

به نام خدا

## DSP CA2

نام و نام خانوادگی: محمد تقی زاده

شماره دانشجویی: 810198373

# Question 1

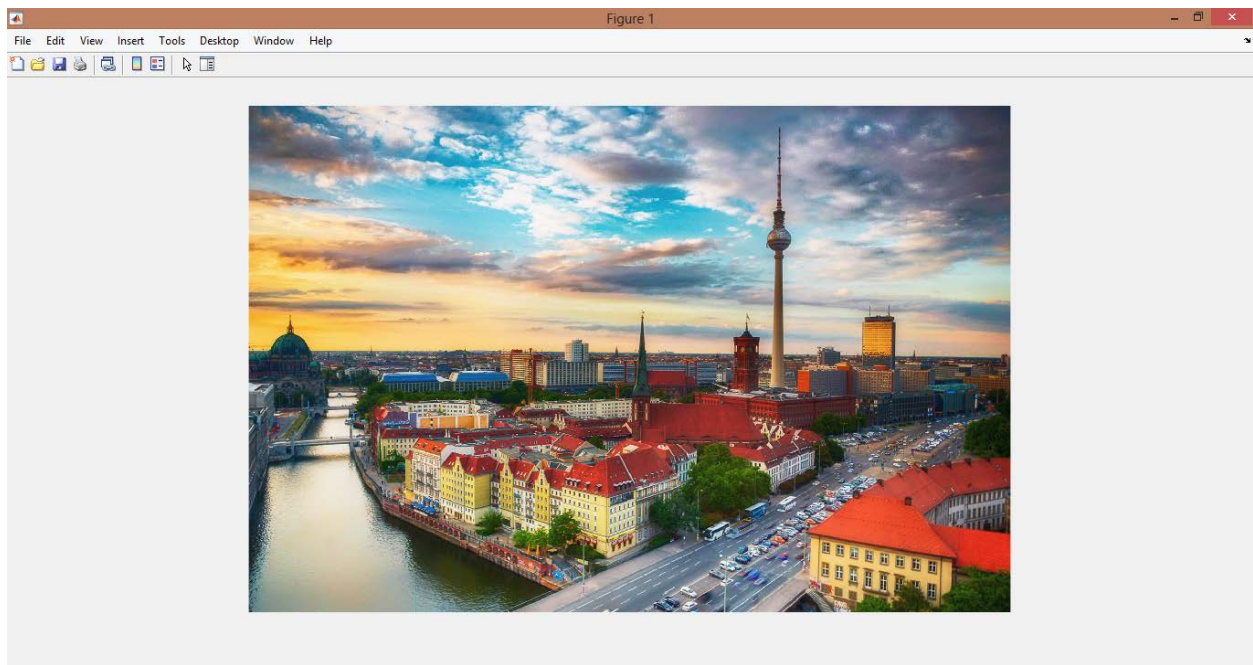
## Part 1 – Gaussian

```
D:\matlab\Projects\DSP_CA2_Q1.m

EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment % Indent Find Breakpoints Run Run and Advance Run Section Advance Run and Time

FILE NAVIGATE EDIT BREAKPOINTS RUN

12 subplot(2, 1, 2)
13 Blurring_Filter = fspecial('disk', 30);
14 imshow(imfilter(input_image, Blurring_Filter, 'replicate'));
15
16 figure
17 Blurred_Image = rgb2gray(Blurred_Image);
18 imagesc(abs(fftshift(fft2(Blurred_Image))));
19 colorbar;
20
21 %%
22 [row, column, colors] = size(input_image);
23 Gaussian_Filter = fspecial('gaussian', [row, column], 0.5);
24 Gaussian_Filtered_Image = imfilter(input_image, Gaussian_Filter, 'replicate');
25 imshow(Gaussian_Filtered_Image);
26
27 figure
28 subplot(2, 1, 1)
29 imshow(Gaussian_Filtered_Image)
30 subplot(2, 1, 2)
31 Gaussian_Filter = fspecial('gaussian', [row, column], 0.1);
32 imshow(imfilter(input_image, Gaussian_Filter, 'replicate'));
33
34 figure
35 Gaussian_Filtered_Image = rgb2gray(Gaussian_Filtered_Image);
36 imagesc(abs(fftshift(fft2(Gaussian_Filtered_Image))));
37 colorbar;
38
39 %%
40 Laplacian_Filter = fspecial('laplacian', 0.2);
41 Laplacian_Filtered_Image = imfilter(input_image, Laplacian_Filter, 'replicate');
42 imshow(Laplacian_Filtered_Image);
43
44 figure
```



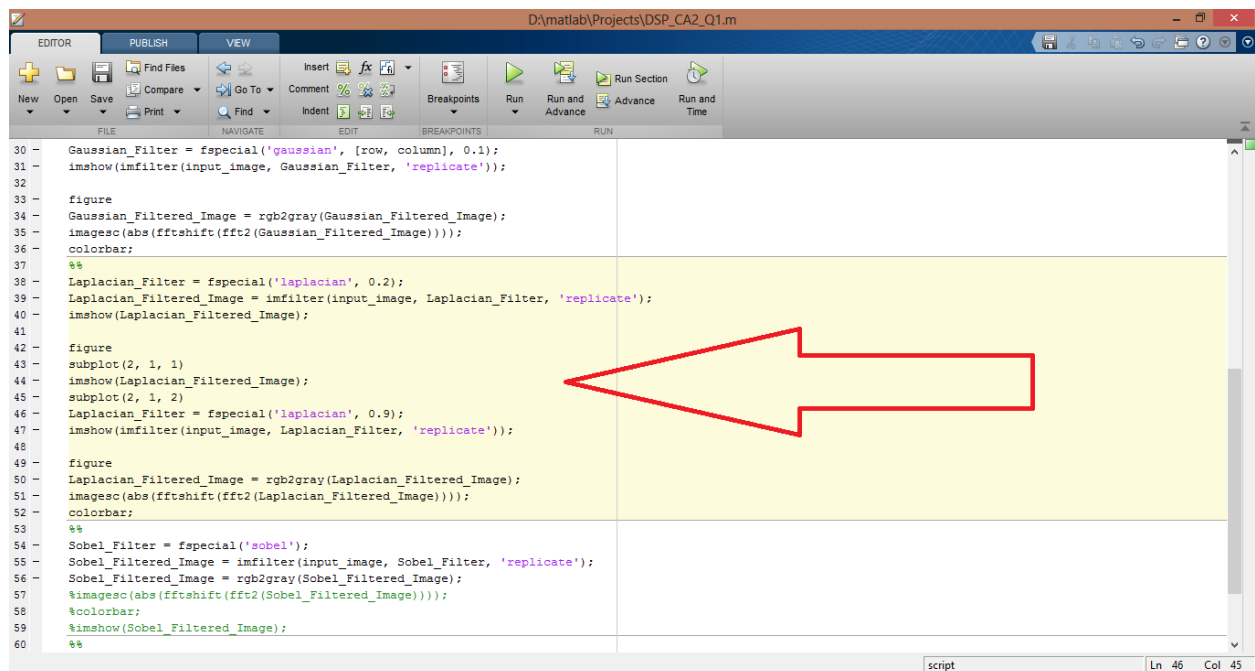
فیلتر گاوسی از رابطه زیر تبعیت می کند:

$$h_g(n_1, n_2) = e^{\frac{-(n_1^2 + n_2^2)}{2\sigma^2}}$$

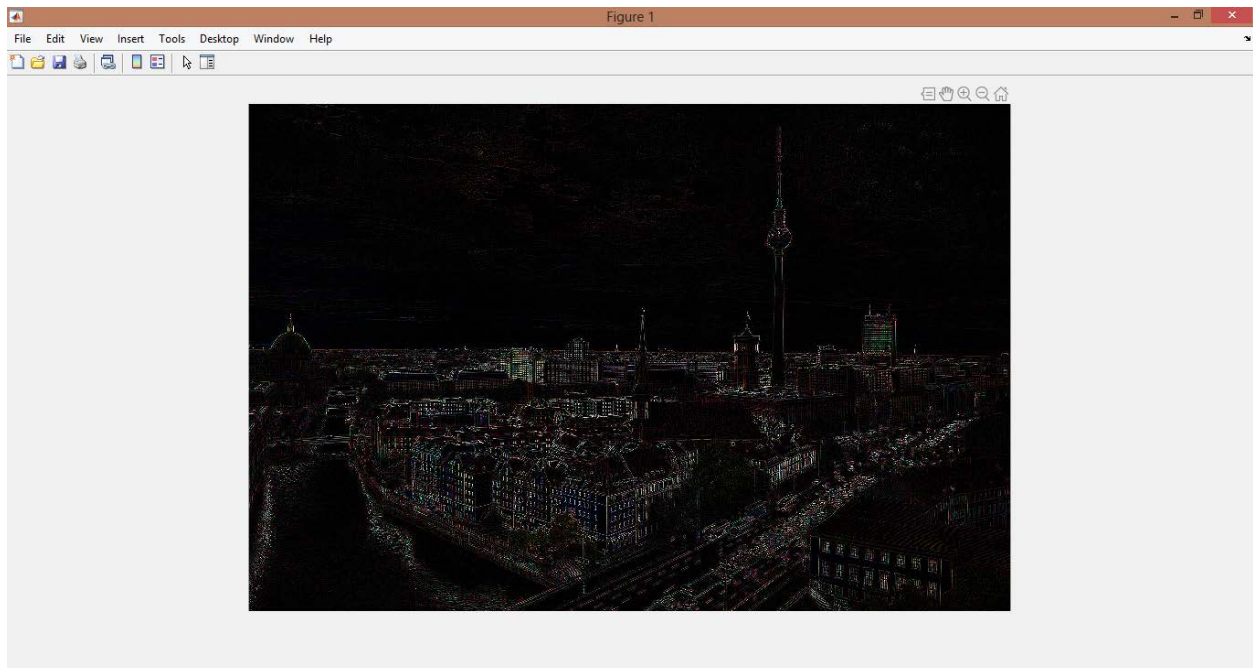
$$h(n_1, n_2) = \frac{h_g(n_1, n_2)}{\sum_{n_1} \sum_{n_2} h_g}$$

همان طور که مشاهده می شود، این فیلتر پایین گذر بوده و به همین دلیل در لبه ها که تغییرات زیاد است، خروجی این فیلتر 0 شده (تغییرات زیاد را عبور نمی دهد) و در نتیجه همان طور که در تصویر حاصل از این فیلتر پیداست، لبه ها سیاه شده اند.

## Part 1 – Laplacian



```
30 - Gaussian_Filter = fspecial('gaussian', [row, column], 0.1);
31 - imshow(imfilter(input_image, Gaussian_Filter, 'replicate'));
32
33 - figure
34 - Gaussian_Filtered_Image = rgb2gray(Gaussian_Filtered_Image);
35 - imagesc(abs(fftshift(fft2(Gaussian_Filtered_Image))));
36 - colorbar;
37 - %%
38 - Laplacian_Filter = fspecial('laplacian', 0.2);
39 - Laplacian_Filtered_Image = imfilter(input_image, Laplacian_Filter, 'replicate');
40 - imshow(Laplacian_Filtered_Image);
41
42 - figure
43 - subplot(2, 1, 1)
44 - imshow(Laplacian_Filtered_Image);
45 - subplot(2, 1, 2)
46 - Laplacian_Filter = fspecial('laplacian', 0.9);
47 - imshow(imfilter(input_image, Laplacian_Filter, 'replicate'));
48
49 - figure
50 - Laplacian_Filtered_Image = rgb2gray(Laplacian_Filtered_Image);
51 - imagesc(abs(fftshift(fft2(Laplacian_Filtered_Image))));
52 - colorbar;
53 - %%
54 - Sobel_Filter = fspecial('sobel');
55 - Sobel_Filtered_Image = imfilter(input_image, Sobel_Filter, 'replicate');
56 - Sobel_Filtered_Image = rgb2gray(Sobel_Filtered_Image);
57 - %imagesc(abs(fftshift(fft2(Sobel_Filtered_Image))));
58 - %colorbar;
59 - %imshow(Sobel_Filtered_Image);
60 - %%
```



فیلتر لاپلاسیان از رابطه زیر تبعیت می کند:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

این فیلتر، مشتق و تغییرات هر پیکسل را نسبت به پیکسل های مجاورش محاسبه کرده و مقدار محاسبه شده را جایگزین مقداری که در پیکسل مورد بررسی هست، می کند. از جایی که در لبه ها مقادیر پیکسل ها تغییر زیادی می کند، فقط لبه ها در تصویر بالا مشخص هستند و بقیه پیکسل ها چون تغییراتشان اندک (0) است در نتیجه مقدار پیکسل نیز 0 شده و رنگ آن سیاه می شود.

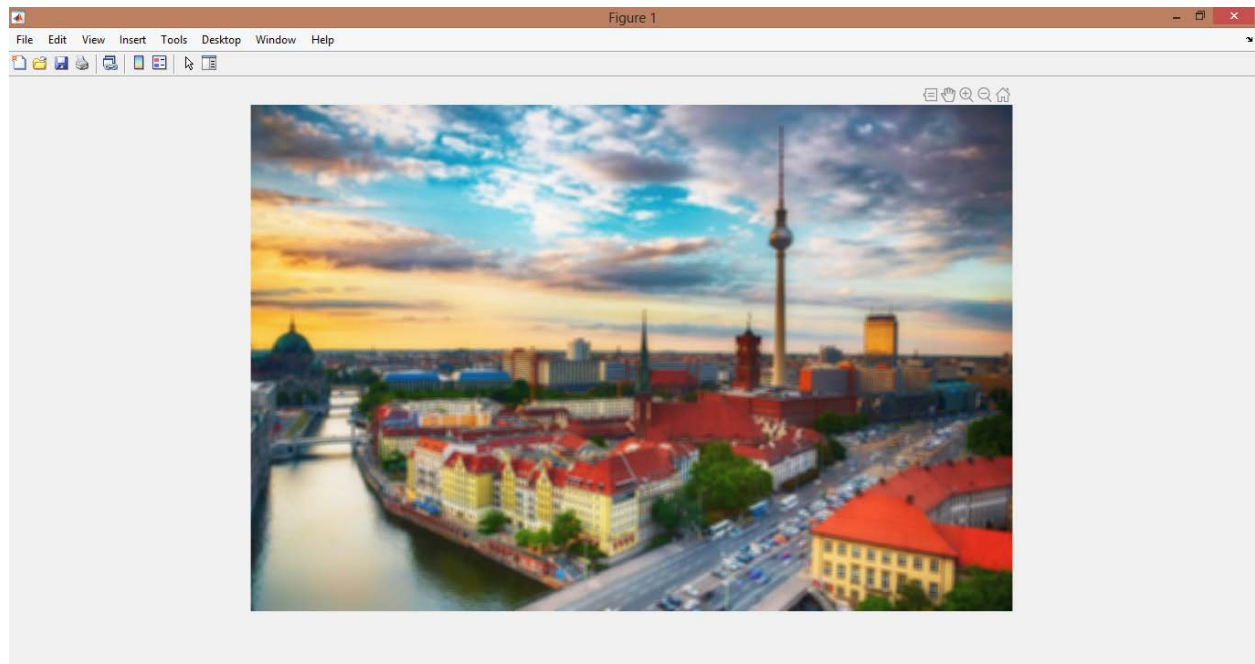
## Part 1 – Unsharp Average

```
D:\matlab\Projects\DSP_CA2_Q1.m

EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Insert Comment Breakpoints Run Run and Advance Run Section Advance Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN

46 Laplacian_Filter = fspecial('laplacian', 0.9);
47 imshow(imfilter(input_image, Laplacian_Filter, 'replicate'));
48
49 figure
50 Laplacian_Filtered_Image = rgb2gray(Laplacian_Filtered_Image);
51 imagesc(abs(fftshift(fft2(Laplacian_Filtered_Image))));
52 colorbar;
53 %%
54 Sobel_Filter = fspecial('sobel');
55 Sobel_Filtered_Image = imfilter(input_image, Sobel_Filter, 'replicate');
56 Sobel_Filtered_Image = rgb2gray(Sobel_Filtered_Image);
57 imagesc(abs(fftshift(fft2(Sobel_Filtered_Image))));
58 colorbar;
59 imshow(Sobel_Filtered_Image);
60 %%
61 Average_Filter = fspecial('average', 10);
62 Averaged_Image = imfilter(input_image, Average_Filter, 'replicate');
63 imshow(Averaged_Image);
64
65 figure
66 subplot(2, 1, 1)
67 imshow(Averaged_Image);
68 subplot(2, 1, 2)
69 Average_Filter = fspecial('average', 30);
70 imshow(imfilter(input_image, Average_Filter, 'replicate'));
71
72 figure
73 Averaged_Image = rgb2gray(Averaged_Image);
74 imagesc(abs(fftshift(fft2(Averaged_Image))));
75 colorbar;
76
```

script Ln 69 Col 42



این فیلتر از رابطه زیر تبعیت می کند:

$$h(n) = \text{ones}(n(1), n(2)) / (n(1) * n(2))$$

این فیلتر میانگین مقادیر پیکسل های مجاور پیکسل را جایگزین مقدار موجود در پیکسل مورد بررسی می کند که این باعث می شود رنگ های پیکسل ها بر اثر میانگیری درهم دیگر آمیخته شوند و در نتیجه عکس، حالت تار و بلوری پیدا کند.

