# Signals and Systems

## Homework #3


전기정보공학부

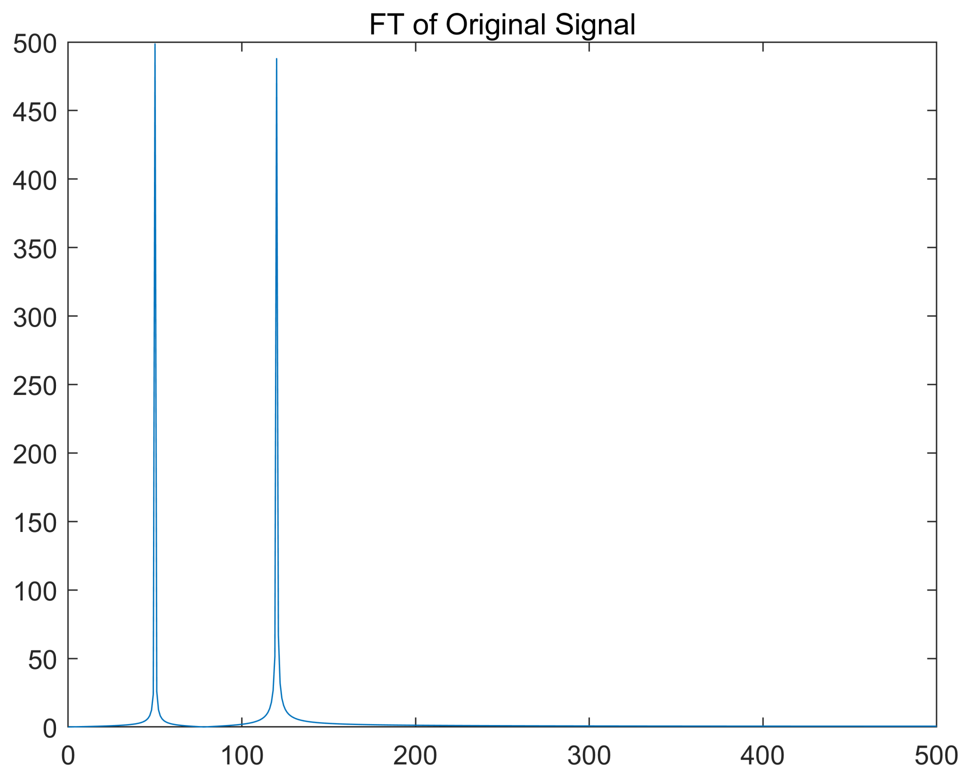
2017-17088 박찬정

# 1. Noise Filtering with Fourier Transform

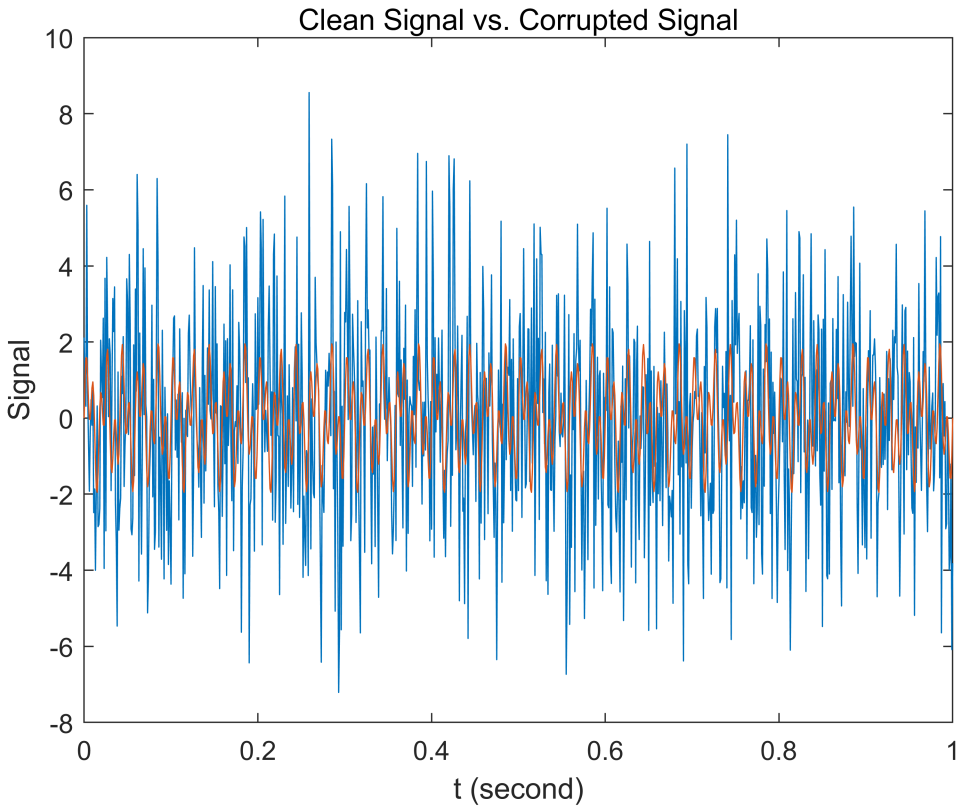
```
clearvars();
```

## 1.(a)

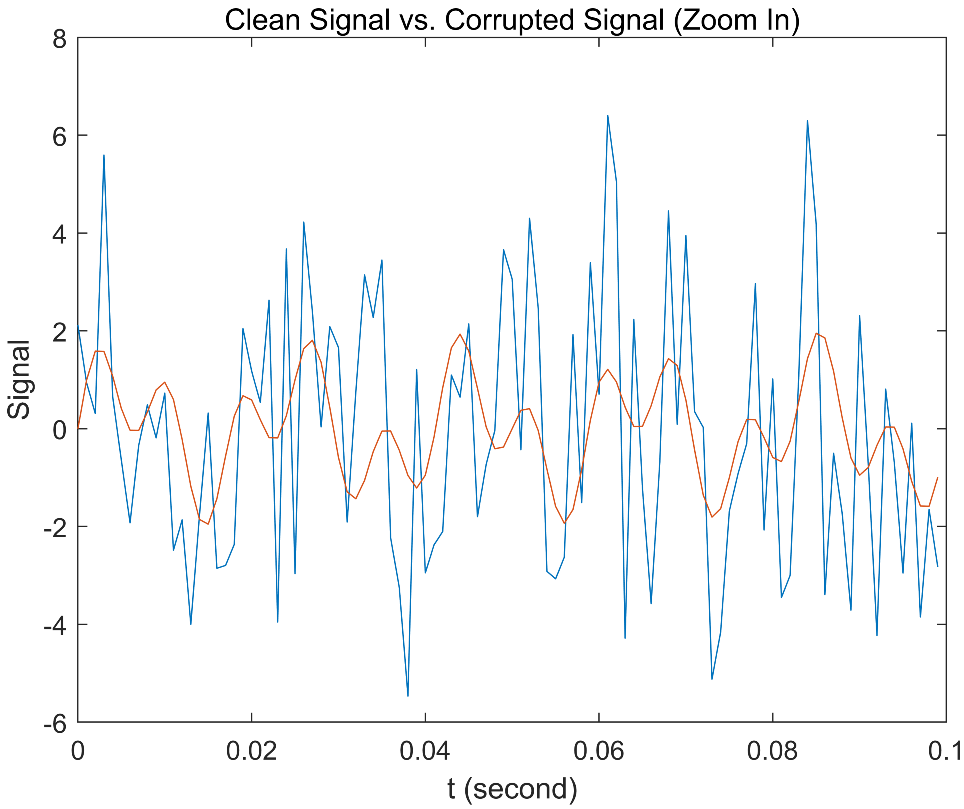
```
L = 1000;
dt = 1/L;
t = 0:dt:1;
f_clean = sin(2*pi*50*t) + sin(2*pi*120*t);
period = lcm(50, 120);
F_clean = fft(f_clean);
figure, plot(0:L/2, abs(F_clean(1:L/2+1)));
title('FT of Original Signal');
```




```
f_noise = f_clean + 2.5 * randn(size(t));
figure;
plot(t, f_noise);
hold on;
plot(t, f_clean);
hold off;
title('Clean Signal vs. Corrupted Signal')
xlabel('t (second)')
ylabel('Signal')
```

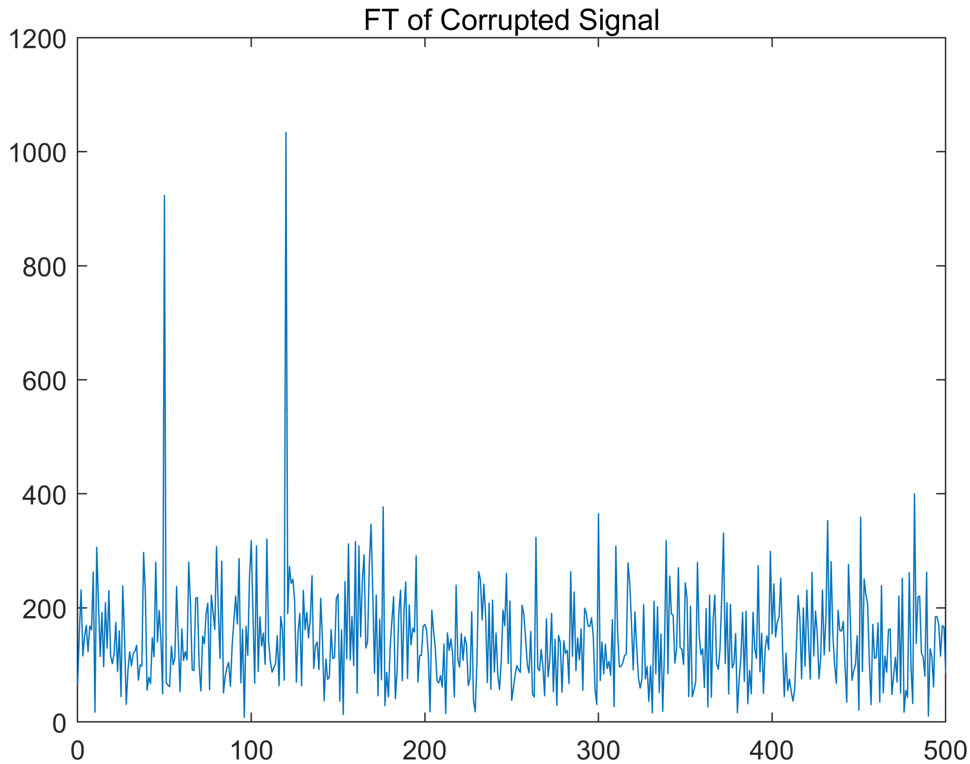



```
figure;
plot(t(1:100), f_noise(1:100));
hold on;
plot(t(1:100), f_clean(1:100));
hold off;
title('Clean Signal vs. Corrupted Signal (Zoom In)')
xlabel('t (second)')
ylabel('Signal')
```

