CS663: Digital Image Processing

Assignment 1, Question 2

(a) Linear Contrast Stretching}

function [outputImage] = myLinearContrastStretching(inputImage)

[sizeX, sizeY] = size(inputImage);

outputImage = zeros(sizeX, sizeY, 'uint8');

minVal = min(min(inputImage));

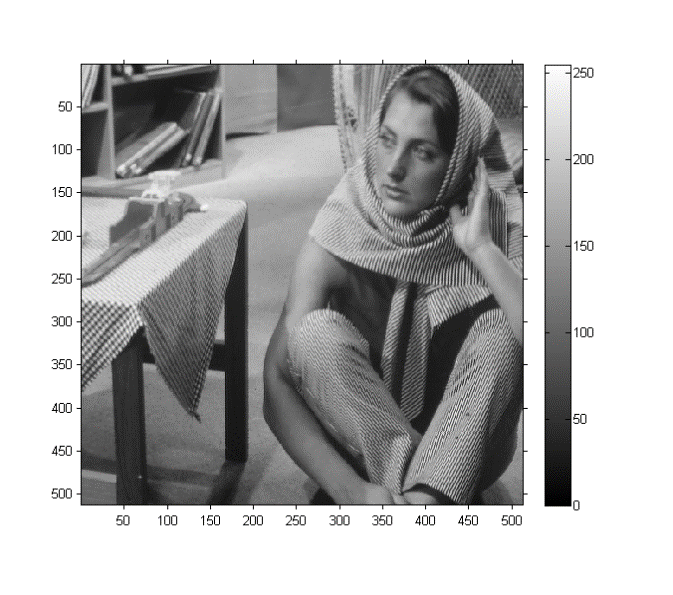
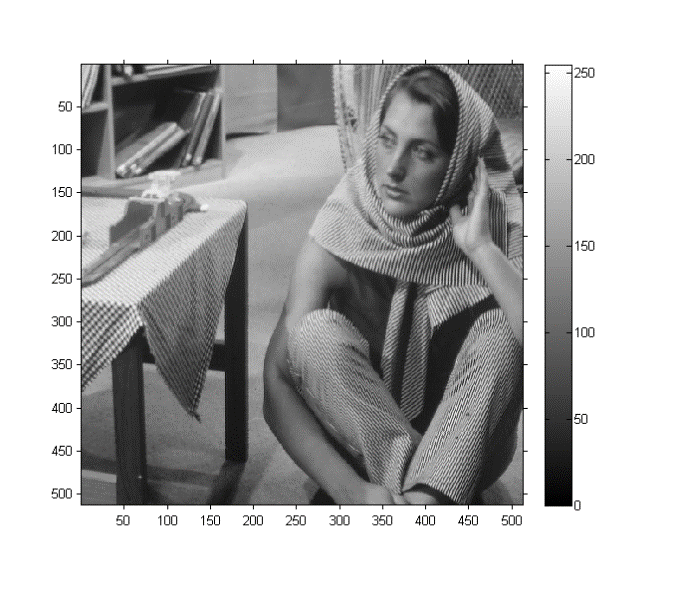
maxVal = max(max(inputImage));

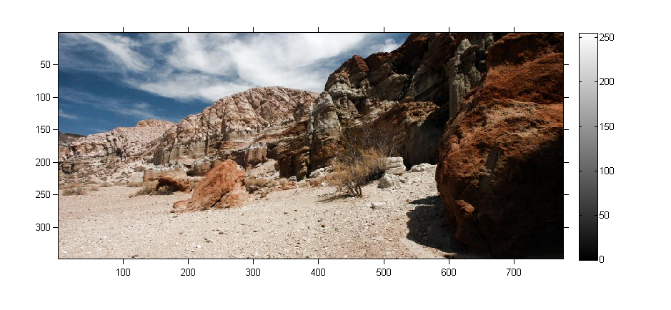
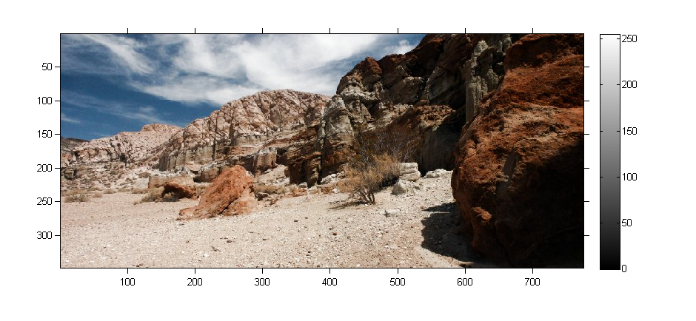
f = @(x) uint8(255.0 .\* double(x - minVal) ./ double(maxVal - minVal));

