HW4 Program Assignment

By: 105060012 張育菘

P1

Let $x[n] = n(0.9)^n u[n]$,

(a) Determine the DTFT $\widetilde{X}(e^{j\omega})$ of $x[n]$. Please write your calculations and answer on your .mlx file.

Ans.

$$a^n u[n] \leftrightarrow \frac{1}{1-\mathrm{az}^{-1}}, \mathrm{nx}[n] \leftrightarrow -z\frac{\mathrm{dX}(z)}{\mathrm{dz}}, z = e^{j\omega} \Longrightarrow \mathrm{na}^n u[n] \leftrightarrow \frac{\mathrm{az}^{-1}}{(1-\mathrm{az}^{-1})^2} = \frac{ae^{-j\omega}}{(1-ae^{-j\omega})^2}$$

$$\Longrightarrow n(0.9)^n u[n] \leftrightarrow \frac{0.9e^{-j\omega}}{(1-0.9e^{-j\omega})^2}$$

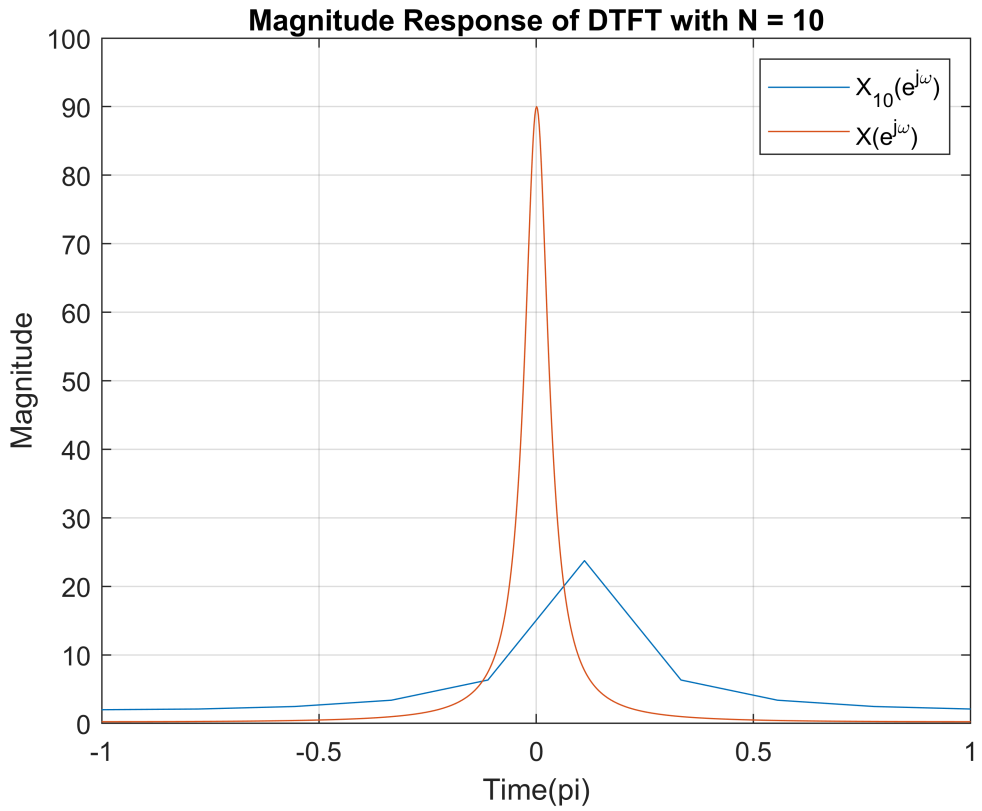
(b) Choose first N = 20 samples of x[n] and compute the approximate DTFT $\widetilde{X}_N(e^{j\omega})$ using the fft function.

Plot magnitudes of $\widetilde{X}(e^{j\omega})$ and $\widetilde{X}_N(e^{j\omega})$ in one plot and compare your results.

```
close all; clear;
fprintf('1(b)\n');
```

```
1(b)
```

```
n = linspace(0, 1023, 1024);

x = n.*((0.9).^n);
X_ft = fft(x); X_ft = fftshift(X_ft);
om = linspace(-pi, pi, 1024);
X10_ft = fft(x, 10); X10_ft = fftshift(X10_ft);
om_b = linspace(-pi, pi, 10);

figure;
plot(om_b/pi,abs(X10_ft)); hold on;
plot(om/pi, abs(X_ft)); hold off; grid on;

title('Magnitude Response of DTFT with N = 10');
legend('X_1_0(e^j^\omega)','X(e^j^\omega)')
ylabel('Magnitude');
xlabel('Time(pi)');
```