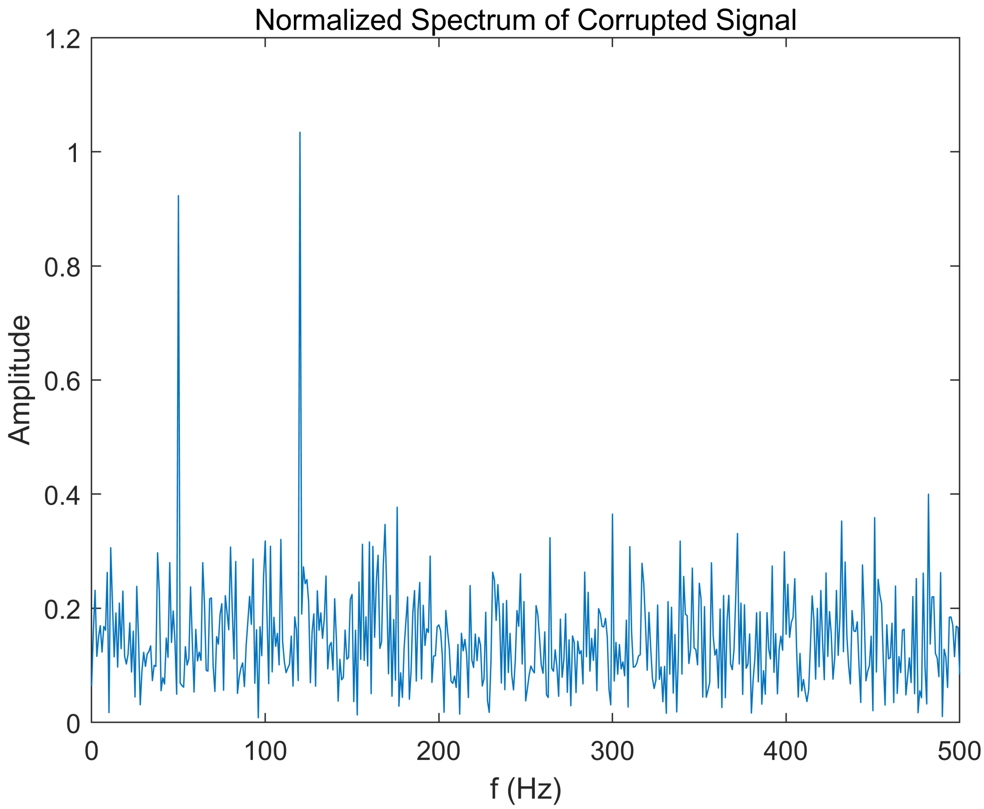



## 1.(b)

```
F_noise = fft(f_noise);
F_noise_even = abs(F_noise(1:L/2+1));
F_noise_even(2:end-1) = 2 * F_noise_even(2:end-1);
figure, plot(0:L/2, F_noise_even);
title('FT of Corrupted Signal')
```

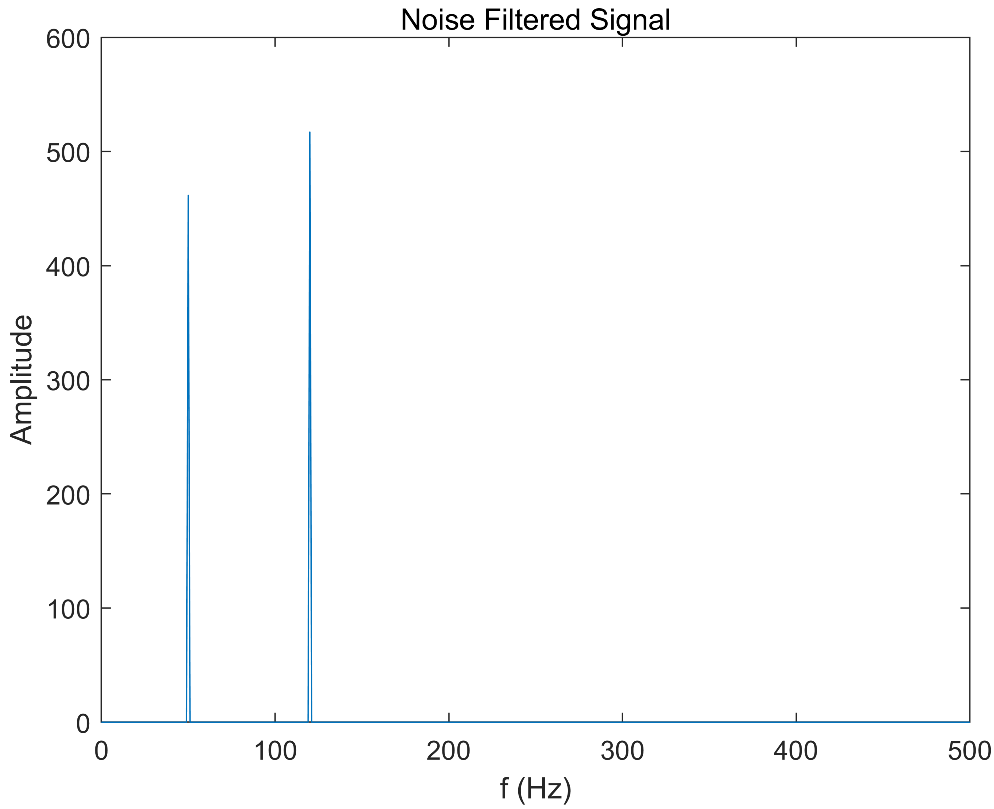



```
F_noise_even = F_noise_even/L;
figure, plot(0:L/2, F_noise_even);
title('Normalized Spectrum of Corrupted Signal')
xlabel('f (Hz)')
ylabel('Amplitude')
```

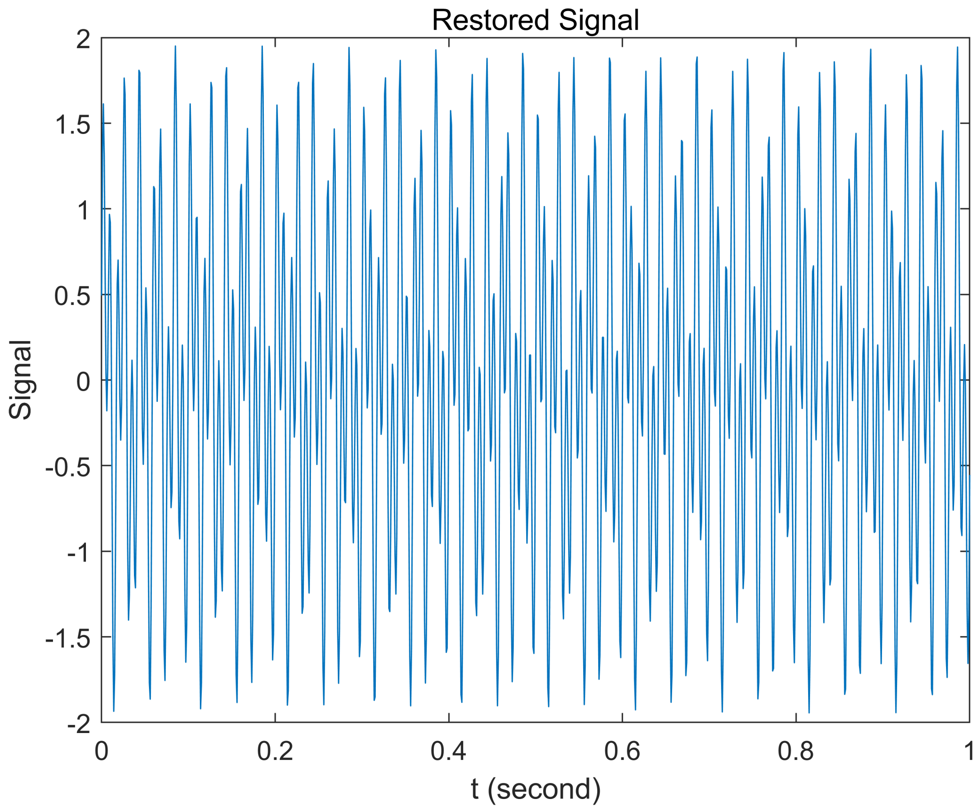



## 1.(c)

```
filter = (abs(F_noise) > L/2 * 0.8);
F_filtered = F_noise .* filter;
figure, plot(0:L/2, abs(F_filtered(1:L/2+1)));
title('Noise Filtered Signal')
xlabel('f (Hz)')
ylabel('Amplitude')
```




```
f_restored = ifft(F_filtered);
plot(t, f_restored);

title('Restored Signal')
xlabel('t (second)')
ylabel('Signal')
```

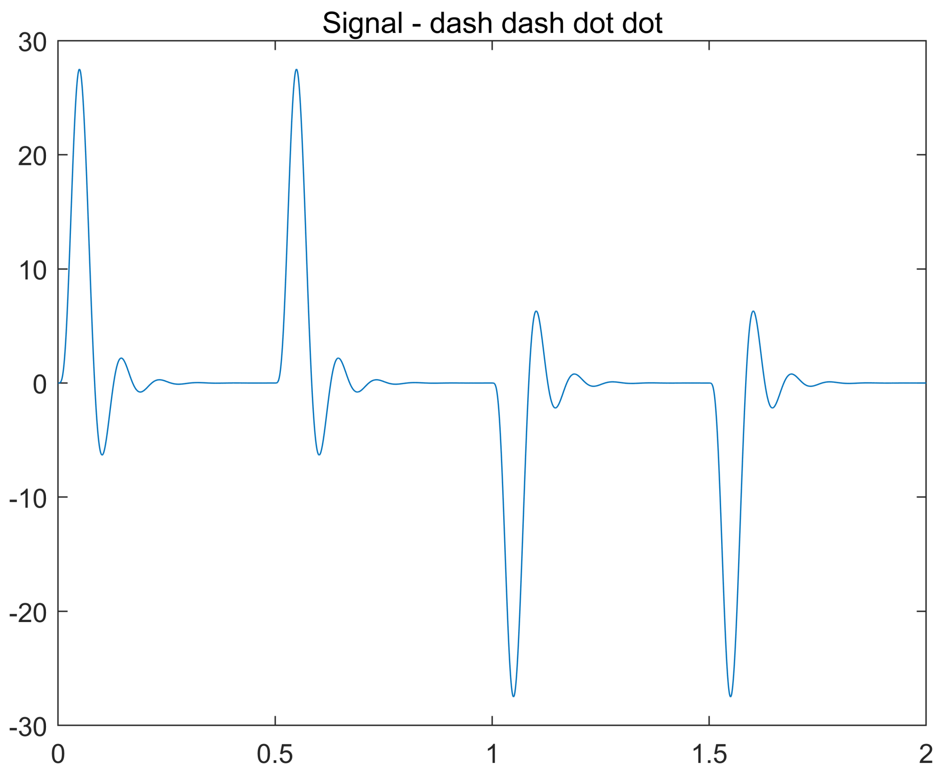

Restored Signal
Signal
t (second)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
0.2
0.4
0.6
0.8
1