## Part 1 – Disk

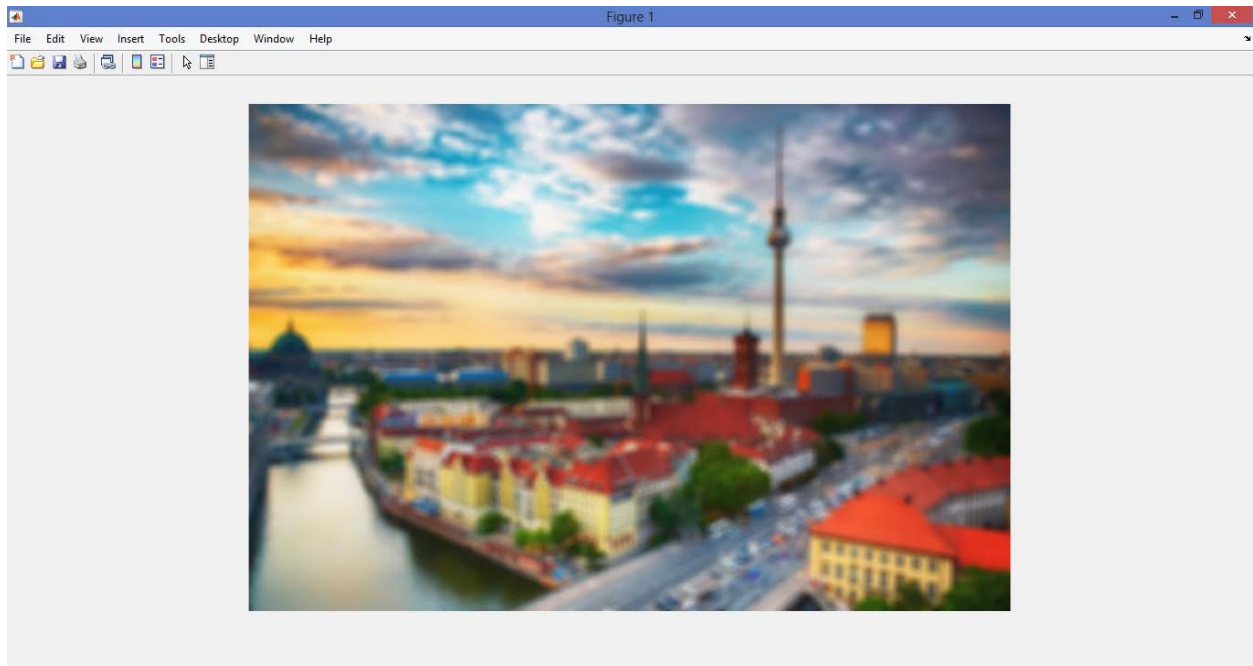
The screenshot shows the MATLAB Editor window with the following script:

```

1 input_image = imread('img.jpg');
2
3 Blurring_Filter = fspecial('disk', 10);
4 Blurred_Image = imfilter(input_image, Blurring_Filter, 'replicate');
5
6
7 imshow(Blurred_Image)
8
9 figure
10 subplot(2, 1, 1)
11 imshow(Blurred_Image)
12 subplot(2, 1, 2)
13 Blurring_Filter = fspecial('disk', 30);
14 imshow(imfilter(input_image, Blurring_Filter, 'replicate'));
15
16 figure
17 Blurred_Image = rgb2gray(Blurred_Image);
18 imagesc(abs(fftshift(fft2(Blurred_Image))));
19 colorbar;
20
21 %%
22 [row, column, colors] = size(input_image);
23 Gaussian_Filter = fspecial('gaussian', [row, column], 0.5);
24 Gaussian_Filtered_Image = imfilter(input_image, Gaussian_Filter, 'replicate');
25 imshow(Gaussian_Filtered_Image);
26
27 figure
28 subplot(2, 1, 1)
29 imshow(Gaussian_Filtered_Image)
30 subplot(2, 1, 2)
31 Gaussian_Filter = fspecial('gaussian', [row, column], 0.1);
32 imshow(imfilter(input_image, Gaussian_Filter, 'replicate'));

```

A red arrow points to the `figure` command on line 9.



این فیلتر، عکس را تار و بلور می کند. درواقع، به ازای هر پیکسل، محدوده ای دیسک شکل به مرکز آن پیکسل، در نظر گرفته، و در آن محدوده با انجام یک سری عملیات روی پیکسل های آن محدوده، رنگ پیکسل های آن ناحیه را در هم می آمیزد که این منجر به تار شدن و بلوری شدن تصویر می شود.

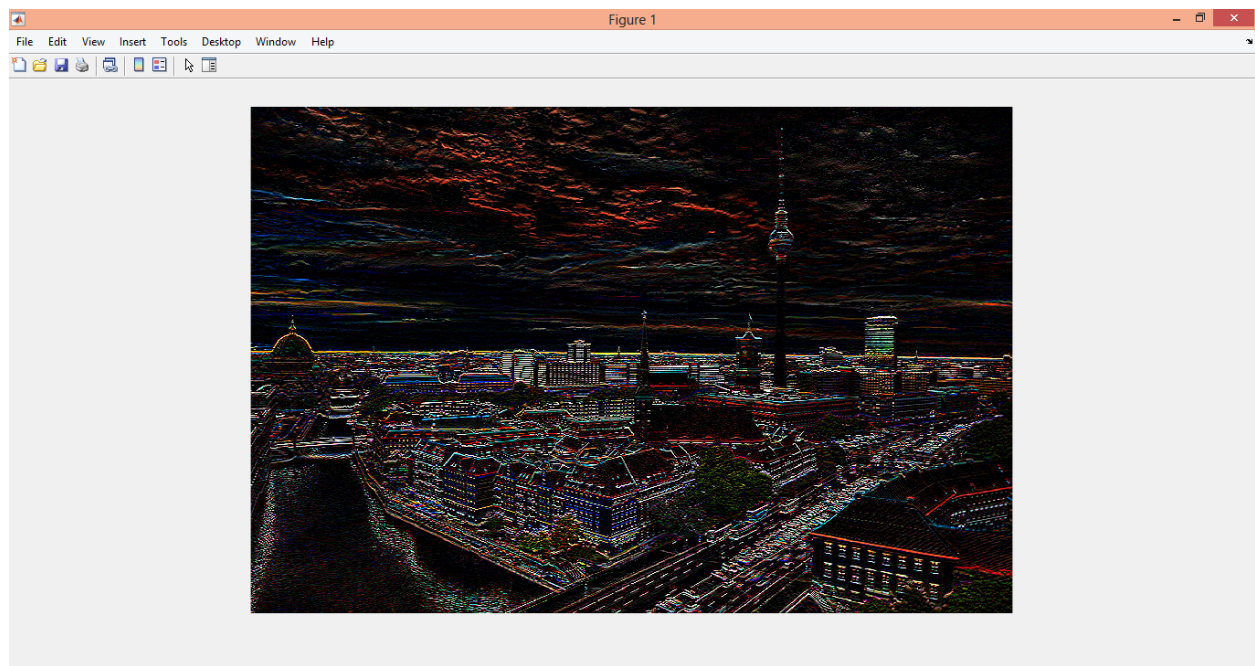


## Part 1 – Sobel

```
D:\matlab\Projects\DSP_CA2_Q1.m

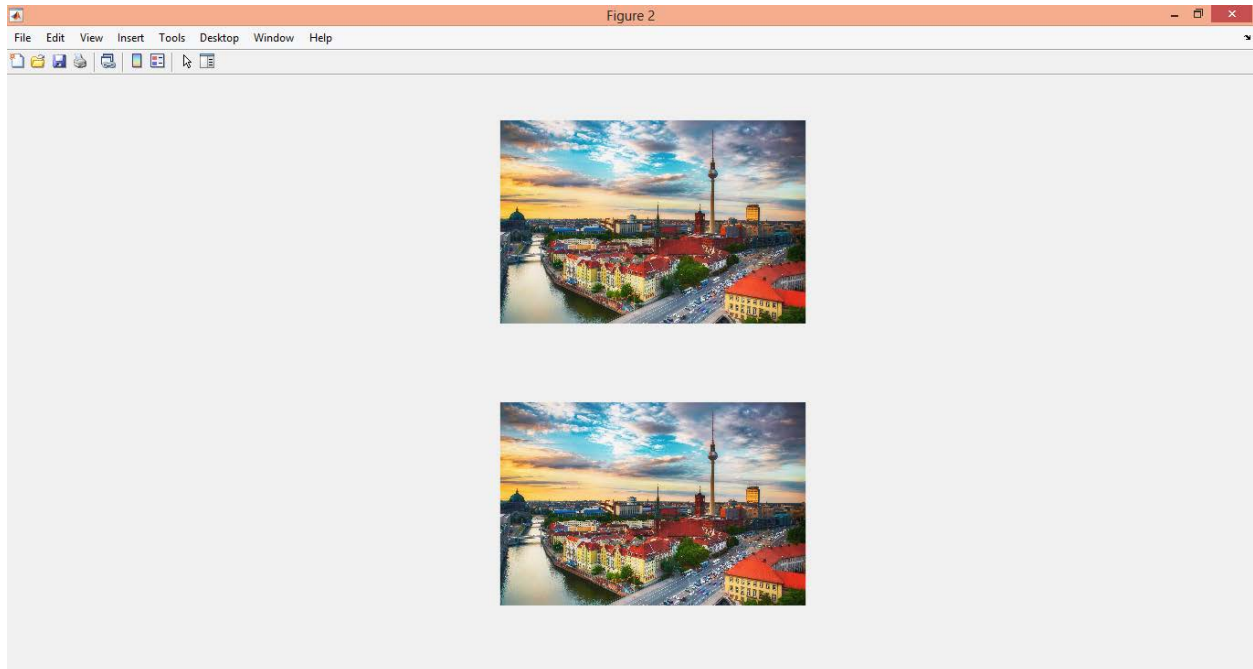
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Insert Comment Breakpoints Run Run and Advance Run Section Advance Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN

44 - imshow(Laplacian_Filtered_Image);
45 - subplot(2, 1, 2)
46 - Laplacian_Filter = fspecial('laplacian', 0.9);
47 - imshow(imfilter(input_image, Laplacian_Filter, 'replicate'));
48
49 - figure
50 - Laplacian_Filtered_Image = rgb2gray(Laplacian_Filtered_Image);
51 - imagesc(abs(fftshift(fft2(Laplacian_Filtered_Image))));
52 - colorbar;
53 - %%
54 - Sobel_Filter = fspecial('sobel');
55 - Sobel_Filtered_Image = imfilter(input_image, Sobel_Filter, 'replicate');
56 - imshow(Sobel_Filtered_Image);
57
58 - figure
59 - Sobel_Filtered_Image = rgb2gray(Sobel_Filtered_Image);
60 - imagesc(abs(fftshift(fft2(Sobel_Filtered_Image))));
61 - colorbar;
62 - %%
63 - Average_Filter = fspecial('average', 10);
64 - Averaged_Image = imfilter(input_image, Average_Filter, 'replicate');
65 - imshow(Averaged_Image);
66
67 - figure
68 - subplot(2, 1, 1)
69 - imshow(Averaged_Image);
70 - subplot(2, 1, 2)
71 - Average_Filter = fspecial('average', 30);
72 - imshow(imfilter(input_image, Average_Filter, 'replicate'));
73
74 - figure
```



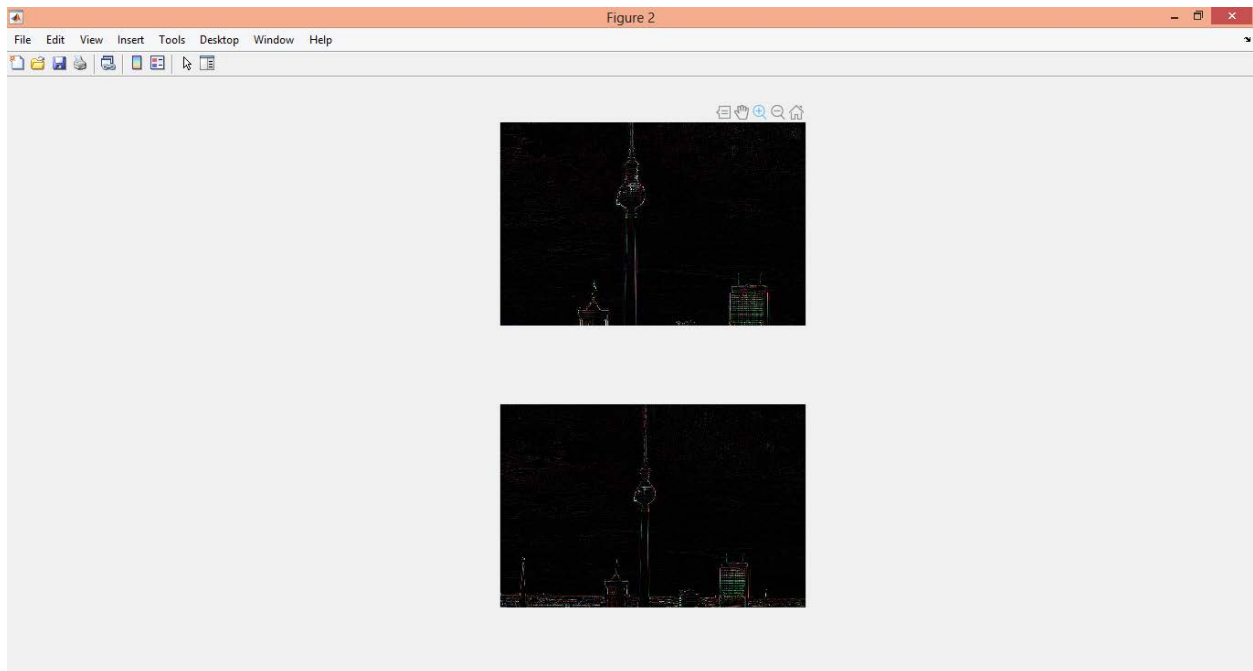
این فیلتر، لبه های افقی تصویر را تشخیص می دهد. درواقع، این فیلتر برای هر پیکسل، گرادیان عمودی را محاسبه کرده و مقدار محاسبه شده را جایگزین مقدار پیکسل مورد بررسی می کند. از جایی که در لبه ها مقادیر پیکسل ها تغییر زیادی می کند، فقط لبه های افقی در تصویر بالا مشخص هستند و بقیه پیکسل ها چون گرادیان شان (تغییراتشان) اندک (0) است در نتیجه مقدار پیکسل نیز 0 شده و رنگ آن سیاه می شود.

