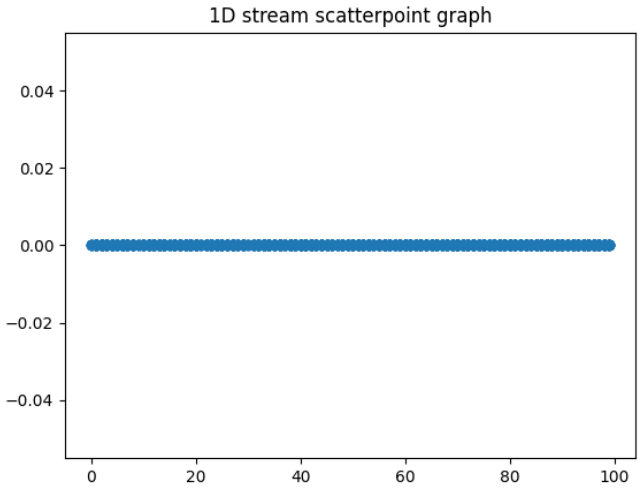
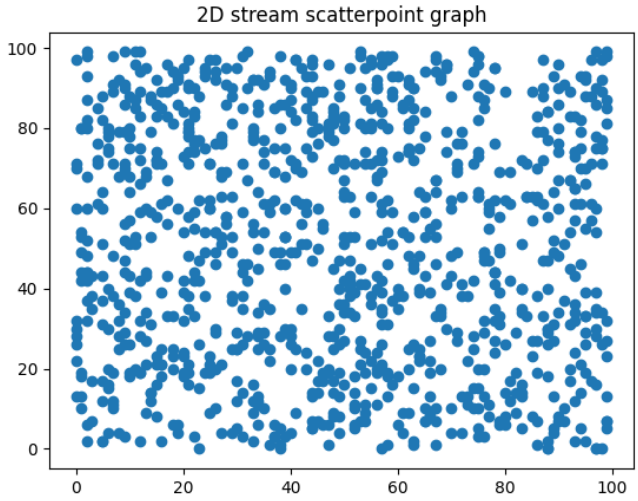
I. Random number generation （each 8 points）:

1. Generate three streams (1D, 2D) of random numbers with 1,000 samples, you may use the Matlab command rand.
2. Visualize the generated samples, you may use a scatterplot.

