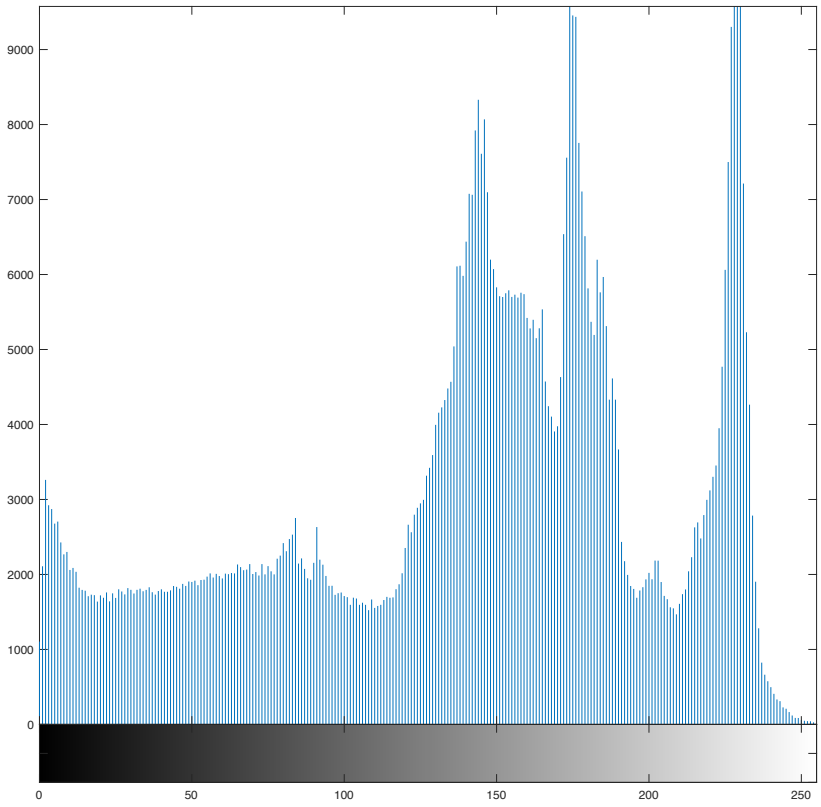
Histogram Analysis and Intensity Processing

A useful tool to start analysing images and processing them is to use a Histogram of the image pixel intensities, that is, obtain a histogram of how many times each pixel Intensity is present in the image i.e. how many pixels have an intensity of 0 (Black) and how many pixels have an Intensity of 1, and so on upto 255 (White). Matlab has a convenient function to show us this using the command:

>> imhist(Pic)

resulting in:



Notice a lot of the image (Pic) is made up of 'light-gray' intensities with almost no pure white in this case.

We can adjust the intensity profile using a number of commands in Matlab, one of which is *imadjust*. This function takes a normalised representation (between 0 and 1) of the intensities in the image range (0 to 255) and 'stretches' them over a given range.

```
PicIntensity = imadjust(Pic,[low_in high_in],[low_out high_out])
```
maps intensity values in `I` to new values in `J` such that values between `low_in` and `high_in` map to values between `low_out` and `high_out`.

For instance:

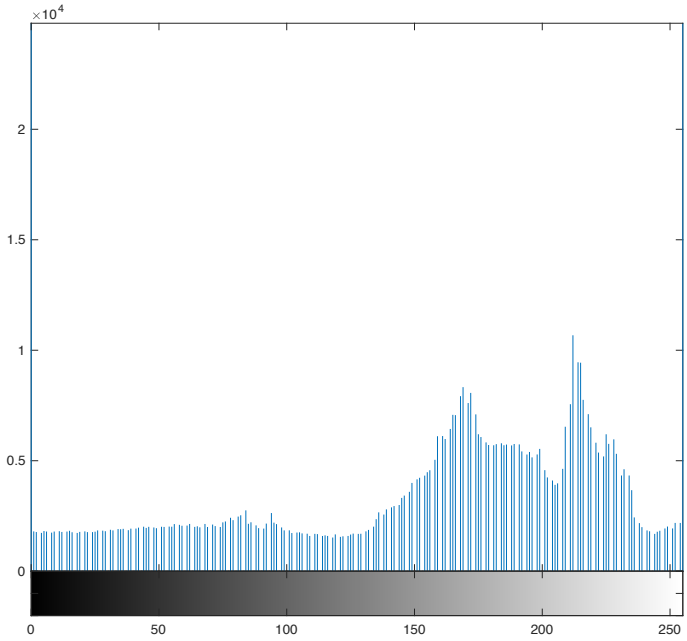
```
PicIntensity=imadjust(Pic,[0.2 0.8],[0 1]);
```

