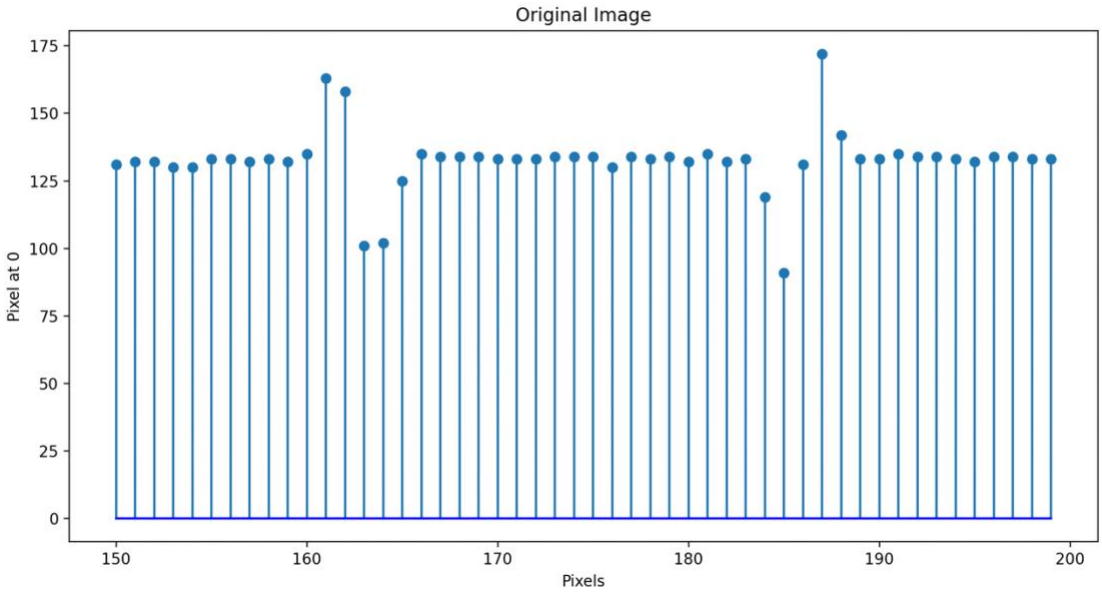
```
import numpy as np
import matplotlib.pyplot as plt
import skimage.io as io
from PIL import Image
import cv2

elvis_image = cv2.imread("/Users/kasi/Downloads/elvis-1.bmp")

columnMin = 150
columnMax = 200
row = 120
region = elvis_image[row, columnMin:columnMax, 0]
x = np.arange(columnMin, columnMax)

plt.grid(True)
plt.figure(figsize=(12, 6))
plt.stem(x, region, basefmt='b-')
plt.xlabel('Pixels')
plt.ylabel('Pixel at 0')
plt.title('Original Image')
plt.show()
```



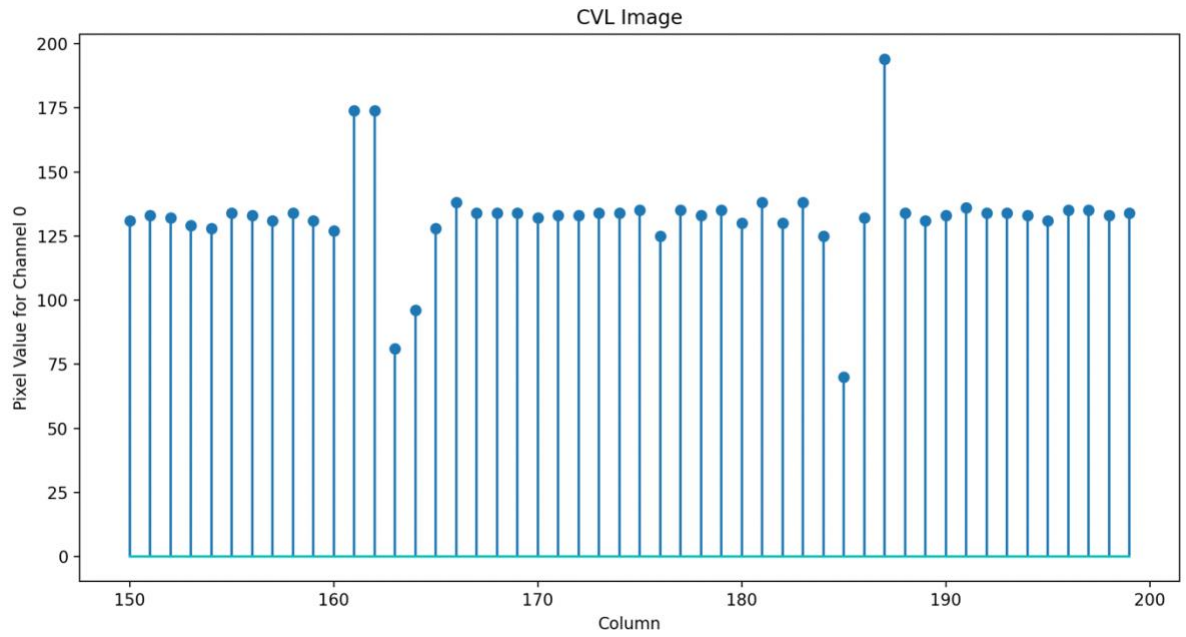
1b) The convolved image is sharper than the original image. The edges are more defined than the original image.

```
kernel = np.array([[0, -0.25, -0],
                   [-0.25,  2, -0.25],
                   [0, -0.25, 0]])

convolved_image = cv2.filter2D(elvis_image, -1, kernel)

#comparing both images
plt.figure(figsize=(12, 6))
plt.subplot(121)
plt.title('Original Image')
plt.imshow(cv2.cvtColor(elvis_image, cv2.COLOR_BGR2RGB))
plt.axis('off')

plt.subplot(122)
plt.title('Convolved Image')
plt.imshow(cv2.cvtColor(convolved_image, cv2.COLOR_BGR2RGB))
plt.axis('off')
plt.show()
```

