Medical Image Filtering and Denoising

Xuhui Wang

January 2021

Introduction

Image filtering techniques, widely used in medical imaging and medical image processing, have a variety of potential and present applications. The design of image filters primarily legerages the use of priori knowledge about the noise corrupting the data. In this project, upon testing on medical lung CT and chest X-ray, I introduce different methods of filtering and denoising such as average, Gaussian, and so on reduce noisy artifacts by smoothing. On the other hand, sharpening filters enhance the high-frequency details, making the image nonsmooth, is are also discussed.

Loading, Manipulating and Storing Images

(Note that, in this report, the wordings that describe the questions are from the course material by Dr. Albu and I will give the appropriate citation at the end of each corresponding paragraph. On the other hand, all figures, tables, methods implementation, statistics, analysis, explanations and conclusions are all from my personal work.) DICOM (Digital Imaging and Communications in Medicine) is a standard for handling, storing, printing, and transmitting information in medical imaging. DICOM files are 4-D files with the '.dcm' format that encompass a sequence of images received from an imaging device. The dimensions of these files are [rows, columns, samples, slices] where samples represents the number of color channels per image (each image is represented by an unit of rows $\times$ columns), and slices comprises the number of images. [1]

We replace all the negative values in the first slice of a lung CT DICOM file by zero and normalize the values between [0, 1] before doing any processing. The original first slice and the Gaussian noise slice are in Figure 1.

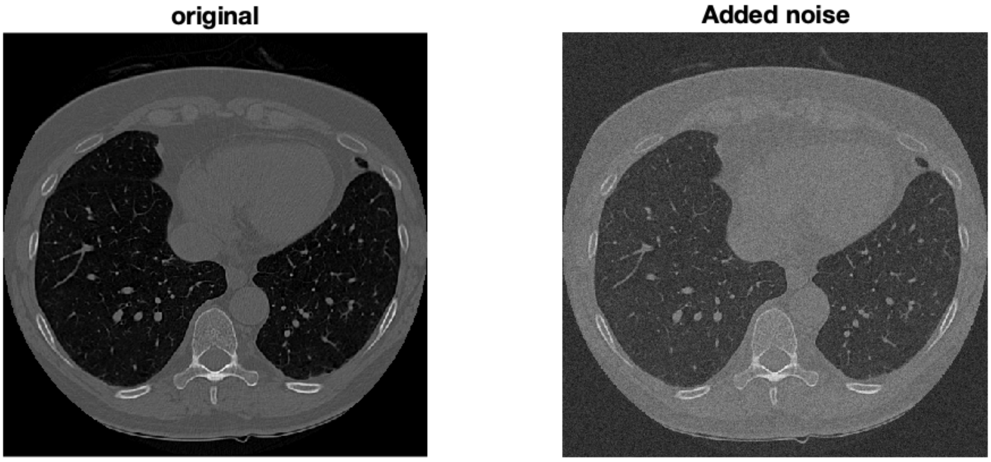

Figure 1: Gaussian

There is no way to save the noisy image with double values in the '.png' format and then load it in MATLAB without loss of information. We have normalized the grayscale image values in $[0, 1]$ range, so values of pixels are floating point, instead of int. If we use the functions, such as *imwrite*, to save the image as a '.png' file, functions assume that the dynamic range is [0,1] and automatically scale the data by a finite integer before writing it to the file as n-bit int values, because of the formation and construction of '.png' protocol. So we may lose a bit precision but not too much during the process of scaling floating point to int. So, there is basically no way to save the noisy image with *double* values in the '.png' format and then load it in MATLAB without loss of information.

Regarding the dimension of the lung CT DICOM file, upon loading the information of the given DICOM file, there are 51 slices embedded in the file. The size of the DICOM file is 4-dimension, while the size of the slice is 2-dimension.