Here, the function normalises the grayscale range from 0 to 1 (i.e. 0 -> 0 and 255 ->1) and then expands the grayscale interval between 0.2 and 0.8 to the full range 0 to 1.

```
PicIntensity=imadjust(Pic,[0.2 0.8],[0 1]);
figure;
imshow(PicIntensity);
title('PicIntensity=imadjust(Pic,[0.2 0.8],[0 1]);');
```

You can look at the resulting histogram again using:
>>imhist(PicIntensity)

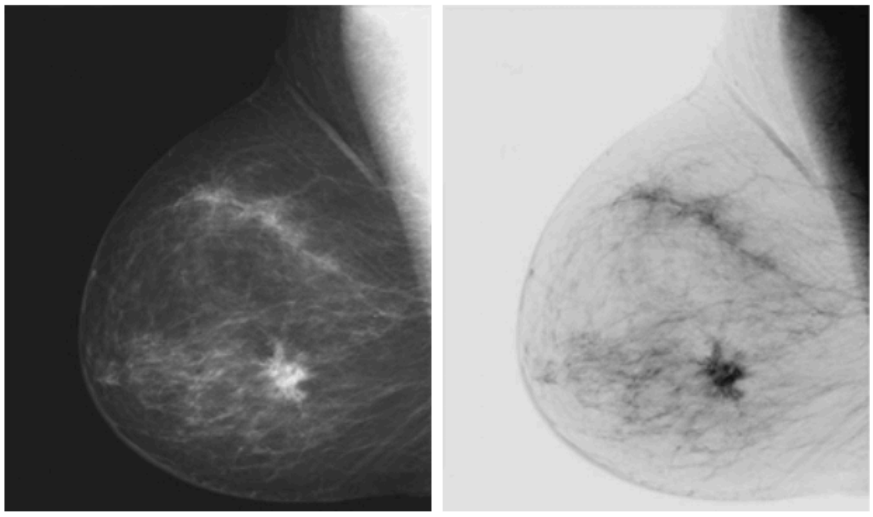
Notice how the histogram has been stretched. This is often useful when highlighting a particular intensity band of interest.



The imadjust function is also useful when the 'negative' of an image is required.

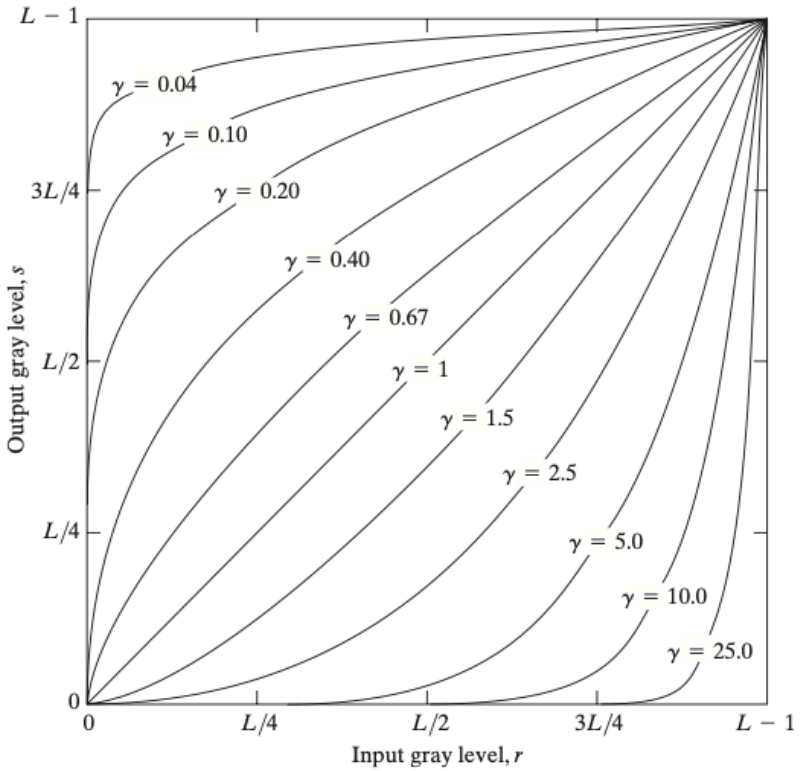
```
PicIntensity=imadjust(Pic,[0 1],[1 0]);
figure;
imshow(PicIntensity);
```

