Group Members:
Patrick Guerin
Semilore Omoyinmi
Andrew Romanof

GitHub: https://github.com/guerinp19/ComputatationalScience_finalPoject

Image Deblurring
Scientific Computation: Case Study 6

Introduction:

Suppose you are given some image that has been blurred by some known means, for example the motion blur from a camera or an unfocused lens. In some cases it is impossible to retake the picture so that leaves the question how do we deblur the image we have been given? The answer to this question is to use a linear system of equations. Where we can represent the blur as a matrix and the blurred and unblurred images as vectors. We can then represent the problem like this:

$$Kf = g$$

Where K is the matrix which represents the blurring of the image and g and f are vectors representing the blurred and unblurred image respectively, we can then simply solve for f to get the unblurred image. However, if we naively try to solve this problem, the resulting image looks completely destroyed, which means nothing. This happens because of noise, when we take a picture there is typically some noise caused by some outside force which we cannot predict, whether this noise be random, salt and pepper, gaussian or any other kind. This means we actually need to represent the problem like this:

$$Kf + ή = g$$

Where ή represents some noise we can't predict. Typically, noise causes the problem to be ill-posed meaning, the problem can't be solved with trivial methods. If we assume that the amount of noise is too small to reasonably affect the result of the problem and try to solve the first equation, again we destroy the image and we cannot retain any useful information. This is because we actually amplify noise when solving the system. Therefore we need to try another method to solve this problem.

To solve this issue, we need regularization. Regularization is a process used to simplify the answer of an ill-posed problem to prevent overfitting. For our purposes we will use it to minimize the effect that noise amplification has on the deblurring and prevent the deblurring image from being destroyed. There are many methods for regularization but the two we will focus on are Tikhonov Regularization and Truncated SVD.

Image blurring:

It's difficult to understand how we deblur an image without first understanding how we blur it. To blur an image we need to use a process called convolution. Convolution is the

process by which we take a two matrix and iterate the first matrix over each cell of the second centering the first matrix over top of each cell and taking a weighted average of all cells that overlap. The first matrix is called a point spread function (PSF). The PSF is a matrix that is typically smaller than the image we are blurring and represents which pixels we want to consider when blurring the image. For our purposes we will be using Gaussian blur. Gaussian blur works by weighting each pixel less the further it is from the pixel being blurred.

Now that we have the PSF we need to know how to use to represent it in the form for deblurring $Kf = g$. We can represent convolution as a linear system for a vectorized image by using a n by n Toeplitz matrix where n is the length of the vectorized image. A Toeplitz matrix is a banded diagonal matrix such that we can use it as the K matrix.

Kronecker Products and separable PSF's:

If we are given an image to deblur with dimension n by m our vector with length n times m and our blur matrix will be n times m by n times m. Since most images can have dimension in the hundred or thousands, trying to construct the matrix K a and solve for f is often extremely computationally expensive and for this reason we need a better way to represent the problem so that we can use this process on more realistic larger images. The answer to this problem is to separate the K matrix into its horizontal and vertical components such that we can say K is equal to kronecker product of these two matrices this allows us to write the problem as this:

$$g = A_c x A^T_r$$

Where A_c and A_r represent the horizontal and vertical components of blur matrix K and x and g represent the original and blurred image respectively. We want to do everything we can from constructing the K matrix since it is typically too big to store in regular computer memory. Therefore we should try and construct the A_c and A_r matrices from the PSF as to not construct the K matrix. We can get A_c and A_r by first taking the sparse SVD of the PSF and create A_c and A_r By creating convolution matrices the same shape as the image, convolution matrices are simply Toeplitz matrices that have been shifted such that we can use them for convolution.

Fast Fourier Transforms:

Fast Fourier Transforms (FFT) is an algorithm used to efficiently compute the Discrete Fourier Transform (DFT). DFT is a mathematical transformation that represents a sequence of discrete values as a sum of complex exponential functions. One of the main differences between FFT and DFT is the computational efficiency as DFT has a computational efficiency of

$O(n^2)$ while FFT has a computational efficiency of $O(n \log n)$. FFT helps with image deblurring by applying a deconvolution filter in the frequency domain which means that blurred images can be modeled as the result of a convolution operation between the original image and a blurring kernel which is represented as a multiplication in the frequency domain. After taking the Fourier transform of the blurred image and the blurring kernel, you get their respective frequency domain representations where you can divide the frequency domain representation of the blurred image by the frequency domain representation of the blurring kernel to get the frequency domain representation of the original image.