1D Stream Scatterplot

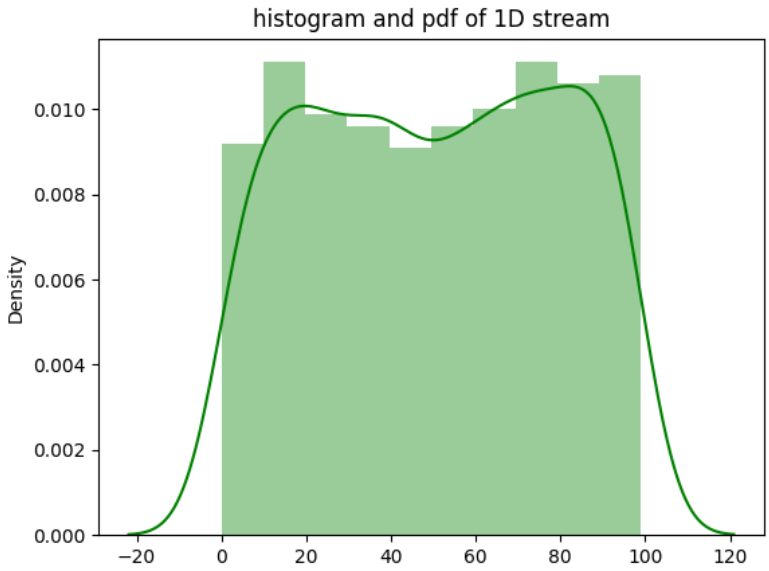


2D Stream Scatterplot

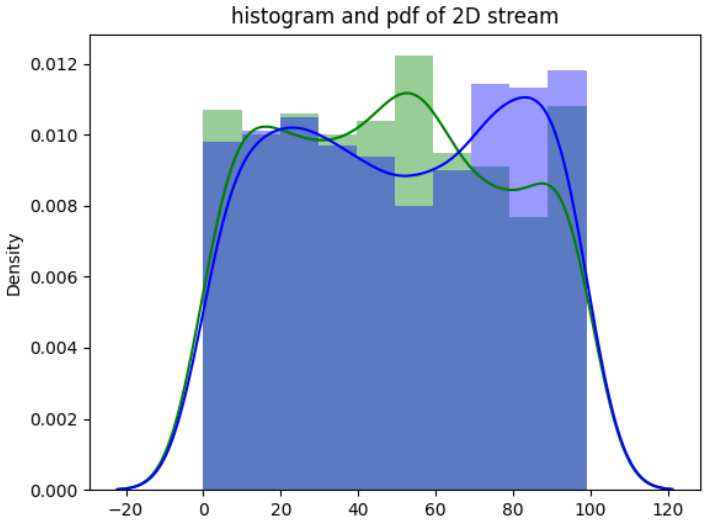


3. Compute the histogram of the three streams, then normalize them to become a probability density function (pdf).
4. Visualize the pdf's of the three streams. Are the samples uniformly distributed? Do the pdf's represent a standard uniform distributions? Comment.

