4307
1.1403
3.5043
0
0
0
0

max_val = 19.5120
ind = 1

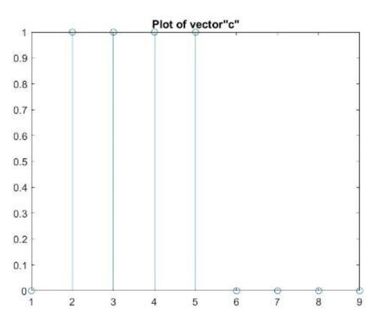
Comment: Here the vector b is the circular shift of vector a by negative 2 samples but there is an addition of noise function multiplied by 10, giving high noise signal to the original signal. Since this noise signal is higher than previous and gives value randomly, we are getting little deviations in the ‘ind’ and ‘max_val’. Example: max_value = 19 ind = 1, providing a delay of ind- max_val = 1 – 19 = |-18| samples. Since the signal is shifted left (negative), the index value is smaller than max_val.Therefore, the SNR ratio is low providing bad signal strength with less noise.

- From above three cases we can conclude that the random values of b are giving different values for ind and max_val and with the increase in noise signal. The SNR value becomes low leading to bad signal strength.

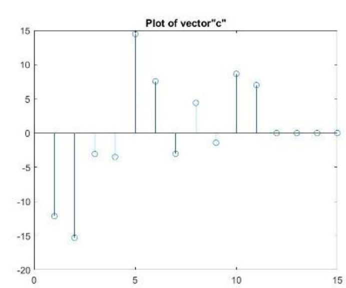
Below are the plotted results for above three cases respectively.



2b_1result.jpg



2b_3result.jpg



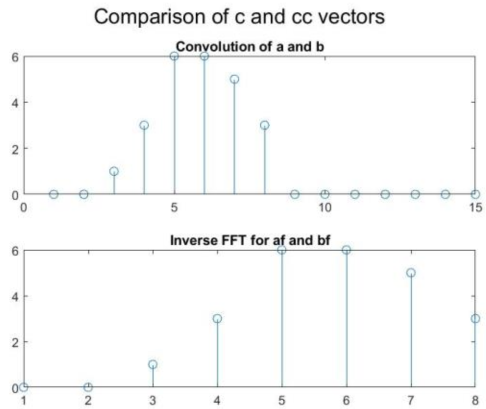
2b_5result.jpg

Question3. a)(Time-delay impacts on frequency domain)
1.) Let vectors a = [1 1 1 1 0 0 0 0] and b = [0 0 1 2 3 0 0 0]. Plot them in Matlab.
2.) Let vector c = conv(a; b) and plot it in Matlab.
3.) Let discrete Fourier transforms af = fft(a) and bf = fft(b). Plot the absolute values of af and bf, i.e., abs(af) and abs(bf) in Matlab.
4.) Let cc = ifft(af: _ bf) and plot cc.
5.) Compare with cc and c and make comments of your results.

MATLAB Script:

```
% Consider vectors a = [1 1 1 1 0 0 0 0] and b = [0 0 1 2 3 0 0 0]
a = [1 1 1 1 0 0 0 0]
figure()
subplot(2,1,1)
stem(a)
title('Plot for Vector "a"')

% vector b = [0 0 1 2 3 0 0 0]
b = [0 0 1 2 3 0 0 0]
subplot(2,1,2)
stem(b)
title('Plot for Vector "b"')

% convolution of vectors a and b is equated to c
c = conv(a, b)
figure()
subplot(2,1,1)
stem(c)
title('Convolution of a and b')

% taking fast fourier trasnform and their absolute values for vectors a,b
af = fft(a)
fft_a = abs(af)
subplot(2,1,1)
stem(fft_a)

% title('Absolute value of fft(a)')
bf = fft(b)
fft_b = abs(bf)
subplot(2,1,2)
stem(fft_b)
title('Absolute value of fft(b)')

% Applying inverse fft on af, bf and plotting the graph
cc = ifft(af.* bf)
subplot(2,1,2)
stem(cc)
title('Inverse FFT for af and bf')
sgtitle('Comparison of c and cc vectors')
```