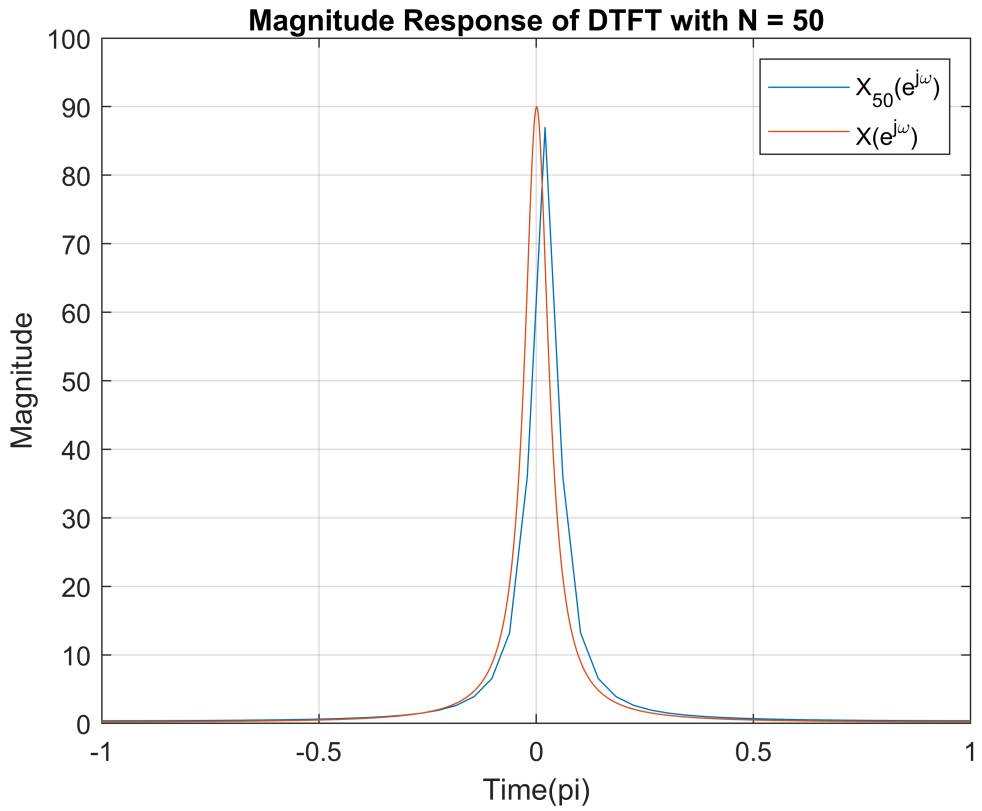


(c) Repeat part (b) using N = 50.

```
fprintf('1(c)\n');
```

```
1(c)
```

```
X50_ft = fft(x, 50); X50_ft = fftshift(X50_ft);
om_c = linspace(-pi, pi, 50);

figure;
plot(om_c/pi,abs(X50_ft)); hold on;
plot(om/pi, abs(X_ft)); hold off; grid on;

title('Magnitude Response of DTFT with N = 50');
legend('X_5_0(e^j^\omega)','X(e^j^\omega)')
ylabel('Magnitude');
xlabel('Time(pi)');
```

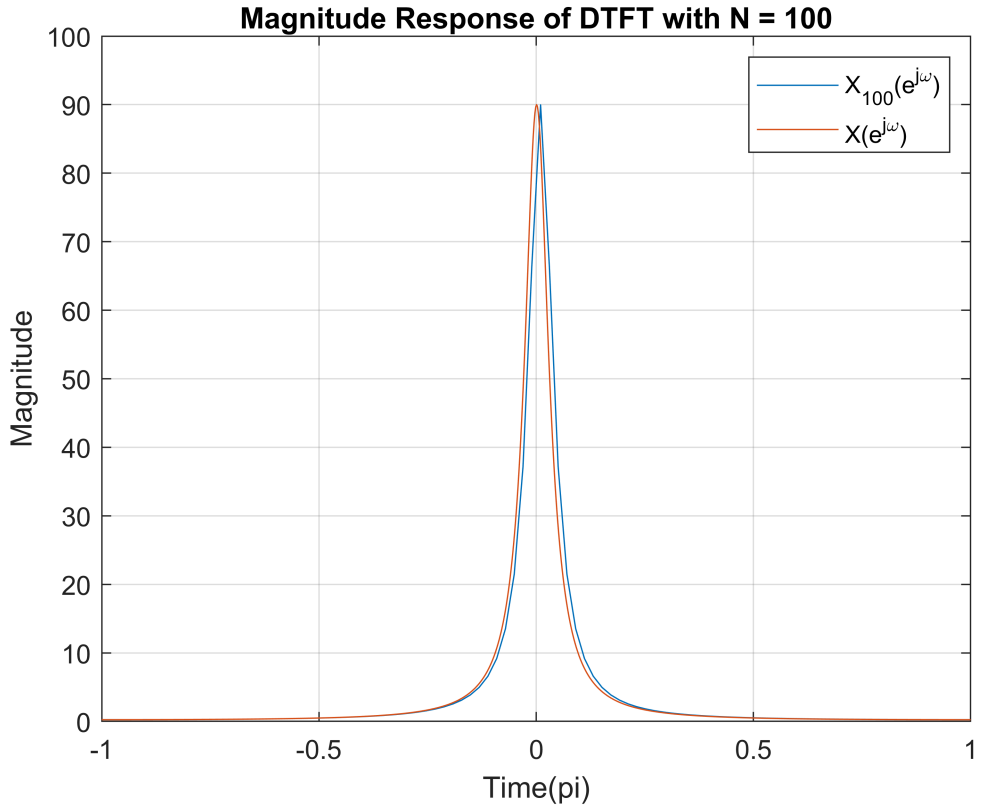


(d) Repeat part (b) using N = 100.

```
fprintf('1(d)\n');
```

```
1(d)
```

```
X100_ft = fft(x, 100); X100_ft = fftshift(X100_ft);
om_d = linspace(-pi, pi, 100);

figure;
plot(om_d/pi,abs(X100_ft)); hold on;
plot(om/pi, abs(X_ft)); hold off; grid on;

title('Magnitude Response of DTFT with N = 100');
legend('X_1_0_0(e^j^\omega)','X(e^j^\omega)')
ylabel('Magnitude');
xlabel('Time(pi)');
```



P2

Let $x[n] = x_1[n] + \mathrm{j}x_2[n]$ where sequences $x_1[n]$ and $x_2[n]$ are real-valued.