MSE is a standard measure to calculate the distance between two sets of observations (in our case, images). For two images I_1 and I_2, MSE is defined as follows:

$$MSE = \frac{\Sigma_{M,N}[I_1(m,n) - I_2(m,n)]^2}{M \times N} \tag{1}$$

where M and N are the number of rows and columns in the input image. The comparisons between the original image and noisy image for 2 different noise are in Figure 2 and 3 respectively.[1]

MSE result for 'salt & pepper' noise with $noise\ density = 0.05$ is 0.0189. MSE result for 'speckle' with $var_{speckle} = 0.05$ is 0.0029.

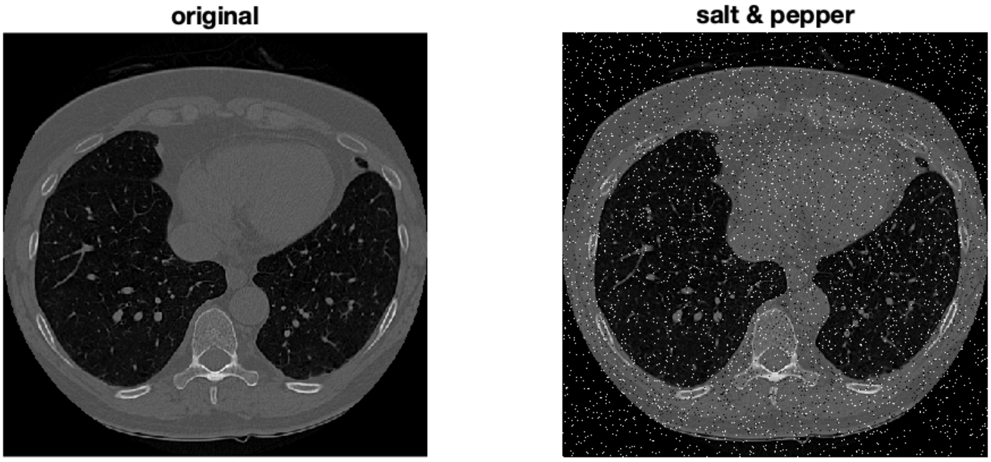

Figure 2: Salt & Pepper

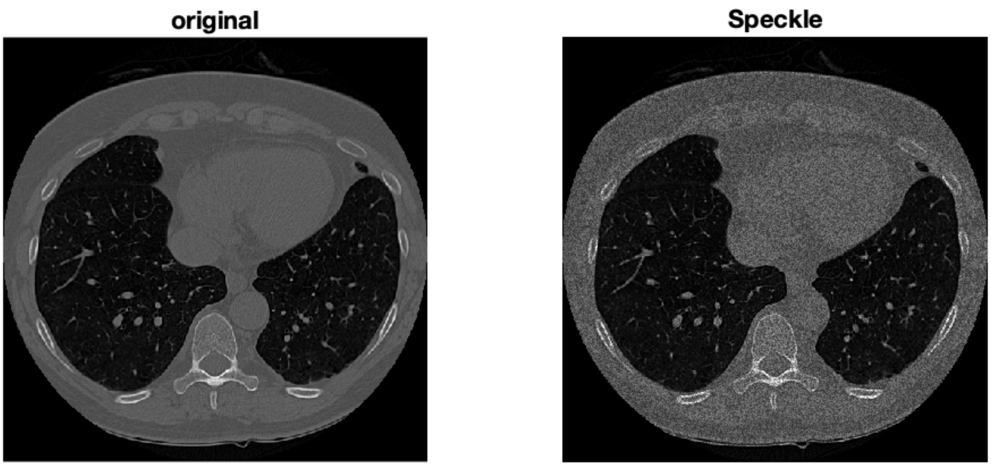

Figure 3: Speckle

Image Filtering

When acquiring images with imaging systems, one would typically perform filtering operations to mitigate noise, interference, variation in illumination, poor contrast, among others, before further processing the image. Thus, in virtually all computer vision systems, some sort of pre-processing step is required. Linear and nonlinear filters play important roles in this context. In this section of the assignment we will investigate the difference between convolution and correlation, which are operators used in linear filtering, and apply some filters to reduce the effects of noise in the input images. When sliding a filter on the image, the size of the output image would be less than that of the input because of a lack of information in the neighborhood of border pixels. To have an output that preserves the dimensions of the input, the size of the original image should be enlarged based on the size of the desired filter, by adding some extra rows and columns. This process is called image padding and can be automatically done in most of the filtering functions offered by MATLAB.[1]

After using zero-padding to preserve the size of the input, we used a Prewitt operator as the kernel to filter the image.

