

---

## Table of Contents

1 - Getting Started .....	1
Q1 - Pixel Value .....	1
Q2 - Histograms .....	2
Q3 - Explain differences .....	4
Q4 - Make images brighter .....	4
Q5 - Make images lower contrast .....	5
Q6 - Pixel Wise Transforms .....	6
Q7 - Histogram Equalization napoleon - Histogram .....	10
Q7 - Histogram Equalization napoleon - Images .....	13
Q7 - Histogram, transform and cumulative histogram - Regular Image .....	15
Q7 - Histogram, transform and cumulative histogram - Light .....	17
Q7 - Histogram, transform and cumulative histogram - Dark .....	19
Q8 - Interpolation and low-pass filter .....	21
Q9 - Aliasing .....	22
Q10/Q11 - Mean Image .....	22
Q13 - Geometric Transforms .....	22
Q15 - Scripting and Looping .....	23
Q16 - Histogram Equation .....	24

## 1 - Getting Started

Load the data that we will use.

```
clear;
I = imread('lab1/images/napoleon.png');
I1 = imread('lab1/images/napoleon_light.png');
I2 = imread('lab1/images/napoleon_dark.png');
Z = imread('lab1/images/zebra.png');
Is = single(I);
B1 = imread('lab1/images/brain1.png');
B2 = imread('lab1/images/brain2.png');
B3 = imread('lab1/images/brain3.png');
W = imread('lab1/images/wrench.png');
D = imread('lab1/images/doggo.jpg');
D2 = imresize(D, [128,128], 'nearest', 'antialiasing', true);
GD2 = rgb2gray(D2);
```

## Q1 - Pixel Value

Print out value. We see that it is of type UINT-8 and the value is 89.

```
I(1, 1)           % It prints out 89
```

```
ans =
```

```
uint8
```

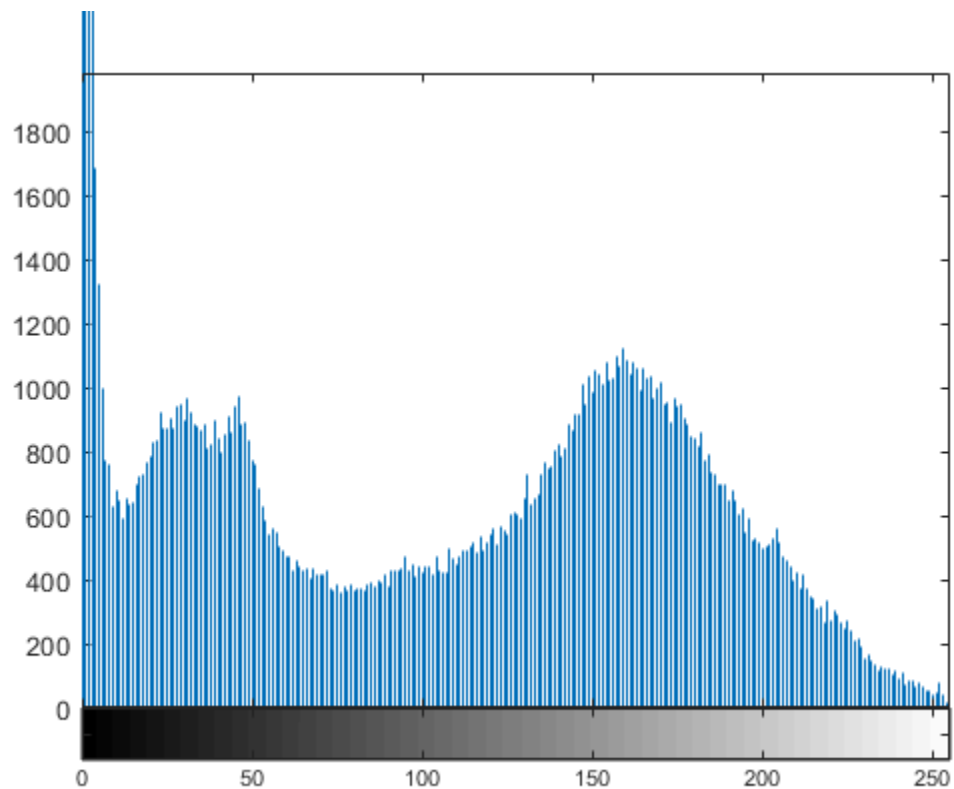
## Q2 - Histograms

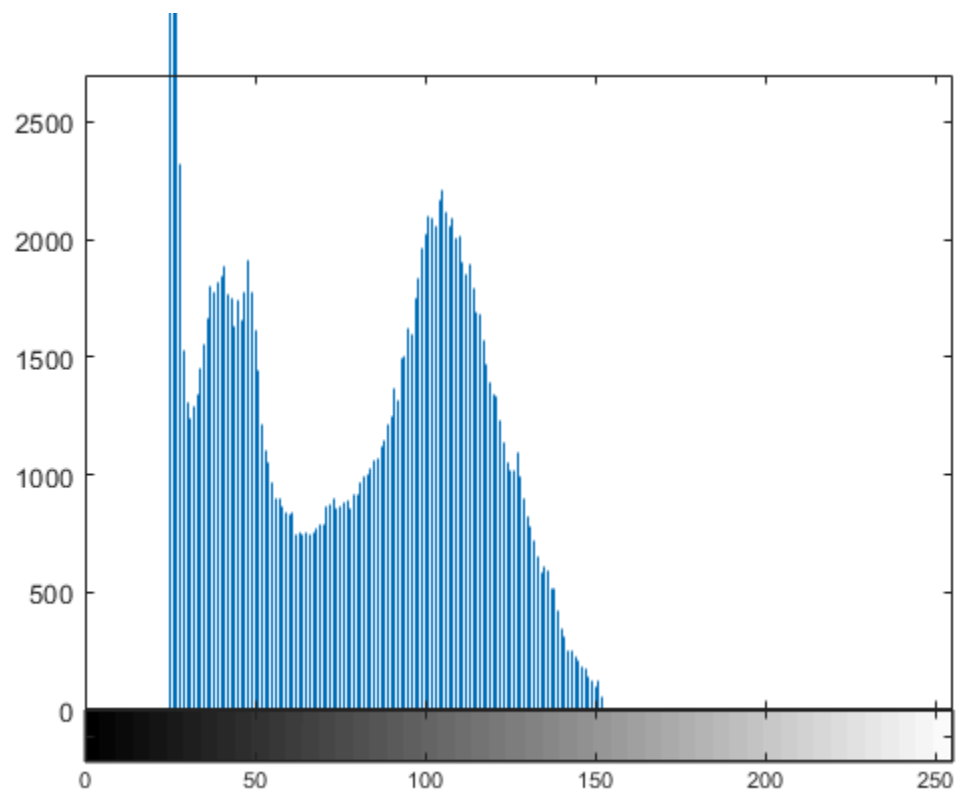
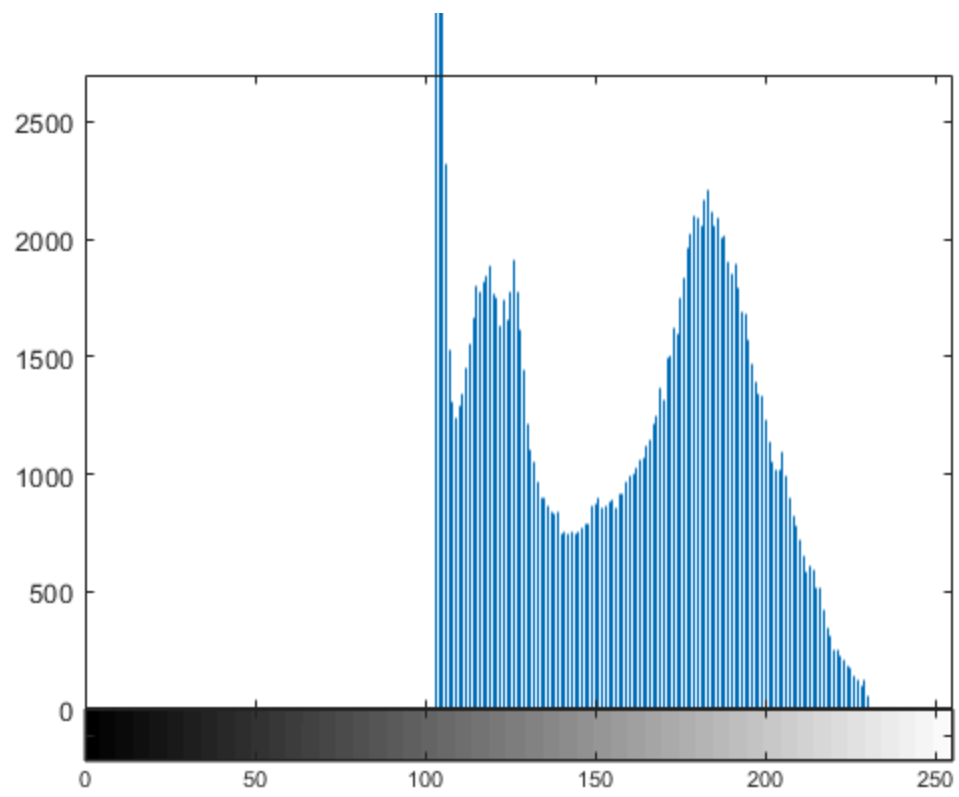
From figure 1, which is the regular image, we see that the pixel values are spread across the domain  $[0, 255]$ . The second figure is skewed towards the lighter pixel values and this image should be the high contrast one. The third histogram shows the third image and how it is skewed in the other direction and is thus the histogram of the low contrast image.

```
figure;  
imhist(I);
```

```
figure;  
imhist(I1);
```

```
figure;  
imhist(Id);
```





