4-22-2008

Singular Value Decomposition in Image Noise Filtering and Reconstruction

Tsegaselassie Workalemahu

SINGULAR VALUE DECOMPOSITION IN IMAGE NOISE FILTERING AND RECONSTRUCTION

by

TSEGASELASSIE WORKALEMAHU

Under the Direction of Dr. Marina Arav

ABSTRACT

The Singular Value Decomposition (SVD) has many applications in image processing. The SVD can be used to restore a corrupted image by separating significant information from the noise in the image data set. This thesis outlines broad applications that address current problems in digital image processing. In conjunction with SVD filtering, image compression using the SVD is discussed, including the process of reconstructing or estimating a rank reduced matrix representing the compressed image. Numerical plots and error measurement calculations are used to compare results of the two SVD image restoration techniques, as well as SVD image compression. The filtering methods assume that the images have been degraded by the application of a blurring function and the addition of noise. Finally, we present numerical experiments for the SVD restoration and compression to evaluate our computation.

INDEX WORDS: Singular Value Decomposition, Rank, Eigenvectors, Eigenvalues, Singular Values, Filtering, Least Squares, Condition Number, Convolution, Discrete Fourier Transform, Frequency, Pseudo-Inverse, Point Spread Function

SINGULAR VALUE DECOMPOSITION IN IMAGE NOISE FILTERING AND RECONSTRUCTION

by

TSEGASELASSIE WORKALEMAHU

A Thesis Presented in Partial Fulfillment of the Requirements for the Degree of

Master of Science

in College of Arts and Sciences

Georgia State University

2008

SINGULAR VALUE DECOMPOSITION IN IMAGE NOISE FILTERING AND RECONSTRUCTION

by

TSEGASELASSIE WORKALEMAHU

Major Professor: Marina Arav

Committee: Saeid Belkasim
Frank Hall
Zhongshan Li
Michael Stewart

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

April 2008

ACKNOWLEDGMENTS

The author wishes to gratefully acknowledge the assistance of Drs. Marina Arav, Saeid Belkasim, Frank Hall, Zhongshan Li, and Michael Stewart, without whose guidance this thesis would not have been possible. He also would like to thank Drs. Margo Alexander, George Davis, Lifeng Ding, Donald Edwards, Kyle Frantz, Alexandra Smirnova, and Ying Zhu for support and encouragement in his course work and in his research towards this thesis.

TABLE OF CONTENTS

1. INTRODUCTION

Linear Algebra and Matrix Theory are fundamental mathematical disciplines that have numerous applications in various fields such as computer science, biology, physics, chemistry, economics and psychology. Image processing is used to address problems in medicine, physics, biology, astronomy, geology, etc. Matrix analysis methods are a primary tool in image processing with applications to wavelets, computer vision, transformations in computer graphics, linear and nonlinear optimization, etc. Such applications have given rise to the development of a more recent computing technology [4], [19], [15]. This thesis outlines broad applications of the Singular Value Decomposition (SVD) that address current problems in digital images.

Hansen et al. (2006) discussed the approximation of a clear image from its linear representation, by which we can solve an inverse problem to reconstruct the true image. We will compare different techniques for filtering and denoising images discussed by Hansen et al. and a similar approach to the inverse SVD filtering. The outlined techniques include block denoising, inverse filtering, and deconvolution of noisy and distorted images using the SVD. A widely known SVD application to image compression is discussed to show the similarity of its approach to noise reduction.

Most images are obtained by optical, electronic, or electro-optic means and then digitized for processing, displaying, or archiving. Due to errors or noise in the data collection process, image data may contain artifacts [16]. For example, medical images obtained from MRI and PET scans often contain image signals corrupted by radiation and many other factors. The display process can also introduce artifacts that obscure the data contained in the image. It is often difficult to distinguish between "foreign objects" that were not part of the image and the "true" signal

itself. Before applying image manipulation techniques, such as segmentation or 3D reconstruction, it is often necessary to remove noise for productive reconstruction results [2].

Denoising is also a reconstruction process in images that is intended to remove values that represent the noise or "foreign objects". Noise filtering techniques, such as mean and median filtering, have been used to restore corrupted image data. In this thesis, we will investigate and discuss different types of models for the representation of noisy and blurred measurements. After applying particular noise and blur to images, we can compute the SVD of the image and apply an appropriate filtering method to obtain a restored signal.

2. MATHEMATICAL BACKGROUND ON THE SVD

2.1. Definitions

We provide the following definitions of concepts that arise frequently throughout this thesis.

Definition 2.1.1. Let matrix $A \in \mathbb{R}^{n\times n}$ and $x \in \mathbb{R}^n$. We consider the equation

$$Ax = \lambda x, x \neq 0,$$

where $\lambda \in \mathbb{R}$. If a scalar λ and a nonzero vector x satisfy this equation, then λ is called an *eigenvalue* of A, and x is called an *eigenvector* of A associated with λ.

Definition 2.1.2. Let matrix $A \in \mathbb{R}^{m\times n}$, then the *rank* of A is the largest number of columns of A that constitute a linearly independent set.

Since the rank of A^T equals to the rank of A, the rank may be equivalently defined in terms of linearly independent rows.

Definition 2.1.3. An *orthonormal* set of vectors is a set of mutually orthogonal unit vectors.

Such a set cannot contain a zero-vector and is necessarily linearly independent.

Definition 2.1.4. Let matrix $U \in \mathbb{R}^{n\times n}$. Then U is *orthogonal* if its inverse equals to its transpose, $U^{-1} = U^T$.

The set of columns of U is orthonormal.

Definition 2.1.5. An *orthonormal basis* for a vector subspace is a basis whose vectors constitute an orthonormal set.

Since any basis may be transformed to an orthonormal basis, any finite-dimensional complex vector space has an orthonormal basis.

2.2. The SVD, Properties, and Observations

E. Beltrami in 1873, (see [15]), studied the relationship between the SVD factorization of an input matrix A to the eigenvalue decomposition of the matrices

A^TA and AA^T. He discovered that for each matrix $A \in \mathbb{R}^{n\times n}$ there are always orthogonal matrices $Q_1, Q_2 \in \mathbb{R}^{n\times n}$, such that

$$Q_1^T A Q_2 = \Sigma = \text{diag}(\sigma_1(A), \ldots, \sigma_n(A)) \tag{1}$$

is a nonnegative diagonal matrix, where $\sigma_1(A)^2 \geq \cdots \geq \sigma_n(A)^2$ are the eigenvalues of AA^T (and also of A^TA). Moreover, he found that the (orthogonal) columns of Q_1 and Q_2 are eigenvectors of AA^T and A^TA, respectively.

Eckart and Young (1939), (see [20]), gave a clear and complete statement of the singular value decomposition for a rectangular complex matrix. They view the factorization $A = V\Sigma V^*$ as a generalization of the "principle axis transformation" for Hermitian matrices. While algebraists were developing the singular value and polar decompositions for finite matrices, there was a parallel and apparently quite independent development of related ideas by researchers in the theory of integral equations. A pair of integral equations were introduced in the following form

$$\varphi(s) = \lambda \int_a^b K(s,t)\psi(t)dt \quad \text{and} \quad \psi(s) = \lambda \int_a^b K(t,s)\varphi(t)dt, \tag{2}$$

where the functions $\varphi(s)$ and $\psi(s)$ are not identically zero. It was shown that the scalar λ must be real since λ^2 is an eigenvalue of the symmetric (and positive semidefinite) kernel

$$H(s,t) = \int_a^b K(s,\tau)K(t,\tau)d\tau.$$

If one thinks of $K(s,t)$ as an analog of a matrix A, then $H(s,t)$ is an analog of AA^T. Traditionally, the "eigenvalue" parameter λ in the integral equation literature is the reciprocal of what matrix theorists call an eigenvalue. Recognizing that such scalars λ together with their associated pairs of functions $\varphi(s)$ and $\psi(s)$ are, for many purposes, the natural generalization to the nonsymmetric case of the eigenvalues and eigenfunctions that play a key role in the theory of integral equations with symmetric kernels called λ an "eigenvalue" and the associated pair

of functions $\varphi(s)$ and $\psi(s)$ "adjoint eigenfunctions" associated with λ [20]. These "eigenvalues" and "adjoint eigenfunctions" gave the SVD certain properties that will be discussed in the next section.

Let matrix $A \in \mathbb{R}^{m\times n}$. The *Singular Value Decomposition* is given by

$$A = U\Sigma V^T, \tag{3}$$

where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\Sigma \in \mathbb{R}^{m\times m}$ is a matrix whose off-diagonal entries are all zeros and whose diagonal elements satisfy $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$:

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

The σ_i's are unique and called the *singular values* of A. The column vectors of U (also called the *left singular vectors* of A) are eigenvectors of the matrix AA^T. The columns of V (also called the *right singular vectors* of A) are eigenvectors of the matrix A^TA. Next, we state the SVD theorem and its proof provided by S. Leon (see [19]).

Theorem 2.2.1 (The SVD Theorem) *If matrix $A \in \mathbb{R}^{m\times n}$, then A has a singular value decomposition.*

Proof. Let $A^TA \in \mathbb{R}^{n\times n}$ be a symmetric matrix. Then its eigenvalues are all real and it has an orthogonal diagonalizing matrix V. Furthermore, its eigenvalues must all be nonnegative. To see this, let λ be an eigenvalue of A^TA and x be an eigenvector associated with λ. It follows that

$$||Ax||^2 = x^TA^TAx = \lambda x^Tx = \lambda||x||^2.$$

Hence,

$$\lambda = \frac{||Ax||^2}{||x||^2} \geq 0.$$

We may assume that the columns of V have been ordered so that the corresponding eigenvalues satisfy

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0.$$

The singular values of A are given by

$$\sigma_j = \sqrt{\lambda_j} \qquad j = 1, \ldots, n.$$

Let r denote the rank of A. The matrix $A^T A$ will also have rank r. Since $A^T A$ is symmetric, its rank equals the number of nonzero eigenvalues. Thus

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r \geq 0 \quad \text{and} \quad \lambda_{r+1} = \lambda_{r+2} = \ldots = \lambda_n = 0.$$

The same relation holds for the singular values

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq 0 \quad \text{and} \quad \sigma_{r+1} = \sigma_{r+2} = \ldots = \sigma_n = 0.$$

The SVD will give us the rank of matrix A by simply getting the number of nonzero singular values of A or the nonzero diagonal elements of Σ.