Tikhonov Regularization:

The first method we will use for image deblurring is Tikhonov Regularization. Tikhonov regularization is a method that we will use to stop the deblurring after a certain point to stop the deblurring from over-fitting to the noise in the image. The formula for Tikhonov regularization is as follows, Where g is the blurred image K is the blur matrix f is the original image and alpha is a parameter that helps impose a penalty for over fitting :

$$min_f\{||g - Kf||^2_2 + \alpha^2||f||^2_2\}$$

Next can rewrite this as

$$\widehat{f} = ||g - Kf||^2_2 + ||\Gamma f||^2_2$$

Where f hat is the approximation of our deblurred image and Γ is equal α multiplied by the identity matrix I lastly we can rewrite it in its explicit form:

$$\widehat{f} = (K^T K + \Gamma^T \Gamma)^{-1} K^T g$$

We could try to directly solve for this if the image is small enough however most images are represented by a fairly large matrix which makes solving directly impractical. Therefore the better way we can do this is by representing the Tikhonov regularization as a fast fourier transformation. First we need to take a fast fourier transform of the PSF(P) and image(g) such that they are the same size. We can then use the following to solve for the f hat using:

$$\widehat{f} = (conj(F(P))/(|F(P)|^2 + \alpha))F(G)$$

Where F(P) and F(G) are the fourier transforms of the PSF and the image respectively and conj is the complex conjugate of a given value. By taking the real values of f hat, we get the approximation . This will allow us to avoid needing to construct the K matrix and cut down the on the amount of computations we need to solve the problem

Truncated SVD:

The Second method we will use is Truncated SVD. normal Singular value decomposition or takes a matrix K and decomposes it into the form $K = U\Sigma V^T$, where U and V are unitary matrices and Σ is diagonal matrix such that $\boldsymbol{\sigma}_1 > \boldsymbol{\sigma}_2 > \boldsymbol{\sigma}_3 \ldots > \boldsymbol{\sigma}_n$ for all values in the matrix. We can solve SVD by taking the matrices KK^T and K^TK and finding their singular values and eigenvalues and eigenvectors. Then we can then take Σ as the square root of eigenvalues and take the columns of U and V as the eigenvectors. With truncated SVD we stop calculating after a certain point replacing the rest of values in the matrices with zeros. The more we truncate the SVD the less we deblur but the less noise we see in the end product. The more $\boldsymbol{\sigma}$ we take into account the more we will deblur the image but the more the noise will get amplified. We can represent truncated SVD with this equation:

$$\hat{f} = \sum_{i=1}^{n}((u_i^T g/\boldsymbol{\sigma}_i)v_i) \equiv K_n^{\dagger} g$$

Where

$$K_n^{\dagger} = \begin{bmatrix} v_1 & \ldots & v_n \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ldots & \\ & & \sigma_n \end{bmatrix} \begin{bmatrix} u_1 \\ \ldots \\ u_n \end{bmatrix}$$

n is a variable that lets us control how much we deblur the image similarly to the Tikhonov regularization the larger n is the more we deblur the image but the more noise is amplified in the image. Trying to brute force solve this problem often takes a unreasonably long time to compute due to the large size of K.However, if the K is separable we can rewrite the problem as

$$\hat{f} = (K_r)_n^{\dagger} G((K_c)_n^{\dagger})^T$$

Where K_r and K_c are the matrices that represent the horizontal and vertical blurring of the image and G is the given image that has not been vectorized. This again helps cut down computation time since we avoid constructing a gigantic K matrix.

