University of Waterloo

Faculty of Engineering

# SYDE575 Lab 3: Filter Design and Image Restoration in Frequency Domain

Prepared By
Sarthak Tamboli | 20665466
Sheen Thusoo | 20728766

Date: 29 October, 2021
Prepared for: Professor David A. Clausi

## 1. Introduction

The purpose of this lab is to study Fourier analysis and image restoration concepts in the frequency domain as well as filter design. The images used in this lab are *lena.tiff*, *camerman.tif*, *frequnoisy.tif*. In Section 2, the characteristics of images in the frequency domain are studied. In Section 3, noise reduction techniques based on frequency domain filtering as well as the effect of filter parameters on image quality is studied. In Section 4, a frequency domain filter that filters out noise is designed. All of the code for this lab was written in MATLAB and can be found in Appendix A.

## 2. Fourier Analysis

In Section 2, we studied the characteristics of an image in the frequency domain. We created a test image and rotated it by 45 degrees; we plotted the Fourier spectra of both the original and rotated image. The original image is seen in Figure 1 and its Fourier spectrum can be seen in Figure 2. The rotated image and its Fourier spectrum can be seen in Figure 3 and 4, respectively. The code for this section can be seen in Appendix A, Figure A.1.1 and Figure A.1.2.

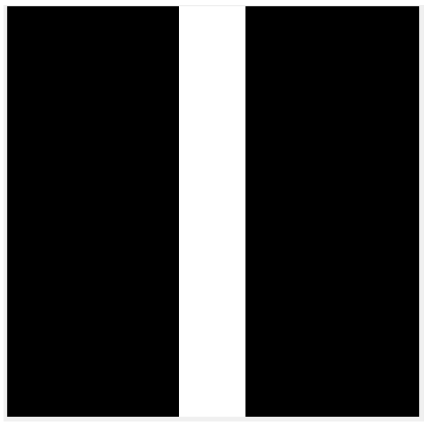


Figure 1: Test Image

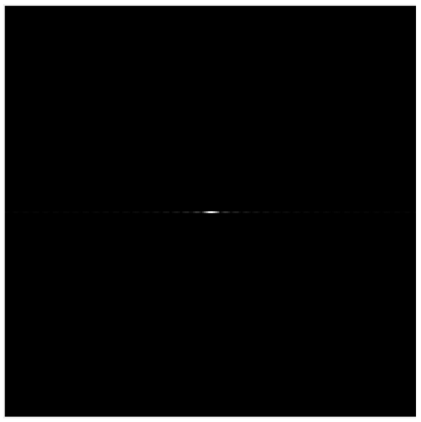


Figure 2: Fourier Spectrum of Test Image

**1.** In Figure 2, the distribution of the energy of the Fourier spectrum can be seen; each point shows a particular frequency in the spatial domain image. Specifically, the amplitude represents the prevalence of the frequencies and the angle of the spectrum represents the orientation of the frequencies. In Figure 2, it is clear that the largest energy distribution is the center of the spectrum, along the horizontal direction. This is because, in the test image, the grey level intensities change as you move from left to right - from black to white and to black again. Thus, the energy distribution of the image is changing in the horizontal direction. The center of the Fourier spectrum corresponds to coarse, low frequency details in the image. Since the original image does not have any fine details, its energy is in the center of the spectrum. This makes sense since the image is just one white rectangle in the middle of the image with a black background; there are no fine details here (only 2 edges).

2. From the Fourier Spectrum, it can be inferred that the test image does not have a change of grey level intensities in the vertical direction; the change of intensities only occur in the horizontal direction. This can be inferred because the spectrum is only along the horizontal direction implying that there is only a change of intensities along this direction. Another inference that can be made is that the test image has mostly low frequency (coarse) details. This can be inferred because the energy is concentrated in the center of the spectrum; it is much brighter than the outer parts of Figure 2. This is true since the image has more coarse details and only has two vertical edges. Thus, the middle of the spectrum image is white, and its outer regions are black.

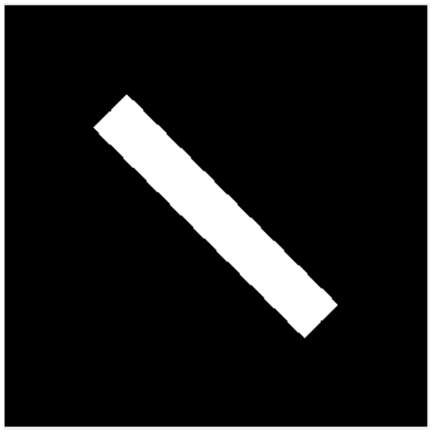


Figure 3: Rotated Test Image

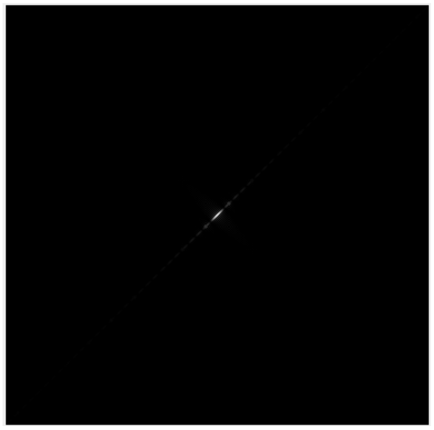


Figure 4: Fourier Spectrum of Rotated Test Image

3. The Fourier spectrum is now in a diagonal orientation, going from bottom left to top right. It seems that the spectrum has also been rotated 45 degrees from the original in Figure 2. The original spectrum was a straight horizontal line, however this new spectrum, Figure 4, is in a diagonal orientation. This makes sense because the intensities in the original image, Figure 3, are no longer only changing in the horizontal direction, they are now changing in both the vertical and horizontal directions. The test image now also introduces two new edges on the top and bottom of the white rectangle. Since there are 4 edges in the image now, introducing more high frequency components, the spectrum also seems to be wider (further spread from the center) than the previous.

4. Based on the Fourier spectra of both the original and rotated images, it can be concluded that the Fourier spectrum represents the frequencies in an image along with their orientations. We can see this because as we rotated the original image 45 degrees, the Fourier spectrum also rotated as well. Hence, the change of the orientation of frequencies in the spatial domain image affects the orientation of its Fourier spectrum. The spectrum can also give information about the level of

details in an image, that being high frequency details (edges, noise etc.) or low frequency details (coarse details). If an image is more coarse, with low frequency components, its Fourier spectrum will be brighter in its center.

Next, we studied the contribution of Fourier amplitude and phase to an underlying image, the Lena image. The Lena image was loaded and converted to grayscale. Then, its amplitude and phase was computed. The amplitude component $A$ is the magnitude of the Fourier complex component and the phase component θis computed by dividing the Fourier Component $F(\omega)$ by the amplitude A as

$$F(\omega) = A \times (cos(\theta) + jsin(\theta)) \quad (1)$$

The inverse Fourier transform was then applied to the amplitude and phase component. The original image can be seen in Figure 5, the image reconstructed using only amplitude can be seen in Figure 6, and the image reconstructed using only phase can be seen in Figure 7.

