3/18/2015

Project Assignment 2 Report (Group 1)

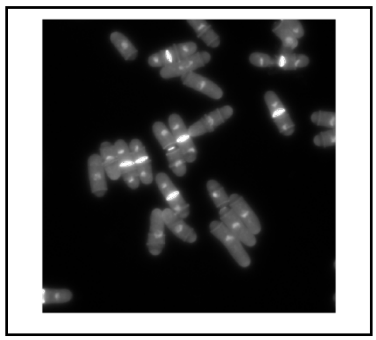
Siddharth Gurdasani (`sgurdasa`)
Easwaran Ramamurthy (`eramamur`)
Aniket Rao (`aniketr`)
Patrick Ropp (`propp`)

I. Implementation of Gaussian Filters

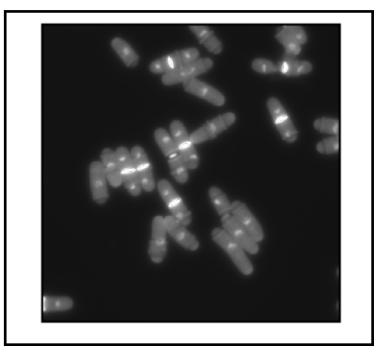
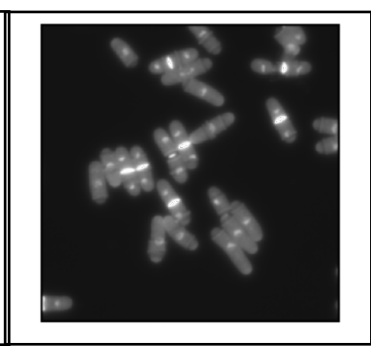
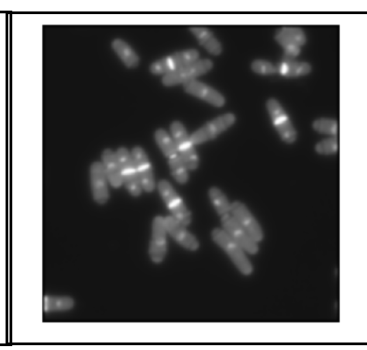
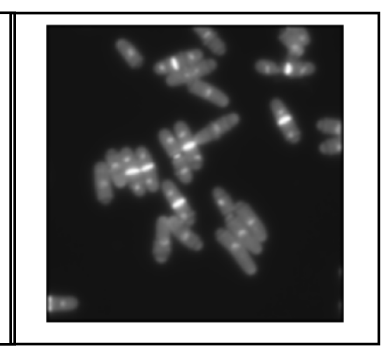
PART 1:

The aim of question 1.1 was to implement a Gaussian filter in MATLAB with varying standard deviation sigma as user input. The code for this question is given below and also submitted in `q1part1.m`.

The original image is shown below:



The images below are the ones with sigma = 1,2,5,7 respectively.

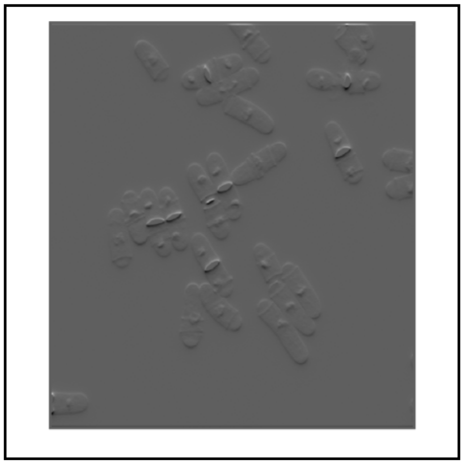





As sigma increases, the images get blurrier. A Gaussian filter is essentially a low pass filter. Thus, it removes noise but also removes detail. A large sigma means a wider Gaussian filter that in turn means greater smoothening. Thus, as sigma goes from 1 to 7, a greater blur is seen as greater amount of detail is removed.

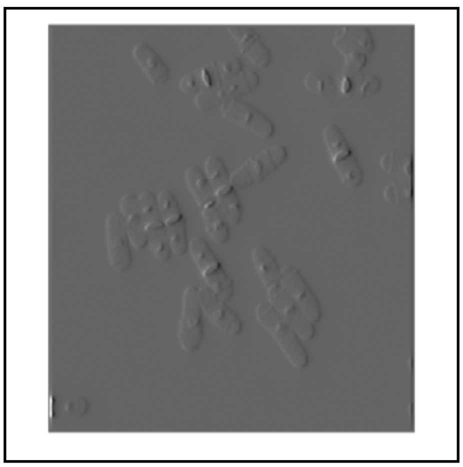
PART 2:

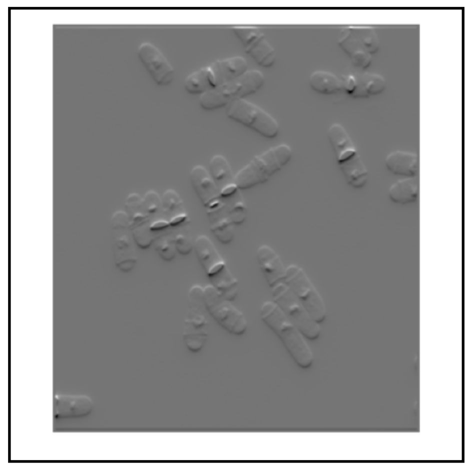
The aim of question 1.2 was to calculate image derivates. The original image used was the same as question 1.1. The code is given below and also submitted in file `q1part2.m`.

For sigma = 1, the derivative in the horizontal and vertical direction are shown on the next page respectively.

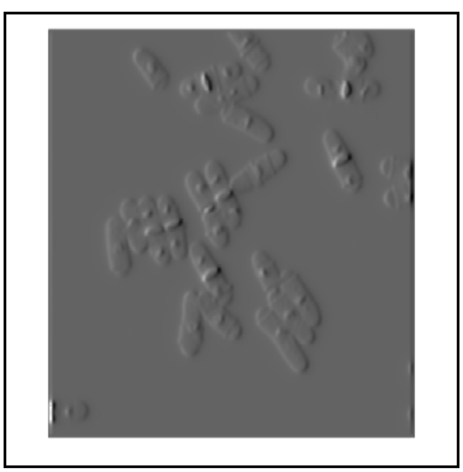


Similarly for sigma = 2,





And for sigma = 5 ,



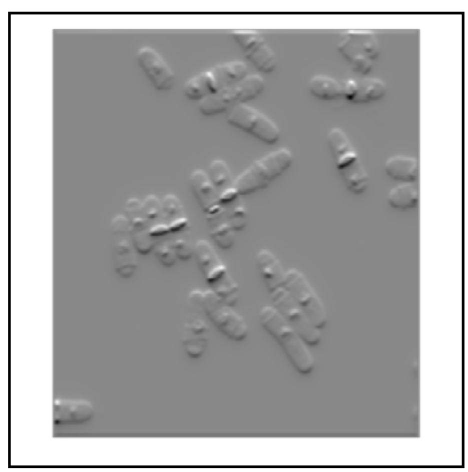
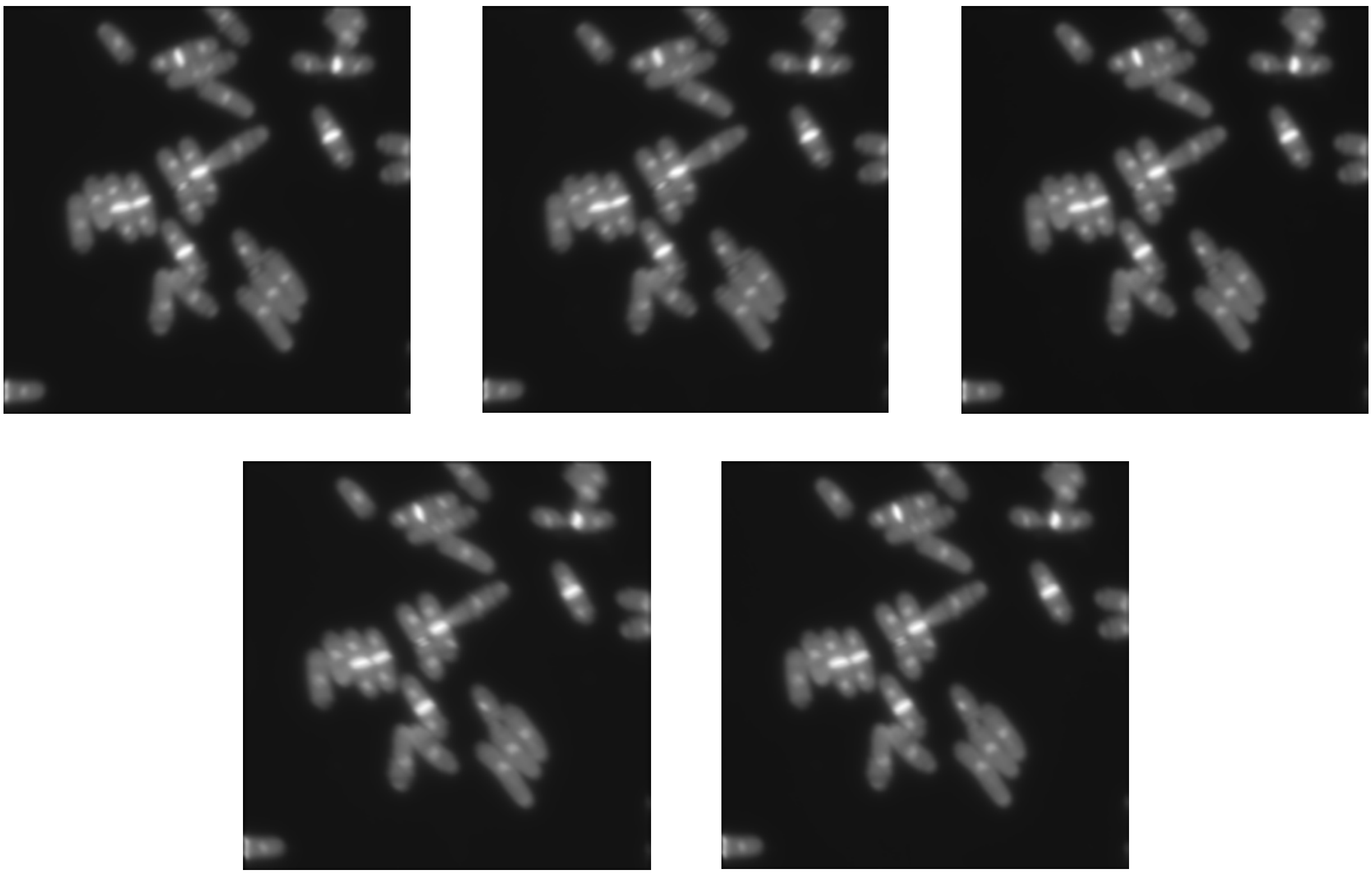


Image gradients measure the change in intensity levels of an image along a particular direction. The images on the left have derivatives taken along horizontal direction and any change in intensity level along the horizontal direction can be seen prominently. The vertical bright bands are more prominent on the images on the left because these represent a sudden change in intensity along the x-direction. Conversely, for the images on the right, the derivatives are taken along the vertical direction and the horizontal bright bands are more prominent as the filter looks for sudden changes in intensity in the y-direction.