This often occurs in medical applications when it can be easier to identify lesions in a negative image.



Mammogram and its negative image

More generally, we can define a function to map the range of input Intensities to desired output Intensities in a nonlinear manner. Many methods can be used for this, but a common choice is termed 'gamma correction', which is essentially a power-law mapping between input to output intensities. In the figure below, *L* is the number of Intensity levels—typically 256 (0 to 255) in our case. Gamma correction allows us to expand gray levels for instance when images are quite dark.




```
PicIntensity=imadjust(Pic,[0 1],[0 1], gamma);
```

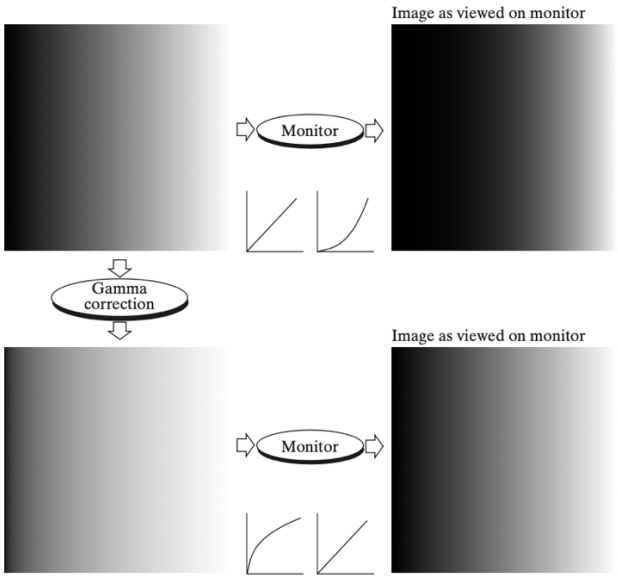
Examples

```
PicIntensity=imadjust(Pic,[0 1],[0 1], 2);
```

```
imshow(PicIntensity);

or

PicIntensity=imadjust(Pic,[0 1],[0 1], 0.5);
imshow(PicIntensity);
```

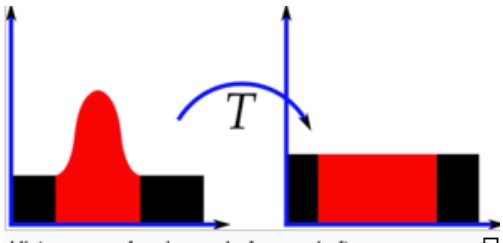
gamma= 2 gamma=0.5

Gamma correction is sometimes used to pre-process images before displaying or printing eg. on a CRT monitor which has an intensity-to-voltage response that is a power function (eg. 2.5 requiring a gamma of 1/2.5=0.4) that makes the image darker than expected; or a printer/scanner.