A Prewitt kernel is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Prewitt operator is either the corresponding gradient vector or the norm of this vector. The Prewitt operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical directions and is therefore relatively inexpensive in terms of computations like Sobel and Kayyali operators. So a horizontal Prewitt kernel can be used to detect the edge of the image, or to emphasizes vertical edges by approximating a horizontal gradient.

From the Figure 4, we could observe the horizontal gradient change of the vertical edge from the left perspective on the correlation graph, while we could observe the horizontal gradient change of the vertical edge from the right perspective on the convolution graph. They seem relatively symmetric due to the different 'rotation' operation step between correlation and convolution calculation.

There are a lot of advantages with convolution that make it a better option for filtering.
For example, convolution possesses Associative Law, which is very practical in scenarios that require multiple consecutive original convolutions: we can first perform convolution operations on each convolution kernel to obtain a new overlapping convolution kernel, and then perform a convolution with the input data to get the result; if the weight parameters of these convolution kernels are known constants, then we can go one step further: pre-calculate and store the overlapped convolution kernels to considerably enhance the computational efficiency.

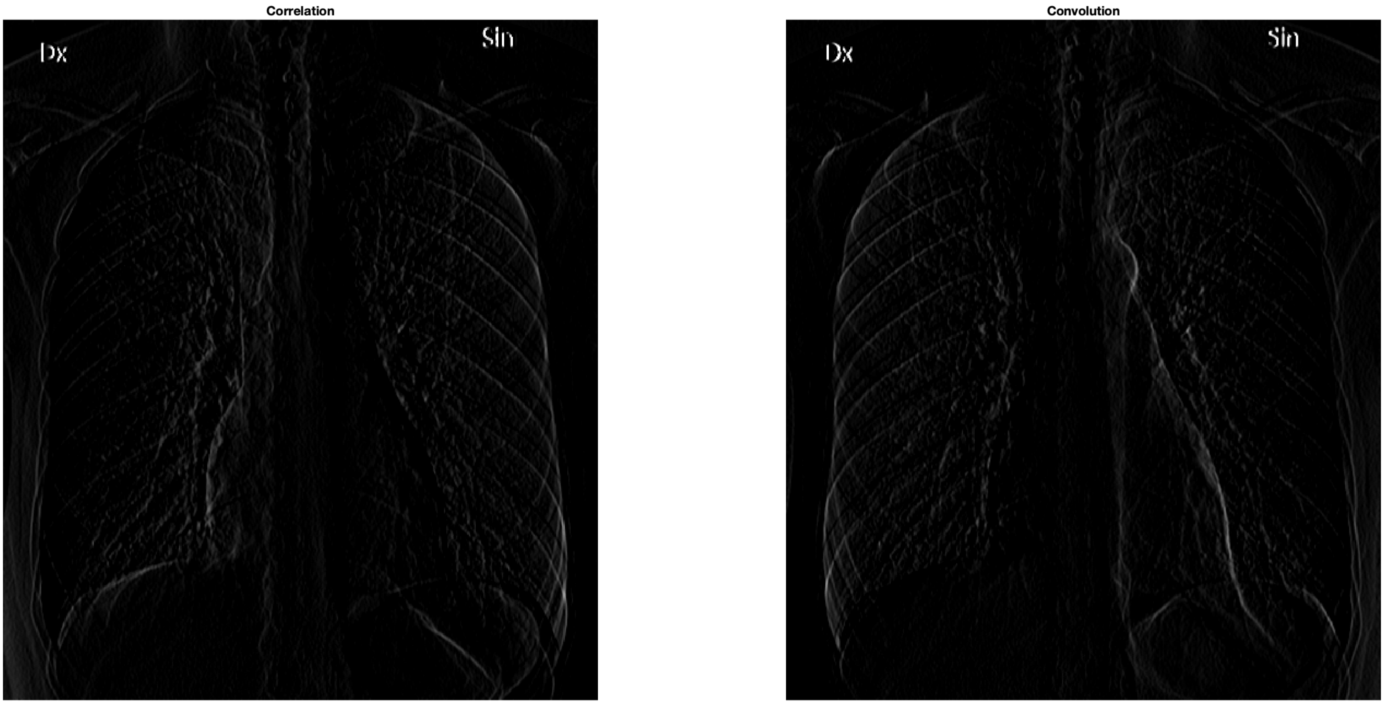

Figure 4: correlation and convolution

Image Denoising

The Gaussian noise image from the part 1 and the image after Gaussian filtering are in Figure 5.
The MSE between the original image and noisy image is 0.0171, while the MSE between the original image and the filtered image is 0.0113.