PART 3:

For part 3, we implemented the non-recursive anisotropic Gaussian Convolution filter given in Geusebroek et al., 2003, IEEE Trans. Image Processing as follows. The code is also submitted in file `q1part3AnisotropicFilter.m`.

The figures obtained at different orientations at $\sigma = 10$ in longitudinal direction and $\sigma = 5$ in lateral direction are given below. These are in increasing order of angle from left to right with angles = 30, 60, 90, 120 and 150 degrees respectively.



The image filtered at 90 degrees is equal to the image filtered in part 1 of this question 1. Further, we can see a slight tilt in the bright spots when comparing the 30 degree filtered image with the 60 degree filtered image and also when comparing these two with the 90 degree filtered image. The bright spots in the filtered output progressively show more tilt as the angle of the t-axis increases.

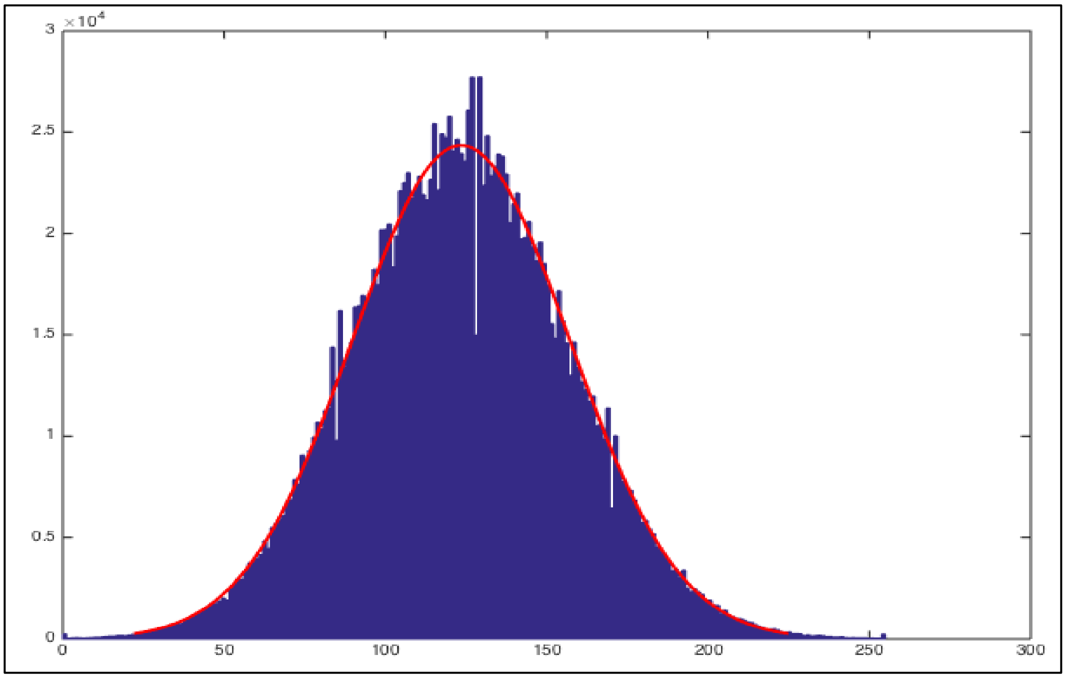
II. Microscope Characterization

PART 1:

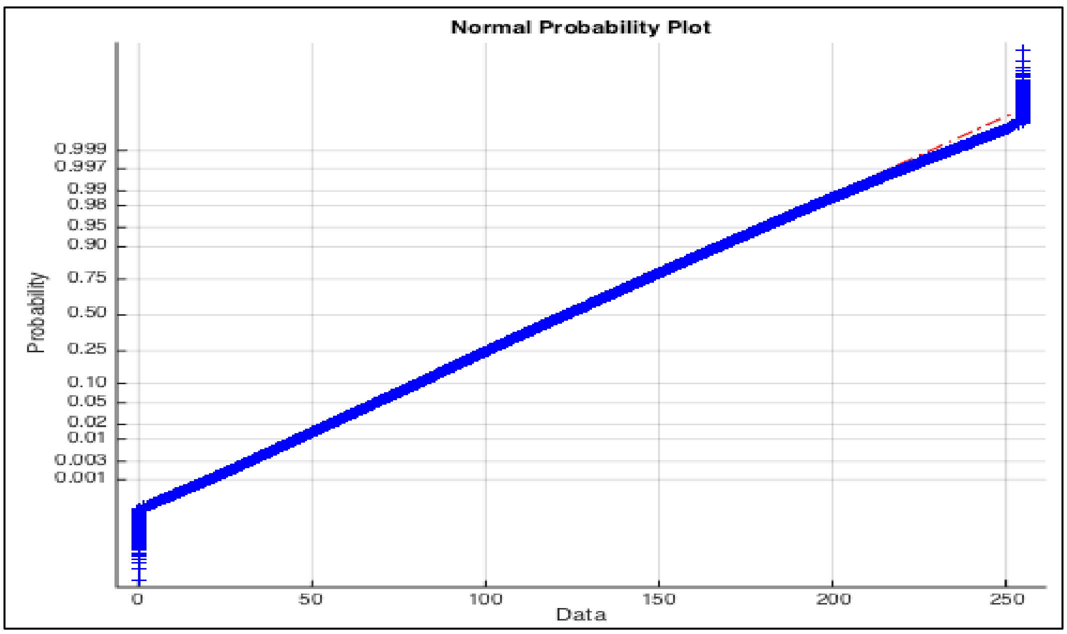
The information was downloaded as provided.

PART 2:

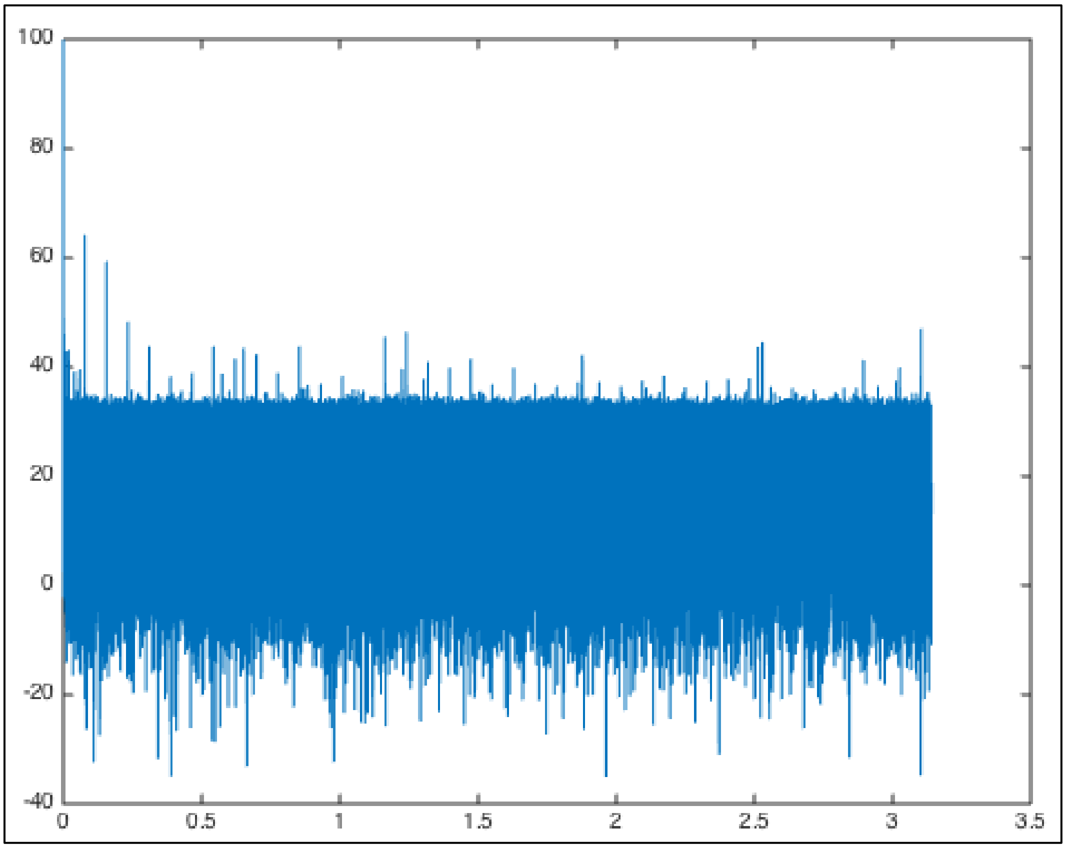
Q2.2.1: We can see that the noise signal follows the normal distribution as shown by the plot on the next page



The histogram shows the noise of the overall sequence of images fitted with a Gaussian curve. From the figure, we can observe that the noise more or less follows normal distribution, though there are some points where there are deviations.



Above is shown the normal probability plot of the noise overall the sequence of images. The blue represents the observed data while the red line represents the normal distribution. It can be observed from the plot, that the noise is not exactly linear and deviates from the normal distribution near the right hand side of the plot. From the above two plots, it can be concluded that the noise signal follows normal distribution.