outputImage = f(inputImage);

end

**Code Explanation:** The function myLinearContrastStreching takes inputImage as a single channel 8 bit image and returns outputImage i.e. a single channel 8 bit image with same dimensions. We first find out maximum and minimum of the image, *maxVal* and *minVal*. Then the range of image, [minVal, maxVal] is stretched to [0,255] by applying the transformation on





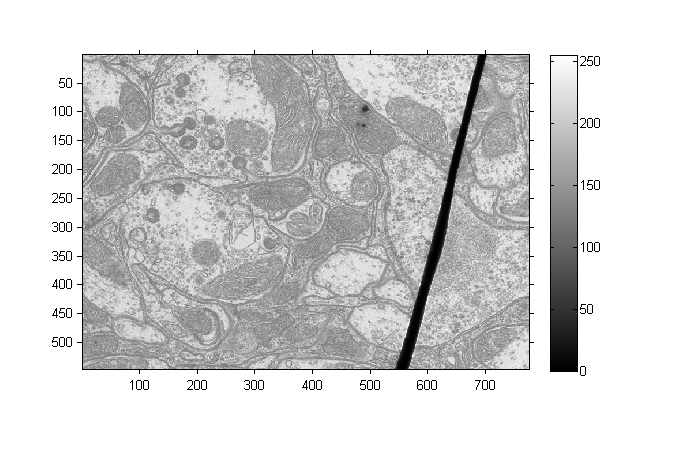
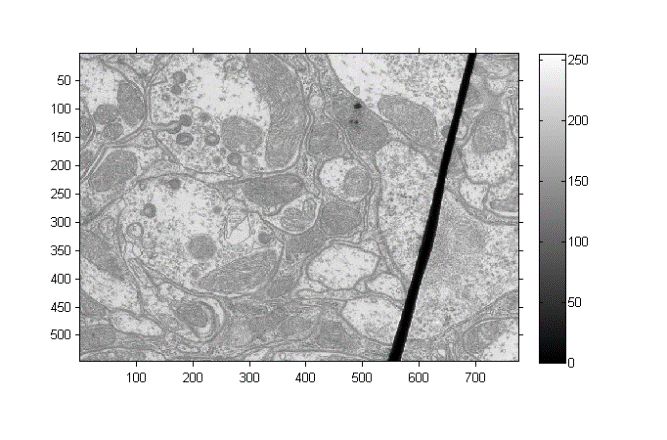


Figure. Images before and after Linear Contrast Stretching.

**Comments:** Note that there was no noticeable difference between two images as the minimum and maximum values of the intensities in image were close 0 and 255 respectively.

(b) Global Histogram Equalization

function [outputImage] = myHE(inputImage)

% myAHE takes the inputImage, 8 bit single channel image as the input and

% enhances it using Histogram equalisation

[sizeX, sizeY] = size(inputImage);

outputImage = zeros(sizeX, sizeY, 'uint8');

ColorNumber = 256;

bins = zeros(ColorNumber, 1);

cdf = zeros(ColorNumber, 1);

for i = 1:sizeX

for j = 1:sizeY

bins(int32(inputImage(i,j))+1, 1) = bins(int32(inputImage(i,j))+1, 1) + 1;

end

end

bins = bins ./ (sizeX\*sizeY);

sum = 0;

for i = 1:ColorNumber

sum = sum + bins(i);

cdf(i,1) = sum;

end

for i = 1:sizeX

for j = 1:sizeY

outputImage(i,j) = uint8(255.0\*cdf(inputImage(i,j)+1));

end

end

end

**Code Explanation:** The function myHE takes inputImage as a single channel 8 bit image and returns outputImage i.e. a single channel 8 bit image with same dimensions. We first calculate the histogram by calculating the total number of pixels with intensity a, for all intensities [0,255]. Then the cumulative distribution function (c.d.f.) is calculated and the intensity x is applied the following transformation for each pixel:

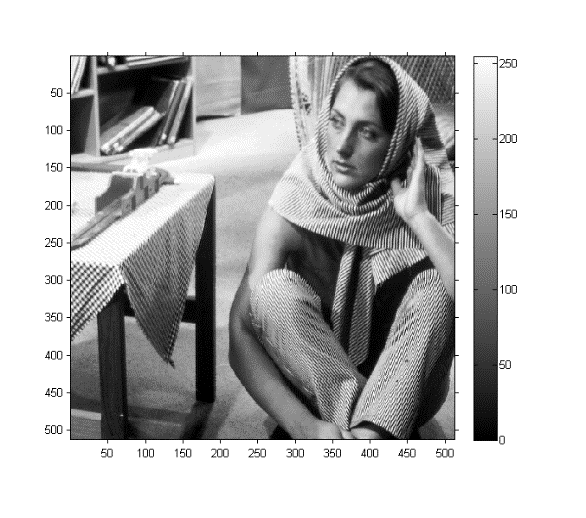
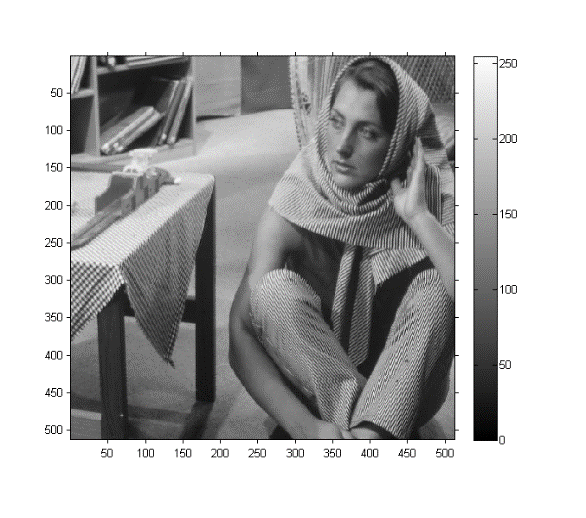


Figure. (left) Barbara, original image, (right) After global histogram equalization

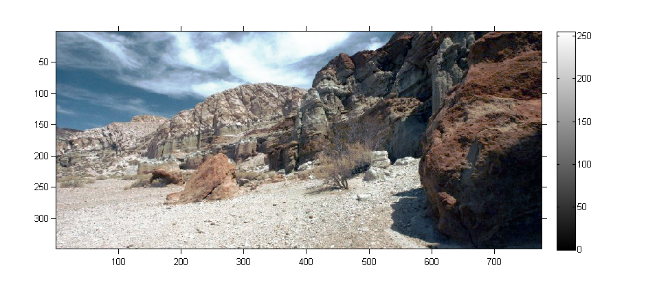
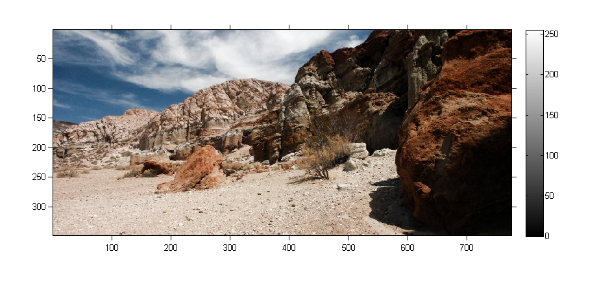


Figure. (left) canyon, original image, (right) After global histogram equalization

