Introduction to Digital Image Processing

Homework 6

December 2, 2020

Student: Ivan Arevalo
Perm Number: 5613567
Email: ifa@ucsb.com

Department of Electrical and Computer Engineering, UCSB

Problem 1, 2

ECE 178 HW 6 Ivan Arevalo

Problem 1

$$f[m,n] = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}$$

a) $F[k,l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n]\, e^{-j2\pi\left(\frac{km}{M} + \frac{ln}{N}\right)}$ $k = 0, \ldots, M-1$; $l = 0, \ldots, N-1$

$= \sum_{m=0}^{M-1} e^{-j2\pi \frac{km}{M}} \sum_{n=0}^{N-1} f[m,n]\, e^{-j2\pi \frac{ln}{N}}$

$F[0,0] = \sum_{m=0}^{1} \underbrace{e^{-j2\pi \frac{0m}{2}}}_{1} \left(\sum_{n=0}^{N-1} f[m,n] \underbrace{e^{-j2\pi \frac{0n}{2}}}_{1} \right)$

$= f[0,0] + f[0,1] + f[1,0] + f[1,1] = \boxed{6}$

$F[0,1] = \sum_{m=0}^{1} \underbrace{e^{-j2\pi \frac{0m}{M}}}_{1} \left(\sum_{n=0}^{1} f[m,n]\, e^{-j2\pi \frac{1 \cdot n}{2}} \right)$

$= f[0,0] + f[0,1] \underbrace{e^{-j\pi}}_{-1} + f[1,0] + f[1,1] \underbrace{e^{-j\pi}}_{-1} = \boxed{-2}$

$F[1,0] = \sum_{m=0}^{1} e^{-j2\pi \frac{1m}{2}} \sum_{n=0}^{1} f[m,n] \underbrace{e^{-j2\pi \frac{0 \cdot n}{2}}}_{1}$

$= f[0,0] + f[0,1] + f[1,0] \underbrace{e^{-j\pi}}_{-1} + f[1,1] \underbrace{e^{-j\pi}}_{-1} = \boxed{-4}$

$F[1,1] = \sum_{m=0}^{1} e^{-j2\pi \frac{1m}{2}} \sum_{n=0}^{1} f[m,n]\, e^{-j2\pi \frac{1n}{2}}$

$= f[0,0] + f[0,1] e^{-j\pi} + f[1,0] e^{-j\pi} + f[1,1] \underbrace{e^{-j\pi} e^{-j\pi}}_{1} = \boxed{0}$

$$\boxed{F[k,l] = \begin{bmatrix} 6 & -2 \\ -4 & 0 \end{bmatrix}}$$

b) Basis Functions $\quad e^{j2\pi\left(\frac{km}{M}+\frac{\ell n}{N}\right)}$

$$\begin{bmatrix} e^{j2\pi\left(\frac{0\cdot m}{2}+\frac{0n}{2}\right)} & e^{j2\pi\left(\frac{0m}{2}+\frac{1\cdot n}{2}\right)} \\ e^{j2\pi\left(\frac{1m}{2}+\frac{0n}{2}\right)} & e^{-j2\pi\left(\frac{1\cdot m}{2}+\frac{1n}{2}\right)} \end{bmatrix}$$

$$\text{Basis Functions} = \begin{bmatrix} 1 & e^{j\pi n} \\ e^{j\pi m} & e^{j\pi(m+n)} \end{bmatrix}$$

c) $$f[m,n] = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{\ell=0}^{N-1} F[k,\ell]\,e^{j2\pi\left(\frac{km}{M}+\frac{\ell n}{N}\right)}$$

$$f[0,0] = \frac{1}{4}\left(F[0,0]\cdot 1 + F[0,1]\cdot \overset{1}{\cancel{e^{j\pi 0}}} + F[1,0]\cdot \overset{1}{\cancel{e^{j\pi 0}}} + F[1,1]e^{j\pi 0}\right) = 0$$

$$f[0,1] = \frac{1}{4}\left(F[0,0]\cdot 1 + F[0,1]\overset{-1}{\cancel{e^{j\pi}}} + F[1,0]\cdot \overset{1}{\cancel{e^{j\pi 0}}} + F[1,1]\overset{-1}{\cancel{e^{j\pi}}}\right) = 1$$