1c)

As we can see when we compare the two stem plots, one of the original image and one for the convolved image, we can see that the stem plot of the original image has data points at (160, 130) and at (161, 160). The pixel difference between the two columns is 30 pixels. The convolved image has points at (160, 125) and (161, 175) and the pixel difference between these two points is 50 pixels. It is clear to see that the difference in the convolved image is greater than the original image. This is because we are making the image sharper and the edges more defined in the convoled image.

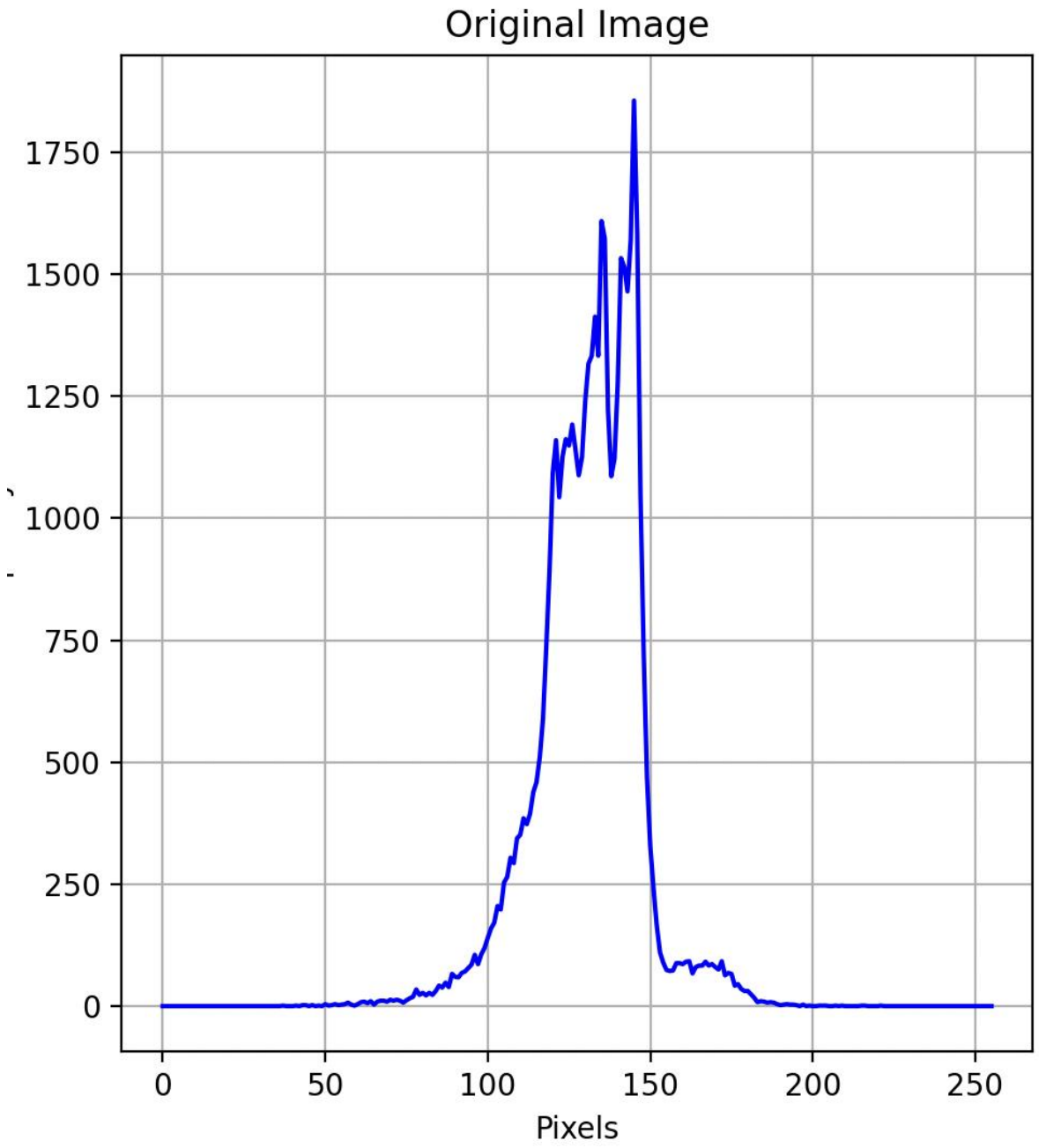
```
region = convolved_image[row, columnMin:columnMax, 0]
x = np.arange(columnMin, columnMax)
plt.grid(True)
plt.figure(figsize=(12, 6))
plt.stem(x, region, basefmt='c-')
plt.title('CVL Image')
plt.xlabel('Pixels')
plt.ylabel('Pixels at 0')
plt.show()
```

