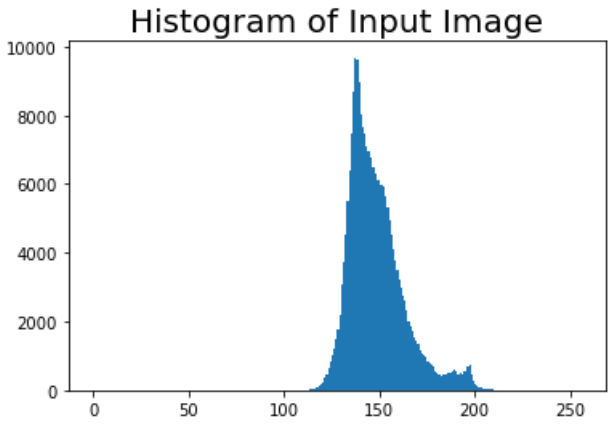
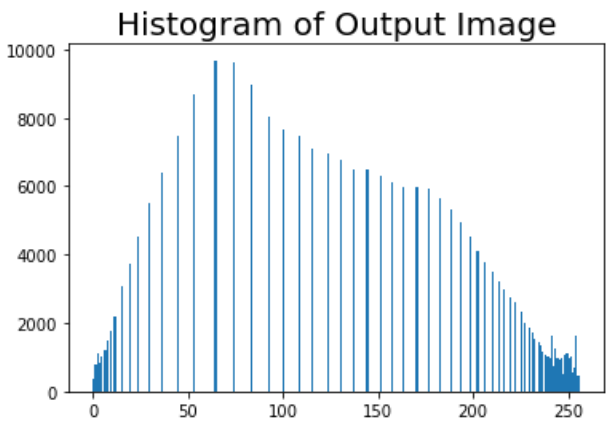
# Histogram Equalization