The salt & pepper noise image from the problem 2 and the image after Gaussian filtering are in Figure 6.
The MSE between the original image and noisy image is 0.0189, while the MSE between the original image and the filtered image is 0.0040.

The speckle noise image from the problem 2 and the image after Gaussian filtering are in Figure 7.
The MSE between the original image and noisy image is 0.0029, while the MSE between the original image and the filtered image is 7.5416e-04.

The decreased MSEs after filtering on 3 different images show that Gaussian filtering successfully mitigated noise, interference, variation in illumination, poor contrast and so on for 3 different types of noise. By observing Figure 5, 6 and 7, we can state that the noises considerably disappeared from the images after Gaussian filtering and the images became smooth.

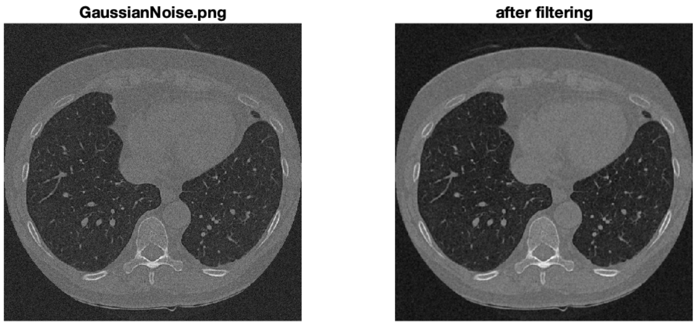

Figure 5: Gaussian noise

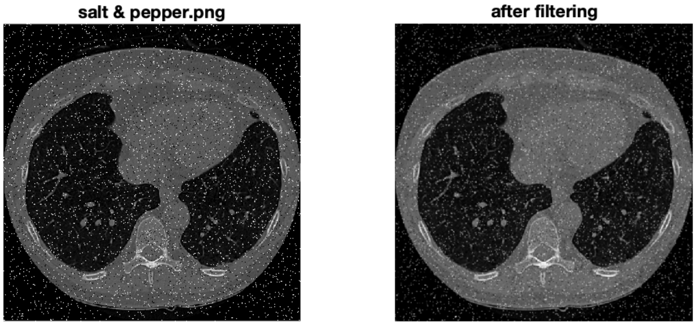

Figure 6: SaltPepper

Gaussian filtering is a linear filtering that uses the distribution of two-dimensional Gaussian functions to smooth the image.
The advantages of Gaussian filtering can be focused on the characteristics of Gaussian function:

- First of all, the two-dimensional Gaussian function is rotationally symmetric, with the same degree of smoothness in all directions, and will not change the edge direction of the original image.
- Second, the Gaussian function is a single-valued function. The anchor point of the Gaussian convolution kernel is an extreme value, which is monotonously decreasing in all directions. Anchor point pixels will not be highly affected by pixels far from the anchor point, ensuring the characteristics of feature points and edges.