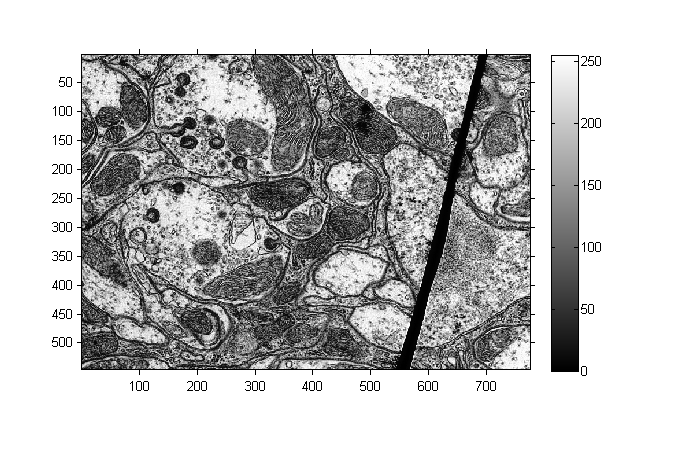
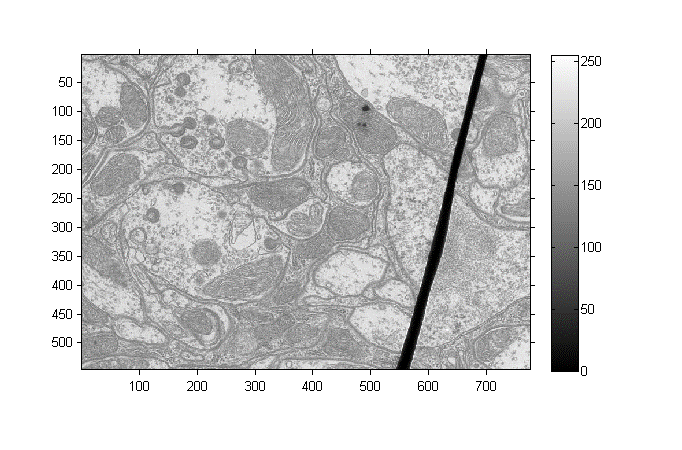


Figure. (left) TEM, original image, (right) After global histogram equalization

The enhancement in contrast is clearly visible in the images on the right.

(c) Adaptive Histogram Equalization

function [outputImage] = myAHE(inputImage, N)

% Applies adaptive histogram equalization on inputImage with window size N

% and returns output Image

funAHE = @(x) myAHEHelper(x);

outputImage = nlfilter(inputImage, [N, N], funAHE);

end

function [outputValue] = myAHEHelper(inputImage)

% myAHE takes the inputImage, 8 bit single channel image as the input and

% enhances it using Adaptive Histogram equalisation

[sizeX, sizeY] = size(inputImage);

ColorNumber = 256;

bins = zeros(ColorNumber, 1);

cdf = zeros(ColorNumber, 1);

for i = 1:sizeX

for j = 1:sizeY

bins(int32(inputImage(i,j))+1, 1) = bins(int32(inputImage(i,j))+1, 1) + 1;

end

end

bins = bins ./ (sizeX\*sizeY);

sum = 0;

for i = 1:ColorNumber

sum = sum + bins(i);

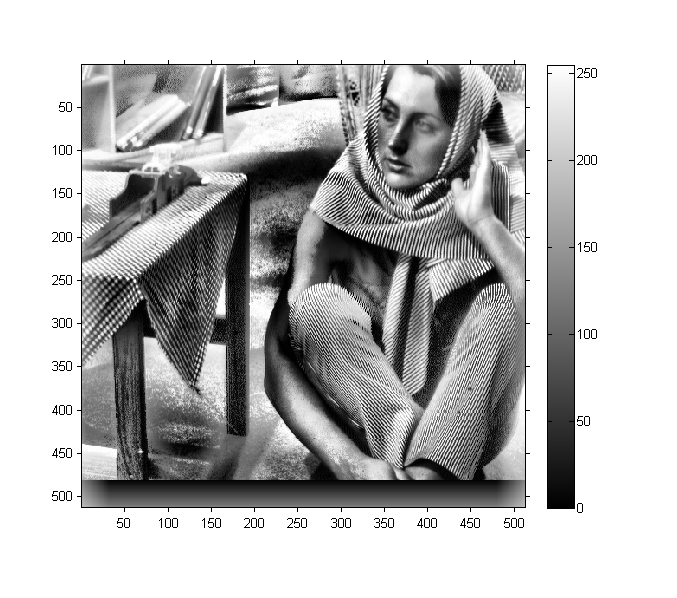
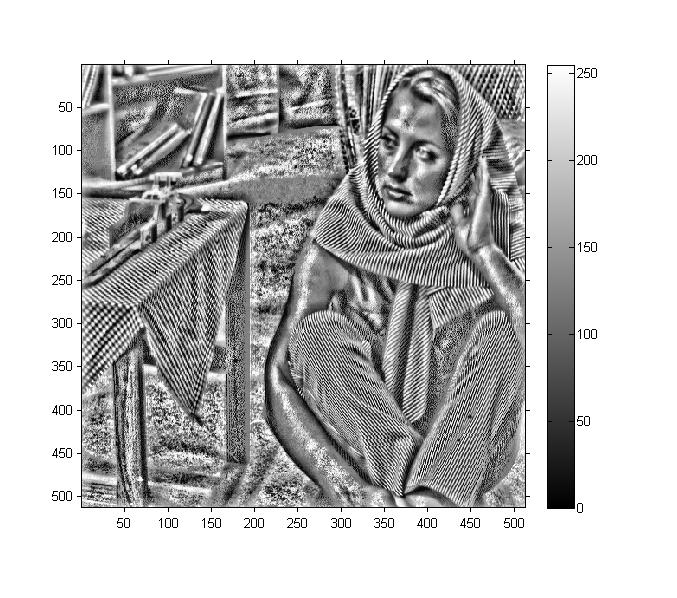
cdf(i,1) = sum;