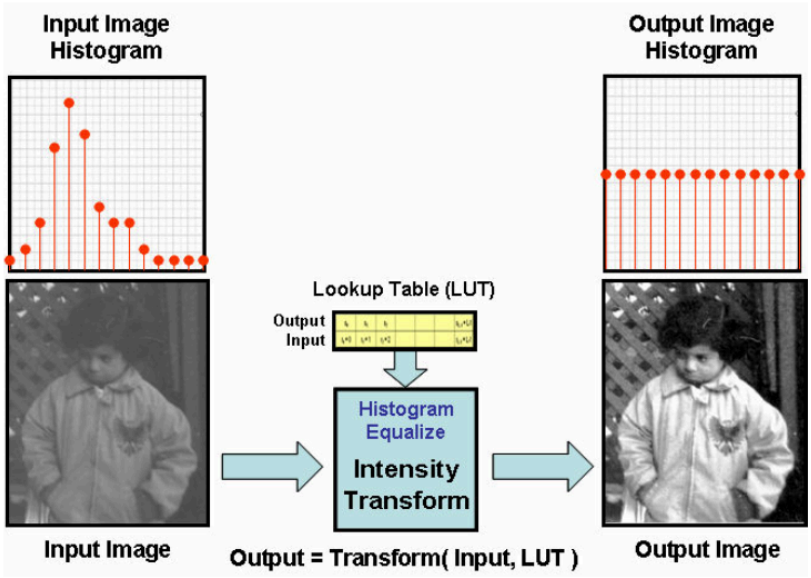


Histogram Equalisation

The human eye tends to be more sensitive to contrast rather than absolute pixel intensity, and this means that we perceive less information from images with poor intensity distributions. To address this Histogram Equalisation is a common technique to redistribute the histogram intensity profile to more evenly spread the intensities of the image over the full range of levels. From a medical imaging perspective, this can typically provide better bone structure images in X-rays. More generally, it allows areas of an image with low contrast to gain a higher contrast. The process of histogram equalisation is essentially the development of a look-up take that maps input to output intensities based on the probability density function of the image intensities.



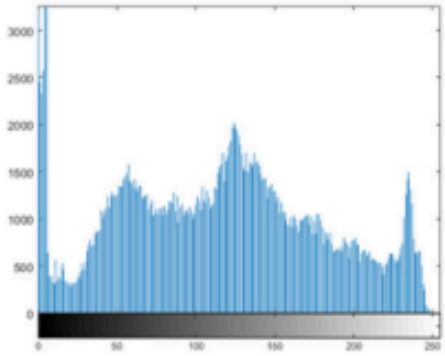
Histograms of an image before and after equalization.