Gaussian filtering is the process of weighted averaging the entire image. The value of each pixel is obtained by weighted averaging of itself and other pixel values in the neighborhood.

Specifically, we have very good performance when we apply Gaussian filtering on images with Gaussian noise. Gaussian noise is approximately normally distributed in amplitude, but distributed on each pixel, and the mean of the normal distribution is 0. When the mean value is 0, the Gaussian filter will replace that pixel with the mean value, so that the noise is gone.

Furthermore, we have very good performance when we apply Gaussian filtering on images with speckle noise. Speckle noise is the noise that arises due to the effect of environmental conditions on the imaging sensor during image acquisition. Speckle noise is mostly detected in case of medical images, active Radar images and Synthetic Aperture Radar (SAR) images. Speckle noise is essentially a multiplicative noise, which may have an additive noise as well. Given the property of multiplicative and additive noise, it is reasonable that Gaussian filtering is good to smooth the image and remove speckle noise.

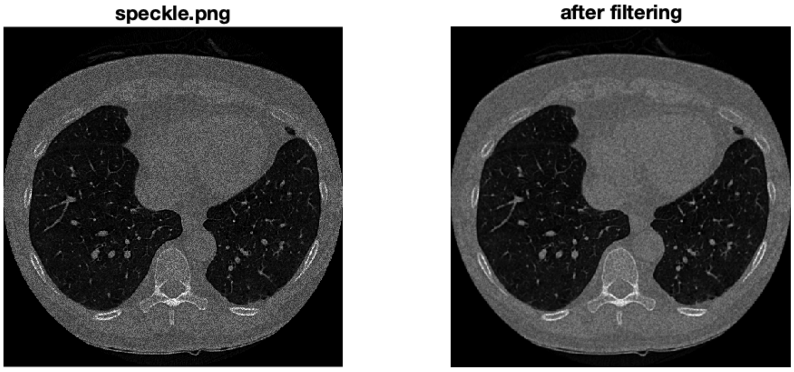

Figure 7: Speckle

There is no big difference between convolution and correlation in this case. Because the kernel we use is completely symmetric, actually convolution and correlation are the same one operation in this case.

The Gaussian noise image from the part 1 and the image after NLFiltering are in Figure 8.
The MSE between the original image and noisy image is 0.0171, while the MSE between the original image and the filtered image is 0.0162.
The salt & pepper noise image from the problem 2 and the image after NLFiltering are in Figure 9.
The MSE between the original image and noisy image is 0.0189, while the MSE between the original image and the filtered image is 3.8284e-04.
The speckle noise image from the problem 2 and the image after NLFiltering are in Figure 10.
The MSE between the original image and noisy image is 0.0029, while the MSE between the original image and the filtered image is 0.0012.

The decreased MSEs after filtering on 3 different images show that NLFiltering successfully mitigated noise, interference, variation in illumination, poor contrast and so on for 3 different types of noise. By observing Figure 8, 9 and 10, we can state that the noises considerably disappeared from the images after NLFiltering and the images became smooth.

Specifically, We have very good performance when we apply median filtering on images with speckle and salt & pepper noise. Median filter is a typical nonlinear filtering technique. The basic idea is to replace the gray value of the threshold with the median gray value of the neighborhood of the reverse point. This method not only eliminates impulse noise, salt & pepper and speckle noise but also retains the edge details of the image at the same time. Moreover, median filter is a nonlinear signal processing technology based on the sorting statistical theory that can effectively suppress noise. The basic principle is to replace the value of a point in a digital image or digital sequence by the median value of each point in a neighborhood, so that the surrounding pixel values are close to the true value, thereby eliminating isolated noise points. This filter is excellent at removing speckle noise and salt & pepper noise, because the filter does not highly depend on the values in the neighborhood that are way larger than the true value.