Comparing Methods:

It can be difficult to compare the performance of the two methods since the main goal of image deblurring is to regain data that was lost through the process of the image being blurred , which can be a subjective thing in some respects, and the hue of the image can change a bit from the original when deblurring. There are 2 possible ways which we can quantify how well an image deblur worked. The first is mean squared error. We can use this to check how similar the deblurred image is to the original image. However, this is not a great technique since it only tells how similar the deblurred image is to the original image and does not give much information on how much information we recovered from the blurred image. The method we decided to use was to measure sharpness by measuring the sharpness of the image. Sharpness of an image basically tells us how well defined the edges of the object in the image are and can

help quantify how well we deblurred a given image. The last method we can use is Peak Signal-to-Noise Ratio or PSNR of the deblurred image with the original image. This can tell us how much power the noise in the image affects image quality.

Laplacian-based approach:

The method we are using to measure image sharpness to compare the images is the Laplacian-based approach. The Laplacian-based approach involves computing the second order derivative of the image intensity and then using it to quantify the local variations in the image. We do this using the Laplacian operator which is a mathematical operator that calculates the sum of the second-order partial derivatives of the image intensity at each pixel location and is defined as:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Where f is the intensity function and the equations are the second-order partial derivatives of f with respect to x and y.

The resulting Laplacian image highlights areas of the image that have high local variations in intensity, which indicates edges, features, or other sharp details in the image. Each pixel is given a Laplacian value. We then find the variance of all the pixels to find the overall sharpness of the image. The higher the magnitude of the Laplacian value, the sharper the image is.

Peak Signal-to-Noise Ratio:

PSNR measures the ratio of the power of the image and the effect that noise has on distorting the image. The power of the image is the highest value any pixel can have; this will be 255 in our case. To solve this we can use the following formula:

$$PSNR = 10 \log_{10} \left(\frac{MAX_f}{\sqrt{MSE}} \right)$$

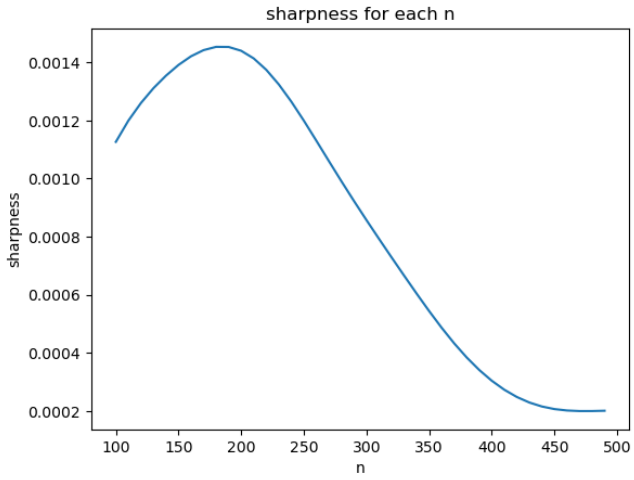
Where MAX f is power of the image 255 and MSE in the mean squared error between the original image and the deburred image.

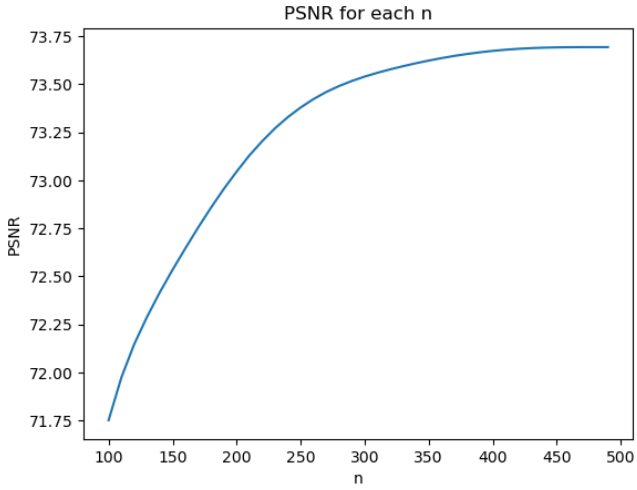
Note: the lower the PSNR value the less noise amplification.