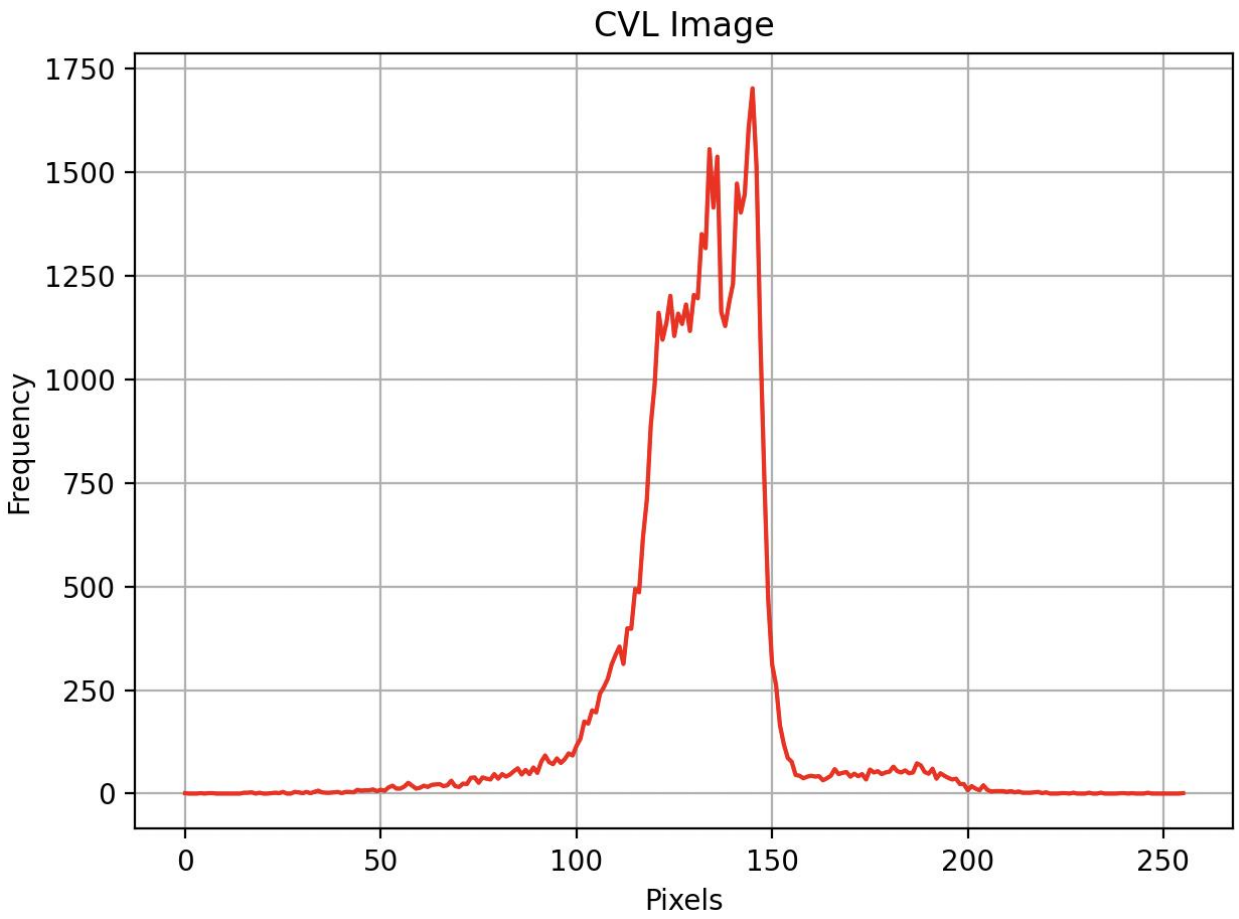


1d)

```
#original
conv_gray = cv2.cvtColor(convolved_image, cv2.COLOR_BGR2GRAY)
og_gray = cv2.cvtColor(elvis_image, cv2.COLOR_BGR2GRAY)
cvlhistogram = cv2.calcHist([conv_gray], [0], None, [256], [0, 256])
originalhistogram = cv2.calcHist([og_gray], [0], None, [256], [0, 256])

plt.figure(figsize=(12, 6))
```

```
plt.subplot(1, 2, 1)
plt.grid(True)
plt.plot(originalhistogram, color='b')
plt.xlabel('Pixels')
plt.ylabel('Frequency')
plt.title('Original Image')
plt.show()

#cvl image
plt.grid(True)
plt.plot(cvlhistogram, color='r')
plt.xlabel('Pixels')
plt.ylabel('Frequency')
plt.title('CVL Image')
plt.tight_layout()
plt.show()
```