## Part 2 – Gaussian



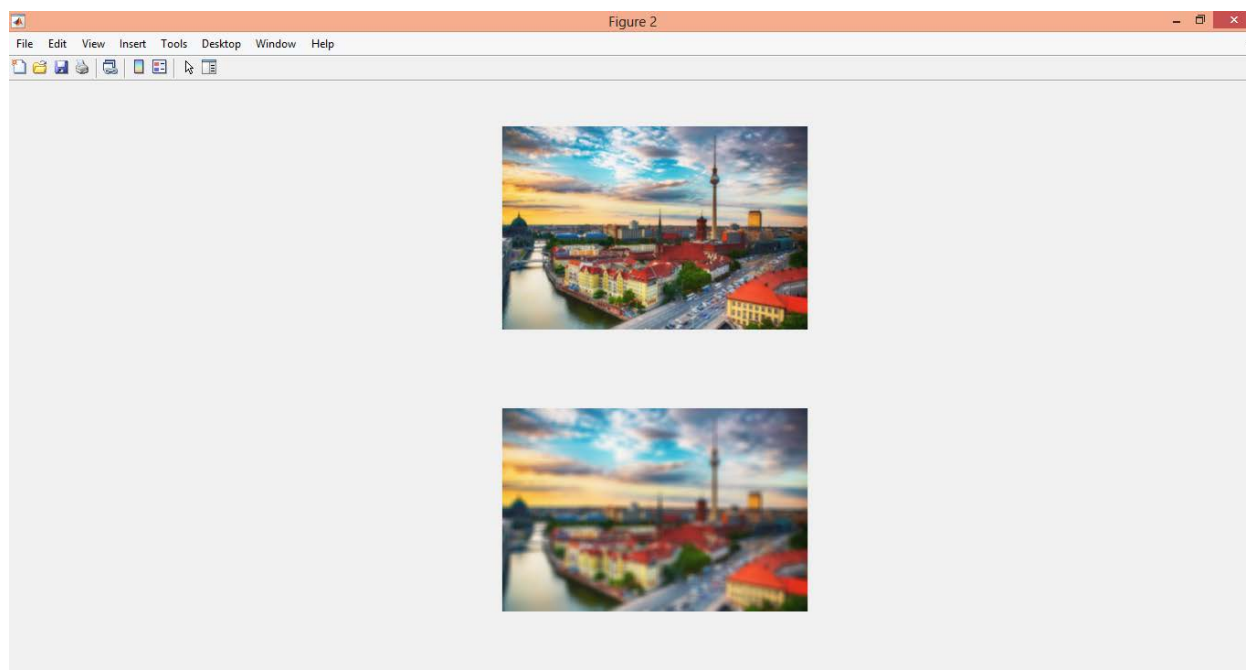
بر اثر زیاد شدن سیگما در فیلتر گاوسی،  $h_g(n_1, n_2)$  به سمت 1 میل کرده و در نتیجه، فیلتر، فیلتر پایین گذرتری می شود. در نتیجه لبه ها که فرکانس تغییراتشان زیاد است، بیش تر فیلتر شده و بنابراین، لبه ها سیاه تر و تاریک تر می شوند.

## Part 2 – Laplacian



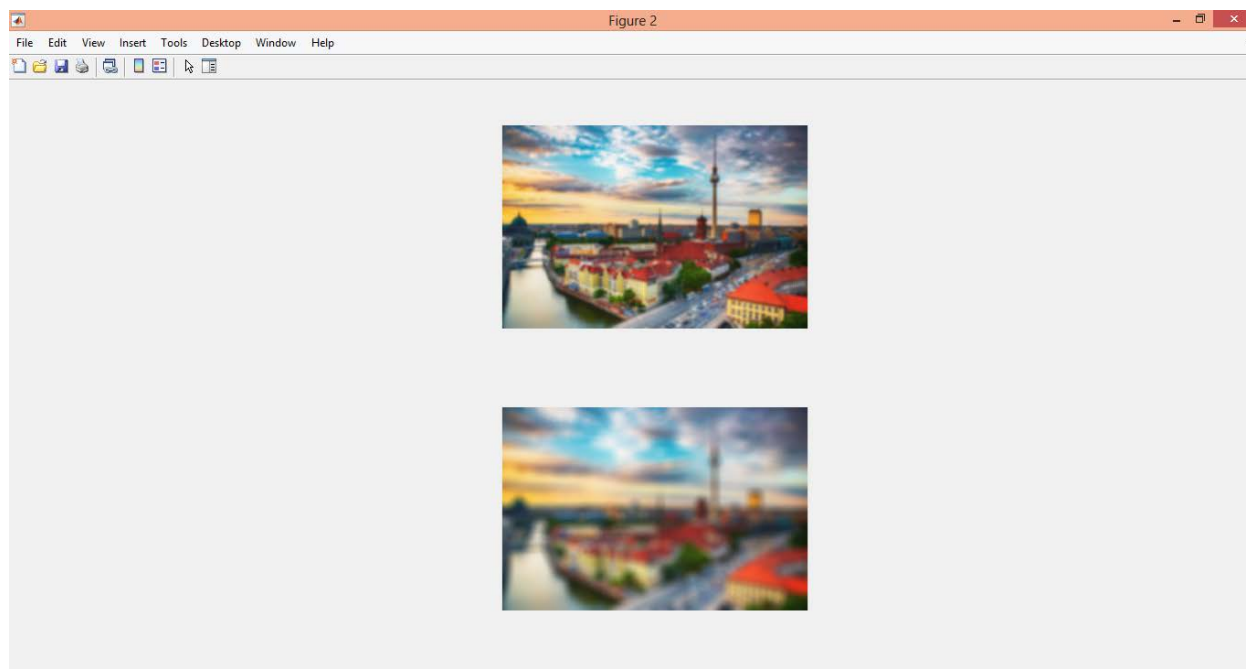
بر اثر زیاد شدن آلفا، ضریب ضرب شده در گرادیان پیکسل، کوچک تر شده که این منجر به متمایل شدن مقدار پیکسل ها به 0 (سیاه) شده و در نتیجه رنگ پیکسل ها تاریک تر می شود که این امر در سیاه تر بودن رنگ پس زمینه عکس فوقانی (عکس بالاتر) مشخص شده است. به عبارتی لبه ها بیشتر برجسته شده و نمایان می شوند.

## Part 2 – Unsharp Average



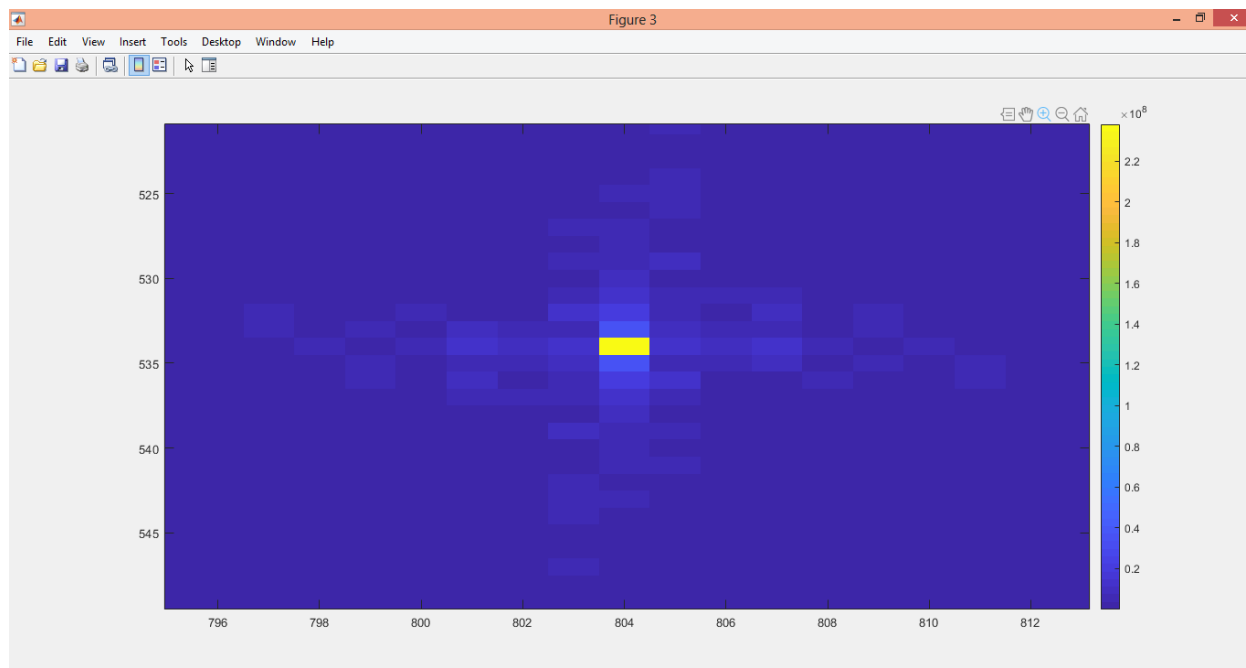
برای زیاد شدن سایز فیلتر، محدوده ای که به ازای هر پیکسل، میانگین گیری انجام میشه (شعاعی که در آن از مقادیر پیکسل ها میانگین گرفته شده و جایگزین مقدار پیکسل مورد بررسی می شود.) بزرگتر شده و در نتیجه، رنگ پیکسل ها بیش تر به هم آمیخته می شود. در نتیجه تصویر تار تر می شود.

## Part 2 – Disk



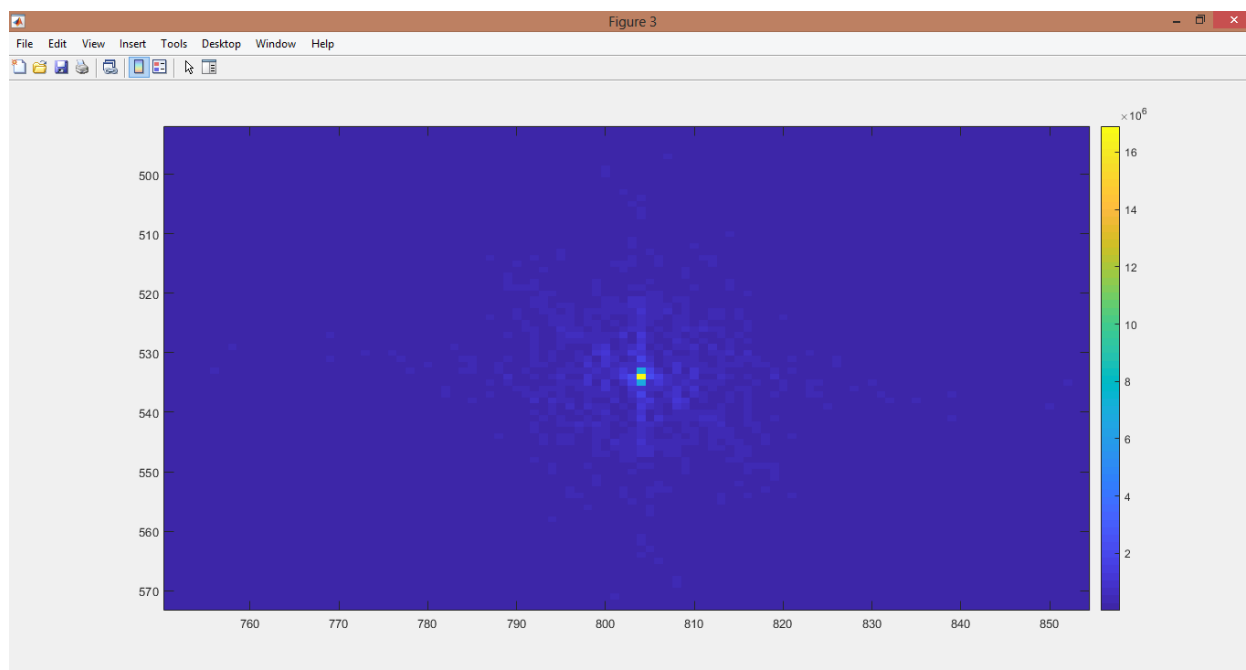
بر اثر زیاد شدن شعاع فیلتر، محدوده ای که به ازای هر پیکسل، میانگین گیری انجام میشه (شعاعی که در آن از مقادیر پیکسل ها میانگین گرفته شده و جایگزین مقدار پیکسل مورد بررسی می شود.) بزرگتر شده و در نتیجه، رنگ پیکسل ها بیش تر به هم آمیخته می شود. در نتیجه تصویر تار تر می شود

## Part 3 – Gaussian



باتوجه به تبدیل فوریه رسم شده در تصویر فوق، پهنای باند عبور حدودا از 796 تا 812 است و همان طور که پیداست، از جایی که پهنای باند، متمرکز در فرکانس های کوچک است، این فیلتر، پایین گذر است.

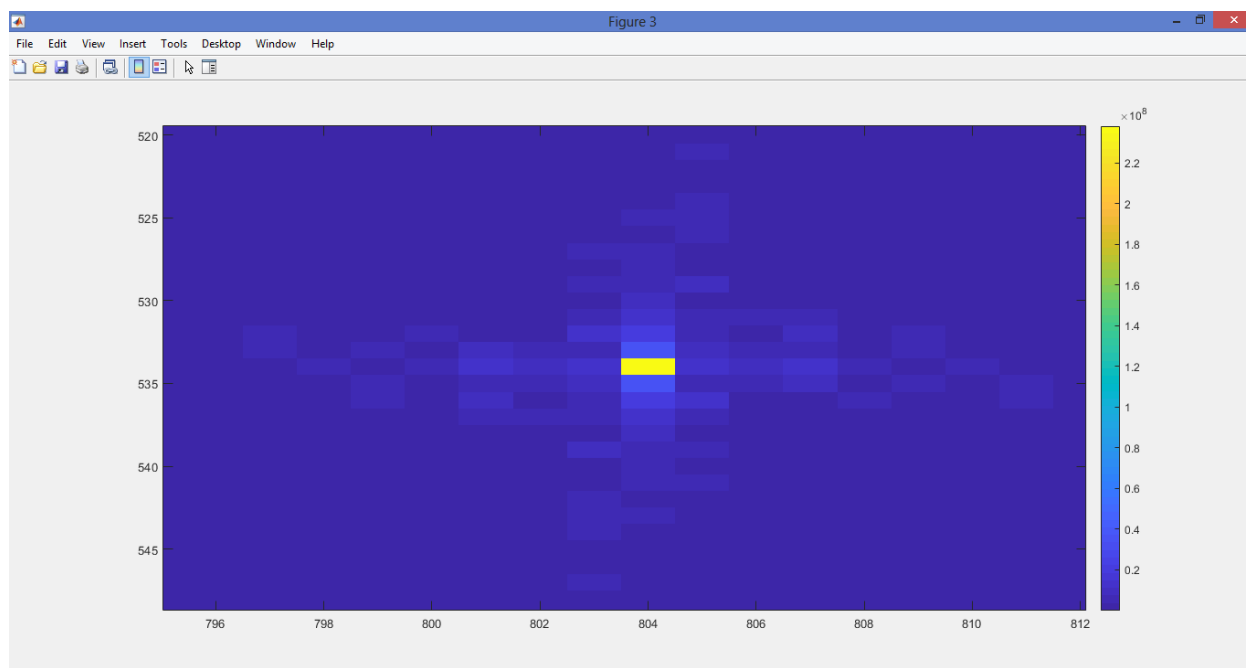
## Part 3 Laplacian



باتوجه به تبدیل فوریه رسم شده در تصویر فوق، پهنای باند عبور حدودا از 757 تا 853 است و همان طور که پیداست، از جایی که پهنای باند، نسبت به پهنای باند فیلتر گاوسی بیش تر است، این فیلتر، بالا گذر است.