Now let

$$V_1 = (v_1, \ldots, v_r), \quad V_2 = (v_{r+1}, \ldots, v_n)$$

and

$$\Sigma_1 = \begin{bmatrix} \sigma_1 & 0 & \ldots & 0 \\ 0 & \sigma_2 & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ 0 & & \ldots & \sigma_r \end{bmatrix}. \tag{4}$$

Thus $\Sigma_1 \in \mathbb{R}^{r \times r}$ is a diagonal matrix whose diagonal entries are the nonzero singular values $\sigma_1, \ldots, \sigma_r$. The matrix $\Sigma \in \mathbb{R}^{m \times n}$ is then given by

$$\Sigma = \begin{bmatrix} \Sigma_1 & O \\ O & O \end{bmatrix}.$$

The column vectors of V_2 are eigenvectors of $A^T A$ associated with $\lambda = 0$. Thus

$$A^T A v_j = 0, \qquad j = r+1, \ldots, n$$

and, consequently, the column vectors of V_2 form an orthonormal basis for $N(A^TA) = N(A)$. Therefore,

$$AV_2 = 0$$

and, since V is an orthogonal matrix, it follows that

$$I = VV^T = V_1V_1^T + V_2V_2^T$$

$$A = AI = AV_1V_1^T + AV_2V_2^T = AV_1V_1^T. \tag{5}$$

So far we have shown how to construct the matrices V and Σ of the singular value decomposition. To complete the proof, we must show how to construct an orthogonal matrix $U \in \mathbb{R}^{m\times m}$ such that

$$A = U\Sigma V^T,$$

or, equivalently,

$$AV = U\Sigma. \tag{6}$$

Comparing the first r columns of each side of (6), we see that

$$Av_j = \sigma_j u_j \qquad j = 1, \ldots, r.$$

Thus, if we define

$$u_j = \frac{1}{\sigma_j} Av_j \qquad j = 1, \ldots, r \tag{7}$$

and

$$U_1 = (u_1, \ldots, u_r),$$

then it follows that

$$AV_1 = U_1\Sigma_1. \tag{8}$$

The column vectors of U_1 form an orthonormal set since

$$u_i^T u_j = (\frac{1}{\sigma_i} v_i^T A^T)(\frac{1}{\sigma_j} A v_j) = \frac{1}{\sigma_i \sigma_j} v_i^T (A^T A v_j) = \frac{\sigma_j}{\sigma_i} v_i^T v_j = \delta_{ij},$$

where $1 \leq i \leq r$, $1 \leq j \leq r$.

It follows from (7) that each u_j, $1 \leq j \leq r$, is in the column space of A. The dimension of the column space is r, so $u_1, \ldots, u_r$ form an orthonormal basis for $R(A)$. The vector space $R(A)^\perp = N(A^T)$ has dimension $m - r$. Let $\{u_{r+1}, u_{r+2}, \ldots, u_m\}$ be an orthonormal basis for $N(A^T)$ and set

$$U_2 = (u_{r+1}, u_{r+2}, \ldots, u_m),$$

$$U = [U_1 \quad U_2].$$

Thus, $u_1, \ldots, u_m$ form an orthonormal basis for $\mathbb{R}^m$. Hence U is an orthogonal matrix. We still must show that $U\Sigma V^T$ actually equals A. This follows from (8) and (5) since

$$U\Sigma V^T = [U_1 \quad U_2] \begin{bmatrix} \Sigma_1 & O \\ O & O \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = U_1 \Sigma_1 V_1^T = A V_1 V_1^T = A. \qquad \blacksquare$$

Now that we have shown how matrix $A \in \mathbb{R}^{m \times n}$ can be decomposed into the product $U\Sigma V^T$, let's consider the following properties of the SVD.

1. The singular values $\sigma_1, \ldots, \sigma_n$ of A are unique; however, the matrices U and V are not unique.

2. Since $AA^T = (U\Sigma V^T)(U\Sigma V^T)^T = (U\Sigma V^T)(V\Sigma^T U^T) = U\Sigma\Sigma^T U^T$, it follows that U diagonalizes AA^T and that the u_j's are eigenvectors of AA^T. Similarly, since V diagonalizes $A^T A$, it follows that the v_j's are eigenvectors of $A^T A$.

3. Comparing the jth columns of each side of the equation

$$AV = U\Sigma,$$

we get

$$Av_j = \sigma_j u_j \quad j = 1, \ldots, n.$$

Similarly,

$$A^T U = V\Sigma^T,$$

and hence

$$A^T u_j = \sigma_j v_j \qquad \text{for} \qquad j = 1, \ldots, n$$

$$A^T u_j = 0 \qquad \text{for} \qquad j = n+1, \ldots, m.$$

4. If A has rank r, then

(i) $v_1, \ldots, v_r$ form an orthonormal basis for $R(A^T)$.

(ii) $v_{r+1}, \ldots, v_n$ form an orthonormal basis for $N(A)$.

(iii) $u_1, \ldots, u_r$ form an orthonormal basis for $R(A)$.

(iv) $u_{r+1}, \ldots, u_m$ form an orthonormal basis for $N(A^T)$.

5. The rank of the matrix A is equal to the number of its nonzero singular values (where singular values are counted according to multiplicity). A similar assumption about eigenvalues is not true. For example, the matrix

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

has rank 3 even though all of its eigenvalues are 0.

6. In the case that A has rank $r < n$, if we set

$$U_1 = (u_1, u_2, \ldots, u_r), \qquad V_1 = (v_1, v_2, \ldots, v_r)$$

and define Σ_1 as in (4), then we obtain the following factorization called *compact form of the singular value decomposition* of A:

$$A = U_1 \Sigma_1 V_1^T. \tag{9}$$

Obtaining the rank of a matrix is useful in many applications of Linear Algebra. One example can be computing the number of solutions of a system of linear

equations. In many applications, it is necessary to either determine the rank of a matrix or to determine whether the matrix is deficient in rank. Gaussian elimination is one approach to obtain the rank by reducing the matrix to the echelon form and then counting the number of nonzero rows. However, this approach will often produce errors during the elimination process. The SVD presents a method for determining how close the given matrix is to a matrix of smaller rank. The next chapter demonstrates the SVD compression technique and shows how to get rank reduced matrix for a desired reconstruction.

3. THE SVD IMAGE RECONSTRUCTION AND MATLAB

3.1. Image Compression using the SVD

The compact form (9) of the singular value decomposition of A is useful in many applications, particularly in image compression. Furthermore, the SVD image compression algorithm discards elements representing small singular values. The analysis of image compression is given by J. Demmel (see [5]). To show how the SVD compression works, let A be a matrix representing a given image (see section 3.2 for details on image representation), and $A = U\Sigma V^T$, then A can be written as

$$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + ... + \sigma_n u_n v_n^T,$$

we can then obtain the truncated sum after the first k terms

$$A_k = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + ... + \sigma_k u_k v_k^T.$$

We can choose k to be considerably less than n and still have the image corresponding to A_k very close to the original. The value k represents the rank of the matrix A_k. The magnitude of the smallest nonzero singular value provides a measure of how close A is to a matrix of lower rank. A_k is a compression of the data represented by matrix A, where the amount of storage necessary for the original matrix A was reduced. The SVD also has an interesting ability to adapt to the local statistical variations of noise levels [27]. Small variations on the matrix A may not affect the reconstruction and may well adapt to loss of information. If A_k has rank k by construction, then

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T. \tag{10}$$

This means that we can represent the image A as a linear combination of the basis images ($u_i v_i^T$). All the basis images are rank one and form an orthonormal basis for image representation. Here, (10) is the best rank-k approximation of A. In a sense, the following equation is minimized

$$||A - A_k||_2 = || \sum_{i=k+1}^{n} \sigma_i u_i v_i^T ||_2 = ||U \begin{bmatrix} 0 & \dots & 0 & \\ 0 & \sigma_{k+1} & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ 0 & & \dots & \sigma_n \end{bmatrix} V^T ||_2 = \sigma_{k+1}. \qquad (11)$$

The vector space spanned by $v_1, \dots, v_{k+1}$ has dimension $k+1$. In an $m \times n$ size image, an effective compression technique is needed to represent the image rather than transmitting or storing the entire $m \times n$ matrix. In (10), it will take $m \times k + n \times k = (m+n) \times k$ words to store u_1 through u_k, and $\sigma_1 v_1$ through $\sigma_k v_k$, from which we can reconstruct A_k. Thus, after compression A_k will be stored using $(m+n) \times k$ words. Relative errors and compression ratios are approximated by σ_{k+1}/σ_1 and $(m+n) \times k/(m \times n)$, respectively, see also [5].

The following simple procedure implements the SVD image compression algorithm using Matlab. An $m \times n$ matrix representing an input image is compressed by retaining $k < \min\{m, n\}$ singular values and the corresponding singular vectors.

```
Obtain an input matlab loaded figure
load clown
I = ind2gray(X,map);
[m,n] = size(Z);
Show the original image
imshow(I,64);
Convert to double precision
I = im2double(I);
Compute the SVD of the matrix and obtain the singular values
[U,S,V] = svd(I);
```

```
sigma = diag(S);
Initialize rank k=1 approximation
I1 = sigma(1)*U(:,1)*V(:,1)';
Reduce the singular values by a chosen rank k of the matrix
for i=1:k
    I1 = I1 + sigma(i)*U(:,i)*V(:,i)';
end
imshow(I1,64);
For displaying the double class matrix as an image use
imagesc(I1); colormap(gray)
To plot singular values use the logarithmic based scale plot
semilogy(s/s(1),'.-');
ylabel('singular values');
grid;
```

More results of applying the SVD compression algorithm to the $n \times n$ matrix representing "Lena" image are shown in Figure 2. Compression by applying the SVD to the blocks of the matrix is discussed in Chapter 4, and shown in Figures 10, 11, and 12. In Chapter 5, results obtained via SVD compression are compared to the original image data by using Mean Square Error (MSE) and Peak Signal-to-Noise Ratio (PSNR). The MSE measures the average of the square of the error between the approximated result and the original image. The PSNR is the ratio in log scale between the maximum possible signal verses the MSE. The MSE and PSNR plots are shown in Figure 3. Compression ratio (CR) is also used for estimating the ratio of image quality by computer load time, thus determining the efficiency of the SVD compression technique.

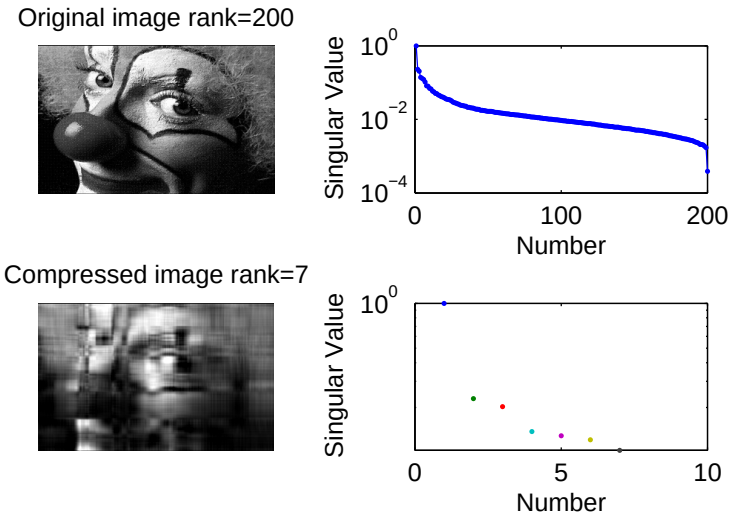

Figure 1: "clown" image obtained from Matlab mat files. Its reduced rank compression with corresponding singular value plots that show the number of ranks reduced.