end

outputValue = uint8(255.0\*cdf(inputImage(int32(sizeX/2), int32(sizeY/2))+1));

end

**Code Explanation:** The function myAHE takes inputImage as a single channel 8 bit image and window size N as an input and returns outputImage i.e. a single channel 8 bit image with same dimensions. Using function *myAHEHelper* and *nfilter*, histogram equalisation is done for window of size NXN surrounding each pixel. myAHEHelper does the histogram equalization for each window and returns a scalar, the intensity of the central pixel of the window after histogram equalization.



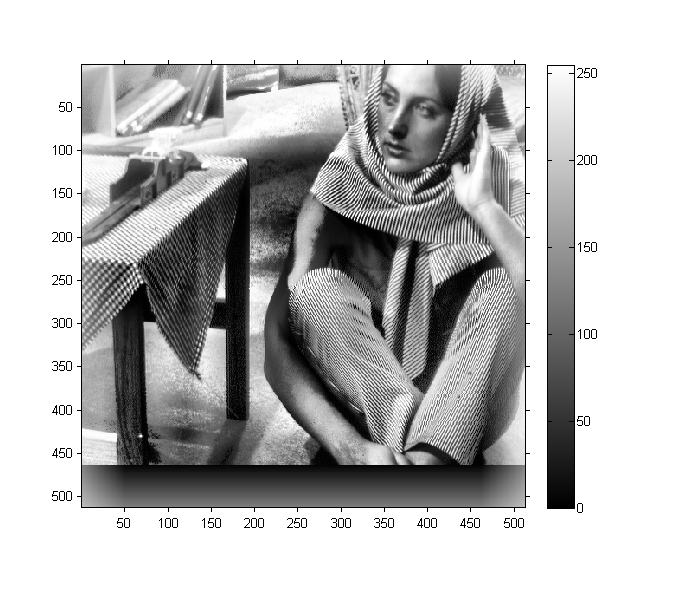
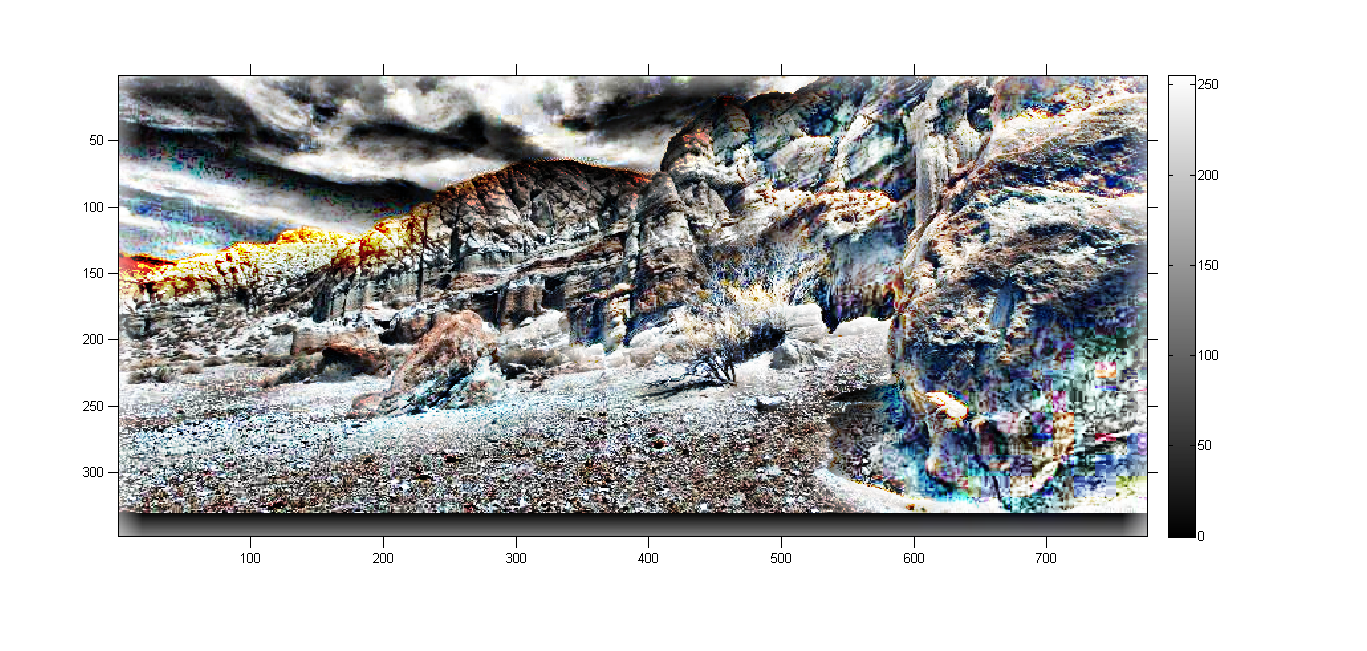
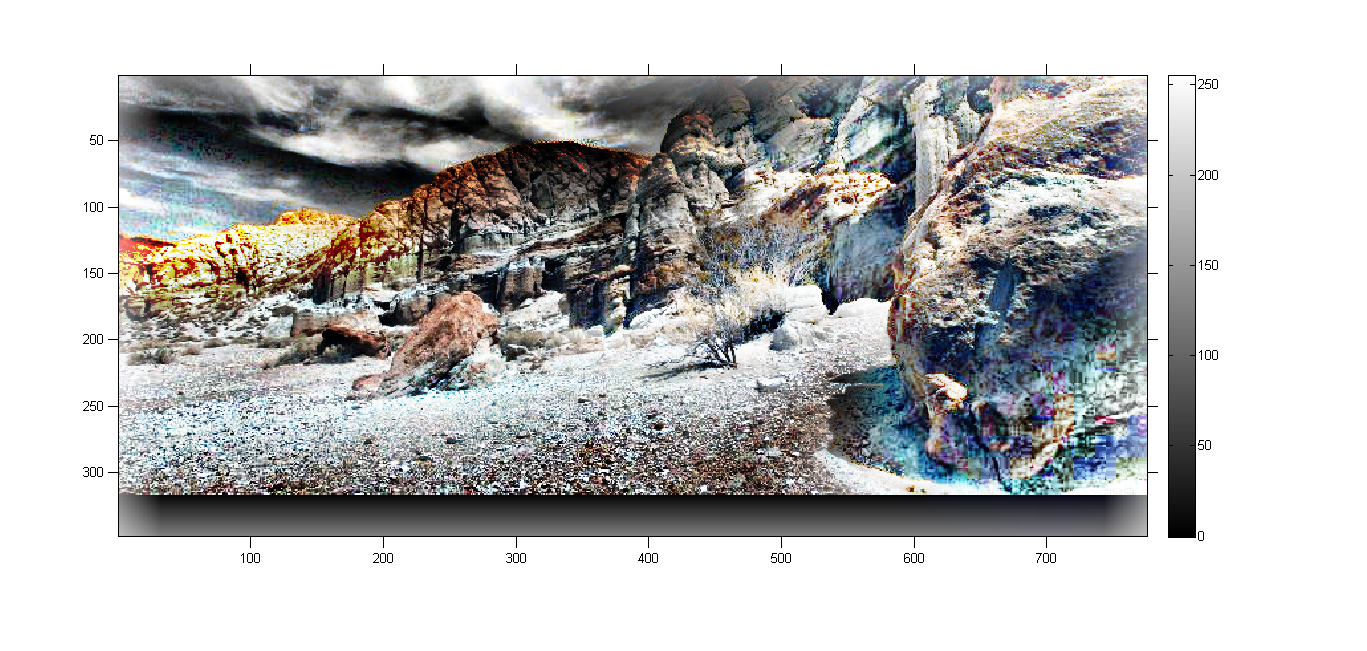