Original Image
1750
1500
1250
1000
750
500
250
0
0
50
100
150
200
250
Pixels



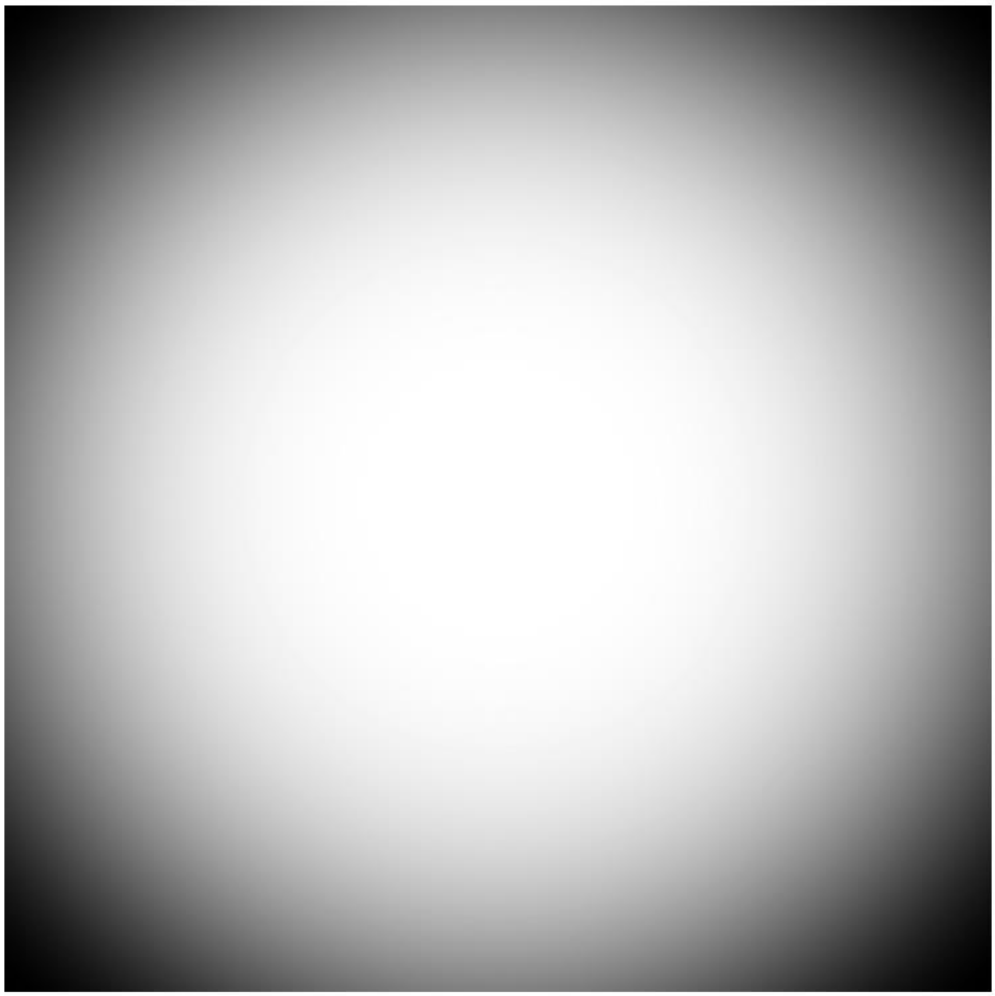
Because the kernel enhances the image, we see the histogram in the CVL image have more frequent higher data points than the original image.

2a)

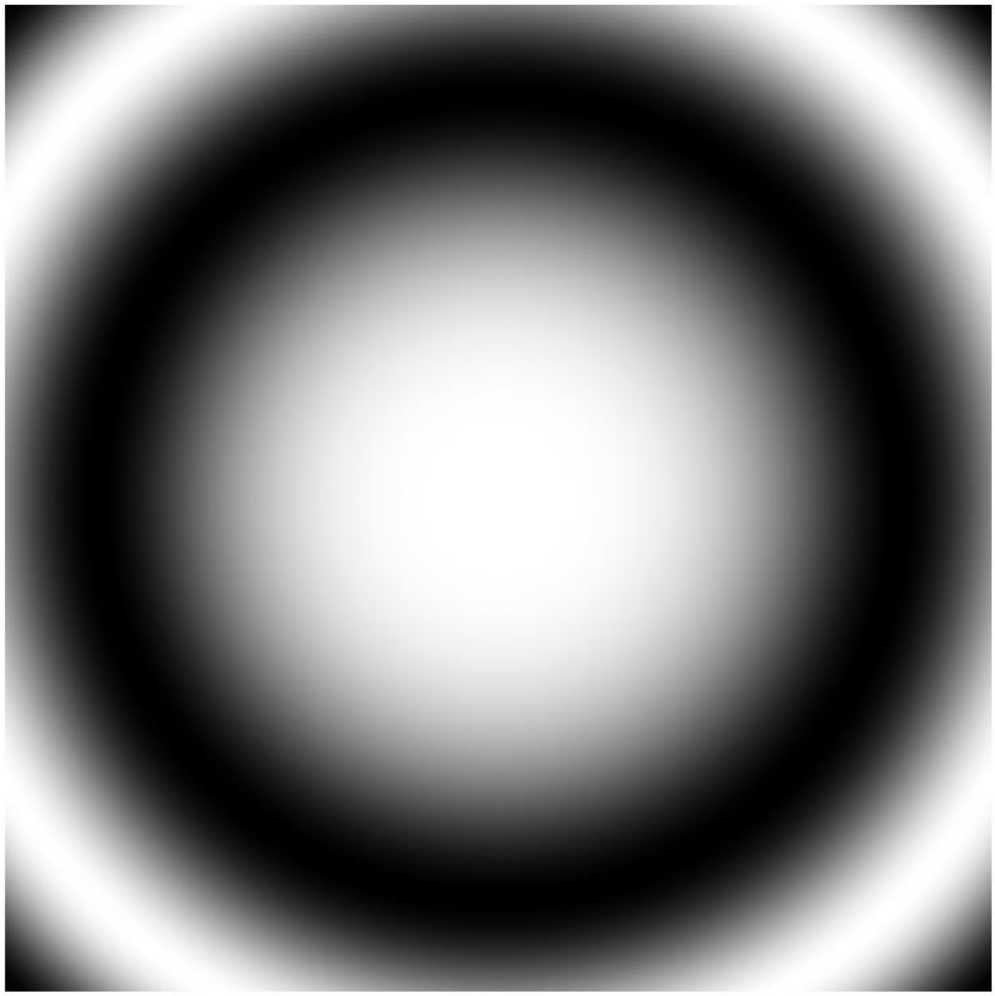
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation
plt.rcParams['animation.ffmpeg_path'] = '/Users/kasi/Downloads/ffmpeg'