$$f[1,0] = \frac{1}{4}\left(F[0,0]\cdot 1 + F[0,1]\overset{1}{\cancel{e^{j\pi 0}}} + F[1,0]\overset{-1}{\cancel{e^{j\pi}}} + F[1,1]\overset{-1}{\cancel{e^{j\pi}}}\right) = 2$$

$$f[1,1] = \frac{1}{4}\left(F[0,0]\cdot 1 + F[0,1]\overset{-1}{\cancel{e^{j\pi}}} + F[1,0]\cdot \overset{-1}{\cancel{e^{j\pi}}} + F[1,1]\cdot \overset{1}{\cancel{e^{j2\pi}}}\right) = 3$$

$$\boxed{f[m,n] = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}}$$ ✓

Problem 2

$$f[m,n] = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}$$

$$h[m,n] = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

find $f \circledast h$

a) 2D-DFT of h to get $H[k,l]$

$$H[k,l] = \sum_{m=0}^{1} e^{-j2\pi \frac{km}{2}} \sum_{n=0}^{1} h[m,n]\, e^{-j2\pi \frac{ln}{2}}$$

$$H[0,0] = h[0,0] + h[0,1] + h[1,0] + h[1,1] = \boxed{10}$$

$$H[0,1] = h[0,0] + h[0,1]\cancelto{-1}{e^{-j\pi}} + h[1,0] + h[1,1]\cancelto{-1}{e^{-j\pi}} = \boxed{-2}$$

$$H[1,0] = h[0,0] + h[0,1] + h[1,0]\cancelto{-1}{e^{-j\pi}} + h[1,1]\cancelto{-1}{e^{-j\pi}} = \boxed{-4}$$

$$H[1,1] = h[0,0] + h[0,1]e^{-j\pi} + h[1,0]e^{-j\pi} + h[1,1] = \boxed{0}$$

$$G[k,l] = F[k,l] \cdot H[k,l]$$

$$= \begin{bmatrix} 6 & -2 \\ 4 & 0 \end{bmatrix} \cdot \begin{bmatrix} 10 & -2 \\ -4 & 0 \end{bmatrix} = \begin{bmatrix} 60 & 4 \\ -16 & 0 \end{bmatrix}$$

$$g[m,n] = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} F[k,l]\, e^{j2\pi\left(\frac{km}{M} + \frac{ln}{N}\right)}$$

$$g[0,0] = \frac{1}{4}\left(G[0,0] + G[0,1]\cdot\cancelto{1}{e^{j\pi 0}} + G[1,0]\cdot\cancelto{1}{e^{j\pi 0}} + G[1,1]\cancelto{1}{e^{j\pi 0}}\right) = \boxed{12}$$

$$g[0,1] = \frac{1}{4}\left(G[0,0] + G[0,1]\cdot\cancelto{-1}{e^{j\pi}} + G[1,0]\cancelto{1}{e^{j\pi 0}} + G[1,1]\cdot\cancelto{-1}{e^{j\pi}}\right) = \boxed{10}$$

$$g[1,0] = \frac{1}{4}\left(G[0,0] + G[0,1]\cancelto{1}{e^{j\pi 0}} + G[1,0]\cancelto{-1}{e^{j\pi}} + G[1,1]\cancelto{-1}{e^{j\pi}} \right) = \boxed{20}$$

$$g[1,1] = \frac{1}{4}\left(G[0,0] + G[0,1]\cancelto{-1}{e^{j\pi}} + G[1,0]\cdot\cancelto{-1}{e^{j\pi}} + G[1,1]\cancelto{1}{e^{j2\pi}} \right) = \boxed{18}$$

$$g[m,n] = \begin{bmatrix} 12 & 10 \\ 20 & 18 \end{bmatrix}$$

b) $f[m,n] \circledast h[m,n] = g[m,n]$

$$\begin{bmatrix} 1 & 0 & 1 \\ 3 & 2 & 3 \\ 1 & 0 & 3 \end{bmatrix} * \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$= \begin{bmatrix} 4+6+2 & 3+4+3 \\ 12+6+2 & 8+9+1 \end{bmatrix} = \begin{bmatrix} 12 & 10 \\ 20 & 18 \end{bmatrix} \checkmark$$