# 2. Amplitude Modulation (AM) and the continuous-time Fourier transform

```
clearvars();
load data.mat
```

## 2.(a)

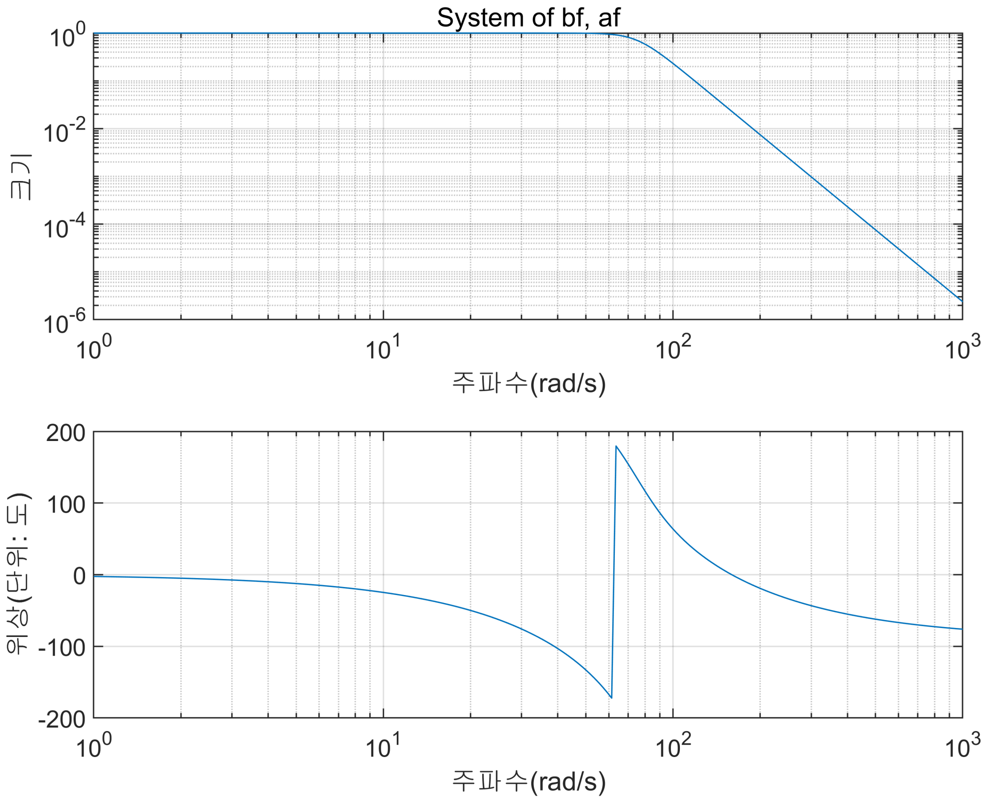
```
z = [dash dash dot dot];
figure, plot(t, z);
title('Signal - dash dash dot dot')
```




## 2.(b)

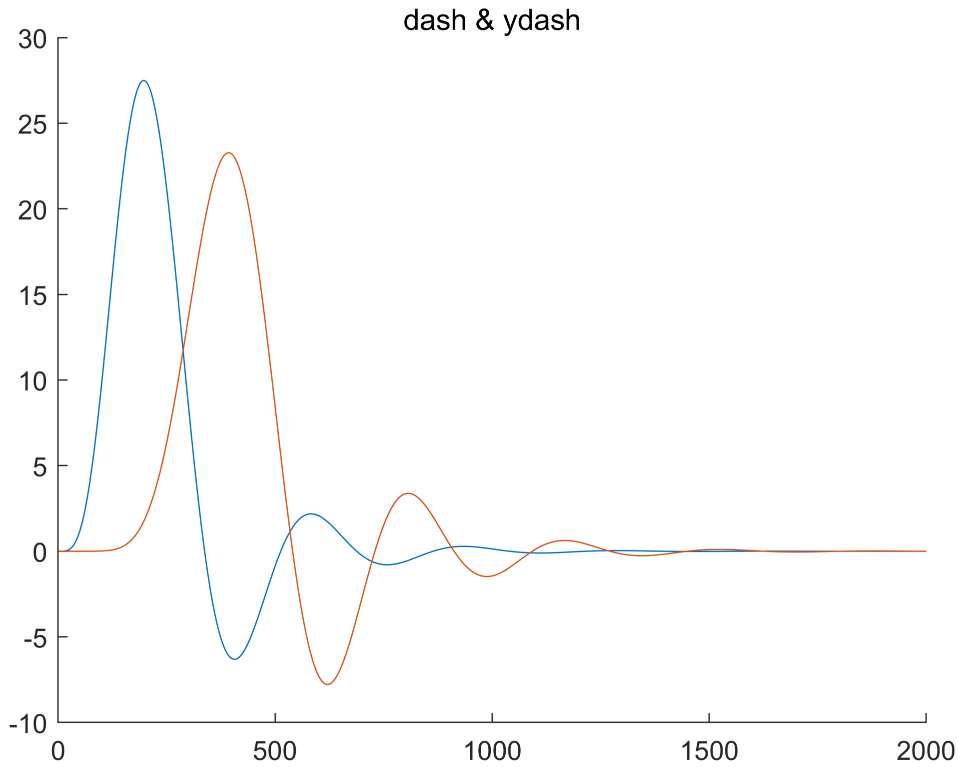
It's low-pass filter, allowing roughly f<10^2.

```
freqs(bf, af);
title('System of bf, af')
```

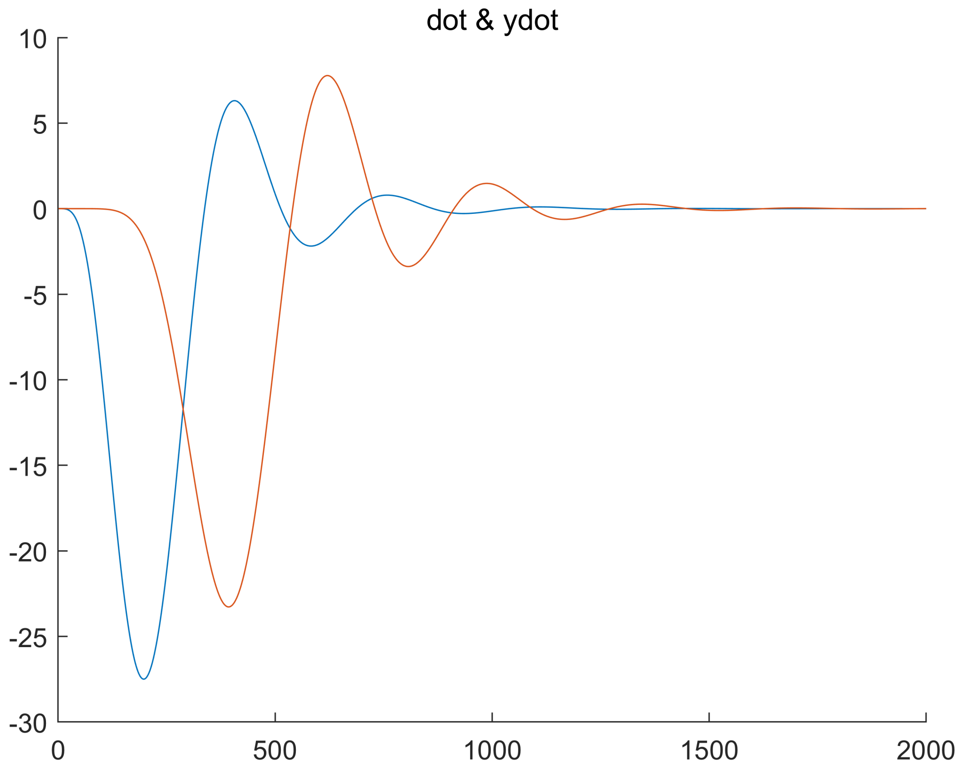



## 2.(c)

```
ydash = lsim(bf, af, dash, t(1:length(dash)));
ydot = lsim(bf, af, dot, t(1:length(dot)));

figure, hold on;
plot(dash), plot(ydash), title('dash & ydash');
hold off;
```




```
figure, hold on;
plot(dot), plot(ydot), title('dot & ydot');
hold off;
```

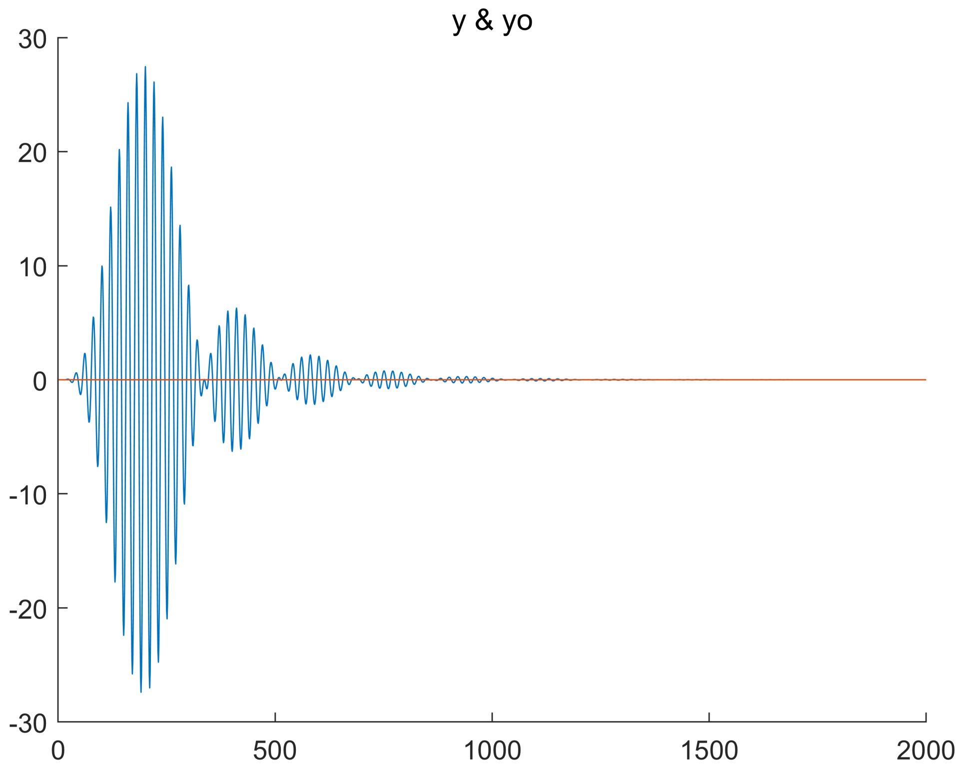



As dash and dot lies roughly within the passband of lowpass filter, no special change occurs for their figures.