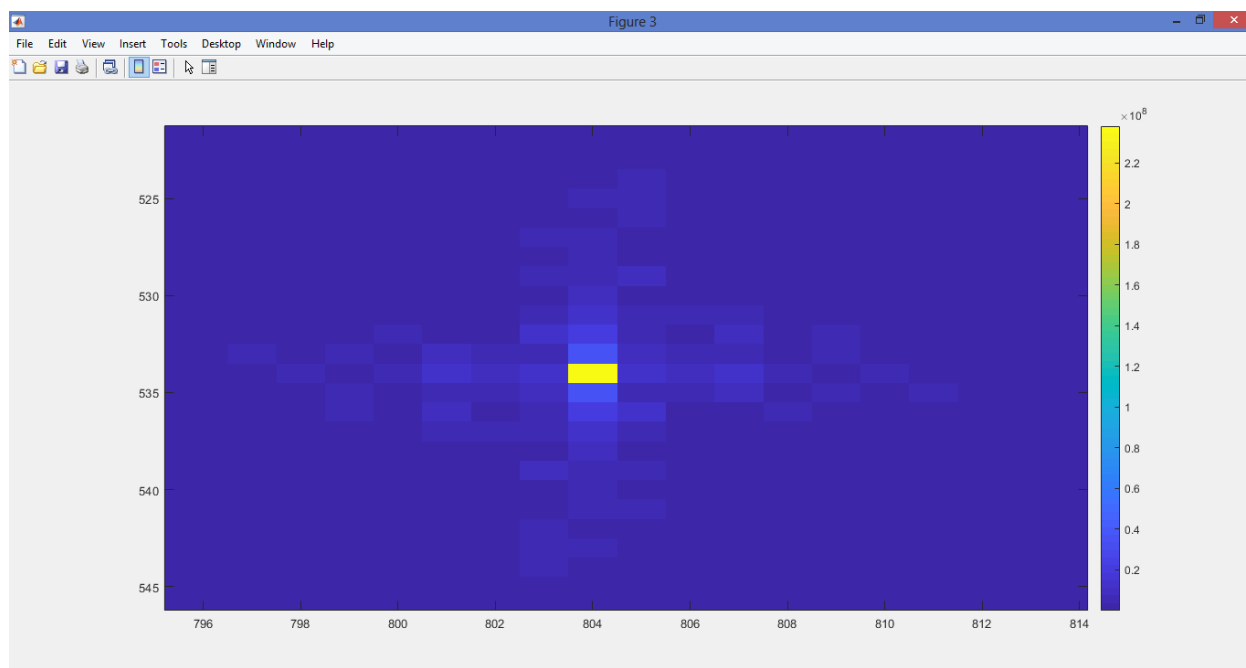


## Part 3 – Unsharp Average



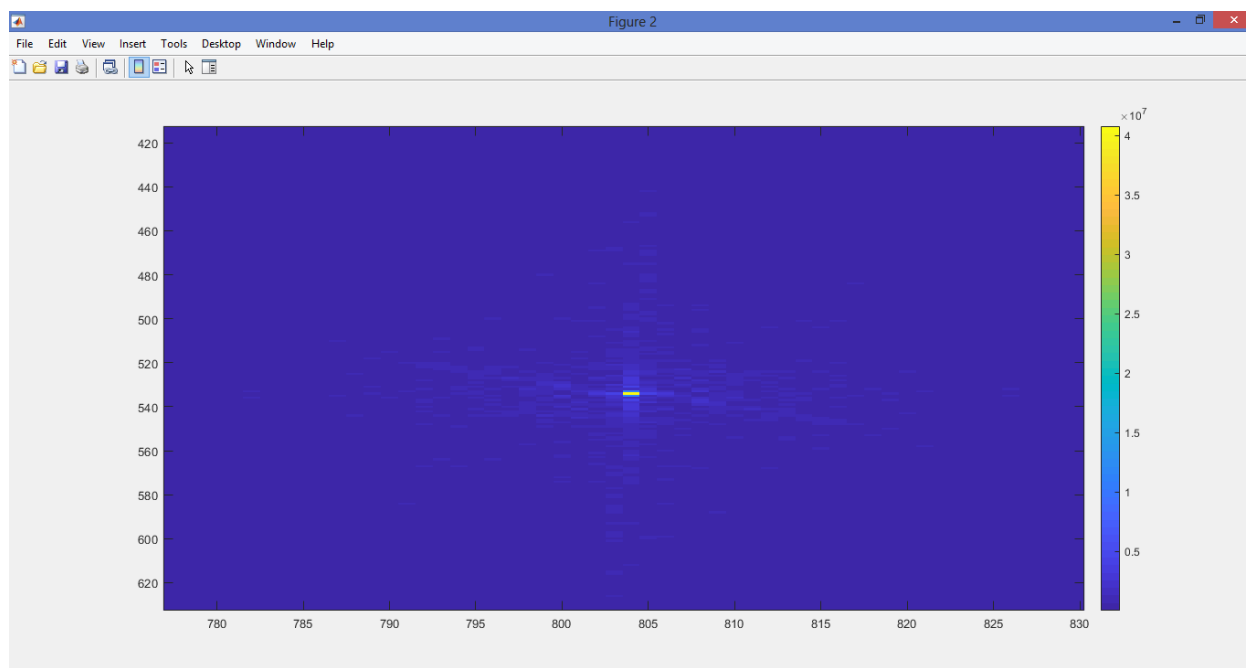
باتوجه به تبدیل فوریه رسم شده در تصویر فوق، پهنای باند عبور حدودا از 796 تا 812 است و همان طور که پیداست، از جایی که پهنای باند، متمرکز در فرکانس های کوچک است، این فیلترمانند فیلتر گاوسی، پایین گذر است.

## Part 3 Disk



باتوجه به تبدیل فوریه رسم شده در تصویر فوق، پهنای باند عبور حدودا از 796 تا 812 است و همان طور که پیداست، از جایی که پهنای باند، متمرکز در فرکانس های کوچک است، این فیلترمانند فیلترهای گاوسی و میانگین، پایین گذر است.

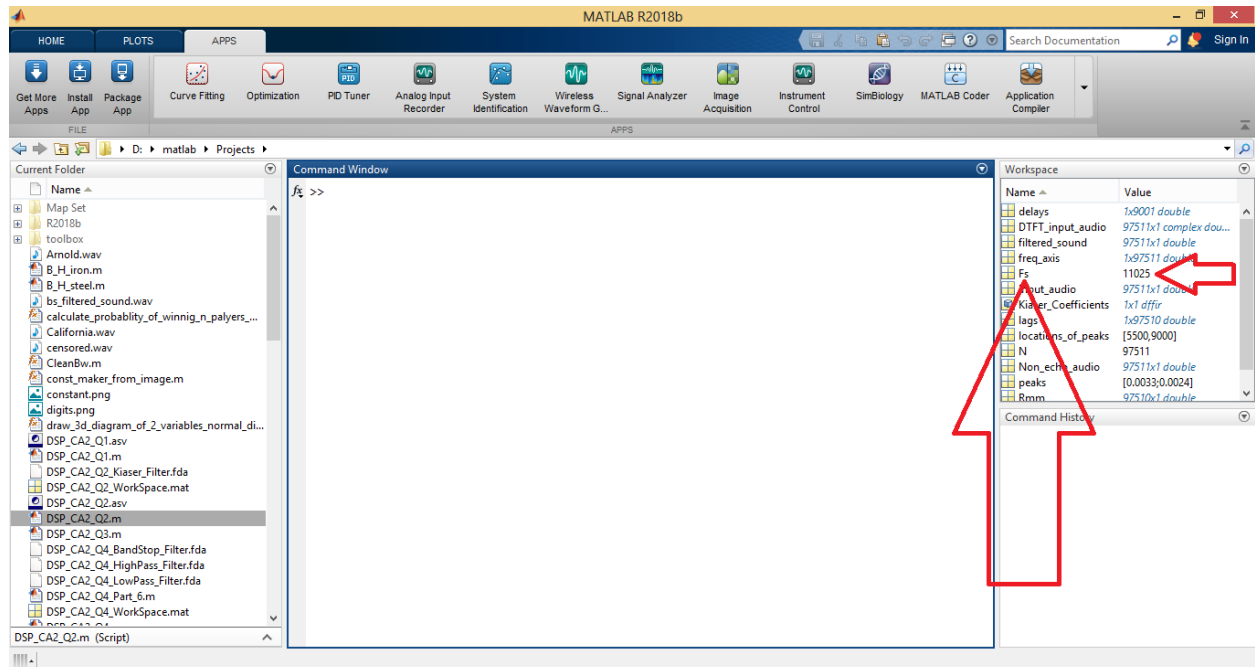
## Part 3 Sobel



باتوجه به تبدیل فوریه رسم شده در تصویر فوق، پهنای باند عبور حدودا از 782 تا 827 است و همان طور که پیداست، از جایی که پهنای باند، نسبت به پهنای باند فیلتر گاوسی بیش تر است، این فیلتر، مانند فیلتر لاپلاسین، (در محور افق) بالا گذر است.

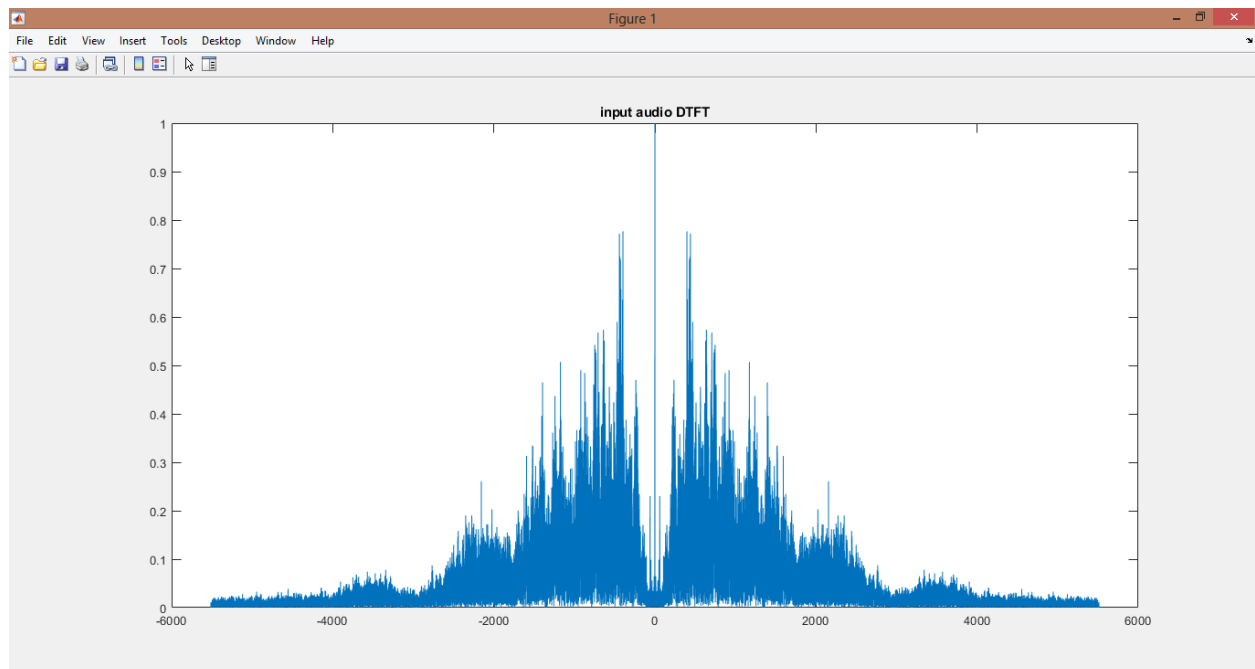
# Question 2

## Part 1



=>  $F_s = 11025$  Hz

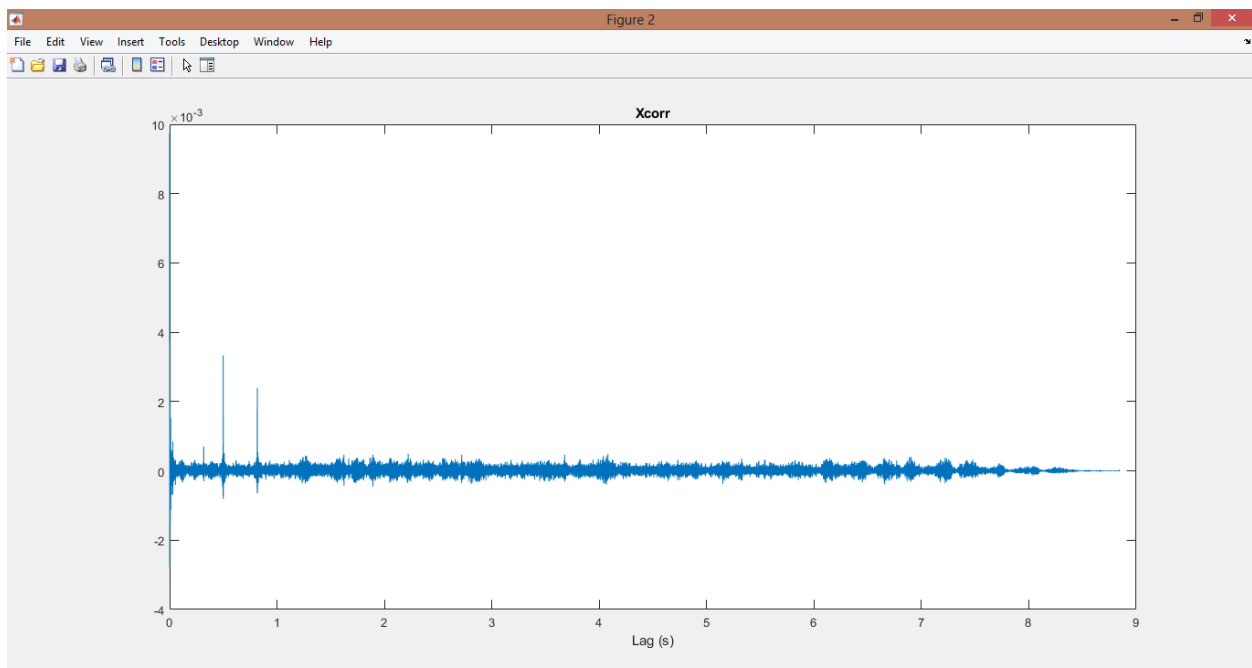
## Part 2



=> Band Width  $\approx$  5500 Hz

## Part 3, 4

برای بدست آوردن مقادیر آلفا و بتا، ابتدا با استفاده از تابع `xcorr` همبستگی سیگنال با شیفته هایش بدست می آوریم. سه پیک مشاهده خواهد شد که یکی در نقطه 0 و دو تا دیگر به ترتیب در K1 و K2 هستند. پس با پیدا کردن مکان هایی که پیک داریم، مقادیر تاخیر بدست می آیند. در ادامه برای محاسبه آلفا کافیست که مقدار پیک در نقطه K1 را بر مقدار پیک در نقطه 0 تقسیم کنیم. به طور مشابه برای بتا کافیست که مقدار پیک در نقطه K2 را بر مقدار پیک در نقطه 0 تقسیم کنیم.



$$\Rightarrow K1 = 0.4988 \times F_s = 0.4988 \times 11025 = 5500$$

$$\Rightarrow K2 = 0.8163 \times F_s = 0.8163 \times 11025 = 9000$$

$$\Rightarrow \alpha = \text{xcorr}(K1) / \text{xcorr}(1) = 0.0033 / 0.0098 \approx 0.3$$

$$\Rightarrow \beta = \text{xcorr}(K2) / \text{xcorr}(1) = 0.0024 / 0.0098 \approx 0.2$$

## Part 5

$$y[n] = x'[n] + \alpha x'[n - k1] + \beta x'[n - k1]$$

$$Z \text{ Transform } \Rightarrow Y(z) = X'(z) + \alpha z^{-k1} X'(z) + \beta z^{-k2} X'(z)$$

$$\Rightarrow \frac{Y(z)}{X'(z)} = 1 + \alpha z^{-k1} + \beta z^{-k2} = h(z)$$

$$\Rightarrow h(n) = \delta(n) + \alpha \delta(n - k1) + \beta \delta(n - k2)$$

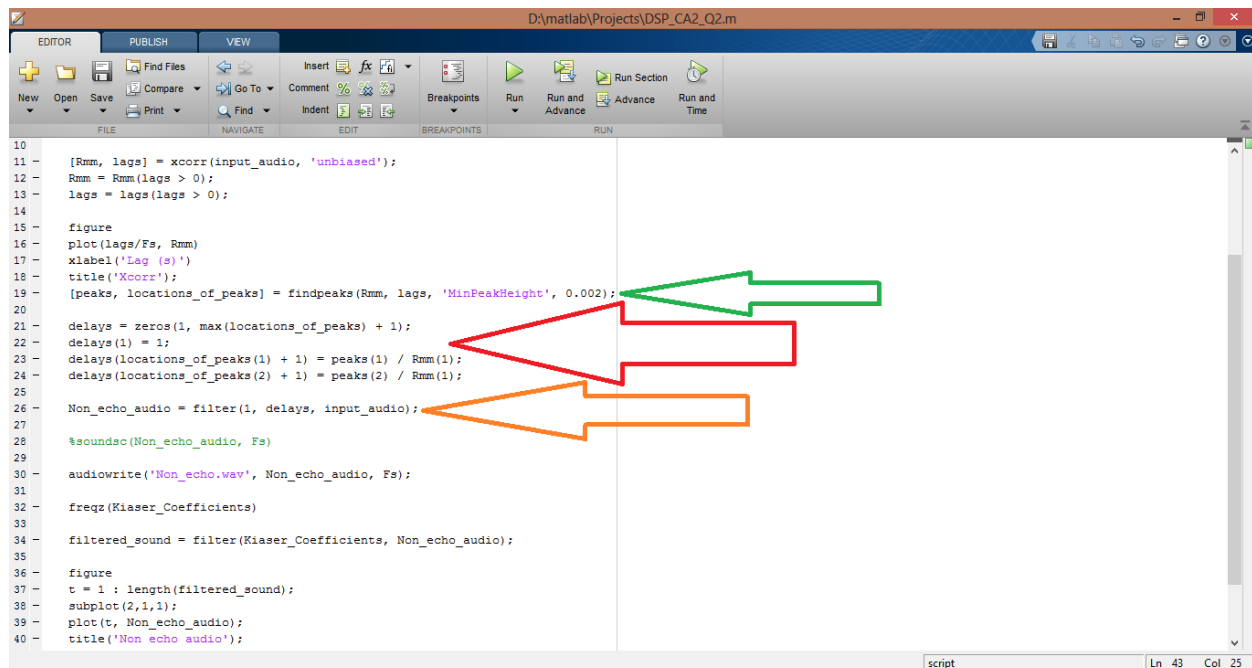
## Part 6

$$\Rightarrow X'(z) = Y(z) \times \frac{X'(z)}{Y(z)} \Rightarrow \frac{X'(z)}{Y(z)} = \frac{1}{1 + \alpha z^{-k1} + \beta z^{-k2}}$$

