



Western Washington University
Western CEDAR

WWU Honors Program Senior Projects

WWU Graduate and Undergraduate Scholarship


Fall 2018

Deblurring Images

Jamie McMullen

Western Washington University

Follow this and additional works at: https://cedar.wvu.edu/wwu_honors

 Part of the [Algebra Commons](#), and the [Higher Education Commons](#)

Recommended Citation

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

This Project is brought to you for free and open access by the WWU Graduate and Undergraduate Scholarship at Western CEDAR. It has been accepted for inclusion in WWU Honors Program Senior Projects by an authorized administrator of Western CEDAR. For more information, please contact westerncedar@wwu.edu.

Deblurring Images

Senior Project

Jamie McMullen

Advisor: Tjalling Ypma

Abstract

Let the matrix B be a blurred version of a sharp image represented by the matrix X . Given B , we would like to recover X .

To accomplish this, we construct linear models of the blurring process that produced B from X . The idea is that we could then reverse the blurring to reproduce the original image.

For example, if the blurred image satisfies

$$B = CXR^T$$

for some invertible matrices C and R , then we could recover X as

$$X = C^{-1}B(R^T)^{-1}.$$

However, the blurring model usually fails to account for all the blurring that actually occurred. Likely, the blurred image actually satisfies a relation like

$$B = CXR^T + E$$

where E is a matrix representing random errors and other blurring effects not accounted for by the model.

If we were to proceed as above, we would produce

$$C^{-1}B(R^T)^{-1} = X + C^{-1}E(R^T)^{-1}.$$

The term $C^{-1}E(R^T)^{-1}$ often severely compromises the accuracy of the clear image X . We will explore ways to modify the reconstruction process to produce an image close to X that minimizes contamination by the error term E .

This report is comprised of three parts. In the first, we examine the construction of blurring models, in the second we discuss methods of deblurring images using these models, and in the third we will work with an example photograph to illustrate the deblurring process.

The mathematical techniques we use include the singular value decomposition, matrix norms, certain matrix structures such as Kronecker products, and related theorems. The relevant details of these topics are provided in the appendix.

1. Constructing a Model of Blurring

A digital image is represented by matrices, with entries corresponding to the hues of each pixel in the image. Grayscale images need only one matrix, with numerical entries ranging on a predetermined scale from black to white. Color images require three matrices, one for each of the red, green, and blue (RGB) components of each pixel.

Say we have a grayscale digital image that is unclear, or blurred, and we want to sharpen this image. We call the matrix representing the given blurry image B , and we define the matrix X to be the sharp, clear version of B . We vectorize the matrix X by concatenating its columns, in order, to create one long column vector. This creates $\vec{x} = \text{vec}(X)$. We create the vector $\vec{b} = \text{vec}(B)$ similarly. Then, a linear system representing the blurring process is given by

$$A\vec{x} = \vec{b}$$

Our first goal is to construct the blurring matrix A . We will begin by examining the concept of a point spread function and its utilization in the creation of our blurring matrix.

Point Spread Functions

A point spread function (PSF) establishes how a single pixel, the point source, is spread through the image when blurring occurs. The point spread function dictates what portion of the point source ends up in each of the pixels around it. Here is one simple example of a point spread function:

$$\text{Clear} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \longrightarrow \text{Blurred} \begin{bmatrix} 0 & \frac{1}{8} & 0 \\ \frac{1}{8} & \frac{1}{2} & \frac{1}{8} \\ 0 & \frac{1}{8} & 0 \end{bmatrix}$$

This visual representation of the PSF is called a point spread array. Note that the entries in the blurred version add up to 1.

When the PSF is identical for all pixels in an image, the blurring is said to be spatially invariant. In this report, we assume spatial invariance.

The PSF is usually expressed as a mapping from the point source to each pixel of its array. For example, circular blurring from an out-of-focus lens is represented as

$$p_{ij} = \begin{cases} \frac{1}{\pi r^2} & \text{if } (i - k)^2 + (j - l)^2 \leq r^2 \\ 0 & \text{elsewhere} \end{cases}$$

Here, the center of the array is at (k, l) and the radius of the blur is r .

Another example is Gaussian blur, given by

$$p_{ij} = e^{-\frac{1}{2} \begin{bmatrix} i-k \\ j-l \end{bmatrix}^T \begin{bmatrix} s_1^2 & \rho^2 \\ \rho^2 & s_2^2 \end{bmatrix}^{-1} \begin{bmatrix} i-k \\ j-l \end{bmatrix}}$$

with s_1, s_2 , and ρ determining the width and orientation of the PSF, and center (k, l) . When $s_1 = s_2$ and $\rho = 0$, row and column blurring become independent of one another. This is an example of separable blurring, where the term “separable” reflects that since row and column blurring are independent, the PSF can be expressed as

$$P = \vec{c} \vec{r}^T = \begin{bmatrix} c_1 \\ \vdots \\ c_m \end{bmatrix} [r_1 \quad \cdots \quad r_n].$$

As the point source is represented by an $m \times n$ matrix of all zeros other than a single pixel of value 1, its vectorized version is simply the i th unit vector in \mathbb{R}^{mn} . We denote this i th unit vector as \vec{e}_i , not to be confused with a column of the error matrix E discussed later.

We observe that since $A\vec{x} = \vec{b}$, the blur of a single pixel is given by $A\vec{e}_i$, which is equivalent to the i th column of A . Hence, once we have established the point spread function, we construct the matrix A columnwise by vectorizing the point spread array. Because of spatial invariance, we can construct A by placing the appropriate shift of this vector in each column of A . For example, with the first 3×3 example of a point spread function, we obtain:

$$A = \begin{bmatrix} \ddots & \ddots & & & & & & & & & \\ \ddots & \frac{1}{2} & \frac{1}{8} & 0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} & 0 & \frac{1}{8} & \\ & \frac{1}{8} & \frac{1}{2} & \frac{1}{8} & 0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} & 0 & \\ & 0 & \frac{1}{8} & \frac{1}{2} & \frac{1}{8} & 0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \\ & \frac{1}{8} & 0 & \frac{1}{8} & \frac{1}{2} & \frac{1}{8} & 0 & \frac{1}{8} & 0 & 0 & \\ & 0 & \frac{1}{8} & 0 & \frac{1}{8} & \frac{1}{2} & \frac{1}{8} & 0 & \frac{1}{8} & 0 & \\ & \frac{1}{8} & 0 & 0 & \frac{1}{8} & 0 & \frac{1}{8} & \frac{1}{2} & \frac{1}{8} & 0 & \\ & 0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} & 0 & \frac{1}{8} & \frac{1}{2} & \frac{1}{8} & \\ & \frac{1}{8} & 0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} & 0 & \frac{1}{8} & \frac{1}{2} & \ddots \\ & & & & & & & & & \ddots & \ddots \end{bmatrix}$$

To compute the blurry image B pixel by pixel, we work with its vectorized version. The entry b_i of \vec{b} is given by $b_i = \vec{e}_i^T \vec{b} = \vec{e}_i^T A\vec{x} = (i$ th row of $A)\vec{x}$. Note that here we are working with the rows, not the columns, of A . Each blurred pixel is then a weighted average of its surrounding pixels, whose weights are given by the rows of A . This is called convolution.

Boundary Conditions

What about the pixels on or near the edge of a blurred image? In principle, the pixels near the edge of a blurred image are affected by the pixels beyond the boundary of the exact image. This is because each pixel of the blurred image is a weighted average of a pixel with its neighboring pixels, called convolution. To cope with this, assumptions have to be made about the pixels surrounding the outer boundary of the exact image. These assumptions are called boundary conditions. When boundary conditions are assumed, our representation of the exact image is extended beyond its original borders. The idea is that the blurred image arises from blurring applied to this extended image.

The simplest boundary condition is called the zero boundary condition, in which we assume that all pixels beyond the dimensions of the exact image have entries 0. Visually, this represents an assumption that everything outside of the field of view of the exact image is black. If X is our exact image, then we impose zero boundary conditions by padding the matrix with zeros, creating

$$X_{ext} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & X & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Zero boundary conditions are very easy to work with, but they tend to cause black rings around the inner borders of pictures upon deblurring, much like a vignette effect.