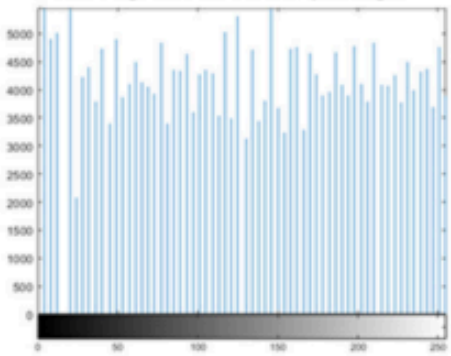
Original Image

Histeq Image

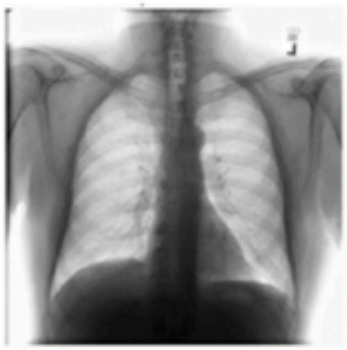
Histogram of Original Image



Histogram of Histeq Image

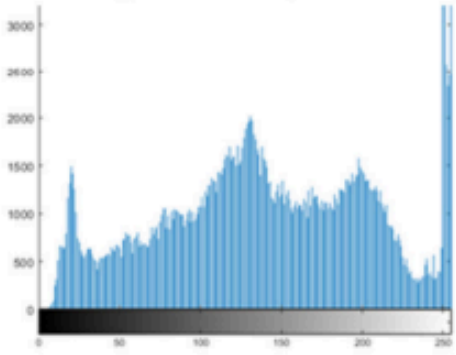


Complement Image

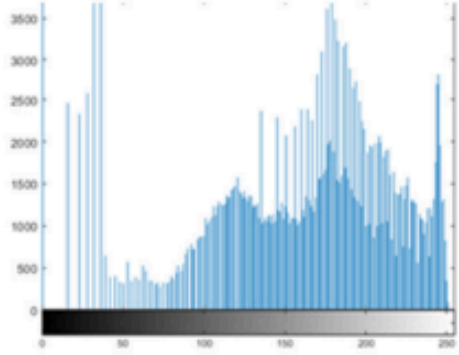


Gamma Image

Histogram of Complement Image



Histogram of Gamma Image



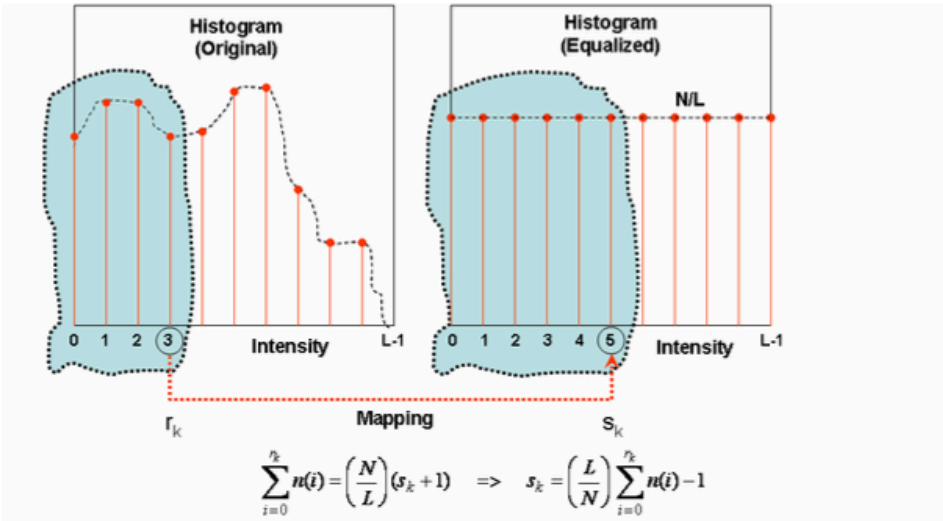
Method

Begin with the smallest intensity level (0), and proceed through to the highest possible intensity level (L-1)—typically 255 in our case. Note that the process must preserve the number of pixels having any mapped intensity level in order to maintain the original image content, and be equal to N/L where N is the total number of pixels and L is the number of intensity levels. Let r_k (r_k=k; k=0,1,2,...,L-1) be an original intensity "k", and s_k be the mapped intensity corresponding to this original intensity "k" (i.e., r_k) for histogram equalization. Up to any intensity r_k (r_k=k; k=0,1,2, ..., L-1) in the original image, the total number of image pixels with intensities up to r_k (cumulative histogram) should be the same as the total number of image pixels up to s_k in the mapped intensities to maintain the image.

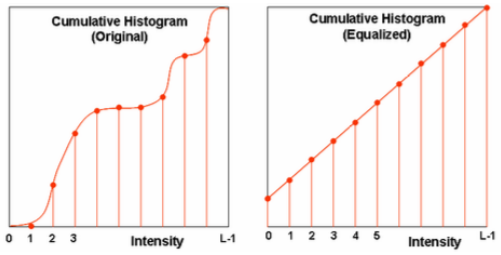
As mentioned, for histogram equalization, the total number of pixels at any mapped image intensity s_k (k=0,1,2,...,L-1) is the same, and equal to "N/L". Thus we can write (in terms of the number of image pixels at each intensity level):

$$\sum_{i=0}^{r_k} n_R(i) = \left(\frac{N}{L}\right)(s_k+1) \quad : \quad k=0,1,2,3,...,L\text{-}1;\ r_k = k$$

(Note: "k" begins at zero, hence the term s_k+1 in the above equation).



where *n(i)* is the number of image pixels in the (original) image with intensity "i" . The citation below is where this treatment originated and it gives a more detailed information. However, the easiest way to understand the procedure is by example.