def make_zone_plate(rows, cols, f):
    x = np.linspace(-1, 1, cols)
    y = np.linspace(-1, 1, rows)
    X, Y = np.meshgrid(x, y)
    radius_squared = X**2 + Y**2
    zone_plate = 0.5 + 0.5 * np.cos(np.pi * f * radius_squared)
    return zone_plate
```

```
def show_zone_plate(zone_plate):
    plt.imshow(zone_plate, cmap='gray')
    plt.title(f"Zone Plate (f = {f})")
    plt.axis('off')
    plt.show()


f_values = [0.5, 1.0, 1.5, 2.0, 2.5, 20, 50, 75, 100]

for f in f_values:
    rows, cols = 512, 512
    zone_plate = make_zone_plate(rows, cols, f)
    show_zone_plate(zone_plate)
```

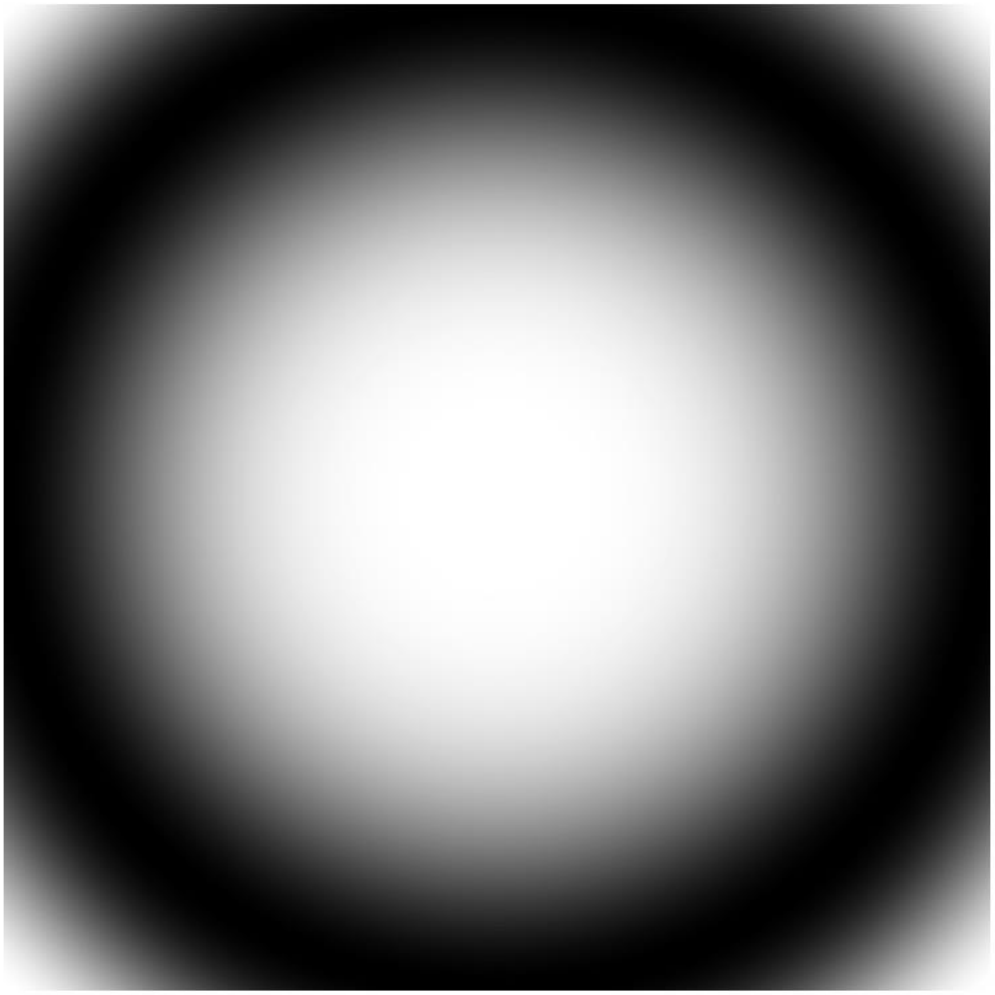
Zone Plate (f = 0.5)