One alternative method that avoids black vignetting is the periodic boundary condition, in which the clear image is assumed to repeat outside of itself in all directions. In this case,

$$X_{ext} = \begin{bmatrix} X & X & X \\ X & X & X \\ X & X & X \end{bmatrix}$$

A third option is to impose reflexive boundary conditions, in which the clear image is reflected outside of itself in all directions. Here, a vertical flip of X is represented by X_V , a horizontal flip by X_H , and a flip in both directions by $X_{VH} = X_{HV}$. Then

$$X_{ext} = \begin{bmatrix} X_{VH} & X_V & X_{VH} \\ X_H & X & X_H \\ X_{VH} & X_V & X_{VH} \end{bmatrix}$$

Here is one simple 2×2 example of the reflexive boundary condition:

$$X = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, X_{ext} = \begin{bmatrix} 4 & 3 & 3 & 4 & 4 & 3 \\ 2 & 1 & 1 & 2 & 2 & 1 \\ 2 & 1 & 1 & 2 & 2 & 1 \\ 4 & 3 & 3 & 4 & 4 & 3 \\ 4 & 3 & 3 & 4 & 4 & 3 \\ 2 & 1 & 2 & 1 & 2 & 1 \end{bmatrix}$$

We create an extended exact image \vec{x}_{ext} , with the entries w_1, w_2, y_1, y_2 representing pixels that are outside the field of view of the actual image (on the boundary):

$$\vec{x}_{ext} = \begin{bmatrix} w_1 \\ w_2 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ y_1 \\ y_2 \end{bmatrix}$$

Then, convolution is represented as

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} p_5 & p_4 & p_3 & p_2 & p_1 & 0 & 0 & 0 & 0 \\ 0 & p_5 & p_4 & p_3 & p_2 & p_1 & 0 & 0 & 0 \\ 0 & 0 & p_5 & p_4 & p_3 & p_2 & p_1 & 0 & 0 \\ 0 & 0 & 0 & p_5 & p_4 & p_3 & p_2 & p_1 & 0 \\ 0 & 0 & 0 & 0 & p_5 & p_4 & p_3 & p_2 & p_1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ y_1 \\ y_2 \end{bmatrix} .$$

Notice that pixels from outside the true image impact pixels near the edge of the blurred image. This is why choosing appropriate boundary conditions is so important.

A convolution computation is condensed depending on the selected boundary conditions. In this example, with zero boundary conditions, $w_1, w_2, y_1, y_2 = 0$ and the A matrix becomes a 5×5 Toeplitz matrix:

$$A = \begin{bmatrix} p_3 & p_2 & p_1 & 0 & 0 \\ p_4 & p_3 & p_2 & p_1 & 0 \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ 0 & p_5 & p_4 & p_3 & p_2 \\ 0 & 0 & p_5 & p_4 & p_3 \end{bmatrix}$$

With periodic boundary conditions, $w_1 = x_4, w_2 = x_5, y_1 = x_1, y_2 = x_2$. In this case, the A matrix becomes a 5×5 circulant matrix:

$$A = \begin{bmatrix} p_3 & p_2 & p_1 & p_5 & p_4 \\ p_4 & p_3 & p_2 & p_1 & p_5 \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ p_1 & p_5 & p_4 & p_3 & p_2 \\ p_2 & p_1 & p_5 & p_4 & p_3 \end{bmatrix}$$

With reflexive boundary conditions, we have $w_1 = x_2, w_2 = x_1, y_1 = x_5, y_2 = x_4$. Then, A becomes a 5×5 matrix that is the sum of a Toeplitz and a Hankel matrix, a Toeplitz-plus-Hankel matrix:

$$A = \begin{bmatrix} p_3 & p_2 & p_1 & 0 & 0 \\ p_4 & p_3 & p_2 & p_1 & 0 \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ 0 & p_5 & p_4 & p_3 & p_2 \\ 0 & 0 & p_5 & p_4 & p_3 \end{bmatrix} + \begin{bmatrix} p_4 & p_5 & 0 & 0 & 0 \\ p_5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & p_1 \\ 0 & 0 & 0 & p_1 & p_2 \end{bmatrix}$$

Two-dimensional convolution is analogous to one-dimensional convolution, except that the clear and blurred images are represented by matrices rather than vectors. For example, if we have a clear 3×3 image and corresponding point spread function,

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}, P = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}, B = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

where X and P are known and we want to find B , if we assume zero boundary conditions we vectorize X and B and obtain

$$\begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{22} \\ b_{23} \\ b_{31} \\ b_{32} \\ b_{33} \end{bmatrix} = \begin{bmatrix} p_{22} & p_{12} & 0 & p_{21} & p_{11} & 0 & 0 & 0 & 0 \\ p_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} & 0 & 0 & 0 \\ 0 & p_{32} & p_{22} & 0 & p_{31} & p_{21} & 0 & 0 & 0 \\ p_{23} & p_{13} & 0 & p_{22} & p_{12} & 0 & p_{21} & p_{11} & 0 \\ p_{33} & p_{23} & p_{13} & p_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} \\ 0 & p_{33} & p_{23} & 0 & p_{32} & p_{22} & 0 & p_{31} & p_{21} \\ 0 & 0 & 0 & p_{23} & p_{13} & 0 & p_{22} & p_{12} & 0 \\ 0 & 0 & 0 & p_{33} & p_{23} & p_{13} & p_{32} & p_{22} & p_{12} \\ 0 & 0 & 0 & 0 & p_{33} & p_{23} & 0 & p_{32} & p_{22} \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{23} \\ x_{31} \\ x_{32} \\ x_{33} \end{bmatrix}$$

We observe that the structure of the large matrix A is block Toeplitz with 3×3 Toeplitz blocks (BTTB). In the case of periodic boundary conditions, A is block circulant with circulant blocks (BCCB), and with reflexive boundary conditions A is a sum of block Toeplitz with Toeplitz blocks (BTTB), block Toeplitz with Hankel blocks (BTHB), block Hankel with Toeplitz blocks (BHTB), and block Hankel with Hankel blocks (BHHB) matrices.

Separable Blurring

In some cases, blurring occurs such that the blurring of the columns of the clear image is independent of the blurring of the rows. This is called separable blurring. At the matrix level, when blurring is separable, we can relate the blurry image B and the clear image X using invertible matrices A_c and A_r . We express the relationship as $A_c X A_r^T = B$, where A_c transforms the columns of X and A_r^T transforms the rows.

To understand why A_c affects the columns and A_r^T affects the rows, we observe that if $X = [\vec{x}_1 \cdots \vec{x}_n]$, then $A_c X = A_c [\vec{x}_1 \cdots \vec{x}_n] = [A_c \vec{x}_1 \cdots A_c \vec{x}_n]$. The structure behind $X A_r^T$ is analogous. Of course, since matrix multiplication is associative, it does not matter which

blurring operation is applied to X first; the blurred image B will be the same.

It is easy to see that in the vectorized form, $B = A_c X A_r^T$ corresponds to $\vec{b} = A \vec{x}$, where the blurring matrix A is given by the Kronecker product $A = A_r \otimes A_c$.

In the case of a separable blur, we decompose the $m \times n$ point spread array P into two vectors, such that $P = \vec{c}(\vec{r}^T)$, where \vec{c} is $m \times 1$ and represents blur across the columns of an image, and \vec{r} is $n \times 1$ and represents blur across the rows. In this case, P has rank 1, with $p_{ij} = c_i r_j$. Assuming spatial invariance, if we replace each p_{ij} in the $mn \times mn$ matrix A by its corresponding $c_i r_j$, we observe that A takes the form of a Kronecker product $A = A_r \otimes A_c$, where A_r is $n \times n$, with entries a pattern of the r_i , while A_c is $m \times m$ and has entries a pattern of the c_i . For example:

$$A = \begin{bmatrix} r_2 \begin{bmatrix} c_2 & c_1 \\ c_3 & c_2 & c_1 \\ & c_3 & c_2 \end{bmatrix} & r_1 \begin{bmatrix} c_2 & c_1 \\ c_3 & c_2 & c_1 \\ & c_3 & c_2 \end{bmatrix} & 0 \\ r_3 \begin{bmatrix} c_2 & c_1 \\ c_3 & c_2 & c_1 \\ & c_3 & c_2 \end{bmatrix} & r_2 \begin{bmatrix} c_2 & c_1 \\ c_3 & c_2 & c_1 \\ & c_3 & c_2 \end{bmatrix} & r_1 \begin{bmatrix} c_2 & c_1 \\ c_3 & c_2 & c_1 \\ & c_3 & c_2 \end{bmatrix} \\ 0 & r_3 \begin{bmatrix} c_2 & c_1 \\ c_3 & c_2 & c_1 \\ & c_3 & c_2 \end{bmatrix} & r_2 \begin{bmatrix} c_2 & c_1 \\ c_3 & c_2 & c_1 \\ & c_3 & c_2 \end{bmatrix} \end{bmatrix}$$

The ability to express A_r and A_c based on single vectors \vec{r} and \vec{c} is a direct consequence of our assumption of spatial invariance. If we have our matrix A_c acting on X such that every pixel in X is smeared out by A_c the exact same way, then the columns of A_c are shifted versions of each other. Then, A_c is characterized by just one of its columns, namely \vec{c} . Similarly, A_r is characterized by one vector \vec{r} .