(a)Show that $X_1[k] = X^{\mathrm{cce}}[k]$ and $\mathrm{j}X_2[k] = X^{\mathrm{cco}}[k]$. Please write your calculations and answer on your .mlx file.

Ans.

$$X^{\mathrm{cce}}[k] = \frac{1}{2}\left(X[k] + X^*[\langle -k\rangle_N]\right), X^{\mathrm{cco}}[k] = \frac{1}{2}\left(X[k] - X^*[\langle -k\rangle_N]\right),\ \ x^*[n] \leftrightarrow X^*[\langle -k\rangle_N]$$

$$x[n] = x_1^{\mathrm{ce}}[n] + x_1^{\mathrm{co}}[n] + \mathrm{j}x_2^{\mathrm{ce}}[n] + \mathrm{j}x_2^{\mathrm{co}}[n] \leftrightarrow X[k] = X_1^{\mathrm{ce}}[k] + \mathrm{j}X_1^{\mathrm{co}}[k] + \mathrm{j}X_2^{\mathrm{ce}}[k] + X_2^{\mathrm{co}}[k]$$

$$x^*[n] = x_1^{\mathrm{ce}}[n] + x_1^{\mathrm{co}}[n] - \mathrm{j}x_2^{\mathrm{ce}}[n] - \mathrm{j}x_2^{\mathrm{co}}[n] \leftrightarrow X^*[\langle -k\rangle_N] = X_1^{\mathrm{ce}}[k] + \mathrm{j}X_1^{\mathrm{co}}[k] - \mathrm{j}X_2^{\mathrm{ce}}[k] - X_2^{\mathrm{co}}[k]$$

$$\Longrightarrow X_1[k] = X_1^{\mathrm{ce}}[k] + \mathrm{j}X_1^{\mathrm{co}}[k] = \frac{1}{2}\left(X[k] + X^*[\langle -k\rangle_N]\right) = X^{\mathrm{cce}}[k]$$

$$\Longrightarrow \mathrm{j}X_2[k] = \mathrm{j}X_2^{\mathrm{ce}}[k] + X_2^{\mathrm{co}}[k] = \frac{1}{2}\left(X[k] - X^*[\langle -k\rangle_N]\right) = X^{\mathrm{cco}}[k]$$

(b) Write a MATLAB function [X1,X2] = tworealDFTs(x1,x2) that implements the results in part (a).

```
close all; clear;
fprintf('2(b)\n');
```

```
2(b)
```

```
open tworealDFTs.m;
```

(c) Verify your function on the following two sequences: $x_1[n] = 0.9^n, x_2[n] = (1 - 0.8^n); 0 \le n \le 49$

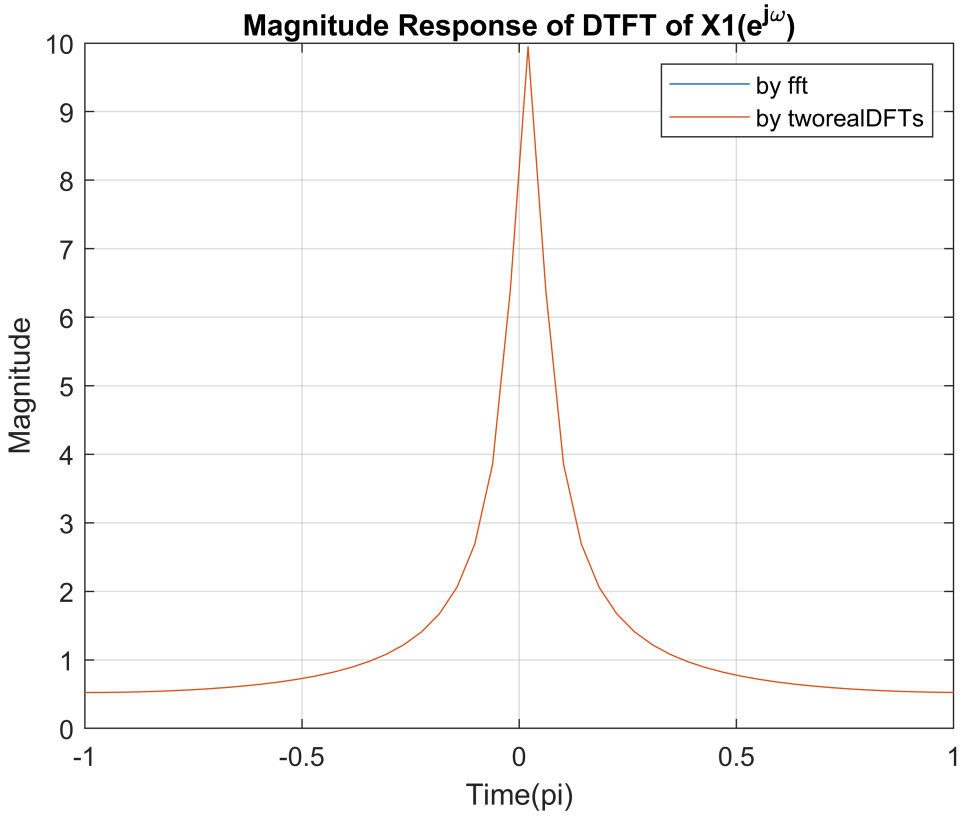
```
fprintf('2(c)\n');
```

```
2(c)
```