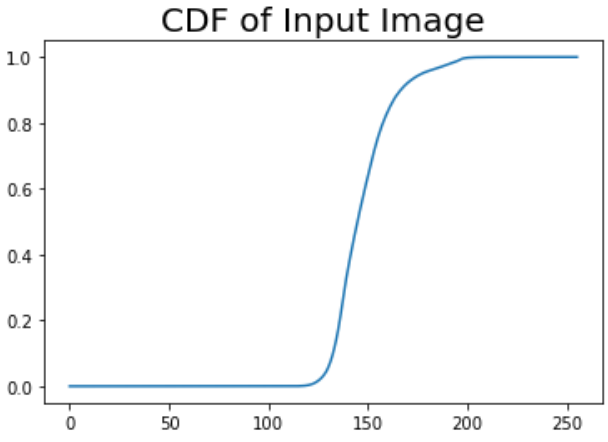
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
img = cv2.imread('histogram.jpg',0)
plt.imshow(cv2.cvtColor(img,0))
plt.show()
plt.title(label="Histogram of Input Image",fontsize=20,color="black")
plt.hist(img.ravel(),256,[0,256])
plt.show()
im_H = img.shape[0]
im_W = img.shape[1]
frame = im_H * im_W
output = np.zeros((im_H,im_W))
icount = np.zeros(256)
for i in range(im_H):
    for j in range(im_W):
        intensity = img[i,j]
        icount[intensity] += 1
pdf = icount/frame
cdf = np.zeros(256)
cdf[0] = pdf[0]
for i in range(1,256):
    cdf[i] = cdf[i-1]+pdf[i]
for i in range(im_H):
    for j in range(im_W):
        intensity = img[i,j]
        output[i,j] = np.round(255*cdf[intensity])
plt.title(label="CDF of Input Image", fontsize=20,color="black")
plt.plot(cdf)
plt.show()
output = output.astype(np.uint8)
icounto = np.zeros(256)
for i in range(im_H):
    for j in range(im_W):
        intensity = output[i,j]
        icounto[intensity] += 1
pdfo = icounto/frame
cdfo = np.zeros(256)
cdfo[0] = pdfo[0]
for i in range(1,256):
    cdfo[i] = cdfo[i-1]+pdfo[i]
plt.imshow(cv2.cvtColor(output,0))
plt.show()
plt.title(label="Histogram of Output Image",fontsize=20,color="black")
plt.hist(output.ravel(),256,[0,256])
plt.show()
plt.title(label="CDF of Output Image",fontsize=20,color="black")
plt.plot(cdfo)
plt.show()
```

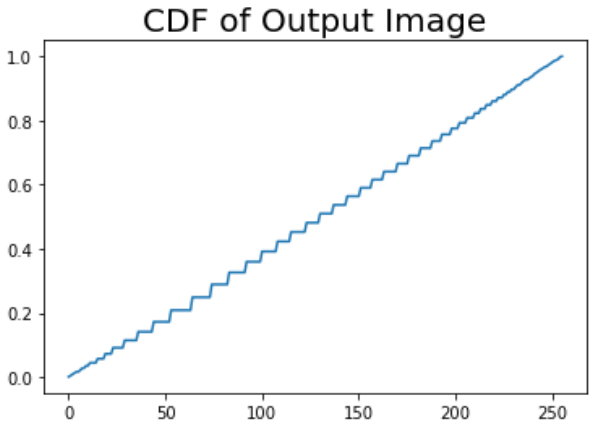
Fig 1.1: Input Image

Fig 1.2: Output Image



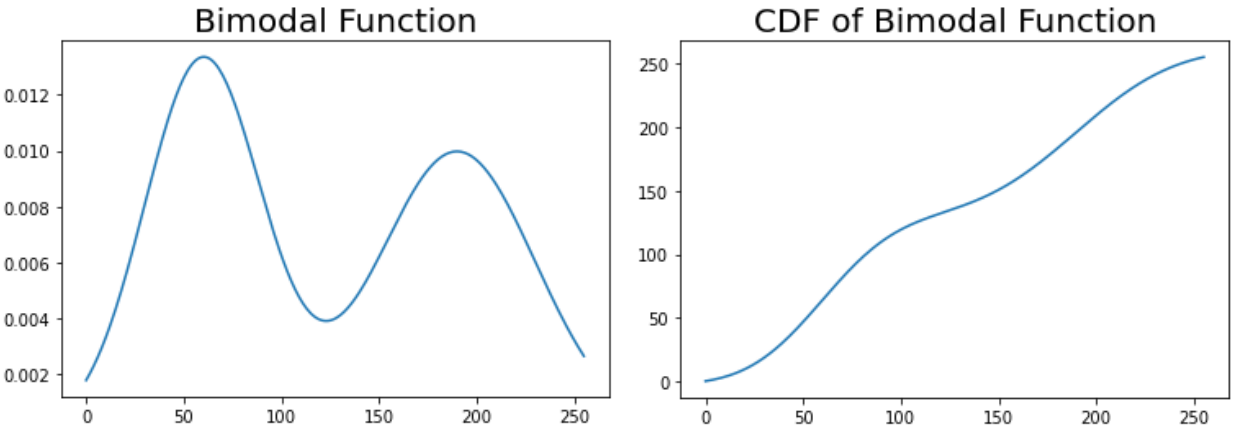
# Histogram Matching(Specification)

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
def search(a,arr):
    for i in range(256):
        if(a == arr[i]):
            return i
        elif (a < arr[i]):
            b = arr[i]
            c = arr[i-1]
            if((b-a)>(a-c)):
                return i-1
            else:
                return i
    return 255
def gaussian(miu,sigma):
    variance = sigma*sigma
    constant = 1/(np.sqrt(2*3.1416)*sigma)
    g = np.empty(shape=256)
    for i in range (256):
        g[i] = np.exp(-((i-miu)**2)/(2*variance))*constant
    return g
u1,sigma1 = [int(x) for x in input('Enter the values of miu1 and sigma1:').split()]
u2,sigma2 = [int(x) for x in input('Enter the values of miu2 and sigma2:').split()]
img = cv2.imread("histogram.jpg",cv2.IMREAD_GRAYSCALE)
im_H = img.shape[0]
im_W = img.shape[1]
frame = im_H*im_W
intensities = np.zeros(256)
cdf = np.zeros(256,np.float32)
output = np.zeros((im_H,im_W),np.uint8)
for i in range(im_H):
    for j in range(im_W):
        intensities[img[i,j]] += 1
pdf = intensities/frame
cdf[0] = pdf[0]
for i in range(1,len(pdf)):
    cdf[i] = cdf[i-1] + pdf[i]
cdf *= 255
g1 = gaussian(u1, sigma1)
g2 = gaussian(u2,sigma2)
g = g1 + g2
frameg = np.sum(g)
pdfg = g/frameg
cdfg = np.zeros(256)
cdfg[0] = pdfg[0]
for i in range(1,len(pdf)):
    cdfg[i] = cdfg[i-1] + pdfg[i]
cdfg *= 255
cdfg = np.round(cdfg).astype(np.uint8)
```

```
for i in range(im_H):
    for j in range(im_W):
        a = np.round(cdf[img[i,j]])
        b = search(a,cdfg)
        output[i,j] = b
```

Fig 2.1: Input Image

Fig 2.2: Output Image




miu1 = 60, sigma1 = 30, miu2 = 190 sigma2 = 40