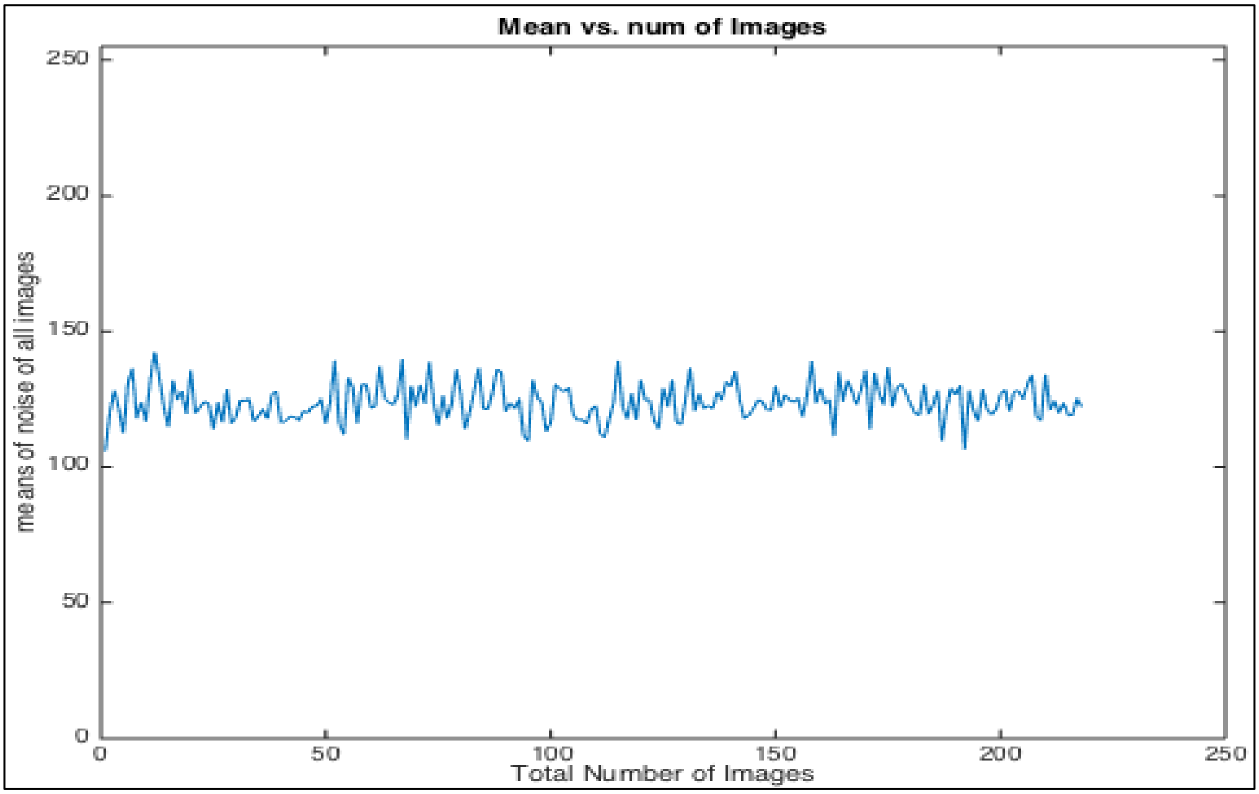


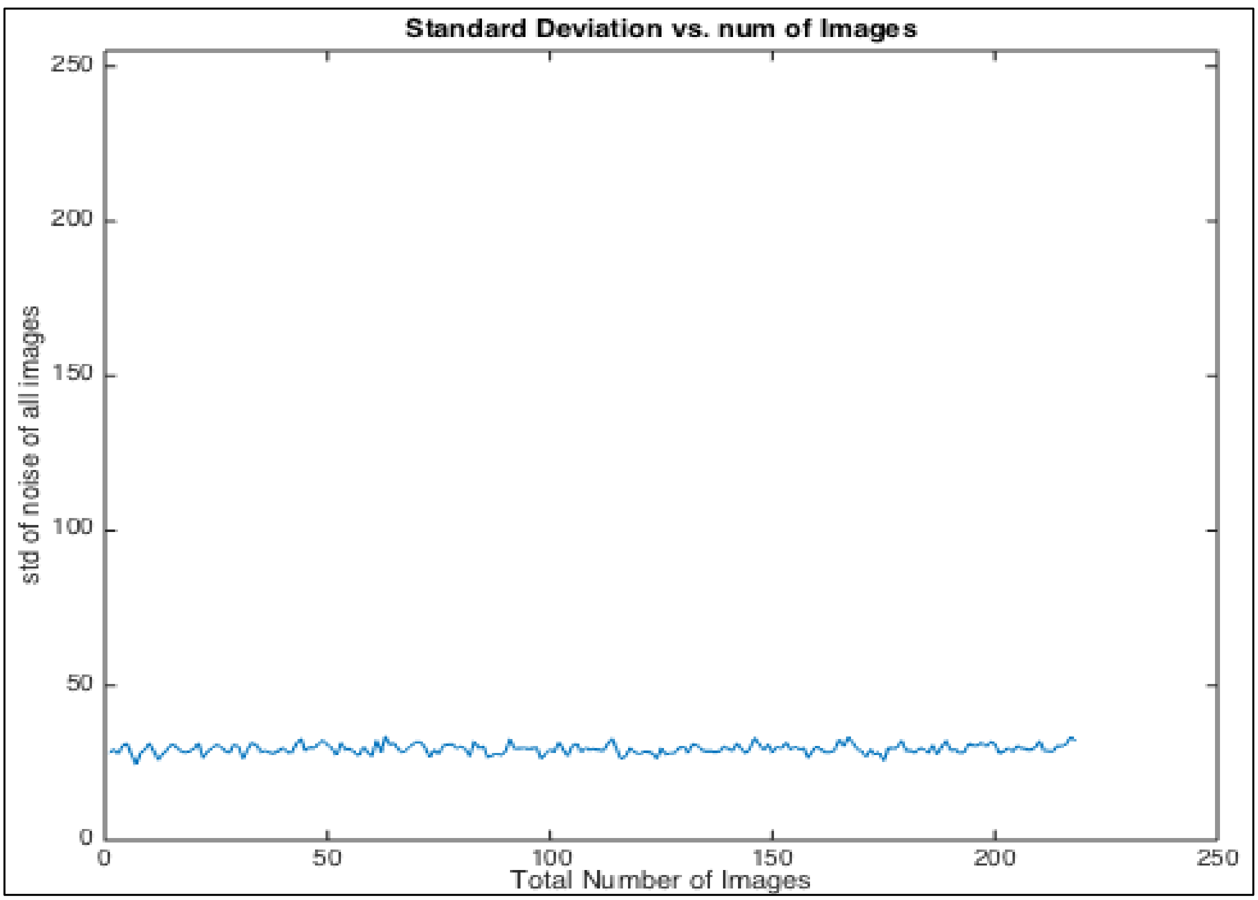
Above shown is the power spectrum density estimate plot, which should be constant for the noise to called as white noise. Also, the slope of the line passing through the power spectrum density estimate turns out to be closer to zero (-0.024982) using `polyfit` function in matlab (code submitted on Box), which is a characteristic of white noise. Thus, it can be said that the noise signal here is a white noise.

Code submitted: `q2background.m`

Images submitted: `q2normplot.tif`, `q2powerspectraldensityspectrum.tif`, `q2noisehistogram.tif`

Q2.2.2: No, the noise distribution doesn't change over time. Following are the 2D plots of the mean of the noise of each image and the standard deviation of the noise of each image in the sequence versus the number of images.


Mean vs. num of Images
means of noise of all images
Total Number of Images
0
50
100
150
200
250

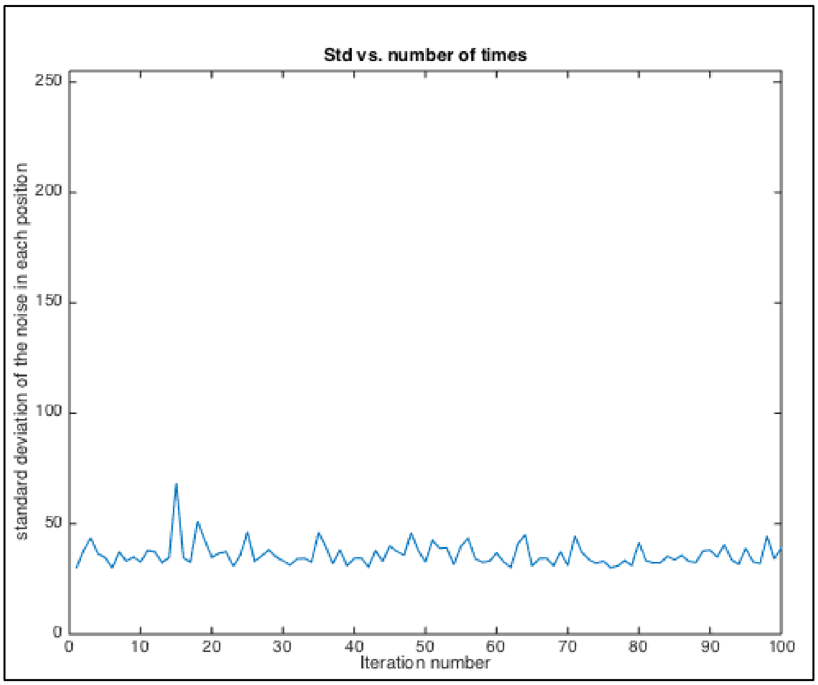

Standard Deviation vs. num of Images
std of noise of all images
Total Number of Images
0
50
100
150
200
250

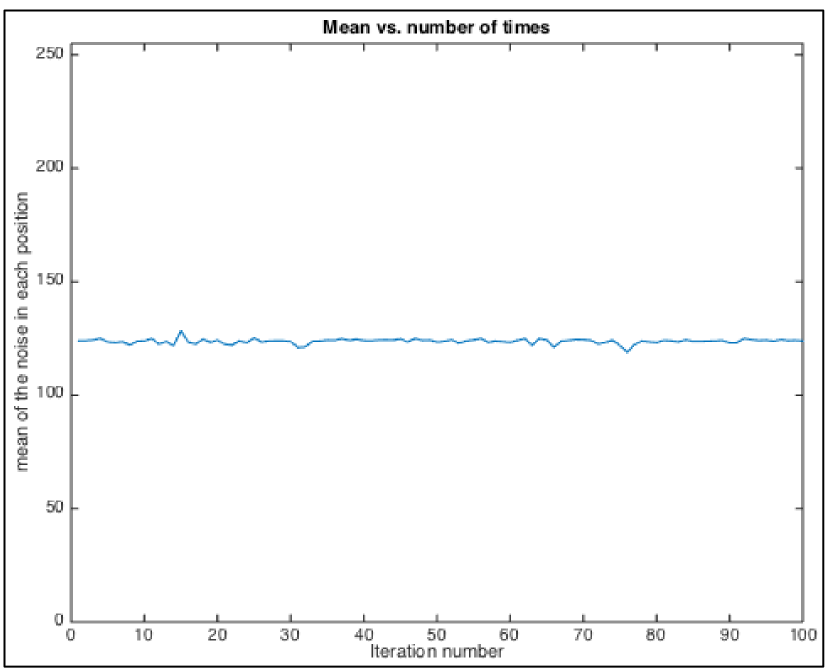
From the plots, we can observe that there is not much variation in the mean of noises over all images in the sequence and similar trends in the standard deviations of the noises over all images in the sequence. The mean and standard deviation over the noise was calculated over all the images in the sequence and the mean noise vector was plotted against the total number of images, similar calculations for standard deviation. Thus, we can say from the plots that the noise distribution doesn't significantly change over time.