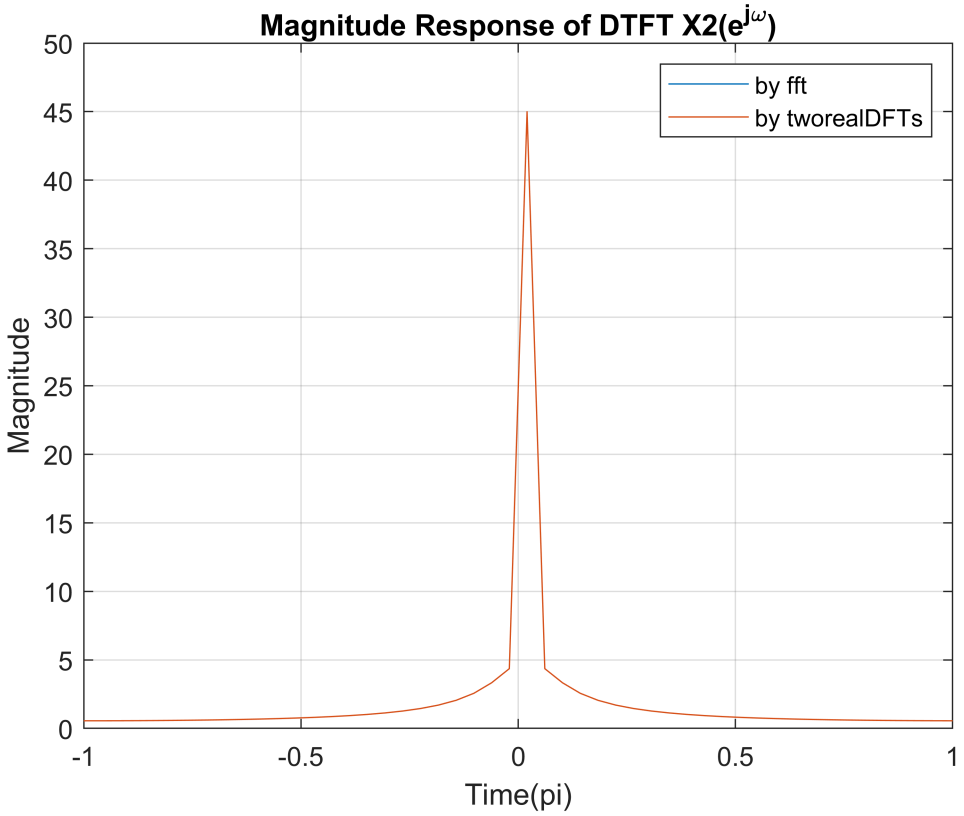
```
N = 50;
n = linspace(0, N-1, N);
x1 = ((0.9).^n);
x2 = (1-((0.8).^n));

[X1_ft,X2_ft] = tworealDFTs(x1,x2)
```

```
X1_ft = 1×50 complex
   9.9485 + 0.0000i   4.4039 - 4.6384i   1.9176 - 3.3459i   1.1903 - 2.4164i ···
X2_ft = 1×50 complex
  45.0001 + 0.0000i  -3.9209 + 1.9056i  -2.4940 + 2.2040i  -1.6814 + 1.9329i ···
```

```
X1_ft = fftshift(X1_ft); X2_ft = fftshift(X2_ft);
om = linspace(-pi, pi, N);

X1_fft = fft(x1); X1_fft = fftshift(X1_fft);
X2_fft = fft(x2); X2_fft = fftshift(X2_fft);

figure;
plot(om/pi,abs(X1_fft)); hold on;
plot(om/pi,abs(X1_ft)); hold off; grid on;
title('Magnitude Response of DTFT of X1(e^j^\omega)');
legend('by fft','by tworealDFTs');
ylabel('Magnitude');
xlabel('Time(pi)');
```




```
figure;
plot(om/pi,abs(X2_fft)); hold on;
plot(om/pi,abs(X2_ft)); hold off; grid on;
title('Magnitude Response of DTFT X2(e^j^\omega)');
legend('by fft','by tworealDFTs');
ylabel('Magnitude');
xlabel('Time(pi)');
```



P3

Let $x_1[n] = \{1_{n=0}, 2, 3, 4, 5\}$ be a 5-point sequence and let $x_2[n] = \{2_{n=0}, -1, 1, -1\}$ be a 4-point sequence.

(a) Determine $x_1[n] \odot x_2[n]$ using hand calculations. Please write your calculations and answer on your .mlx file.

Ans.