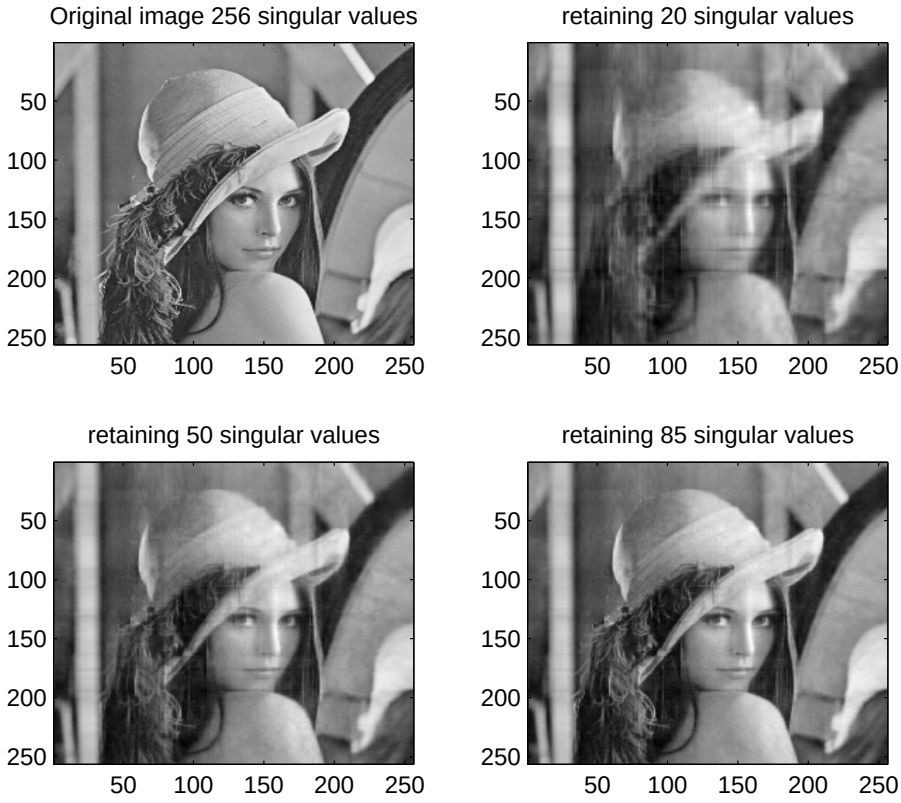

Figure 2: The Figure shows more examples of the SVD compression using "Lena" image sample obtained from [2]. The image becomes compressed as we retain small number of singular values and their corresponding singular vectors. However, critical information of the image is lost due to the removal of significant singular values. Here, the best compression is shown after retaining 85 out of 256 singular values.

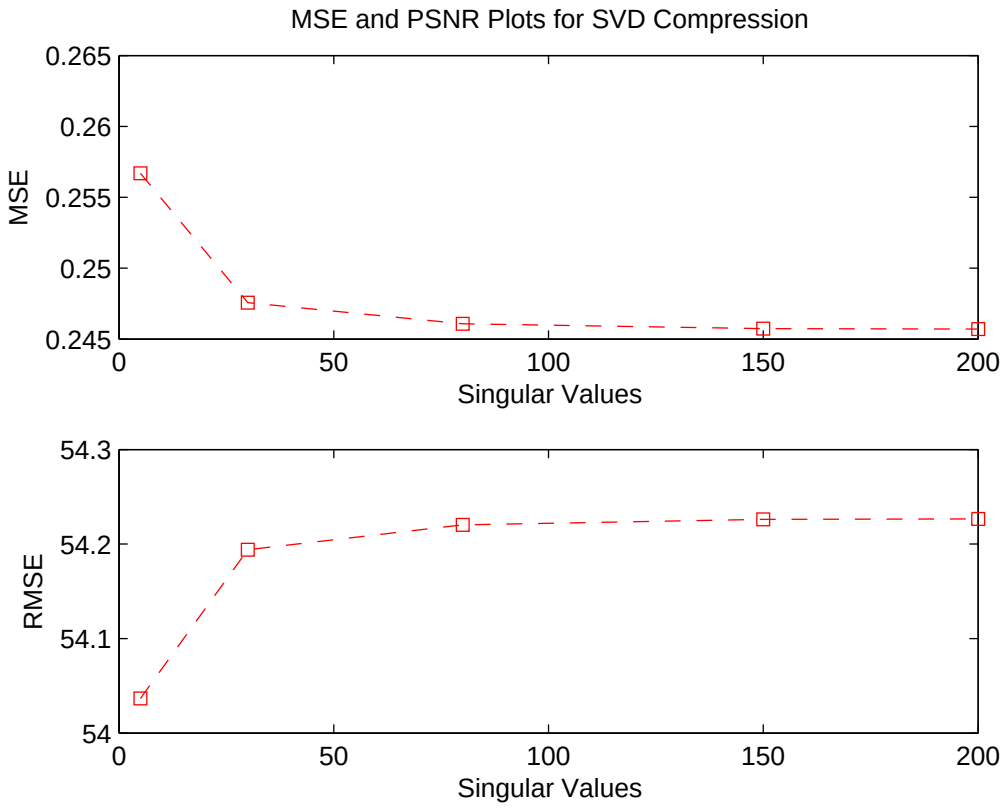

Figure 3: Mean square error and peak signal-to-noise ratio plots for results in Figure 2. The singular values are indicated by the "boxes" on the graph.

3.2. Image Representation

There are several artifacts that arise in an image data during the display or data collection process. For example, MRI (Magnetic Resonance Imaging) scanned images are obtained by an instrument that contains magnetic coils that can resonate with parts of the object (e.g. tissue) by a magnetic field. High resonance produces different density than lower resonance in scanned images. As a result, high-level of noise is generated, since density plays important role in determining how much intensity is present in the image data. PET (Positron Emission Tomography) scan is another example of medical image sample that often gets affected by noise interference. The PET imaging technique scans and outputs an image based on the

energy quantity it radiates to reflect the intensity of an object in the image. There is a likelihood that random radiation can corrupt the obtained image [16], [2]. In addition, MRI and PET scanned images often are very large in data size, requiring a large storage space. Before applying further image manipulation techniques such as segmentation, compression, or 3D reconstruction, it is often necessary to remove noise for productive reconstruction results.

A digital image in Matlab is represented by a matrix of values. A pixel in an image corresponds to an intensity value in the matrix. We have shown that the SVD compression works by discarding insignificant pixel values without affecting the quality of the image. In our Matlab implementation, we use monochrome images that can be represented as an image with array of two dimensions. Digital monochrome (black-and-white) images are referred to as *intensity images*, or *gray-scale images*. This type of image can be thought of as a discretized two-dimensional function, where each point represents the light intensity at a particular spatial coordinate (see section 3.3, Figure 4). These spatial coordinates (otherwise, interpreted as the brightness of pixel) are usually represented in a Cartesian system as a pair of nonnegative integer values typically denoted as (i, j) [24]. The entries of our matrix A are nonnegative numbers corresponding to the measures of the gray levels.

3.3. The Fourier Transform, DCT and Convolution

Spatial filters are often used to suppress corrupted pixels (noise) in images. When we work in the spatial domain of images, we are operating directly on the images' pixel values. However, in the frequency domain operation, mathematical tools such as, the Discrete Fourier Transform (DFT) are used to convert the 2D function that an image represents into an alternate formulation. This formulation consists of coefficients correlating to spatial frequencies. The analysis of the level of correlation helps retain only significant components while dumping other factors

[24], [11].

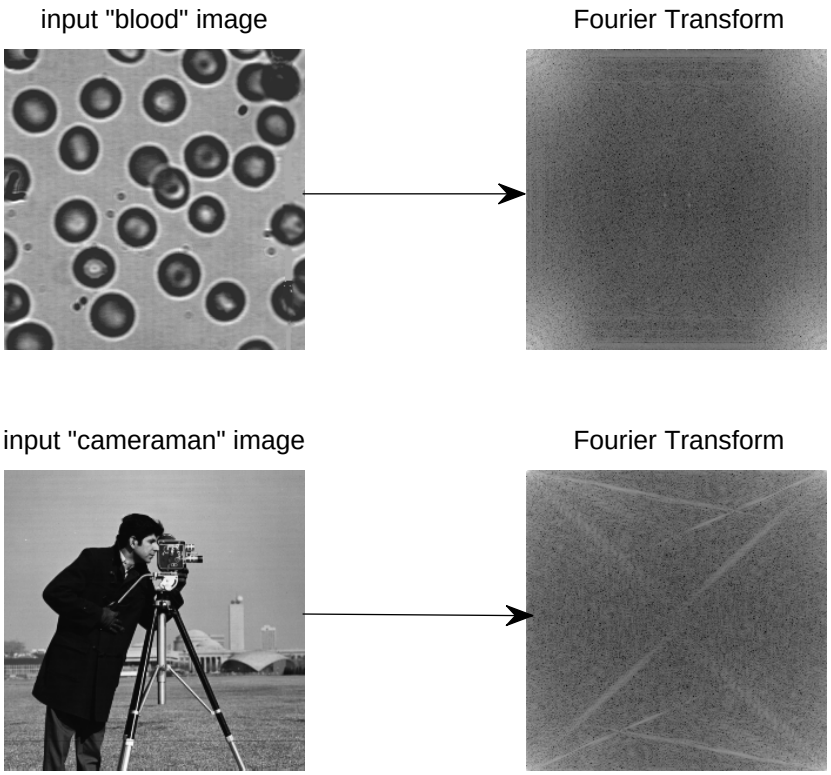

Figure 4: Displaying Fourier transform of "blood" and "cameraman" images.

In Figure 4, the low frequency components are located at the corners of the Fourier transform display for each image. Where as, the high frequency components are located in the middle. The bright stripes in the Fourier transform display of the "cameraman" image is as a result of the high spatial frequencies that are present in its background.

The Discrete Fourier Transform (DFT) of an n-sized vector v is the vector $y = \phi x$, where $\phi \in \mathbb{R}^{n \times n}$ is a matrix defined as $\phi_{jk} = \omega^{jk}$, and $\omega = \cos 2\pi/n - i \sin 2\pi/n$. The inverse form of vector y is given by $x = \phi^{-1} y$, and this is known as Inverse Discrete Fourier Transform (IDFT) [5]. The inverse transform is applied to

map the frequency coefficients back to gray-level pixel intensities. Using DFT, the transformation kernel is linear, separable and symmetric. Fast implementations are possible. Results for image filtering in the Fourier domain are shown in Chapter 5 of this thesis.

A similar linear transform that uses real numbers is the Discrete Cosine Transform (DCT). As defined in [1], we can mathematically define the DCT by an orthogonal matrix U of order m as follows

$$U(i,j) = \begin{cases} \sqrt{\frac{1}{m}}, & \text{if } i = 1, \\ \sqrt{\frac{2}{m}} \cos(\frac{\pi(2j-1)(i-1)}{2m}), & \text{if } 2 \leq i \leq m. \end{cases}$$

The DCT of an image matrix $A \in \mathbb{R}^{m \times n}$ is then defined by the matrix $X = UAV^T$, where U and V are orthogonal matrices. DCT helps decorrelate an image data, after which each linear transform coefficient can be encoded independently without losing efficiency. To get its Inverse Discrete Cosine Transform (IDCT), we define back the matrix, such that $A = U^T XV$. The DCT is linear, because the orthogonal matrices U and V are independent of A.

Convolution is an operation in Fourier analysis, Fourier series, or DFT. As shown in [5], consider the Fourier transform $F(f * g) = F(f) \times F(g)$. This means that Fourier transform of the convolution is the product of the Fourier transforms. The convolution theorem states that convolution in the time domain is equivalent to multiplication in the frequency domain, that is Fourier transform, or DFT reduce the convolution operation $F(t) * F(g)$ to multiplication $F(f) \times F(g)$. The Fourier transform using the above functions f and g is given by

$$(f * g)(x) \equiv \int_{-\infty}^{\infty} f(x-y)g(y)dy.$$

On the other hand, let $a(x) = \sum_{k=0}^{n-1} a_k x^k$ and $b(x) = \sum_{k=0}^{n-1} b_k x^k$ be degree $(n-1)$ polynomials. Then their product is given by $c(x) \equiv a(x).b(x) = \sum_{k=0}^{2n-1} c_k x^k$, where the coefficients $c_0, \ldots, c_{2n-1}$ are provided by the discrete convolution [5].