Findings:

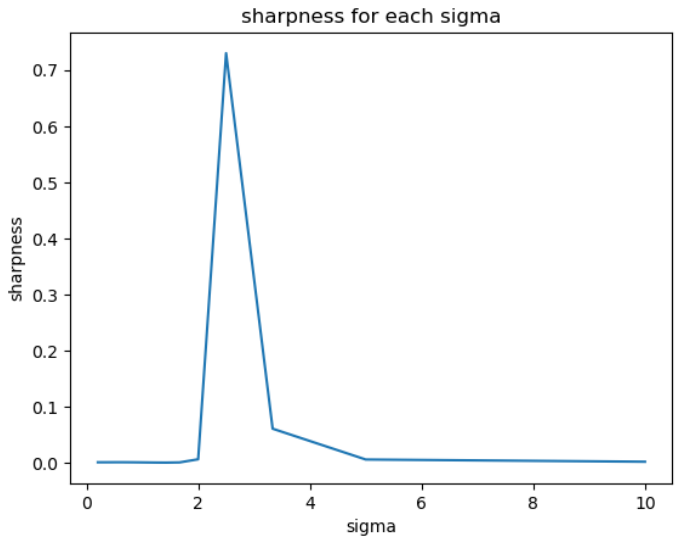
To test the algorithms we will check how different PSF's and deblurring parameters affect the PSNR to measure the effect of noise in the deblur and the laplacian operator to measure the sharpness of the image and plot these on a graph. We will look at the truncated SVD, to check

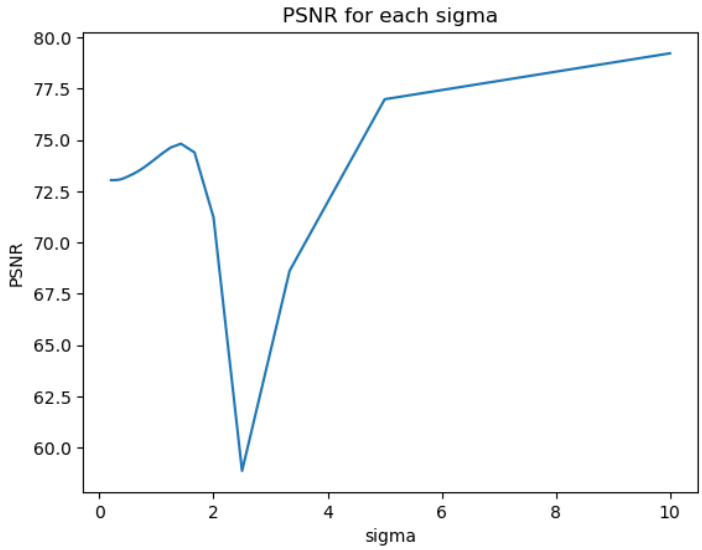
the performance of SVD we will check how different n parameters affect the PSNR and laplacian operator of the image.