A_r and A_c take on particular structures based on the chosen boundary conditions. With zero boundary conditions, A_r and A_c are both Toeplitz matrices as in the example above. With periodic boundary conditions, A_r and A_c are circulant matrices, and with reflexive boundary conditions A_r and A_c are Toeplitz-plus-Hankel matrices.

If the point spread function is separable, then it is easy to find a \vec{c} and \vec{r} such that $P = \vec{c} \vec{r}^T$. To do so, we take the singular value decomposition $P = U \Sigma V^T$ and use the largest (and only nonzero) singular value, σ_1 , of P . Then $\vec{c} = \sqrt{\sigma_1} \vec{u}_1$ and $\vec{r} = \sqrt{\sigma_1} \vec{v}_1$. From here, we represent P in terms of the c_i and r_i , and construct A based on the assumed boundary conditions, with $A = A_r \otimes A_c$.

2. Deblurring an Image

Error

If we have $A\vec{x} = \vec{b}$ and wish to recover the clear image \vec{x} , we write $\vec{x} = A^{-1}\vec{b}$, and if we know the matrix A then we are done. However, this simple approach is flawed and may lead to even worse blurring than before.

The reason that this “naïve” solution does not work is that the model $A\vec{x} = \vec{b}$ is usually not quite correct. This model does not account for noise or other errors not captured by this model. Often the neglect of noise in the naïve solution leads to an even worse quality image than before, because the noise may get significantly blown up during the naïve deblurring. This may be seen as follows.

Using the singular value decomposition,

$$A = U\Sigma V^T = \sum_{i=1}^{mn} \sigma_i \vec{u}_i \vec{v}_i^T,$$

so

$$A^{-1} = V\Sigma^{-1}U^T = \sum_{i=1}^{mn} \frac{1}{\sigma_i} \vec{v}_i \vec{u}_i^T.$$

Then

$$\vec{x}_{\text{naïve}} = A^{-1}\vec{b} = V\Sigma^{-1}U^T\vec{b} = \sum_{i=1}^{mn} \frac{\vec{u}_i^T \vec{b}}{\sigma_i} \vec{v}_i.$$

As i increases, more sign changes tend to occur in the entries of the vectors \vec{u}_i and \vec{v}_i . In other words, the information frequency of the vectors increases as i increases. Low frequency information corresponds to more significant, major features of the image, while high frequency information corresponds to fine detail, and often noise. We want low frequency information to dominate the solution in order to capture the overall features of the image. To accomplish this, $|\vec{u}_i^T \vec{b}|$ must decay faster than the σ_i . This is called the Discrete Picard Condition, and is always satisfied by the blurring matrices constructed in image deblurring problems.

A more accurate representation for the image blurring process is $\vec{b} = A\vec{x} + \vec{e}$, where \vec{e} is the vector accounting for the inevitable error. Then when the naïve solution is applied, we really obtain $\vec{x}_{\text{naïve}} = A^{-1}\vec{b} = A^{-1}(A\vec{x} + \vec{e}) = \vec{x} + A^{-1}\vec{e}$. If the second term, the inverted noise term, has large values, then the noise may dominate the solution and the reconstructed image will potentially have very low quality. Using SVD, this is represented by

$$\begin{aligned} \vec{x}_{\text{naïve}} &= \vec{x} + A^{-1}\vec{e} \\ &= \vec{x} + V\Sigma^{-1}U^T\vec{e} \\ &= \vec{x} + \sum_{i=1}^{mn} \frac{\vec{u}_i^T \vec{e}}{\sigma_i} \vec{v}_i. \end{aligned}$$

The inverted noise term of the naïve solution typically displays a few significant properties. The $|\vec{u}_i^T \vec{e}|$ stay fairly consistent for all i . As i increases, σ_i decreases, causing $\frac{1}{\sigma_i}$ to increase. Thus, as i increases, the terms $\frac{\vec{u}_i^T \vec{e}}{\sigma_i}$ become very large in magnitude and thus the error component \vec{e} becomes overrepresented. These effects combined allow for the error term to dominate the naïve solution.

Spectral Filtering

In order to control the error term, it is necessary to limit the impact of the terms $\frac{\vec{u}_i^T \vec{e}}{\sigma_i}$ for small σ_i . One way to do this is through spectral filtering.

Spectral filtering, or regularization, is when the computed solution \vec{x} has the form $\vec{x}_{\text{filt}} = \sum_{i=1}^{mn} \phi_i \frac{\vec{u}_i^T \vec{b}}{\sigma_i} \vec{v}_i$, with filter factors $\phi_i \approx 1$ for large σ_i and $\phi_i \approx 0$ for small σ_i .

Truncated SVD

One example of spectral filtering is the truncated singular value decomposition. This method simply discards all components of the singular value decomposition of A that are dominated by noise. That is, take $\vec{x}_k = \sum_{i=1}^{k < mn} \frac{\vec{u}_i^T \vec{b}}{\sigma_i} \vec{v}_i$. In this case, $\phi_i = 1$ for $i \leq k$ and $\phi_i = 0$ for $i > k$. Equivalently, we could replace σ_i by 0 if $|\sigma_i| \leq \varepsilon$, where ε is some selected threshold value.

Equivalently, to combat the dominance by high-frequency information, we are truncating the matrices U and V to include only the singular vectors \vec{u}_i and \vec{v}_i with low enough indices. When a suitable end-index position k has been chosen, the rank- k approximation $A_k = \sum_{i=1}^k \sigma_i \vec{u}_i \vec{v}_i^T$ given by the Eckart-Young-Mirsky Theorem is substituted for our blurring matrix A . This replacement is effective in eliminating the greatly magnified high-frequency singular vectors, and thus serves to curb the inflation of error in our solution.

Of course, there is a balance in selecting the appropriate k value. If k is too high, noise will still dominate the solution; but if k is too low, too much basic information will be lost. The scalar quantities $|\vec{u}_i^T \vec{b}|$ of A tend to lessen as i increases, but plateau off eventually due to noise. We demonstrate this in an example at the end of this report. The wider the blur, the higher the level of this noise plateau occurs, and the faster the singular values of A decline to zero. The value of k chosen for a truncated singular value decomposition should be the last index at which the singular value is greater than the coefficient $|\vec{u}_i^T \vec{b}|$.

Tikhonov Regularization

Another form of regularization is Tikhonov filtering. Here, $\phi_i = \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2}$ where $\alpha > 0$ acts as the regularization parameter.

Our filtered solution then satisfies

$$\begin{aligned}
\vec{x}_\alpha &= \sum_{i=1}^n \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \frac{\vec{u}_i^T \vec{b}}{\sigma_i} \vec{v}_i \\
&= V(\Sigma^2 + \alpha^2 I)^{-1} \Sigma U^T \vec{b} \\
&= V(\Sigma^2 + \alpha^2 I)^{-1} V^T V \Sigma U^T \vec{b} \\
&= (V \Sigma^2 V^T + \alpha^2 V V^T)^{-1} V \Sigma U^T \vec{b} \\
&= (A^T A + \alpha^2 I)^{-1} A^T \vec{b}.
\end{aligned}$$

This is true if and only if $(A^T A + \alpha^2 I) \vec{x} = A^T \vec{b}$, and thus $\begin{bmatrix} A \\ \alpha I \end{bmatrix}^T \begin{bmatrix} A \\ \alpha I \end{bmatrix} \vec{x} = \begin{bmatrix} A \\ \alpha I \end{bmatrix}^T \begin{bmatrix} \vec{b} \\ \vec{0} \end{bmatrix}$.