If length($x_1[n]$) ≠ length($x_2[n]$) #zero padding

$x_1[n] = \{1_{n=0}, 2, 3, 4, 5\}, x_2[n] = \{2_{n=0}, -1, 1, -1, 0\}$

$$y[n] = x_1[n] \odot x_2[n] = \sum_{m=0}^{4} x_1[m] x_2[\langle n - m \rangle_5]$$

$$\Longrightarrow y[0] = x_1[0]x_2[0] + x_1[1]x_2[-1] + x_1[2]x_2[-2] + x_1[3]x_2[-3] + x_1[4]x_2[-4] = 2 + 0 - 3 + 4 - 5 = -2$$
$$y[1] = x_1[0]x_2[1] + x_1[1]x_2[0] + x_1[2]x_2[-1] + x_1[3]x_2[-2] + x_1[4]x_2[-3] = -1 + 4 + 0 - 4 + 5 = 4$$
$$y[2] = x_1[0]x_2[2] + x_1[1]x_2[1] + x_1[2]x_2[0] + x_1[3]x_2[-1] + x_1[4]x_2[-2] = 1 - 2 + 6 + 0 - 5 = 0$$
$$y[3] = x_1[0]x_2[3] + x_1[1]x_2[2] + x_1[2]x_2[1] + x_1[3]x_2[0] + x_1[4]x_2[-1] = -1 + 2 - 3 + 8 + 0 = 6$$
$$y[4] = x_1[0]x_2[4] + x_1[1]x_2[3] + x_1[2]x_2[2] + x_1[3]x_2[1] + x_1[4]x_2[0] = 0 - 2 + 3 - 4 + 10 = 7$$
$$\Longrightarrow y[n] = \{-2_{n=0}, 4, 0, 6, 7\}$$

(b) Verify your calculations in (a) using the circonv function.

```
close all; clear;
fprintf('3(b)\n');
```

```
3(b)
```

```
x1 = [1 2 3 4 5]';
x2 = [2 -1 1 -1]';
x2n = [2 -1 1 -1 0]'; % zero padding
y = circonv(x1, x2n);
y = y'
```

```
y = 1×5
    -2     4     0     6     7
```

We can find that the results of 3(a) and 3(b) are the same.

(c) Verify your calculations in (a) by computing the DFTs and IDFT.

```
fprintf('3(c)\n');
```

```
3(c)
```

```
ydft = ifft(fft(x1).*fft(x2n));
ydft = ydft'
```

```
ydft = 1×5
   -2.0000    4.0000    0.0000    6.0000    7.0000
```

We can find that the results of 3(a) and 3(c) are the same.

P4

Let $x_1[n]$ be an N_1-point and $x_2[n]$ be an N_2-point sequence. Let N ≥ max(N1,N2). Their N-point circular convolution is shown to be equal to the aliased version of their linear convolution in (7.209) in Program Assignment 3. This result can be used to compute the circular convolution via the linear convolution.

$$x_3[n] = x_1[n] * x_2[n], x_4[n] = x_1[n] \odot x_2[n]$$

$$x_4[n] = \sum_{l=-\infty}^{\infty} x_3[n + \mathrm{lN}] \qquad (7.209)$$

(a) Develop a MATLAB function y = lin2circonv(x,h) that implements this approach.

```
close all; clear;
fprintf('4(a)\n');
```

```
4(a)
```

```
open lin2circonv.m;
```

(b) For $x[n] = \{1_{n=0}, 2, 3, 4\}$ and $h[n] = \{1_{n=0}, -1, 1, -1\}$ determine their 4-point circular convolution using the lin2circonv function and verify using the circonv function.

```
fprintf('4(b)\n');
```

```
4(b)
```

```
x = [1 2 3 4]'; h = [1 -1 1 -1]';
hx_circonv = circonv(h, x);
hx_circonv = hx_circonv'
```

```
hx_circonv = 1×4
    -2     2    -2     2
```

```
hx_lin2circonv = lin2circonv(h', x')
```

```
hx_lin2circonv = 1×4
    -2     2    -2     2
```

We can find that the results by the function circonv and lin2circonv are the same.

P5

Let a 2D filter impulse response h[m, n] be given by

$$h[m,n] = \begin{cases} \frac{1}{2\pi\sigma^2} e^{\frac{-m^2+n^2}{2\sigma^2}} & , -128 \le m, n \le 127 \\ 0 & , \text{otherwise} \end{cases}$$

where σ is a parameter. For this problem use the “Lena” image.

(a) For σ = 4, determine h[m, n] and compute its 2D-DFT H[k, l] via the fft2 function taking care of shifting the origin of the array from the middle to the beginning (using the ifftshift function). Show the log-magnitude of H[k, l] as an image.

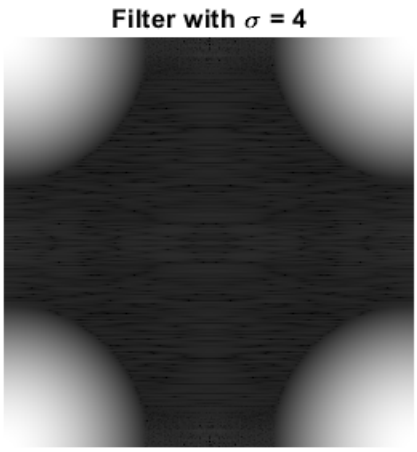
```
close all; clear;
fprintf('5(a)\n');
```

```
5(a)
```

