**Biomedical Image Processing Report**

**Assignment 1 Part1**

**Creating histogram function, histogram with binary masking, max-min linear contrast stretch, and linear contrast stretch with 1% 5% intensity pixel discard**

**Dewi Endah Kharismawati**

**14231619 / dek8v5**

1. Histogram Computation

Histogram computation is an important component for image processing, it is used widely to represent image pixel intensity values. Histogram showing the number of pixels in an image at each intensity value in the image.

**In part a,** the assignment required to write a histogram function, with less loops as possible. Algorithm implemented in this part as follow:

Myimhist function:

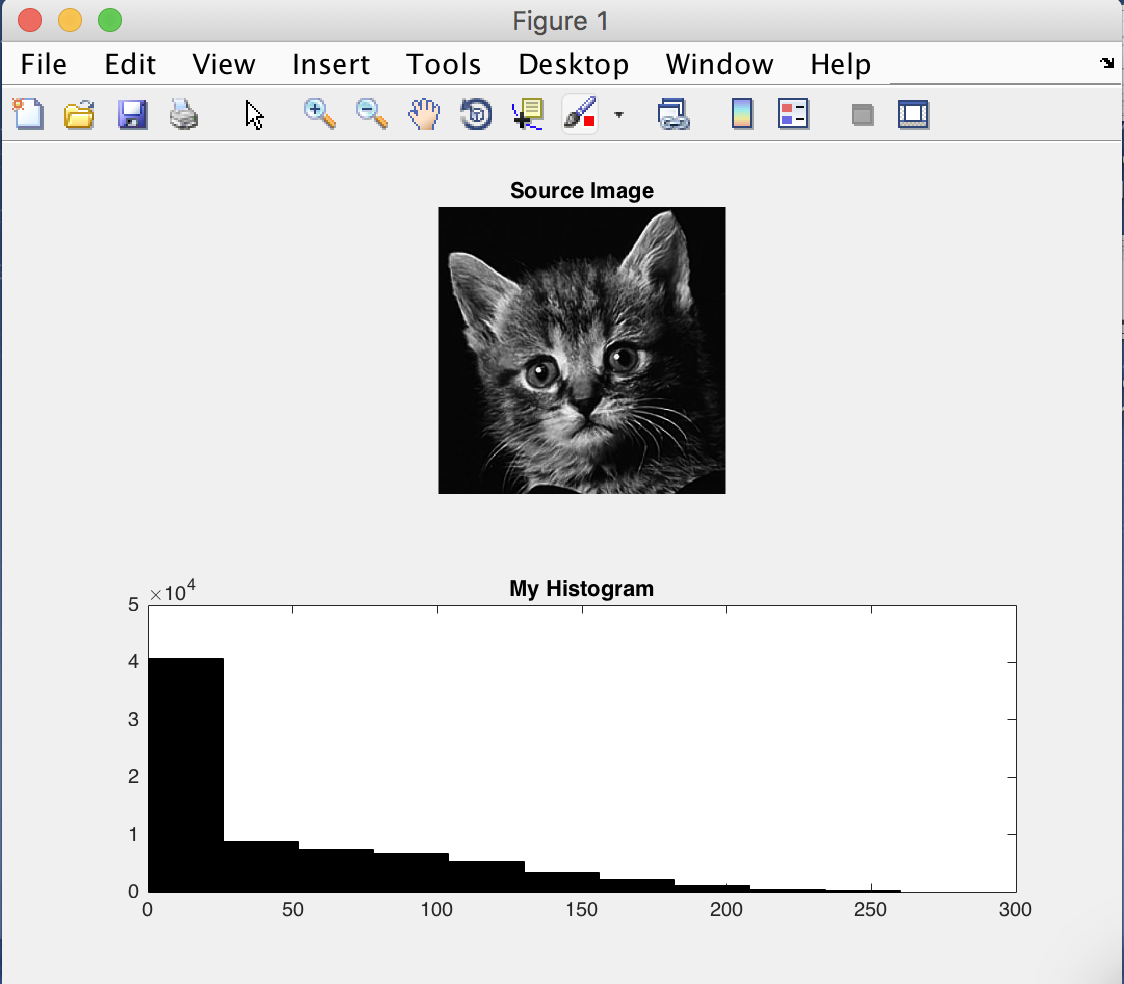
1. Convert image 2D matrix into 1D vector
2. Go over the vector, and retrieve each intensity to the designated histogram gray levels