---

## Q3 - Explain differences

The problem here is that when `imshow` gets passed a `uint` image it is able to adjust the visualisation range between 0 and 255 but when it gets passed a single precision matrix, which has values from 0 to 255, it tries to plot it as a image with values between 0 and 1. Therefore it needs to be normalised to be plotted correctly. Below we have plotted the difference between the original image and the examples from the assignment. We see that in the `UINT8` one there is considerable difference between the two and this is related to doingh arithmetic on `UINT8` variables.

```
imshow(I - (I/64)*64);  
imshow((I/64)*64);  
  
imshow((Is/255) - (Is/64)*64/255);  
imshow((Is/64)*64/255);
```



## Q4 - Make images brighter

To make images brighter. The take the original image  $f(x, y)$  and apply the trasform  $T = f(x, y) + C$  where  $C > 1$ .

```
imshow(I + 50);  
imshow(I);
```



## Q5 - Make images lower contrast

To make the image lower contrast we apply the transform  $T = f(x, y) \times C$  where  $C < 1$

```
imshow(I);  
imshow(I * 0.5);
```



## Q6 - Pixel Wise Transforms

Below we have performed a gamma transform which is  $T = C \times f(x, y)^\gamma$  where  $C = 1$  and  $\gamma \in \{0.5, 2\}$ . We also compare this with the original image.

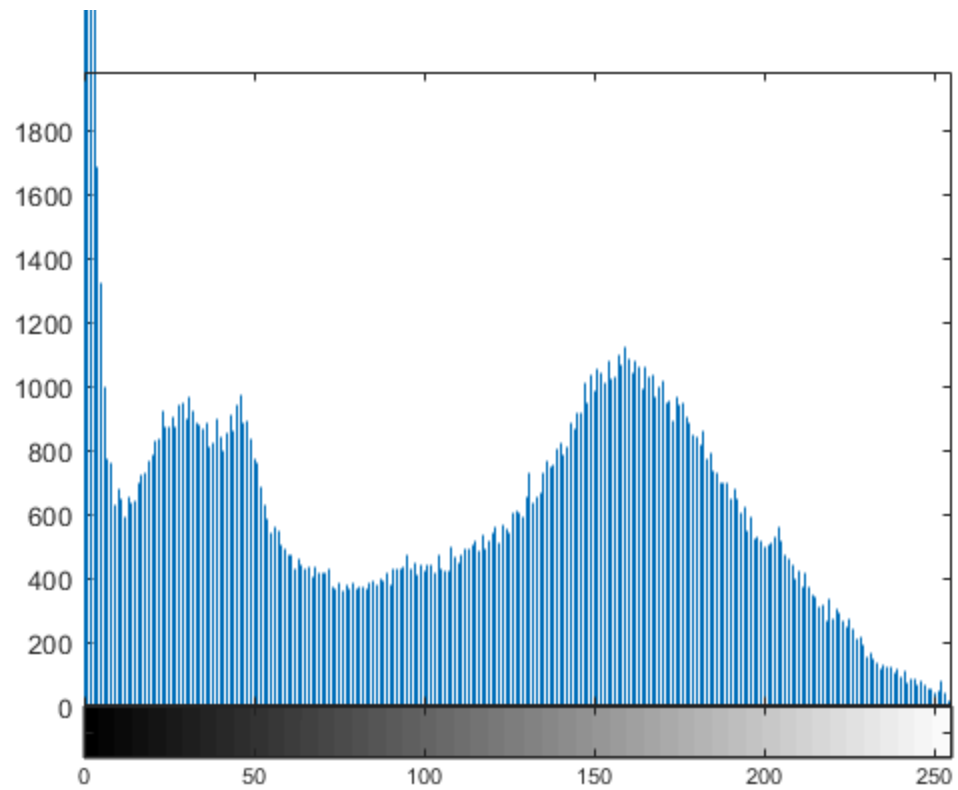
Error updating Text.

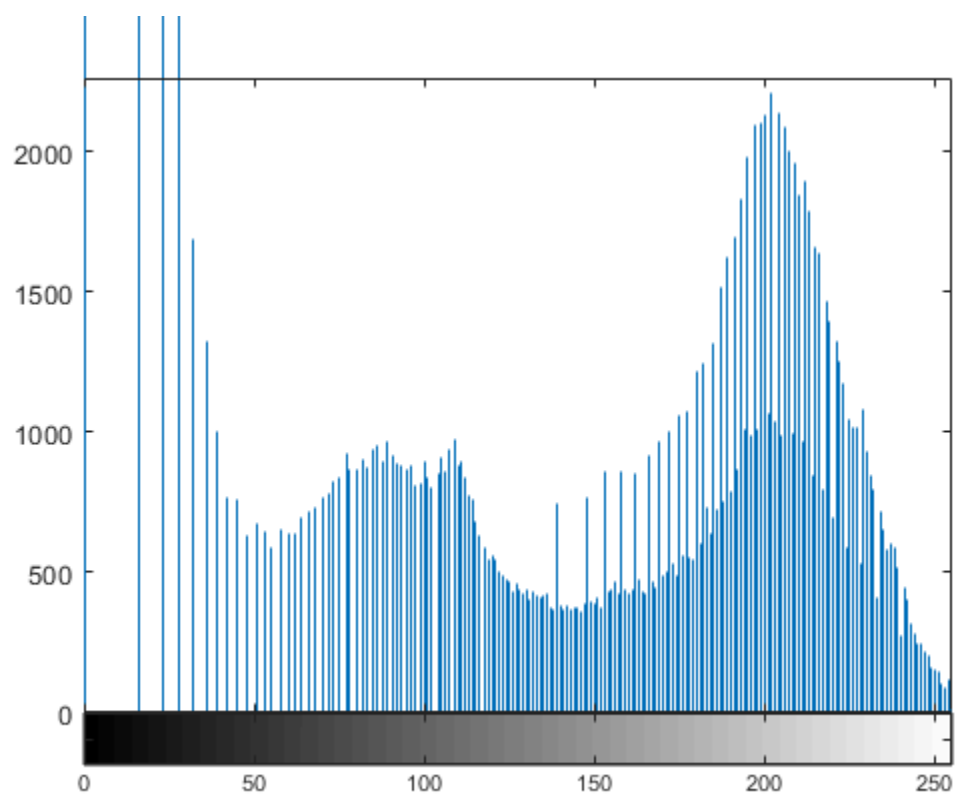
String scalar or character vector must have valid interpreter syntax:  
 $\gamma \in \{0.5, 2\}$

```
figure;  
imhist(I);  
  
figure;  
imshow(I);  
  
g = 0.5;  
L = double(I).^g;  
out = uint8(L .* (255/max(max(L))));  
  
figure;  
imhist(out);  
  
figure;  
imshow(out);
```