## 2.(d)

Frequency 200 is out of passband of the filter, so it will shrink dramatically.

```
y = dash.*cos(2*pi*f1*t(1:length(dash)));
yo = lsim(bf, af, y, t(1:length(dash)));

figure;
hold on;
plot(y), plot(yo);
title('y & yo');
hold off;
```




Got expected result.

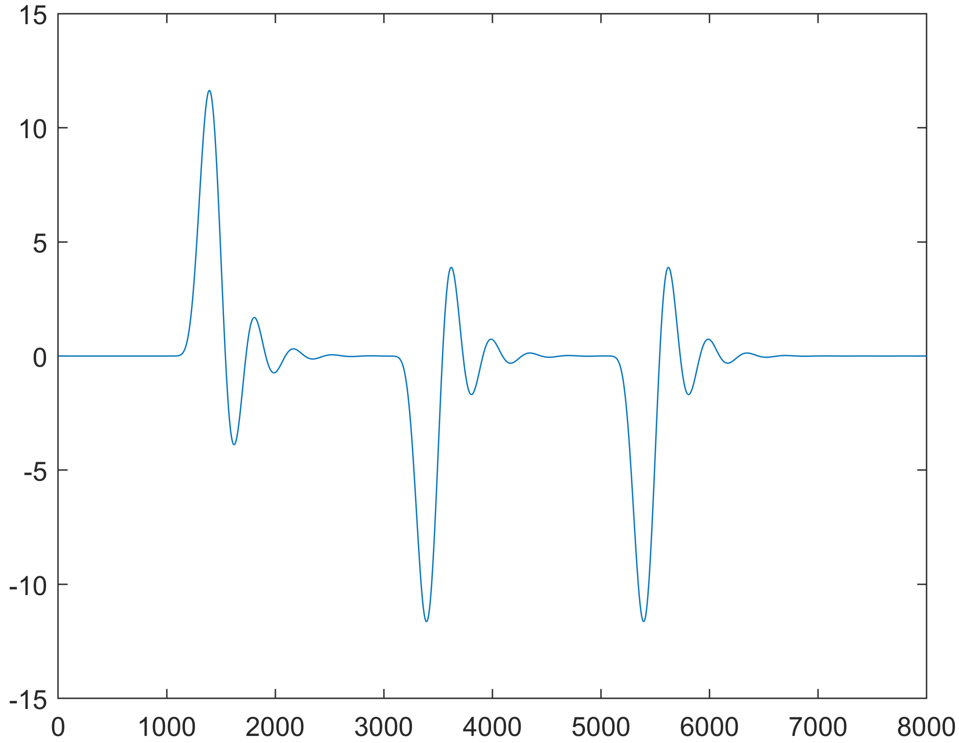
## 2.(e)

Attatched far below...

## 2.(f)

dash-dot-dot: m1 is 'D'

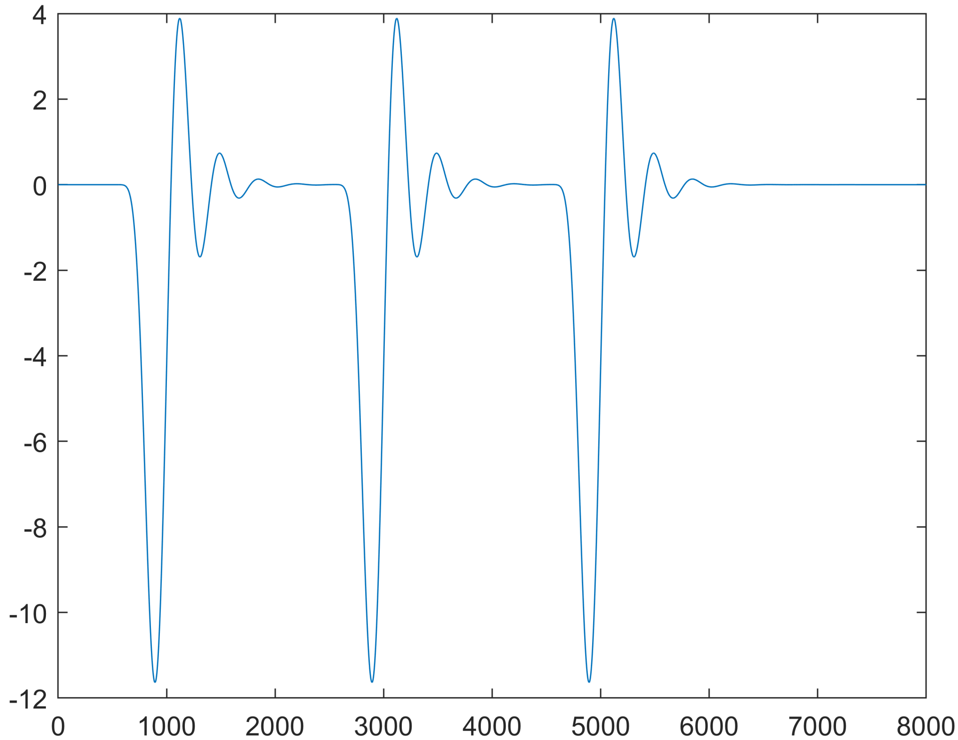
```
x1 = x.*cos(2*pi*f1*t);
y1 = lsim(bf, af, x1, t(1:length(x1)));
figure, plot(y1);
```




## 2.(g)

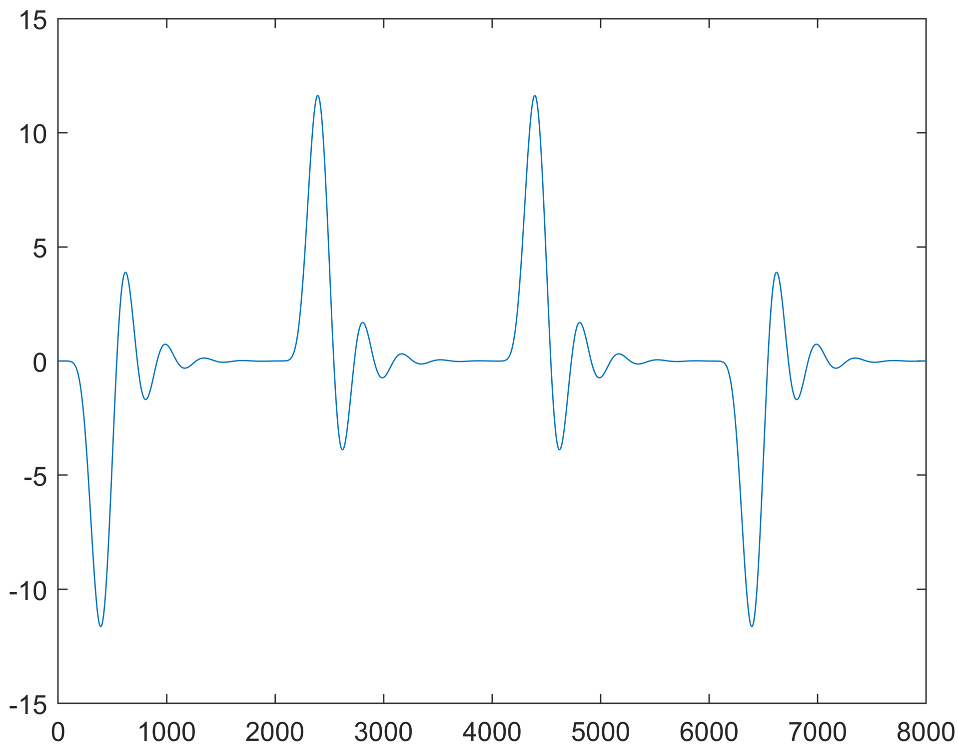
dot-dot-dot: m2 is 'S'

```
x2 = x.*sin(2*pi*f2*t);
y2 = lsim(bf, af, x2, t(1:length(x2)));
plot(y2);
```




dot-dash-dash-dot: m3 is 'P'

```
x3 = x.*sin(2*pi*f1*t);
y3 = lsim(bf, af, x3, t(1:length(x3)));
plot(y3);
```




As a result, encrypted word is 'DSP'.

I love Digital Signal Processing!

# 3. A picture from sky

```
clearvars();
```

## 3.(a)

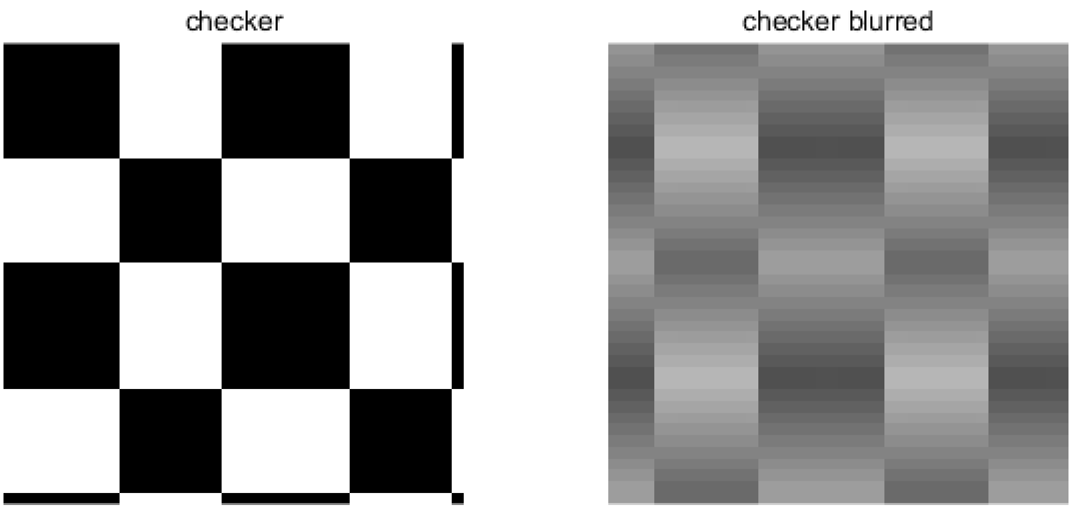
```
img = imread('mystery_img.png');
dimg = im2double(img);
dimg_gray = double(im2gray(img));
imshow(img);
title('Mysterious Image...')
```

Mysterious Image...

