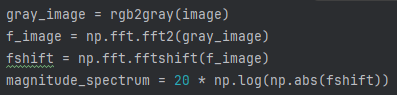
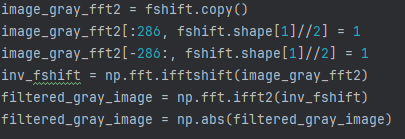
# **Manual:**

## **Fourier:**

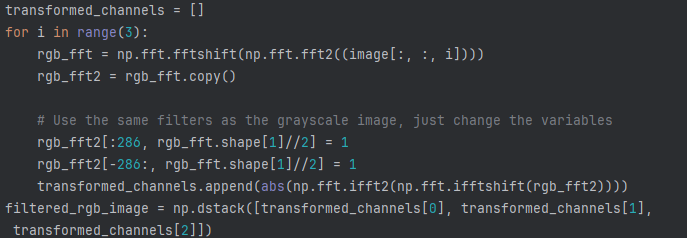
1. The screenshot below shows transforming the lung scan image from rgb to grayscale, then we apply the 2d fourier transformation using the method in numpy which takes in the image as a 2d array, then we shift the array so that it is centered then taking transformed image into a logarithmic scale to enhance the spectrum visualisation.

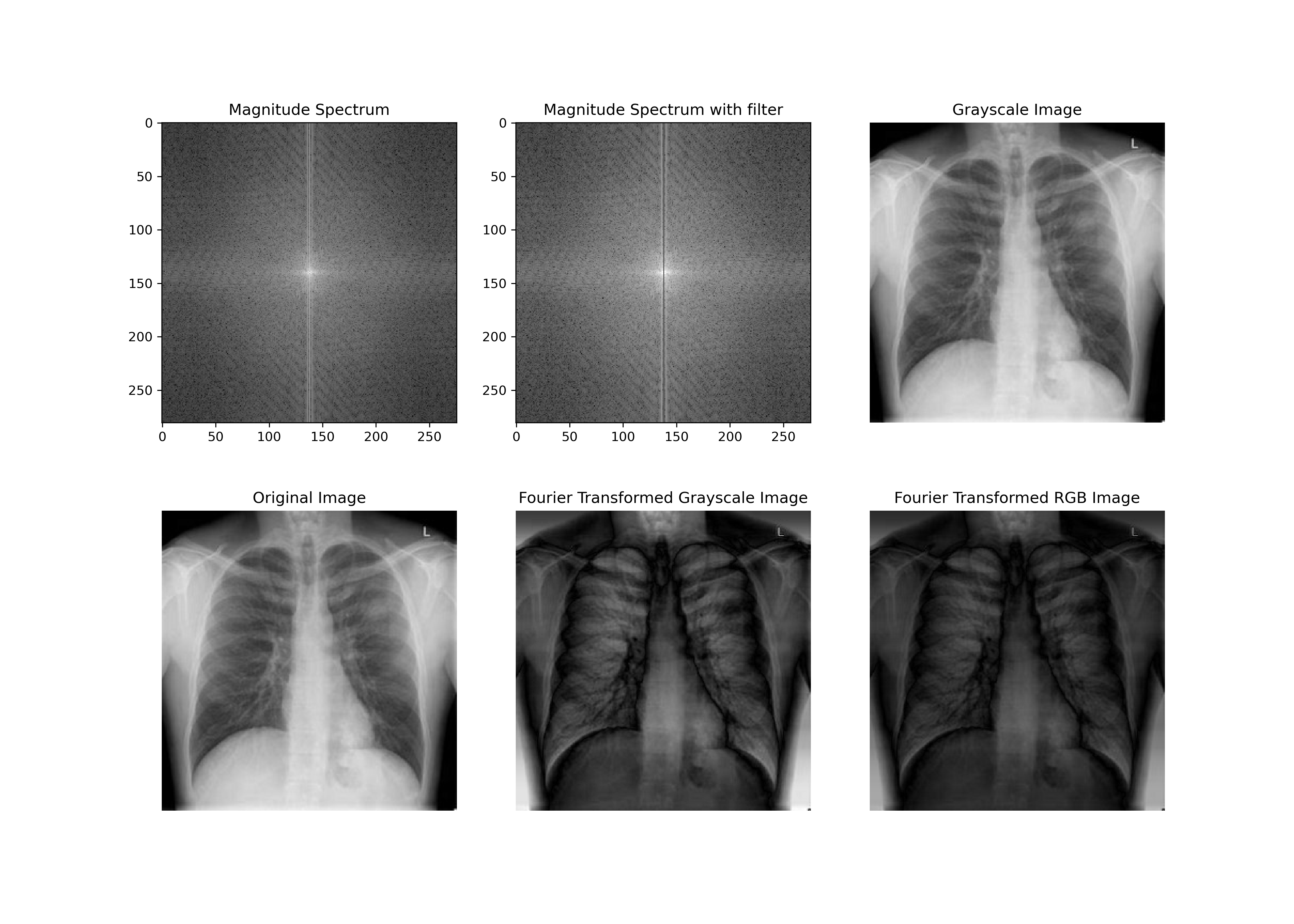


1. Then the tranformation applied was to remove a vertical line from the center (in this image the center is 286) from the image in the frequency domain then we just inverse the image back to the spatial domain using the ifft from np.fft and take absolute value to eliminate any negative values.