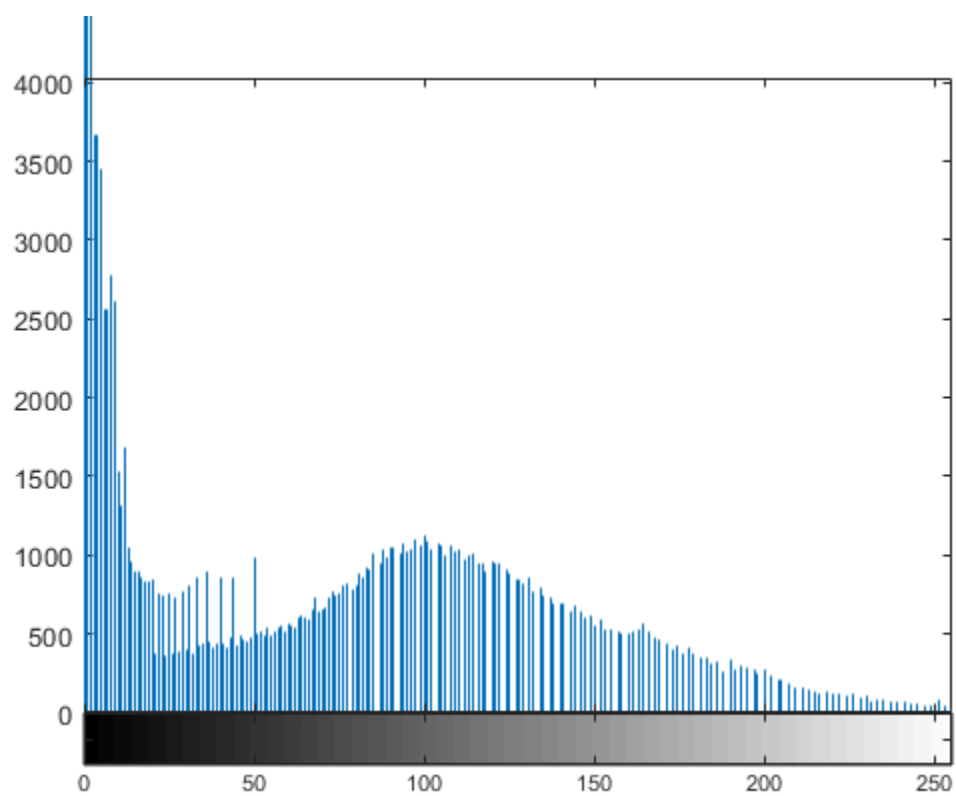
---

```
g = 2;  
L = double(I).^g;  
out = uint8(L .* (255/max(max(L))));  
  
figure;  
imhist(out);  
  
figure;  
imshow(out);
```











## Q7 - Histogram Equalization napoleon - Histogram

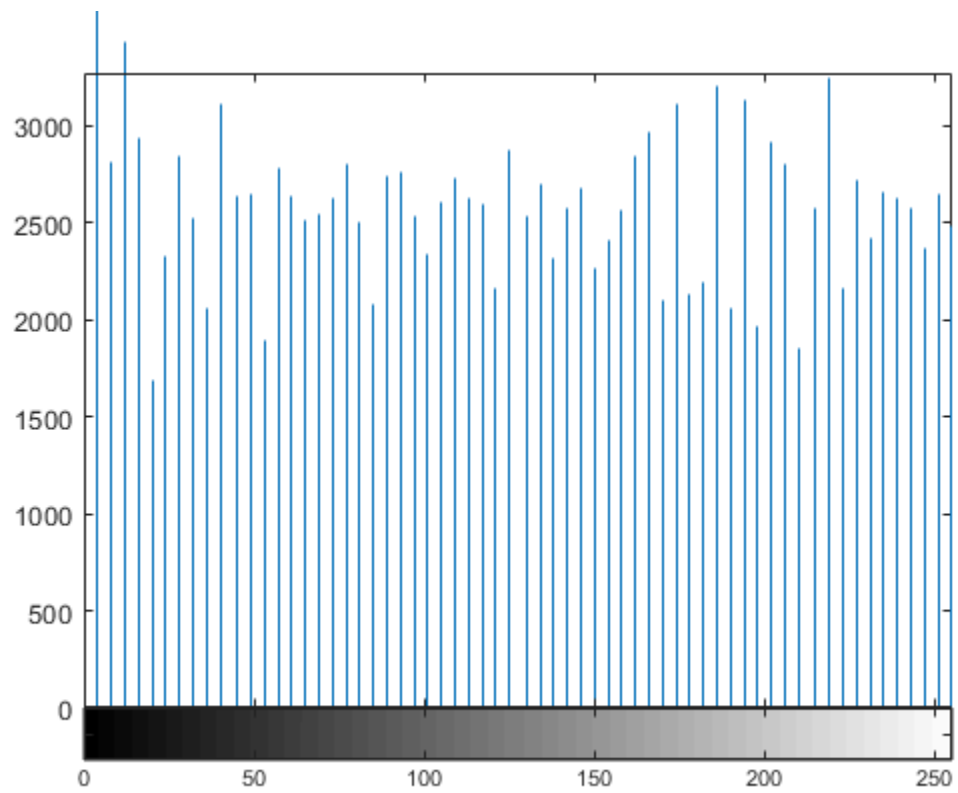
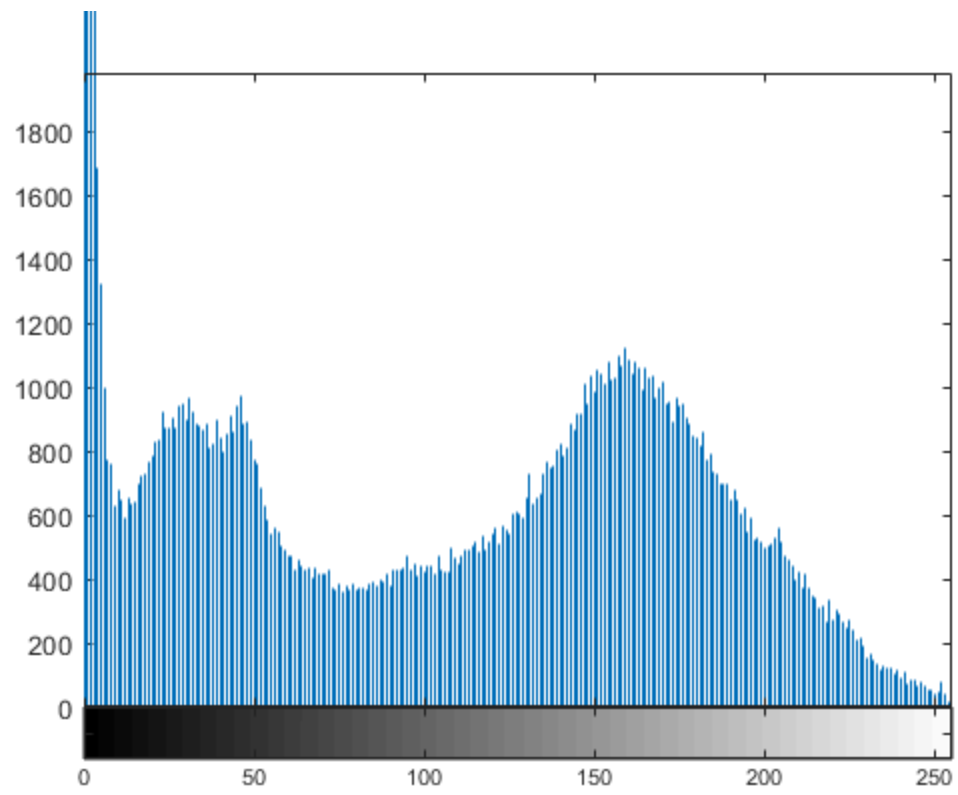
Below is the histograms of the original image, and the original, light and dark image and their equalized histograms.