In signal and image processing, convolution is of fundamental importance. Two-dimensional convolution, which is used for image signals, provides the smoothing operation that uses a filter mask known as “kernel”. Image filtering techniques are applied by the smoothing operation, which simply uses a moving average kernel. Convolving or multiplying an image signal by a blur parameter will introduce a blur to an image (shown on the top right corner of Figure 5). In the next chapter, we will define what a kernel is and how image filtering operates.

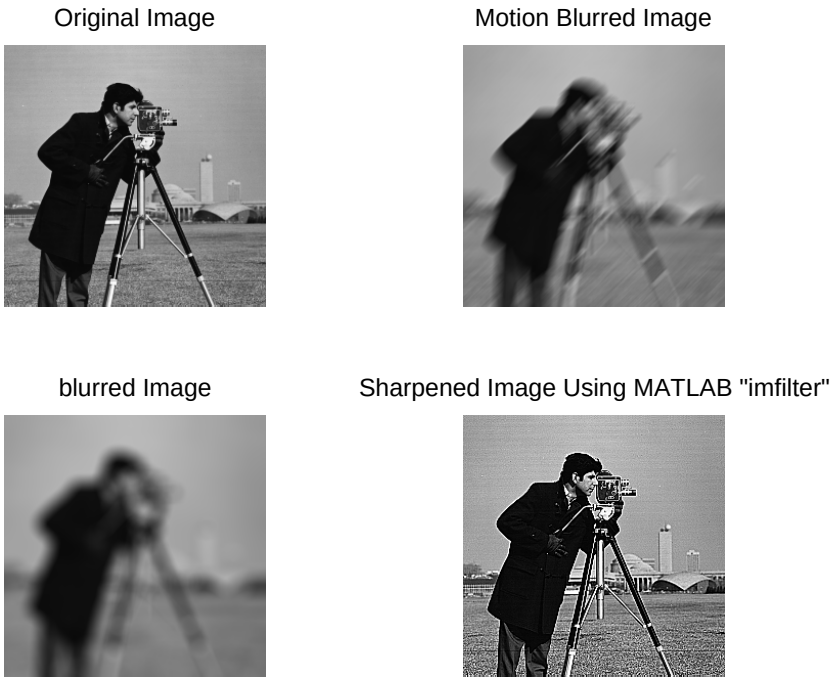

Figure 5: Blur filtering example using the Matlab imfilter function. The top right picture shows motion blurred image at 45 degree angle, and the bottom left picture shows Gaussian filter. Both images are deblurred and sharpened using Matlab operation “imfilter”.

4. IMAGE FILTERING AND DENOISING TECHNIQUES

The filtering operation commonly uses a "neighborhood mask", also known as kernel, and slides it across the input image at each point. The kernel contains multiplication factors and its filter works by applying a kernel matrix pixels and their neighbors. Once all values have been multiplied, the pixel is replaced with the sum of the products [9]. Pixels within the current neighborhood are combined by using some formula, then finally the output pixel is computed. For example, by using a 3×3 kernel, the kernel filtered rows are combined via summation across the orthogonal (column-wise) direction, thereby producing pixels in the output image that have been filtered in a 2D fashion. Different types of filtering operations can be applied by adjusting kernel size. In the following sections, we will discuss median and averaging filter techniques by using a $k \times k$ kernel size. First, we will discuss the types of noise and blur parameters that are used during the filtering process.

The following general steps were taken to experiment on filtering techniques in digital images:

1. Obtain a "clear" sample image.

2. Convolve the image with a PSF filter to introduce a blur (noise-free case) or apply zero-mean Gaussian noise.

3. Run the corrupted image through the filter.

4. Compare the processed image to the original image.

4.1 Noise Model and Blur Parameters

The addition of noise in an image causes an appearance of false, spotty elements. In our image sample, Gaussian white noise was used to introduce a spatially independent noise. Gaussian white noise is produced randomly by a probability

distribution function. The magnitude of this noise tracks the local variance of the image. We will use a zero-mean Gaussian white noise by adjusting its noise variance (σ). In general, Gaussian noise with variance σ and mean μ has probability density function

$$\eta(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2}.$$

Adding the noise represented by η to our image sample A will produce a corrupted image g, such that $g = A + \eta$.

In noise free cases, a blur can be modeled using a shift-invariant operation, which allows a transform that shifts the input signal independent of blur position. It can be expressed using the least squares model, since the blurring is assumed to be a linear operation. To blur an image, we can convolve it with a PSF (Point Spread Function) that describes the response of an imaging system to a point source or point object. The PSF allows every point in the original image to spread out the same way in forming the blurry image [26]. Once convolved with an image, the Matlab function imfilter('motion',n,θ) filters the image, given the linear motion of a camera by n pixels and an angle of θ degrees in a counter-clockwise direction. The blurring model using the convolution operation '$*$' is given by

$$g(x, y) = PSF(x, y) * X(x, y) + \eta(x, y), \tag{12}$$

where the PSF is to be convolved by the sample image $X(x, y)$, and $n(x, y)$ is a matrix representing the Gaussian white noise. It is also shown in [7] that singular vectors of a degraded image can be used to estimate an unknown PSF, while the smallest singular values are used to estimate the noise variance.

4.2 Mean and Median Filtering

Very simple noise removal techniques that are widely used for comparing with other filtering methods include Mean and Median filtering. In Median filtering

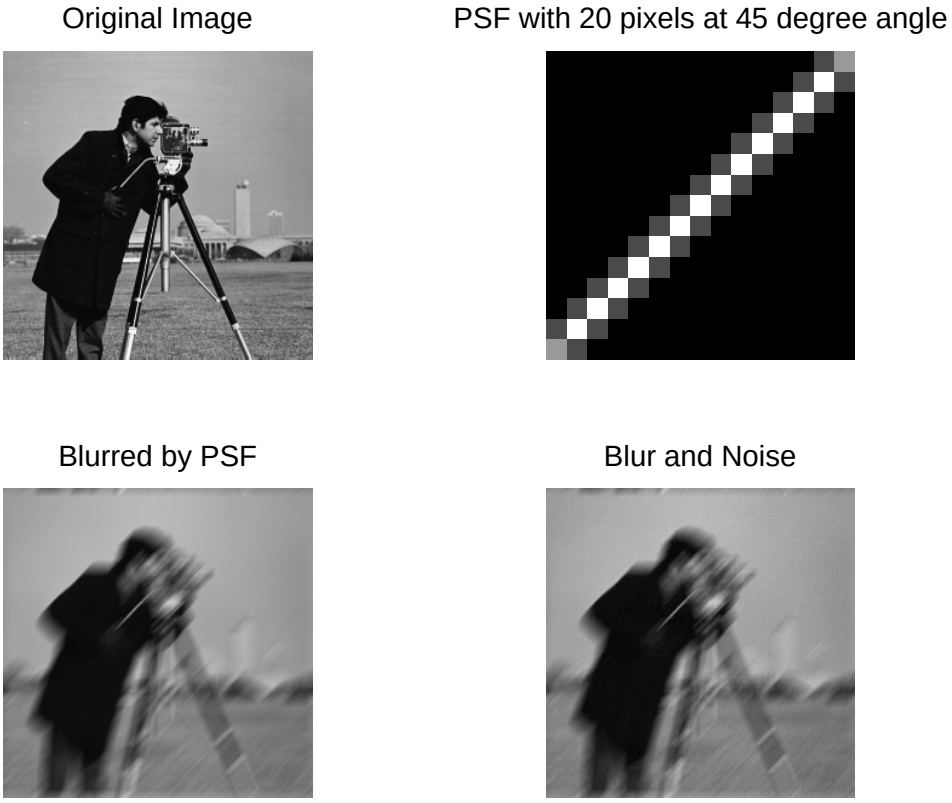

Figure 6: Convolving the "cameraman" image by a PSF and adding a Gaussian noise; In the spatial domain, the PSF (shown on the top right corner) describes the degree to which an optical system blurs (spreads) a point of light.

of noisy images, the value of the output pixels are determined by the Median of the neighborhood pixels within a defined mask. It is able to reduce pixel outliers without affecting the sharpness or the quality of the image. In Mean filtering, the output pixels are set to an average of the pixel values in the neighborhood mask of the corresponding input pixels. The Median is much less sensitive than the Mean to pixel outliers [2]. For this reason, Median filtering is considered a better approach than Mean filtering for reducing noise in images. Visual inspection may not lead to the conclusion that Median filter produces better result than Mean filter. However, PSNR (Signal-to-Noise Ratio) plots of the Mean and Median filtering for Figure 6 and Figure 7 can be used for comparison of the two filter methods. The higher the PSNR, the better the quality of the filtered, or reconstructed image.

