Image Processing & Filtering

Jared Woelfel – 01570415 – 14 May 2021

Introduction The goal of this project was to explore methods of image processing in MATLAB by expanding the use of FIR filters into two dimensions. A brain BMP scan was used as the base image signal. Noise was randomly added to each pixel of the base image. Two FIR filters were tested to analyze and compare their performance in terms of SNR improvement and output image quality.

Solution For the first part of the project, the brain BMP scan image was loaded into MATLAB and recorded in Figure 1. Since the PSD analysis of an image must be conducted in two dimensions according to pixel value and position, the PSD estimate for the noiseless BMP scan was calculated and displayed using two methods. First, the meshgrid representation of the PSD estimate was recorded in 3D in Figure 2. Then, the greyscale 2D representation of the PSD estimate was recorded in Figure 3. These figures were produced using the code under the label "Standard PSD" in the MATLAB code of Figure 4.

The second part of the project consisted of adding random noise to the BMP image. This was done by adding noise with a magnitude between 0% and 20% of the maximum 255 amplitude (UINT8 standard) to each pixel in the image as seen in the MATLAB code of Figure 5. The resulting "noisy" image was displayed in Figure 6 and the SNR was calculated by averaging the ratio between the base image and noise signal at each pixel [1]. The PSD was then estimated using code identical in format to the code in Figure 3. The resulting PSD estimate visualizations were recorded in meshgrid form in Figure 7 and grayscale form in Figure 8. The PSD estimate visualizations of the "noisy" image looked similar to that of the noiseless image except that the dark regions void of information in the noiseless image were slightly lighter (indicating some power added).

The third part of the project involved the design of two FIR filters meant to recover as much of the original image as possible. The first filter was a 30th order ideal low pass filter with a normalized cutoff frequency of 0.3333 created using the MATLAB code in Figure 9 [3] and visualized in the frequency space plot of Figure 10. An increase in filter order or normalized cutoff frequency resulted in a blurrier image. A blurrier image resulted in higher SNR but also loss of information from the original image. Therefore, a unique filter was created to improve upon the performance of the ideal LPF. The filter order remained 30th order. However, the lowest band of frequencies (less than the 0.1111 normalized cutoff) were amplified by 1. The middle band of frequencies (between 0.1111 and 0.4286 normalized) were amplified by 0.05. The rest of the frequencies were amplified by 0.01. This filter was created using the MATLAB code in Figure 11 and visualized in the frequency space plot of Figure 12. As seen in the next section, this filter performed significantly better than the ideal low pass filter without any increase in filter order. An IIR filter was not designed because IIR filters are effective in image processing. This is due to the instability and quantization error that results from these filters being expanded to 2 dimensions and applied to an image [2].

Finally, both filters were applied to the "noisy" image. The image output from the ideal lowpass filter was recorded in Figure 13. The PSD estimate visualizations of the lowpass filtered image were created using MATLAB code similar in format to the code seen in Figure 3. The visualizations were recorded in meshgrid form in Figure 14 and grayscale form in Figure 15. The PSD of the lowpass filtered image had ringing in the darker regions that resulted in blurriness of the overall image. The image output from the unique filter design was recorded in Figure 16. The PSD estimate visualizations were recorded in meshgrid form in Figure 17 and grayscale form in Figure 18. The PSD of the uniquely filtered image had no ringing but allowed a small amount of information to pass through from the edges of the PSD in a uniform shape. This PSD matched the original image much more closely than the lowpass filtered PSD. The unique filter performed better than the ideal lowpass filter in every way. The SNR was improved, and more information was recovered from the original BMP scan. This unique filter design was more complicated and required more tuning to ensure a good performance for this image, but the lowpass filter was not precise enough to remove the noise without also blurring the information. Since image processing is done exclusively within the context of digital systems, the unique filter design would be preferred almost universally because the complexity of the filter does not affect its realization in a digital system. Both filters were the same size (order) and would take approximately the same amount of time to process an image.

Conclusion

The main issue encountered in this project was that SNR and blurriness seemed to be proportional. Therefore, as the blurriness of the image increased, the SNR increased. The unique filter seemed to overcome this issue by sloping down to a minimum amplitude instead of having a sharp cutoff like the ideal lowpass filter. For future improvement, a higher order filter with amplitude roll-off described by something like a cosine wave with its peak at the center of the filter might produce even better results. However, it seems unlikely that any filter could produce an SNR in the 25-30dB range without severely blurring the image and losing important information. Additionally, MATLAB's Image Processing Toolbox was unavailable for this project but would provide a variety of interesting noise injection and image filtering methods to test for future projects.