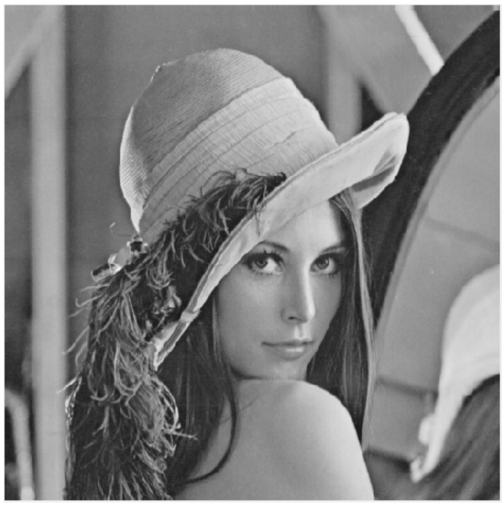


Figure 5: Original Lena Image

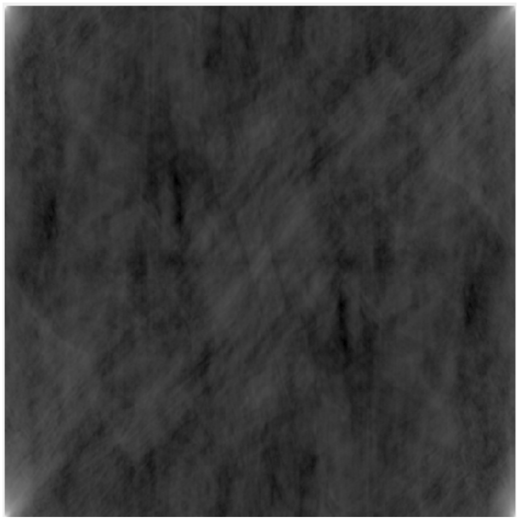


Figure 6: Lena Image Reconstructed with Amplitude

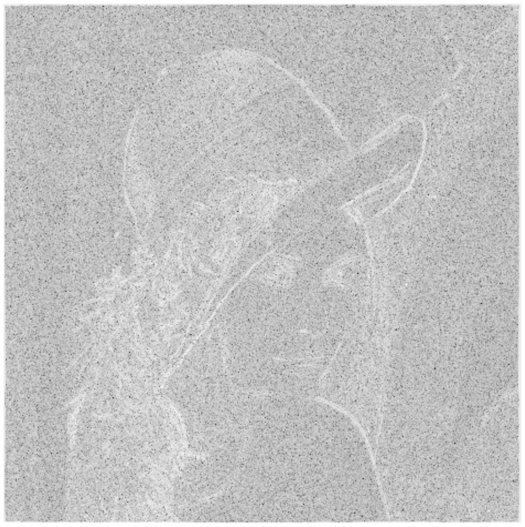


Figure 7: Lena Image Reconstructed with Phase

5. The reconstructed image from the amplitude has different grey level intensities that do not resemble any shape similar to that in the original image. It has a few different darker and lighter grey level intensities. The image intensities seem to be mirrored along a diagonal line from the bottom left to the top right. This is because the amplitude component has no phase so it does not give any information about the orientations of frequencies in the original image. The amplitude component captures information about how often a frequency occurs in the image. The sinusoids in the original image have zero phase in the image so there is not any clear resemblance to the original Lena image.

6. The reconstructed image from the phase looks like an outline of the Lena image with the edges highlighted. The brightest intensities (whiter pixels) seem to occur at the edges in the

image. The phase component captures the orientations of frequencies in the spatial domain image so it makes sense that we are able to discern the shape or structure of the image. Sinusoids have the same phase where there are edges in the original image so we are able to see edges clearly. The phase component clearly does a better job at reconstructing the original image as it retains the structural information when compared to the amplitude component. Edge and structural information is very important to the human visual system for discerning an image.

## 3. Noise Reduction in Frequency Domain

In Section 3, we explored different noise reduction techniques based on domain filtering coupled with the impact to image quality by filter parameters. Three low-pass filters were applied, specifically one with a cut-off radius of 60 and 20 and a Gaussian filter. The resulting Fourier spectra are analyzed, along with the PSNR values and their overall performance on noise reduction. The code for this section can be found in Appendix A, Figure A.2.1 to Figure A.2.4, and Figure A.3.1.

Table 1: PSNR Values for Denoised Images with specific filters

| **Filter Applied** | **PSNR Value** |
|---|---|
| Low-pass filter w/ cut-off radius 60 | 27.8862420905315 |
| Low-pass filter w/ cut-off radius 20 | 23.1180699212726 |
| Gaussian low-pass filter w/ stddev 60, normalized on max(kernel value) | 29.5305060285601 |

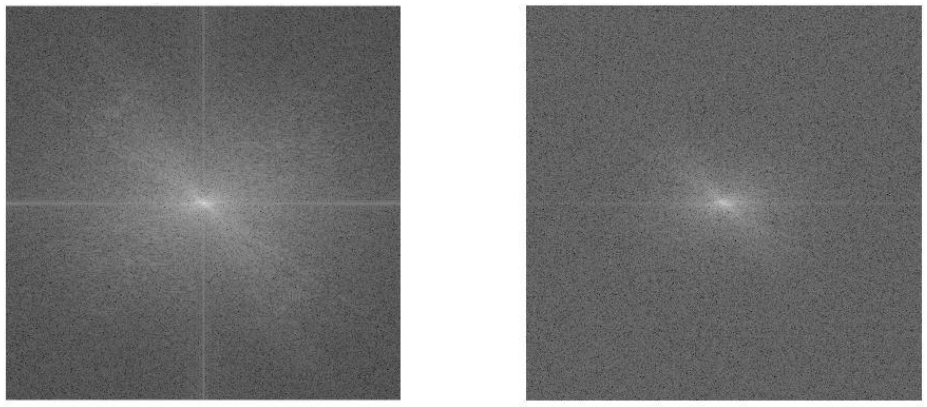


Figure 8: Log Fourier Spectra of original (left) and noisy (right) Lena image

7. A few differences can be noted when comparing the two Fourier spectra in Figure 8. The noisy Fourier spectrum is brighter than the original which can be attributed to the additive Gaussian noise model. This implies that the magnitudes of the sinusoids in the frequency domain are higher in the noisy image as compared to the original image, more evident in the higher

frequencies (away from the origin). Furthermore, due to the noise introduced, the 3 distinct bands centered in the original spectrum are more difficult to distinguish in the noisy spectrum. Initially, it can be seen that the majority of the energy for the Lena image occurs around the origin at low frequencies. After adding the noise, there is a more noticeable effect in the higher frequencies, where there was lesser energy, originally.

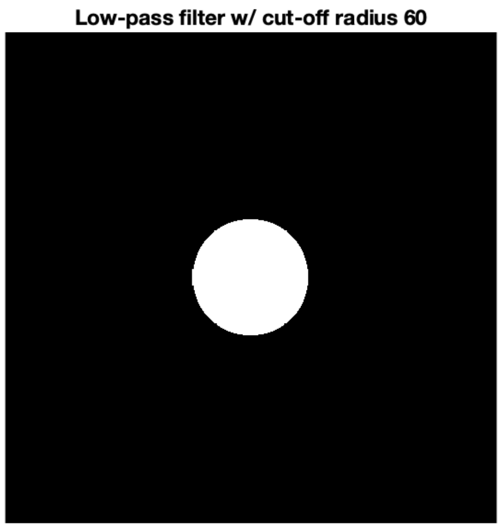


Figure 9: Low-pass Filter with radius of 60

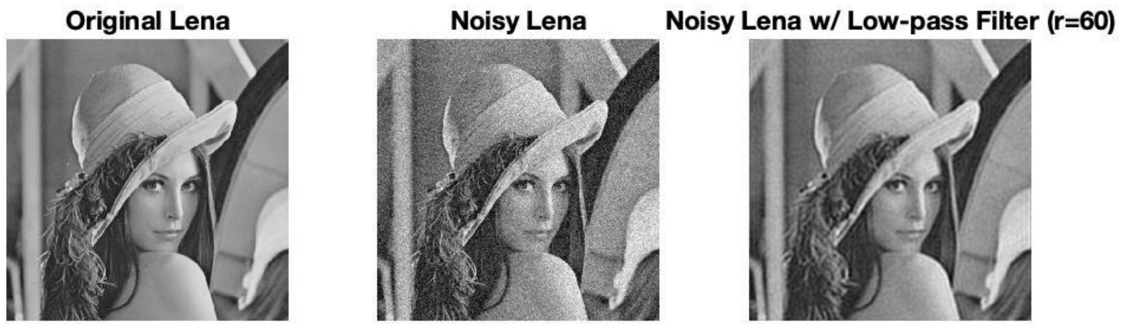


Figure 10: Original (left), Noisy (middle), Filtered Lena w/ Low-pass Filter, r=60 (right)