برای رسیدن از سیگنال اکو به سیگنال بدون اکو مطابق روابط بالا، سیگنال دارای اکو را از فیلتری با

$$h'(z) = \frac{1}{1 + \alpha z^{-k1} + \beta z^{-k2}}$$

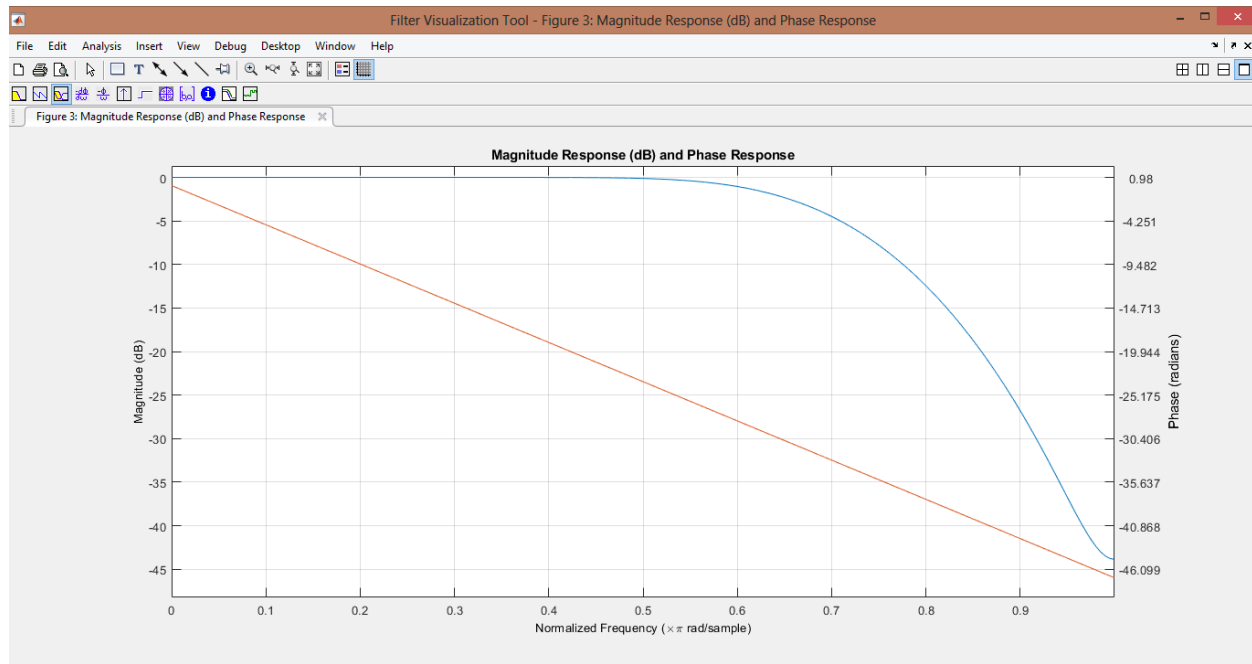
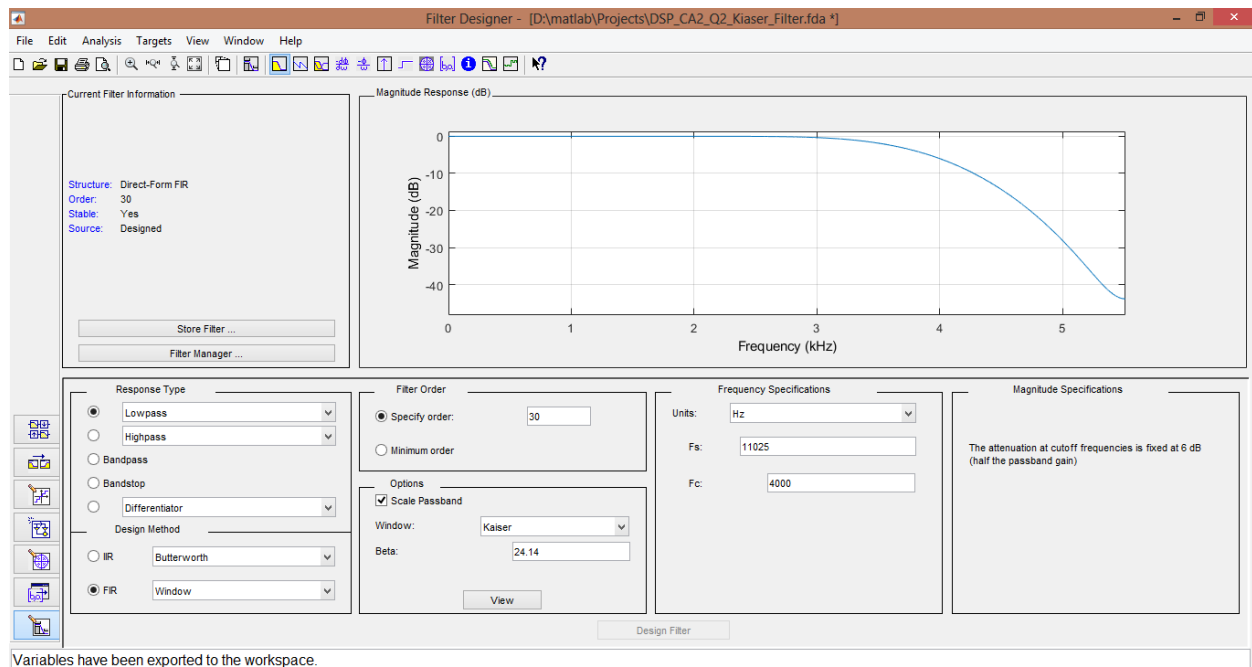
عبور می دهیم.



```
10
11 [Rmm, lags] = xcorr(input_audio, 'unbiased');
12 Rmm = Rmm(lags > 0);
13 lags = lags(lags > 0);
14
15 figure
16 plot(lags/Fs, Rmm)
17 xlabel('Lag (s)')
18 title('Xcorr');
19 [peaks, locations_of_peaks] = findpeaks(Rmm, lags, 'MinPeakHeight', 0.002);
20
21 delays = zeros(1, max(locations_of_peaks) + 1);
22 delays(1) = 1;
23 delays(locations_of_peaks(1) + 1) = peaks(1) / Rmm(1);
24 delays(locations_of_peaks(2) + 1) = peaks(2) / Rmm(1);
25
26 Non_echo_audio = filter(1, delays, input_audio);
27
28 %soundsc(Non_echo_audio, Fs)
29
30 audiowrite('Non_echo.wav', Non_echo_audio, Fs);
31
32 freqz(Kiaser_Coefficients)
33
34 filtered_sound = filter(Kiaser_Coefficients, Non_echo_audio);
35
36 figure
37 t = 1 : length(filtered_sound);
38 subplot(2,1,1);
39 plot(t, Non_echo_audio);
40 title('Non echo audio');
```

تا حد زیادی اکو حذف شده است ولی هنوز مقدار کمی اکو در صدا وجود دارد.

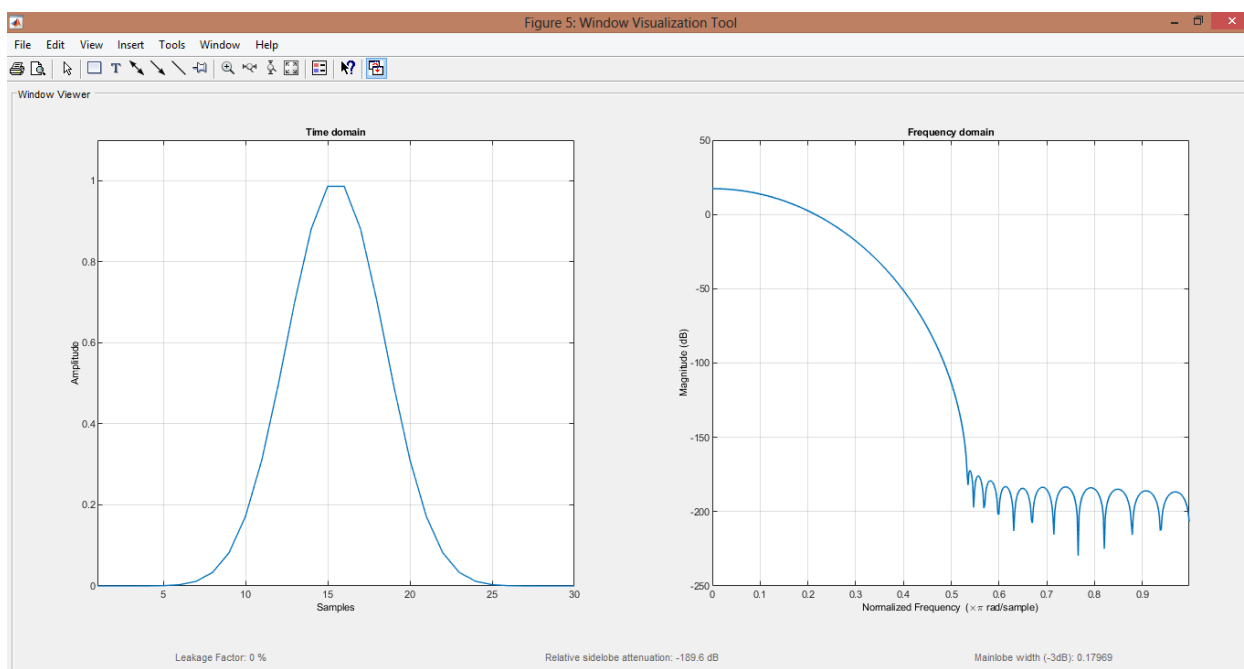
## Part 8



همان طور که مشاهده می شود فیلتر خروجی، طبق تنظیماتی که در فیلتر دیزاین متلب کرده بودیم، پایین گذر است و فرکانس قطع آن (اگر به نمودار دقت کنید در فرکانس 0.72 اندازه فیلتر منفی 3 دسیبل شده است) نیز 0.72 است که درواقع نسبت 4000 (فرکانس قطعی که تنظیم کرده بودیم) به نصف فرکانس



نمونه برداری است. از جایی که این فیلتر توسط پنجره کیسر (عکس زیر) ساخته شده، فیلتر خروجی هم مانند شکل پنجره کیسر درآمده است.

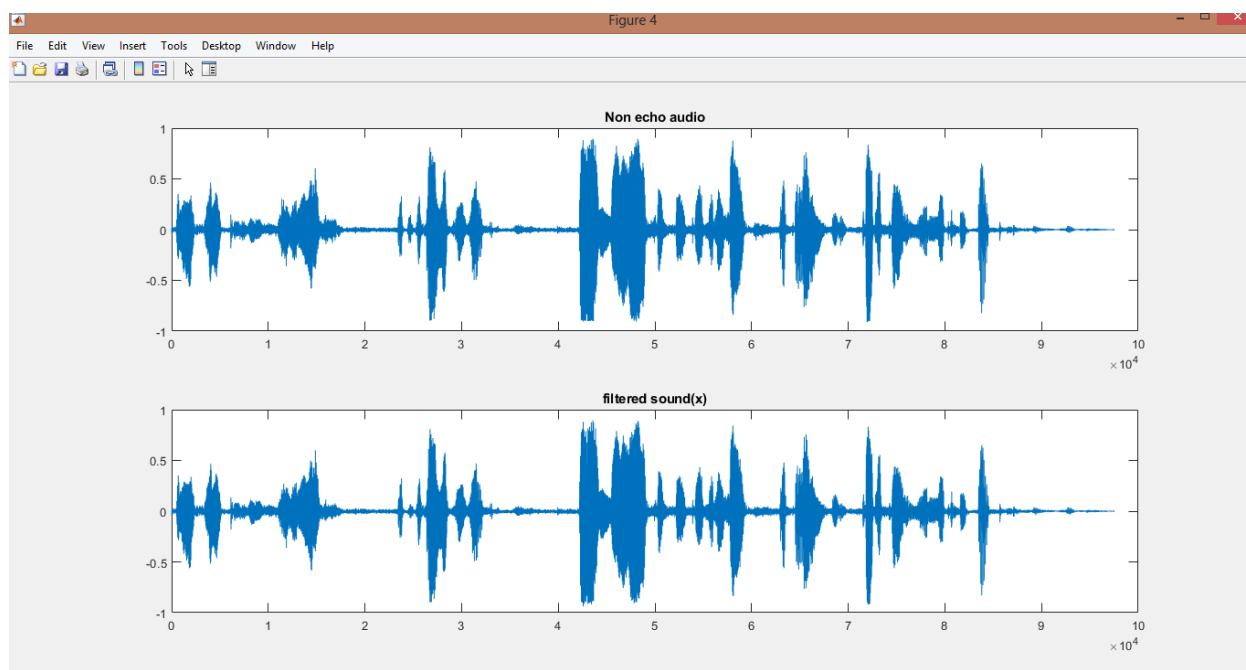


## Part 9

بله.

نویز کم شده است ولی هنوز مقداری نویز در سیگنال وجود دارد. علت این امر این است که بخشی از نویز دارای فرکانسی است که در محدوده عبور فیلتر (300 تا 3800 هرتز) است و در نتیجه همراه با سیگنال بدون نویز، از فیلتر عبور می کند و منجر به شنیده شدن مقداری نویز در سیگنال خروجی می شود.

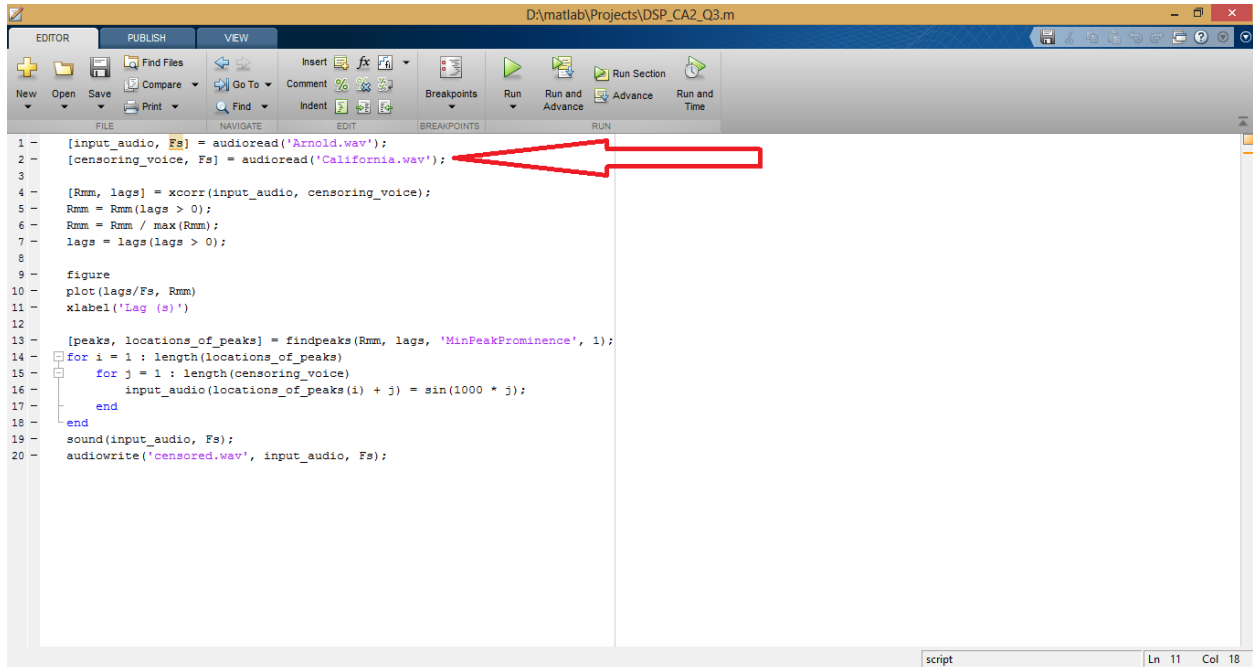
## Part 10



سیگنال دارای نویز، همان سیگنال بدون نویز هست که نویز (سیگنال سینوسی با فرکانس مشخص) با آن جمع شده است. ازجایی که مقدار نویز کم (بین 1- تا 1) است و مقدار سیگنال اصلی در حدود 10000 هست، نویز مقدار سیگنال اصلی را خیلی تغییر نمی دهد و در نتیجه اثرش دیده نمی شود. به همین دلیل در تصویر بالا شکل سیگنال بدون نویز و سیگنال دارای نویز شبیه به هم شده است.