Thus \vec{x}_α solves the least-squares problem $\min_{\vec{x}} \left\| \begin{bmatrix} A \\ \alpha I \end{bmatrix} \vec{x} - \begin{bmatrix} \vec{b} \\ \vec{0} \end{bmatrix} \right\|_2$.

We use the fact that for any vectors \vec{p} and \vec{q} ,

$$\left\| \begin{bmatrix} \vec{p} \\ \vec{q} \end{bmatrix} \right\|_2^2 = \begin{bmatrix} \vec{p} \\ \vec{q} \end{bmatrix}^T \begin{bmatrix} \vec{p} \\ \vec{q} \end{bmatrix} = \vec{p}^T \vec{p} + \vec{q}^T \vec{q} = \|\vec{p}\|_2^2 + \|\vec{q}\|_2^2$$

to rewrite our solution as

$$\vec{x}_\alpha = \min_{\vec{x}} \left\| \begin{bmatrix} A \\ \alpha I \end{bmatrix} \vec{x} - \begin{bmatrix} \vec{b} \\ \vec{0} \end{bmatrix} \right\|_2 = \min_{\vec{x}} \{ \|A\vec{x} - \vec{b}\|_2^2 + \alpha^2 \|\vec{x}\|_2^2 \}$$

So, the Tikhonov method provides the solution to the minimization problem $\min_{\vec{x}} \{ \|A\vec{x} - \vec{b}\|_2^2 + \alpha^2 \|\vec{x}\|_2^2 \}$. The reason that this specific minimization problem presents itself is that of course we want a small $\|A\vec{x} - \vec{b}\|_2^2$, but if it were minimized all the way to zero then we would have $A\vec{x} = \vec{b}$, running into the naïve solution $\vec{x} = A^{-1} \vec{b}$ with lots of noise blowup present and large $\|\vec{x}\|_2^2$. If we keep $\|\vec{x}\|_2^2$ small as well, then noise is kept under control. The regularization parameter α determines how heavily important one norm component is in comparison to the other.

Choosing an appropriate value for α depends on the singular values σ_i of the blurring matrix. A series expansion shows that

$$\phi_i = \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} = \begin{cases} 1 - \left(\frac{\alpha}{\sigma_i}\right)^2 + \mathcal{O}\left(\left(\frac{\alpha}{\sigma_i}\right)^4\right) & \sigma_i \gg \alpha \\ \left(\frac{\sigma_i}{\alpha}\right)^2 + \mathcal{O}\left(\left(\frac{\sigma_i}{\alpha}\right)^4\right) & \sigma_i \ll \alpha \end{cases}$$

We can infer that a reasonable α satisfies $\sigma_{mn} \ll \alpha \ll \sigma_1$. For early indices, we have $\sigma_i \gg \alpha$, so $\phi_i \approx 1$. For late indices when $\sigma_i \ll \alpha$, we have $\phi_i \approx \frac{\sigma_i^2}{\alpha^2} \approx 0$. Of course, the turning point occurs when $\sigma_i \approx \alpha$.

Error in Spectral Filtering

Recall that the naïve solution is described by $\vec{x}_{\text{naïve}} = A^{-1}\vec{b} = V\Sigma^{-1}U^T\vec{b}$. In general, a solution obtained by spectral filtering can be written as $\vec{x}_{\text{filt}} = V\Phi\Sigma^{-1}U^T\vec{b}$, where Φ is a diagonal matrix with entries ϕ_i . Now,

$$\begin{aligned}\vec{x}_{\text{filt}} &= V\Phi\Sigma^{-1}U^T\vec{b} \\ &= V\Phi\Sigma^{-1}U^T(\vec{b}_{\text{exact}} + \vec{e}) \\ &= V\Phi\Sigma^{-1}U^T\vec{b}_{\text{exact}} + V\Phi\Sigma^{-1}U^T\vec{e} \\ &= V\Phi\Sigma^{-1}U^T A\vec{x}_{\text{exact}} + V\Phi\Sigma^{-1}U^T\vec{e} \\ &= V\Phi V^T\vec{x}_{\text{exact}} + V\Phi\Sigma^{-1}U^T\vec{e}.\end{aligned}$$

Consequently, the error in spectral filtering solutions is $\vec{x}_{\text{exact}} - \vec{x}_{\text{filt}} = (I - V\Phi V^T)\vec{x}_{\text{exact}} - V\Phi\Sigma^{-1}U^T\vec{e}$. The two terms respectively represent different types of error, regularization error and perturbation error.

Regularization error, represented by $(I - V\Phi V^T)\vec{x}_{\text{exact}}$, is caused by using the regularized inverse matrix $V\Phi\Sigma^{-1}U^T$ instead of the original inverse matrix $A^{-1} = V\Sigma^{-1}U^T$. $V\Phi V^T$ describes the mapping between \vec{x}_{exact} and \vec{x}_{filt} . Thus if the diagonal entries of Φ are quite small, causing Φ to approach the zero matrix, regularization error becomes huge. On the contrary, as Φ approaches I_{mn} , $V\Phi V^T \rightarrow VV^T = I_{mn}$, causing the regularization error to shrink to zero.

Perturbation error, represented by $V\Phi\Sigma^{-1}U^T\vec{e}$, consists of inverted and filtered noise. As Φ approaches zero, the perturbation error shrinks to zero. This manifests as an undersmoothed solution image. But as Φ approaches I_{mn} , we have $V\Phi\Sigma^{-1}U^T\vec{e} \rightarrow V\Sigma^{-1}U^T\vec{e} = A^{-1}\vec{e}$, creating large perturbation error. This is the case when the solution image is oversmoothed.

The reason that it is possible to compute reasonable approximations to exactly sharp images by filtering is due to deblurring problems satisfying the Discrete Picard Condition. To understand this, we consider the norm of the regularization error:

$$\begin{aligned}& \|(I - V\Phi V^T)\vec{x}_{\text{exact}}\|_2^2 \\ &= \|(I - \Phi)V^T\vec{x}_{\text{exact}}\|_2^2 \\ &= \|(I - \Phi)\Sigma^{-1}U^T\vec{b}_{\text{exact}}\|_2^2 \\ &= \sum_{i=1}^{mn} \left((1 - \phi_i) \frac{\vec{u}_i^T \vec{b}_{\text{exact}}}{\sigma_i} \right)^2.\end{aligned}$$

Since the Discrete Picard Condition is satisfied, the coefficients $|\frac{\vec{u}_i^T \vec{b}_{\text{exact}}}{\sigma_i}|$ rapidly decay on average. For small i , the filter factors ϕ_i are close to one, so $(1 - \phi_i)$ dampen the larger early coefficients. For large i , filter factors are quite small, causing $(1 - \phi_i)$ to approach one, so the smaller coefficients are not as damped. This provides balance in the regularization error, preserving a reasonable error norm.

We see now that one type of error shrinks as the other grows. The two kinds of error are most balanced when the appropriate regularization parameter value is chosen.

One way to decide on the most suitable regularization parameter is using the L-Curve Criterion. Often shaped like an L, the L-curve is a log-log plot of the norm $\|A\vec{x}_{\text{filt}} - \vec{b}\|_2$ of the regularized solution versus the corresponding residual norm $\|\vec{x}_{\text{filt}}\|_2$ for different parameter values. In TSVD, an excessively low choice of k causes the residual norm to grow too large, and an excessively high k causes the solution norm to be too large. In Tikhonov regularization, too low a choice of α causes the solution norm to be too large but too high a choice of α causes too great an increase in the residual norm. Since we wish to avoid either norm being too large, the optimum parameter value is at the corner of the L-curve.

Separable Deblurring

When blurring is separable, it is helpful to exploit the singular value decompositions $A_r = U_r \Sigma_r V_r^T$ and $A_c = U_c \Sigma_c V_c^T$, where the properties of Kronecker products gives us that

$$A = A_r \otimes A_c = (U_r \otimes U_c)(\Sigma_r \otimes \Sigma_c)(V_r \otimes V_c)^T.$$

This itself resembles a singular value decomposition, with the exception that the diagonal entries of $\Sigma_r \otimes \Sigma_c$ are not necessarily in nonincreasing order. However, we do not need to construct these Kronecker products.