Appendix

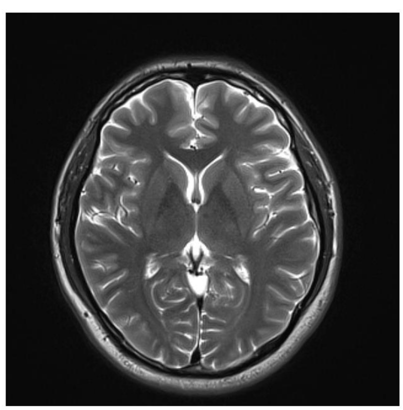

Figure 1: Brain BMP Scan Base Image

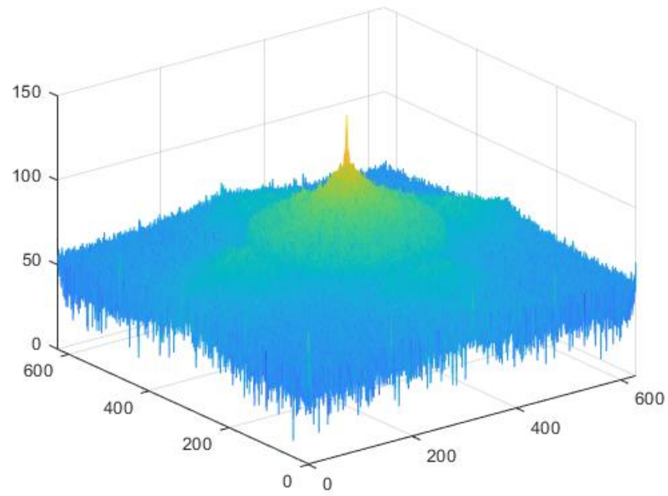

Figure 2: Meshgrid PSD Estimate Visualization of Base Image

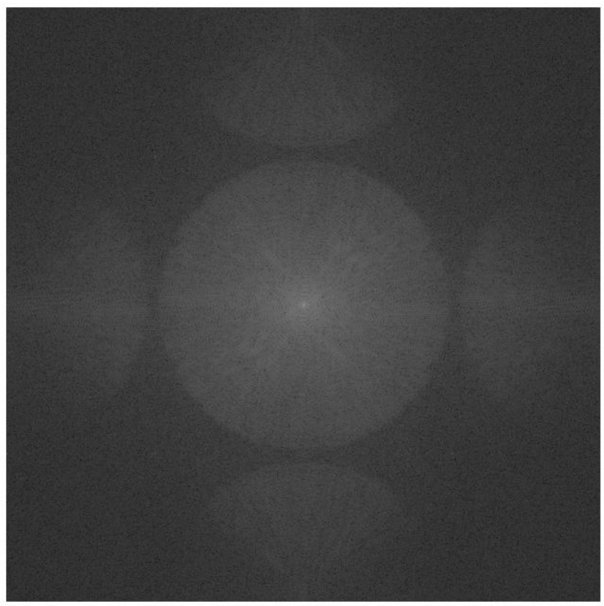

Figure 3: Grayscale PSD Estimate Visualization of Base Image

```
%% Standard PSD
psd = 10*log10(abs(fftshift(fft2(BRAINBMP))).^2 );
figure; clf
mesh(psd);
figure
imshow(uint8(psd));
```

Figure 4: MATLAB Code for PSD Estimation

```
%% Noise Addition
noise = uint8(0.3*255*rand(length(BRAINBMP),length(BRAINBMP)))
noisyBRAINBMP = BRAINBMP + noise;
figure
imshow(noisyBRAINBMP);
UnfilterSNR = 20*log10(mean(BRAINBMP(:)./noise(:)))
```