8. Comparing the denoised image to the original and the noisy image we can say that the majority of noise introduced has been eliminated with a minute blur effect added resulting in a much smoother image. In the frequency domain, high frequency noise is generally introduced by the additive Gaussian noise model. An ideal low-pass filter removes all frequencies that exist outside the filter radius. Only the frequencies that lie within the filter radius are inverted back to the spatial domain. We know that in the frequency domain, the high frequencies exist in the outer edges of the Fourier spectrum, away from the origin. So with a radius of 60, we are effectively removing the noise added at the higher frequencies and inverting the image back to the spatial domain only with the low-frequency information. This results in a loss of high-frequency detailing such as edges and noise however, it is replaced with low-frequency data which is why there is blurring and blurry lines occurring at places where there used to be strong edges. However, it does remove much of the noise that we see in the noisy image.

9. Both denoised images (r=60 and r=20) have unusual blurring that occurs at locations where edges exist in the original image. The blurring is in the form of lines that take the shape of the edge boundaries on both sides of the original edge. It is most prominent in the stronger, more distinct edges. After some research, it was concluded that this blurring is actually a ringing artifact from the Gibbs phenomenon. Since the ideal low-pass filters eliminate the high-frequencies, it increases the number of discontinuities that occur in the image which creates ringing artifacts at locations where strong edges exist. When abrupt discontinuities occur and the signal jumps from value to value, ringing occurs. In the original image, in the frequency domain, we have sinusoids with various amplitudes and frequencies. Inverting the image to reconstruct the spatial domain image is done with fewer frequency terms than the original image, producing the rippling effect at previous locations of strong edges.

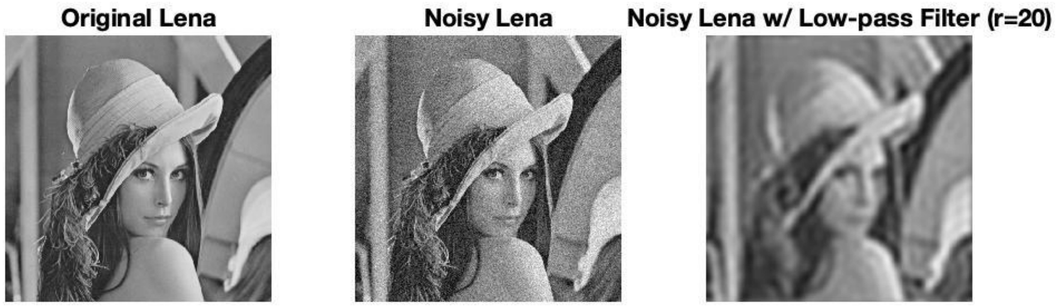


Figure 11: Original (left), Noisy (middle), Filtered Lena w/ Low-pass Filter, r=20 (right)

10. Compared to the denoised image using a cut-off radius of 60, this denoised image is far more blurry and the ringing artifacts are also more significant. This reduces the image quality. This is because with a lower cut-off radius, there is a higher amount of frequencies being filtered out of the image. In the denoised image using the cut-off radius of 60, only the highest frequencies components are being filtered; however, in the denoised image using the cut-off radius of 20, lower frequencies start being filtered out as well. Thus, its denoised image appears very blurry as

more of the fine details (higher frequency components) are being eliminated. These high frequency components are important to the human visual system to be able to discern images. All of this information is reflected in the PSNR values. The PSNR value of the filter with a cut-off radius of 60 is around 27.89 whereas the PSNR value of the filter with a cut-off radius of 20 is around 23.12. Thus, the image quality of the denoised image using the cut-off radius of 60 is higher (higher PSNR value) than that of the denoised image using the cut-off radius of 20 (lower PSNR value).

11. Based on the results, we can conclude that a smaller cut-off radius in the ideal low-pass filter results in a denoised image that has more noise reduction but is much blurrier. This is because the lower cut-off radius eliminates high frequency components in an image which causes a ringing effect. As we reduce the cut-off radius in the filter, we reduce more noise in the image but we also increase the amount of frequencies that are removed in the denoised image. Therefore, the trade-off is that although we reduce the noise in the image, we also remove high frequency components in the image which could be crucial to its structure (i.e. edges).

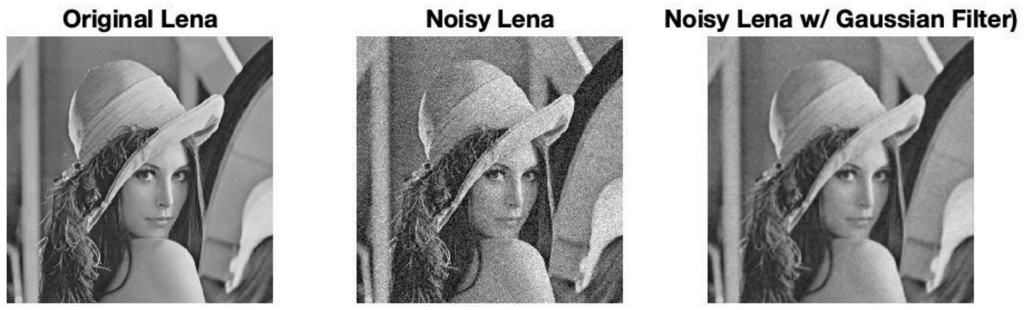


Figure 12: Original (left), Noisy (middle), Filtered Lena w/ Gaussian Filter (right)

12. Compared to the denoised image using the ideal low-pass filter with a cut-off radius of 20, the denoised image using the Gaussian filter visually has a higher image quality. It is much more sharp and we are able to discern different components in the image easily. Compared to the denoised image using the ideal low-pass filter with a cut-off radius of 60, the denoised image using the Gaussian filter visually looks very similar, but slightly smoother. The PSNR values reflect these visual examinations as the PSNR value for the ideal low-pass filter with a cut-off radius is around 27.89 which is similar but slightly lower than the PSNR value for the Gaussian filter which is around 29.53. The PSNR value for the ideal low-pass filter with a cut-off radius of 20 is around 23.12 which is much lower than both the previous two filters mentioned. This makes sense as its image quality is much worse. All in all, these PSNR values make sense since the Gaussian filter denoised image had the highest quality and the highest PSNR, the low-pass filter with a cut-off radius of 60 denoised image had the second highest image quality and second highest PSNR value, and the low-pass filter with a cut-off radius of 20 denoised image had the worst image quality and the lowest PSNR value.

The benefit of using a Gaussian filter is that it removes the ringing artifacts of the image which were present in the images denoised using the ideal low-pass filters. The ideal low-pass filters completely remove frequency components that are beyond the cut-off radius whereas the Gaussian filter weighs these components lower. Thus, the noise is reduced and the high frequency components still exist in the image.

## 4. Filter Design

In section 4, the *frequnoisy* image and its Fourier spectrum was examined. There were peaks in the spectrum corresponding to the periodic noise source which had been added to the original image. Here, we designed a frequency domain filter which filters out this noise. The code for this section is shown in Appendix A, Figures A.4.1 and A.4.2.