In matrix form, $X_{\text{naïve}} = A_c^{-1} B A_r^{-T}$. Substituting the SVD, we obtain

$$X_{\text{naïve}} = V_c \Sigma_c^{-1} U_c^T B U_r \Sigma_r^{-1} V_r^T.$$

To compute X_{filt} , we define $\vec{\sigma}_c$ to consist of the entries on the main diagonal of Σ_c , and define $\vec{\sigma}_r$ similarly in terms of Σ_r .

Note that the products of these entries are the singular values of A . An $m \times n$ matrix containing these mn values can be computed as $S = \vec{\sigma}_c \vec{\sigma}_r^T$. We collect the corresponding filter factors in an $m \times n$ matrix Φ .

Let $S_{\text{filt}} = \Phi ./ S$ where $./$ denotes elementwise division.

Then $X_{\text{filt}} = V_c ((U_c^T B U_r) .* S_{\text{filt}}) V_r^T$, where $.*$ denotes elementwise multiplication.

This construction of X_{filt} is especially useful when deblurring using numerical computing software such as MATLAB, which have built-in operations for elementwise multiplication and division. This lets us avoid working with the huge matrix A .

3. Example: Lunar Photograph

We examine the deblurring process using a crop of the following image:

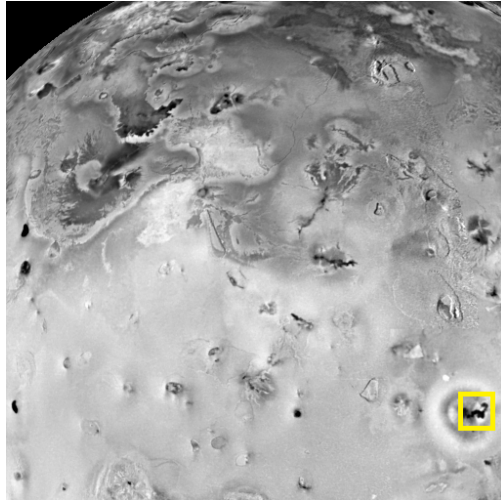


Figure 1: A photograph of the surface of the moon.



Figure 2: A clear 31×31 pixel crop of the moon photograph.

We create three different versions of Gaussian blur PSFs and their corresponding blurring matrices. The figures below show three sizes of symmetric Gaussian PSFs, a plot of the singular values of their corresponding A matrices, and blurred versions of the clear pictures corresponding to each PSF. Note that this PSF is separable, thus we use the techniques relevant to separable cases.

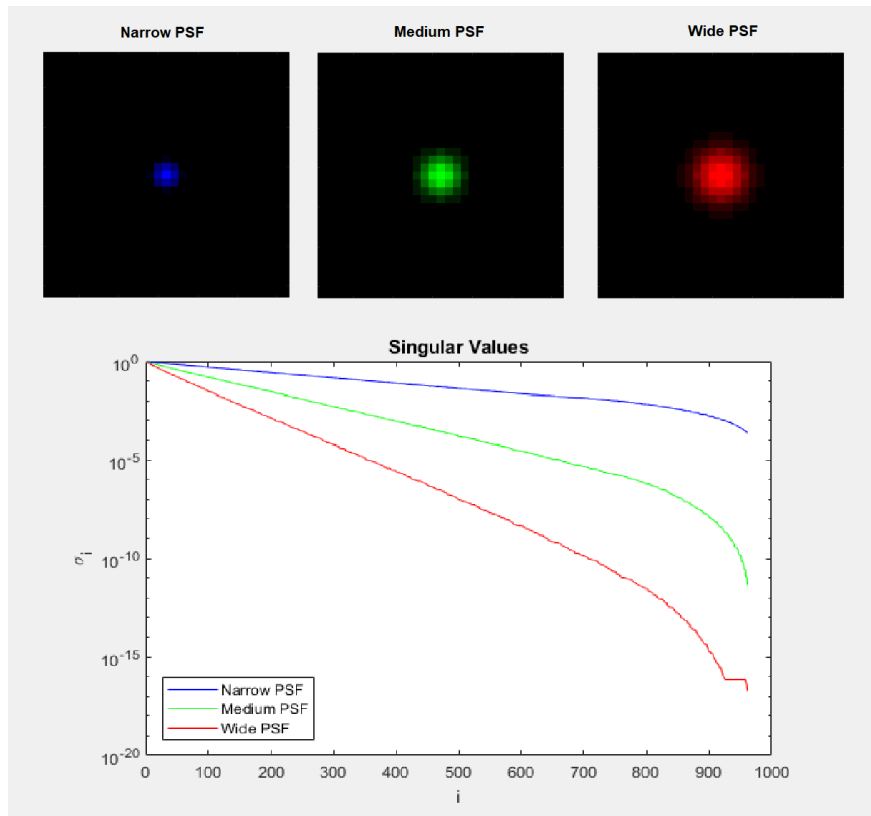


Figure 3: Gaussian PSFs with $\rho = 0$, $s_1 = s_2 = 1, 1.7, 2.4$ respectively, and the singular values σ_i of the corresponding blurring matrices A , assuming zero boundary conditions.

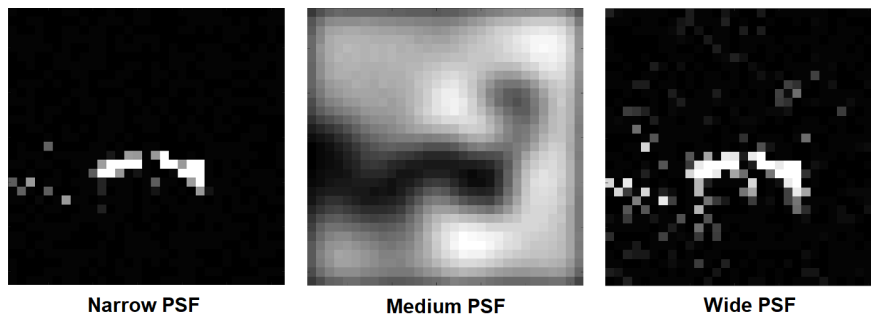


Figure 4: Blurred versions B of the clear lunar crop using the three Gaussian PSFs above.