Figure 7: Example using Mean filter

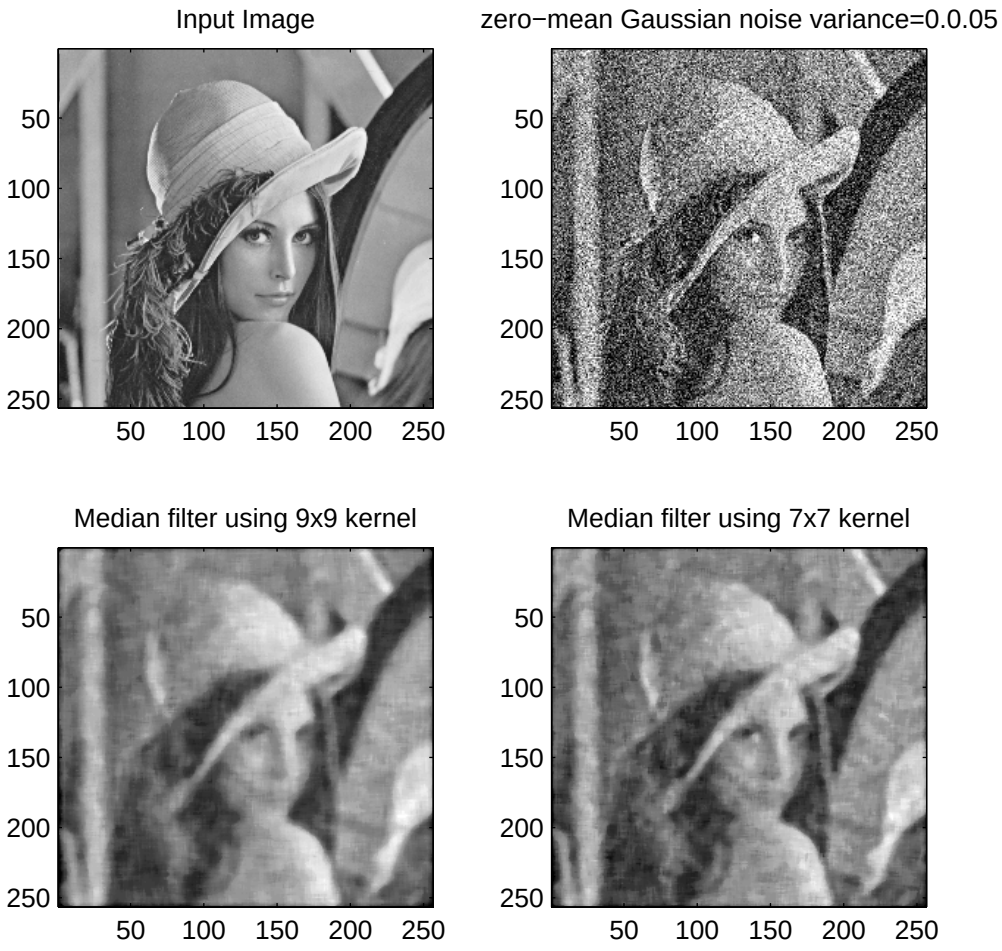

Figure 8: Examples using Median filter

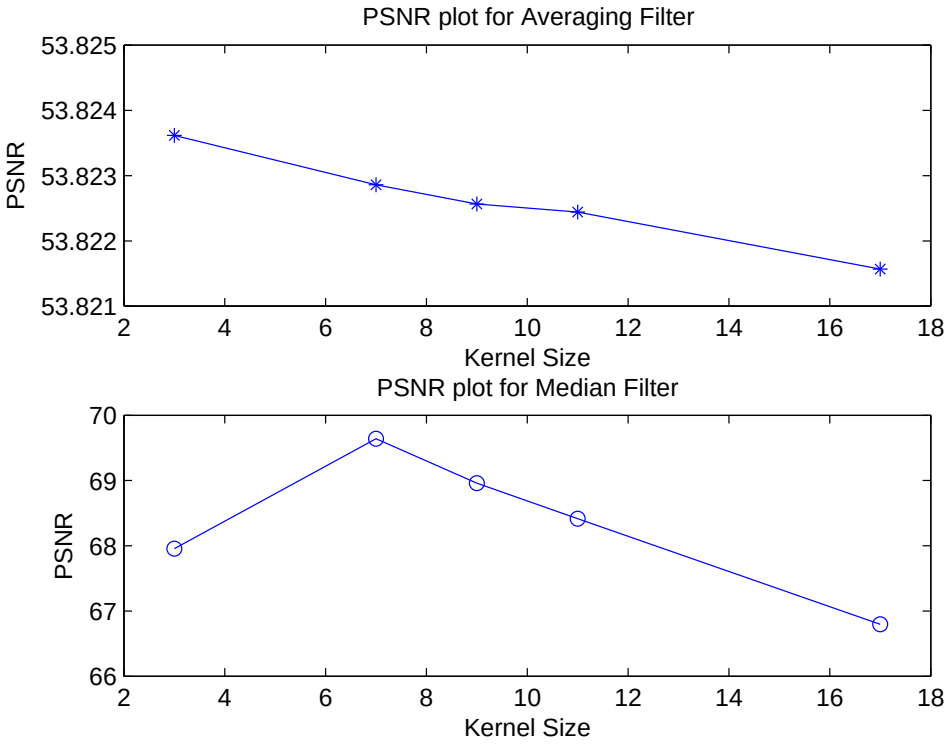

Figure 9: PSNR plots for the Mean and Median filter results using adjusted kernel sizes: 3×3, 7×7, 9×9, 11×11, and 17×17. The PSNR for the Mean filter decreases as window size increases, which implies that there will be significant loss in image quality. Although the PSNR for the Median filter technique generally decreases as the window size increases, the bottom plot shows higher PSNR values compared with the corresponding PSNR values for the Averaging filter technique.

4.3 Noise Suppression by Compression Using the SVD Blocks

Noisy elements in our sample image contain representation of the base images that contain spots. These base images correspond to the singular vectors and singular values of matrix A, see Figure 13. For the SVD block denoising procedure, we want to divide the image sample A into square blocks of size $b \times b$ by forming a $w \times l$ block matrix A. Singular values and corresponding singular vectors will contain complete information about the image blocks. The noise is reflected on the changes in singular values and singular vectors, and this is true for the SVD representation of the square $b \times b$ blocks as well as the whole image [8]. Moreover, high frequency information also corresponds to noise. An optimal threshold can be chosen to discard higher frequency values. We can discard small singular values that correspond to these higher frequency values by truncating the SVD of each blocks. Algorithm for nonlinear image noise filtering based on SVD processing of image blocks is presented by Devcic et. al. [8].

A given image is divided into blocks by the following steps and compressed by removing singular values less than a threshold value.

1. Divide the $m \times n$ matrix A into $b \times b$ submatrices $A^{(w,l)}$ such that, $1 \leq w \leq m/b$.
2. Transform each submatrix $A^{(w,l)}$ into $A_1^{(w,l)}$ by using the SVD. Set singular values that are smaller than a threshold ϵ equal to zero.
3. Collect all (i, j) elements of $A_1^{(w,l)}$ to make an $m/b \times n/b$ matrix $A_2^{(i,j)}$.
4. The $A_2^{(i,j)}$ matrices are put in the (i, j) position to produce the $m \times n$ matrix A_3.

After applying noise to an input sample image, the above procedure tries to suppress noise by discarding small singular values. Using the block SVD procedure did not work well for noisy data, see Figure 10. Instead, we applied the algorithm for compression of a signal without noise.

4.4 The SVD Inverse Filtering

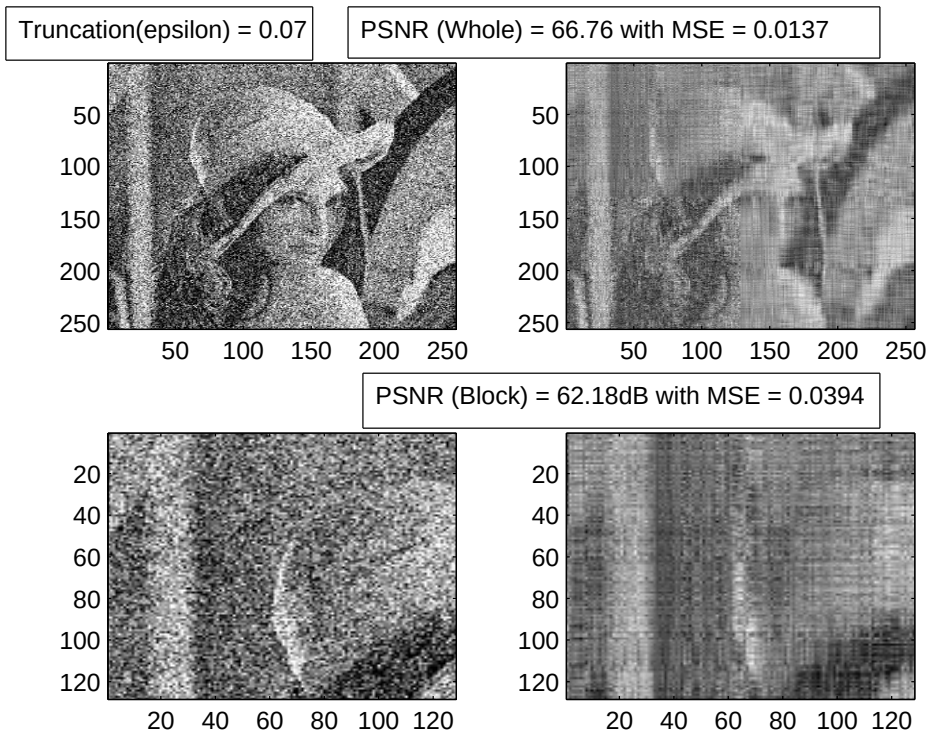

Figure 10: Figure shows noise suppression by computing the SVD of 128×128 blocks and setting singular values less than epsilon $= 0.07$ to zero.

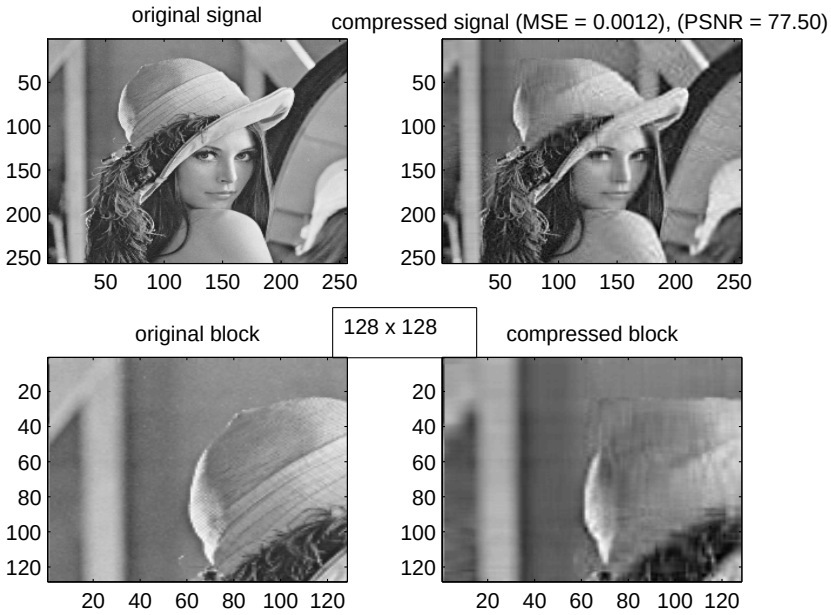

Figure 11: Image compression using the SVD blocks of 128×128.

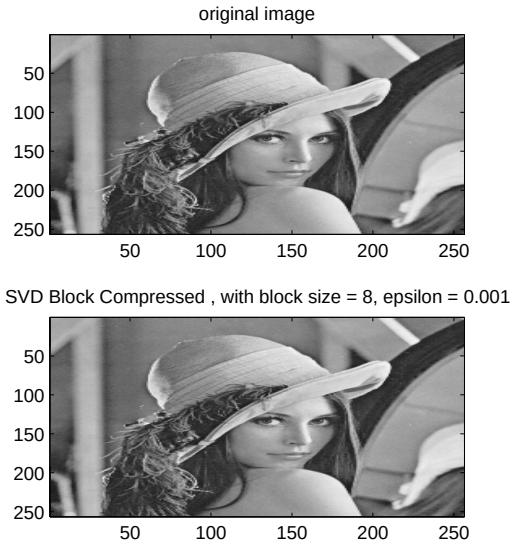

Figure 12: Input signal was compressed by applying the SVD of its 8×8 blocks and then discarding small singular values less than the threshold value, epsilon $= 0.001$, PSNR $= 102.21$, and MSE $= 3.91 \times 10^{-6}$.