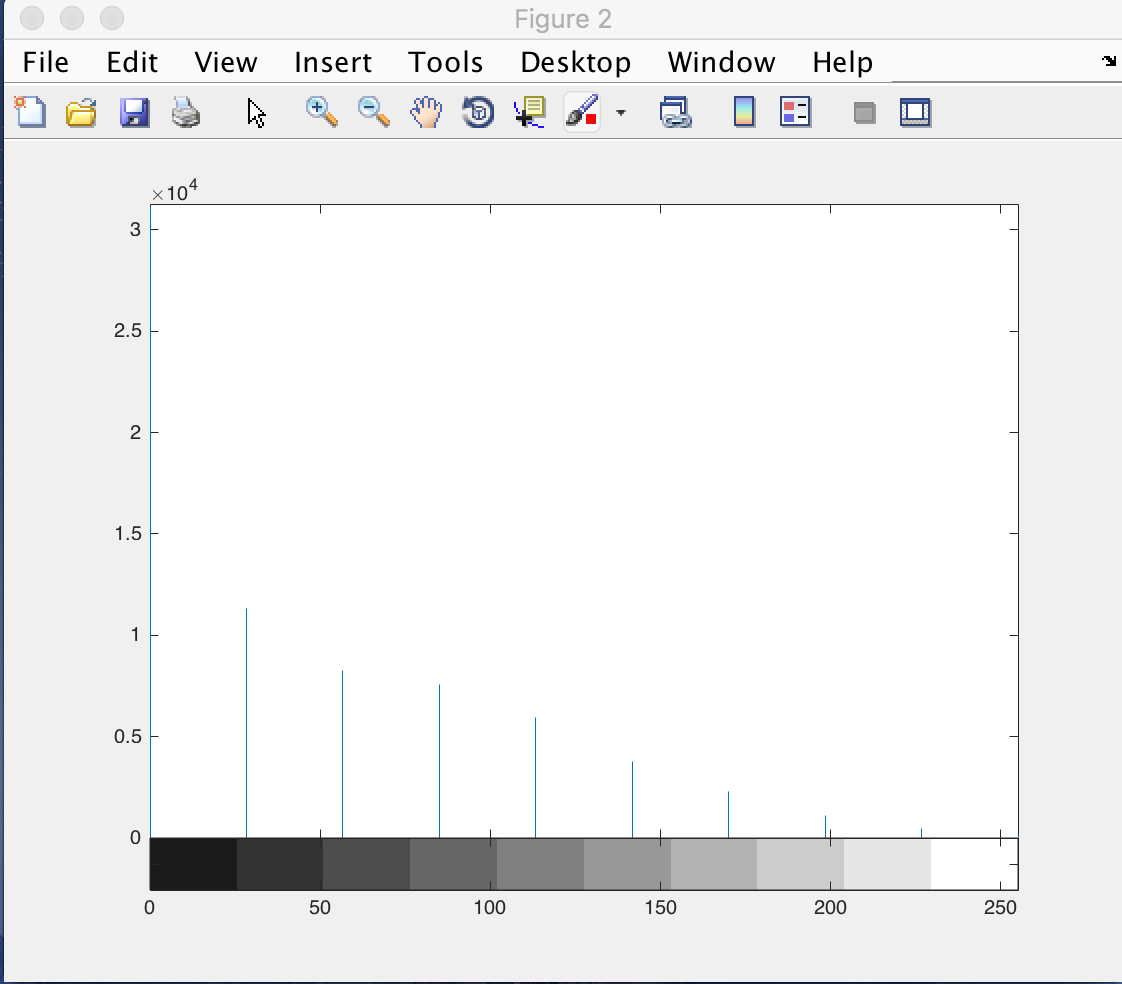
*histo(Ivector(i)+1) = histo(Ivector(i)+1)+1;*

1. Format in a way that the gray levels divided into desired bins
2. Display the bar

Main function:

1. Read the grayscale image
2. Convert the uint8 type into double
3. Call the myimhist that has been created

The result:



Based on this experiment, the time elapsed for both histogram are the following:

time for my own histogram function

Elapsed time is 0.024851 seconds.

time for build in imhist

Elapsed time is 0.367328 seconds.