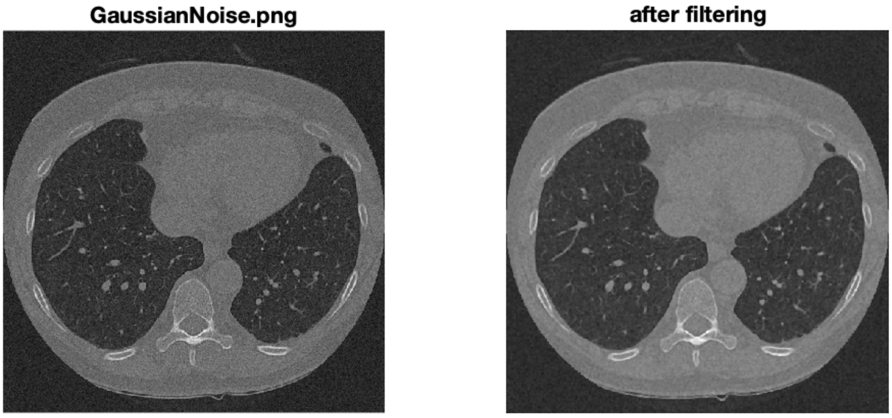

Figure 8: Gaussian

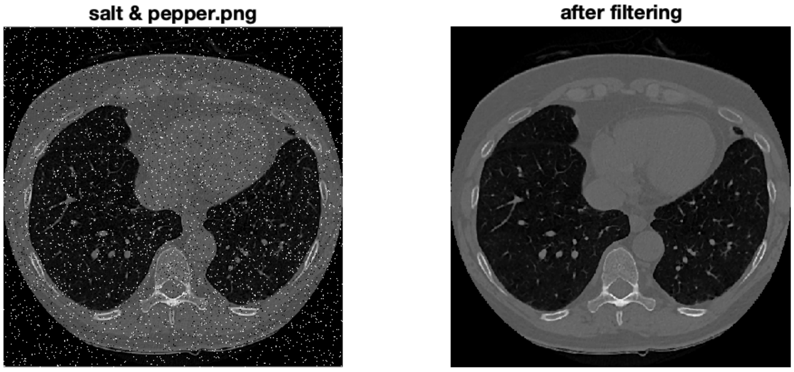

Figure 9: SaltPepper

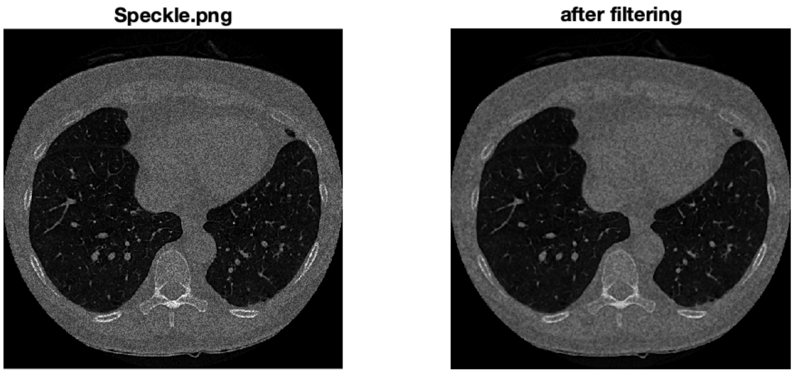

Figure 10: Speckle

There are 3 different MATLAB methods used in this part. Blockproc means distinct block processing function for image.

- It processes an image by applying a function to each distinct block of specific size and concatenating the results into the output matrix.
- It is a non-overlapping block operation commands.
- 'B = blockproc(A,[M N],fun)', means that the function 'fun' is performed on each block of the image A that has a size of [m n] which is not overlapped, and the result of the concatenation is B.
- Blockproc is a professional function that processes images with colors and big sizes.

Nlfilter means general sliding-neighborhood operations.

- It applies a function to each specific-size sliding block of the grayscale image.
- 'B = nlfilter(A, [m n], fun)', is a matrix where A is the image, [m n] is the size of the image processing area, fun is the function handle, and B is a scalar returned by the function.
- It means that the patch function is performed on each patch of size [m n] of the image A. The patch is sliding, that is, the center pixel of the patch traverses each point of the image, and the boundary padding is needed when the center pixel goes to boundary.
- It can only handle 2D matrix, which is grayscale image.