Problem 3

Figure 1: tree-per-channel-hist-eq

Figure 2: tree-y-channel-hist-eq

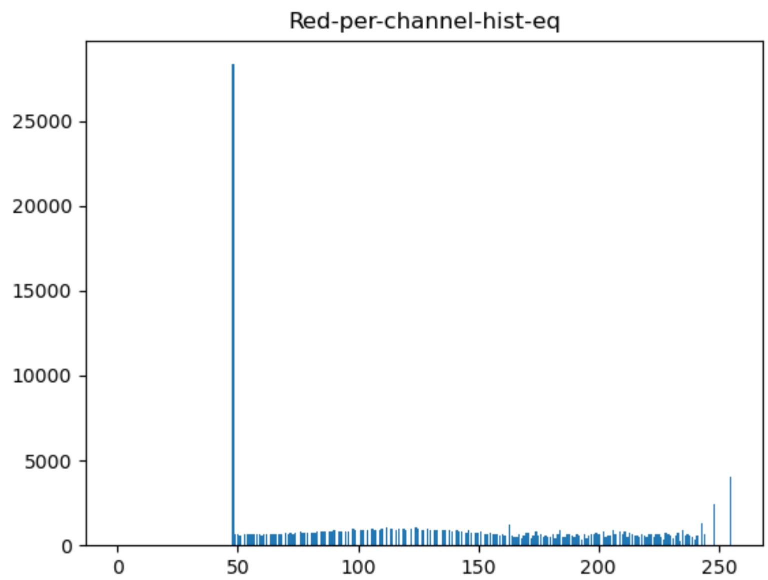

Figure 3: Red-per-channel-hist-eq

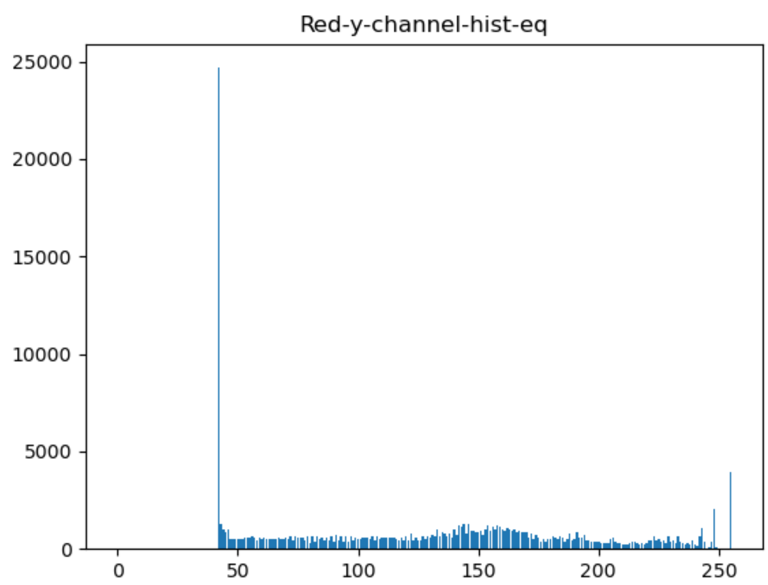

Figure 4: Red-y-channel-hist-eq

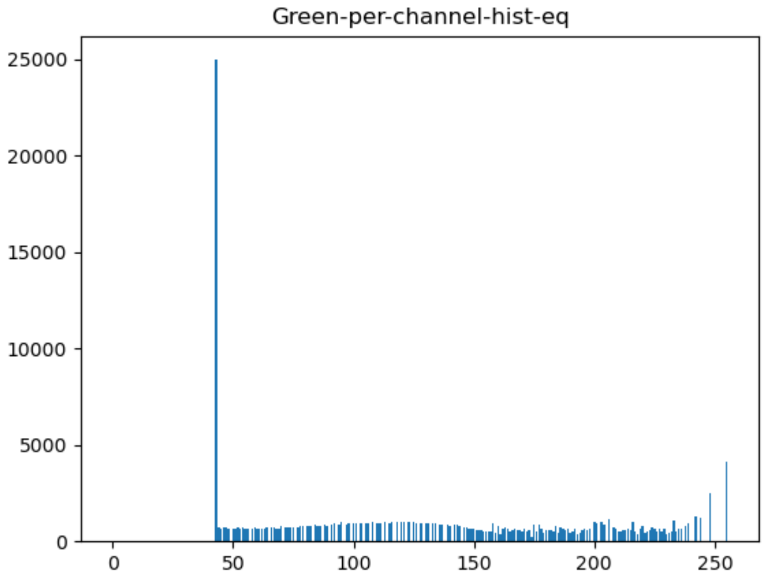

Figure 5: Green-per-channel-hist-eq

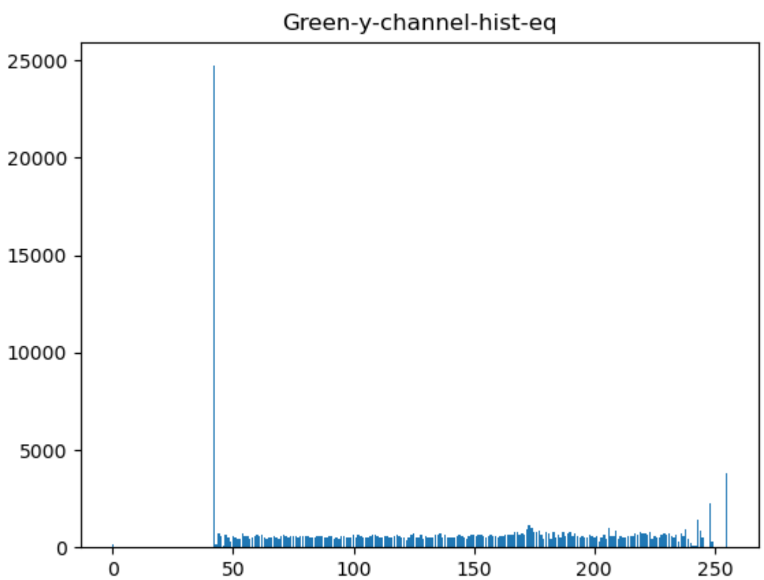

Figure 6: Green-y-channel-hist-eq

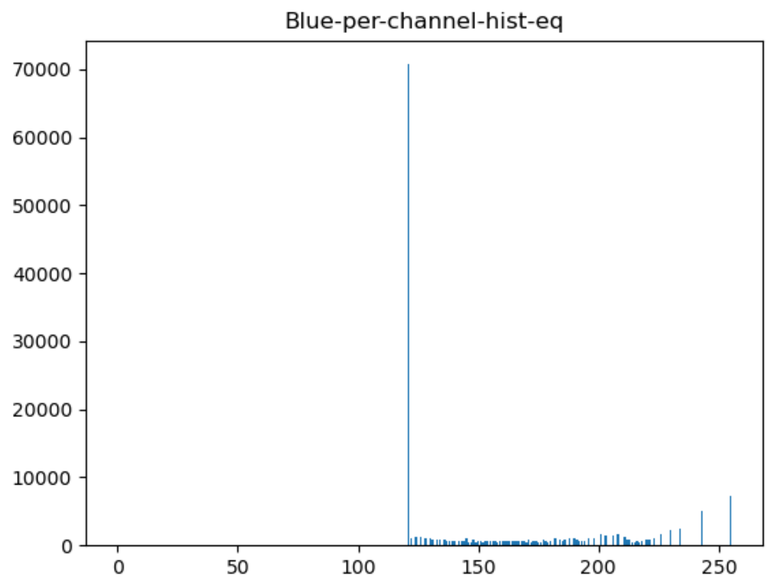

Figure 7: Blue-per-channel-hist-eq

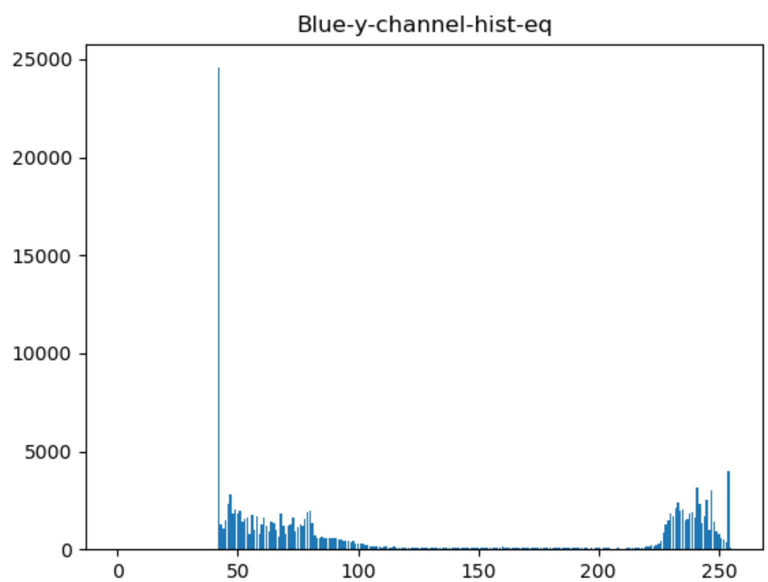

Figure 8: Blue-y-channel-hist-eq

Furthermore Comparing the histogram for each apprach, we can see that the y-channel- histograms take full advantage of the dynamic range while per channel RGB equalization doesn't, especially in the blue histogram.

Code

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


def histogram_EQ(channel, num_levels = 256):
    MN = channel.size
    histogram = np.zeros(256)
    out_channel = np.zeros(channel.shape)
    for j in range(channel.shape[0]):
        for i in range(channel.shape[1]):
            pxl_lvl = channel[j, i]
            histogram[pxl_lvl] += 1