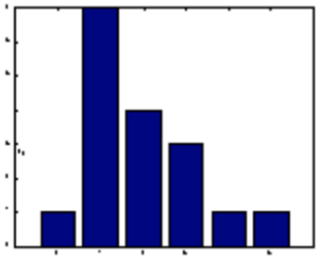
AC 2009-1086: SIMPLIFIED TEACHING AND UNDERSTANDING OF HISTOGRAM EQUALIZATION IN DIGITAL IMAGE PROCESSING, Shanmugalingam Easwaran, Pacific Lutheran University

Example 1

Consider a simple input image and associated histogram (N=16, L=5)

4	1	3	2
3	1	1	1
0	1	5	2
1	1	2	2

input image



Process:

1. Compute the histogram of the image
2. Calculate the normalized sum of the histogram
3. Transform the input image to an output image ('round' the values to give integers)

Calculate the cumulative sum of the pixel intensities:

intensity	sum	normalized sum
0	1	1/16*5=0.31255
1	8	2.5
2	12	3.75
3	14	4.375
4	15	4.6875
5	16	5.0

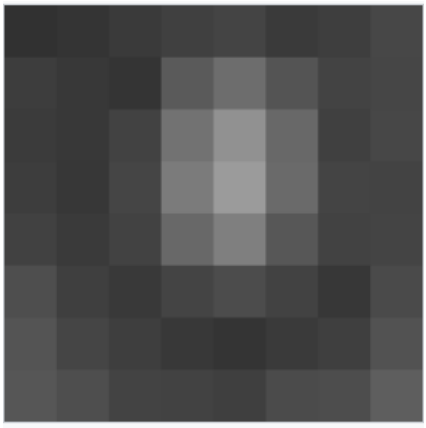
5	3	4	4
4	3	3	3
1	3	5	4
3	3	4	4

output image

The next example gives a more generic mathematical treatment of the process.

Example 2 (https://en.wikipedia.org/wiki/Histogram_equalization)

Consider an 8x8 image with 8-bit intensity levels:



The 8×8 sub-image shown in 8-bit grayscale

52	55	61	59	79	61	76	61
62	59	55	104	94	85	59	71
63	65	66	113	144	104	63	72
64	70	70	126	154	109	71	69
67	73	68	106	122	88	68	68
68	79	60	70	77	66	58	75
69	85	64	58	55	61	65	83
70	87	69	68	65	73	78	90

A table of the histogram data is given below eg. the intensity value 68 occurs in 5 pixels in the image.

Value	Count	Value	Count	Value	Count	Value	Count	Value	Count
52	1	64	2	72	1	85	2	113	1
55	3	65	3	73	2	87	1	122	1
58	2	66	2	75	1	88	1	126	1
59	3	67	1	76	1	90	1	144	1
60	1	68	5	77	1	94	1	154	1
61	4	69	3	78	1	104	2		
62	1	70	4	79	2	106	1		
63	2	71	2	83	1	109	1		

The cumulative histogram of intensity values are given in the table below.

This cdf shows that the minimum value in the image is 52 and the maximum value is 154. The cdf of 64 for value 154 coincides with the number of pixels in the image. The cdf must be normalized to a range of 0 to 255. The general histogram equalization formula is:

$$h(v) = \operatorname{round}\left(\frac{\operatorname{cdf}(v) - \operatorname{cdf}_{\min}}{(M \times N) - \operatorname{cdf}_{\min}} \times (L-1)\right)$$

where $\operatorname{cdf}_{\min}$ is the minimum non-zero value of the cumulative distribution function (in this case 1), M × N gives the image's number of pixels (for the example above 64, where M is width and N the height) and L is the number of grey levels used (in most cases, like this one, 256). In the simple case that $\operatorname{cdf}_{\min}=0$, it can be seen that the equation/mapping is essentially a straight line of gradient *(L-1)/(M x N)*.