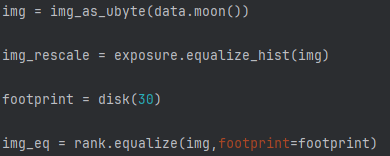
1. Then we do the exact same filter but on rgb image, the only difference is that in the beginning we will separate each channel, shift them to the frequency domain then apply the transformation on them then stack them on top of each other after the transformation.

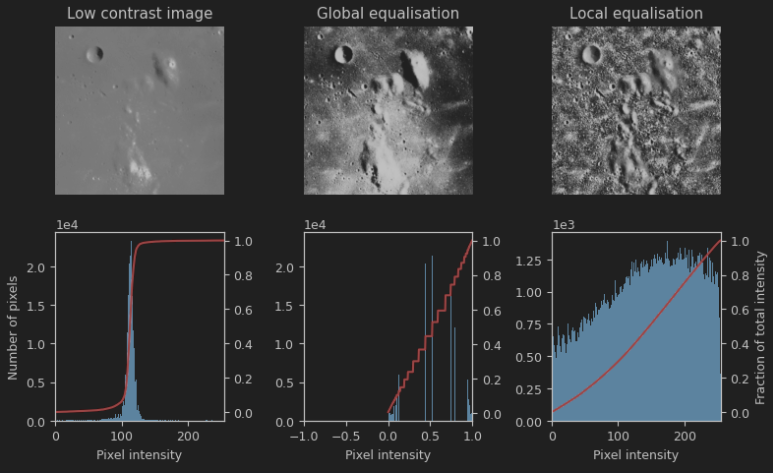




## **Equalization:**

1. You convert the image to the ubyte format which is basically stands for unassigned 8-bit dtype (very common when processing an image), then simply using the equalise\_hist method from skimage.exposure which returns the histogram of the image provided. The using a footprint (this case it is disk(30)) we equalise the histogram.





# Assignment:

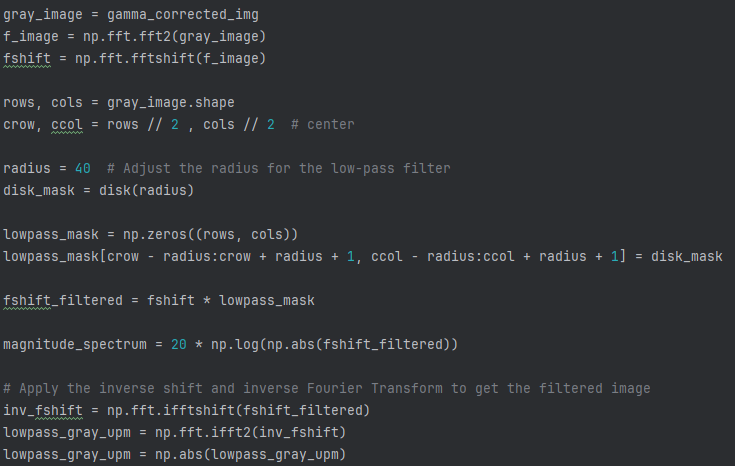
1. After reading the image and applying histogram equalization (like the example above) we apply gamma correction to decrease the brightness of the image. Value chosen was 1.5 to dim the image slightly as it was too bright.



1. We save the enhanced image using plt.imsave with gray cmap



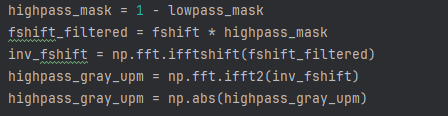
1. We first create a low pass mask by first extracting the midpoint of the image both in the x and y. then we decide the radius (in this case it is 40 as it gave the best results subjectively) then we fill the middle part of the disk with ones after filling the rest with zeroes. Then we multiply the fourier transformed image with the lowpass mask and the rest are the same steps as the manual.



1. We save the low pass filtered image using plt.imsave with gray cmap



1. High pass mask will just be 1 – lowpass mask then the same steps as before



1. We save the high pass filtered image using plt.imsave with gray cmap