    norm_histogram = histogram/MN
    cdf_histogram = norm_histogram

    for i in range(1, cdf_histogram.size):
        cdf_histogram[i] = cdf_histogram[i] + cdf_histogram[i-1]

    eq_histogram = (num_levels - 1) * cdf_histogram
    round_eq_histogram = np.round(eq_histogram).astype(np.uint8)

    for j in range(channel.shape[0]):
        for i in range(channel.shape[1]):
            pxl_lvl = channel[j, i]
            new_pxl_lvl = round_eq_histogram[pxl_lvl]
            out_channel[j, i] = new_pxl_lvl

    return out_channel

def compute_histogram(channel):
    histogram = np.zeros(256)
    for j in range(channel.shape[0]):
        for i in range(channel.shape[1]):
            pxl_lvl = channel[j, i]
            histogram[pxl_lvl] += 1
    return histogram

# Approach 1: Perform histogram equalization on each of the red, green, blue channels
    separately
```

```
img = Image.open("tree-dark.png")
plt.imshow(img, cmap='gray', vmin=0, vmax=255)
plt.show()
img_pxl = np.asarray(img)
output_img = np.zeros(img_pxl.shape)
output_img2 = np.zeros(img_pxl.shape)
output_img3 = np.zeros(img_pxl.shape)