# Question 3

## Part 1, 2

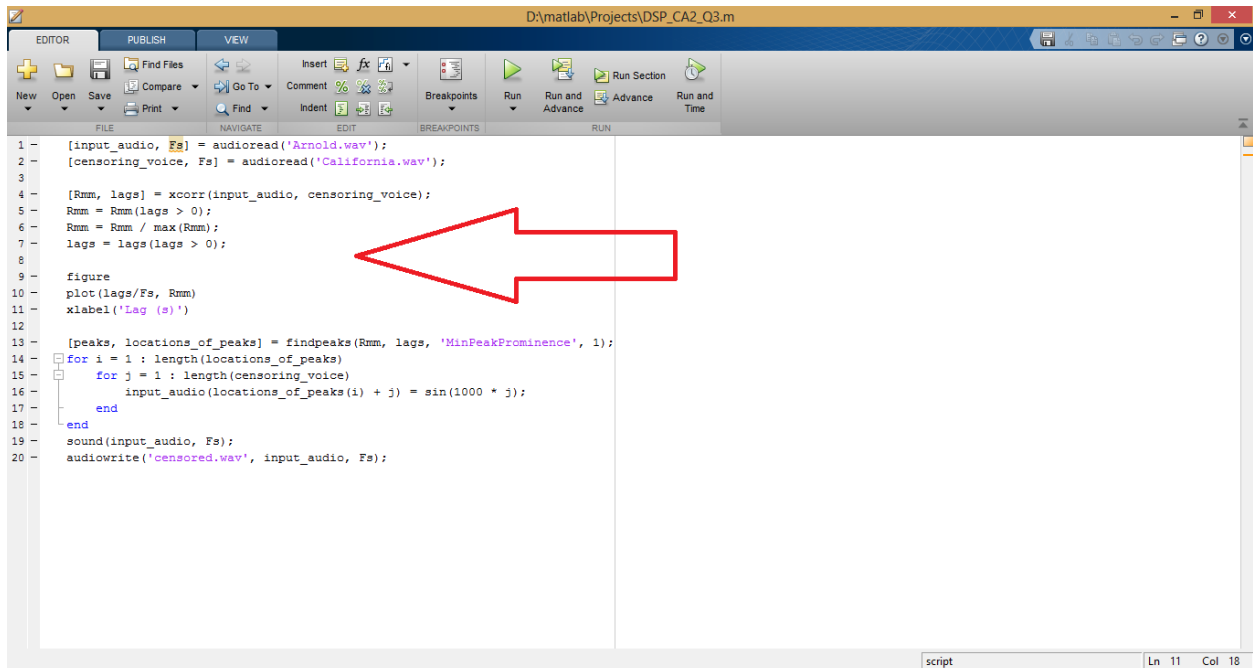


The image shows a MATLAB editor window titled "D:\matlab\Projects\DSP\_CA2\_Q3.m". The script contains the following code:

```
1 [input_audio, Fs] = audioread('Arnold.wav');
2 [censoring_voice, Fs] = audioread('California.wav');
3
4 [Rmm, lags] = xcorr(input_audio, censoring_voice);
5 Rmm = Rmm(lags > 0);
6 Rmm = Rmm / max(Rmm);
7 lags = lags(lags > 0);
8
9 figure
10 plot(lags/Fs, Rmm)
11 xlabel('Lag (s)')
12
13 [peaks, locations_of_peaks] = findpeaks(Rmm, lags, 'MinPeakProminence', 1);
14 for i = 1 : length(locations_of_peaks)
15     for j = 1 : length(censoring_voice)
16         input_audio(locations_of_peaks(i) + j) = sin(1000 * j);
17     end
18 end
19 sound(input_audio, Fs);
20 audiowrite('censored.wav', input_audio, Fs);
```

A red arrow points to line 2, which is: `[censoring_voice, Fs] = audioread('California.wav');`

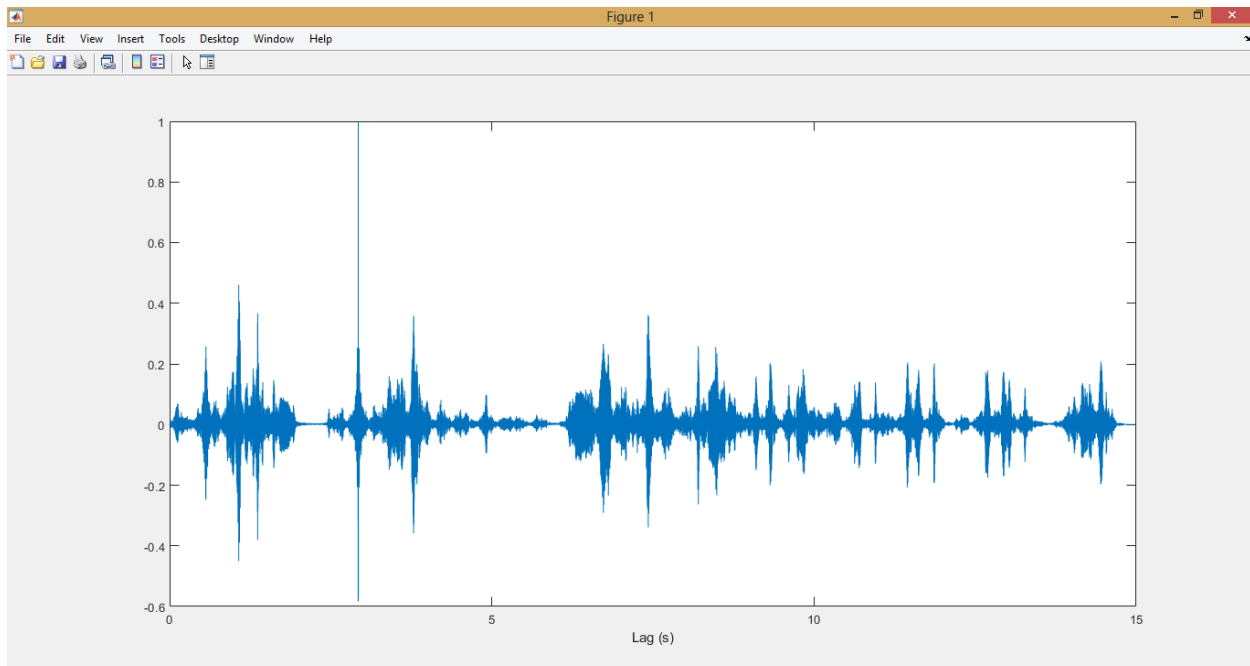
## Part 3



The image shows a MATLAB editor window titled "D:\matlab\Projects\DSP\_CA2\_Q3.m". The script contains the following code:

```
1 [input_audio, Fs] = audioread('Arnold.wav');
2 [censoring_voice, Fs] = audioread('California.wav');
3
4 [Rmm, lags] = xcorr(input_audio, censoring_voice);
5 Rmm = Rmm(lags > 0);
6 Rmm = Rmm / max(Rmm);
7 lags = lags(lags > 0);
8
9 figure
10 plot(lags/Fs, Rmm)
11 xlabel('Lag (s)')
12
13 [peaks, locations_of_peaks] = findpeaks(Rmm, lags, 'MinPeakProminence', 1);
14 for i = 1 : length(locations_of_peaks)
15     for j = 1 : length(censoring_voice)
16         input_audio(locations_of_peaks(i) + j) = sin(1000 * j);
17     end
18 end
19 sound(input_audio, Fs);
20 audiowrite('censored.wav', input_audio, Fs);
```

A red arrow points to line 7, which is: `lags = lags(lags > 0);`



## Part 4

```
D:\matlab\Projects\DSP_CA2_Q3.m

EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment % Indent Breakpoints Run Run and Advance Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN

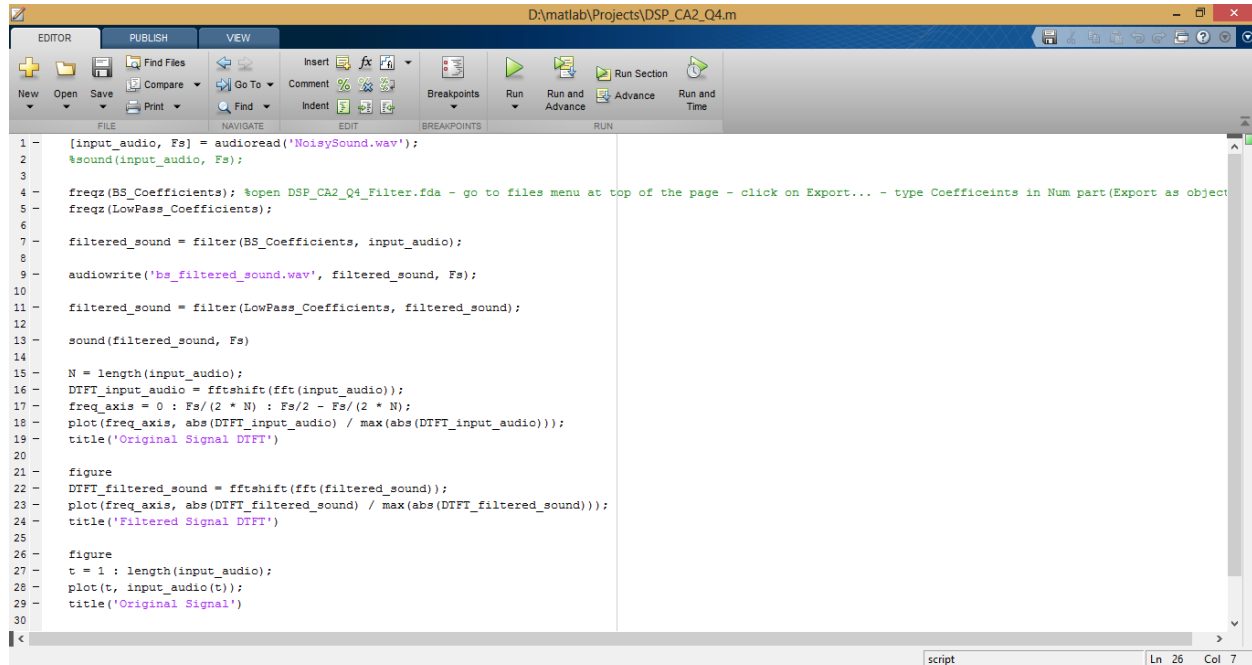
1 [input_audio, Fs] = audioread('Arnold.wav');
2 [censoring_voice, Fs] = audioread('California.wav');
3
4 [Rmm, lags] = xcorr(input_audio, censoring_voice);
5 Rmm = Rmm(lags > 0);
6 Rmm = Rmm / max(Rmm);
7 lags = lags(lags > 0);
8
9 figure
10 plot(lags/Fs, Rmm)
11 xlabel('Lag (s)')
12
13 [peaks, locations_of_peaks] = findpeaks(Rmm, lags, 'MinPeakProminence', 1);
14 for i = 1 : length(locations_of_peaks)
15     for j = 1 : length(censoring_voice)
16         input_audio(locations_of_peaks(i) + j) = sin(1000 * j);
17     end
18 end
19 sound(input_audio, Fs);
20 audiowrite('censored.wav', input_audio, Fs);

script Ln 11 Col 18
```

علت اینکه در صوت خروجی فقط اولین پیک(اولین جایی که عبارت California گفته شده) شناسایی و سانسور شده این است که xcorr برای هم اندازه شدن دو ورودی، تعدادی 0 به انتهای سیگنال سانسور اضافه می کند. در نتیجه correlation بین سیگنال و صدای کالیفورنیا که بعدش 0 هست محاسبه شده که این باعث می شود که فقط پیک اول شناسایی شود.

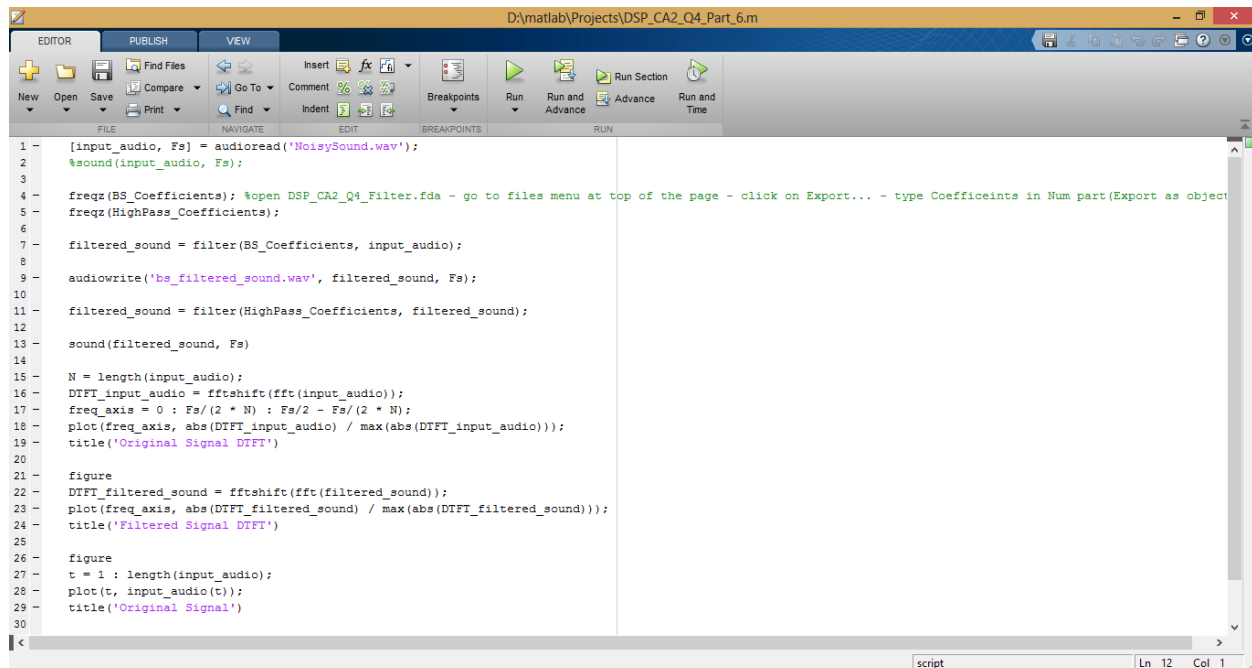
# Question 4

## Matlab Code



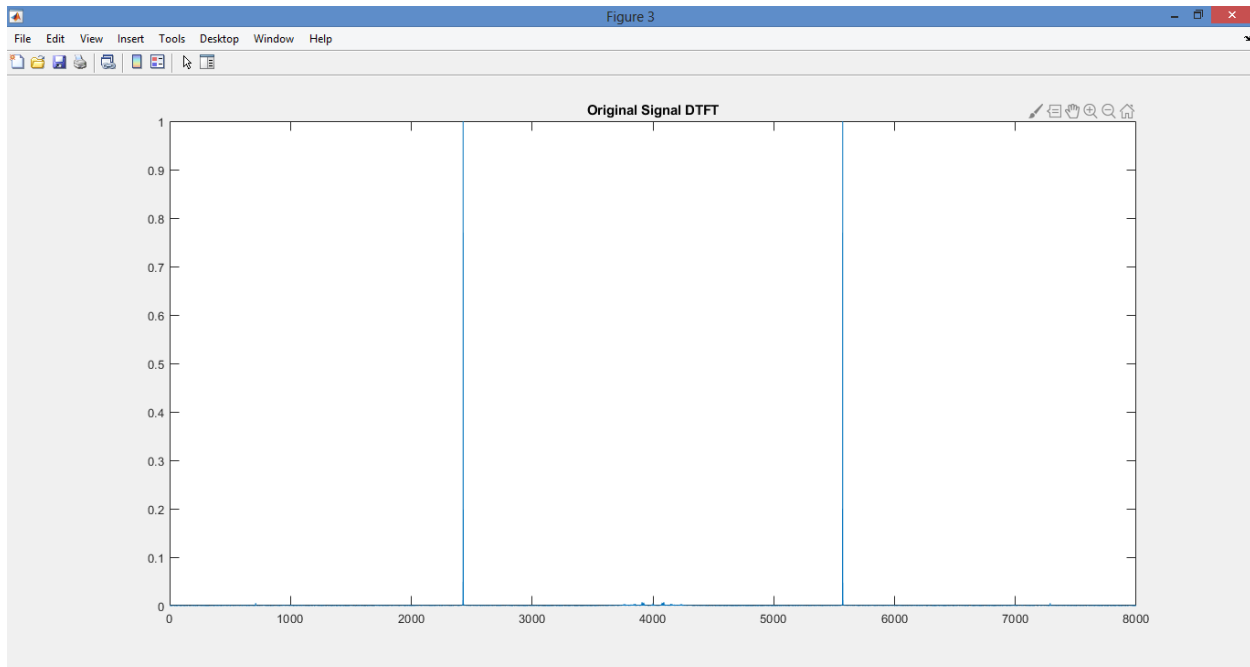
```
1 [input_audio, Fs] = audioread('NoisySound.wav');
2 %sound(input_audio, Fs);
3
4 freqz(BS_Coefficients); %open DSP_CA2_Q4_Filter.fda - go to files menu at top of the page - click on Export... - type Coefficeints in Num part(Export as object
5 freqz(LowPass_Coefficients);
6
7 filtered_sound = filter(BS_Coefficients, input_audio);
8
9 audiowrite('bs_filtered_sound.wav', filtered_sound, Fs);
10
11 filtered_sound = filter(LowPass_Coefficients, filtered_sound);
12
13 sound(filtered_sound, Fs)
14
15 N = length(input_audio);
16 DTFT_input_audio = fftshift(fft(input_audio));
17 freq_axis = 0 : Fs/(2 * N) : Fs/2 - Fs/(2 * N);
18 plot(freq_axis, abs(DTFT_input_audio) / max(abs(DTFT_input_audio)));
19 title('Original Signal DTFT')
20
21 figure
22 DTFT_filtered_sound = fftshift(fft(filtered_sound));
23 plot(freq_axis, abs(DTFT_filtered_sound) / max(abs(DTFT_filtered_sound)));
24 title('Filtered Signal DTFT')
25
26 figure
27 t = 1 : length(input_audio);
28 plot(t, input_audio(t));
29 title('Original Signal')
30
```