## 3.(b)

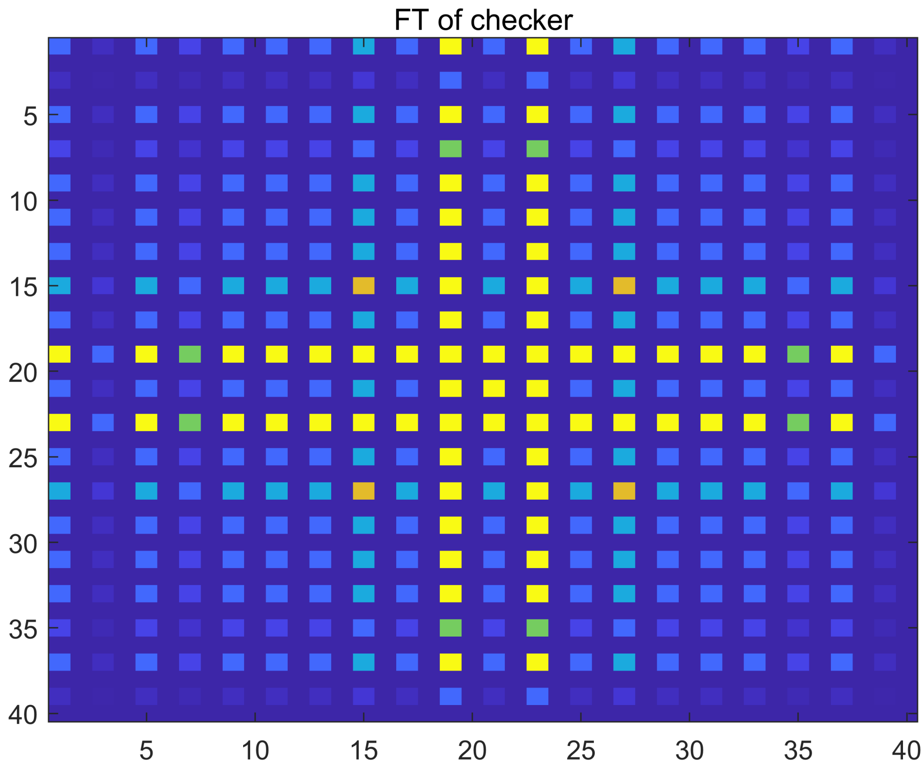
## load image

```
checker = imread('checkerboard.png');
checker_blur = imread('checkerboard_blurred.png');

colormap(gray);
subplot(1, 2, 1), imshow(checker), title('checker');
subplot(1, 2, 2), imshow(checker_blur), title('checker blurred');
```

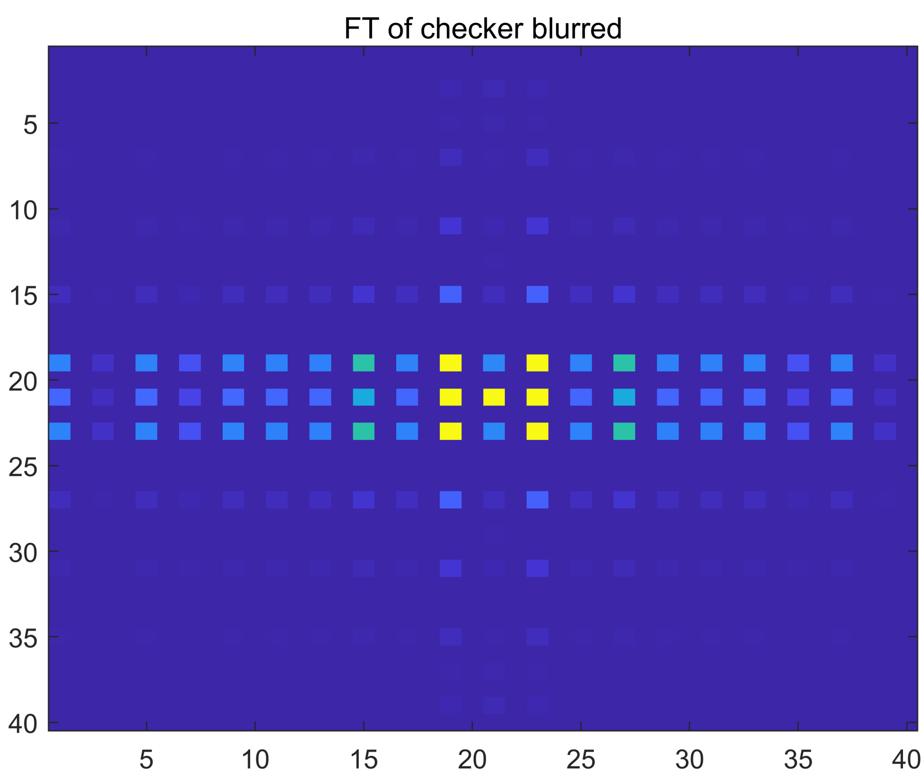



## check FT of checker, checker_blur

```
C = fft2(double(checker));
C_blur = fft2(double(checker_blur));

figure, imagesc(abs(fftshift(C)), [1, 10000]), title('FT of checker');
```




```
figure, imagesc(abs(fftshift(C_blur)), [1, 10000]), title('FT of checker blurred');
```

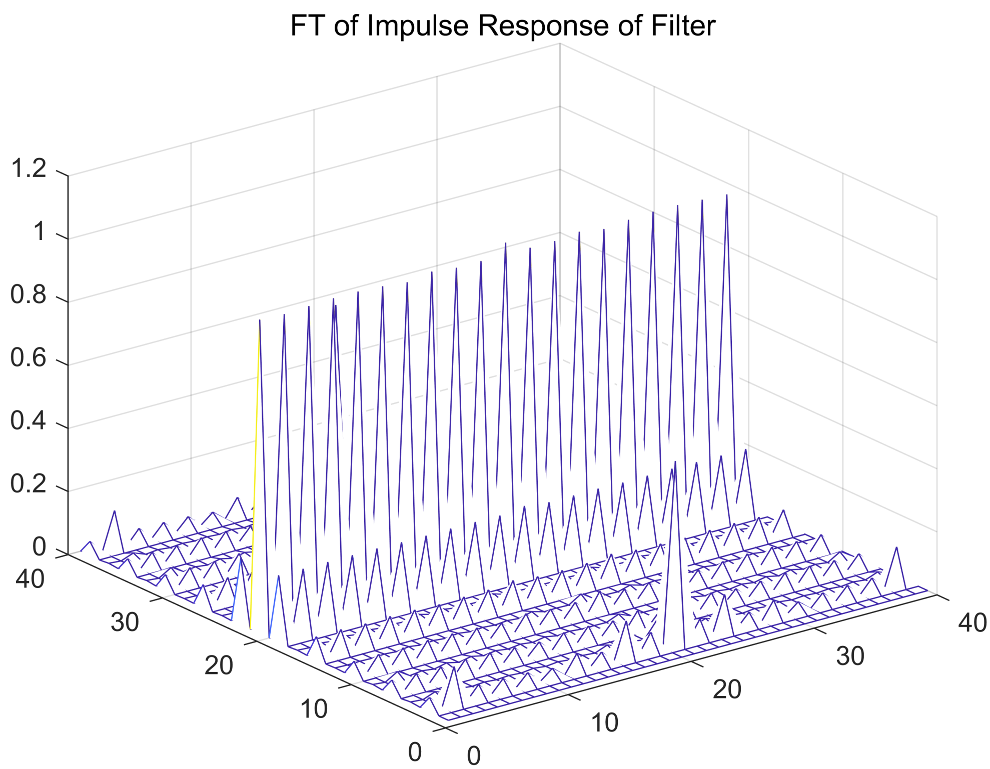


```
changer_FT = C_blur./(C+0.0001);
```

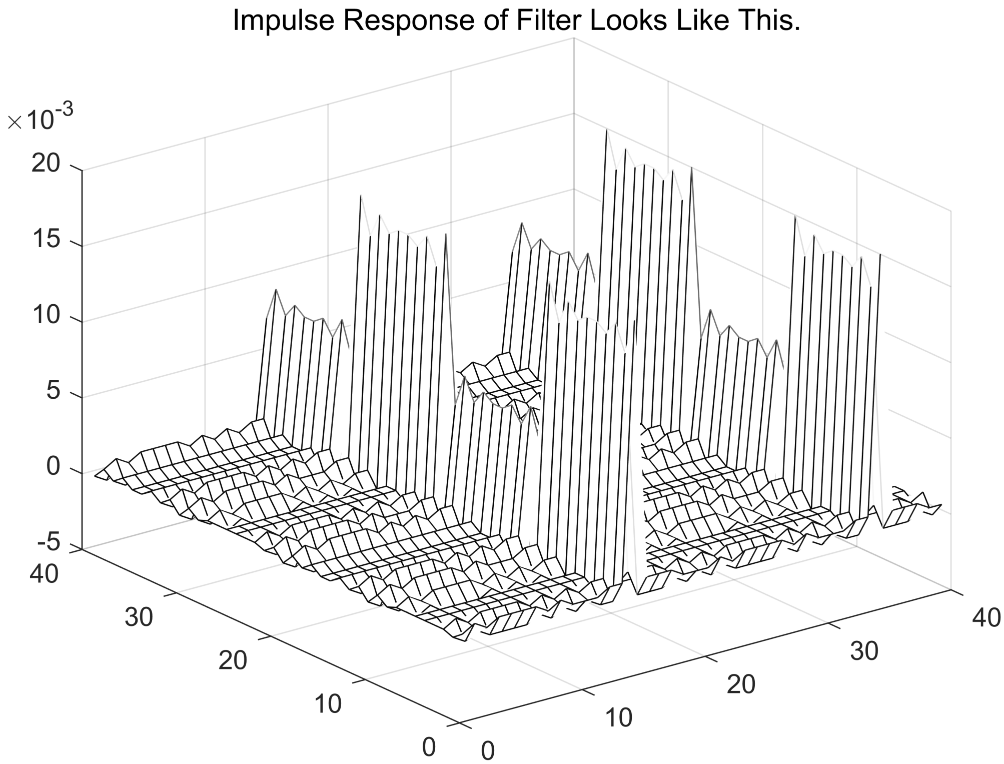
Y축에서 f=0 근방의 값을 주로 유지하므로, y축 방향으로 움직였음을 알 수 있다.