# Options: 0 for Linear transformation on pixel
# Options: 1 for Non-Linear transformation on pixel
# Options: 2 for Histogram EQ
option = 2

for i in range(img_pxl.shape[2]):
    plt.imshow(img_pxl[:, :, i], cmap='gray', vmin=0, vmax=255)
    plt.show()

    if option == 0:
        # Linear transformation on pixel
        im_min = np.min(img_pxl[:,:,i])
        im_max = np.max(img_pxl[:,:,i])
        output_img[:,:,i] = np.asarray((255 / (im_max - im_min)) * (img_pxl[:,:,i] -
            im_min))
        output_img[:,:,i] = np.round(output_img[:,:,i])
        plt.imshow(output_img[:, :, i].astype(np.uint8), cmap='gray', vmin=0, vmax=255)
        plt.show()

    elif option == 1:
        # Non-linear transformation on pixel (No specification on which method to use,
            chose this one)
        gamma = 0.4
        if i == 2:
            gamma = 0.25
        output_img2[:, :, i] = 255 * np.power(img_pxl[:,:,i]/255, gamma)
        output_img2[:, :, i] = np.round(output_img2[:, :, i])
        plt.imshow(output_img2[:, :, i].astype(np.uint8), cmap='gray', vmin=0, vmax=255)
        plt.show()