1D PDF Visualization



2D PDF Visualization

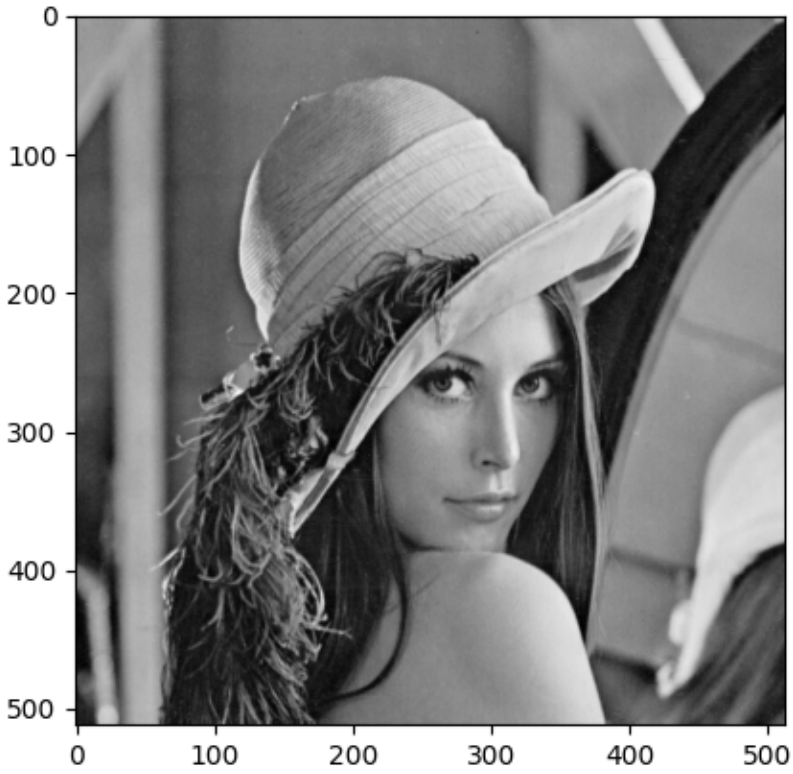


Code

```
'''
Author: SimonCK666 SimonYang223@163.com
Date: 2022-11-05 23:25:00
LastEditors: SimonCK666 SimonYang223@163.com
LastEditTime: 2022-11-06 14:14:14
FilePath: /Project1/lena.py
Description: 这是默认设置,请设置`customMade`, 打开koroFileHeader查看配置 进行设置: https://github.com/OBKoro1/koro1FileHeader/wiki/配置
'''
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from ReadBMP import ImageFile
import cv2

'''
1-2.1   Import LenaGrey to show and see the image.
'''
# import the lena picture
bmpFile = ImageFile()
lena = bmpFile.getBMP('lena.bmp')
lena_gray = cv2.cvtColor(lena, cv2.COLOR_BGR2GRAY) # transform it into greyscale
plt.imshow(lena_gray, cmap='gray')
plt.show()


def get_row_view(a):  # to get mode of the picture
    void_dt = np.dtype((np.void, a.dtype.itemsize * np.prod(a.shape[-1:])))
    a = np.ascontiguousarray(a)
    return a.reshape(-1, a.shape[-1]).view(void_dt).ravel()


def get_mode(lena_gray):  # to get mode of the picture
    unq, idx, count = np.unique(get_row_view(lena_gray), return_index=1, return_counts=1)
    return lena_gray.reshape(-1,lena_gray.shape[-1])[idx[count.argmax()]]


'''
1-2.2   Calculate its mean, standard deviation, median, min, max, and mode.
'''
mean = np.mean(lena_gray)
std = np.std(lena_gray)
median = np.median(lena_gray)
min = np.min(lena_gray)
max = np.max(lena_gray)
count = np.bincount(get_mode(lena_gray))
mode = np.argmax(count)
print("The mean value of the picture is: ", mean)
print("The standard deviation of the picture is: ", std)
print("The median value of the picture is: ", median)
print("The min value of the picture is: ", min)
print("The max deviation value of the picture is: ", max)
print("The mode value of the picture is: ", mode)

'''
1-2.3   Plot the histogram of the LenaGrey.
'''
plt.hist(lena_gray.ravel(),256,[0,256])
plt.title("Histogram of lena_gray")
plt.show()

'''
1-2.4   With the intensity as the third dimension (normalize it), plot its 3D shape
(although this is not its 3D shape but it has some 3D impression.
'''
xx, yy = np.mgrid[0:lena_gray.shape[0], 0:lena_gray.shape[1]]
fig = plt.figure(figsize=(15,15))
ax = fig.gca(projection='3d')
ax.plot_surface(xx, yy, lena_gray ,rstride=1, cstride=1, cmap=plt.cm.gray,linewidth=2)
ax.view_init(80, 30)
plt.title("3D shape with the intensity as the third dimension")
plt.show()
```

II. Image manipulation – the image LenaGrey is formed by 512x512 pixels with intensity from 0 to 255 (each 8 points)

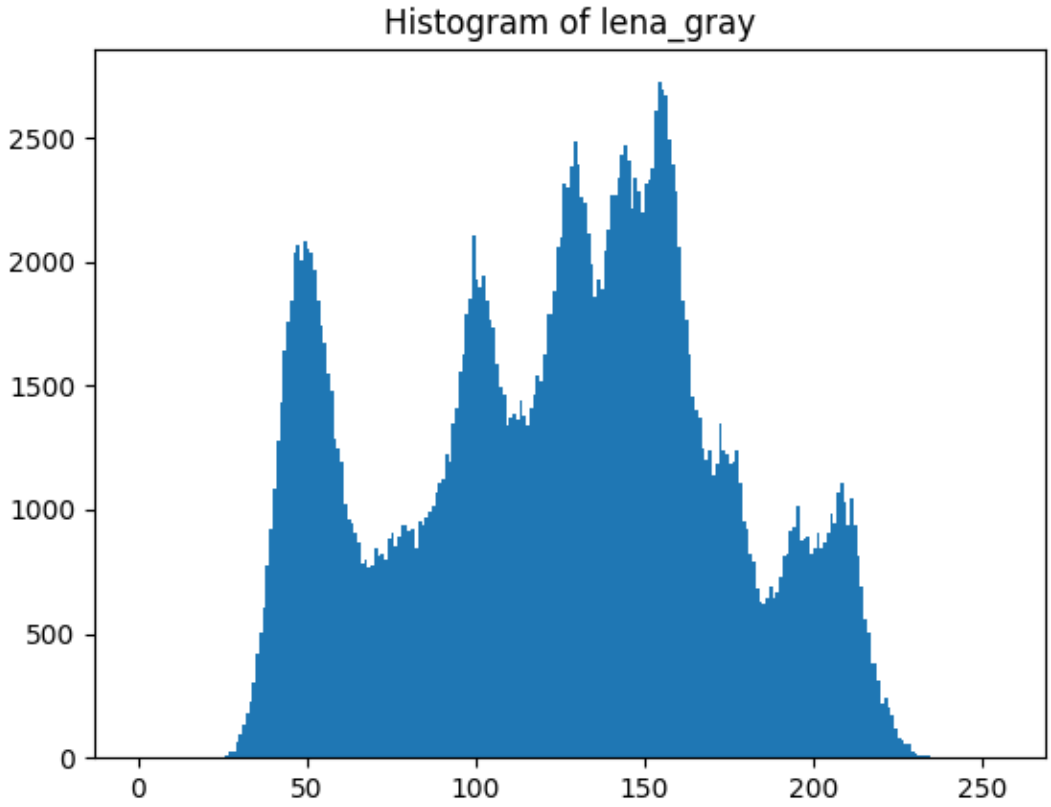
1. Import LenaGrey to show and see the image.