## Part 6 Matlab Code



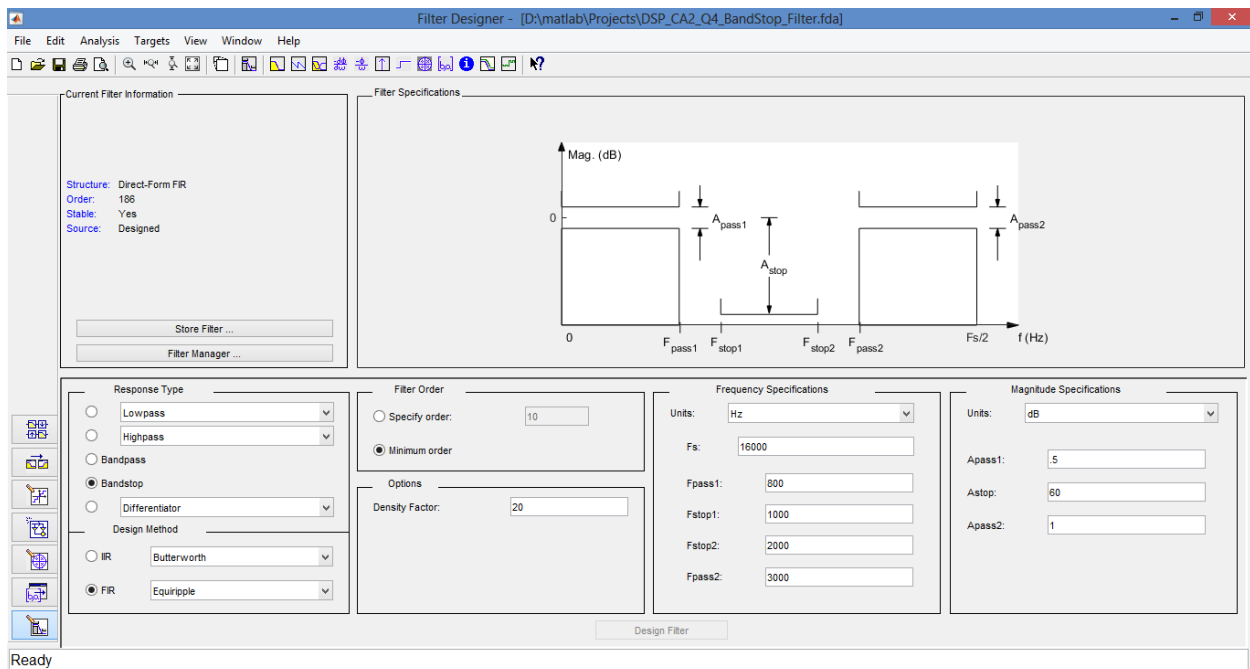
```
1 [input_audio, Fs] = audioread('NoisySound.wav');
2 %sound(input_audio, Fs);
3
4 freqz(BS_Coefficients); %open DSP_CA2_Q4_Filter.fda - go to files menu at top of the page - click on Export... - type Coefficeints in Num part(Export as object
5 freqz(HighPass_Coefficients);
6
7 filtered_sound = filter(BS_Coefficients, input_audio);
8
9 audiowrite('bs_filtered_sound.wav', filtered_sound, Fs);
10
11 filtered_sound = filter(HighPass_Coefficients, filtered_sound);
12
13 sound(filtered_sound, Fs)
14
15 N = length(input_audio);
16 DTFT_input_audio = fftshift(fft(input_audio));
17 freq_axis = 0 : Fs/(2 * N) : Fs/2 - Fs/(2 * N);
18 plot(freq_axis, abs(DTFT_input_audio) / max(abs(DTFT_input_audio)));
19 title('Original Signal DTFT')
20
21 figure
22 DTFT_filtered_sound = fftshift(fft(filtered_sound));
23 plot(freq_axis, abs(DTFT_filtered_sound) / max(abs(DTFT_filtered_sound)));
24 title('Filtered Signal DTFT')
25
26 figure
27 t = 1 : length(input_audio);
28 plot(t, input_audio(t));
29 title('Original Signal')
30
```

## Part 1

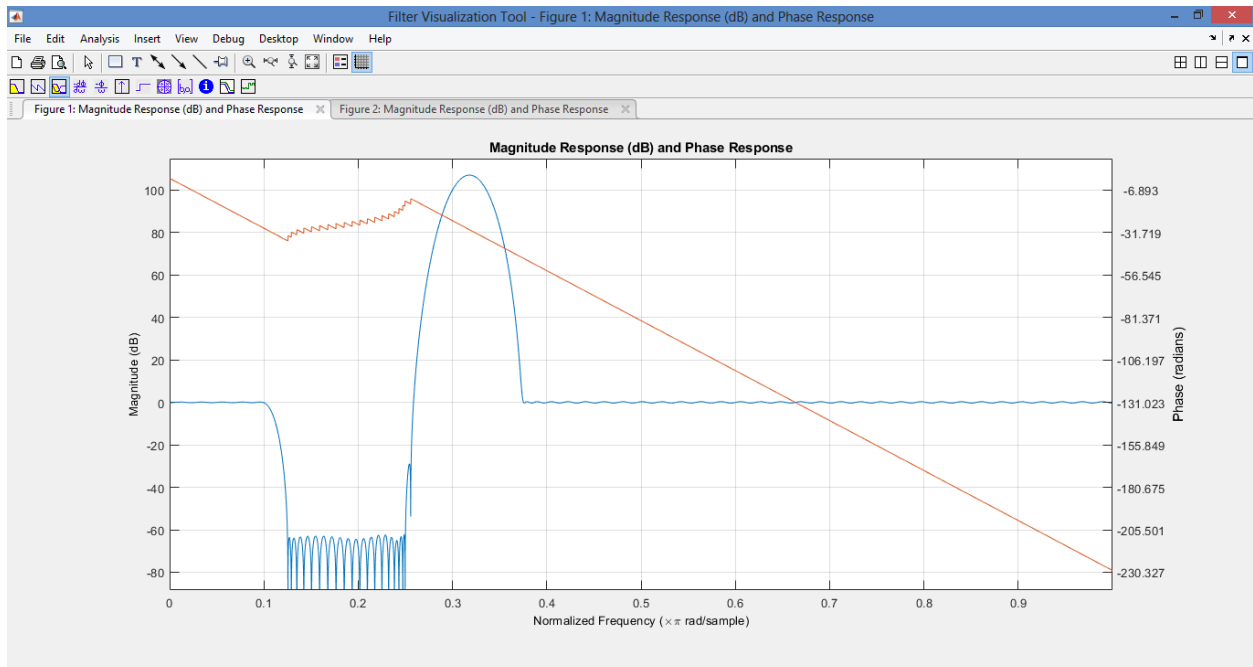


=> Noise Frequency = 1500 Hz

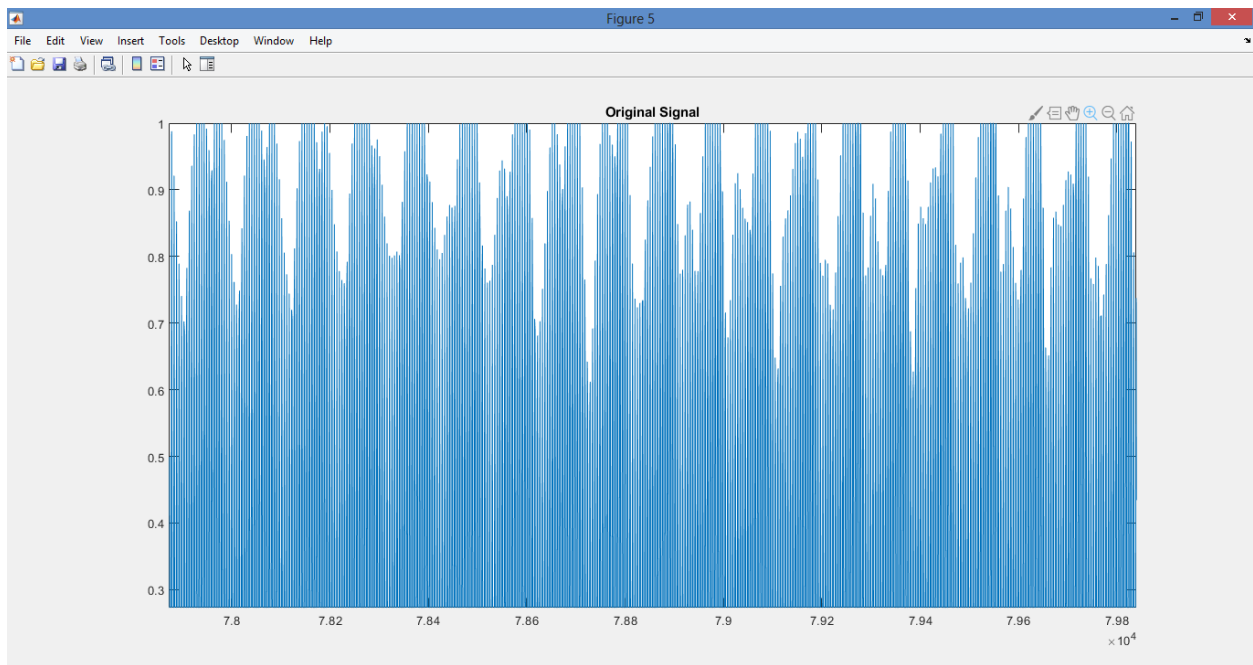
## Part 2



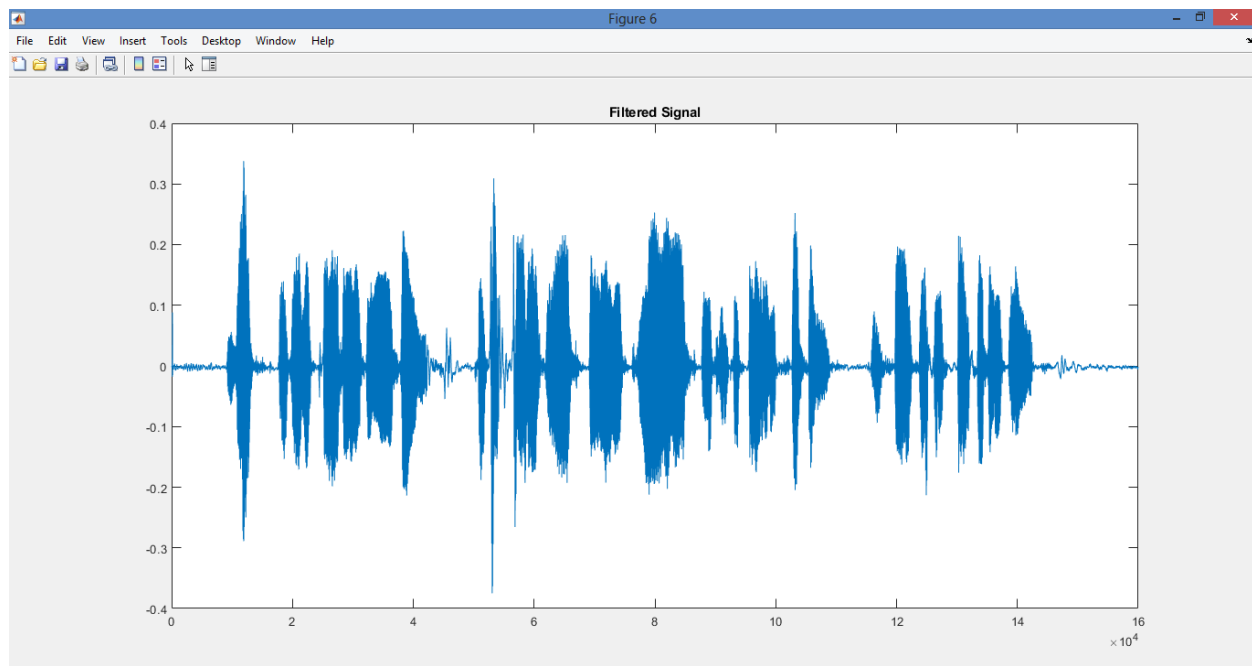
## Part 3



## Part 5 – Part b

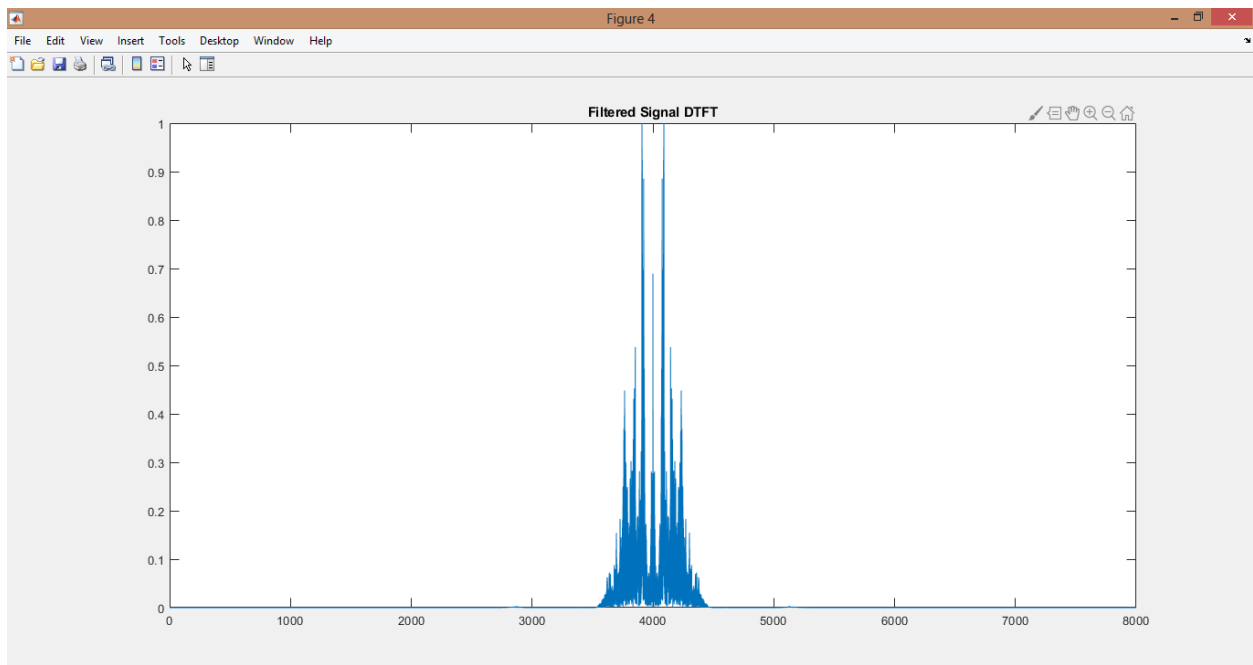
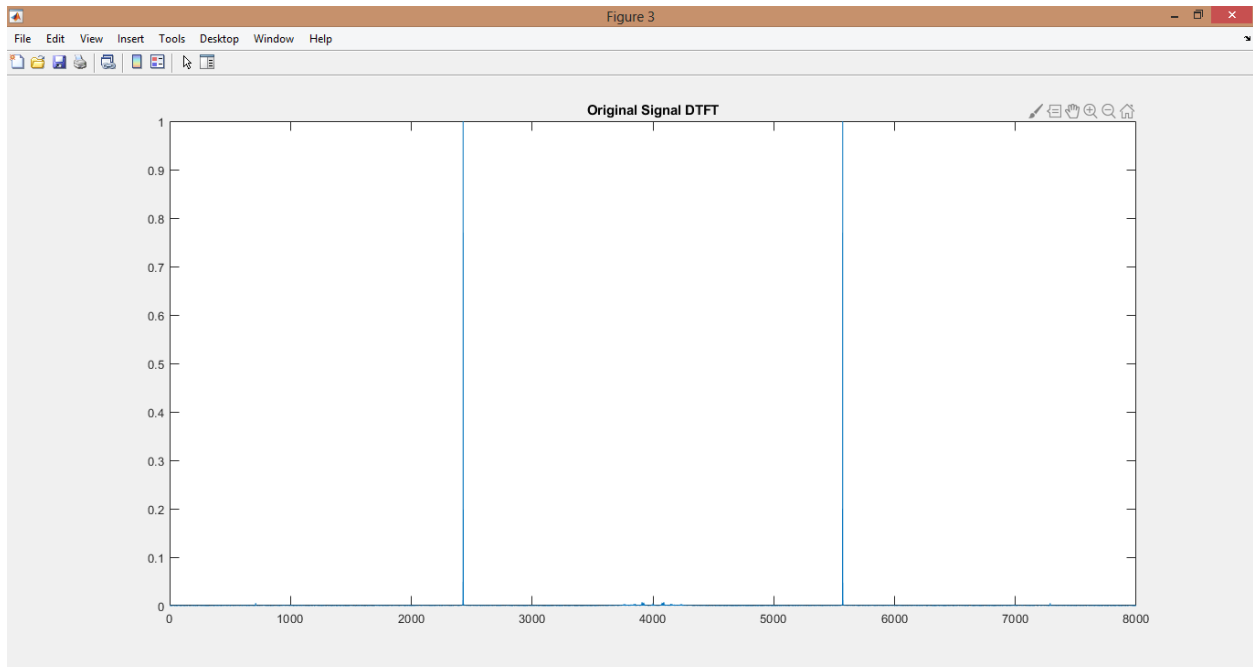