Figure. barbara after Adaptive Histogram Equalization: N= 25 (top), N = 64 (center), N=100 (bottom)





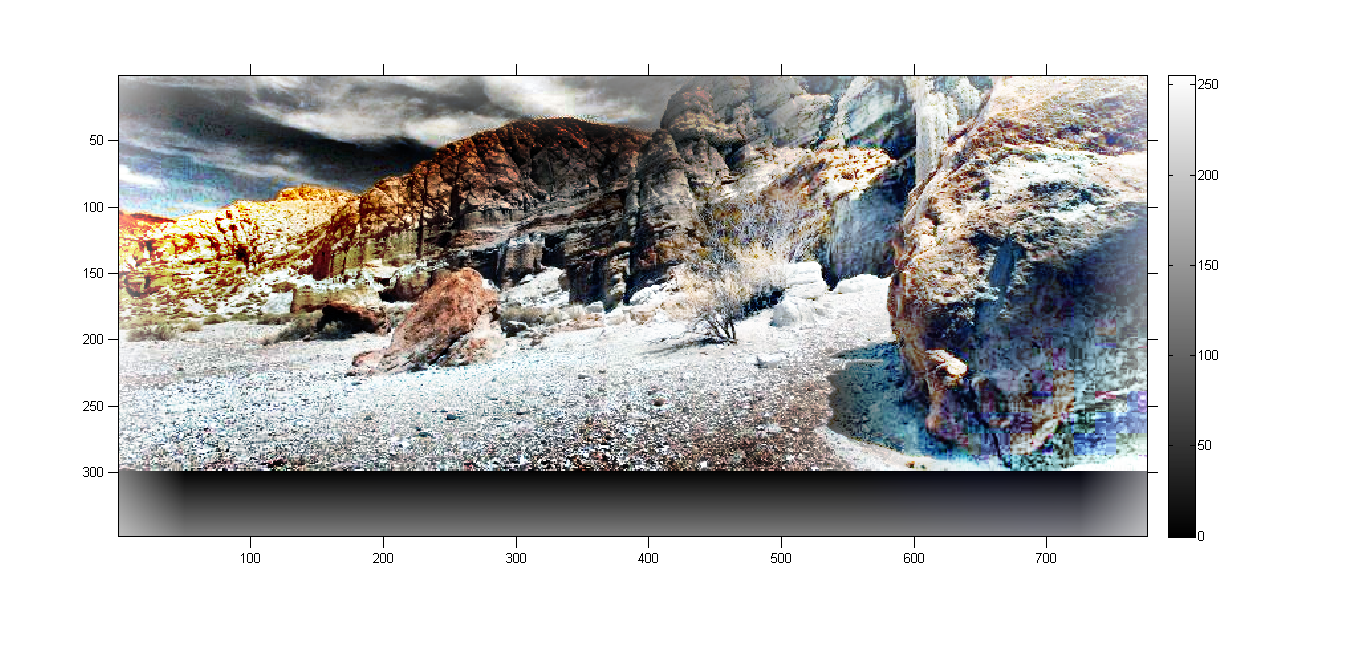
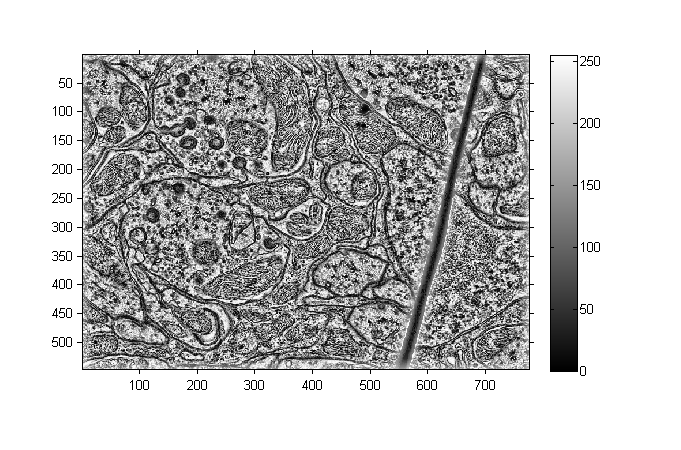
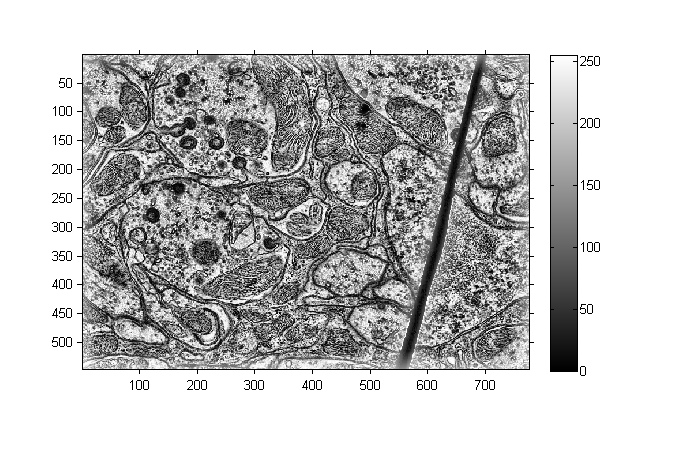