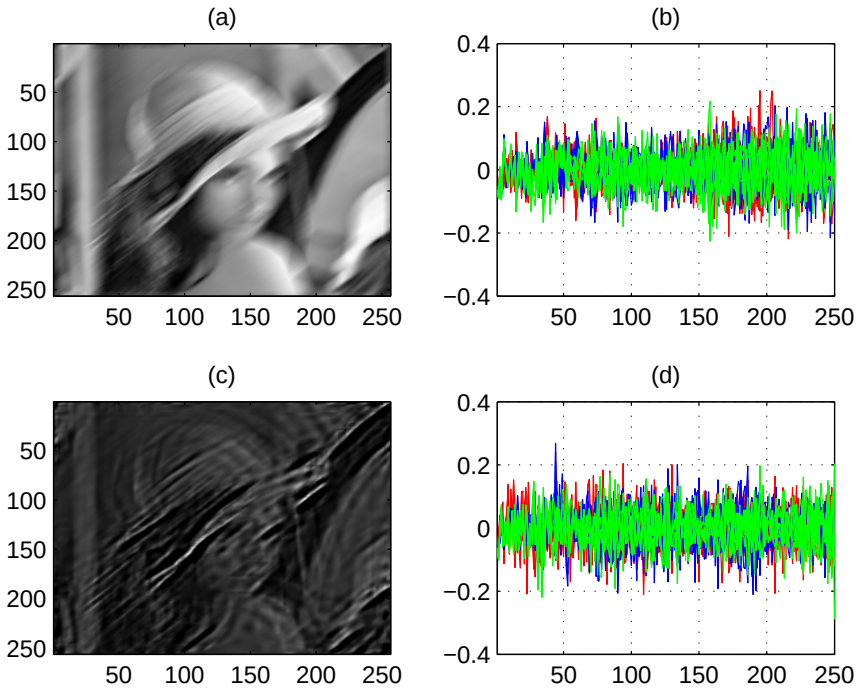

Figure 13: Displaying the number of basis vectors affected by noise. Logarithmic plots of normalized eigenvectors provide a better way to visually detect the noise artifacts.

Digital image processing problems can often be modeled by using systems of linear equations. Systems of linear equations are one of the most fundamental and important problems in linear algebra, and are defined in the form of finding a solution x to the equation

$$Ax = b. \tag{13}$$

Our sample filtering approach can be modeled using the equation $b = Ax + \eta$, where b represents data affected by error, A is the error or blurring factor, x represents the sample image, and η is the error data itself [3]. For numerical definitions, we used the notation δ to represent the perturbation or error factor of our data. The SVD filtering for this method assumes a blurred signal b is the result of a spatially invariant blurring matrix A multiplied by the true signal x. In light of [22], the image restoration model for a PSF matrix $A \in \mathbb{R}^{m\times n}$ using the integral form is expressed as

$$b(m) = \int_{\phi} A(m,n)x(n)dn + \eta(m),$$

where ϕ is a closed region containing the domain of the image. The approach is to find the best reconstruction of the true signal x using the SVD representation of A.

Recall that the full SVD factorization

$$A = \begin{bmatrix} u_1 & u_2 \dots u_r & u_{r+1} \dots u_m \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & & \end{bmatrix} \begin{bmatrix} v_1 & v_2 \dots v_r & v_{r+1} \dots v_n \end{bmatrix}^T \tag{14}$$

reveals the nullspace and range of the matrix A. More specifically, the range of A is spanned by the vectors $u_1, u_2, \dots, u_r$. The range of A^T is spanned by the vectors $v_1, v_2, \dots, v_r$. The nullspace of A is spanned by the vectors $v_{r+1}, v_{r+2}, \dots, v_n$, whereas the nullspace of A^T is spanned by the vectors $u_{r+1}, u_{r+2}, \dots, u_m$. The

columns of V that correspond to the zero singular values of A span the nullspace of A, while the columns of U that correspond to the nonzero singular values of A span the range of A. These properties allow the computation or representation of the vector b from the least squares problem $Ax = b$, such that $b = U\Sigma V^T x$. It follows that

$$b = U\Sigma \begin{bmatrix} v_1^T \\ \vdots \\ v_n^T \end{bmatrix} x.$$

If we project x into V-space, we get the inner product of v_i and x

$$b = U\Sigma \begin{bmatrix} v_1^T x \\ \vdots \\ v_n^T x \end{bmatrix},$$

expanding the matrix Σ and multiplying by the singular values, we get

$$b = U \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{bmatrix} \begin{bmatrix} v_1^T x \\ \vdots \\ v_n^T x \end{bmatrix} = U \begin{bmatrix} \sigma_1 v_1^T x \\ \vdots \\ \sigma_n v_n^T x \end{bmatrix}.$$

The expansion by eigenvectors u_i will give us

$$b = \sum_{i=1}^{n} (\sigma_i v_i^T x) u_i. \tag{15}$$

We have shown that a low-pass filter on the transformation from x-space to b-space has been created, meaning that if we zero out small singular values in the matrix Σ, then the projections of the input vector x onto the corresponding v vectors will not pass through. We can "low-pass" filter noisy measurements in the input [6]. As discussed earlier, most of the noise elements contribute to the high frequency content of an image. Low-pass kernels help us reduce these high spatial frequency components. In contrast, using a high-pass filter helps attenuate low frequency components. By using high-pass kernel, regions in an image of constant brightness are mapped to zero and the fine details of an image are emphasized [24]. High-pass filters have the unfortunate side effect of also highlighting noise along with the fine details of an image, see Figure 14.

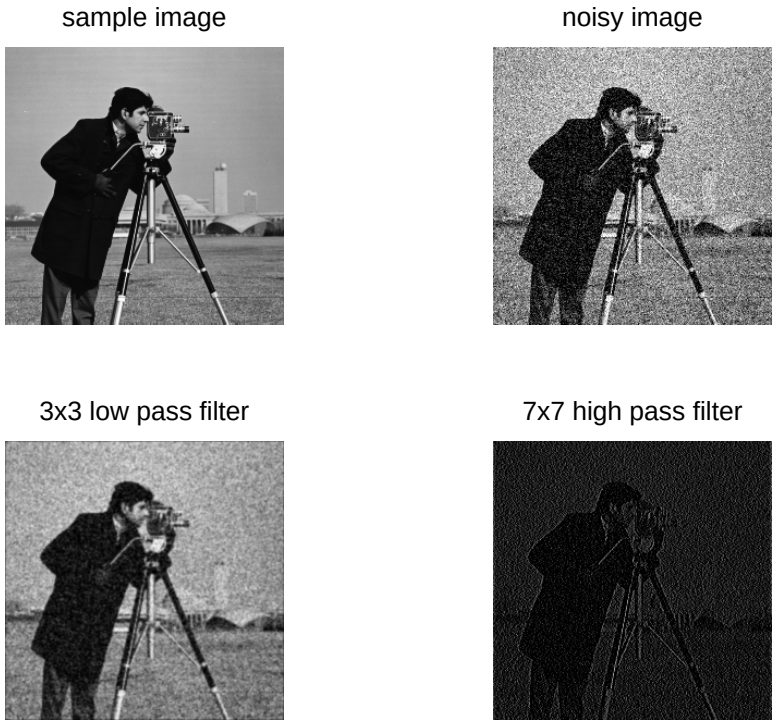

Figure 14: Examples of lowpass and highpass filtering. A Lowpass filter attenuates high frequencies while "passing" low frequencies. A Highpass filter attenuates low frequencies while "passing" high frequencies.

Assuming that all singular values are strictly positive and that A is a square matrix, the inverse of A is given by

$$A^{-1} = V\Sigma^{-1}U^T.$$

Since Σ is a diagonal matrix, its inverse Σ^{-1} is also diagonal, with entries $1/\sigma_i$ for $i = 1, \ldots, n$. If we use the full factorization of square matrix A, the image can be represented as a linear combination of the basis images ($u_i v_i^T$). All the basis images are rank one and form an orthonormal basis for image representation. Similarly,

$$A^{-1} = \sum_{i=1}^{n} \frac{1}{\sigma_i} v_i u_i^T.$$

The inverse solution for our least squares equation can be written as

$$x = A^{-1}b = V\Sigma^{-1}U^T b = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i. \tag{16}$$

In this type of observation, noisy data enters the reconstructed image in the form of the inverted noise $A^{-1}b$. The SVD approach can be used to damp effects caused by division by the small singular values, avoiding any division by zero [13]. Meanwhile, we also know that changes on singular values and singular vectors occur in the presence of noise. We can simply discard small singular values and singular vectors by truncating the above expression by a parameter k. A similar approach is discussed in the previous section, where we minimize the rank of a matrix to reduce noise.

The inverted noise contribution to the above solution is given by

$$A^{-1}e = V\Sigma^{-1}U^T e = \sum_{i=1}^{n} \frac{u_i^T e}{\sigma_i} v_i.$$

Since high frequency information occurs when we divide by small singular value σ_n, we can leave the high-frequency components out by using basis vector representation of the image. Basis vectors (u_i) of this expression provide the representation of a

certain frequency of the image. The amount of information of the frequency can be measured using these basis vectors. High frequency components on the term $u_n^T e$, which are contributed by the division of small singular values σ_n, can be discarded. The error component $u_n^T e$ is magnified to leave high-frequency out by truncating equation (16) expression to

$$x_k = \sum_{i=1}^{k} \frac{u_i^T b}{\sigma_i} v_i \equiv A_k^{\dagger} b. \tag{17}$$

Thus, we have introduced the rank k matrix such that

$$A_k^{\dagger} = [v_1, \ldots, v_k] \begin{bmatrix} \sigma_i & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix}^{-1} \begin{bmatrix} u_1^T \\ \vdots \\ u_k^T \end{bmatrix} = \sum_{i=1}^{k} \frac{1}{\sigma_i} v_i u_i^T.$$

The expression in equation (17) uses the pseudoinverse ($A_k^{\dagger}$) of A. The pseudoinverse (also known as Moore-Penrose inverse) lets us write the solution of the full-rank, overdetermined least squares problem as simply $x = A^{\dagger} b$, [5]. Using pseudoinverse, $A^{\dagger} = V \Sigma^{\dagger} U^T$ leading to $\hat{x} = A^{\dagger} b = V \Sigma^{\dagger} U^T b$, which implies that $\hat{x} = V \Sigma^{\dagger} U^T b$. In general, $\hat{x} = A^{\dagger} b$ is our defined minimum-norm, least squares solution.

We know that $\Sigma^{\dagger}$ is produced by taking the inverses of the non-zero singular values and setting the smallest and zero singular values to zero. The solution x of the linear system is constructed in such a way that a minimum norm solution is considered, where x has no components in the nullspace. Whereas, a least squares solution is considered, where x maps into a b vector in the range, whose difference vector with the measured b has the smallest possible length [21]. If we have a measured vector b with equal number of rows of A, we wish to find the vector $\hat{x}$ which minimizes the error of the following Euclidean norm

$$||b - Ax||_2^2 \tag{18}$$

Given the proof provided by [19], we can compute the least squares solution as follows: given the SVD of A, it follows that

$$||b - Ax||_2^2 = ||U^T b - \Sigma(V^T x)||_2^2.$$

Using Matlab we can compute the solution $\hat{x}$ by setting

$$\hat{x} = V \begin{bmatrix} \frac{c_1}{\sigma_1} \\ \vdots \\ \frac{c_r}{\sigma_r} \\ \alpha_{r+1} \\ \vdots \\ \alpha_n \end{bmatrix},$$

where α_{r+1} through α_n are arbitrarily chosen, and $c = U^T b$. It can be shown that

$$U = \frac{Av_1}{\sigma_1}, \ldots, \frac{Av_r}{\sigma_r}, u_{r+1}, \ldots, u_n,$$

where the columns of U form an orthonormal basis for the range space, and u_{r+1} through u_n are chosen to augment the first r linearly independent columns. This is because if we let $c = U^T b$ and $y = V^T x$, since U is orthogonal it follows that

$$||b - Ax||_2^2 = ||U^T b - \Sigma(V^T x)||_2^2,$$

which can also be expressed as

$$|| \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} - \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} ||_2^2 = || \begin{bmatrix} c_1 - \Sigma_1 y_1 \\ c_2 \end{bmatrix} ||_2^2 = ||c_1 - \Sigma_1 y_1||_2^2 + ||c_2||_2^2,$$

where c_1 and y_1 are vectors in $\mathbb{R}^r$. It can be easily verified that the Euclidean norm $||b - Ax||_2^2$ will be minimal if and only if

$$||c_1 - \Sigma_1 y_1||_2 = 0.$$

The above Euclidean norm can also be expressed in the form

$$||\Sigma \begin{bmatrix} \frac{c_1}{\sigma_1} \\ \vdots \\ \frac{c_r}{\sigma_r} \\ \alpha_{r+1} \\ \vdots \\ \alpha_n \end{bmatrix} - U^T b||_2.$$

We can then compute x such that,

$$\hat{x} = V \begin{bmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = V \Sigma^{\dagger} U^T b.$$

Thus x is a solution to the least squares problem if and only if $x = Vy$. If we set $y_i = (1/\sigma_i) u_i^T b$ for $i = 1, \ldots, r$, we can solve $x = y_1 v_1 + \ldots + y_r v_r$.