In myimhist, it converts the 2D matrix into 1D vector. Then, it only utilizing 1 for loop to go through all the vector. And it shows that myimhist is a lot faster, because it process less loop iterations.

**Part b**, the more advance usage of histogram is performed. In this part, the experiment is done by compute image histogram of a region specified using a binary mask. This problem provides a binary mask, which only has 0 or 1 pixels. Whereas, the original image is an RGB color. Therefore, extraction is needed. In this case, extraction is being done in red layer. Algorithm that is implemented in this part is describe as follow:

Main: read images and call the myimhistmask function

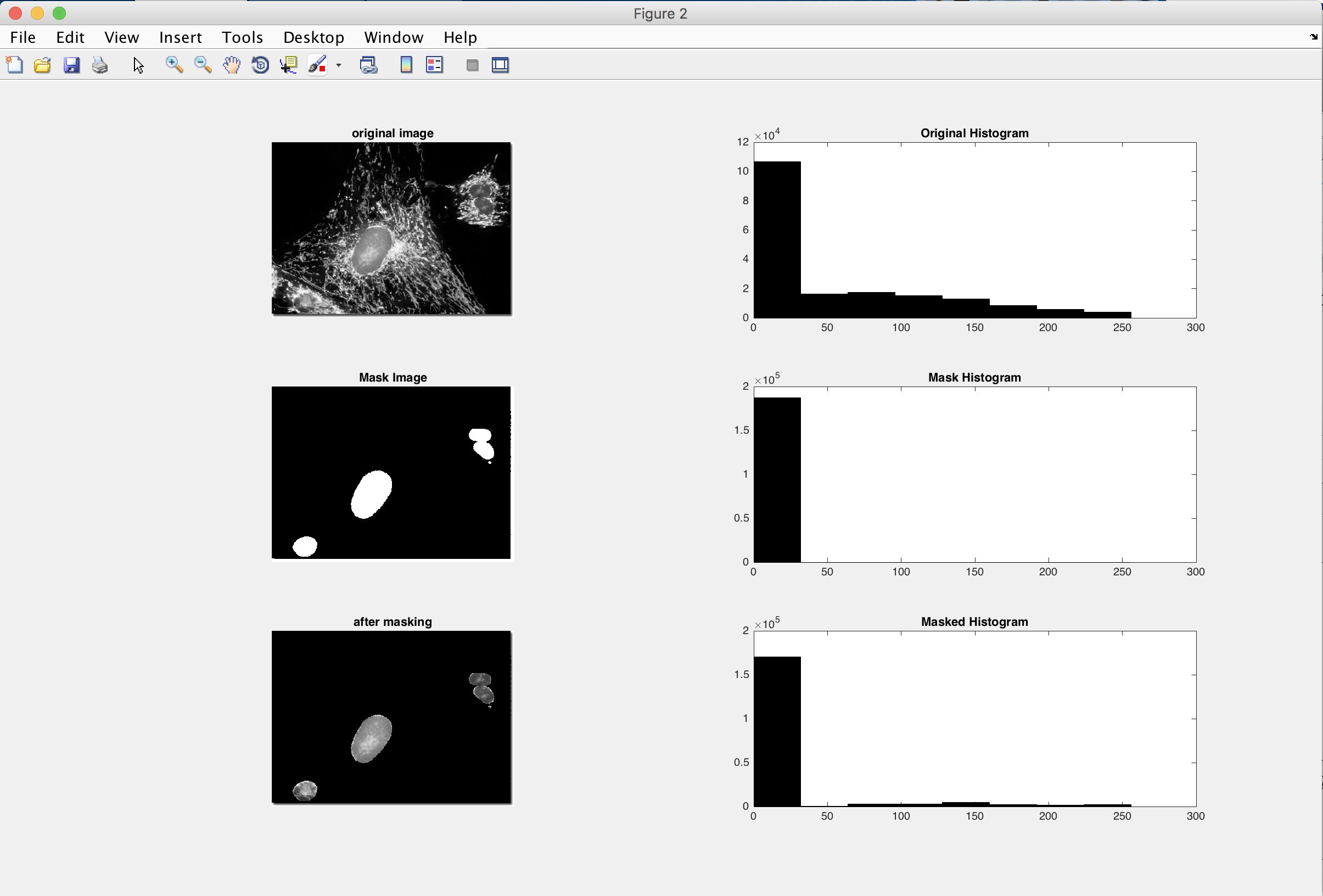
1. Read both original and mask images
2. Convert the type into double
3. Call the myimhistmask function

Myimhistmask: mask operation and histogram representation

1. Do the masking by dot operation between original image and mask. Dot operation is performed since it needs to be done in scalar way.