Comments: (refer Question3a_outputs.pdf for all the outputs and 3a1_1.jpg, 3a1_2.jpg, 3a1_3.jpg, 3a1_4.jpg, 3a1_5result.jpg for figures)

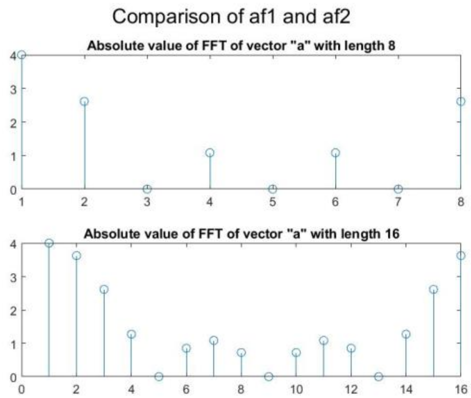


The above figure displays plots for comparison of c and cc vectors output. In this experiment we have convoluted the vectors a and b in time domain and resulted as vector c. Also converted vectors a, b to frequency domain by fft transform and convoluted by dot product and saved its ifft as cc vector. Since c and cc are the outputs of same vectors but convoluted in different domains, we see that the output values are same but there is a little delay in cc vector. This shows that time-delay effects the frequency domain while converting back to ifft.

Question3. b)(Time-delay impacts on frequency domain)
1.) Let vectors a = [1 1 1 1 0 0 0 0] and af1 = fft(a; 8). Plot abs(af1) in Matlab.
2.) Let af2 = fft(a; 16). Plot abs(af2) in Matlab.
3.) Compare with af1 and af2, what is your observation?

MATLAB Script:

```
a = [1 1 1 1 0 0 0 0]
af1 = fft(a, 8)
af1_absolute = abs(af1)
subplot(2,1,1)
stem(af1_absolute)
title('Absolute value of FFT of vector "a" with length 8')

af2 = fft(a, 16)
af2_absolute = abs(af2)
subplot(2,1,2)
stem(af2_absolute)
title('Absolute value of FFT of vector "a" with length 16')

sgtitle('Comparison of af1 and af2')
```