4.5 The SVD Deconvolution with Noise and Blur

For our SVD deconvolution experiment, we will use the blurring matrix suggested by Hansen et al to form our filter matrix A. We will approach the 2D filtering process by using implemented PSF and computing its SVD. Provided that PSF matrix A is space invariant, we will use our least squares model and compute a solution $\hat{x}$ that will represent the reconstructed signal. This method is also compared with pseudo-inverse filtering approach. By applying regularization into the deconvolution process, the SVD method can be shown to restore a corrupted image.

The algorithm for the SVD deconvolution using pseudo-inverse approach discussed in section 4.4 and the PSF provided by Hansen et al (2006) is as follows

1. Inputs:
   a. Sample image I, $x = I(:)$, where x is $n \times 1$.
   b. kernel K of size $p \times 1$, where p is odd integer.
2. Construct PSF obtained via [13]
   to get K to form Toeplitz matrix A of size $n \times n$.
3. Multiply A by x to form the blurred image b such that
   $b = A * x$.
4. Deconvolve the blurred by approximating the pseudo-inverse of Σ
   $\hat{x} = V\Sigma^{\dagger}U^{T}b$.

In our results in Figure 12, we have resized the input signal to minimize high computation time that resulted from vectorizing the input and multiplying it by a large matrix A. Although we have reduced the size of our sample x significantly, while losing quality of the picture, the reconstructed result $\hat{x}$ seems to approximately restore the blurred and noised signal b. However, an appropriate blurring may not be applied on our input signal x. We may need to analyze our blurring matrix

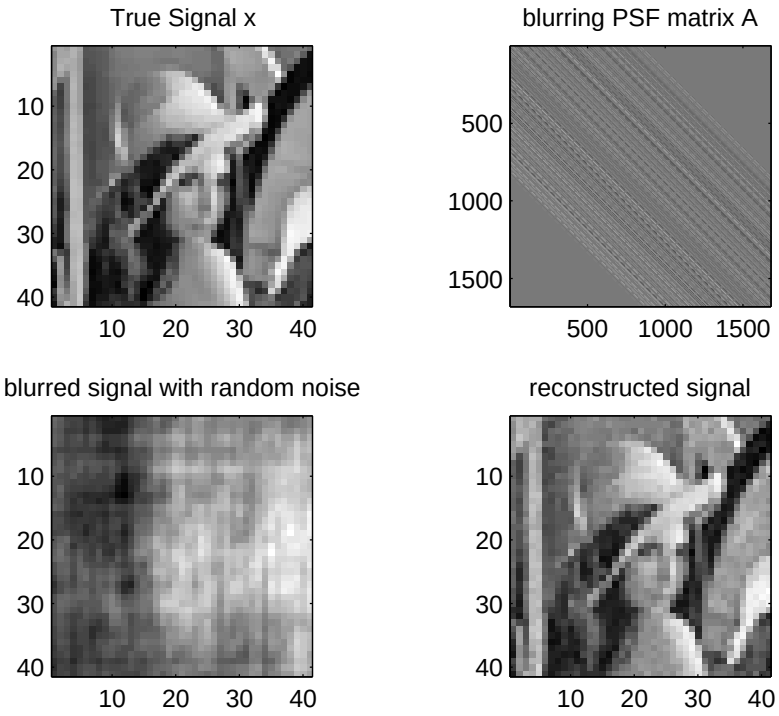

Figure 15: Image filtering example using the SVD. PSF matrix A of size 1681×1681 was created by using a Toeplitz matrix provided by [13]. Input signal Matlab sample "Lena" image resized to size of 41×41 and blurred by A after applying a random noise. The reconstructed signal $\hat{x}$ gives the best approximation for a restored picture.

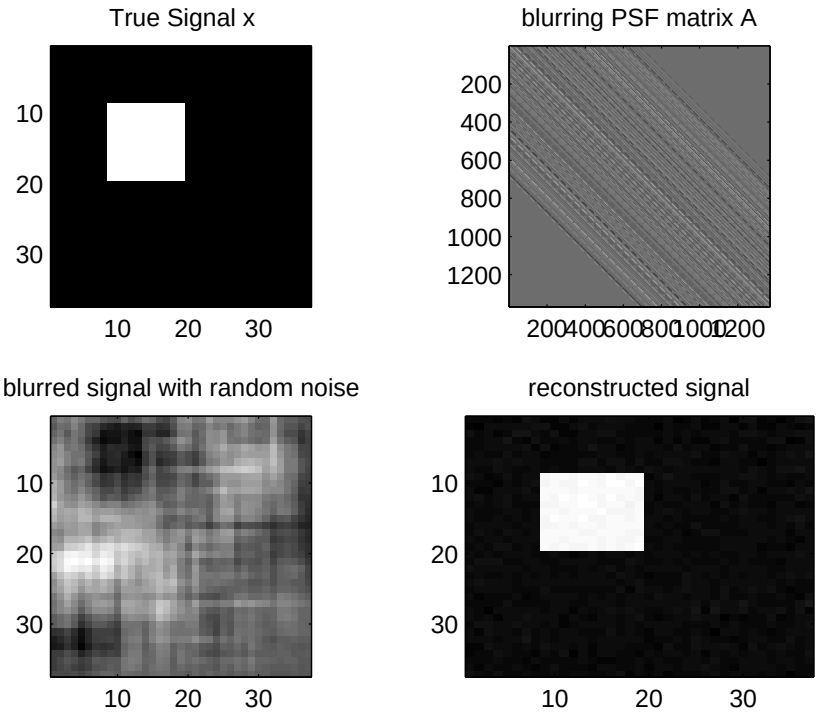

Figure 16: Image filtering example using the SVD. The PSF matrix A of size 1369×1369 was created by using a Toeplitz matrix provided by [13]. Input signal is a constructed binary object of size 37×37. Random noise is still present in the reconstructed signal $\hat{x}$.

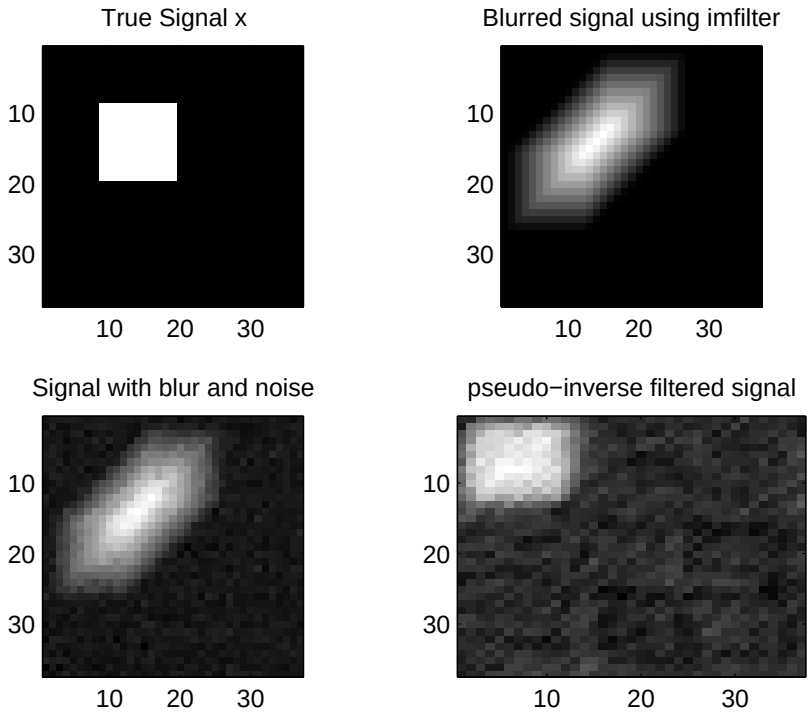

Figure 17: Pseudo-inverse filtering using Matlab generated motion PSF. Frequency components of the PSF that are below the threshold value $\epsilon = 10^{-2}$ are eliminated. The noise term is ignored in the reconstruction process, and the result is highly corrupted with noise due to loss of frequency information.

A. Typically, a blurring matrix or PSF is an *ill-conditioned* matrix. A matrix is ill-conditioned if the condition number is too large. The condition number of our blurring matrix is discussed in the next chapter. In Figure 18, we have plotted the singular values of our PSF matrix comparing with the singular values of signal x. The smallest singular values were not very small. As a result, the condition number was not very large, making the PSF a well conditioned matrix.

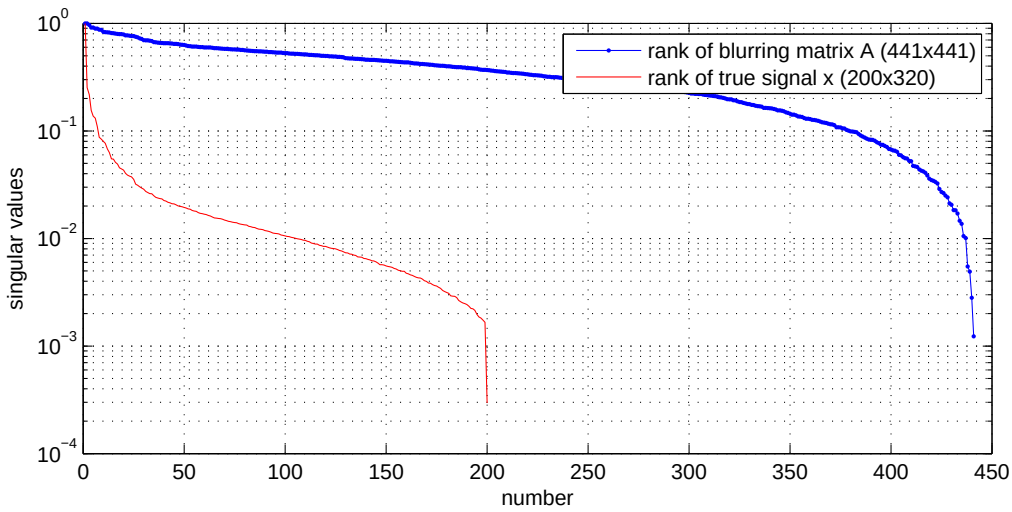

Figure 18: Blurring matrix PSF and signal x singular value plot.

After applying the blurring operation by a well conditioned PSF, the Mean square error between the original and reconstructed signal for our experiment was 5.3056×10^{-4}, while the Peak Signal-to-Noise ratio resulted 80.88 dB. However, we were still able to reconstruct back the signal after applying a significant amout of random noise. For comparing results using MSE and PSNR, we have provided plots for regularized least squares SVD filtering by choosing several regularizing parameters.