Figure 5: MATLAB Code for Noise Addition

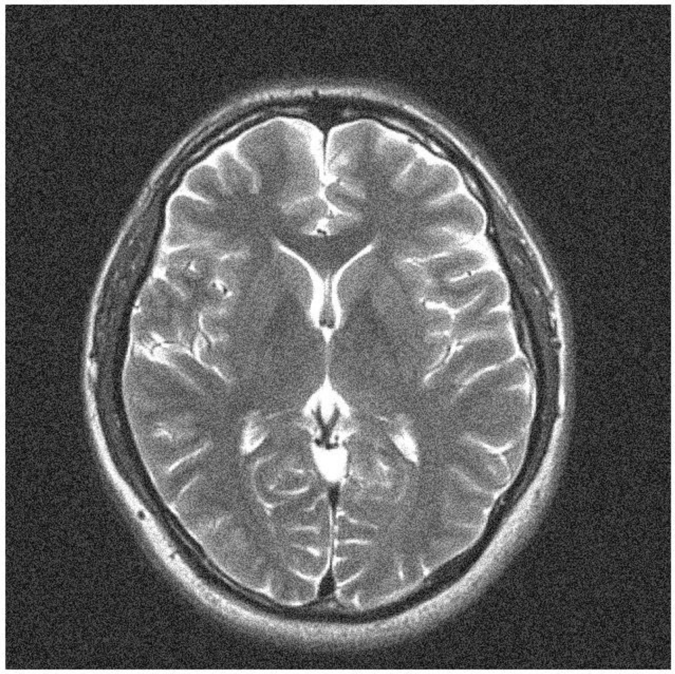

Figure 6: Noisy Brain BMP Scan Image (SNR: 13.96dB)

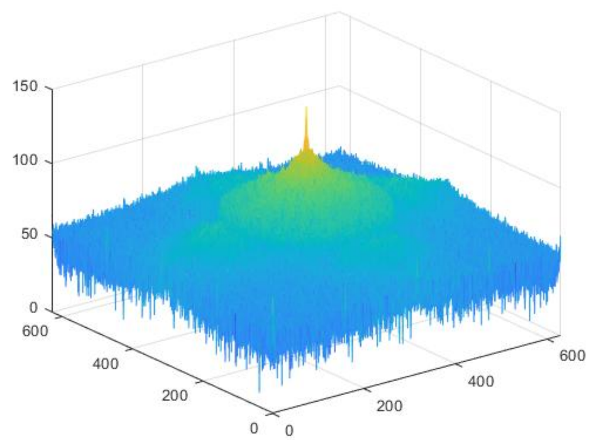

Figure 7: Meshgrid PSD Estimate Visualization of Noisy Image

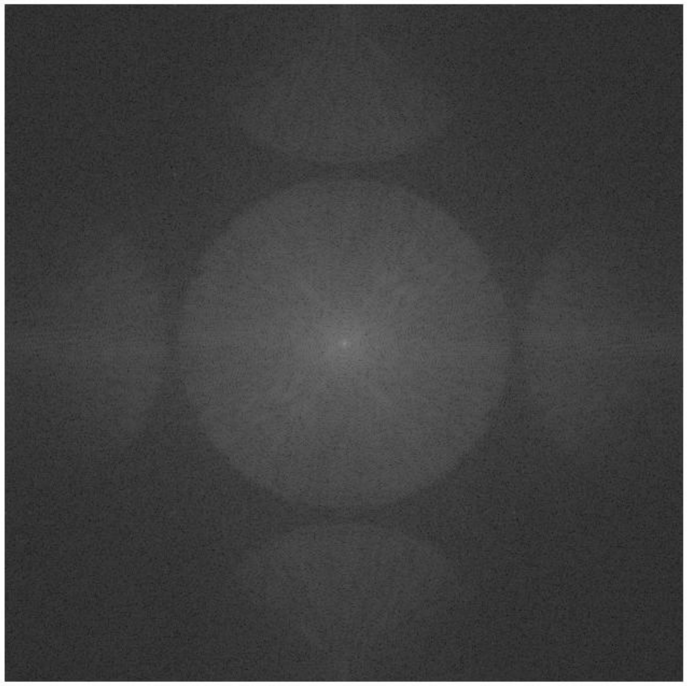

Figure 8: Grayscale PSD Estimate Visualization of Noisy Image

```
%% Ideal Lowpass FIR Filter Design
wn = (420-315)/315
order = 30;
[f1,f2] = freqspace(order,'meshgrid');
Hd = zeros(order,order); d1 = sqrt(f1.^2 + f2.^2) < wn;
Hd(d1) = 1;
Hd = Hd./sum(sum(Hd));
figure
mesh(f1,f2,Hd)
filterOut1 = filter2(Hd,noisyBRAINBMP);
filterOut1 = uint8(filterOut1);
figure
imshow(filterOut1);
LPFilterSNR = 20*log10(mean(BRAINBMP(:)./(filterOut1(:)-BRAINBMP(:))))
```