## (find filter by dividing C with C_blur)

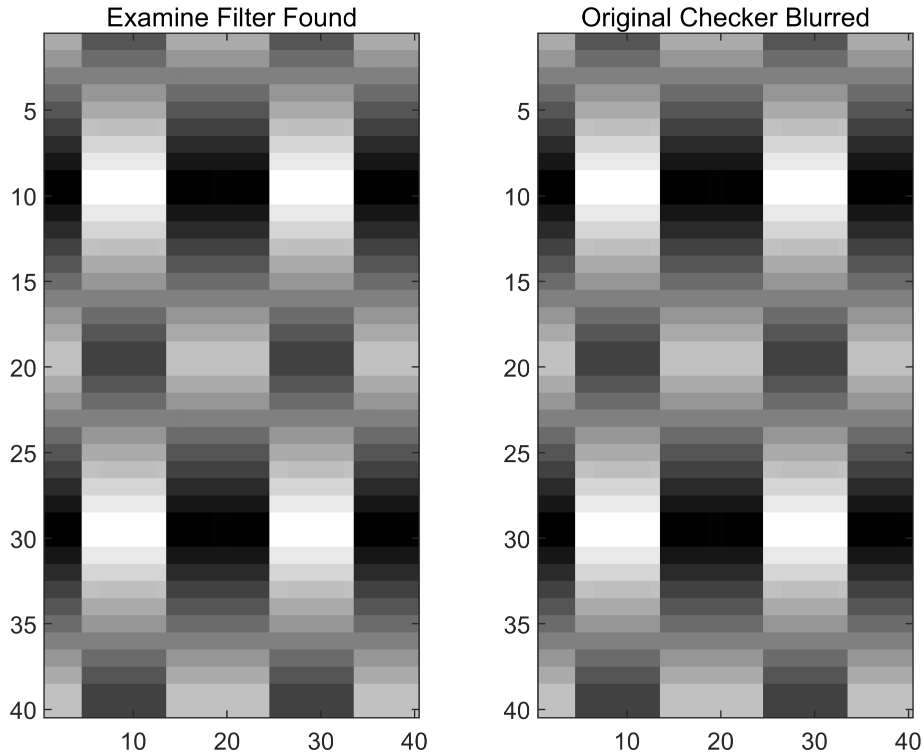
```
figure, mesh(abs(fftshift(changer_FT)));
title('FT of Impulse Response of Filter');
```




```
changer = ifft2(changer_FT);
figure, mesh(changer), colormap(gray);
title('Impulse Response of Filter Looks Like This.');
```

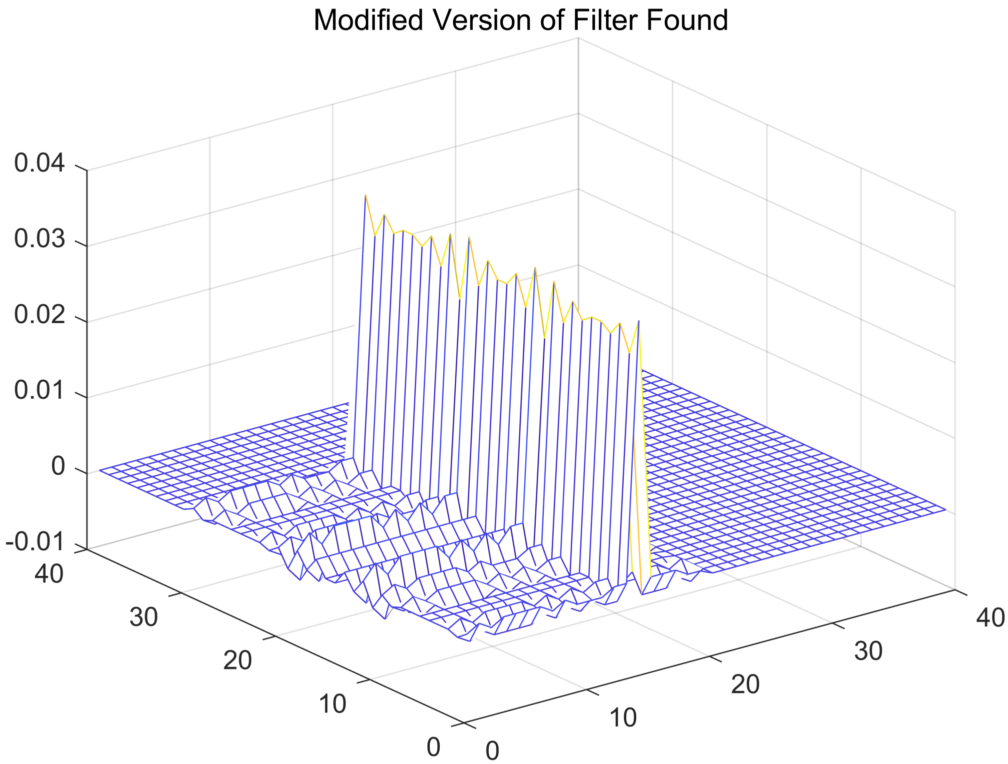



```
figure;
reblurred = ifft2(C.*changer_FT);
subplot(121), imagesc(abs(reblurred)), colormap(gray);
title('Examine Filter Found')
subplot(122), imagesc(checker_blur), colormap(gray);
title('Original Checker Blurred')
```




## (another version of changer -> works!)

```
figure;
changer(1:20, 1:20) = 4*changer(1:20, 1:20);
changer(1:40, 21:40) = 0;
changer(21:40, 1:40) = 0;
changer(21:30, 1:40) = changer(21:30, 1:40) + changer(1:10, 1:40)/2;
changer(1:10, 1:40) = changer(1:10, 1:40)/2;
mesh(changer);
title('Modified Version of Filter Found')
```

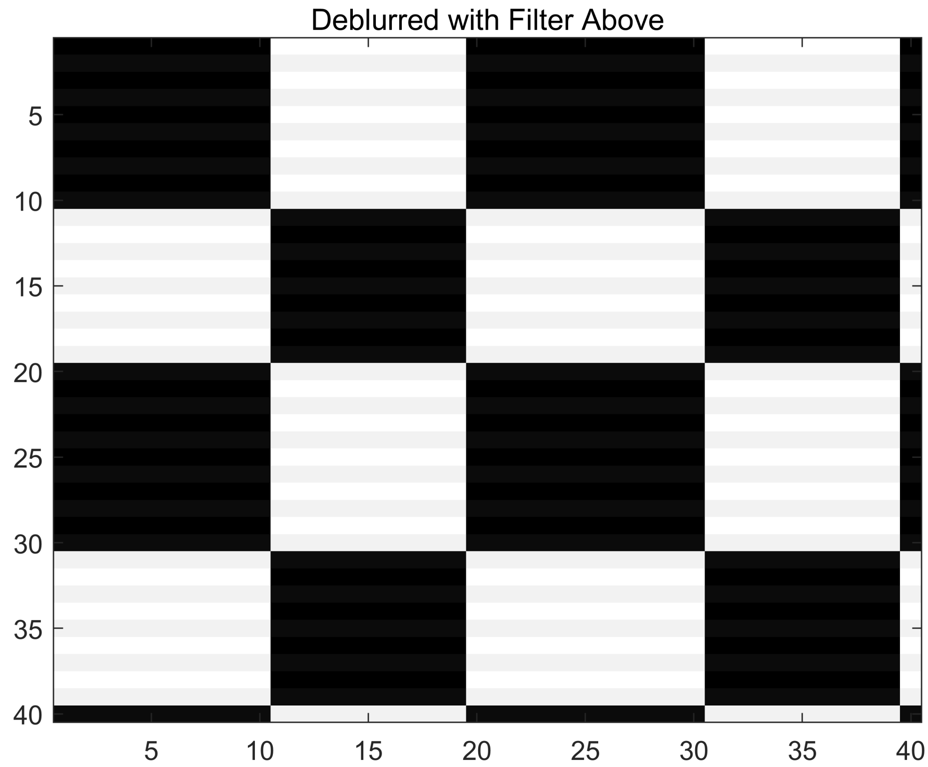



```
changer_FT = fft2(changer);
```

checker 이미지가 20, 20 주기이기도 하다는 점을 참고하면 위와 같이 단순화시킬 수 있다.

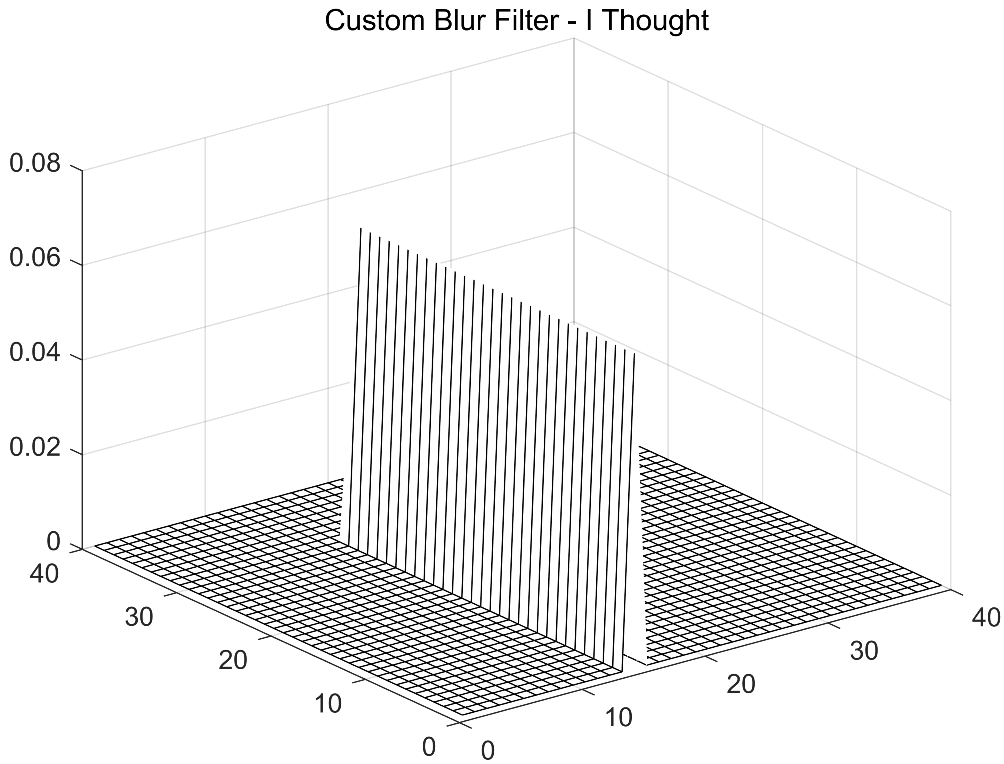
## (restore blurred img using filter above)

```
C_rest = C_blur./(changer_FT+0.000001);
imagesc(ifft2(C_rest), [1, 256]), colormap(gray);
title('Deblurred with Filter Above')
```