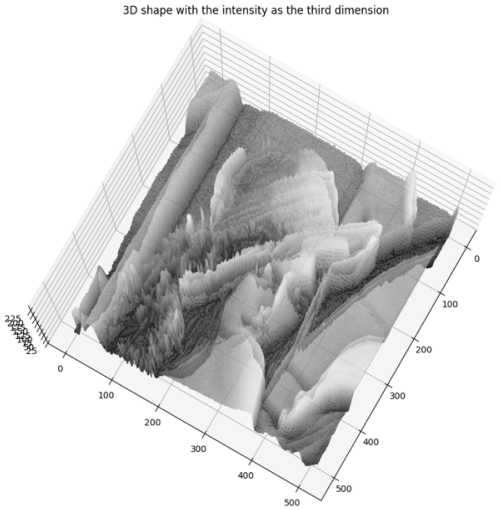
2. Calculate its mean, standard deviation, median, min, max, and mode.

```
The mean value of the picture is:  124.05046081542969
The standard deviation of the picture is:  47.853693850187504
The median value of the picture is:  129.0
The min value of the picture is:  25
The max deviation value of the picture is:  245
The mode value of the picture is:  131
```

3. Plot the histogram of the LenaGrey.



4. With the intensity as the third dimension (normalize it), plot its 3D shape (although this is not its 3D shape but it has some 3D impression.