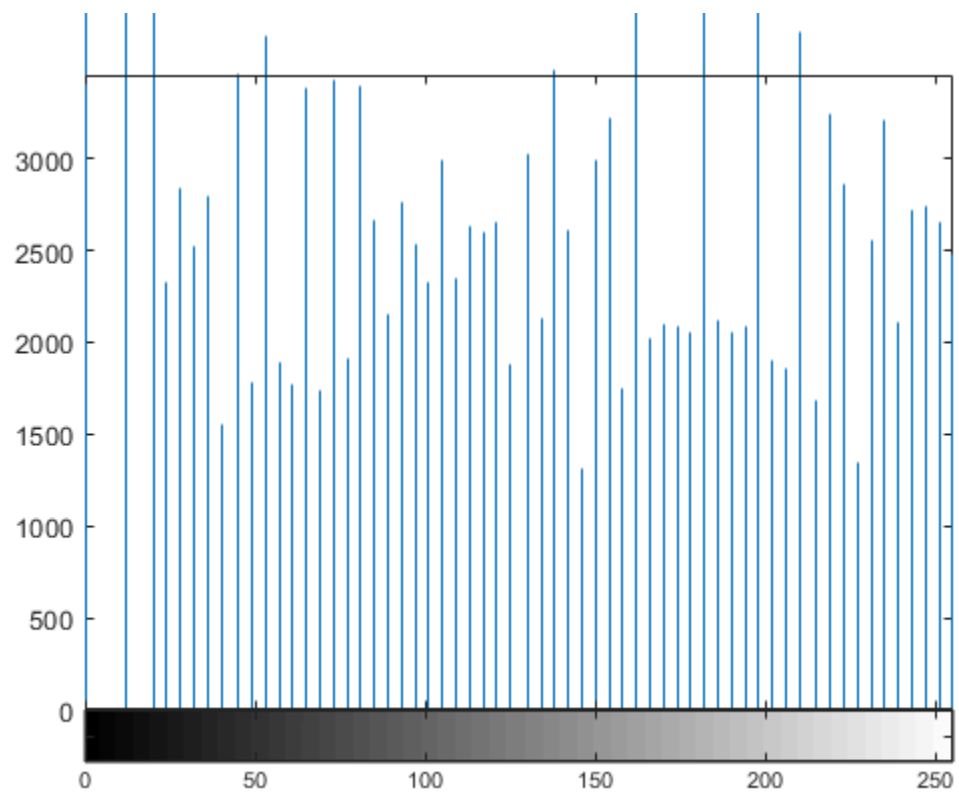
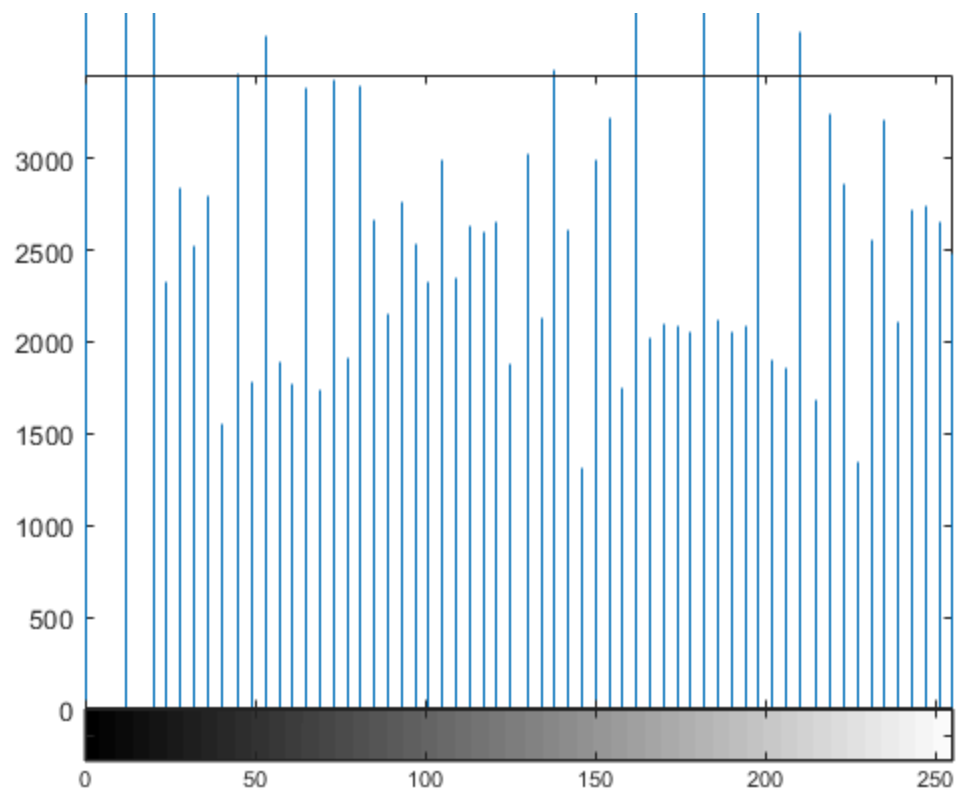
```
figure;  
imhist(I);
```

```
figure;  
imhist(histeq(I));
```

```
figure;  
imhist(histeq(I1));
```

```
figure;  
imhist(histeq(I2));
```





---

## Q7 - Histogram Equalization napoleon - Images

Here we show the equalised images.

```
figure;  
imshow(I);
```

```
figure;  
imshow(histeq(I));
```

```
figure;  
imshow(histeq(I1));
```

```
figure;  
imshow(histeq(Id));
```