Figure 9: MATLAB Code for Ideal Lowpass Filter Generation

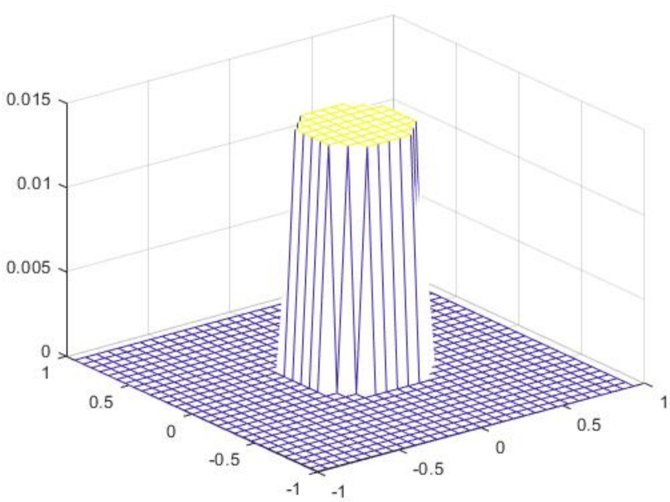

Figure 10: Ideal Lowpass Filter Visualization

```
%% Unique FIR Filter Design
wnl = (350-315)/315
wnh = (450-315)/315
order = 30;
[f1,f2] = freqspace(order,'meshgrid');
Hd = zeros(order,order); d1 = sqrt(f1.^2 + f2.^2) < wnl;
d2 = sqrt(f1.^2 + f2.^2) < wnh;
Hd(:) = 0.01;
Hd(d2) = 0.05;
Hd(d1) = 1;
Hd = Hd./sum(sum(Hd));
figure
mesh(f1,f2,Hd)
filterOut2 = filter2(Hd,noisyBRAINBMP);
filterOut2 = uint8(filterOut2);
figure
imshow(filterOut2);
UniqueFilterSNR = 20*log10(mean(BRAINBMP(:)./(filterOut2(:)-BRAINBMP(:))))
```

Figure 11: MATLAB Code for Unique Filter Generation

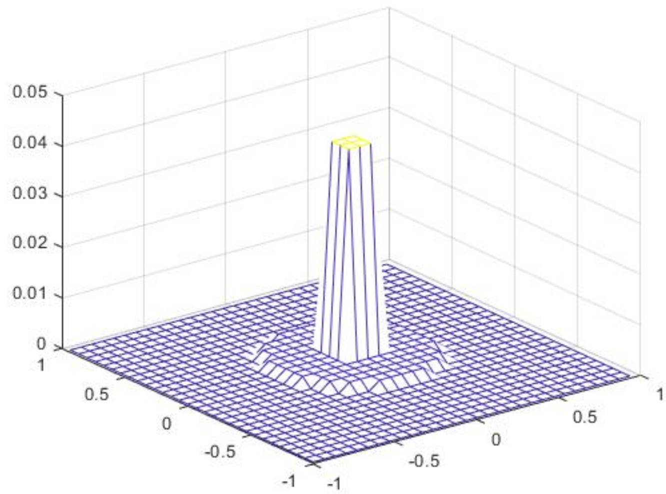

Figure 12: Unique Filter Visualization

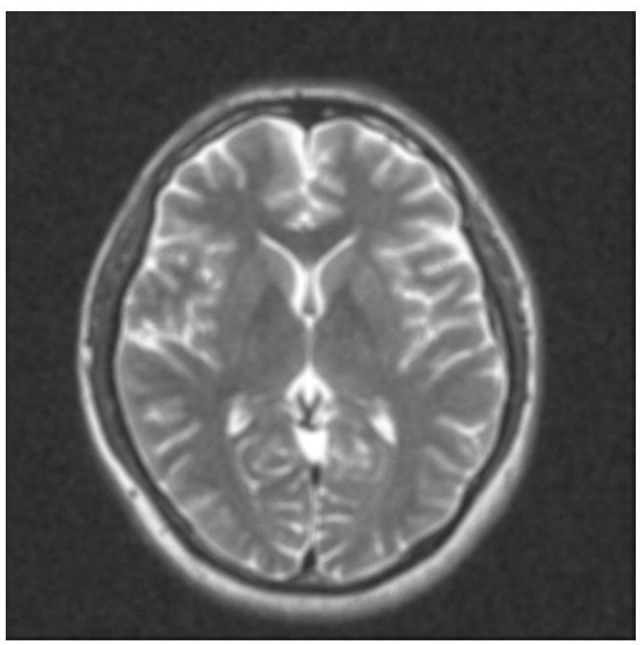

Figure 13: Ideal Lowpass Filter Output Image (SNR: 19.19dB)