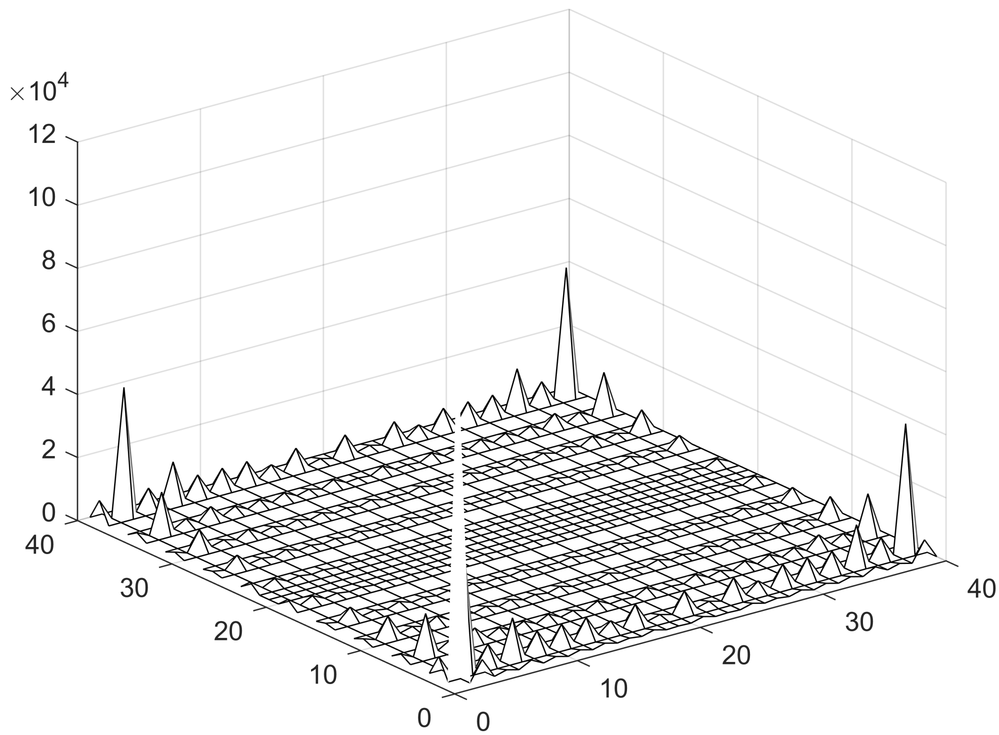

## (custom blur filter - ideal motion blur)

```
c_custom = zeros(40, 40);
c_custom(1:30, 15) = 1;
c_custom = c_custom/15;

mesh(c_custom);
title('Custom Blur Filter - I Thought')
```

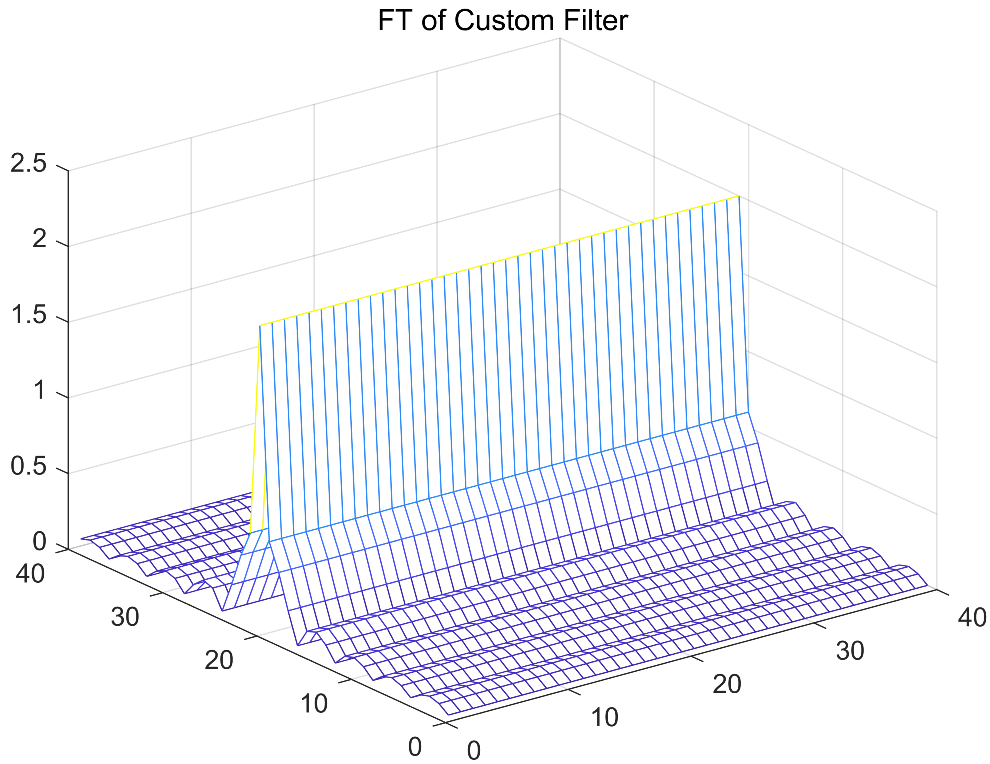



## (custom blur filter - tried to deblur checker blurred)

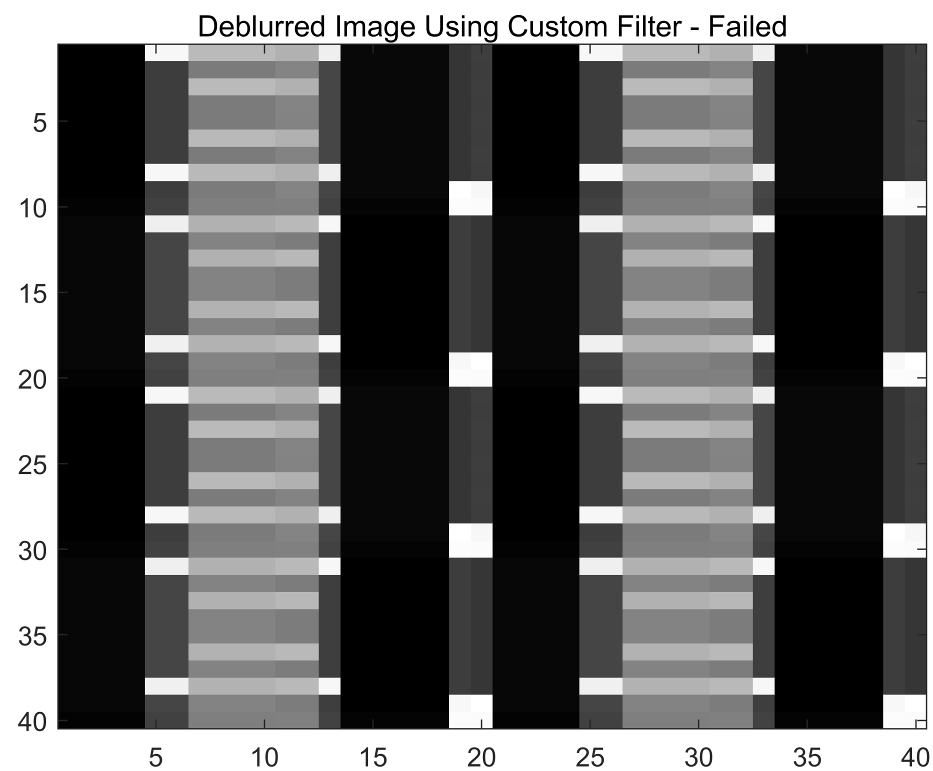
```
dblr_custom = fft2(double(checker_blur))./fft2(c_custom);
dblr_custom(isnan(dblr_custom)) = 0;
mesh(abs(dblr_custom));
```




```
ifft2(dblr_custom);

figure, mesh(abs(fftshift(fft2(c_custom))));
title('FT of Custom Filter')
```



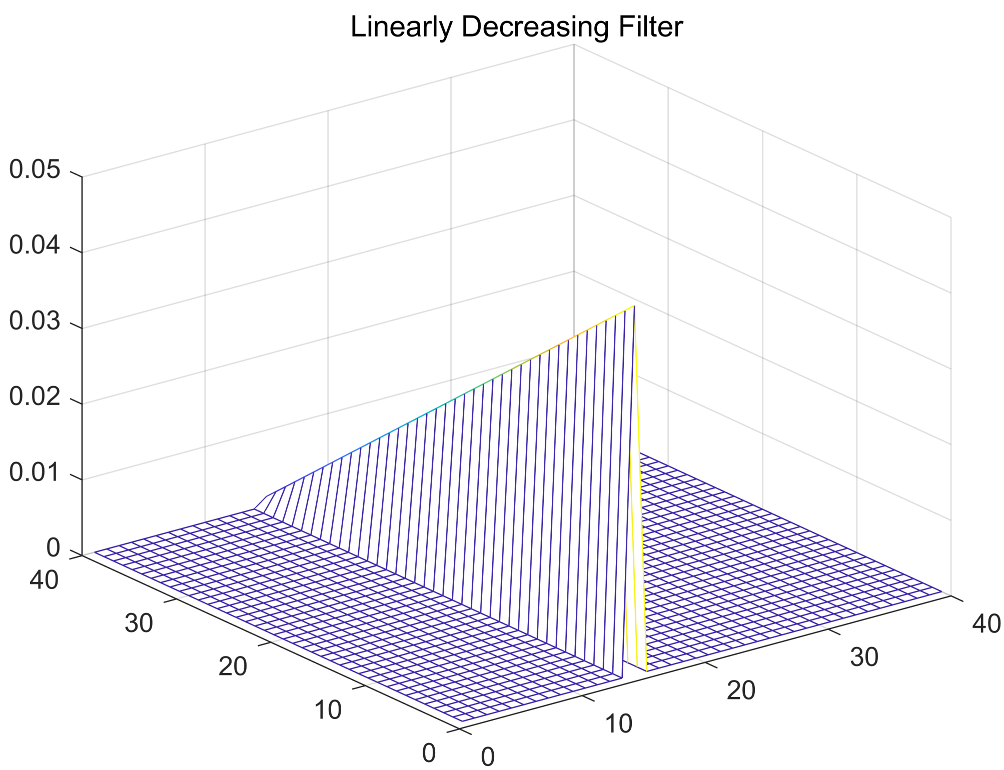

```
figure, imagesc(abs(ifft2(fft2(double(checker_blur))./(fft2(c_custom)+0.0001)))), colormap(gray
title('Deblurred Image Using Custom Filter - Failed')
```



이상적인 필터라고 생각했지만 오히려 실패했다.