Colfilt means columnwise neighborhood operations.

- It processes the image by rearranging each specific block of the image into a column of a temporary matrix, and then applying the function fun to this matrix.
- 'B = colfilt(A,[m n],blocktype,fun)', means that the image A is rearranged into a temporary matrix composed of columns, each of which is a image block of size [m n], and the temporary function is operated on the temporary matrix.
- It can only handle 2D matrix, which is grayscale image.
- May be faster than nlfilter.

There are still some other differences between them.

Although commonly referred as a type of noise, speckle is actually an unwanted modification of the measured signal. If we model the reflectivety function of the surface under imaging as an array of scatters, because of the finite resolution of the imaging device, each individual cell will capture only the summation of all the back scatters coming out of it. Since the backscatteres might have different directions, this summation can be a stronger or weaker version of the expected return signal from that cell. One way to mitigate this phenomenon is to average the sequence of signals obtained in different time steps.[1]

The comparison of the original first slice of 'RIDER_Lung_CT.dcm' and the average image of first 10 slices with speckle noise of $var_{skeckle} = 0.1$ is in Figure 11, page 10.
The MSE between the first slice and its noisy version is 0.0058, while the MSE between the original slice and the average image is 0.0035.
So, we state that the temporal averaging method can considerably reduce the effect of speckle noise and averaging the sequence of signals obtained in different time steps is a good way to mitigate this phenomenon and speckle noise.

There is some drawbacks of the averaging method. Firstly, image averaging is susceptible to movement and motion. If we want everything in the image to be sharp in the final image average, we have to make sure that neither the camera nor the objects move between a sequence of images. So, as nice as it would be if this method worked for some special fields of photography, we have to be very careful about it.
Furthermore, though averaging method is a good way to mitigate the speckle noise, it still has other drawbacks. For instance, it may ignore some important details that change fast as time goes. When we remove additive speckle noise that are of independence spatially, those important details are also likely to be removed or be averaged off.

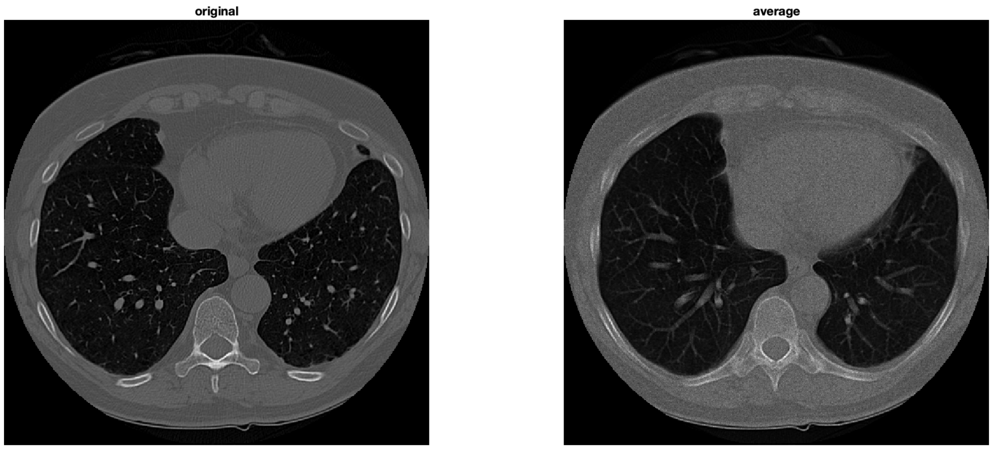

Figure 11: average

Conclusion

In this project, upon testing on medical lung CT and chest X-ray, I introduce different methods of filtering and denoising such as average, Gaussian, and so on reduce noisy artifacts by smoothing. On the other hand, sharpening filters enhance the high-frequency details, making the image nonsmooth, is are also discussed.

References

[1] A. Albu, *Ece435: Medical image processing course materials.*