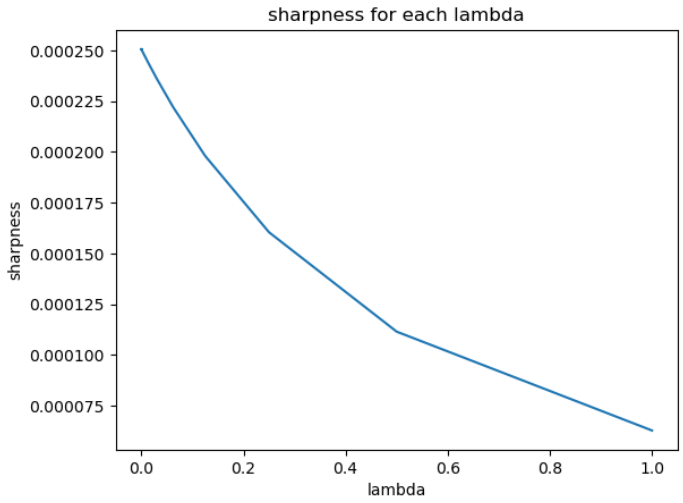


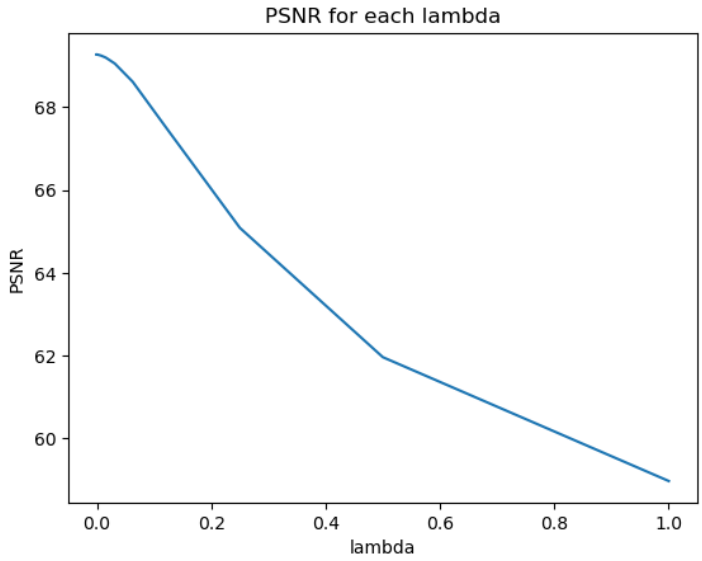
These graphs measure sharpness and the PSNR of the deblurred image with the original given different truncation parameters. We can see from these graphs that for this particular instance the sharpness of the image caps out at around a truncation parameter of about 200 before dipping back down to zero while PSNR graph shoots up quickly from 71.75 to about 73.5 before tapering off. The first graph shows us that there is likely a best parameter to choose for each image and for this particular case the best n is around 200. The PSNR graph shows that the noise amplification has a high rate of change for each rate for the PSNR in early values and low ones in higher values. Show that noise amplification is extremely pronounced in the method.



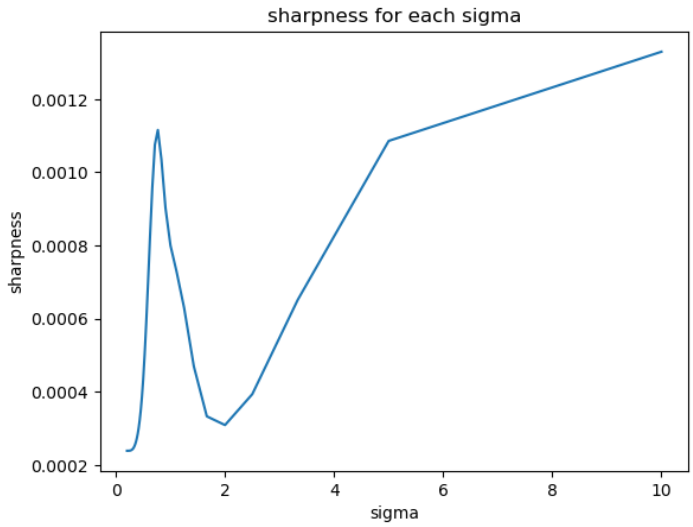


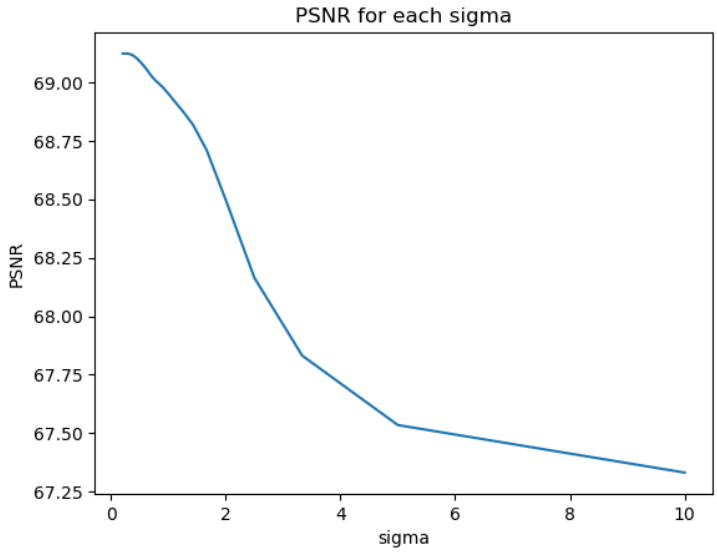
The second set of graphs measure how well the truncated SVD method performs when given A static truncation parameter(n) and PSF's with different sigma values. In these graphs we that the sharpness graph stay close to about 0 before shooting up to 0.7 where lambda is about 2.25, and similarly for the PSNR graph we see that the PSNR starts at around 72.5 beginning to rise before sharply dropping where lambda 2 reaching a minimum soon after an then sharply increasing in value around 2.25. This further indicates that there is an optimal n value for deblurring given a certain parameter since we can seemingly maximize sharpness and minimize noise amplification for a given lambda value.