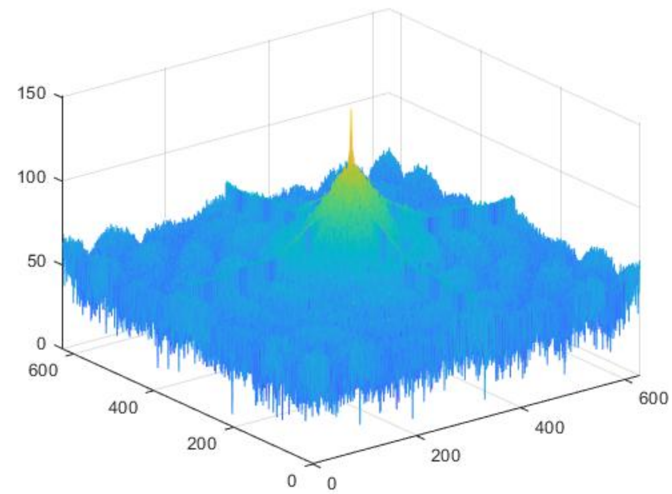

Figure 14: Meshgrid PSD Estimate Visualization of Ideal Lowpass Filter Output

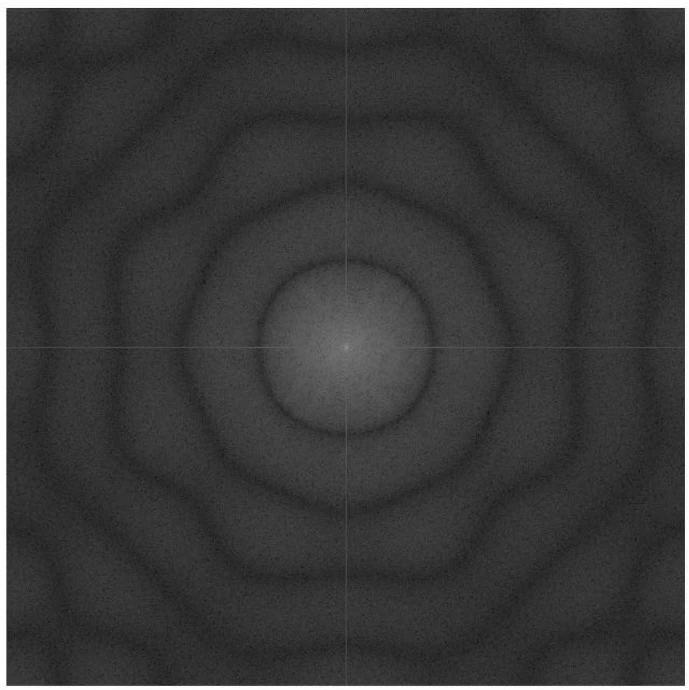

Figure 15: Grayscale PSD Estimate Visualization of Ideal Lowpass Filter Output

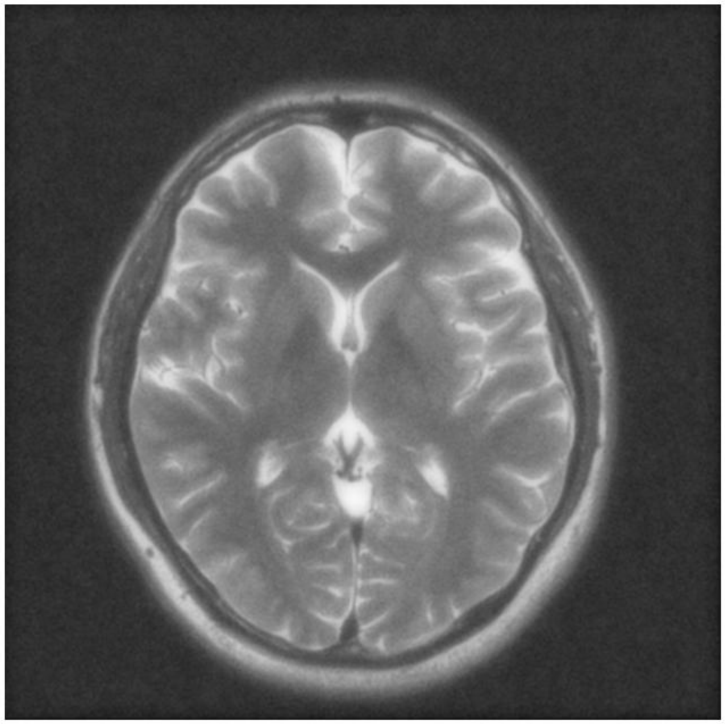

Figure 16: Unique Filter Output Image (SNR: 21.77dB)