*Iresult = I.\*mask;*

1. Do a-c in myimhist function (part a). I did not realize, why I did not just call the myimhist function inside the myimhistmask after the masking. This include in my learning process.

The result :

Time report for plotting all of the histogram respectively

Elapsed time is 0.024431 seconds.

Elapsed time is 0.026116 seconds.

Elapsed time is 0.163144 seconds.

Original and mask histogram were plotted with myimhist, whereas the after masking was plotted by myimhistmask. The difference between these two functions is the dot operation performed by myimhistmask for masking the original to mask image.

In this masking process, the binary mask only contains 0 or 1 in pixels. This means that 0 indicate pixels is part of background, and 1 is the pixels part of the object of interest. When this binary mask is masked into original image, by dot operators, the remaining pixels of the original image is the pixels dotted with mask is equal to one. Therefore, the result of masked image giving us the object of interest as it is, and changed everything else as 0 (black) indicates background.

1. Contrast enhancement

Contrast enhancement can be performed in many ways, and the ultimate purpose for contrast enhancement is to get a better or balance contrast on the image. So that, the image can be clearer, or to be used as the desired purpose.

**Part a:**

In part a, we are implementing a minimum and maximum linear contrast stretch. In the representation of histogram, image can have unbalance pixel intensity. The bar is higher in one place and low in other places. The linear contrast stretch has a goal to make this distribution more even by stretching all the pixels based on minimum and maximum pixels value of an image. Therefore, the result is depending on the corresponding input pixel value, and collected information or parameter of the image.

The algorithm runs:

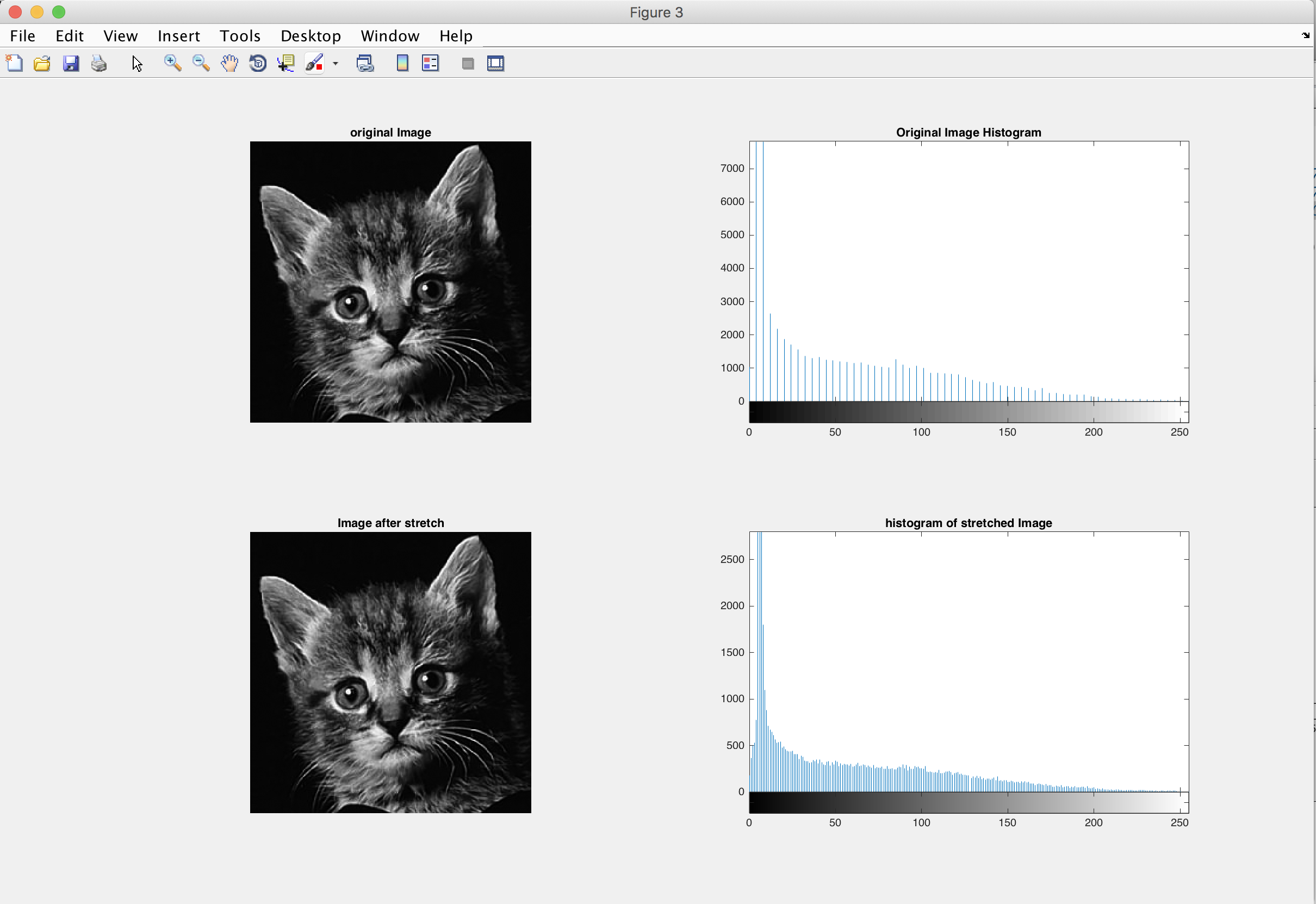
Main: read image and call the function

1. Read grayscale image
2. Call the LCS (linear Contrast stretch) function
3. Display the histogram

LCS function:

1. Convert the uint8 type into double
2. Get the max and min limit of the image pixels
3. Get the gain a, we get a by a =
4. Get bias b by b =
5. Then, get the stretched image Iresult
6. Convert the result into uint8 type

After running the algorithm, the result is provided bellow:



This method does not bring new information at all, instead we may loss information if the stretch is resulting on less than 0 min value, and more than 255 max values. Therefore, and optimum gain is needed to prevent the loss information. From the program, we get the min limit is 0 and max limit is 255. After the calculation performed, gain a is 1.0039 and bias b is 0. The gain and bias is so small. Even though, stretch was performed, there was not a significant change or contrast stretch implemented. From the histogram, we can see that, there is no big difference or big stretch being done. This is because, the min and max limit is too close with the actual pixel range, which are 0 to 255. The amount has been stretch to the allowed range. Therefore, we cannot see the contrast stretch. The solution for this problem is offer in the next part of this problem.

Part b:

In this part, we are using the same image as the previous part, the cat. In this part, we are going to re-calculate of the same calculation as before, but we are going to discard the 1% and 5% of lowest and highest pixels intensity. The purpose is to get rid of those part to get new min and max limit values, so that we have more room for stretching the contrast using previous algorithm. So that, we can actually see the differences.

There are several improvisation on the LCS function, it is now designed as the AdvanceLCS function.

AdvanceLCS:

1. get rid of 1% or 5% or n% of lowest and highest pixel intensity by getting the new max and min limit. So, we basically ignored the first and last n%.
2. get the pixel count and gray level from the image histogram
3. get the cumulative sum of the pixel count

cumulativePC=cumsum(pixelCount);

1. get the cumulative distribution

cumDist = cumulativePC/numel(Img);

1. find the new max and min limit with find function. It is basically find the cumulative distribution which more that %. Get the first index after the cumdist. Take the index and get the value from the grayLevels