The above images show how Tikhonov regularization performs given a constant PSF but different lambda values for the deblurring. We can see that as lambda values increase both our sharpness values decrease in a linear fashion. This likely means that there is no optimal lambda value to use for a given PSF and that we should pick our lambda values according to what needs are for a given situation. For example, if the image needs to be extremely sharp, pick a small lambda.





The graphs above illustrate how Tikhonov regularization performs given a static lambda value for the deblurring and different sigmas for the PSF. we can see the PSNR graph goes down the higher the sigma value. The sharpness graph fluctuates widely and as sigma increases, we also tried these graphs with many different lambda values and while the PSNR graph had the same pattern of high to low the sharpness graph always had wildly different patterns, which made it hard to make and conclusion from these graphs.

Using Laplacian-based approach to measure sharpness of the images here are the results:

Image	**Sharpness**
Original image	0.0008
Blurred image	0.0002
Naive solution image	1.12
Tikhonov solution image	0.02
Truncated SVD image	0.04

The results are a bit odd as the sharpness of the original image is extremely low to begin with. In our application, our original image is most likely full of noise already since laplacian is very sensitive to noise which is why laplacian will give a higher value to sharper images. For our application, our threshold would be the original image sharpness. The blurred image provides a sharpness that is lower than the original image which makes sense. However, the naive solution provides a sharpness that is much higher which is probably because the naive solution also produces a brand new picture that doesn't resemble the original image at all. We can also see that Tikhonov and Truncated SVD do produce an image that resembles the original image and both have a higher sharpness which would pass the threshold and be considered unblurred. In conclusion, truncated SVD has a higher sharpness value compared to Tikhonov so truncated SVD is a better method for deblurring images.

Conclusion:

Image deblurring is a useful technique that has many applications in astronomy, medicine and many other fields of study. In real life we would likely know what is causing an image to become blurred by not how much noise is in the image. For example when we take a picture of space from earth it becomes blurred by interference from the atmosphere, and since we can approximate a PSF of atmospheric blur we can estimate what the image could look like without the blur which is useful to help us see things we normally wouldn't be able to see. Overall both methods performed fairly well, however since truncated SVD seemingly has optimal parameters for each situation and had higher sharpness on average it is likely the better method to use in most situations.

References:

Society for Industrial & Applied Mathematics, U.S. (2007). *Deblurring images: Matrices, Spectra, and filtering*.

NPTEL-NOC IITM. (2021). *Lec 77 - Tikhonov-Miller Regularization*. Retrieved April 9, 2023, from https://www.youtube.com/watch?v=GkN90Rt063M&ab_channel=NPTEL-NOCIITM.

McMullen, Jamie, "Deblurring Images" (2018). WWU Honors Program Senior Projects. 100. https://cedar.wwu.edu/wwu_honors/100

(2018, September 1). *Image deconvolution using Tikhonov regularization*. http://news.zahlt.info/en/optimization/image-deconvolution-using-tikhonov-regularization/

Atul, K. &, & Atul, K. &. (2021, October 30). *Blur detection using the variance of the Laplacian method*. TheAILearner. https://theailearner.com/2021/10/30/blur-detection-using-the-variance-of-the-laplacian-method/

(n.d.). *Lecture 7 - Tikhonov regularization with SVD | University of Helsinki*. Retrieved April 9, 2023, from https://www.helsinki.fi/en/unitube/video/bf6448ee-ff5d-4c0d-bbb6-3a6f77a47251.