Zone Plate (f = 1.5)



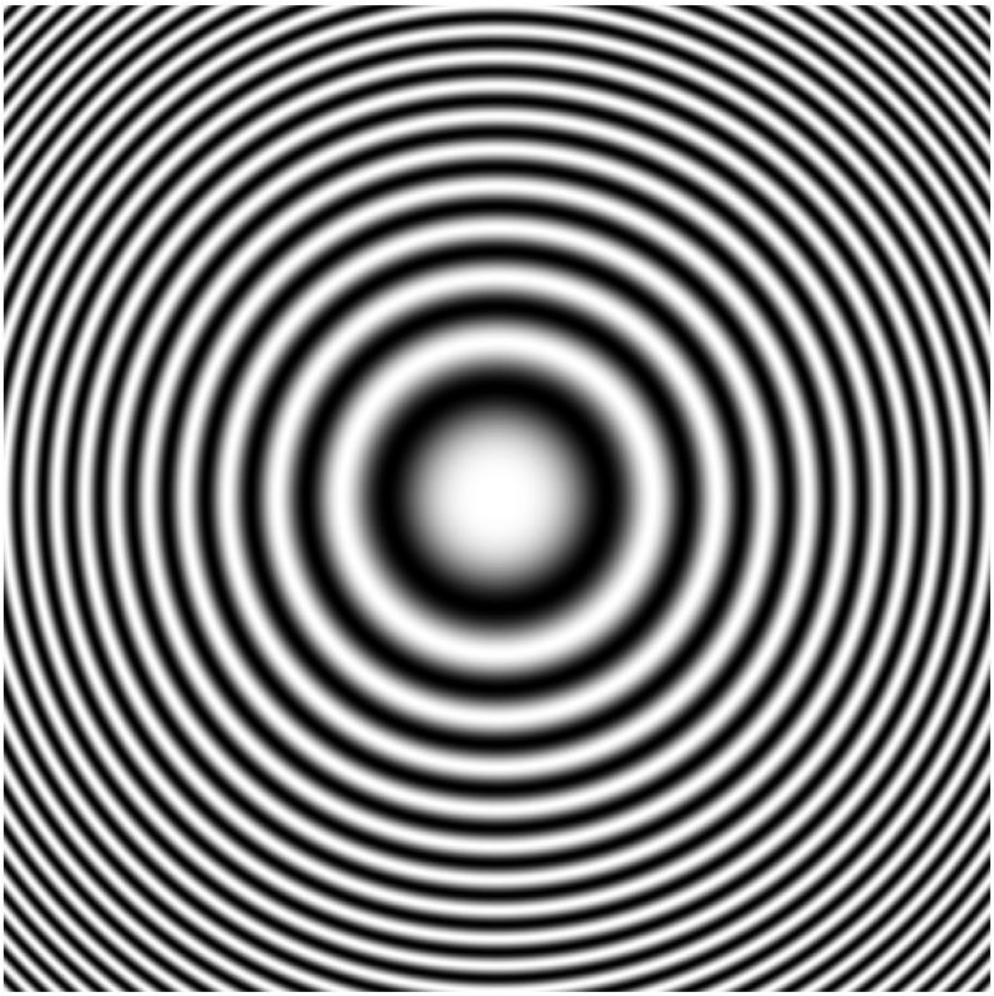
Zone Plate (f = 1.0)



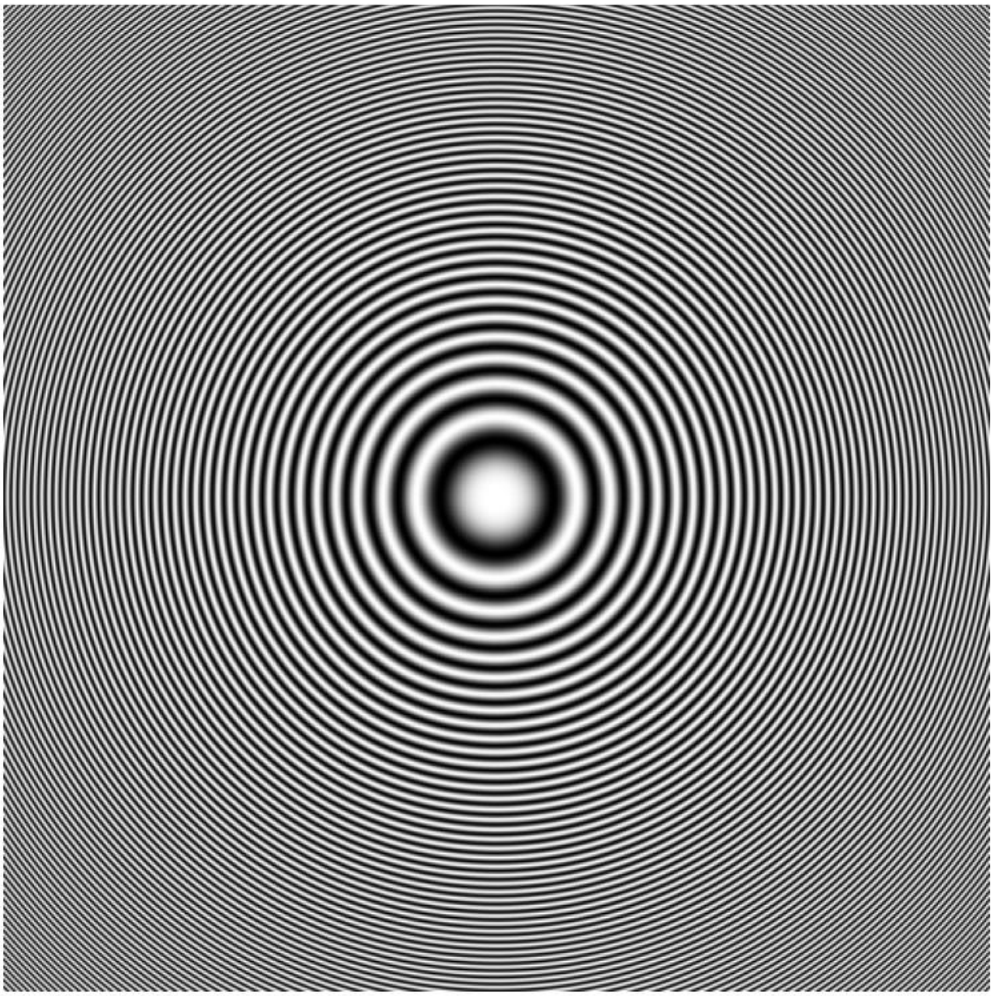
Zone Plate (f = 2.5)

Zone Plate (f = 2.0)

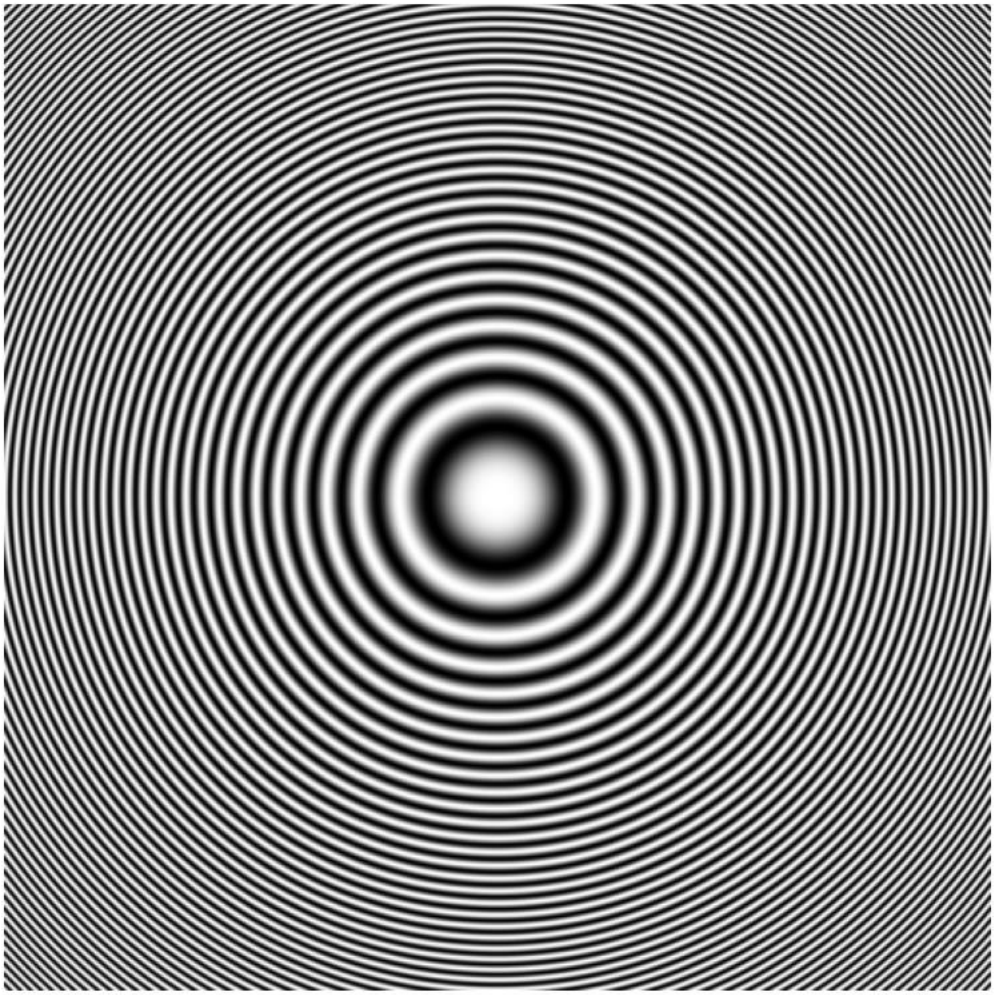
Zone Plate (f = 20)

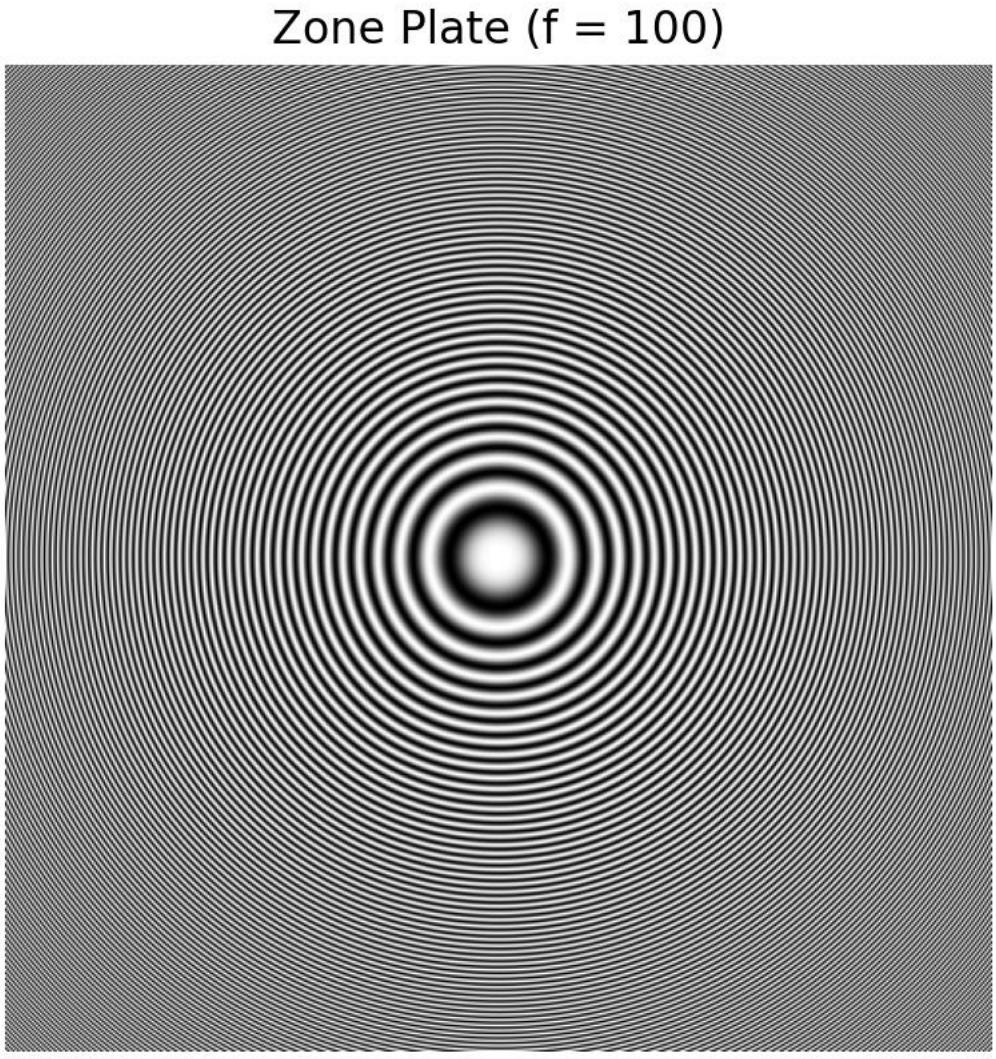


Zone Plate (f = 75)



Zone Plate (f = 50)





2b)