## Q7 - Histogram, transform and cumulative histogram - Regular Image

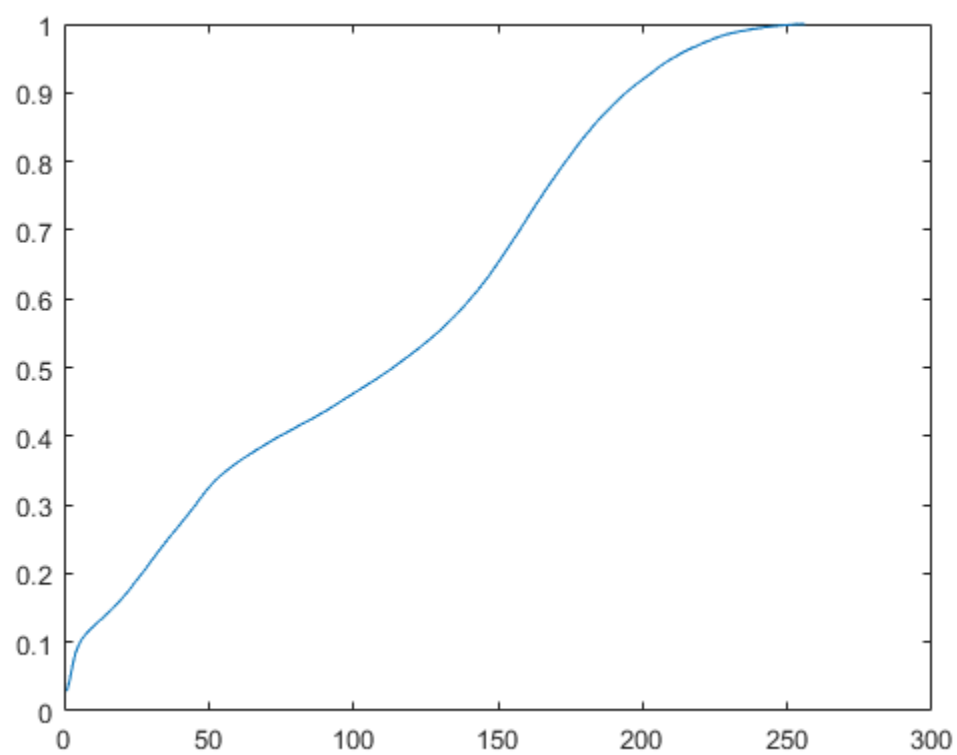
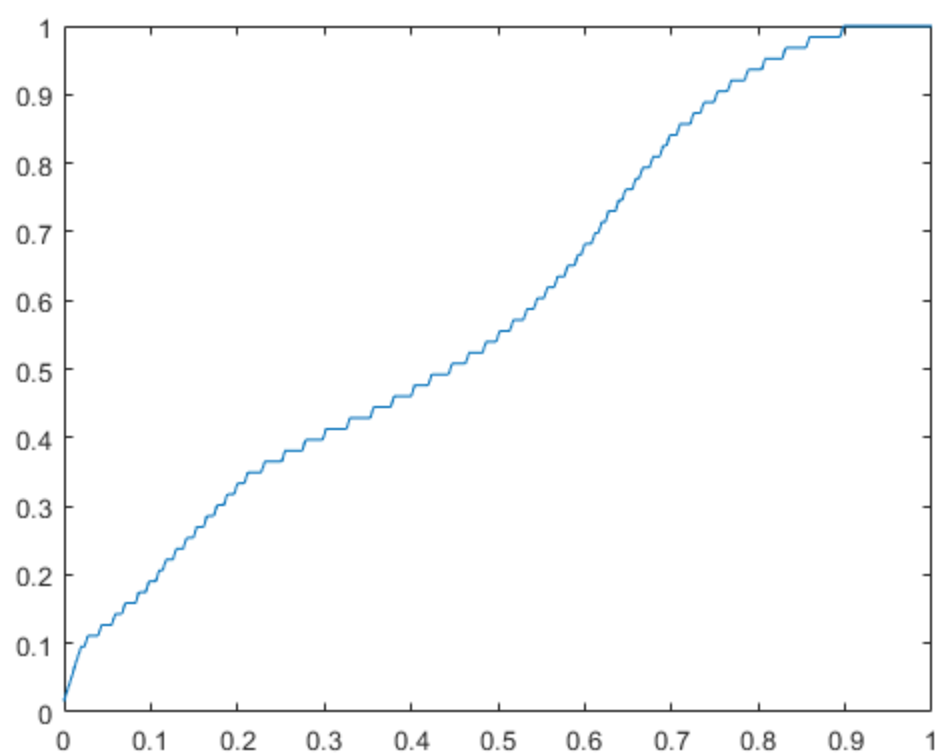
And here is the cumulative histograms. According to the textbook, to equalize a image, you have to take the CDF of the image and use that to transform the pixel values. This is estimated by calculating the cumulative histogram and using that as an approximation for the CDF. If we use the `histeq` function and compare it to the normalied histogram calculated manually, we see that these are the same.

```
[J,T] = histeq(I);
```

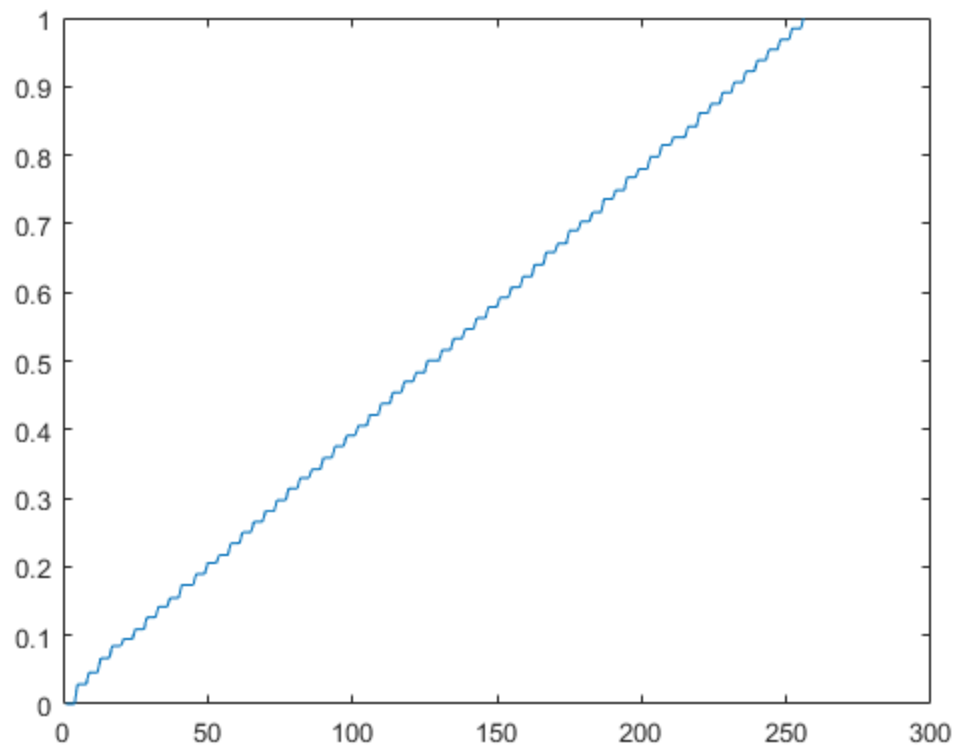
```
figure  
plot((0:255)/255,T);
```

```
figure  
plot(cumsum(imhist(I)) / sum(imhist(I)));
```

```
figure  
plot(cumsum(imhist(J)) / sum(imhist(J)));
```