The equalization formula for the example scaling data from 0 to 255, inclusive, is:

$$h(v) = \text{round}\left(\frac{\text{cdf}(v) - 1}{63} \times 255\right)$$

For example, the cdf of 78 is 46. (The value of 78 is used in the bottom row of the 7th column.) The normalized value becomes:

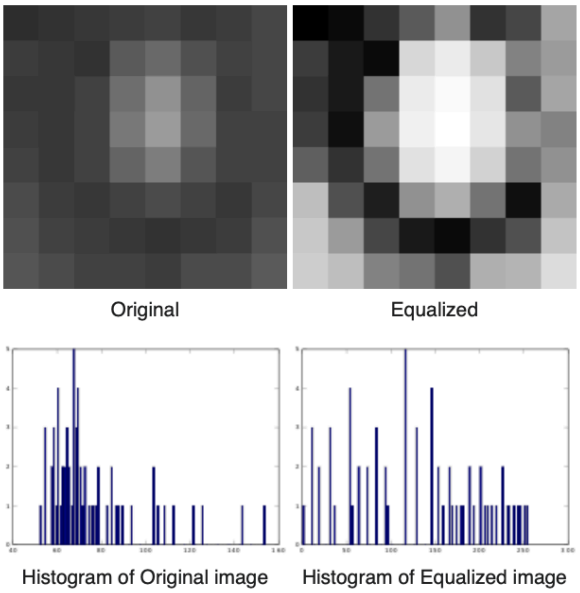
$$h(78) = \text{round}\left(\frac{46 - 1}{63} \times 255\right) = \text{round}\left(0.714286 \times 255\right) = 182$$

This is calculated for all values, and tabulated accordingly. The original intensities are then replaced by the mapped intensities, resulting in (try a few yourself):

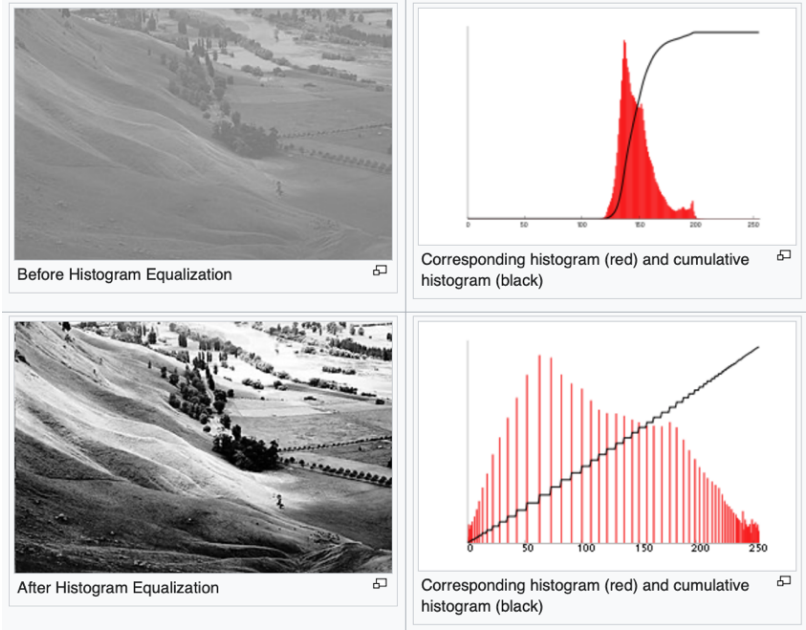
v, Pixel Intensity	cdf(v)	h(v), Equalized v
52	1	0
55	4	12
58	6	20
59	9	32
60	10	36
61	14	53
62	15	57
63	17	65
64	19	73
65	22	85
66	24	93
67	25	97
68	30	117
69	33	130
70	37	146
71	39	154
72	40	158
73	42	166
75	43	170
76	44	174
77	45	178
78	46	182
79	48	190
83	49	194
85	51	202
87	52	206
88	53	210
90	54	215
94	55	219
104	57	227
106	58	231
109	59	235
113	60	239
122	61	243
126	62	247
144	63	251
154	64	255

0	12	53	32	190	53	174	53
57	32	12	227	219	202	32	154
65	85	93	239	251	227	65	158
73	146	146	247	255	235	154	130
97	166	117	231	243	210	117	117
117	190	36	146	178	93	20	170
130	202	73	20	12	53	85	194
146	206	130	117	85	166	182	215

A comparison of the original image and the one resulting from histogram equalisation is given below.