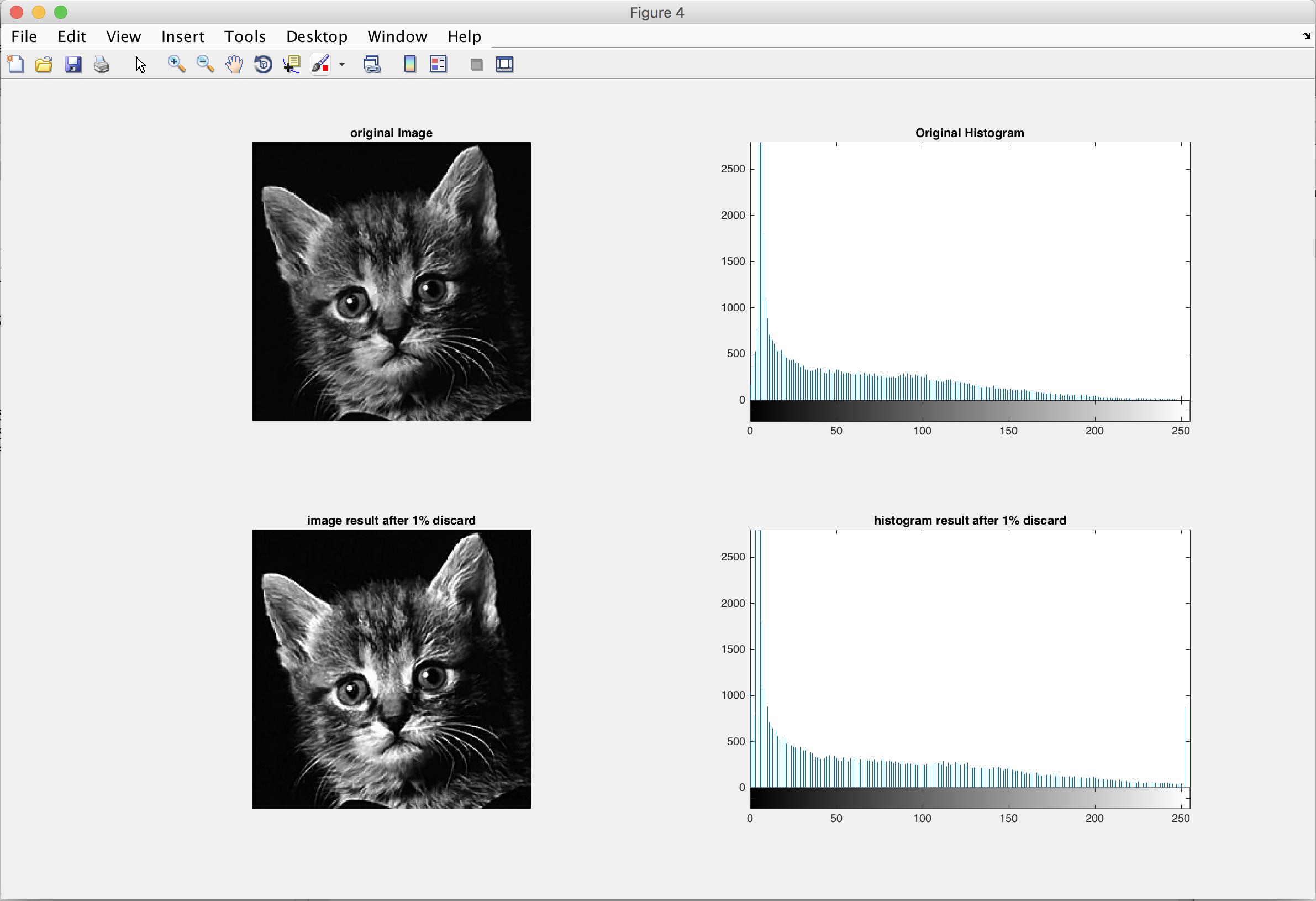
newMinIndex = find(cumDist>percentage, 1, 'first');

newMinLimit = grayLevels(newMinIndex);

1. do LCS step c-f (part a) with the new max and min limit
2. Get the gain a, we get a by a =
3. Get bias b by b =
4. Then, get the stretched image Iresult
5. Convert the result into uint8 type