To begin, we plotted the original image to view the noise components in it; we see in Figure 13 that there has been a periodic noise source added to the image.

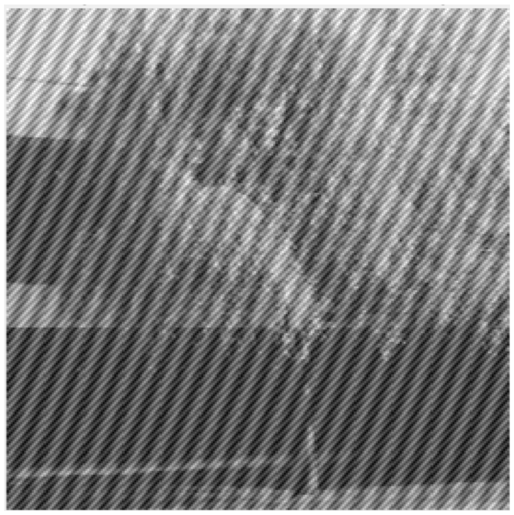


Figure 13: Original Frequnoisy image

Next, we examined the image's Fourier spectrum. This was done by applying the Fourier transform on the image and shifting the zero-frequency component to the center. Next, we applied the absolute and log function to this signal. The Fourier spectrum of the image is shown in Figure 14 below.

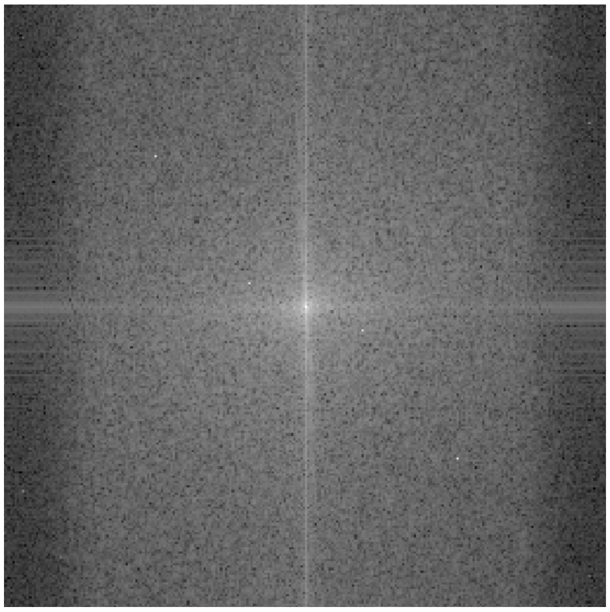


Figure 14: Log of Fourier Spectrum of Frequnoisy Image

In Figure 14, we can see that the brightest area occurs at the center of the image with four bright lines going across the center of the image, splitting it into four parts. The rest of the image has various grey level intensities however the outer left and right borders have darker intensities. Each point in this image represents the occurrence of a particular frequency along with its orientation. In Figure 14, we see four pixels with very bright intensities that seem like anomalies (near the center along almost a diagonal line). These pixels represent the periodic noise that has been applied to the original image. In order to remove this noise, we must locate these pixels and modify their pixel intensity to zero. Here, we use the *imtool* in MATLAB to identify the coordinates of the pixels. Using this tool, we found that the coordinates of these pixels are:

- (65, 65)
- (119, 105)
- (139, 153)
- (193, 193)

Once these coordinates were set to an intensity value of zero, we shifted this new image back to the original transform output (undoing the result of *fftshift*) and performed the inverse Fourier transform. The result of these operations can be seen in Figure 15 below.

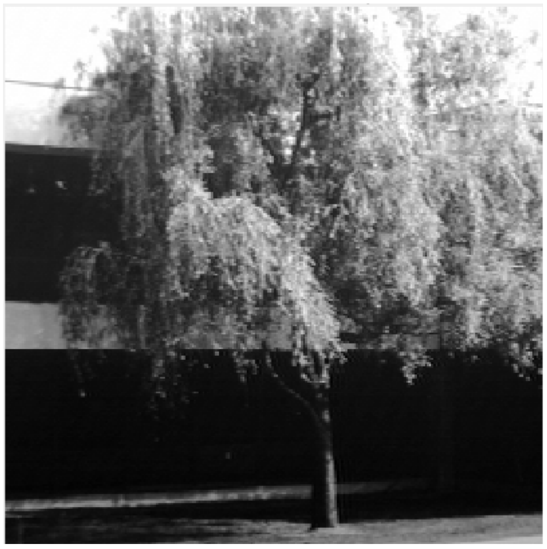


Figure 15: Filtered Frequnoisy Image

From Figure 15, we can see that the periodic noise source has been eliminated from the image. Thus, our frequency domain filter has been successful.