## (custom filter - linear decrease)

```
c_custom = zeros(40, 40);
c_custom(:, 15) = 40:-1:1;
c_custom = c_custom / sum(c_custom, 'all');
figure, mesh(c_custom);
title('Linearly Decreasing Filter')
```



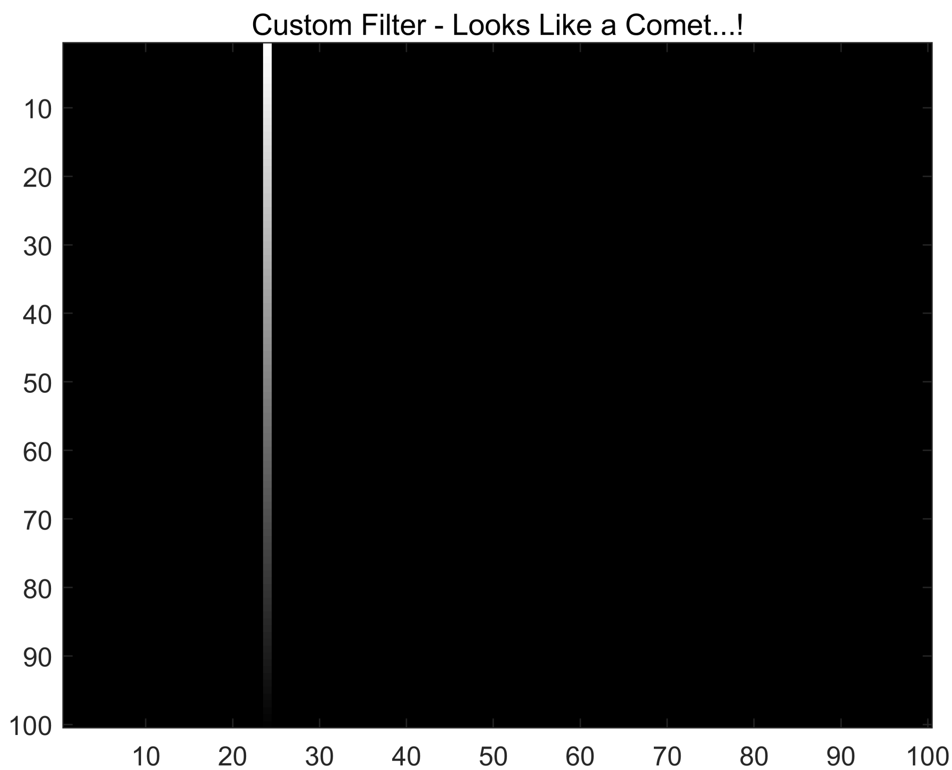

## (deblur with linear drop filter)

```
C = fft2(c_blur, c_size, c_size);
c_deblur = abs(ifft2(C ./ C_custom));
figure, imshow(c_deblur);
title('Deblur Success.')
```

## 3.(c)

### Beautiful Custom Filter - Linear Decrease

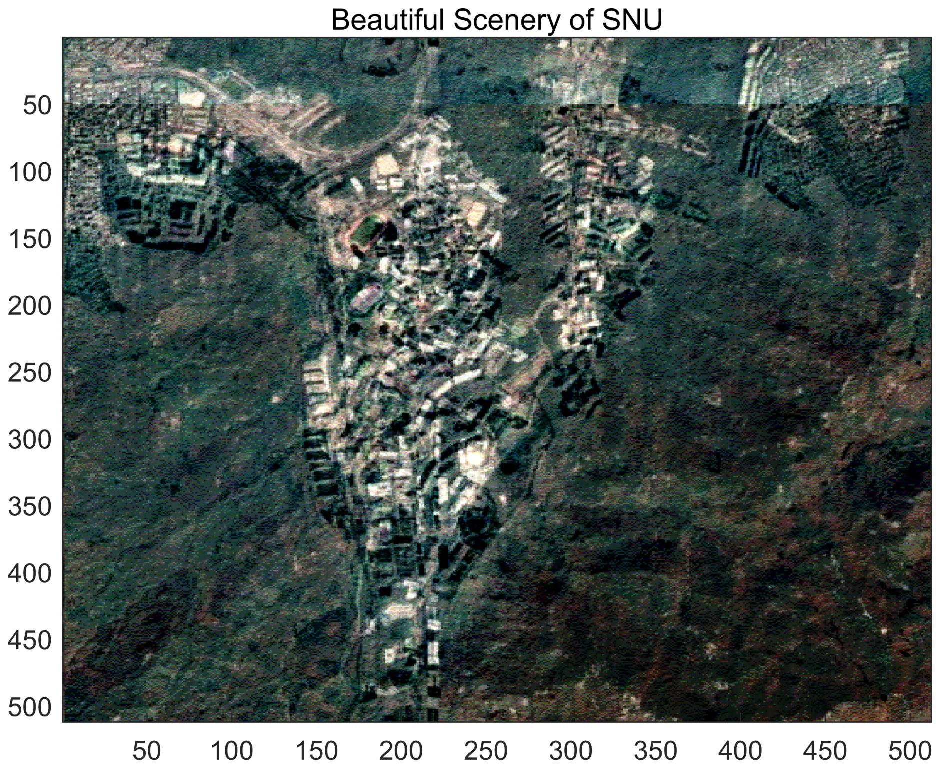
```
cldf_len = 100;
cldf = zeros(100, 100);
cldf(:, 24) = 100:-1:1;
cldf = cldf./sum(cldf, 'all');
figure, imagesc(cldf), colormap(gray);
title('Custom Filter - Looks Like a Comet...!')
```




## 3.(d)

### Deblur with Custom Filter - Success.

```
cldf_FT = fft2(cldf, 512, 512);
mimg_FT = fft2(double(img));
mimg_deblurred_FT = zeros(size(mimg_FT));
mimg_deblurred_FT(:,:,1) = mimg_FT(:,:,1)./cldf_FT;
mimg_deblurred_FT(:,:,2) = mimg_FT(:,:,2)./cldf_FT;
mimg_deblurred_FT(:,:,3) = mimg_FT(:,:,3)./cldf_FT;

mimg_deblurred = ifft2(mimg_deblurred_FT);
imagesc(uint8(mimg_deblurred));
title('Beautiful Scenery of SNU')
```


Beautiful Scenery of SNU
50
100
150
200
250
300
350
400
450
500
50
100
150
200
250
300
350
400
450
500

2.

(e)

$$m(t)\cos(2\pi f_1 t)\cos(2\pi f_1 t)$$

$$= m(t)\left\{\frac{1}{2}e^{j2\pi f_1 t} + \frac{1}{2}e^{-j2\pi f_1 t}\right\}^2$$

$$= m(t)\left(\frac{1}{4}e^{j2\pi\cdot 2f_1 t} + \frac{1}{2} + \frac{1}{4}e^{-j2\pi\cdot 2f_1 t}\right)$$

$$\Big\downarrow \text{FT}$$

$$M(f) * \left(\frac{1}{4}\delta(f-2f_1) + \frac{1}{2}\delta(f) + \frac{1}{4}\delta(f+2f_1)\right)$$

$$= \frac{1}{4}M(f-2f_1) + \frac{1}{2}M(f) + \frac{1}{4}M(f+2f_1)$$

$$= \frac{1}{4}M(f-400) + \underbrace{\frac{1}{2}M(f)}_{\text{inside passband!}} + \frac{1}{4}M(f+400)$$

$$m(t)\cos(2\pi f_1 t)\sin(2\pi f_1 t)$$

$$= m(t)\left(\frac{1}{2}e^{j2\pi f_1 t} + \frac{1}{2}e^{-j2\pi f_1 t}\right)\left(\frac{1}{2j}e^{j2\pi f_1 t} - \frac{1}{2j}e^{-j2\pi f_1 t}\right)$$

$$= m(t)\left(\frac{1}{4j}e^{j2\pi\cdot 2f_1 t} - \frac{1}{4j}e^{-j2\pi\cdot 2f_1 t}\right)$$

$$\Big\downarrow \text{FT}$$

$$M(f) * \left(\frac{1}{4j}\delta(f-2f_1) - \frac{1}{4j}\delta(f+2f_1)\right)$$

$$= \frac{1}{4j}M(f-2f_1) - \frac{1}{4j}M(f+2f_1)$$

$$= \frac{1}{4j}M(f-400) - \frac{1}{4j}M(f+400)$$ → All outside of passband

$m(t)\cos(2\pi f_1 t)\cos(2\pi f_2 t)$

$= m(t)\left(\frac{1}{2}e^{j2\pi f_1 t}+\frac{1}{2}e^{-j2\pi f_1 t}\right)\left(\frac{1}{2}e^{j2\pi f_2 t}+\frac{1}{2}e^{-j2\pi f_2 t}\right)$

$= m(t)\left(\frac{1}{4}e^{j2\pi(f_1+f_2)t}+\frac{1}{4}e^{j2\pi(f_1-f_2)t}+\frac{1}{4}e^{j2\pi(-f_1+f_2)t}+\frac{1}{4}e^{j2\pi(-f_1-f_2)t}\right)$

$\Big\downarrow$ FT

$M(f) * \left(\frac{1}{4}\delta(f-(f_1+f_2))+\frac{1}{4}\delta(f-(f_1-f_2))+\frac{1}{4}\delta(f-(-f_1+f_2))+\frac{1}{4}\delta(f-(-f_1-f_2))\right)$

$= \frac{1}{4}M(f-(f_1+f_2))+\frac{1}{4}M(f-(f_1-f_2))+\frac{1}{4}M(f-(-f_1+f_2))+\frac{1}{4}M(f-(-f_1-f_2))$

$= \frac{1}{4}M(f-600)+\frac{1}{4}M(f+200)+\frac{1}{4}M(f-200)+\frac{1}{4}M(f+600)$

$\rightsquigarrow$ All outside of passband!