The 8x8 image example given above was part of a landscape image. The original image and the equalised image are shown below; notice the improvement in contrast.

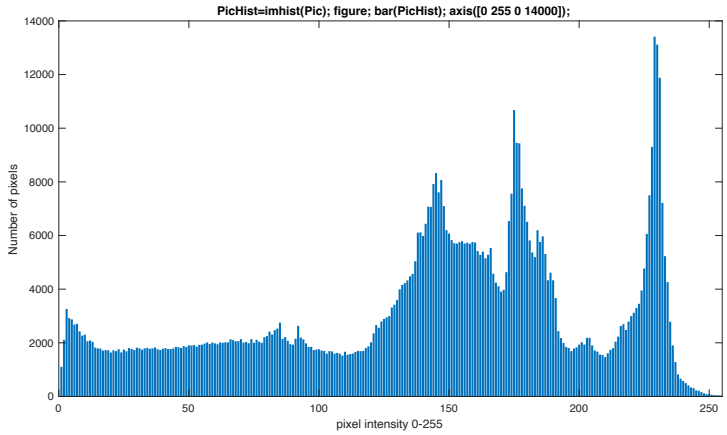


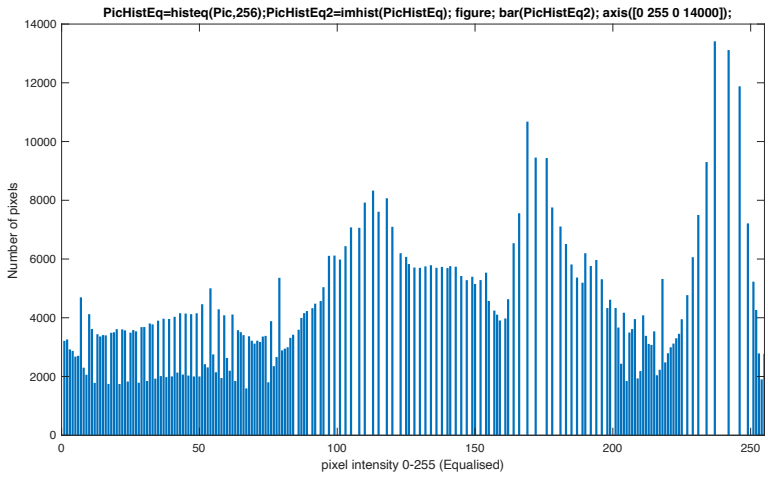
Using Matlab

Although relatively straightforward to perform and tabulate the equalised mapping of an image, Matlab provides a convenient function for Histogram Equalisation (`histeq`). Let's read in Hugo.jpg, grayscale it, and then use histogram Equalisation on the image

```
%Read image and grayscale
PicColour=imread("Hugo.jpg");
Pic = im2gray(PicColour);

%Plot Histogram of original image
PicHist=imhist(Pic);
figure;
bar(PicHist);
```

```
axis([0 255 0 14000]);
title('PicHist=imhist(Pic); figure; bar(PicHist); axis([0 255 0 14000]);');
ylabel('Number of pixels');
xlabel('pixel intensity 0-255');

%Perform Histogram Equalisation on image
PicHistEq=histeq(Pic,256);

%Display Histogram of 'Equalised Image'
PicHistEq2=imhist(PicHistEq);
figure;
bar(PicHistEq2);
axis([0 255 0 14000]);
title('PicHistEq=histeq(Pic,256);PicHistEq2=imhist(PicHistEq); figure;
bar(PicHistEq2); axis([0 255 0 14000]);');
ylabel('Number of pixels');
xlabel('pixel intensity 0-255 (Equalised)');

%Display 'Equalised Image'
figure;
imshow(PicHistEq);
title('PicHistEq=histeq(Pic,256);imshow(PicHistEq);');
```





Notice the increased contrast compared to the original image

Pic = im2gray(PicColour);

Original

PicHistEq=histeq(Pic,256);imshow(PicHistEq);

After Histogram Equalisation