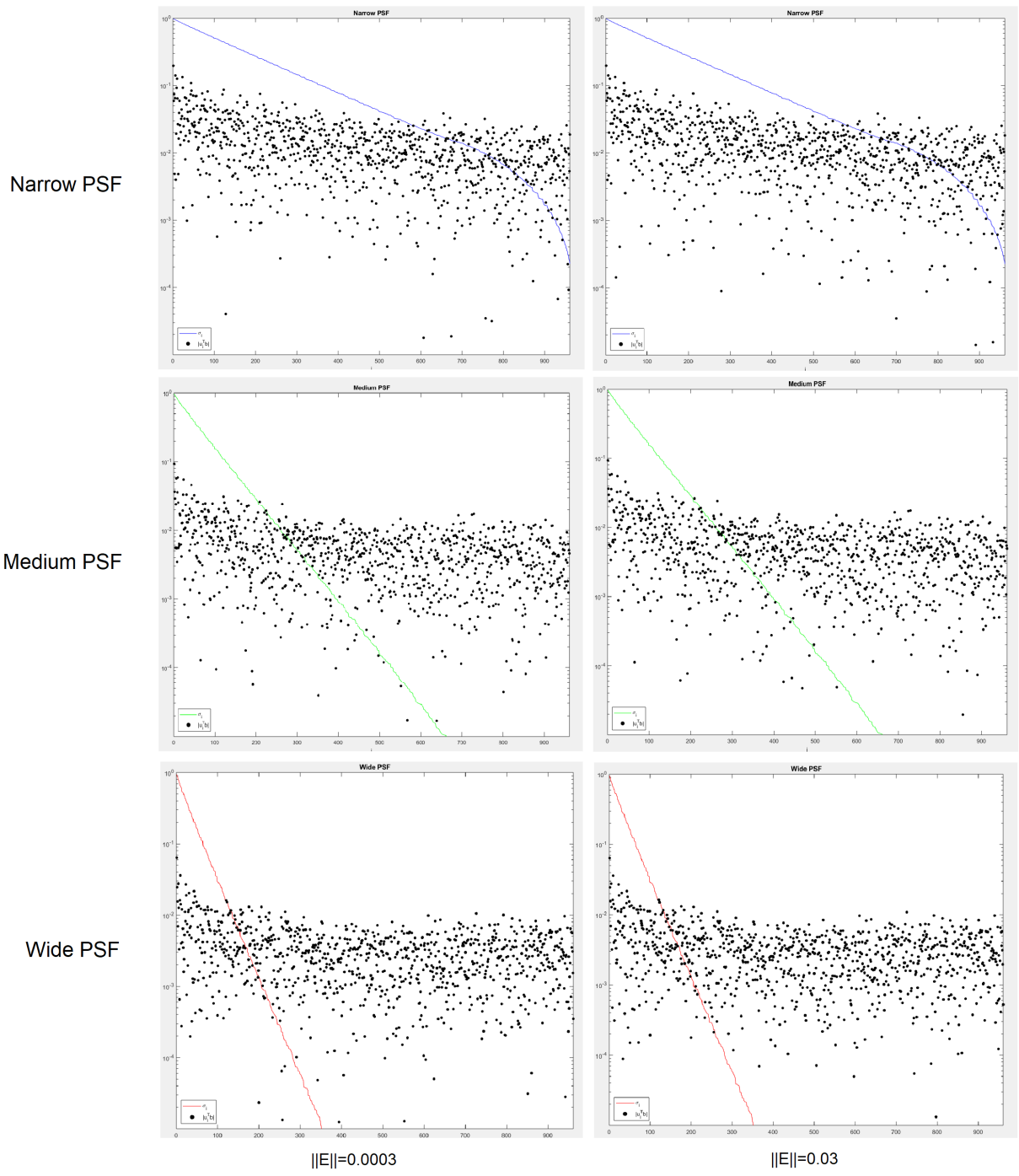


Figure 5: Singular values σ_i (colored lines) and coefficients $|\vec{u}_i^T \vec{b}|$ for the three PSFs from above, with different levels of noise given by $\|E\|_2 = 0.0003$, $\|E\|_2 = 0.03$.

We see that as the blurring gets more intense, the singular values of A decay faster. Figure 5 compares singular values and coefficients $|\vec{u}_i^T \vec{b}|$ for the same three blurring matrices, with varying levels of noise. We know the coefficients are affected by noise present in the system, and observe that the coefficients level off at a noise plateau determined by the level of noise in the images. Thus, only the coefficients smaller in magnitude than their corresponding singular value carry clear information about the data. This gives us insight into the range of k value that is appropriate for a TSVD filter.

Figure 6 displays the singular values σ_i and coefficients $|\vec{u}_i^T \vec{b}|$ before and after TSVD for the medium PSF from Figure 3, with the higher level of noise from Figure 5. We see that the trend in the magnitude of TSVD coefficients crosses paths with the trend in singular values around $k = 200$. This indicates that $k = 200$ is a reasonable choice of truncation parameter, since $k = 300$ has too much noise present in the system but $k = 100$ has an overloss of information. Figure 7 also suggests a choice of $k = 200$ since this is where the magnitudes of perturbation and regularization error cross.

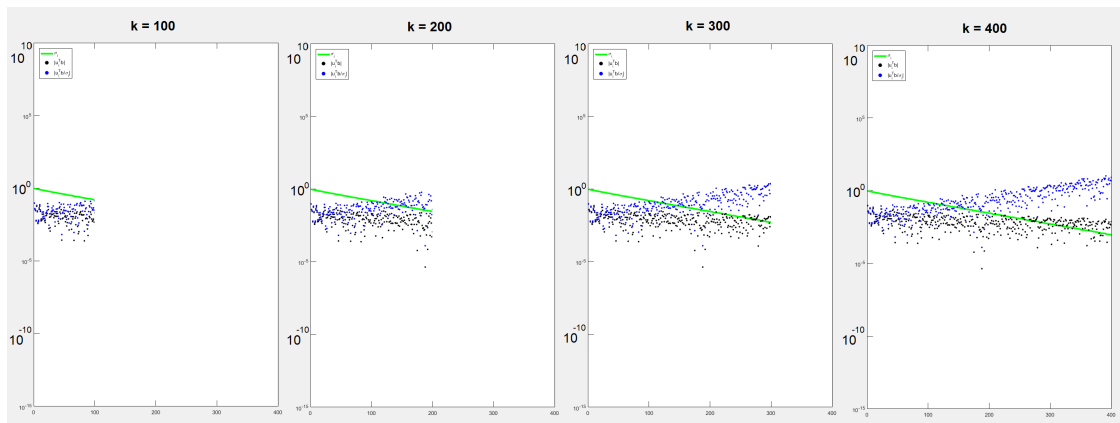


Figure 6: Singular values σ_i (green curve), normal coefficients $|\vec{u}_i^T \vec{b}|$ (black dots), and TSVD coefficients $\frac{|\vec{u}_i^T \vec{b}|}{\sigma_i}$ (blue dots) for the medium PSF from Figure 3 with the higher noise level in Figure 5.

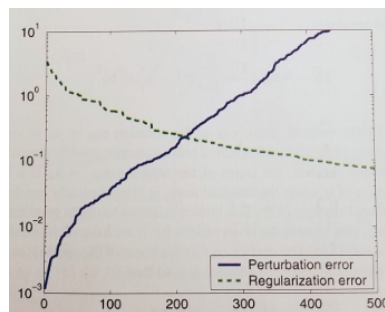


Figure 7: The 2-norms of the regularization error and perturbation error vs. the truncation parameter k for the TSVD method. (Hansen, 2006)

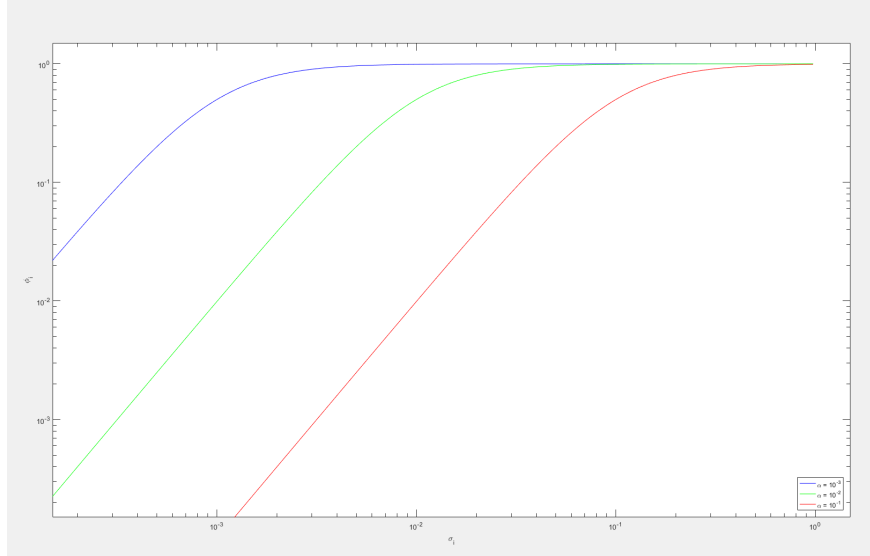


Figure 8: Tikhonov filter factors ϕ_i vs. i for $\alpha = 10^{-3}$ (blue), $\alpha = 10^{-2}$ (green), and $\alpha = 10^{-1}$ (red).

Figure 8 shows how Tikhonov filter factors change with regard to parameter α . The filter factors level off more quickly for low values of α , illustrating that the point at which filter factors change behavior is at around $\sigma_i = \alpha$. The bend in the L-curve in Figure 9 also corresponds to this α . We see that this α satisfies $\sigma_{mn} \ll \alpha \ll \sigma_1$.