```
NROWS, NCOLS = 240, 320


def zone_plate(f, Nx, Ny):
    x = np.linspace(-1, 1, Nx)
    y = np.linspace(-1, 1, Ny)
    X, Y = np.meshgrid(x, y)
    radius_squared = X**2 + Y**2
    zone_plate = 0.5 + 0.5 * np.cos(np.pi * f * radius_squared)
    return zone_plate


fig = plt.figure()
f = 1
```

```
def animate(i):
    image = zone_plate(f+1.0*i, NCOLS, NROWS)
    plt.imshow(image, cmap='gray')
ani = animation.FuncAnimation(fig, animate, frames=50)
FFwriter = animation.FFMpegWriter(codec='rawvideo')
ani.save('zoneplate.mov', writer=FFwriter)
```

2c)

```
NROWS, NCOLS = 240, 320

def zone_plate(f, Nx, Ny):
    x = np.linspace(-1, 1, Nx)
    y = np.linspace(-1, 1, Ny)
    X, Y = np.meshgrid(x, y)
    radius_squared = X**2 + Y**2
    zone_plate = 0.5 + 0.5 * np.cos(np.pi * f * radius_squared)
    return zone_plate

fig = plt.figure()
f = 1
def animate(i):
    image = zone_plate(f+1.0*i, NCOLS, NROWS)
    plt.imshow(image, cmap='gray')
ani = animation.FuncAnimation(fig, animate, frames=50)
FFwriter = animation.FFMpegWriter(codec='rawvideo')
ani.save('zoneplate.mov', writer=FFwriter)
```