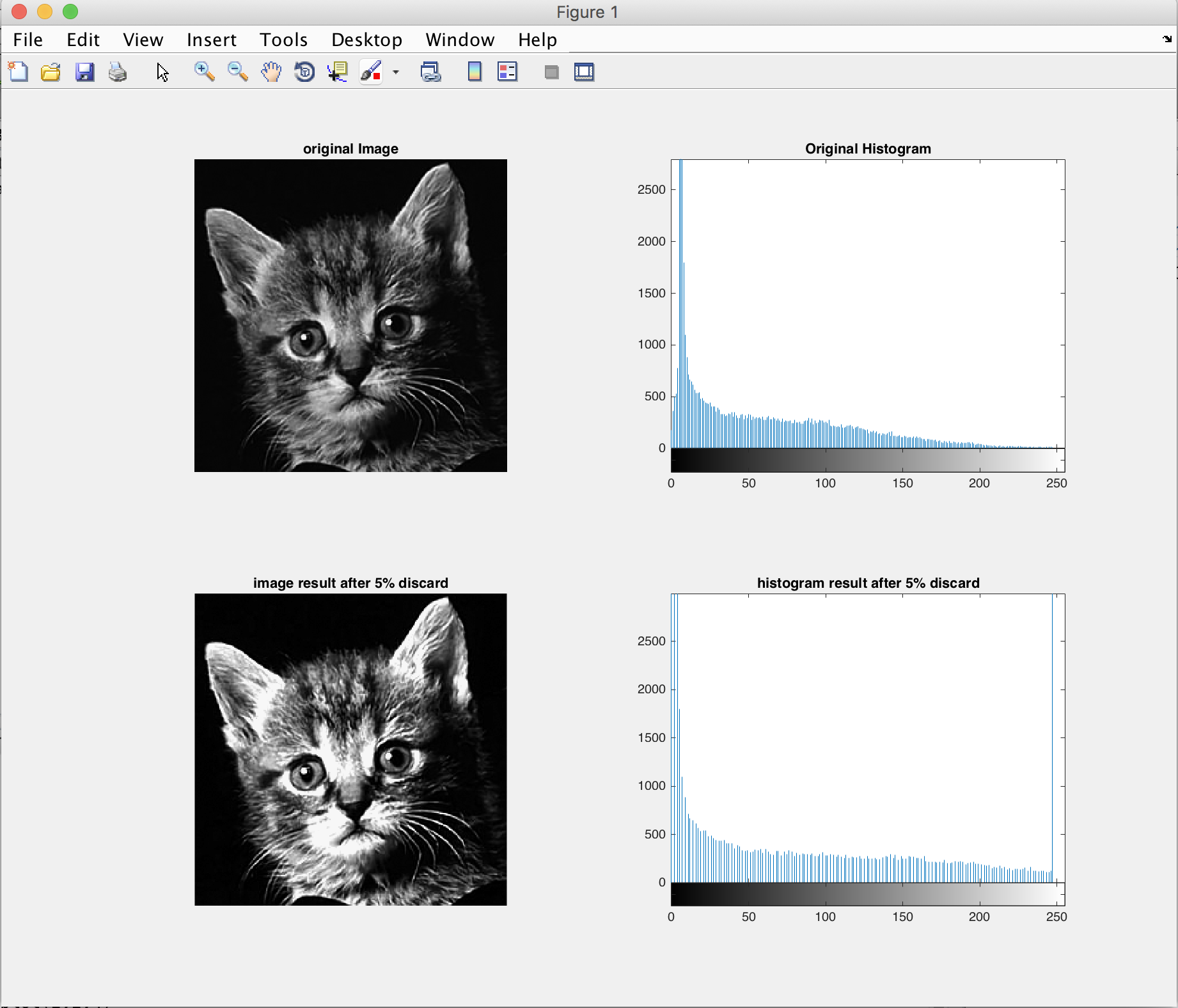
There are 2 trials, which are 1% and 5%. The result is provided bellow

Result after 1% discard



From the image above, we can see that the bottom image (after 1% reduction) has a higher contrast. If we compare this into part a, we can see the difference. In part a, there is no change at all, since their min and max is exactly within the boundaries. However, in this part, we discard 1% lowest and highest pixel, which provide us some more space to stretch in the beginning and the end. In this case, the new min limit is 2 and the new max limit is 204. The calculation gave us gain a = 1.2673 and bias -2.5347. From the histogram, we can see more distribution on the high pixels. It can be seen from the higher bar on that area. Further explanation will be in the next part.

result after 5% discard



The difference, and contrast stretch is clearly explained in this result. We can see that the bottom image (after 5% reduction) has a higher contrast, the image is a lot brighter compare to the original. This happen because discarding 5% lowest and highest pixel gave us a lot more space to stretch the designated pixels. In this case, we get new min is 5 and new max is 157. This range gave us gain 1.6842 and bias -8.4211. From the histogram, we can see more distribution on the high pixels. It can be seen from the higher bar on that area. We can see from the histogram, we discarded pixel 1-4 and 158-255. This space is used to stretch the new range of 5-157. Therefore, we get the better contrast. The contrast will be even higher if we discard more n% from the pixel intensity.