Code submitted: `q2background.m`

Images submitted: `q2.2changeovertimeMean.tif`, `q2.2changeovertimeStd.tif`

Q2.2.3: No, the noise distribution doesn't change over space. Following are the 2D plots of means of noise for different positions in the sequence of images versus the iteration number, and in the similar way the standard deviation versus the iteration number.





Different random positions were selected for each iteration in the sequence of images, i.e. for each iteration random two x and two y coordinates were selected and the minimum of the two x coordinates and two y coordinates was chosen as the starting point coordinate. The difference between the two x and two y coordinates was given as the width and height for the rectangular box to be selected. Then noise is extracted from that box, normalized and then mean, standard deviation over the whole noise for a sequence is calculated. This mean vector and the standard deviation is then plotted to get the above two plots.

From these two plots, it can be observed that the mean and standard deviation don't show much variation across different space segments selected randomly from the sequence of images.

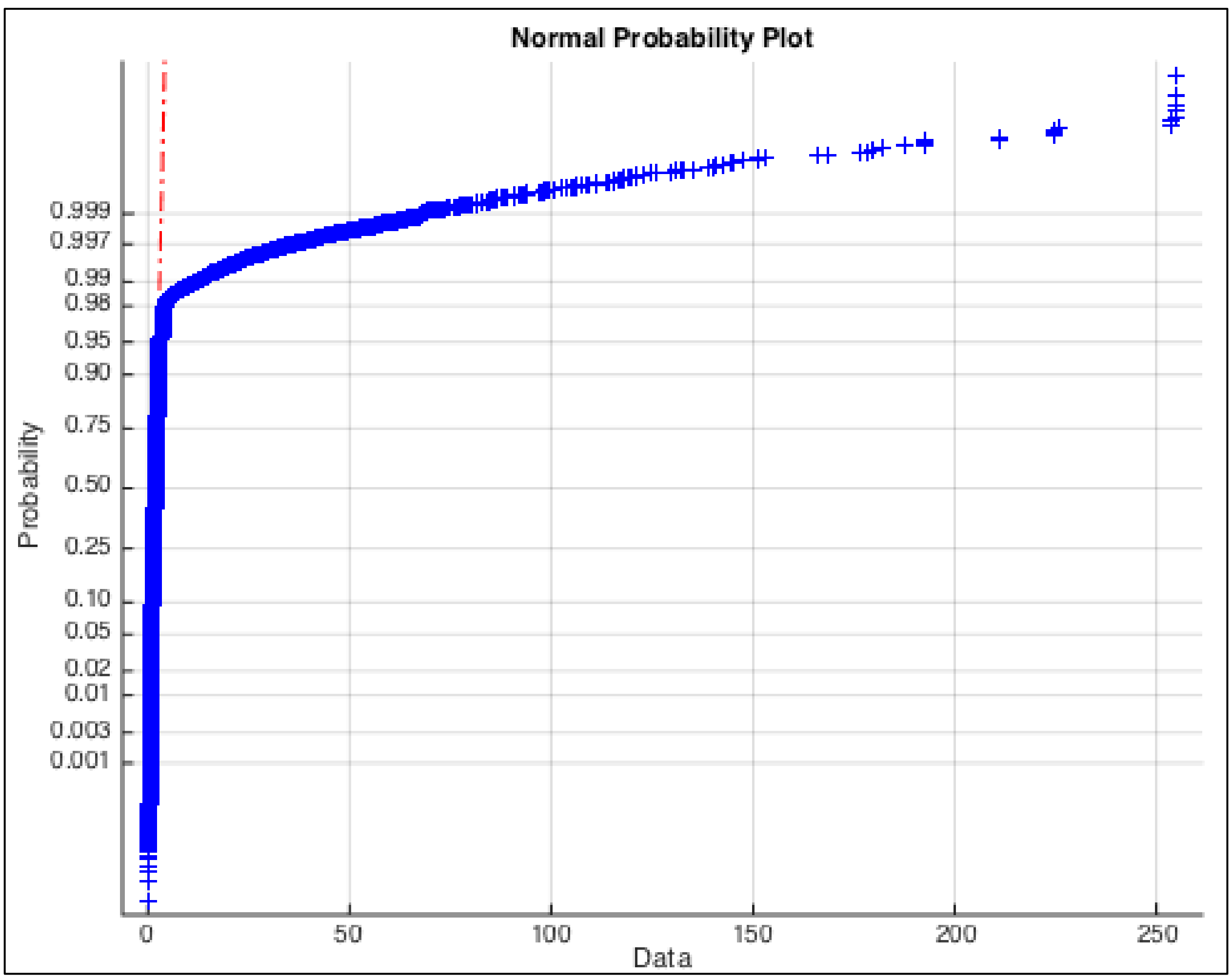
Code submitted: `q2calcchangeoverspace.m`

Images submitted: `q2.2changeoverspacemean.tif, q2.2changeoverspacestd.tif`

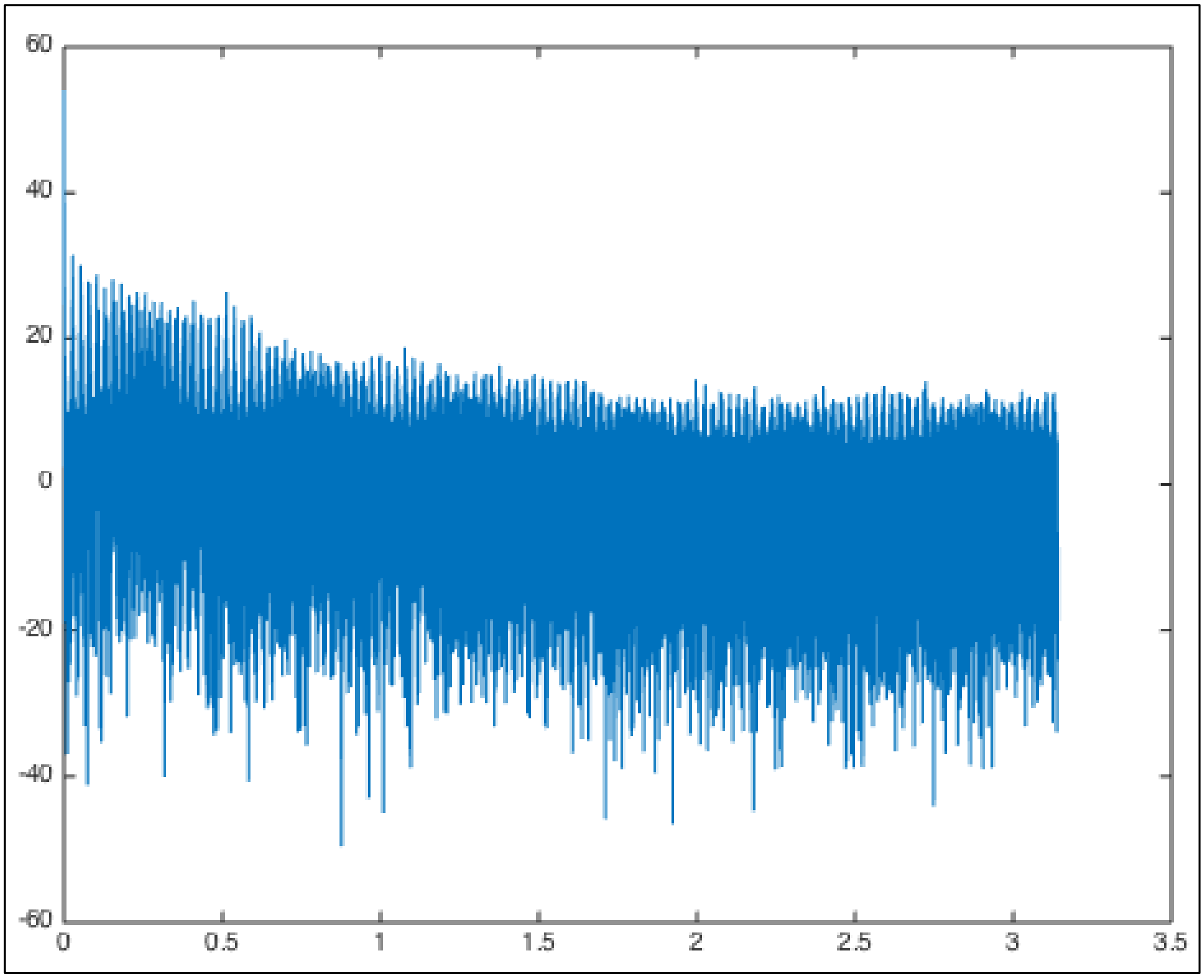
Q2.2.1 for the two images from JCB viewer in Project Assignment 01:

No, both the images from the Project assignment 1 don't follow a normal distribution. It can be observed from the following normal probability plots plotted using matlab in built function `normplot`.