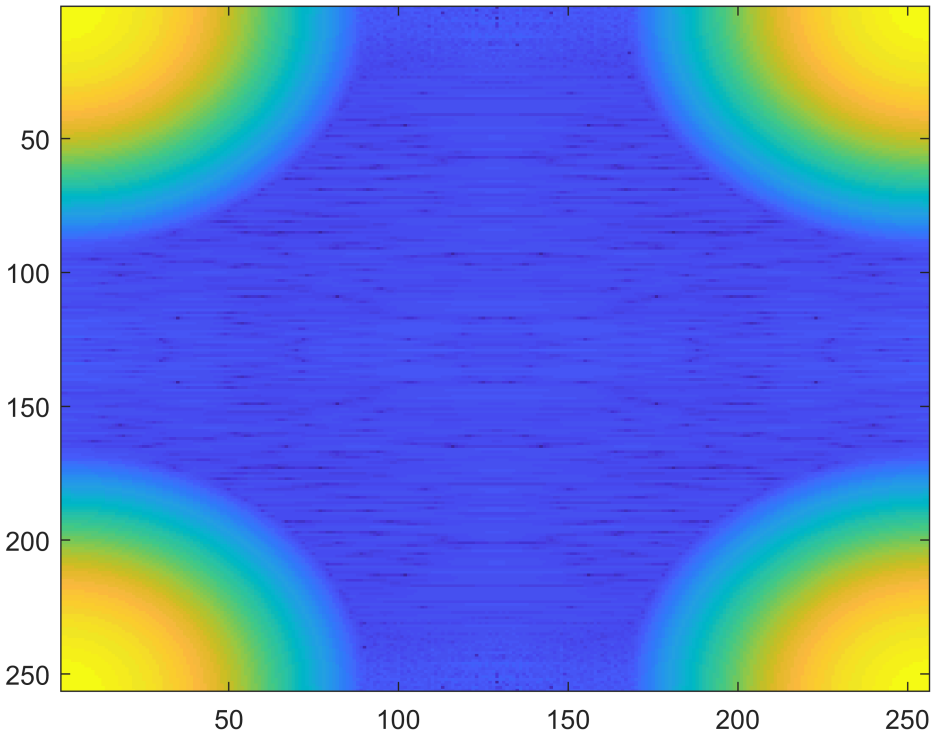
```


sigma = 4;
h = zeros(256, 256);

for m = -128:1:127
    for n = -128:1:127
        h(m+129, n+129) = ((2*pi*(sigma)^2)^-1).*exp(-1*((m^2)+(n^2))./(2*(sigma)^2));
    end
end

H = ifftshift(h);
H = fft2(H)
```

```
H = 256×256 complex
   1.0000 + 0.0000i   0.9952 + 0.0000i   0.9809 + 0.0000i   0.9576 - 0.0000i ···
   0.9952 + 0.0000i   0.9904 + 0.0000i   0.9762 + 0.0000i   0.9530 - 0.0000i
   0.9809 + 0.0000i   0.9762 + 0.0000i   0.9622 + 0.0000i   0.9393 - 0.0000i
   0.9576 + 0.0000i   0.9530 + 0.0000i   0.9393 + 0.0000i   0.9169 - 0.0000i
   0.9258 + 0.0000i   0.9213 + 0.0000i   0.9081 + 0.0000i   0.8865 - 0.0000i
   0.8865 + 0.0000i   0.8822 + 0.0000i   0.8696 + 0.0000i   0.8489 - 0.0000i
   0.8407 - 0.0000i   0.8367 + 0.0000i   0.8247 + 0.0000i   0.8050 - 0.0000i
   0.7897 - 0.0000i   0.7859 + 0.0000i   0.7746 + 0.0000i   0.7562 - 0.0000i
   0.7346 + 0.0000i   0.7311 + 0.0000i   0.7206 + 0.0000i   0.7034 - 0.0000i
   0.6768 + 0.0000i   0.6736 + 0.0000i   0.6639 + 0.0000i   0.6481 + 0.0000i
     ⋮
```

```
H_abs = abs(H);
H_log = log(H_abs);

figure; imshow(H_log,[]); title('Filter with \sigma = 4');
```




```
figure; imagesc(H_log);
```



(b) Process the “Lena” image in the frequency domain using the above H[k, l]. This will involve taking 2D-DFT of the image, multiplying the two DFTs and then taking the inverse of the product. Comment on the visual quality of the resulting filtered image.

```
fprintf('5(b)\n');
```

5(b)

```
sigma = 4;
h = zeros(256, 256);

for m = -128:1:127
    for n = -128:1:127
        h(m+129, n+129) = ((2*pi*(sigma)^2)^-1).*exp(-1*((m^2)+(n^2))./(2*(sigma)^2));
    end
end

H = ifftshift(h); H = fft2(H);

lena = imread('lena.jpg');
lena_fft2 = fft2(lena);
lena_H4 = lena_fft2.*H;
lena_h4_ifft = ifft2(lena_H4);
figure; imshow(abs(lena)); title('Original version');
```


```
figure; imshow(abs(lena_h4_ifft),[]); title('Afer filtering (\sigma = 4)');
```

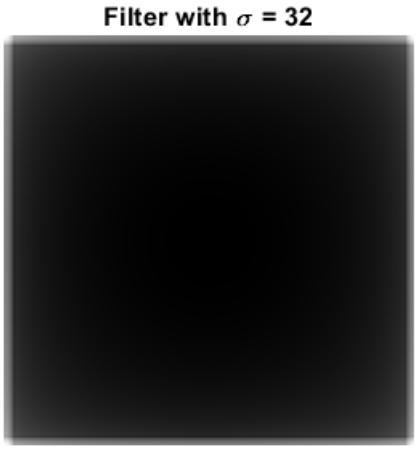
We find that the image after filtering is more blurred than the original image.