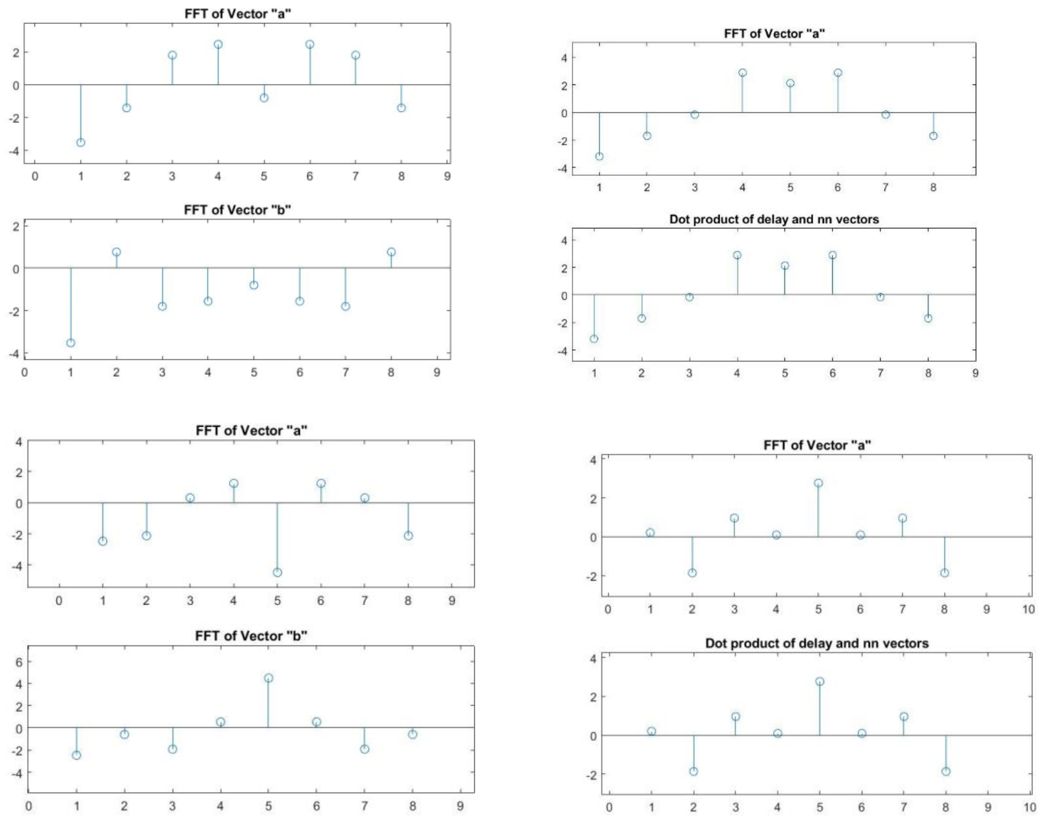
Comments: (refer to Question3b_outputs.pdf and and 3b_1res.jpg for detailed outputs)
Here we take a vector 'a' whose length is 8 and converted it into frequency domain by fft of transform length 8 and with transform length 16 and plotted their graphs. From the above graph of comparison we can conclude the for a vector with increase in transform length while domain conversions, the more the transform length the high is its resolution. We can get more values or impulses giving us better resolution and continuous wave.

Question3. c)(Time-delay impacts on frequency domain)
1.) Let vector a = randn(8; 1) and af = fft(a).
2.) Let vector b = circshift(a; 3) and bf = fft(b). Are af and bf the same?
3.) Let index sequence nn = [0 : 7]' (column vector).
4.) Let delay = exp(j *2 *pi *nn *3/8).
5.) Let afa = delay.*bf and compare afa with af. What is difference between them?
6.) Change vector b = circshift(a;-2) and bf = fft(b). Are af and bf the same?
7.) Let delay = exp(j* 2*pi *nn *(-2)/8).
8.) Let afa = delay.*bf and compare afa with af. What is difference between them?
Why? What is your conclusion now?

MATLAB Script:
```
a = randn(8, 1)
af = fft(a)
subplot(2,1,1)
stem(af)
title('FFT of Vector "a"')
% b = circshift(a, 3)
b = circshift(a,-2)
bf = fft(b)
subplot(2,1,2)
stem(bf)
title('FFT of Vector "b"')

% Let index sequence nn = [0 : 7]' (column vector).
nn = [0 : 7]'

% Let delay = exp(j * 2 * pi * nn * 3/8).
% delay = exp(j * 2 * pi * nn * 3/8)
delay = exp(j * 2 * pi * nn * (-2)/8)
```

```
% Let afa = delay.*bf and compare afa with af.
afa = delay.*bf
subplot(2,1,2)
stem(afa)
title('Dot product of delay and nn vectors')
```

Comment: (refer Question3c_outputs.pdf for detailed outputs)