5. Read BMP Code

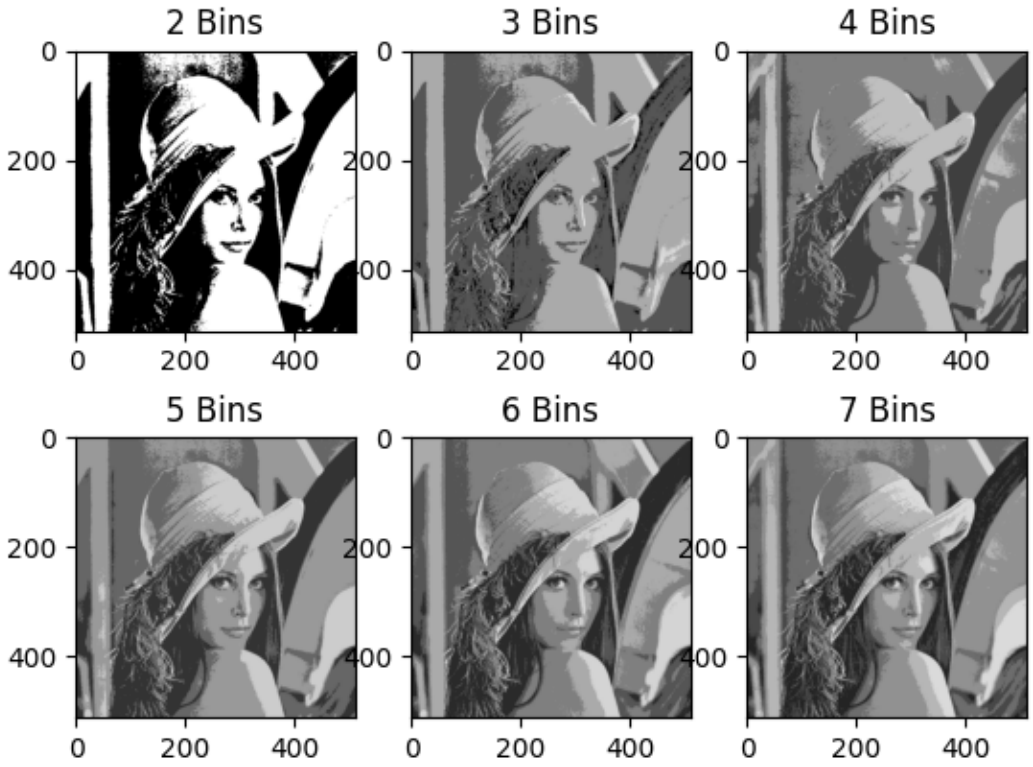
```python
'''
Author: SimonCK666 SimonYang223@163.com
Date: 2022-11-05 23:27:11
LastEditors: SimonCK666 SimonYang223@163.com
LastEditTime: 2022-11-06 12:56:51
FilePath: /Project1/readBMP.py
Description: 这是默认设置,请设置`customMade`, 打开koroFileHeader查看配置 进行设置: https://github.com/OBKoro1/koro1FileHeader/wiki/配置
'''
# -*- coding: UTF-8 -*-
import numpy as np
import struct
from PIL import Image

class ImageFile():

    def getBMP(self, filepath):
        f = open(filepath,'rb')
        f_type = str(f.read(2))
        file_size_byte = f.read(4)
        f.seek(f.tell()+4)
        file_ofset_byte = f.read(4)
        f.seek(f.tell()+4)
        file_wide_byte = f.read(4)
        file_height_byte = f.read(4)
        f.seek(f.tell()+2)
        file_bitcount_byte = f.read(4)

        f_size, = struct.unpack('i', file_size_byte)
        f_ofset, = struct.unpack('i', file_ofset_byte)
        f_wide, = struct.unpack('i', file_wide_byte)
        f_height, = struct.unpack('i', file_height_byte)
        f_bitcount, = struct.unpack('i', file_bitcount_byte)
        print("Type:", f_type, "Size:", f_size, "Bitmap data offset:", f_ofset, "Width:", f_wide, "Height:", f_height, "Bitmap:", f_bitcount)

        color_table = np.empty(shape=[256, 4], dtype=int)
        f.seek(54)
        for i in range(0, 256):
            b=struct.unpack('B', f.read(1))[0]
            g = struct.unpack('B', f.read(1))[0]
            r = struct.unpack('B', f.read(1))[0]
            alpha = struct.unpack('B', f.read(1))[0]
            color_table[i][0] = r
            color_table[i][1] = g
            color_table[i][2] = b
            color_table[i][3] = 255

        f.seek(f_ofset)
        img = np.empty(shape=[f_height, f_wide, 4], dtype=int)
        cout = 0

        for y in range(0, f_height):
            for x in range(0,f_wide):
                cout = cout + 1
                index = struct.unpack('B', f.read(1))[0]
                img[f_height - y - 1, x] = color_table[index]
            while cout % 4 != 0:
                f.read(1)
                cout = cout+1
        f.close()

        fimg = self.ndarry2image(img)

        return fimg

    def ndarry2image(self, ndarry):
        # ndarray to image
        ndarry = ndarry.astype("uint8")
        # ndarry = cv2.cvtColor(ndarry, cv2.COLOR_BGR2RGB)
        # ndarry = Image.fromarray(ndarry)
        # ndarry = ndarry.toqpixmap()
        return ndarry
```

6. Code

```python
'''
Author: SimonCK666 SimonYang223@163.com
Date: 2022-11-05 23:25:00
LastEditors: SimonCK666 SimonYang223@163.com
LastEditTime: 2022-11-06 14:14:14
FilePath: /Project1/lena.py
Description: 这是默认设置,请设置`customMade`, 打开koroFileHeader查看配置 进行设置: https://github.com/OBKoro1/koro1FileHeader/wiki/配置
'''
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from ReadBMP import ImageFile
import cv2

'''
1-2.1   Import LenaGrey to show and see the image.
'''
# import the lena picture
bmpFile = ImageFile()
lena = bmpFile.getBMP('lena.bmp')
lena_gray = cv2.cvtColor(lena, cv2.COLOR_BGR2GRAY) # transform it into greyscale
plt.imshow(lena_gray, cmap='gray')
plt.show()


def get_row_view(a):  # to get mode of the picture
    void_dt = np.dtype((np.void, a.dtype.itemsize * np.prod(a.shape[-1])))
    a = np.ascontiguousarray(a)
    return a.reshape(-1, a.shape[-1]).view(void_dt).ravel()


def get_mode(lena_gray):  # to get mode of the picture
    unq, idx, count = np.unique(get_row_view(lena_gray), return_index=1, return_counts=1)
    return lena_gray.reshape(-1,lena_gray.shape[-1])[idx[count.argmax()]]


'''
1-2.2   Calculate its mean, standard deviation, median, min, max, and mode.
'''
mean = np.mean(lena_gray)
std = np.std(lena_gray)
median = np.median(lena_gray)
min = np.min(lena_gray)
max = np.max(lena_gray)
count = np.bincount(get_mode(lena_gray))
mode = np.argmax(count)
print("The mean value of the picture is: ", mean)
print("The standard deviation of the picture is: ", std)
print("The median value of the picture is: ", median)
print("The min value of the picture is: ", min)
print("The max deviation value of the picture is: ", max)
print("The mode value of the picture is: ", mode)

'''
1-2.3   Plot the histogram of the LenaGrey.
'''
plt.hist(lena_gray.ravel(),256,[0,256])
plt.title("Histogram of lena_gray")
plt.show()

'''
1-2.4   With the intensity as the third dimension (normalize it), plot its 3D shape
(although this is not its 3D shape but it has some 3D impression.
'''
xx, yy = np.mgrid[0:lena_gray.shape[0], 0:lena_gray.shape[1]]
fig = plt.figure(figsize=(15,15))
ax = fig.gca(projection='3d')
ax.plot_surface(xx, yy, lena_gray ,rstride=1, cstride=1, cmap=plt.cm.gray,linewidth=2)
ax.view_init(80, 30)
plt.title("3D shape with the intensity as the third dimension")
plt.show()
```