(c) Repeat (a) and (b) for σ = 32 and comment on the resulting filtered image as well as the difference between the two filtered images.

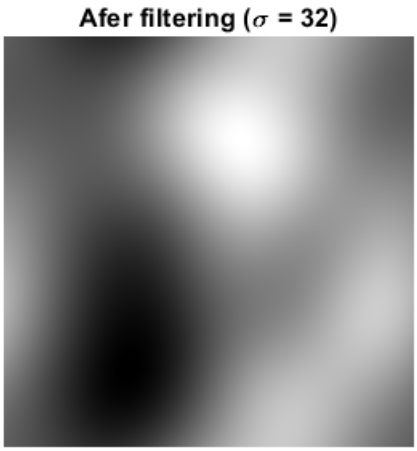
```
fprintf('5(c)\n');
```

```
5(c)
```

```
sigma = 32;
h = zeros(256, 256);

for m = -128:1:127
    for n = -128:1:127
        h(m+129, n+129) = ((2*pi*(sigma)^2)^-1).*exp(-1*((m^2)+(n^2))./(2*(sigma)^2));
    end
end

H = ifftshift(h); H = fft2(H);
H_abs = abs(H); H_log = log(H_abs);
figure; imshow(H_log,[]); title('Filter with \sigma = 32');
```




```
lena = imread('lena.jpg');
lena_fft2 = fft2(lena);
lena_H32 = lena_fft2.*H;
lena_h32_ifft = ifft2(lena_H32);
% figure; imshow(abs(lena)); title('Original version');
figure; imshow(abs(lena_h32_ifft),[]); title('Afer filtering (\sigma = 32)');
```



比較 sigma = 4 與 sigma = 32 的 filter，我們可以發現 filter 從四角是白色變成像是兩片為透明的黑色片沒有對準疊在一起的圖形。

Compared with the previous image with sigma 4, and this image with sigma 32, we can find that the image with sigma 32 is more blurred than the previous image.

(d) The filtered image in part (c) also suffers from an additional distortion due to a spatial-domain aliasing effect in the circular convolution. To eliminate this artifact, consider both the image and the filter h[m, n] as 512 × 512 size images using zero-padding in each dimension. Now perform the frequency-domain filtering and comment on the resulting filtered image.

```
fprintf('5(d)\n');
```

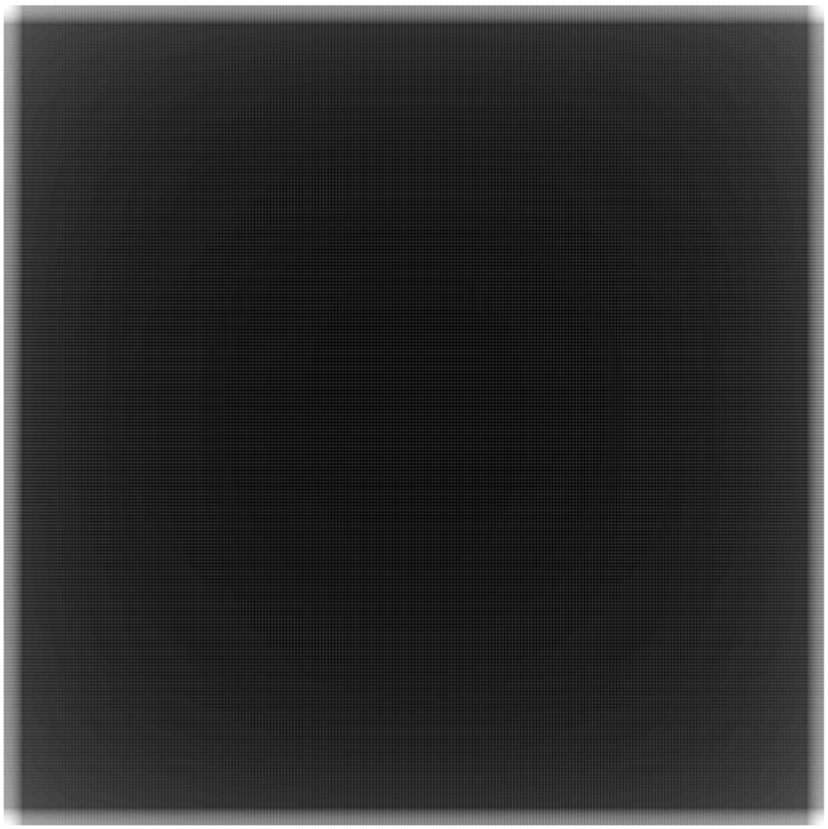
```
5(d)
```

```


sigma = 32;
h = zeros(256, 256);
for m = -128:1:127
    for n = -128:1:127
        h(m+129, n+129) = ((2*pi*(sigma)^2)^-1).*exp(-1*((m^2)+(n^2))./(2*(sigma)^2));
    end
end

h_5d = padarray(h, [128,128]);

H_5d = ifftshift(h_5d);
H_5d = fft2(H_5d)
```