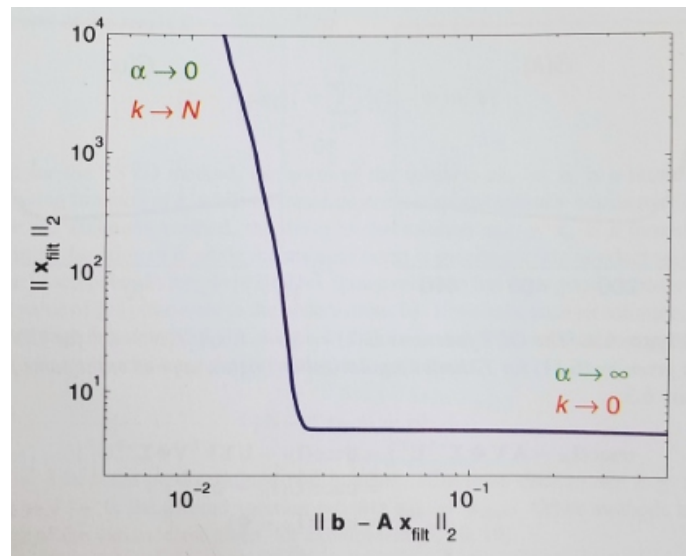


Figure 9: The L-curve for TSVD with parameter k applied to the same model. The L-curve for Tikhonov regularization with parameter α would look similar. (Hansen, 2006)

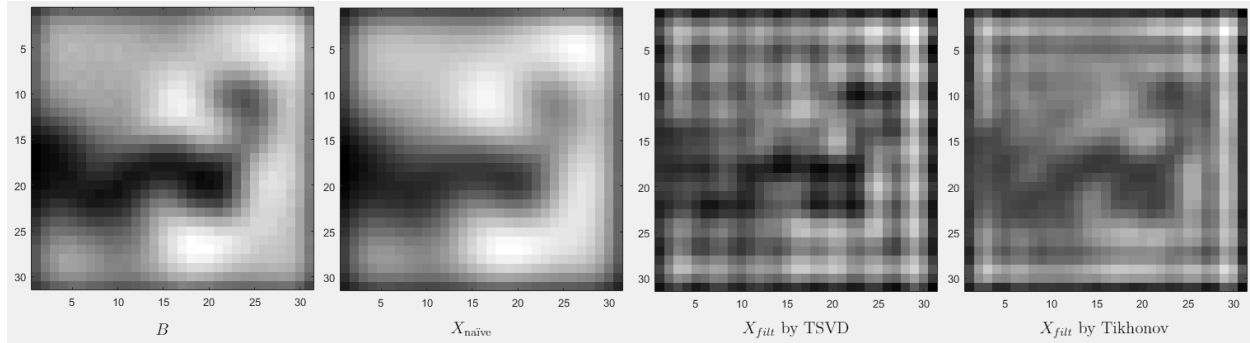


Figure 10: A Gaussian blur of the clear image with added noise, and the results of deblurring using the naïve, truncated singular value decomposition, and Tikhonov regularization methods.

Figure 10 shows the results of deblurring the blurred image given by the medium PSF from Figure 4 with noise added, using the naïve, TSVD, and Tikhonov regularization methods. We see that the naïve approach serves only to worsen the blur of the image. The TSVD and Tikhonov results are imperfect, but have stronger definition of the main features of the original image.

Appendix

Singular Value Decomposition

Let A be any $m \times n$ matrix.

Then A has a factorization $A = U\Sigma V^T$, for orthogonal U, V and diagonal Σ .

Proof:

Since $A^T A$ is symmetric, we know that it has only real, nonnegative eigenvalues. Let $\lambda_1, \lambda_2, \dots, \lambda_n$ be the complete list of eigenvalues of $A^T A$, arranged so that $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0$. Let $\sigma_i = \sqrt{\lambda_i}$. Then $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$. The set $\{\sigma_1, \sigma_2, \dots, \sigma_n\}$ is called the set of singular values of A .

Now, define Σ to be the $m \times n$ matrix with the singular values of A along its diagonal and zeros elsewhere. For example, if $m > n$, then

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_n \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}.$$

Let $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_n$ be the normalized eigenvectors of $A^T A$ associated with $\lambda_1, \lambda_2, \dots, \lambda_n$ respectively. Since $A^T A$ is symmetric, these eigenvectors are orthogonal.

Define $V = [\vec{v}_1 \ \vec{v}_2 \ \dots \ \vec{v}_n]$. Then V is $n \times n$, and its columns are orthonormal. Thus,

$$V^T = \begin{bmatrix} \vec{v}_1^T \\ \vec{v}_2^T \\ \vdots \\ \vec{v}_n^T \end{bmatrix} \text{ is an } n \times n \text{ orthogonal matrix.}$$

Let $r = \text{rank}(A)$, so that $\sigma_1 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_n = 0$.

Now define U to be an $m \times m$ matrix such that

$$U = \begin{bmatrix} A\vec{v}_1 & A\vec{v}_2 & \dots & A\vec{v}_r & \vec{n}_1 & \dots & \vec{n}_k \end{bmatrix}$$

where $\vec{n}_1, \dots, \vec{n}_k$ are chosen to be pairwise orthonormal with all other columns of U .

With this construction, it can be computed that $A = U\Sigma V^T$.

There is a reduced version of the singular value decomposition that also holds for any matrix: Let $r = \text{rank}(A)$. Let Σ_r be the $r \times r$ diagonal matrix with diagonal entries $\sigma_1, \dots, \sigma_r$. Let U_r be the matrix with the first r columns of U , and similarly let V_r be the matrix with the first r columns of V . Then $A = U_r \Sigma_r V_r^T$.

The singular value decomposition of a given matrix is not necessarily unique.

Both U and V are orthogonal matrices, and Σ is a diagonal matrix that is not necessarily square. A and Σ have the same dimensions: A is $m \times n$ if and only if Σ is $m \times n$.

If A is an $n \times n$ invertible matrix, then $\sigma_n \neq 0$ and the condition number of A is defined as $\frac{\sigma_1}{\sigma_n}$. The condition number is a measure of the stability and error potential in numerical calculations.

The Moore-Penrose pseudoinverse of a matrix A , denoted A^+ , can be constructed from the reduced singular value decomposition of A . The pseudoinverse is defined by $A^+ = V_r \Sigma_r^{-1} U_r^T$. Since A can be any matrix with real or complex entries, it may not be square or invertible, so the Moore-Penrose pseudoinverse is extremely useful. When A does happen to be invertible, $A^{-1} = A^+$.

If A is $n \times n$ square, it is often convenient to write $A = \sum_{i=1}^n \sigma_i \vec{u}_i \vec{v}_i^T$, and thus $A^+ = \sum_{i=1}^n \frac{1}{\sigma_i} \vec{v}_i \vec{u}_i^T$.

The Moore-Penrose pseudoinverse can be used to solve least-squares problems, as follows. Given $A\vec{x} = \vec{b}$, we find a least-squares solution \hat{x} by using the pseudoinverse:

$$\hat{x} = A^+ \vec{b} = (V_r \Sigma_r^{-1} U_r^T) \vec{b}$$

Then,

$$\begin{aligned} A\hat{x} &= (U_r \Sigma_r V_r^T)(V_r \Sigma_r^{-1} U_r^T) \vec{b} \\ &= U_r \Sigma_r \Sigma_r^{-1} U_r^T \vec{b} \\ &= U_r U_r^T \vec{b}. \end{aligned}$$

Since the columns of U_r are orthonormal, this gives the orthogonal projection \hat{b} of \vec{b} onto the column space of A . This means that \hat{x} is a least-squares solution of the equation $A\vec{x} = \vec{b}$. Moreover, this particular solution has the minimum norm of any least-squares solution.

Matrix Norms

The norm of a matrix is a measure of the size of its entries. There are many matrix norms, but they all satisfy certain properties. Each definition of a matrix norm must be a function that outputs a nonnegative scalar quantity. The function must have the trivial kernel, preserve scalar multiplication, and retain the triangle inequality.

Some matrix norms are induced by vector norms. For a vector norm $\|\cdot\|_k$, the corresponding matrix norm $\|\cdot\|_K$ is written as

$$\|A\|_K = \sup_{\vec{x} \in \mathbb{R}^n, \vec{x} \neq \vec{0}} \frac{\|A\vec{x}\|_k}{\|\vec{x}\|_k}$$

The most common induced matrix norm is the Euclidean norm, often referred to as the 2-norm.

Denoted $\|A\|_2$ for a matrix A , this norm is the square root of the largest eigenvalue of $A^T A$. This is the matrix norm that is induced by the Euclidean vector norm.

A commonly-used matrix norm is the Frobenius norm.

Denoted $\|A\|_F$ for a matrix A , this is the square root of the sum of the squares of the entries of A .

That is, $\|A\|_F = \sqrt{\sum_i \sum_j a_{ij}^2}$.

Although its definition is somewhat analogous to the Euclidean vector norm, the Frobenius norm is not induced by the Euclidean or any other vector norm.

When A is real-valued, $\|A\|_F^2 = \text{trace}(A^T A)$.

It is always the case that $\|A\|_2 \leq \|A\|_F$.