Figure. canyon after Adaptive Histogram Equalization: N= 36 (top), N = 64 (center), N=100 (bottom)





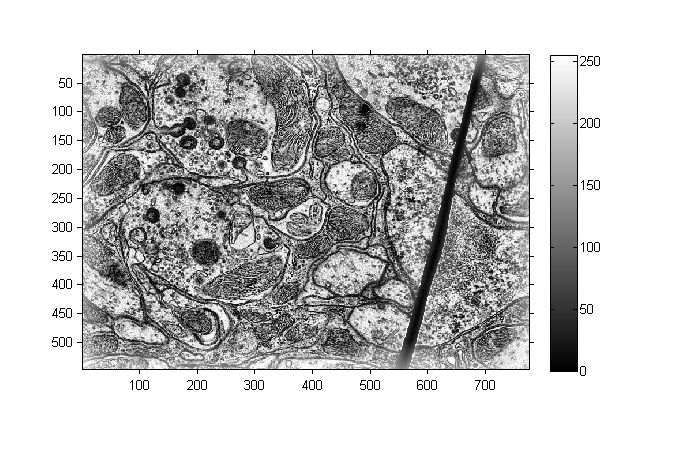


Figure. TEM after Adaptive Histogram Equalization: N= 36 (top), N = 64 (center), N=100 (bottom)

**Comments:** As we increase N from the optimal value, the amount of detail doesn’t increase a lot as seen from the difference in images of N = 100 and N= 64. The image gets effectively cropped and takes significantly larger computational time.

Decreasing N leads higher noise amplification in the image as seen in images with N = 36. The “detail” function is a concave function with a global minimum. So initially on increasing the window size will increase the details but after a certain value, it will keep decreasing.

(c) Contrast Limited Adaptive Histogram Equalization

function [outputValue] = myCLAHEHelper(inputImage, C)

% myAHE takes the inputImage, 8 bit single channel image as the input and

% enhances it using Histogram equalisation

[sizeX, sizeY] = size(inputImage);

ColorNumber = 256;

bins = zeros(ColorNumber, 1);

cdf = zeros(ColorNumber, 1);

for i = 1:sizeX

for j = 1:sizeY

bins(int32(inputImage(i,j))+1, 1) = bins(int32(inputImage(i,j))+1, 1) + 1;

end

end

C = C \* sizeX \* sizeY;

points = 0;

for i = 1:ColorNumber

if bins(i) > C

points = points + (bins(i) - C);

bins(i) = C;

end

end

points = points ./ ColorNumber;

bins = (bins + points) ./ (sizeX \* sizeY);

sum = 0;

for i = 1:ColorNumber

sum = sum + bins(i);

cdf(i,1) = sum;

end

outputValue = uint8(255.0\*cdf(inputImage(int32(sizeX/2), int32(sizeY/2))+1));

end

**Code Explanation:** The function myCLAHE takes inputImage as a single channel 8 bit image, window size N and threshold parameter C as an input and returns outputImage i.e. a single channel 8 bit image with same dimensions. Using function *myCLAHEHelper* and *nfilter*, contrast limited histogram equalisation is done for window of size NXN surrounding each pixel. myCLAHEHelper calculates the histogram for each window. Then the total weight of the histogram lying above the threshold parameter C is cutoff from the histogram and added equally to each bin of the histogram. Then cdf is calculated using the modified histogram and following transformation is applied on the intensity of the center pixel x

and the new intensity of the pixel is given as the output for each window.