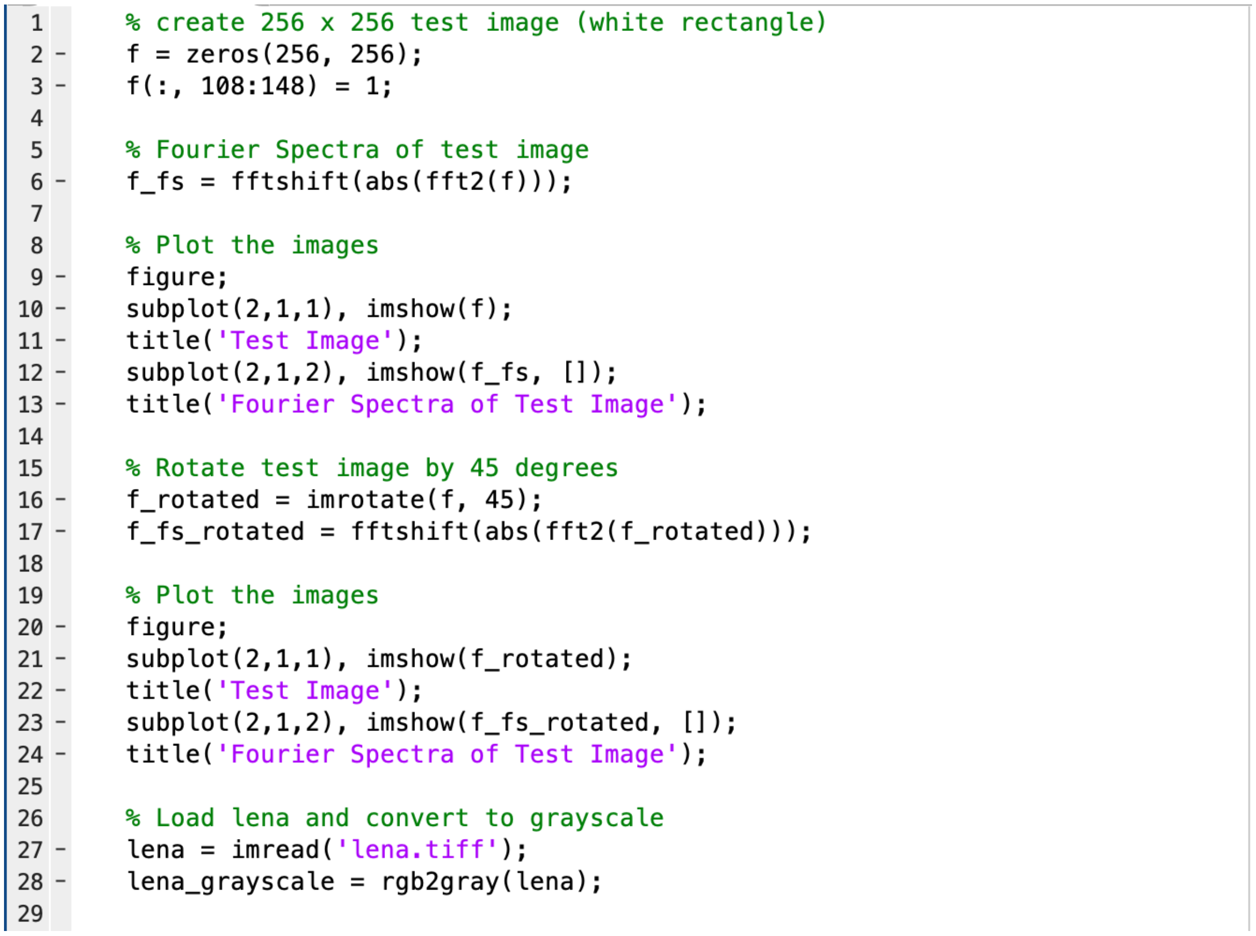
## 5. Conclusion

In conclusion, this lab involved studying images and image restoration in the Fourier domain. In Section 2, we looked at the characteristics of an image in the frequency domain using a test image and its rotated version. We plotted these two original images and their associated Fourier spectra. From this, it was found that the Fourier spectrum of an image represents the prevalence of frequencies as well as their orientation in the original image. Specifically, the amplitude represents how often a frequency occurs and the angle represents the orientation of the frequency. We also looked at reconstructing the Lena image using only amplitude and only phase. Reconstructing the image with only amplitude was not successful as it did not retain any structural information about the original image. This is because the amplitude only captures information about how often a frequency occurs; it does not contain any phase. Reconstructing the image with only phase was more successful as we could see the structural information of the image which is very important to the human visual system. This is because the phase captures the orientations of frequencies in the image. The sinusoids have the same phase in areas where there are edges and thus they can be seen clearly with brighter intensities.

In Section 3, three low-pass filters were introduced to the Lena image with an additive Gaussian noise model. The noise model introduced noise at high frequencies. An ideal low-pass filter with radius 60 and then 20 was used to denoise the noisy image. As previously explored in Section 3, the low-pass filter with radius 20 produced the best denoised image, in terms of PSNR, however, the resulting image was substantially affected by Gibbs phenomena adding ringing artifacts. A Gaussian filter was also used and it performed better at reducing the ringing artifacts, as this specific low-pass filter did not eliminate high-frequency components, instead it just reduced the weight of them. This resulted in a smoother image, with the highest PSNR value.

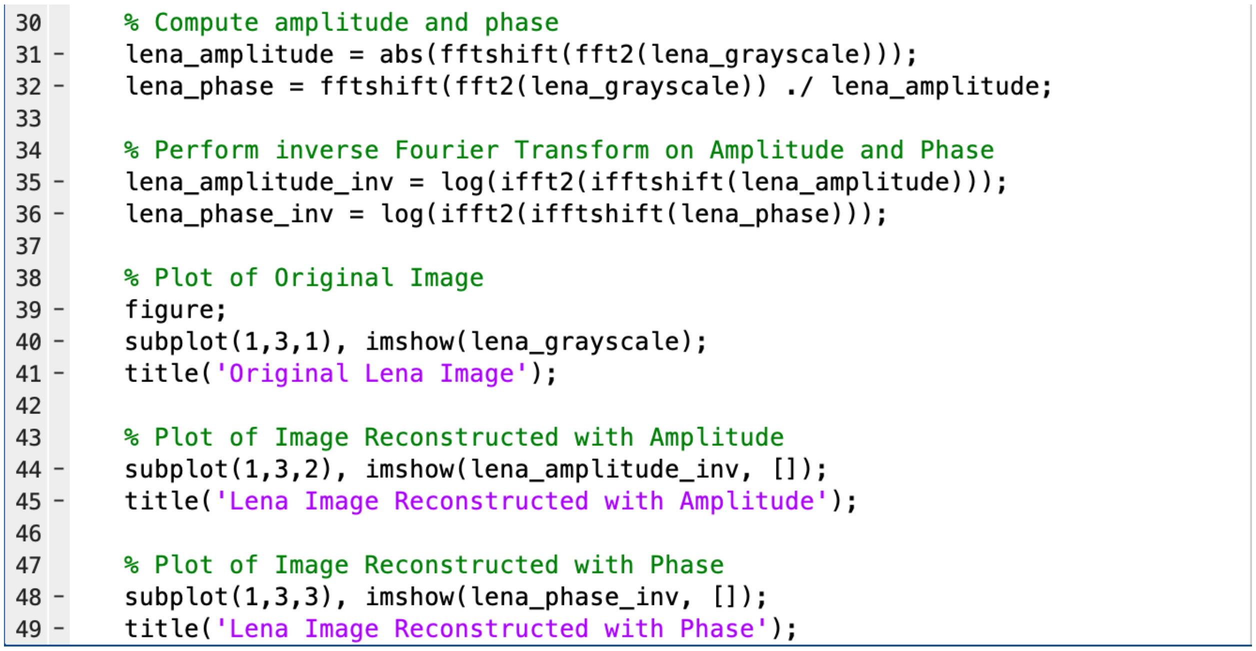
In Section 4, we designed a frequency domain filter to eliminate the periodic noise in the *frequnoisy* image. We examined the image's Fourier spectrum and found four anomaly pixels that represented the noise in the original image. We eliminated these pixels by setting their intensity value to zero. We then performed an inverse Fourier transform to view the image in the spatial domain and found that the noise had been eliminated. Thus, our frequency domain filter was successful.

# Appendix A



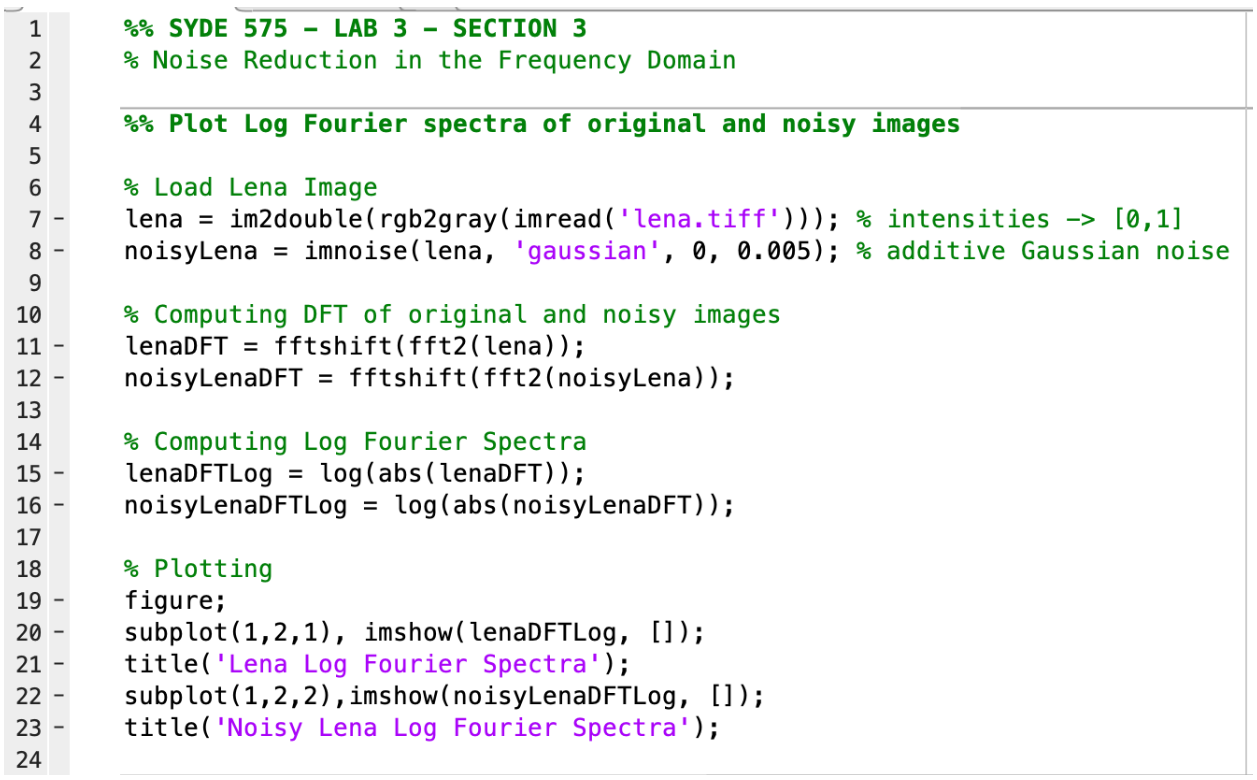
```
% create 256 x 256 test image (white rectangle)
f = zeros(256, 256);
f(:, 108:148) = 1;

% Fourier Spectra of test image
f_fs = fftshift(abs(fft2(f)));

% Plot the images
figure;
subplot(2,1,1), imshow(f);
title('Test Image');
subplot(2,1,2), imshow(f_fs, []);
title('Fourier Spectra of Test Image');

% Rotate test image by 45 degrees
f_rotated = imrotate(f, 45);
f_fs_rotated = fftshift(abs(fft2(f_rotated)));

% Plot the images
figure;
subplot(2,1,1), imshow(f_rotated);
title('Test Image');
subplot(2,1,2), imshow(f_fs_rotated, []);
title('Fourier Spectra of Test Image');

% Load lena and convert to grayscale
lena = imread('lena.tiff');
lena_grayscale = rgb2gray(lena);