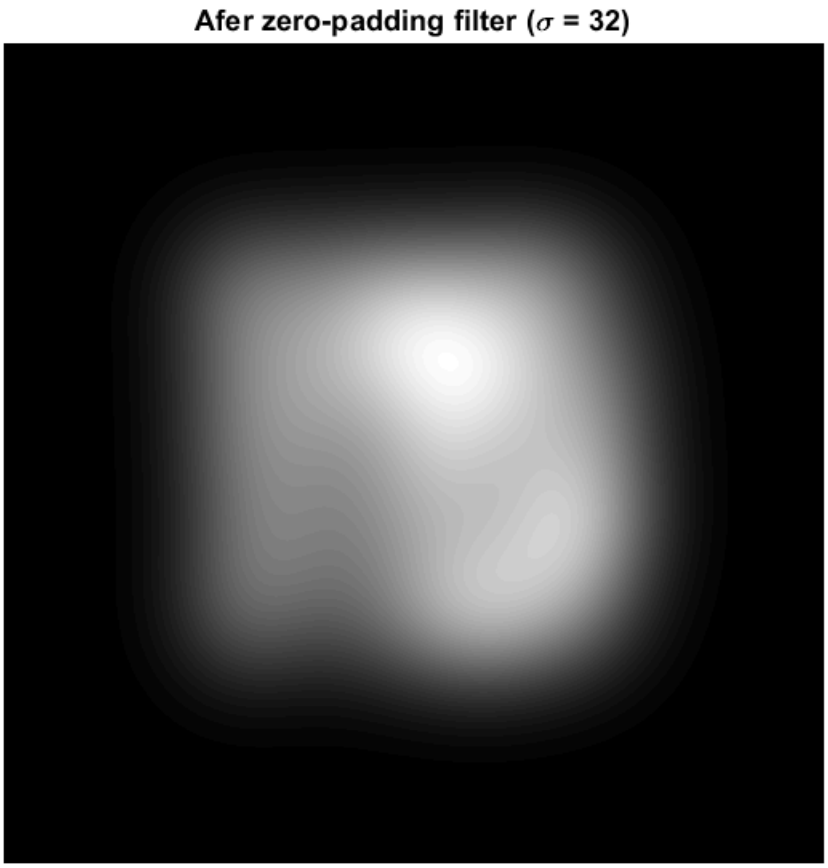
```
H_5d = 512×512 complex
   0.9999 + 0.0000i   0.9257 + 0.0000i   0.7346 + 0.0000i   0.4995 - 0.0000i ···
   0.9257 + 0.0000i   0.8571 + 0.0000i   0.6802 + 0.0000i   0.4625 - 0.0000i
   0.7346 + 0.0000i   0.6802 + 0.0000i   0.5397 + 0.0000i   0.3670 - 0.0000i
   0.4995 - 0.0000i   0.4625 - 0.0000i   0.3670 - 0.0000i   0.2496 - 0.0000i
   0.2911 + 0.0000i   0.2696 + 0.0000i   0.2139 + 0.0000i   0.1455 - 0.0000i
   0.1455 + 0.0000i   0.1347 + 0.0000i   0.1069 + 0.0000i   0.0727 + 0.0000i
   0.0623 + 0.0000i   0.0577 + 0.0000i   0.0458 + 0.0000i   0.0311 - 0.0000i
   0.0228 - 0.0000i   0.0211 - 0.0000i   0.0168 - 0.0000i   0.0114 - 0.0000i
   0.0071 + 0.0000i   0.0066 + 0.0000i   0.0053 - 0.0000i   0.0036 - 0.0000i
   0.0020 + 0.0000i   0.0018 + 0.0000i   0.0014 + 0.0000i   0.0010 + 0.0000i
     ⋮
```

```
H_5d_abs = abs(H_5d); H_5d_log = log(H_5d_abs);
figure; imshow(H_5d_log,[]); title('Filter with zero-padding and \sigma = 32');
```

Filter with zero-padding and σ = 32



```
lena = imread('lena.jpg');
lena_d = padarray(lena, [128 128]);
lena_fft2 = fft2(lena_d);
lena_H32d = lena_fft2.*H_5d;
lena_h32d_ifft = ifft2(lena_H32d);
% figure; imshow(abs(lena_d)); title('Original version');
figure; imshow(abs(lena_h32d_ifft),[]); title('Afer zero-padding filter (\sigma = 32)');
```



我們發現經過 zero-padding 處理過的圖形較沒有經過 zero-padding 處理的圖形來的模糊，甚而言之，有經過 zero-padding 處理過的圖形有邊界扭曲的情形發生。

(e) Repeat part (b) for σ = 4 but now using the frequency response 1 − H[k, l] for the filtering. Compare the resulting filtered image with that in (b).

```
fprintf('5(e)\n');
```

```
5(e)
```

```
sigma = 4;
h = zeros(256, 256);
for m = -128:1:127
    for n = -128:1:127
        h(m+129, n+129) = ((2*pi*(sigma)^2)^-1).*exp(-1*((m^2)+(n^2))./(2*(sigma)^2));
    end
end

H = ifftshift(h);
H = fft2(H);
```

```
H_5e = 1 - H;
H_5e_abs = abs(H_5e); H_5e_log = log(H_5e_abs);
figure; imshow(H_5e_log,[]); title('1 - H Filter with \sigma = 4');
```


```
lena = imread('lena.jpg');
lena_fft2 = fft2(lena);
lena_H4e = lena_fft2.*H_5e;
lena_h4e_ifft = ifft2(lena_H4e);
% figure; imshow(abs(lena)); title('Original version');
figure; imshow(abs(lena_h4e_ifft),[]); title('Afer (1 - H) filter (\sigma = 4)');
```

從 5_e 的結果可發現 5_d 的圖形的人物輪廓比較模糊，但是同時圖片比較白。