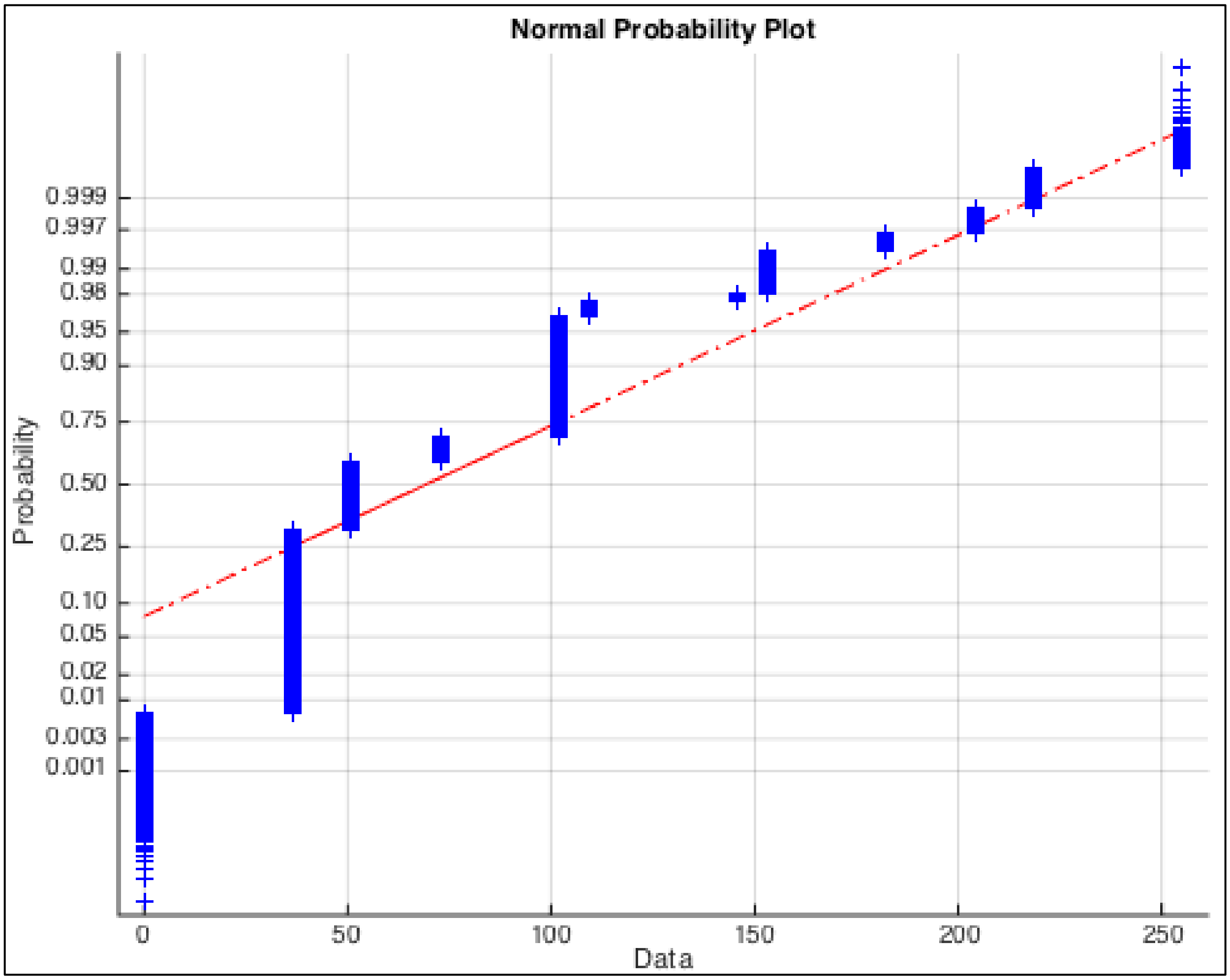
Following is the normal probability plot for image 1 (PA01_Image1.tiff submitted on the Box):



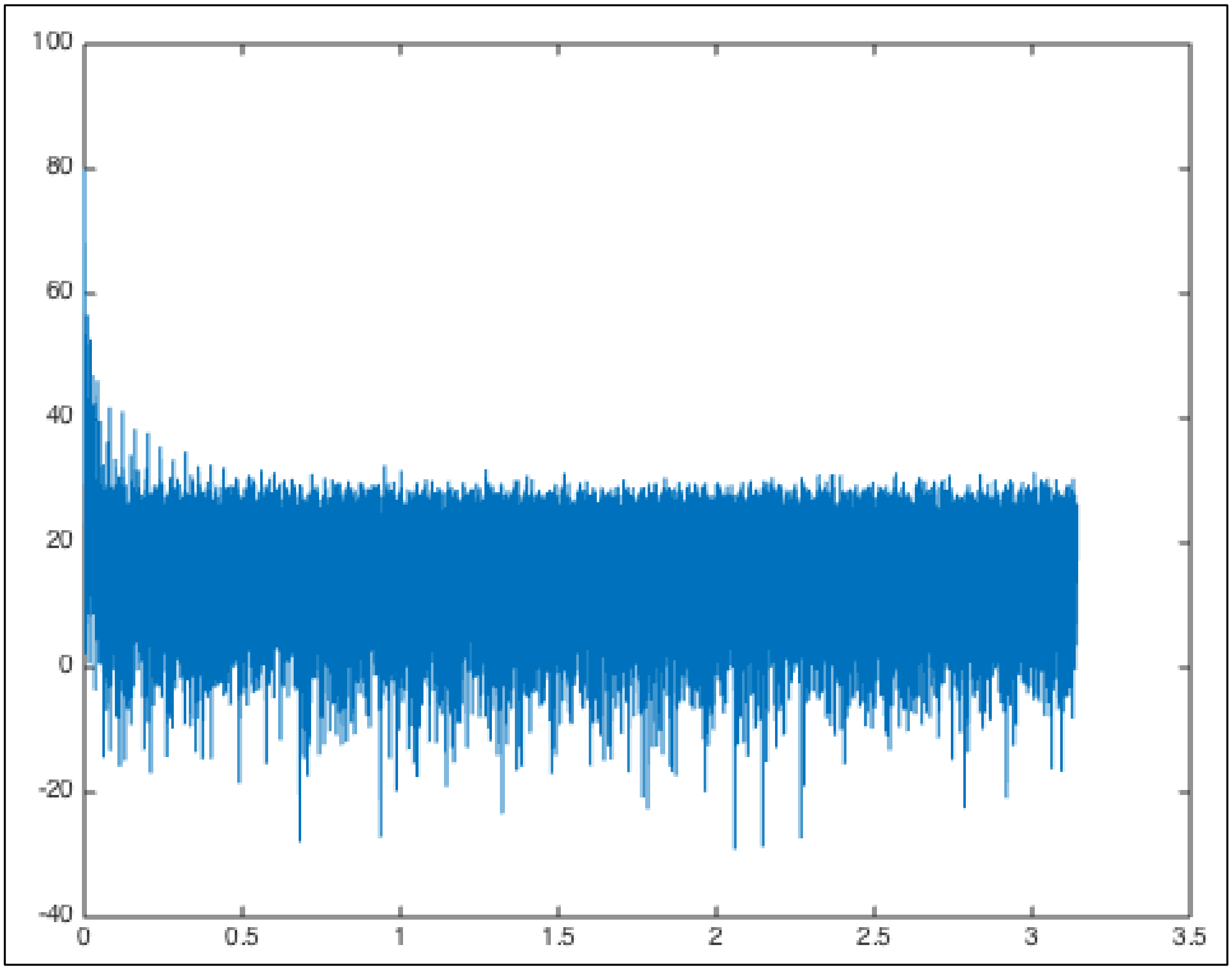
Following is the power spectrum density estimate plot for image 1 (PA01_Image1.tiff)



Following is the normal probability plot for image 2 (PA01_Image2.tiff submitted on the Box):



Following is the power spectrum density estimate plot for image 2 (PA01_Image2.tiff)



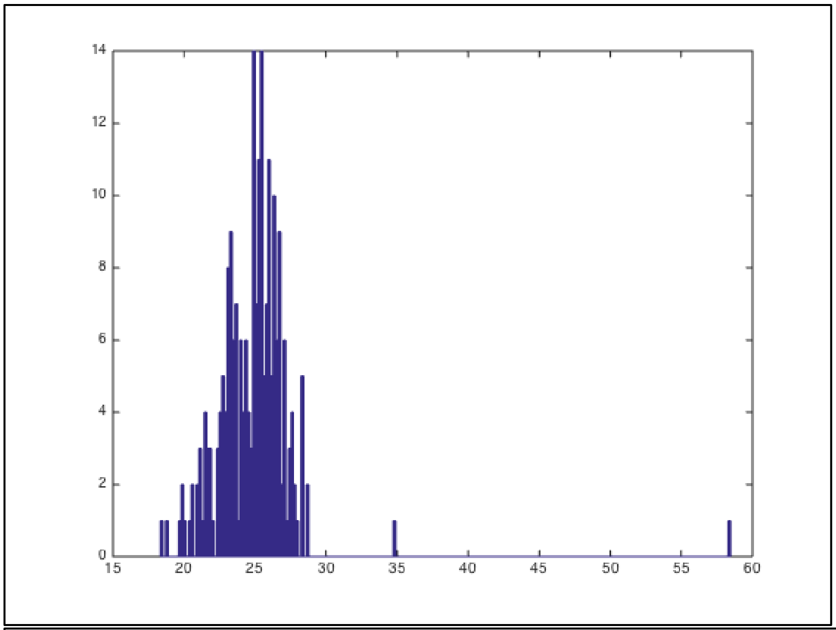
From the density plots for Image 1, it can be observed that the image 1 isn't a white noise, as the power density plot shows a decreasing trend and also the slope of the power density estimate plot is around -3.042377. This also confirms with the decreasing trend and so image 1 is not white noise.