Importantly, $\|A\vec{x}\|_2 \leq \|A\|_2 \|\vec{x}\|_2$.

If O is an orthogonal matrix, then $\|O\|_2 = 1$ and $\|AO\|_2 = \|OA\|_2 = \|A\|_2$.

To prove the Eckart-Young-Mirsky Theorem below, we need the following two lemmas.

Lemma:

If $A = \begin{bmatrix} B & C \\ D & E \end{bmatrix}$, then $\|B\|_2 \leq \|A\|_2$.

Proof:

For any \vec{x} , $\|B\vec{x}\|_2 \leq \left\| \begin{bmatrix} B \\ D \end{bmatrix} \vec{x} \right\|_2 = \left\| \begin{bmatrix} B & C \\ D & E \end{bmatrix} \begin{bmatrix} \vec{x} \\ 0 \end{bmatrix} \right\|_2 \leq \|A\|_2 \|\vec{x}\|_2$.

Lemma:

If A is a square, nonsingular matrix, then the minimum $\|E\|_2$ such that $A + E$ is singular is given by $\|A^{-1}\|_2^{-1}$. This is called the absolute distance to singularity.

Proof:

Let A be an $n \times n$ nonsingular matrix.

Let E be an $n \times n$ matrix such that $A + E$ is singular.

Since $A + E$ is singular, then there exists a nontrivial solution \vec{x} to $(A + E)\vec{x} = \vec{0}$.

$$\begin{aligned}(A + E)\vec{x} &= \vec{0} \\ A\vec{x} + E\vec{x} &= \vec{0} \\ A\vec{x} &= -E\vec{x} \\ \vec{x} &= -A^{-1}E\vec{x} \\ \|\vec{x}\|_2 &= \|A^{-1}E\vec{x}\|_2 \\ \|\vec{x}\|_2 &\leq \|A^{-1}E\|_2 \|\vec{x}\|_2 \\ 1 &\leq \|A^{-1}E\|_2 \\ 1 &\leq \|A^{-1}\|_2 \|E\|_2 \\ \|A^{-1}\|_2^{-1} &\leq \|E\|_2.\end{aligned}$$

The Eckart-Young-Mirsky Theorem

Let A be an $m \times n$ matrix with rank r and singular value decomposition $A = U\Sigma V^T$.

Then the approximation of A of rank $k < r$ that has the least error is given by $A_k = U\Sigma_k V^T = \sum_{i=1}^k \sigma_i \vec{u}_i \vec{v}_i^T$.

That is, for all matrices B with $\text{rank}(B) = k$, $\min \|A - B\|$ occurs when $B = A_k$.

This fact holds for both the Frobenius and Euclidean norms. Here, we will go through the proof using the Euclidean norm. This proof is based on that found in (Ipsen, 2009).

Proof:

$\|A - A_k\|_2 = \|U\Sigma V^T - U\Sigma_k V^T\|_2 = \|U(\Sigma - \Sigma_k)V^T\|_2 = \|\Sigma - \Sigma_k\|_2 = \sigma_{k+1}$.
So, we need to show that for any other rank- k matrix B , $\|A - B\|_2 \geq \sigma_{k+1}$.

Since $\text{rank}(A) = r$, $\text{rank}(A^T A) = r$.

So $A^T A$ has r nonzero eigenvalues, and thus A has r nonzero singular values.

Thus we write $\{\sigma_1, \sigma_2, \dots, \sigma_n\} = \{\sigma_1, \dots, \sigma_r, 0, \dots, 0\}$.

Since $k < r$, A has at least $k + 1$ nonzero singular values.

Write $\Sigma = \begin{bmatrix} \Sigma_{k+1} & 0 \\ 0 & \Sigma_n \end{bmatrix}$, where $\Sigma_{k+1} = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_{k+1} \end{bmatrix}$.

Note that since $\sigma_1, \dots, \sigma_{k+1} \neq 0$, we have that Σ_{k+1} is nonsingular.

Let B be an $m \times n$ matrix with rank k .

Consider the matrix $U^T B V$.

We partition $U^T B V = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$, where B_{11} has dimension $k + 1 \times k + 1$.

Since $\text{rank}(B) = k$, $\text{rank}(B_{11}) \leq k$.

Since B_{11} is $k + 1 \times k + 1$, but $\text{rank}(B_{11}) \leq k$, we have that B_{11} is singular.

Since Σ_{k+1} is a diagonal matrix, $\Sigma_{k+1}^{-1} = \begin{bmatrix} \frac{1}{\sigma_1} & & \\ & \ddots & \\ & & \frac{1}{\sigma_{k+1}} \end{bmatrix}$. Thus $\|\Sigma_{k+1}^{-1}\|_2 = \frac{1}{\sigma_{k+1}}$.

Recalling that B_{11} is singular and Σ_{k+1} is nonsingular, we consider the distance from Σ_{k+1} to singularity and write $B_{11} = \Sigma_{k+1} - (\Sigma_{k+1} - B_{11})$.

Then $\|\Sigma_{k+1} - B_{11}\|_2 \geq \frac{1}{\|\Sigma_{k+1}^{-1}\|_2} = \frac{1}{\frac{1}{\sigma_{k+1}}} = \sigma_{k+1}$.

Now, we have that

$$\begin{aligned} \|A - B\|_2 &= \|U\Sigma V^T - B\|_2 \\ &= \|(U^T)(U\Sigma V^T - B)(V)\|_2 \text{ since } U^T, V \text{ orthogonal} \\ &= \|\Sigma - U^T B V\|_2 \\ &= \left\| \begin{bmatrix} \Sigma_{k+1} & 0 \\ 0 & \Sigma_n \end{bmatrix} - \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \right\|_2 \\ &= \left\| \begin{bmatrix} \Sigma_{k+1} - B_{11} & -B_{12} \\ -B_{21} & \Sigma_n - B_{22} \end{bmatrix} \right\|_2 \\ &\geq \|\Sigma_{k+1} - B_{11}\|_2 \\ &\geq \sigma_{k+1}. \end{aligned}$$

Kronecker Products

The Kronecker product of two matrices is defined in the following way:

$$\text{If } A \text{ is } m \times n, \text{ then } A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix}.$$

The Kronecker product is not commutative.

In this report we use the following properties:

In general, we can vectorize the matrix X by concatenating its columns, in order, to create one long column vector. This creates $\vec{x} = \text{vec}(X)$. Then

$$(A \otimes B)\text{vec}(X) = \text{vec}(BXA^T)$$

$$(A \otimes B)^T = A^T \otimes B^T$$

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$$

$$AB \otimes CD = (A \otimes C)(B \otimes D).$$

In particular, for singular value decompositions $A = U_A \Sigma_A V_A^T$, $B = U_B \Sigma_B V_B^T$ we have $A \otimes B = (U_A \Sigma_A V_A^T) \otimes (U_B \Sigma_B V_B^T) = (U_A \otimes U_B)(\Sigma_A \otimes \Sigma_B)(V_A \otimes V_B)^T$.

Toeplitz, Circulant, and Hankel Matrices

A matrix whose entries are constant on each diagonal is called a Toeplitz matrix. For example:

$$\begin{bmatrix} c & d & e \\ b & c & d \\ a & b & c \end{bmatrix}$$

A Toeplitz matrix with each row and column a periodic shift of the previous is called a circulant matrix. For example:

$$\begin{bmatrix} a & c & b \\ b & a & c \\ c & b & a \end{bmatrix}$$

A matrix whose entries are constant on each antidiagonal is called a Hankel matrix. For example:

$$\begin{bmatrix} a & b & c \\ b & c & d \\ c & d & e \end{bmatrix}$$

Sources

Hansen, P. C., Nagy, J. G., O’Leary, D. P. (2006). *Deblurring Images: Matrices, Spectra, and Filtering*. New York, NY: Society for Industrial Applied Mathematics.

Hansen, P. C. (2010). *Discrete Inverse Problems: Insight and Algorithms*. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Ipsen, I. C. (2009). *Numerical Matrix Analysis: Linear Systems and Least Squares*. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Lay, D. C., Lay, S. R., McDonald, J. J. (2002). *Linear Algebra and its Applications (3rd ed.)*. Harlow: Pearson Education Limited.

Leon, S. J., Bica, I., Hohn, T. (2017). *Linear Algebra With Applications (5th ed.)*. New York, NY: Pearson Learning Solutions.