```

Figure A.1.1: Section 2 Code



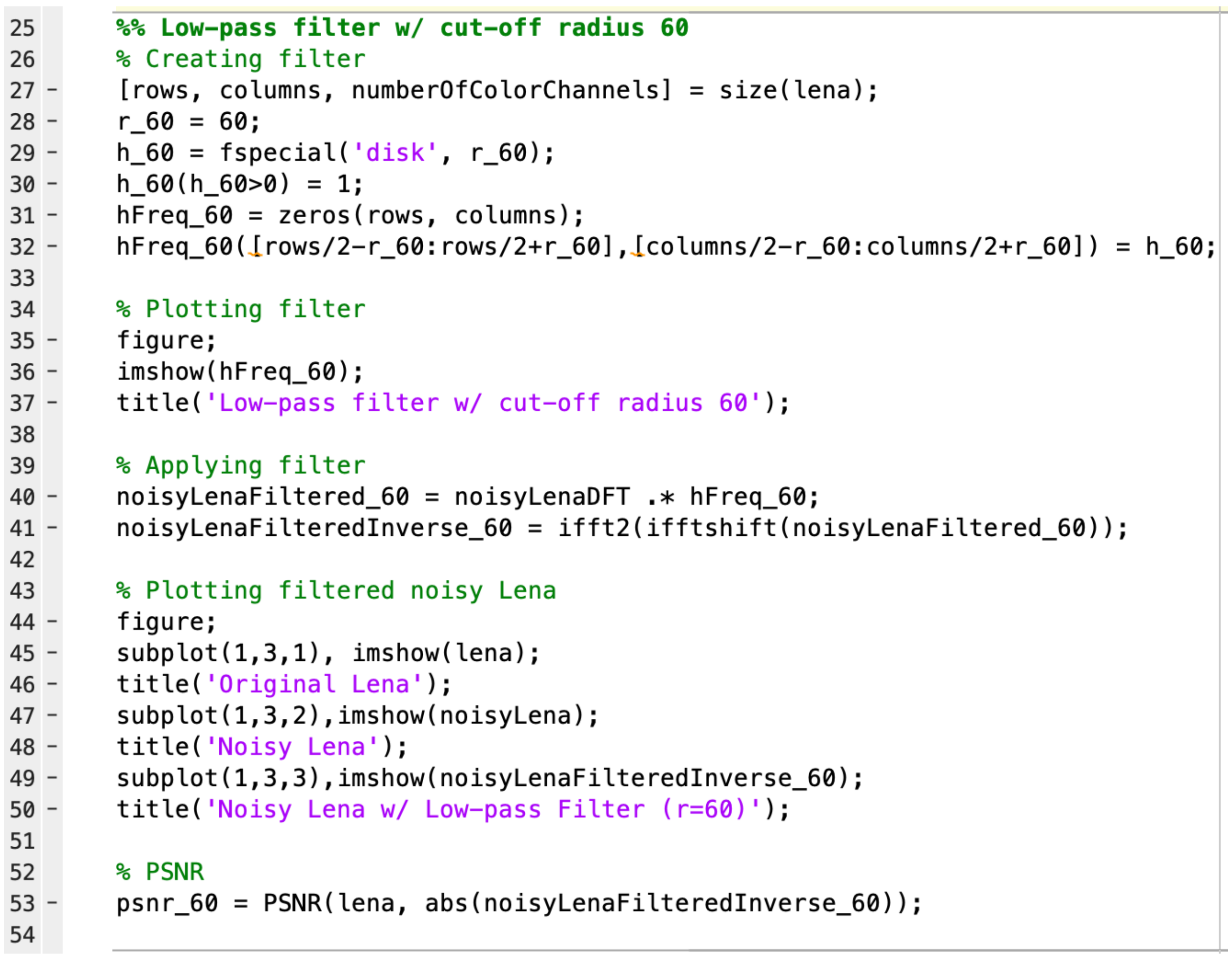
```matlab
% Compute amplitude and phase
lena_amplitude = abs(fftshift(fft2(lena_grayscale)));
lena_phase = fftshift(fft2(lena_grayscale)) ./ lena_amplitude;

% Perform inverse Fourier Transform on Amplitude and Phase
lena_amplitude_inv = log(ifft2(ifftshift(lena_amplitude)));
lena_phase_inv = log(ifft2(ifftshift(lena_phase)));

% Plot of Original Image
figure;
subplot(1,3,1), imshow(lena_grayscale);
title('Original Lena Image');

% Plot of Image Reconstructed with Amplitude
subplot(1,3,2), imshow(lena_amplitude_inv, []);
title('Lena Image Reconstructed with Amplitude');

% Plot of Image Reconstructed with Phase
subplot(1,3,3), imshow(lena_phase_inv, []);
title('Lena Image Reconstructed with Phase');
```

Figure A.1.2: Section 2 Code (continued)