III. Image range reduction – partition image intensity range into several bins and check to see how the image appearance change (each 13 points)

1. Partition image intensity into 2 bins, i.e., change the image to 1 bit image (binary image)
2. Partition image intensity into 3. 4, 5, 6, 7 bins to check image quality change compared with the original Lena image (8 bit image with intensity range from 0 to $2^8 - 1 = 255$).



3. Code

```
'''
Author: SimonCK666 SimonYang223@163.com
Date: 2022-11-05 23:25:00
LastEditors: SimonCK666 SimonYang223@163.com
LastEditTime: 2022-11-06 13:06:00
FilePath: /Project1/transformBins.py
Description: 这是默认设置,请设置`customMade`, 打开koroFileHeader查看配置 进行设置: https://github.com/OBKoro1/koro1FileHeader/wiki/配置
'''
import cv2
import matplotlib.pyplot as plt
import numpy as np
from ReadBMP import ImageFile

'''
change the image to 1 bit image (binary image)
the func complete the bin number is 2 operation
'''
def to_bin(lena2):
    # transform the picture into array type
    arr_lena = np.asarray(lena2)
    for row in range(len(arr_lena)):
        for column in range(len(arr_lena[row])):
            if (arr_lena[row][column] >= 255 / 2).any():
                # if this pixel's value is less than 127.5, make it white
                arr_lena[row][column] = 255
            else:
                # otherwise, make it black
                arr_lena[row][column] = 0
    plt.subplot(2, 3, 1)
    plt.title("2 Bins")
    plt.imshow(arr_lena)


def to_x_bins(lena, bins):
    if (bins == 2):
        to_bin(lena)
        return
    arr_lena = np.asarray(lena)
    for row in range(len(arr_lena)):
        for column in range(len(arr_lena[row])):
            for x in range(1, bins):
                # spilt = half of a bin
                split = 255 * (1 / bins) / 2
                if (arr_lena[row][column] < split).any():
                    # the first bin
                    arr_lena[row][column] = 0
                elif (arr_lena[row][column] >= 255 * (x / bins) - split).any():
                    # if this pixel is in this bin
                    if (arr_lena[row][column] < 255 * (x / bins) + split).any():
                        # change value
                        arr_lena[row][column] = 255 * (x / bins)
                elif (arr_lena[row][column] >= 255 - split).any():
                    # the last bin
                    arr_lena[row][column] = 255
    plt.subplot(2, 3, bins - 1)
    plt.imshow(arr_lena)
    plt.title("%s Bins" % bins)

'''
Visualize 2 to 7 bins
'''
for bin in range(2, 8):
    print(bin)
    # load bmp image
    bmpFile = ImageFile()
    lena2 = bmpFile.getBMP('lena.bmp')
    lena2 = cv2.cvtColor(lena2, cv2.COLOR_BGR2RGB)
    to_x_bins(lena2, bin)
plt.show()
```