## Q7 - Histogram, transform and cumulative histogram - Light

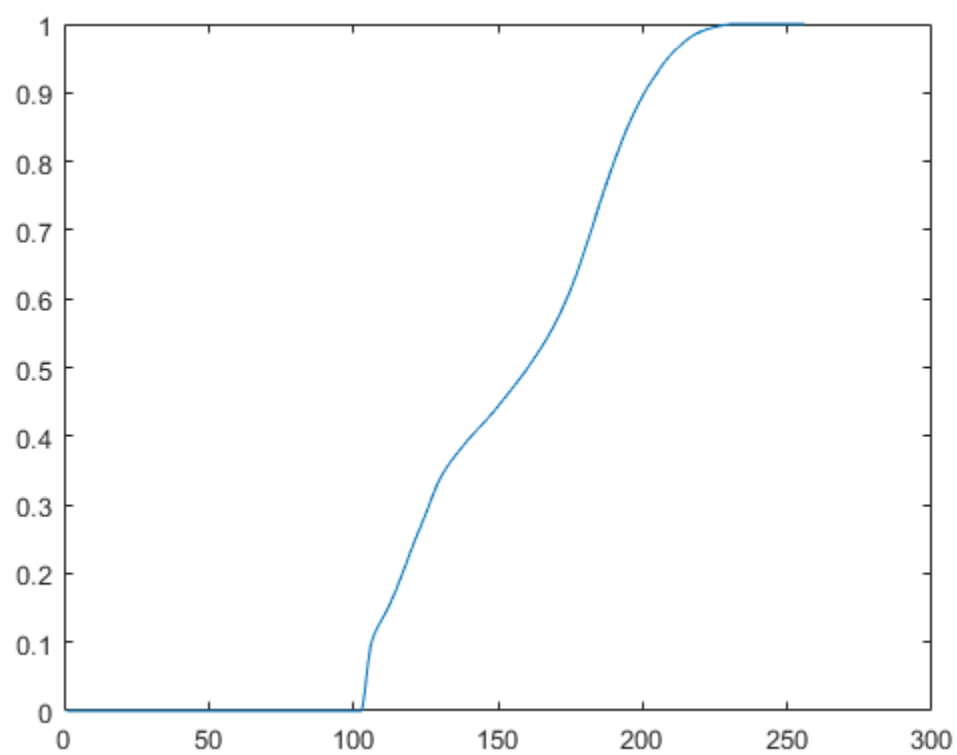
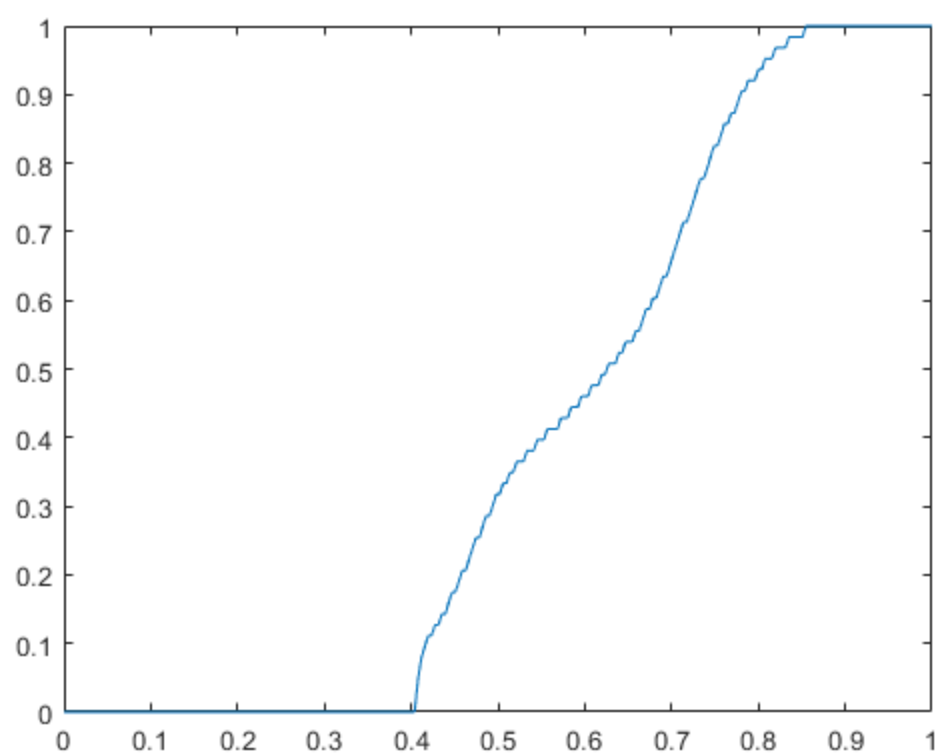
Same as above but for the light image.

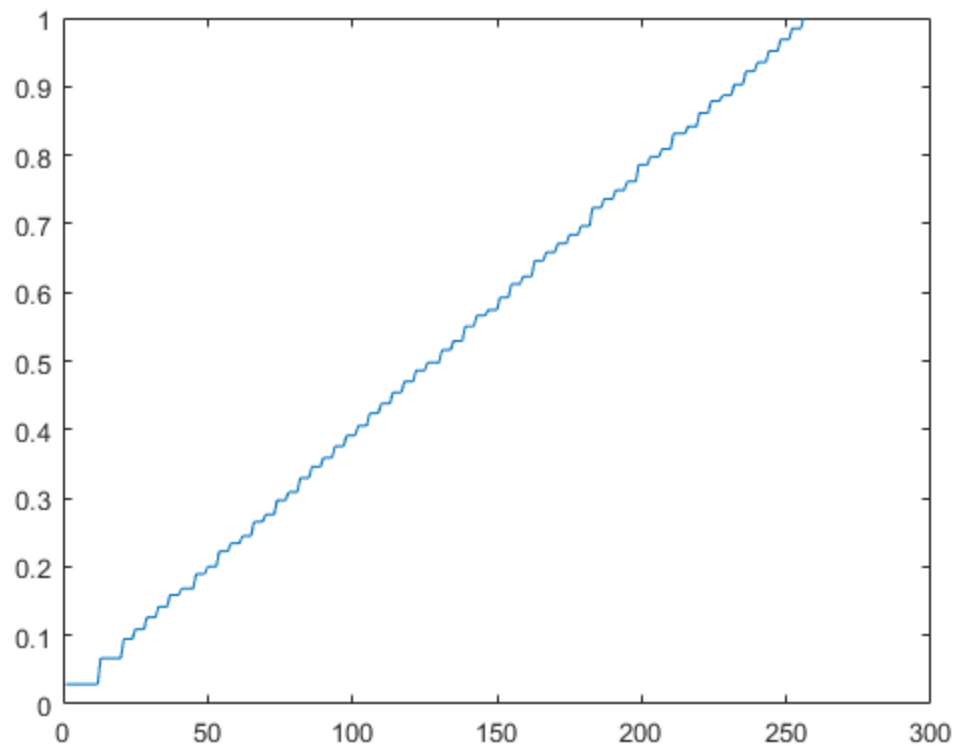
```
[J,T] = histeq(I1);
```

```
figure  
plot((0:255)/255,T);
```

```
figure  
plot(cumsum(imhist(I1)) / sum(imhist(I1)));
```

```
figure  
plot(cumsum(imhist(J)) / sum(imhist(J)));
```