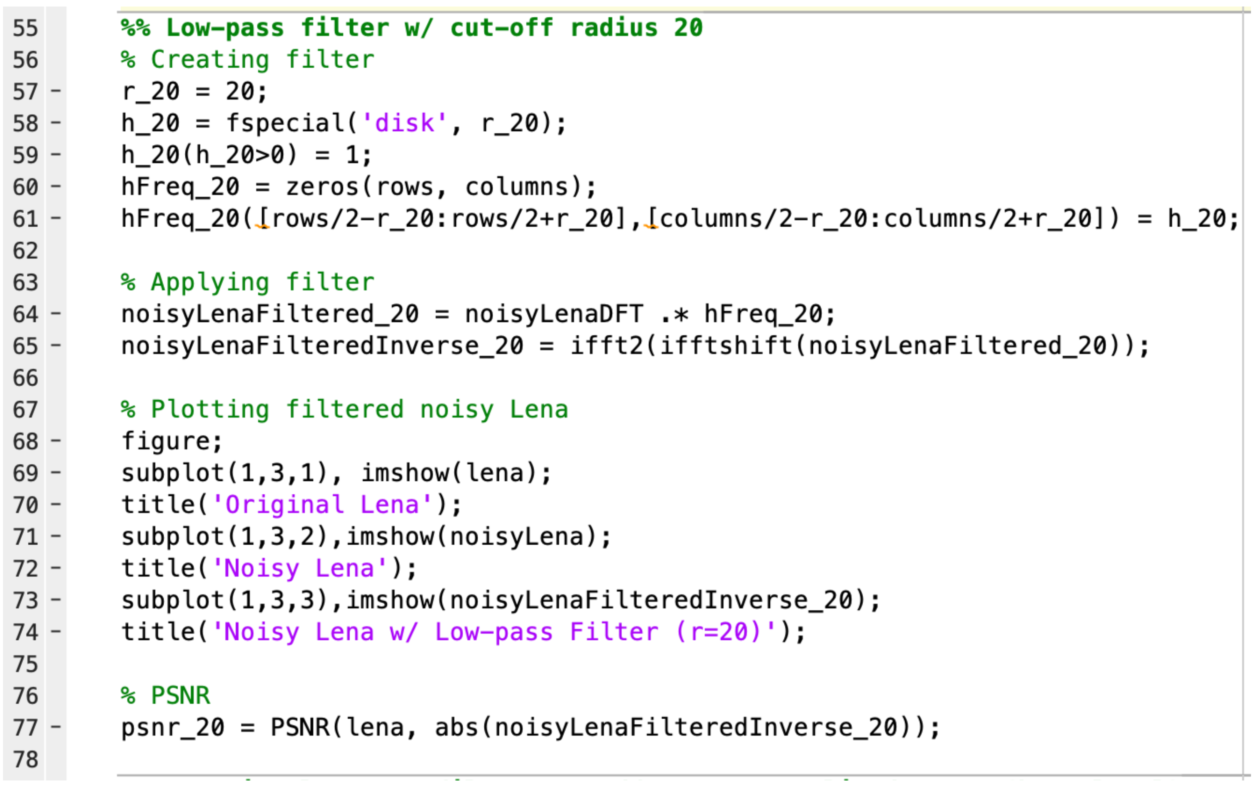
```matlab
%% SYDE 575 - LAB 3 - SECTION 3
% Noise Reduction in the Frequency Domain

%% Plot Log Fourier spectra of original and noisy images

% Load Lena Image
lena = im2double(rgb2gray(imread('lena.tiff'))); % intensities -> [0,1]
noisyLena = imnoise(lena, 'gaussian', 0, 0.005); % additive Gaussian noise

% Computing DFT of original and noisy images
lenaDFT = fftshift(fft2(lena));
noisyLenaDFT = fftshift(fft2(noisyLena));

% Computing Log Fourier Spectra
lenaDFTLog = log(abs(lenaDFT));
noisyLenaDFTLog = log(abs(noisyLenaDFT));

% Plotting
figure;
subplot(1,2,1), imshow(lenaDFTLog, []);
title('Lena Log Fourier Spectra');
subplot(1,2,2),imshow(noisyLenaDFTLog, []);
title('Noisy Lena Log Fourier Spectra');

```

Figure A.2.1: Section 3 Code



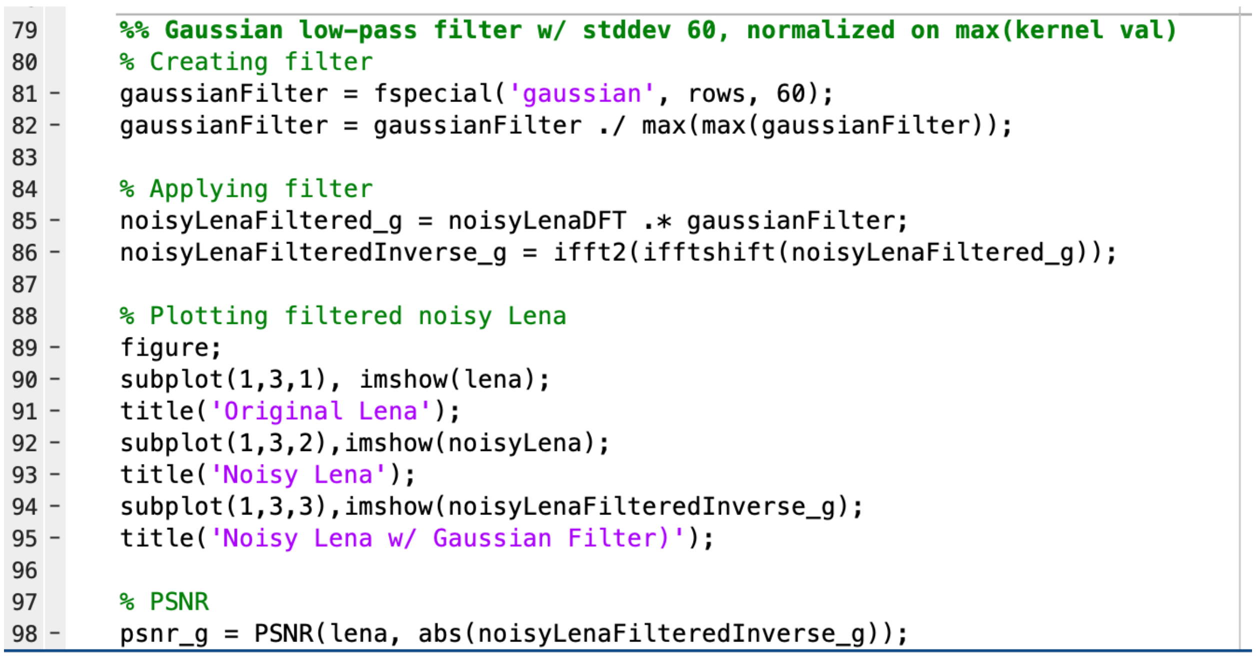
```
%% Low-pass filter w/ cut-off radius 60
% Creating filter
[rows, columns, numberOfColorChannels] = size(lena);
r_60 = 60;
h_60 = fspecial('disk', r_60);
h_60(h_60>0) = 1;
hFreq_60 = zeros(rows, columns);
hFreq_60([rows/2-r_60:rows/2+r_60],[columns/2-r_60:columns/2+r_60]) = h_60;

% Plotting filter
figure;
imshow(hFreq_60);
title('Low-pass filter w/ cut-off radius 60');

% Applying filter
noisyLenaFiltered_60 = noisyLenaDFT .* hFreq_60;
noisyLenaFilteredInverse_60 = ifft2(ifftshift(noisyLenaFiltered_60));

% Plotting filtered noisy Lena
figure;
subplot(1,3,1), imshow(lena);
title('Original Lena');
subplot(1,3,2),imshow(noisyLena);
title('Noisy Lena');
subplot(1,3,3),imshow(noisyLenaFilteredInverse_60);
title('Noisy Lena w/ Low-pass Filter (r=60)');

% PSNR
psnr_60 = PSNR(lena, abs(noisyLenaFilteredInverse_60));

```

Figure A.2.2: Section 3 Code (continued)