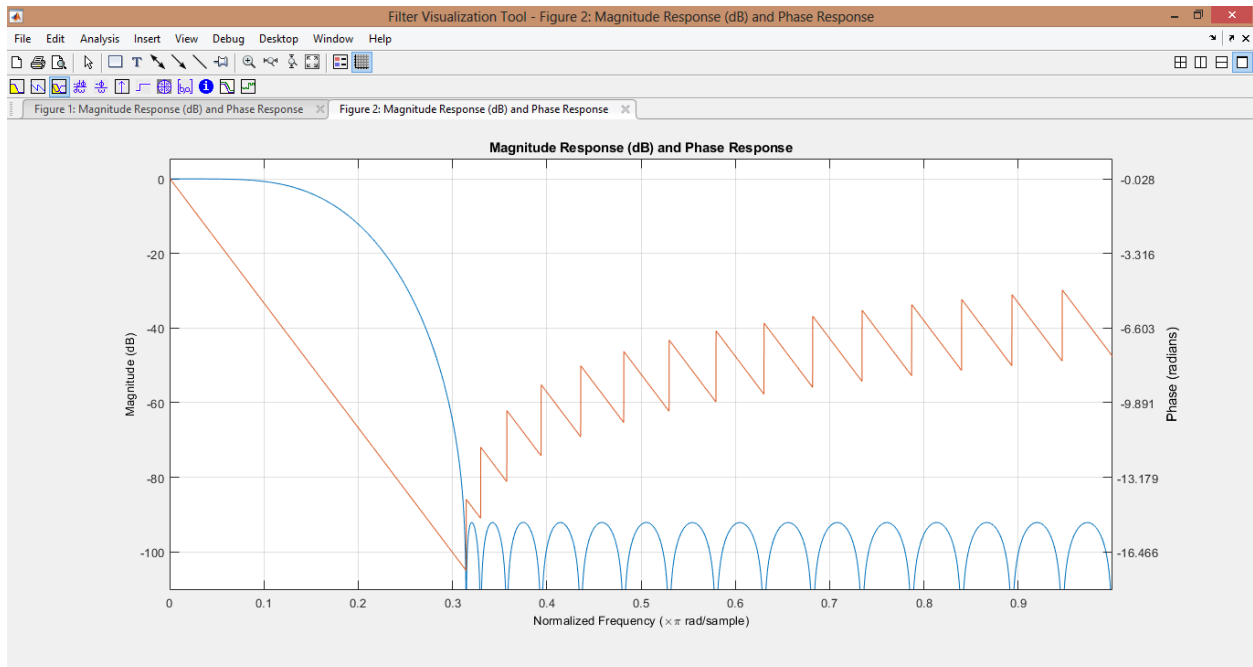


سیگنال اصلی در واقع ترکیب شده سیگنال فیلتر شده (تصویر بالا) با یک سیگنال سینوسی تک تون هست و به این دلیل، شکل حوزه زمان سیگنال اصلی به شکل سینوسی درآمده است. در واقع ابتدا با اعمال شدن فیلتر میان نگذر، نویز (سیگنال سینوسی تک تون) از سیگنال اصلی حذف شده است. در ادامه با اعمال فیلتر پایین گذر، فرکانس های بالا سیگنال اصلی حذف شده و فرکانس های کوچک سیگنال که حاوی سیگنال بدون نویز هستند از فیلتر عبور می کنند و در نهایت سیگنال بدون نویز در سیگنال فیلتر شده بدست می آید.

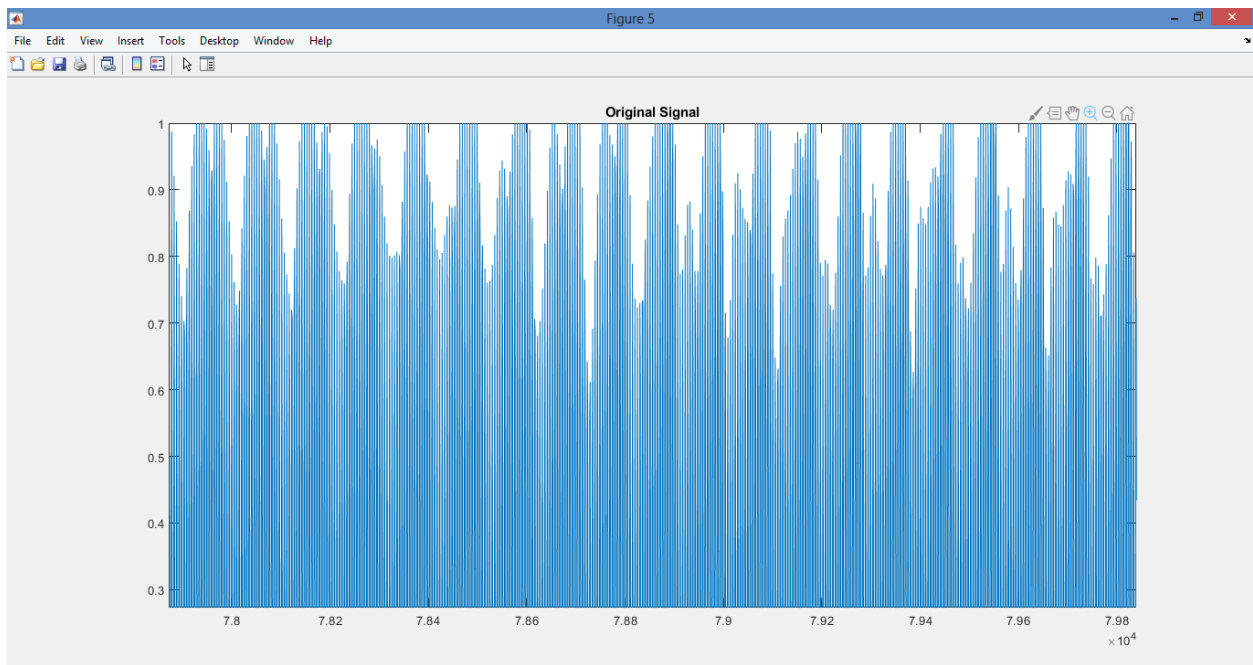
## Part 5 – Part g

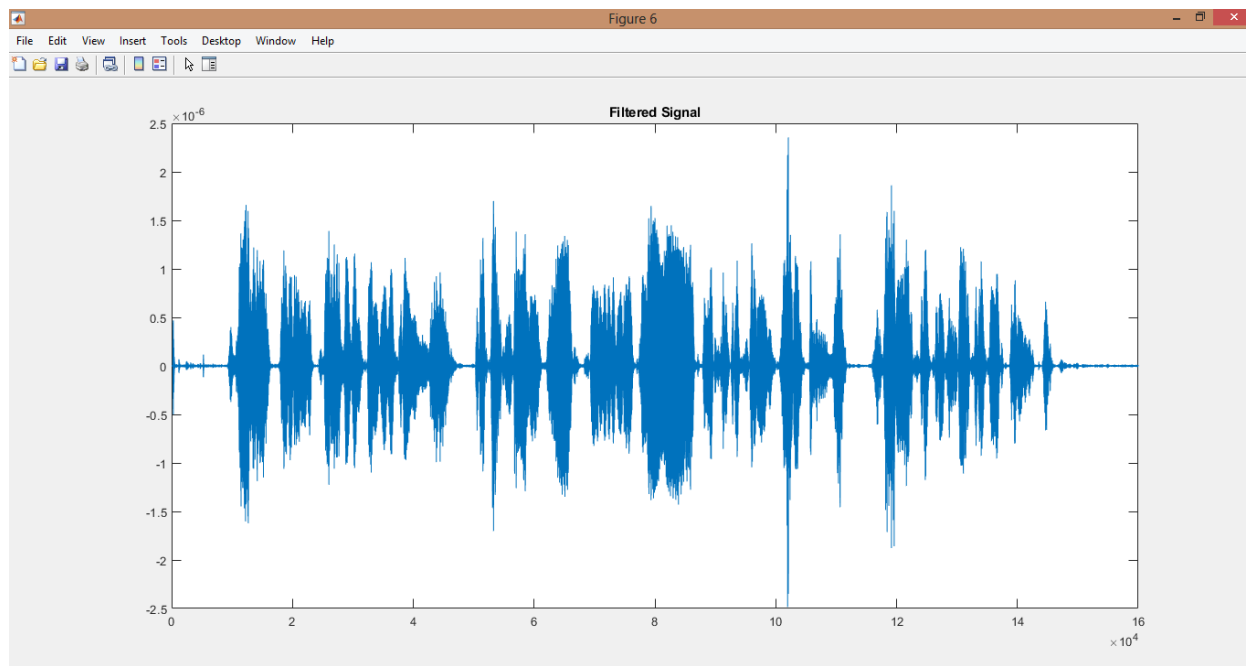


## Part 5 – Part d



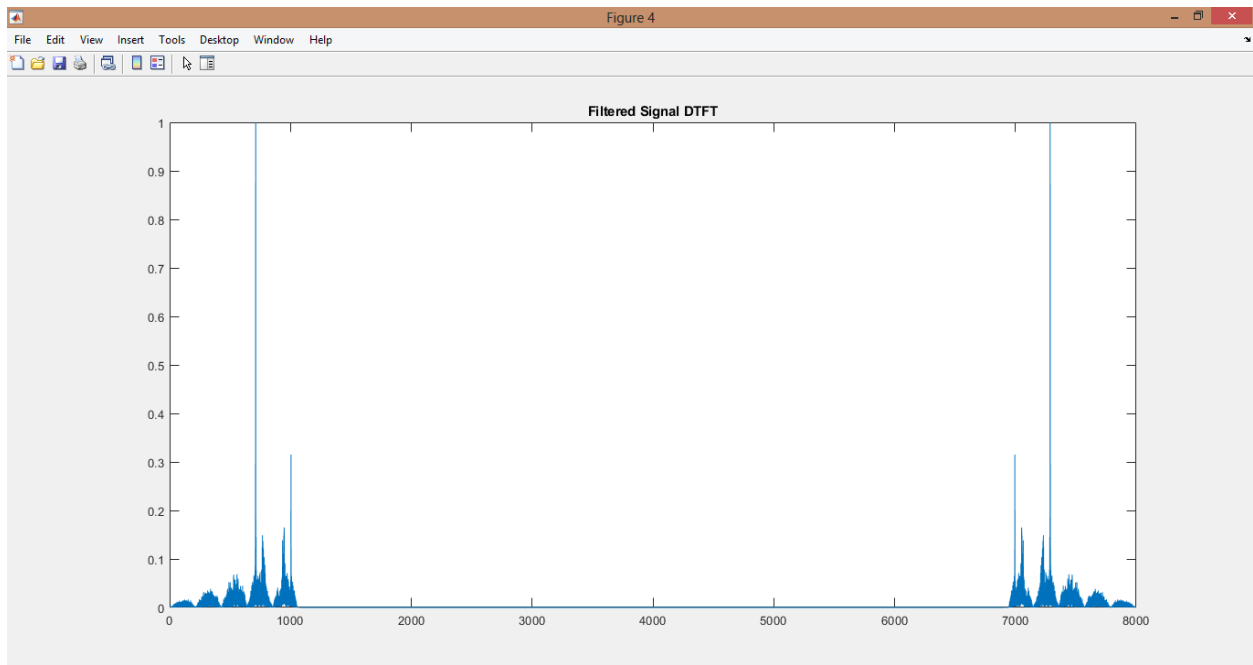
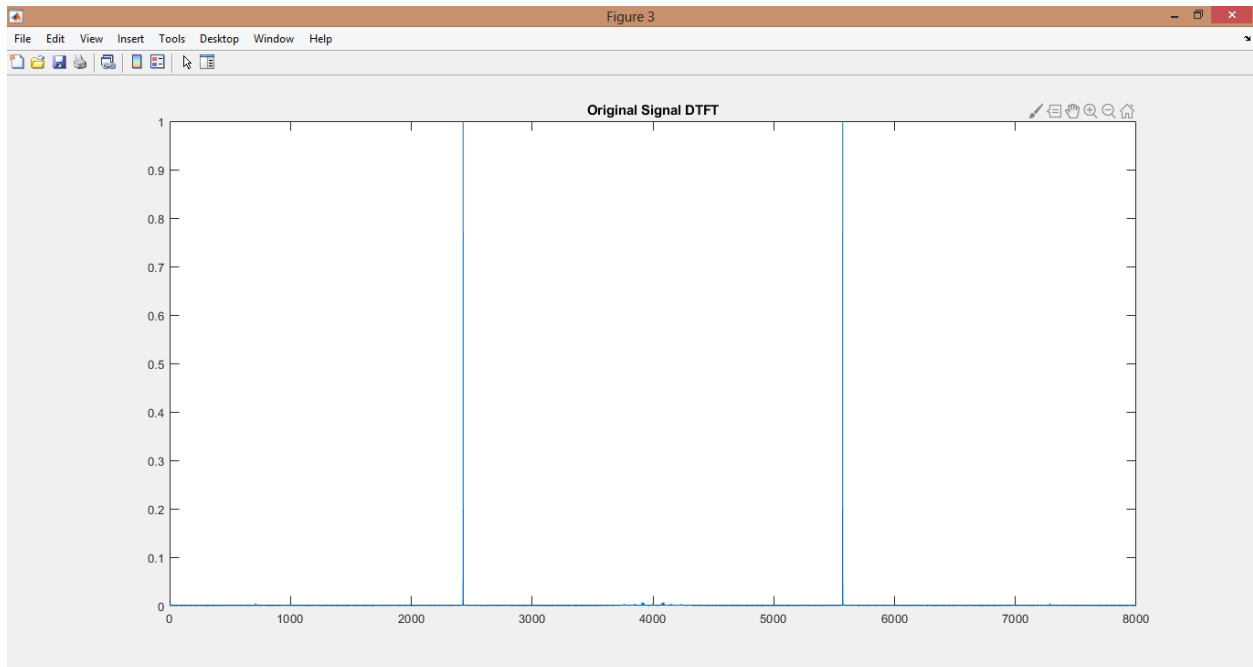
## Part 6 – Part b





تفاوتی که سیگنال فیلتر شده نسبت به سیگنال اصلی کرده است، این است که ابتدا با اعمال شدن فیلتر میان گذر، نویز (سیگنال سینوسی تک تون) از سیگنال اصلی حذف شده است. در ادامه با اعمال فیلتر بالا گذر، فرکانس های بالا سیگنال اصلی از فیلتر عبور داده شده و فرکانس های کوچک سیگنال از سیگنال اصلی حذف شده اند. (اثبات این مورد در نمودار اندازه تبدیل فوریه سیگنال فیلتر شده، در صفحه بعد قابل مشاهده است.)

## Part 6 – Part g



Part 6 – Part d