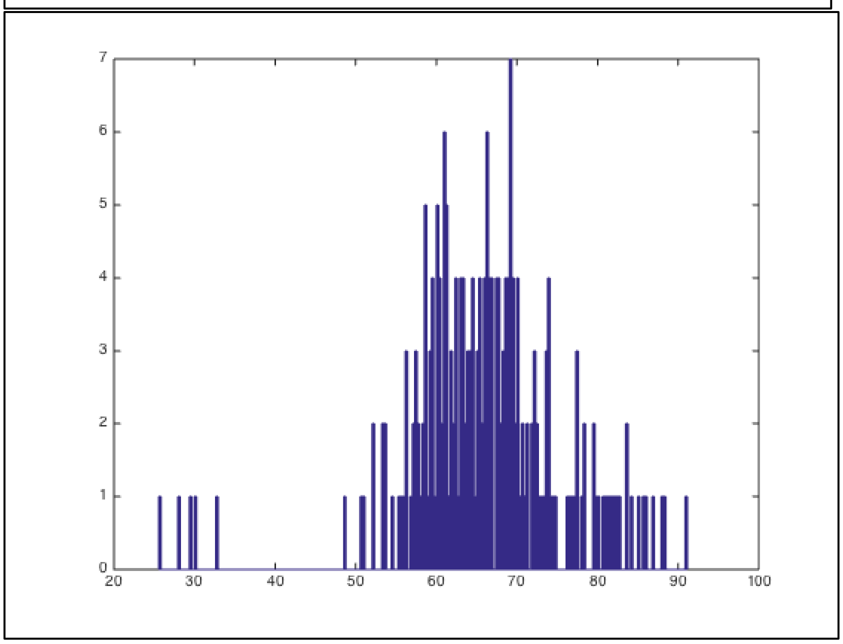
From the density plots for Image 2, it can be observed that the image 2 is a white noise, as the power density estimate plot shows a constant trend with small deviations. Also the slope of the power density estimate plot is around -0.477455. This confirms that the image 2 is a white noise.

Code submitted: `q2PA.m`

Images submitted: `q2.2PA02normplot.tiff, q2.2PA02_powerspectrum.tiff, q2.2PA01powerspectrum.tiff, q2.2PA01normplot.tiff`

Q2.3: We implemented a function that analyses the background section that was cropped by Q2.2 and analyses the coefficient of variation for each picture and plots them to a histogram. This measure is the ratio of the standard deviation to the mean value of pixel illumination and is a common measure for the uniformity of a particular set of numbers. We would expect lower values for a more uniform illumination and higher for a more varied pattern. This function can be found in file `q2illumination.m` and receives the name of a folder that contains the files to be analyzed.



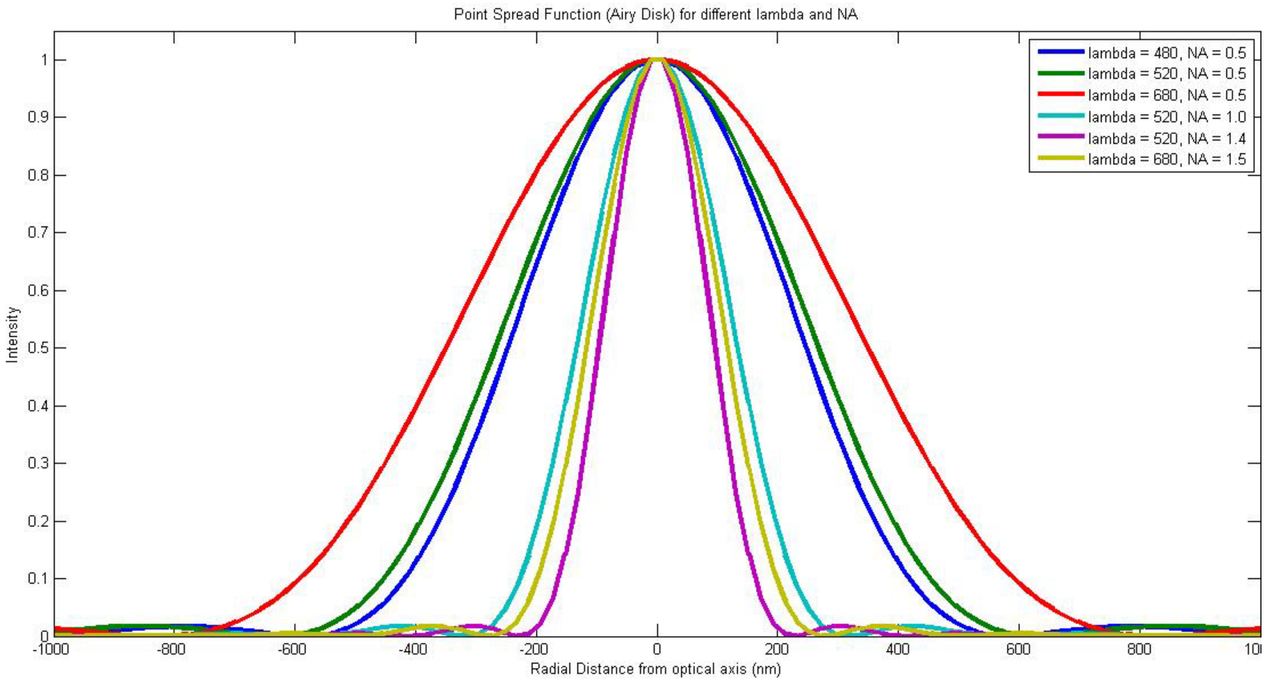


Above is the histogram of coefficients of variance for both the background images (above) and the foreground images (below). This shows that there is a fairly uniform illumination where there are no foreground features.

Q2.4: We have written two functions, `q2calibrationManual.m` and `q2calibrationAuto.m`, which, when given the calibration crosshatch and a specified magnification, will calculate the give number of pixels. The Manual version will ask the user to draw a box from the top of the crosshatch to the bottom. The Automatic version will find the highest and lowest corner pairs using the `Corner` function and calculates the distance in pixels between them. It then considers the magnification and gives an approximate length per pixel in micrometers.

III. Fitting Gaussian to an Airy Disk

The plot showing the airy disk for all combinations of λ and NA is shown below:



Formula used for calculation of airy disk function at a given distance q from optical axis:

$$x = \frac{2\pi q(NA)}{\lambda}$$

$$y = \left[\frac{2J_1(x)}{x}\right]^2$$

where J_1 is a Bessel function of first kind with order 1.

PART 1:

Table showing variation of airy disk radius corresponding to the wavelength and NA given:

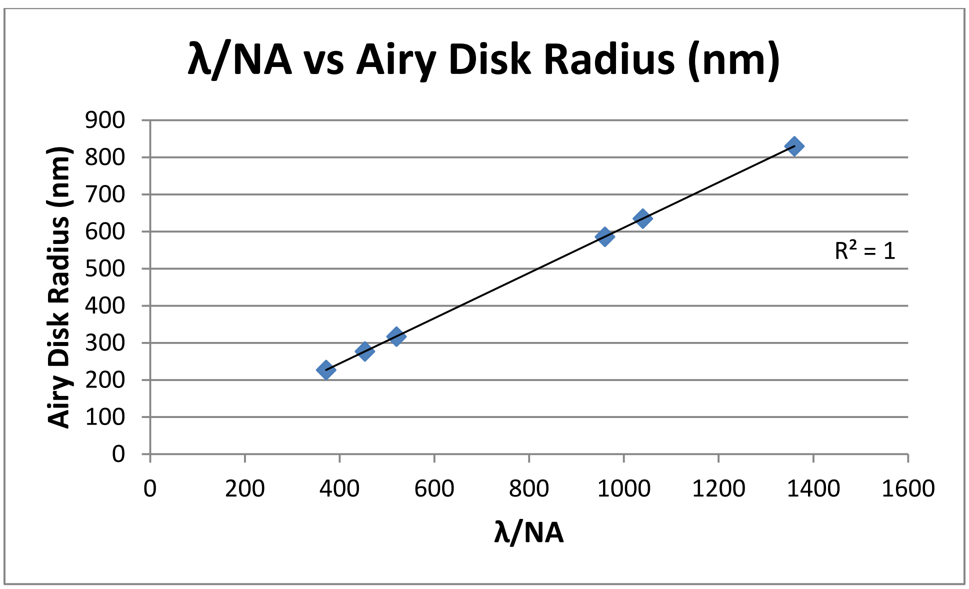
S.No.	λ(nm)	NA	Approximate Airy Disk radius (nm)
1	480	0.5	586
2	520	0.5	635
3	680	0.5	830
4	520	1.0	317
5	520	1.4	227
6	680	1.5	277

NOTE: The approximate airy disk radius was found out by trying to find the lowest q at which the airy disk function y was close to 0. This was done in MATLAB and the approximate airy disk radii given are our best guesses from the data we had.

We can see that for a fixed NA, the radius of the airy disk is directly proportional to the wavelength. This can be seen from observations 1-3 in the table where the NA is fixed at 0.5 and λ is increasing. As we increase λ, we see that the radius increases proportionally.

We can also see that for a fixed λ, the radius of the airy disk is inversely proportional to the numerical aperture NA. This can be seen from observations 2,4 and 5 where λ is fixed at 520 nm and NA is increasing. We see in this case that the airy disk radius is inversely proportional to NA and decreases with increasing NA.

From the plot below we can see that $\frac{\lambda}{NA}$ is fully correlated with the airy disk radius.



The slope of the line is approximately 0.61 and this follows the Rayleigh limit which says that the Airy disk radius $\approx \frac{0.61\lambda}{NA}$

Summary: The airy disk radius increases through observations 1-3 as the wavelength increases at fixed NA. The airy disk radius decreases through observations 2 and 4-5 as the NA increases at fixed λ. In observation 6, although the NA increases compared to observation 5, the huge increase in wavelength from 520 nm to 680 nm makes the radius slightly larger than for observation 5.