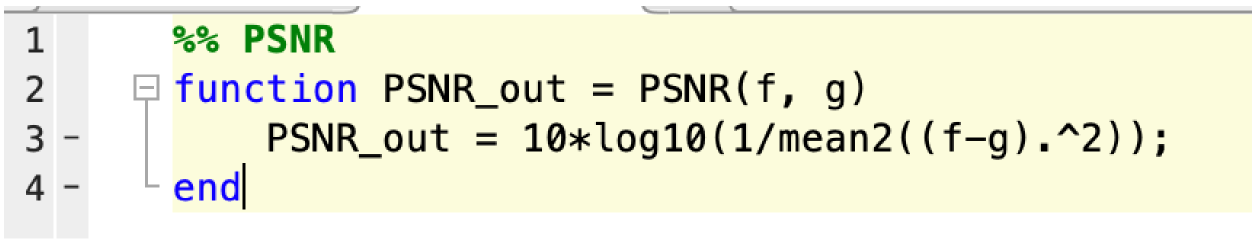
```matlab
%% Low-pass filter w/ cut-off radius 20
% Creating filter
r_20 = 20;
h_20 = fspecial('disk', r_20);
h_20(h_20>0) = 1;
hFreq_20 = zeros(rows, columns);
hFreq_20([rows/2-r_20:rows/2+r_20],[columns/2-r_20:columns/2+r_20]) = h_20;

% Applying filter
noisyLenaFiltered_20 = noisyLenaDFT .* hFreq_20;
noisyLenaFilteredInverse_20 = ifft2(ifftshift(noisyLenaFiltered_20));

% Plotting filtered noisy Lena
figure;
subplot(1,3,1), imshow(lena);
title('Original Lena');
subplot(1,3,2),imshow(noisyLena);
title('Noisy Lena');
subplot(1,3,3),imshow(noisyLenaFilteredInverse_20);
title('Noisy Lena w/ Low-pass Filter (r=20)');

% PSNR
psnr_20 = PSNR(lena, abs(noisyLenaFilteredInverse_20));
```

Figure A.2.3: Section 3 Code (continued)



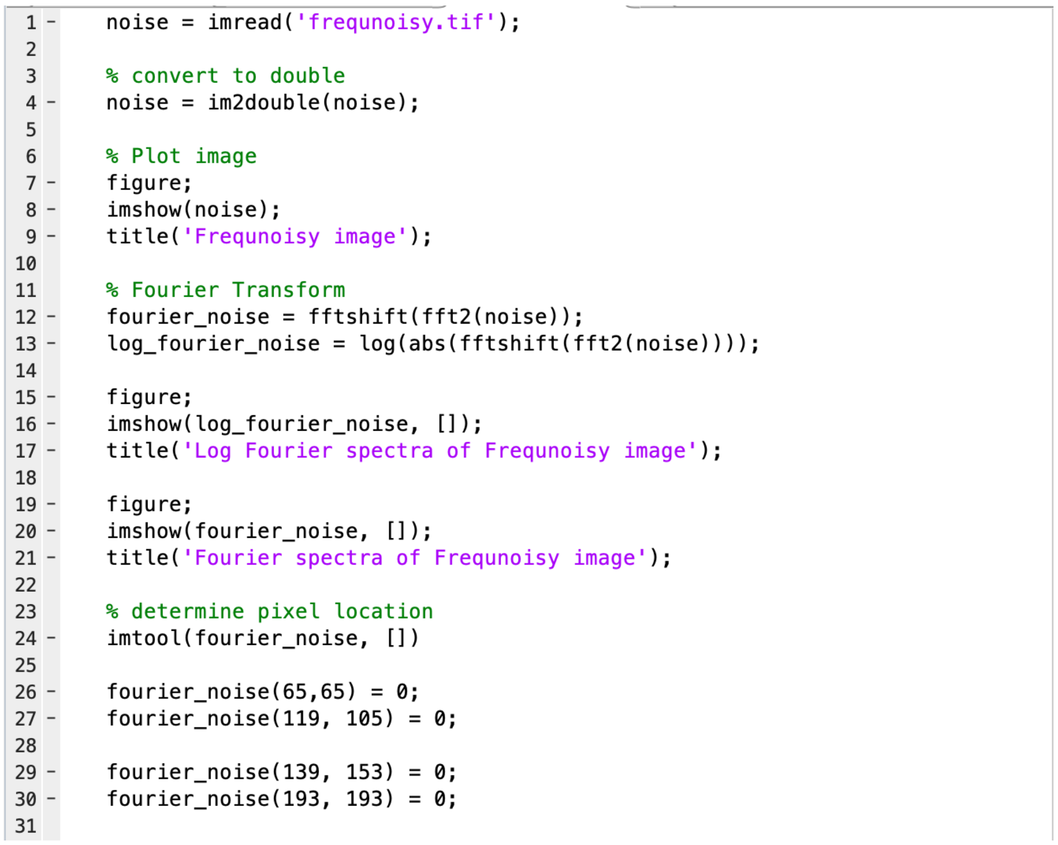
```matlab
%% Gaussian low-pass filter w/ stddev 60, normalized on max(kernel val)
% Creating filter
gaussianFilter = fspecial('gaussian', rows, 60);
gaussianFilter = gaussianFilter ./ max(max(gaussianFilter));

% Applying filter
noisyLenaFiltered_g = noisyLenaDFT .* gaussianFilter;
noisyLenaFilteredInverse_g = ifft2(ifftshift(noisyLenaFiltered_g));

% Plotting filtered noisy Lena
figure;
subplot(1,3,1), imshow(lena);
title('Original Lena');
subplot(1,3,2),imshow(noisyLena);
title('Noisy Lena');
subplot(1,3,3),imshow(noisyLenaFilteredInverse_g);
title('Noisy Lena w/ Gaussian Filter)');

% PSNR
psnr_g = PSNR(lena, abs(noisyLenaFilteredInverse_g));
```

Figure A.2.4: Section 3 Code (continued)



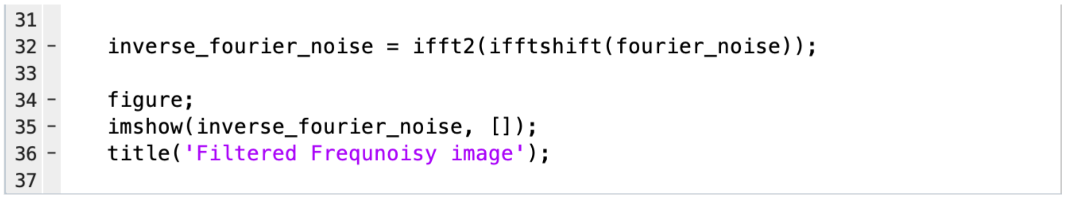
```
%% PSNR
function PSNR_out = PSNR(f, g)
    PSNR_out = 10*log10(1/mean2((f-g).^2));
end
```

Figure A.3.1: PSNR Code



```
noise = imread('frequnoisy.tif');

% convert to double
noise = im2double(noise);

% Plot image
figure;
imshow(noise);
title('Frequnoisy image');

% Fourier Transform
fourier_noise = fftshift(fft2(noise));
log_fourier_noise = log(abs(fftshift(fft2(noise))));

figure;
imshow(log_fourier_noise, []);
title('Log Fourier spectra of Frequnoisy image');

figure;
imshow(fourier_noise, []);
title('Fourier spectra of Frequnoisy image');

% determine pixel location
imtool(fourier_noise, [])

fourier_noise(65,65) = 0;
fourier_noise(119, 105) = 0;

fourier_noise(139, 153) = 0;
fourier_noise(193, 193) = 0;

```

Figure A.4.1: Section 4 Code



```

inverse_fourier_noise = ifft2(ifftshift(fourier_noise));

figure;
imshow(inverse_fourier_noise, []);
title('Filtered Frequnoisy image');

```

Figure A.4.2: Section 4 Code (continued)