    elif option == 2:
        output_img3[:, :, i] = histogram_EQ(img_pxl[:,:,i])
        plt.imshow(output_img3[:, :, i].astype(np.uint8), cmap='gray', vmin=0, vmax=255)
        plt.show()
    else:
        print("Option unavailable")

if option == 0:
    plt.imshow(output_img.astype(np.uint8), cmap='gray', vmin=0, vmax=255)
    plt.show()

elif option == 1:
    plt.imshow(output_img2.astype(np.uint8), cmap='gray', vmin=0, vmax=255)
    plt.show()

elif option == 2:
```

```
    plt.imshow(output_img3.astype(np.uint8), cmap='gray', vmin=0, vmax=255)
    plt.show()
    plt.imsave("tree-per-channel-hist-eq.png", output_img3.astype(np.uint8), cmap='gray')

else:
    print("Option unavailable")

# Approach 2 : Convert YIQ color and perform histogram EQ
YIQ_transform = np.array([[0.299, 0.587, 0.114], [0.596, -0.275, -0.321], [0.212,
    -0.526, 0.311]])
yiq_img = np.zeros(img_pxl.shape)

for j in range(img_pxl.shape[0]):
    for i in range(img_pxl.shape[1]):
        yiq_img[j,i,:] = YIQ_transform@img_pxl[j,i,:]

plt.imshow(yiq_img.astype(np.uint8), vmin=0, vmax=255)
plt.show()

yiq_eq = yiq_img
yiq_eq[:,:,0] = histogram_EQ(np.round(yiq_img[:,:,0]).astype(np.uint8))

RGB_transform = np.array([[1.000, 0.956, 0.602], [1.000, -0.272, -0.647], [1.000,
    -1.108, 1.700]])
rgb_img = np.zeros(img_pxl.shape)

for j in range(img_pxl.shape[0]):
    for i in range(img_pxl.shape[1]):
        rgb_img[j,i,:] = RGB_transform@yiq_eq[j,i,:]

plt.imshow(rgb_img.astype(np.uint8), vmin=0, vmax=255)
plt.imsave('tree-y-channel-hist-eq.png', rgb_img.astype(np.uint8), vmin=0, vmax=255)
plt.show()

# Compare histograms
x_vec = np.arange(256)

Red_RGB_hist = compute_histogram(np.round(output_img3[:,:,0]).astype(np.uint8))
plt.bar(x_vec, Red_RGB_hist)
plt.title("Red-per-channel-hist-eq")
plt.savefig('Red-per-channel-hist-eq.png')
plt.clf()
Green_RGB_hist = compute_histogram(np.round(output_img3[:,:,1]).astype(np.uint8))
plt.bar(x_vec, Green_RGB_hist)
plt.title("Green-per-channel-hist-eq")
plt.savefig('Green-per-channel-hist-eq.png')
plt.clf()
Blue_RGB_hist = compute_histogram(np.round(output_img3[:,:,2]).astype(np.uint8))
plt.bar(x_vec, Blue_RGB_hist)
plt.title("Blue-per-channel-hist-eq")
plt.savefig('Blue-per-channel-hist-eq.png')
plt.clf()

Red_YIQ_hist = compute_histogram(np.round(rgb_img[:,:,0]).astype(np.uint8))
```

```
plt.bar(x_vec, Red_YIQ_hist)
plt.title("Red-y-channel-hist-eq")
plt.savefig('Red-y-channel-hist-eq.png')
plt.clf()
Green_YIQ_hist = compute_histogram(np.round(rgb_img[:,:,1]).astype(np.uint8))
plt.bar(x_vec, Green_YIQ_hist)
plt.title("Green-y-channel-hist-eq")
plt.savefig('Green-y-channel-hist-eq.png')
plt.clf()
Blue_YIQ_hist = compute_histogram(np.round(rgb_img[:,:,2]).astype(np.uint8))
plt.bar(x_vec, Blue_YIQ_hist)
plt.title("Blue-y-channel-hist-eq")
plt.savefig('Blue-y-channel-hist-eq.png')
plt.clf()
```