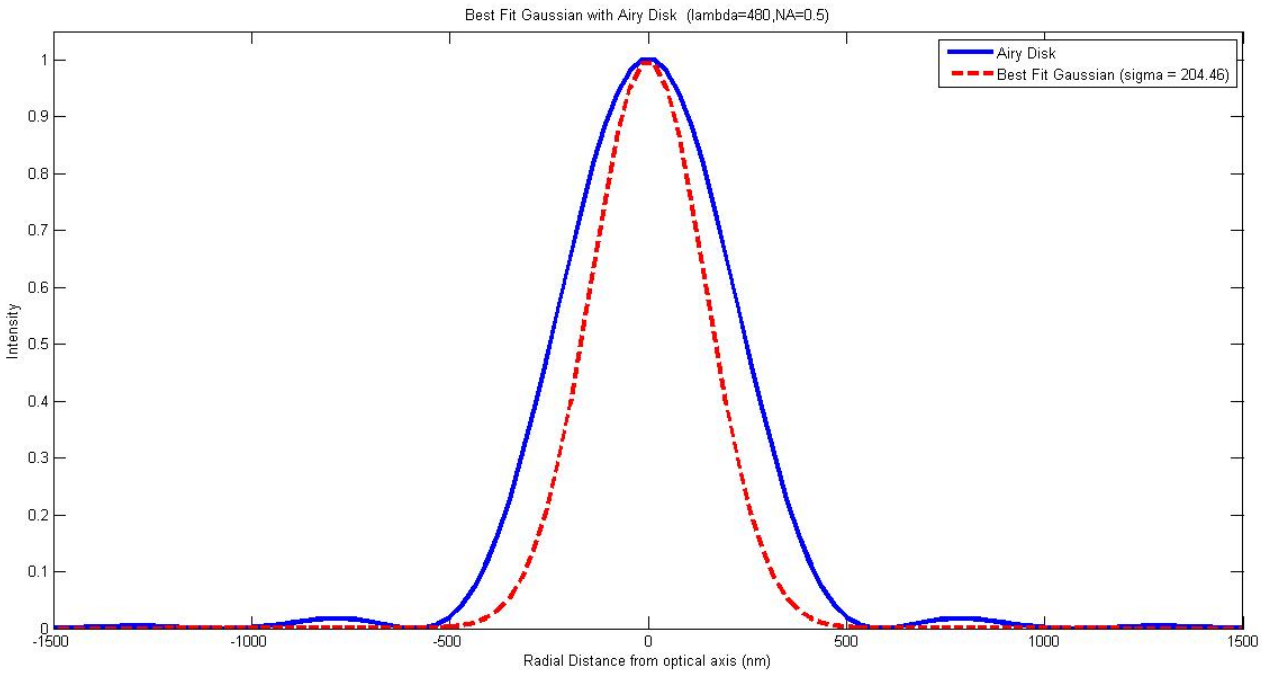
NOTE: Code submitted in file: `q3_plotAiryDisk.m.` This code generates the individual point spread function plots for a particular given λ and NA. This contains a function that also takes as input whether to fit a Gaussian or not as the third argument (required for part 2).

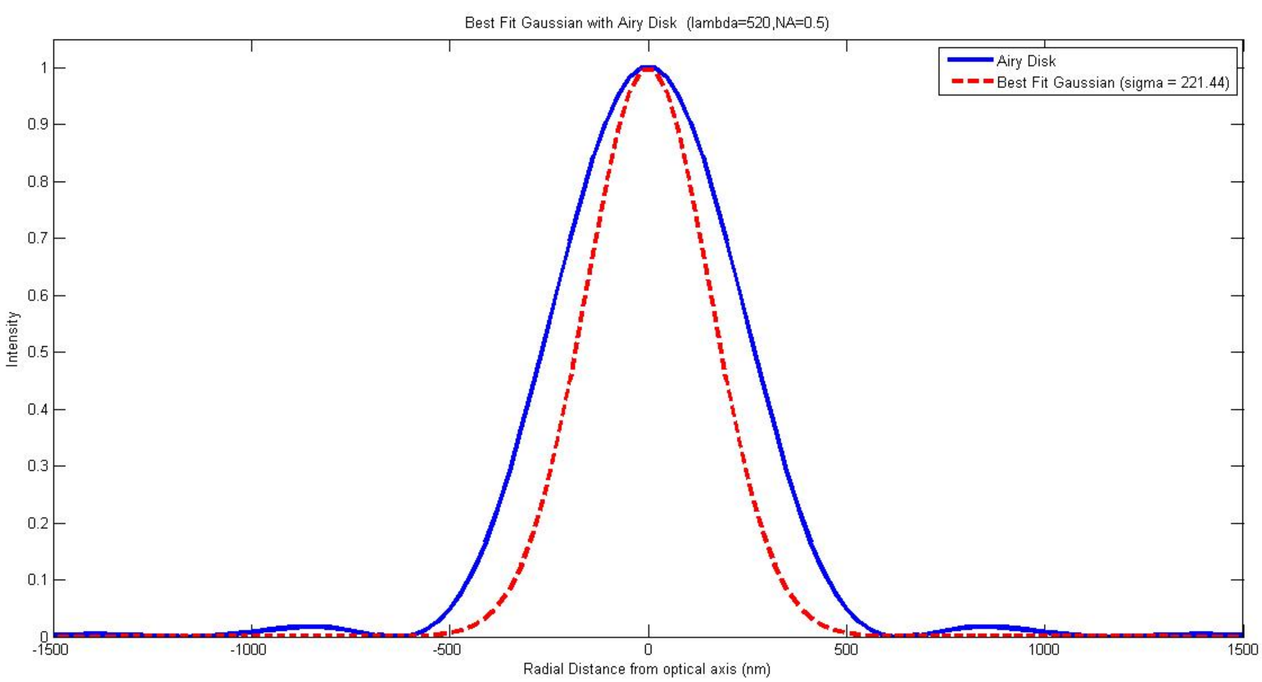
Figure showing all 6 airy disks in one graph submitted in file: `q3part1plot.fig` and `q3part1plot.jpg.` This figure is also shown above.

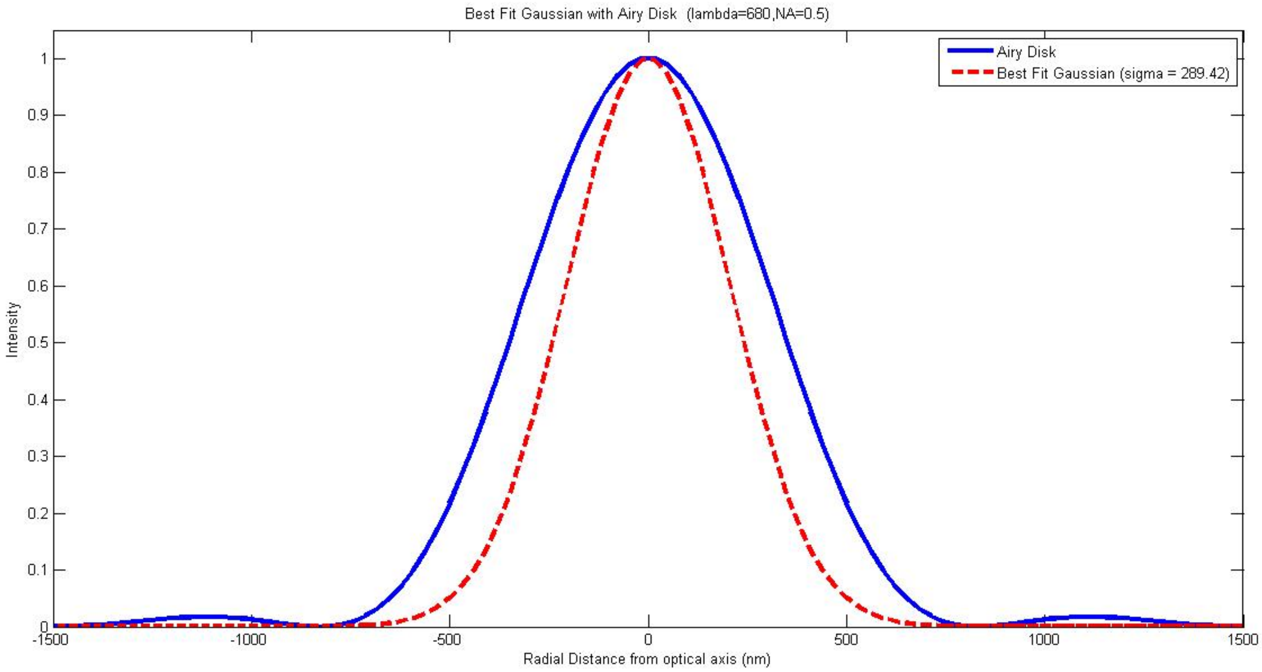
SIDE NOTE: Code to generate the combined airy disk plots (shown above) for the 6 observations is submitted in files: `q3part1_makeFigure.m` and `q3part1_getAiryDiskFunction.m`

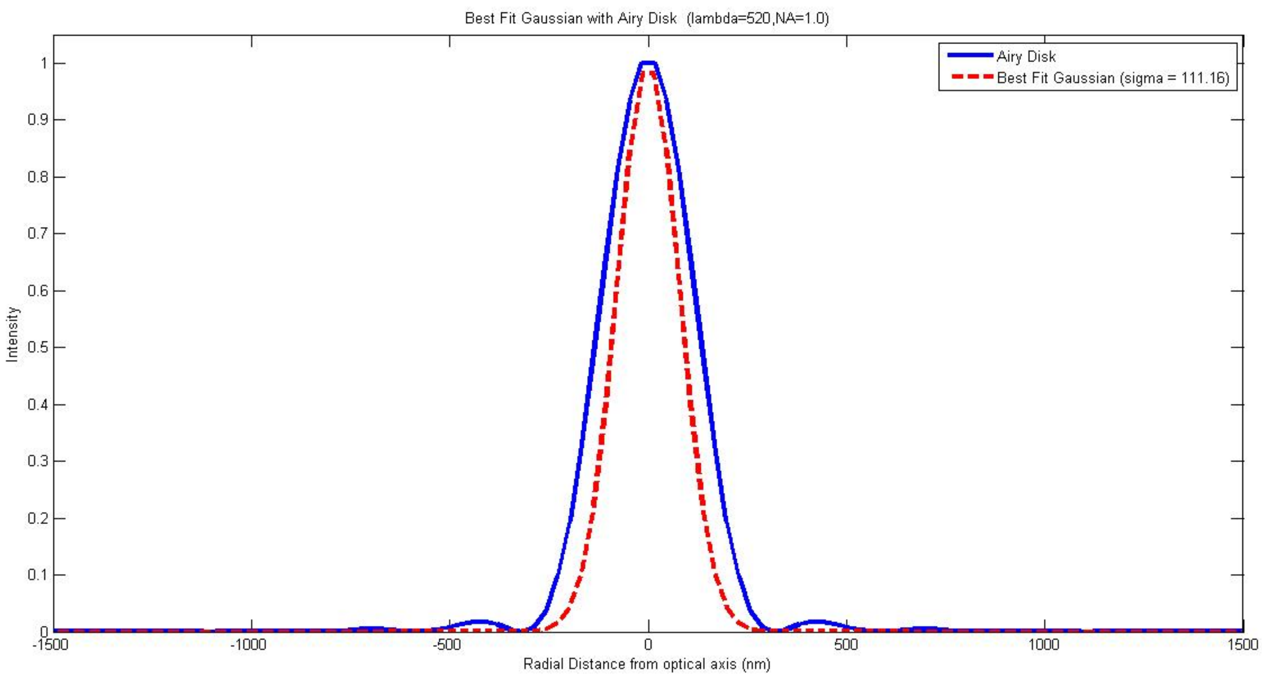
PART 2:

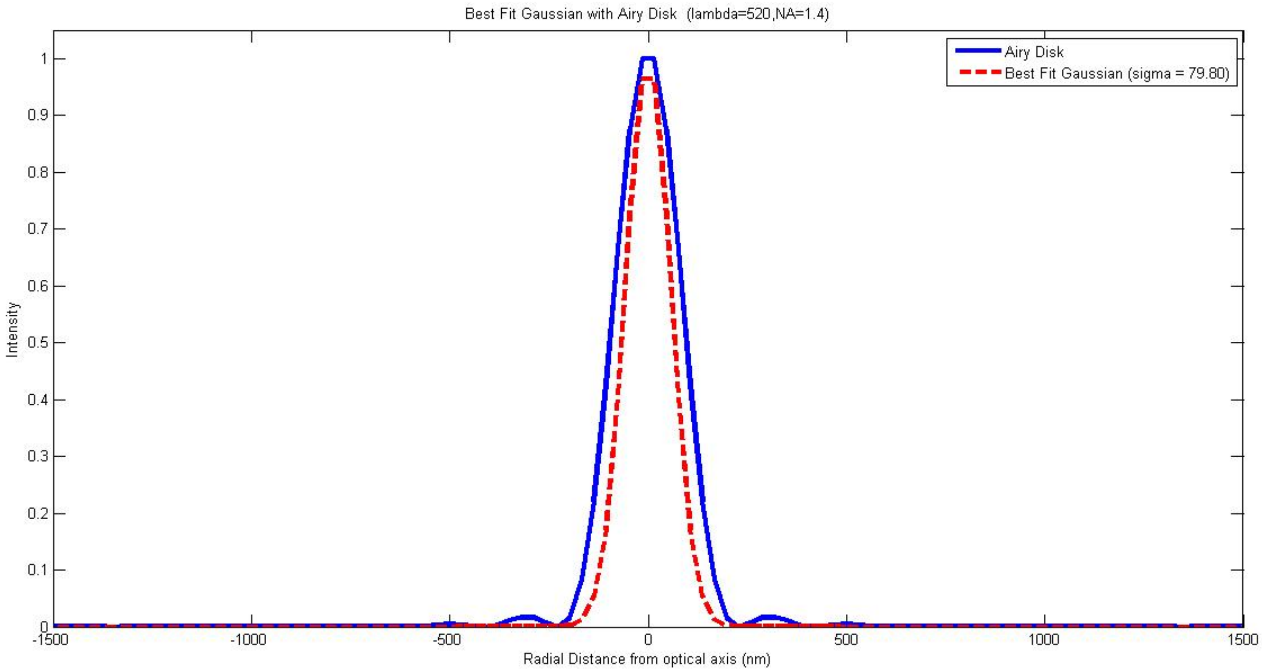
Individual plots showing fitted Gaussians for all 6 observations are shown below in order. The λ and NA used are mentioned in the title of the figures. The standard deviation of the fitted Gaussian is mentioned in the figure legend.






Best Fit Gaussian with Airy Disk (lambda=680,NA=0.5)
Airy Disk
Best Fit Gaussian (sigma = 289.42)
Intensity
Radial Distance from optical axis (nm)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
-1500
-1000
-500
0
500
1000
1500


Best Fit Gaussian with Airy Disk (lambda=520,NA=1.0)
Airy Disk
Best Fit Gaussian (sigma = 111.16)
Intensity
Radial Distance from optical axis (nm)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
-1500
-1000
-500
0
500
1000
1500



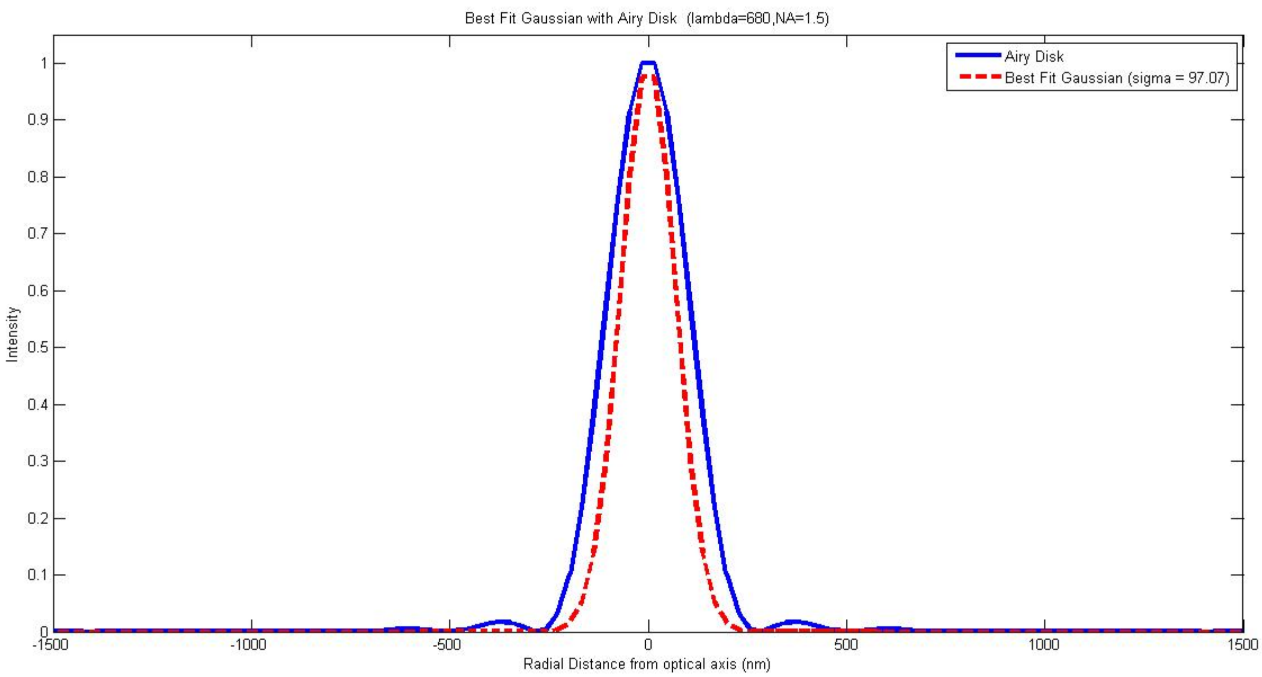
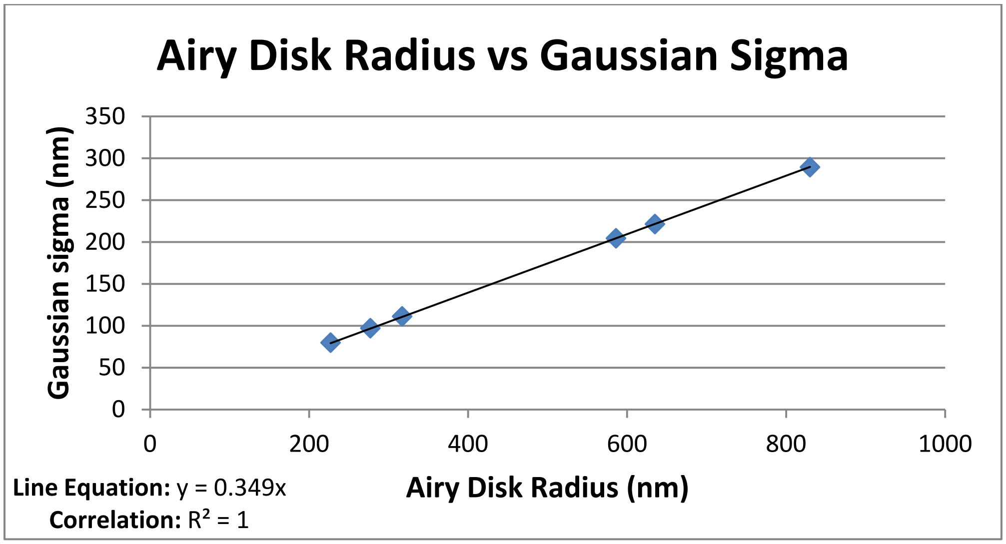


Table summarizing the gaussian standard deviation values found after fitting the Gaussians:

S.No.	λ(nm)	NA	Approximate Airy Disk radius (nm)	Gaussian σ (nm)	r/σ
1	480	0.5	586	204.46	2.866086
2	520	0.5	635	221.44	2.867594
3	680	0.5	830	289.42	2.867805
4	520	1.0	317	111.16	2.851745
5	520	1.4	227	79.80	2.844612
6	680	1.5	277	97.07	2.853611

We can see that the r/σ ratio remains almost constant at around 2.86.

The airy disk radius and the gaussian σ are completely correlated as can be seen from the plot below:



We observe that the airy disk radius is approximately $\frac{1}{0.349} = 2.865$ times the standard deviation of the fitted Gaussian i.e $r = 2.865\sigma$. The fact that we can fit a Gaussian to all these airy disks makes us come to the conclusion that the airy disk can be suitably approximated using a Gaussian kernel with $\sigma = 0.349r$ where r is the airy disk radius.

NOTE: It was assumed that the magnitude of the Gaussian was 1 since the airy disk was normalized before plotting. The Gaussian was fitted using the MATLAB $lsqnonlin$ function that minimizes the least square difference between the gaussian kernel and the point spread function (airy disk).

References

1. Geusebroek, J. M., Smeulders, A. W., & Van De Weijer, J. (2003). Fast anisotropic gauss filtering. *Image Processing, IEEE Transactions on*, *12*(8), 938-943.
2. Moreno, I. (2010). Illumination uniformity assessment based on human vision.*Optics letters*, *35*(23), 4030-4032.