## Q7 - Histogram, transform and cumulative histogram - Dark

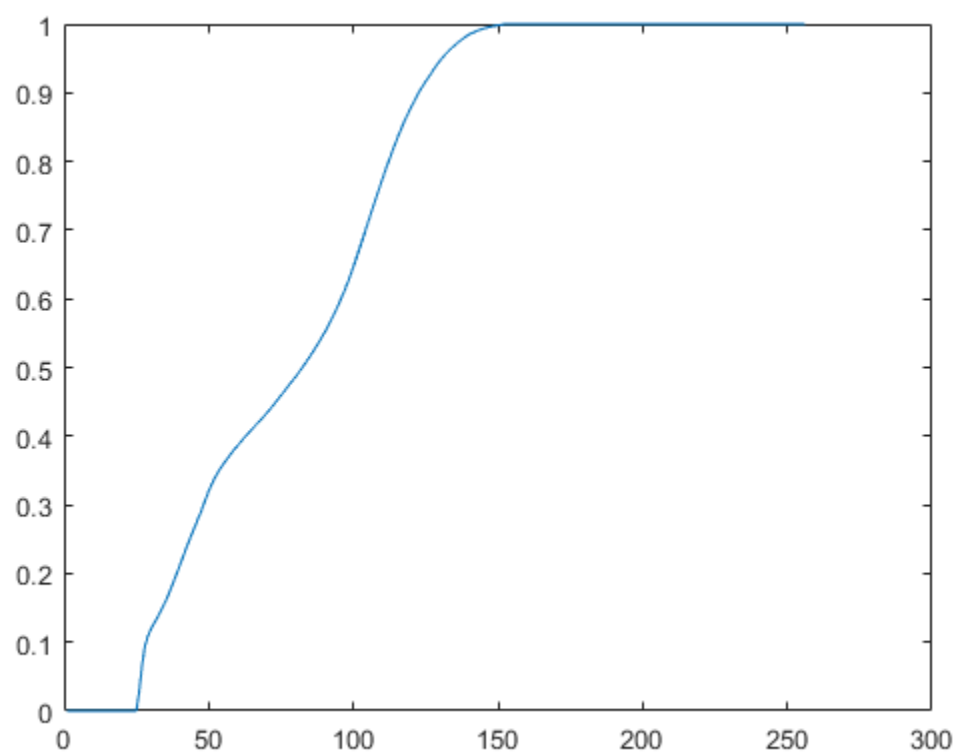
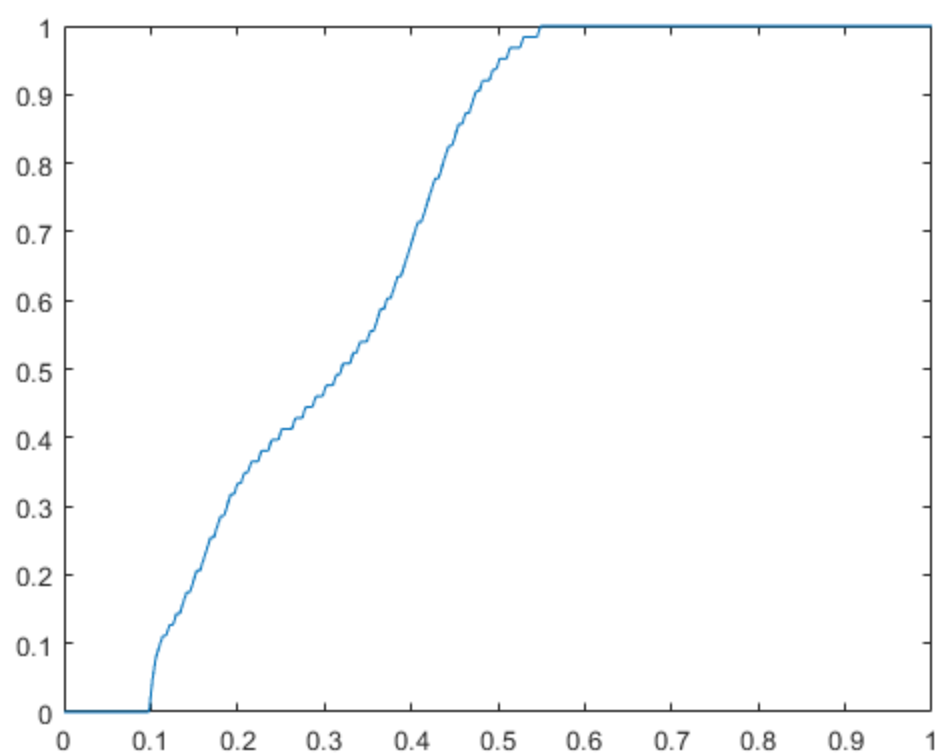
Same as above but for the dark image.

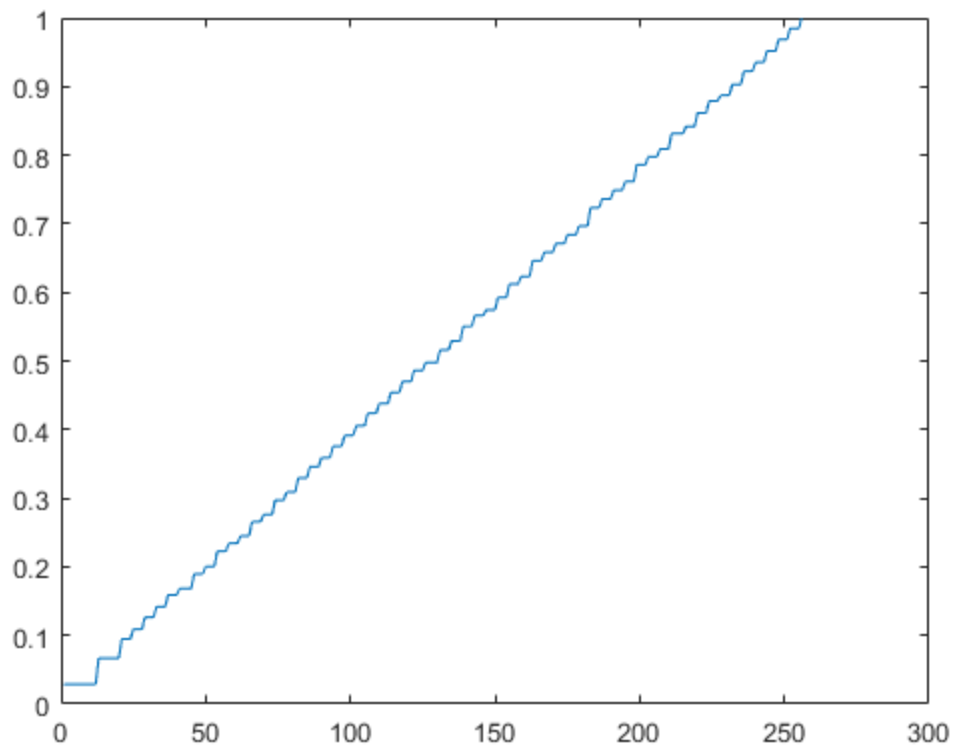
```
[J,T] = histeq(Id);
```

```
figure  
plot((0:255)/255,T);
```

```
figure  
plot(cumsum(imhist(Id)) / sum(imhist(Id)));
```

```
figure  
plot(cumsum(imhist(J)) / sum(imhist(J)));
```





## Q8 - Interpolation and low-pass filter

The difference between the interpolation methods are not very significant but the difference between performing a low pass filter is quite significant. The background of the image without the use of a low pass filter is quite noisy whereas with antialiasing the background is more homogenous and resembles the original image better.

```
Jnf = imresize(I, [78,78], 'nearest', 'antialiasing', false);
Jnt = imresize(I, [78,78], 'nearest', 'antialiasing', true);
Jbf = imresize(I, [78,78], 'bilinear', 'antialiasing', false);
Jbt = imresize(I, [78,78], 'bilinear', 'antialiasing', true);

imshow(Jnf);
imshow(Jnt);
imshow(Jbf);
imshow(Jbt);
```

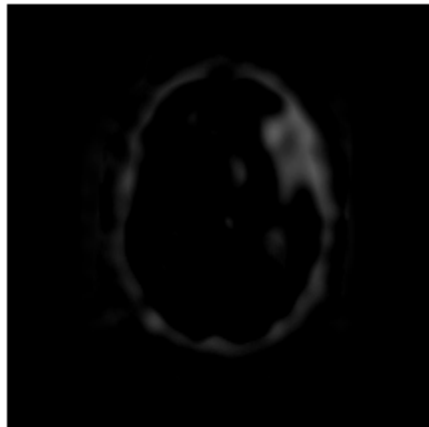


## Q9 - Aliasing

Aliasing is a general problem when sampling a signal and when several signals is indistinguishable when sampled. This could be for example when trying to sample an audiosignal and interpolating it and different possible frequencies are available.