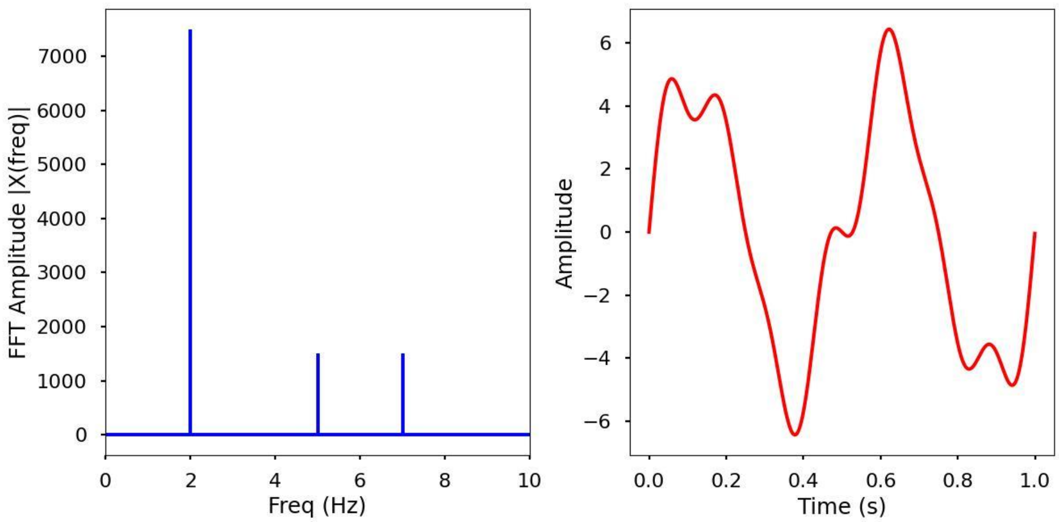
The above figures depict that for both the values of b, the vectors a and b are not similar but have little time delay. Then we add a delay with exponential function to convolute the vectors a, b in frequency domain. This addition of delay in the signals imapact the domain conversion to frequency domain. This resulted output graphs afa and af to be same.

4. Use FFT and IFFT to filter out the single sinusoid with frequency 7 and eliminate the rest.

Python Script:

```
import matplotlib.pyplot as plt
import numpy as np
from numpy.fft import fft, ifft

plt.style.use('seaborn-poster')
"matplotlib inline"
```

```
# Building a sample signal with sampling rate 3000 and sampling interval
1.0/3000
# sampling rate
sr = 3000
# sampling interval
ts = 1.0/sr
t = np.arange(0,1,ts)

freq = 2
x = 5*np.sin(2*np.pi*freq*t)

freq = 5
x += np.sin(2*np.pi*freq*t)

freq = 7
x += 1* np.sin(2*np.pi*freq*t)

plt.figure(figsize = (8, 6))
plt.plot(t, x, 'r')
plt.ylabel('Amplitude')

plt.show()

"""Considering frequency = 7 as the original signal and freq =2,5 as the
interference and noise signal frequecncies
 We are trying to depict the time domain sinusoidal signal into frequency
domain in order to eliminate noise signals  using fft"""

# fft of signal x which is a sinusoidal signal of frequencies 2,5,7
X = fft(x)
N = len(X)
n = np.arange(N)
T = N/sr
freq = n/T

plt.figure(figsize = (12, 6))
plt.subplot(121)

plt.stem(freq, np.abs(X), 'b', \markerfmt=" ", basefmt="-b")
plt.xlabel('Freq (Hz)')
plt.ylabel('FFT Amplitude |X(freq)|')
plt.xlim(0, 10)

plt.subplot(122)
plt.plot(t, ifft(X), 'r')
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.tight_layout()
plt.show()
```



Comment: We have build and compiled a python code for converting a sinusoidal signal from time domain to frequency domain using fast fourier transform and its inverse. Taking a sinusoidal signal with frequencies 2,5,7, we have plotted a graph. In this Frequency = 7 is our original signal and frequency = 2,5 are considered as the noise or interference signals. Since we cannot distinguish between the noise signals and original signal in the time domain, we are transforming this sinusoidal signal into frequency domain using FFT. The above figure displays the frequency and time domain signals. From the frequency domain signal we can easily distinguish between the all the different frequency signals which enables us to easily remove the noise signal from the main signal. This helps us to understand the importance of different domains depending on the application. Therefore, this process can be used when we need to filter out the noise or interference signal from any photo, audio etc.