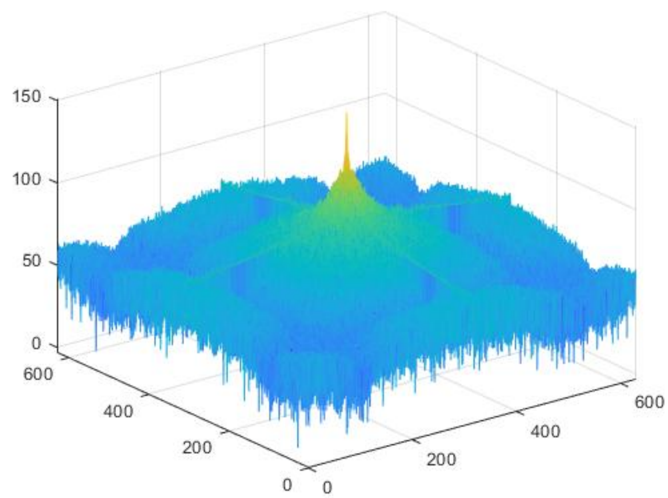

Figure 17: Meshgrid PSD Estimate Visualization of Unique Filter Output

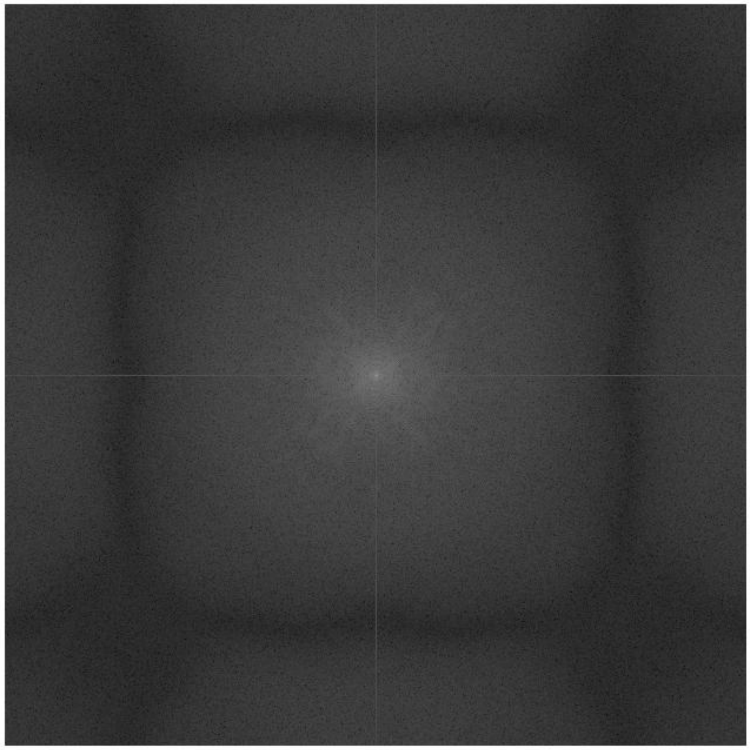

Figure 18: Grayscale PSD Estimate Visualization of Unique Filter Output

Reference

[1] "How to Calculate Actual SNR of Image in MATLAB? - MATLAB Answers - MATLAB Central," Mathworks.com. [Online]. Available: https://www.mathworks.com/matlabcentral/answers/55649-how-to-calculate-actual-snr-of-image-in-matlab. [Accessed: 14-May-2021].

[2] "IIR Filters," Sciencedirect.com. [Online]. Available: https://www.sciencedirect.com/topics/engineering/iir-filters. [Accessed: 14-May-2021].

[3] "Frequency spacing for frequency response - MATLAB freqspace," Mathworks.com. [Online]. Available: https://www.mathworks.com/help/matlab/ref/freqspace.html. [Accessed: 14-May-2021].