## Q10/Q11 - Mean Image

```
Mn = (B1 + B2)/2;  
imshow(Mn - B3);
```



## Q13 - Geometric Transforms

```
J = imrotate(W, 20);  
K = imrotate(W, 20, 'bilinear');  
  
imshow(J);  
imshow(K);
```



## Q15 - Scripting and Looping

```
newI = zeros(130,130,'uint8');
GD2t = padarray(GD2, [2, 2], 0);

for i = 3:130
    for j = 3:130
        sum = uint16(0);
        for k = 1:5
            for l = 1:5
                sum = sum + uint16(GD2t(i+k-3, j+l-3));
            end
        end
        newI(i,j) = uint8(sum/25);
    end
end

Image = newI(3:130, 3:130);

imshow(GD2);
imshow(Image);
imshow(Image-GD2);
```



## Q16 - Histogram Equation

```
f = myhist(GD2);

% Show Original Image
figure;
imshow(GD2);

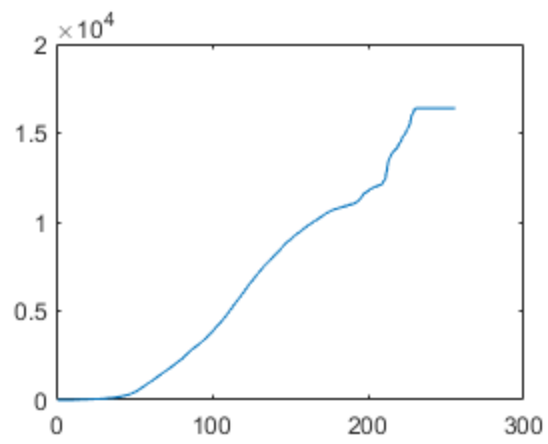
% Show Equalised Image using myhist
figure;
imshow(f);

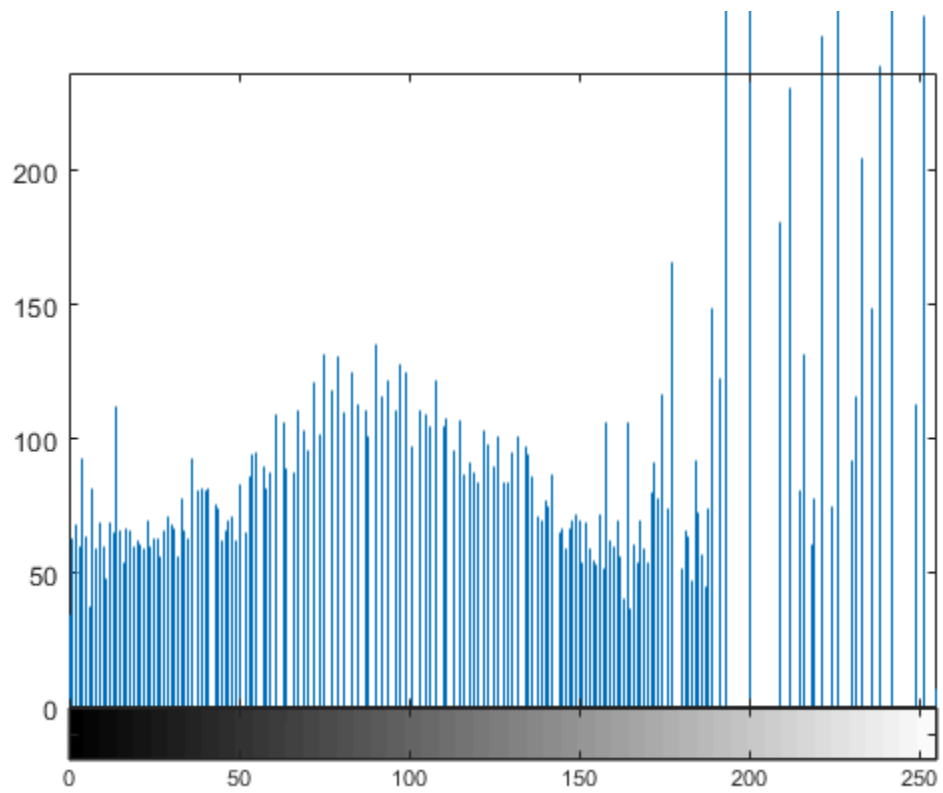
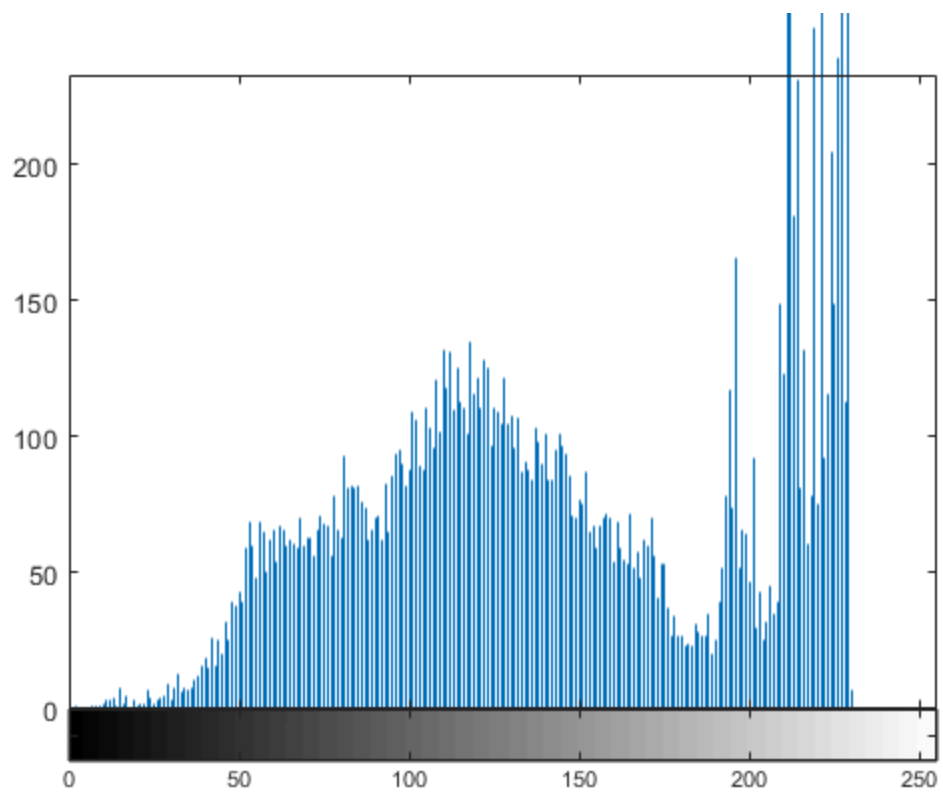
% Show Equalized Image histogram using myhist
figure;
imhist(GD2);

% Show Equalized Image histogram using myhist
figure;
imhist(f);

function f = myhist(image)
    Ihist = hist(reshape(image.',1,[]), 0:255);
    T = cumsum(Ihist);
    norm = T(256);
    plot(T);
    Tn = uint8(T*255 / norm);
    f = Tn(image);
end
```







---

*Published with MATLAB® R2021a*