We have used several other PSFs for our SVD deconvolution algorithm. In [22], a blur that is spatially invariant is given by decomposing it into Kronecker products. The Kronecker product, denoted as $\otimes$, is a matrix operator that maps

two arbitrarily dimensioned matrices into a larger matrix that has a special block structure. The Kronecker product of matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times n}$ is given by

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \vdots & \vdots & & \vdots \\ a_{n1}B & a_{n2}B & \dots & a_{nn}B \end{bmatrix}$$

Using the SVD of A and B, we can obtain the Kronecker product factorization of C, such that

$$C = (U_A \otimes U_B)(\Sigma_A \otimes \Sigma_B)(V_A \otimes V_B)^T.$$

The factorization P, which corresponds to our PSF matrix can be computed using the SVD such that

$$P = \sum_{k=1}^{r} \sigma_k u_k v_k^T,$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$, and thus $a_k = \sqrt{\sigma_k}u_k$ and $b_k = \sqrt{\sigma_k}v_k$. Using the Kronecker product resulted with a large matrix of size $n^2 \times n^2$, because of all possible products between elements of A and B. For a 256×256 input signal, maximum variable size allowed to compute using the Kronecker product is exceeded. In our least squares model, another way to allow a matrix multiplication in the convolution step is to multiply both sides of input signal X by A, such that, $B = AXA^T$. If we let $A = U\Sigma V^T$, then $U^TAV = \Sigma$, and the factorization using the fact that $B = AXA^T$ can be written as

$$U^TBU = (U^TAV)V^TXV(V^TA^TU) = \Sigma(V^TXV)\Sigma,$$

which implies that there is a solution $\hat{X}$ provided that $\hat{X} = V^TXV$, and $X = V\hat{X}V^T$. Thus,

$$\hat{B} = \Sigma\hat{X}\Sigma.$$

The soultion is given by

$$\hat{X}_{i,j} = \frac{\hat{B}_{i,j}}{\sigma_i\sigma_j}.$$

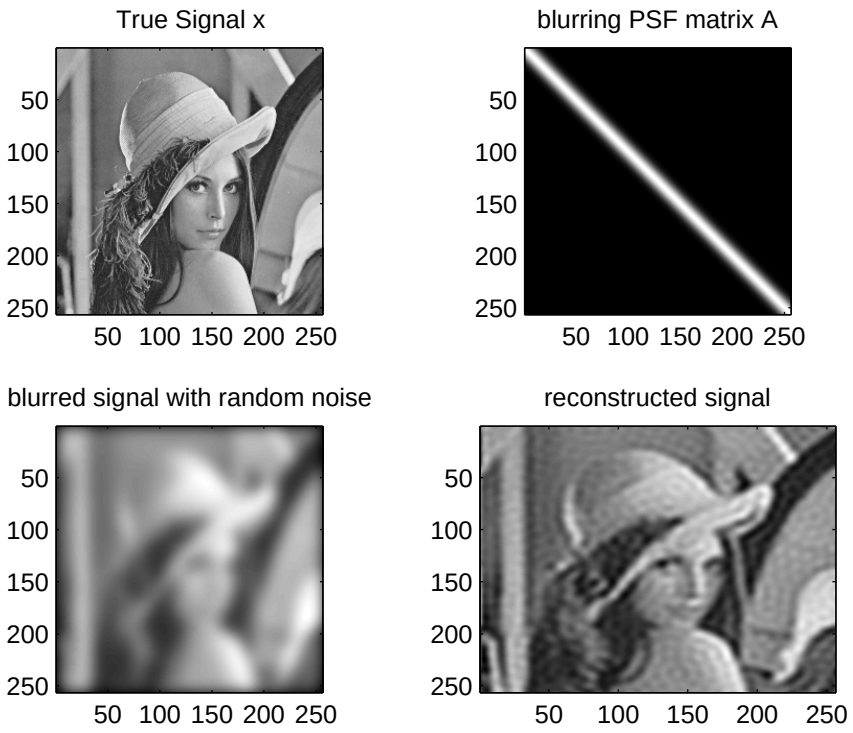

Figure 19: SVD deconvolution using the matrix signal X and PSF P generated using Kronecker products. The MSE between X and reconstructed signal $\hat{X}$ is 0.0036, PSNR = 72.575.

5. EVALUATION OF THE SVD COMPUTATION

5.1 Comparing Obtained Results

Using the SVD, we have shown how to compress the size of matrix $A = U\Sigma V^T$, such that the orthogonal matrices U and V are factored in the form

$$U^T AV = \Sigma = \begin{bmatrix} \Sigma_1 & O \\ O & O \end{bmatrix}.$$

The matrix Σ_1 is nonsingular diagonal matrix consisting of singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. This validates that Σ_1 is the reconstructed form of matrix A and both are considered orthogonally equivalent. If A is a 256 gray scale image, then the representation of each pixel of A requires one byte, that is, eight bits of memory. In the orthogonal matrices V and U, every pixel of U and V needs more than one byte. Although the SVD provides better result for image compression, this full size SVD method is very costly [1]. The table below shows the compression ratio and relative errors calculated for results in Figure 2.

PSNR, the ratio between the peak power of the true signal and the power of the Gaussian noise, measures the amount of mathematical error introduced in an image by compression or noise introduction. PSNR is related to the mean square error (MSE). However, the PSNR is measured using logarithmic scale; in Figures 3 and 9, we have shown Matlab plots for the SVD compression and noise filtering respectively. The peak signal-to-noise ratio (PSNR) and signal-to-noise ratio (SNR) for the input $m \times n$ 256 gray scale image A and its reconstructed image $\hat{x}$ are given by the equations

$$PSNR = 10\log_{10}\left(\frac{255}{\sum_{i=1}^{m}\sum_{j=1}^{n}[A(i,j) - \hat{x}(i,j)]^2}\right) = 10\log_{10}\left(\frac{255}{||A - \hat{x}||_F^2}\right)$$

and

$$SNR = 10\log_{10}\left(\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}A(i,j)^2}{\sum_{i=1}^{m}\sum_{j=1}^{n}[A(i,j) - \hat{x}(i,j)]^2}\right) = 10\log_{10}\left(\frac{||A||_F^2}{||A - \hat{x}||_F^2}\right)$$

	Compression Ratio	Relative Error
20 Singular Values	0.1563	0.0251
50 Singular Values	0.3906	0.0109
85 Singular Values	0.6641	0.0062

Figure 20: Relative error (σ_{k+1}/σ_1) and compression ratio for results in Figure 2.

The relative error decreases as the number of singular values being retained increase. The best compression ratio with the corresponding minimum relative error is given by retaining only 85 singular values. High compression ratio is needed for best for best compression and reconstruction result. The best rank-k approximation of matrix A representing the sample "Lena" image is minimizing $||A-A_k||_2 = \sigma_{k+1}$, which is also the goal of minimizing relative error.

5.2 Condition Number and Numerical Rank of the SVD

In numerical linear algebra, we use the condition number and the numerical rank to evaluate the numerical stability of our inverse problem. Low condition number usually means that computation is well-conditioned or stable. When condition number is high, the computation is ill-conditioned. A small perturbation on A such that $A + \delta A$ will cause relatively large changes in the solutions to $Ax = b$ (for a given error level δ). The condition number with respect to the 2-norm is defined

by $\kappa_2(A) = \sigma_{max}(A)/\sigma_{min}(A)$. If we could measure the condition number of A, this measure could be used to derive a bound for the relative error in the computed solution.

In general, for matrix $A \in \mathbb{R}^{m\times n}$ with rank k, it easily seen that $k \leq \min\{m, n\}$. When $k = \min\{m, n\}$, then matrix A has full rank. Otherwise, A is said to be rank deficient. In our least square model $Ax - b = \eta$, if A is nearly rank deficient (σ_{min} is small), then the solution x is ill-conditioned and possibly very large. There have been studies addressing the ill-conditioned situation of rank-deficient least squares problem and how to solve them accurately. The system $Ax = b$ can be changed to a symmetric positive definite system, in which $x^TAx > 0$ for all $n \times 1$ vectors and $A^T = A$, by solving the normal equations $A^TAx = A^Tb$. This includes the least squares problem $min_x||Ax - b||_2$. Solving the equation $Ax = b$ will lead to slow convergence if A is ill-conditioned and since the condition number of A^TA or AA^T is the square of the condition number of A. The goal was to compute the minimum norm solution x that is possibly unique. The condition number of this minimum norm solution is determined by the smallest nonzero singular value and also the rank of A. The main difficulty is that the rank of the matrix changes discontinuously as a function of the matrix. The number of nonzero singular values of the blurring matrix A is 571, and its smallest singular value is $\sigma_n = 5.81 \times 10^{-51}$. The condition number of the matrix is 1.62×10^{50}.

Each computed singular value $\hat{\sigma}_i$ satisfies $|\hat{\sigma}_i - \sigma_i| \leq O(\epsilon)||A||_2$, where $O(\epsilon)||A||_2$ is magnitude of the perturbation of matrix A. The condition number can be increased from $1/\sigma$ to $1/\epsilon$. With backward stability, the computed SVD will be the exact SVD of a slightly different matrix: $\hat{A} = \hat{U}\hat{\Sigma}\hat{V}^T = A + \delta A$, with $||\delta A|| = O(\epsilon)||A||$. Computed singular values less than $O(\epsilon)||A||_2$ can be treated as zero, because roundoff makes it distinguishable from zero [5]. This could raise the smallest singular value from ϵ to 1 and correspondingly decrease the condition

number from $1/\epsilon$ to $1/\sigma = 1$.

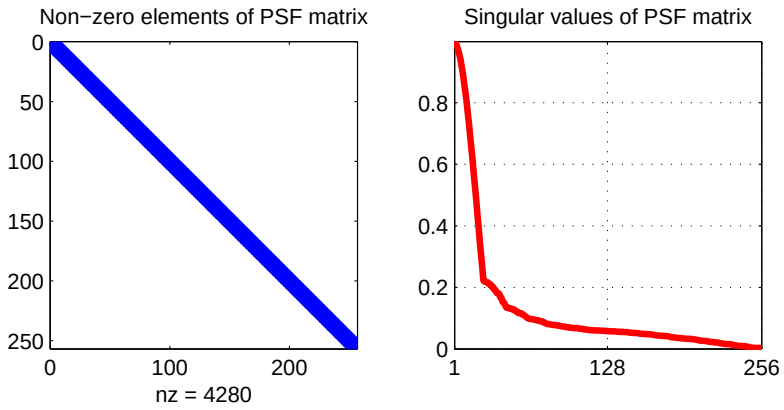

Figure 21: Displaying zeros and singular values of a 256×256 well conditioned blurring matrix A, with number of nonzero elements = 4280.

For a blurring matrix, all the singular values decay gradually to zero, while clustering at zero, see Figure 21, [22]. As a result, the condition number $\kappa(A) = \sigma_1/\sigma_n$ is very large. The minimum norm solution x is unique and may be well conditioned if the smallest nonzero singular value is not too small.

5.3 Reguralization Technique and Dealing with the Inverse SVD

In numerical linear algebra the singular values can be used to determine the effective rank of a matrix, as rounding error may lead to small but nonzero singular values in a rank deficient matrix. Obtaining the singular value decomposition of a large matrix, such as A, can be very computationally expensive. In our SVD filtering model, we applied an inverse operation to reconstruct the corrupted image.

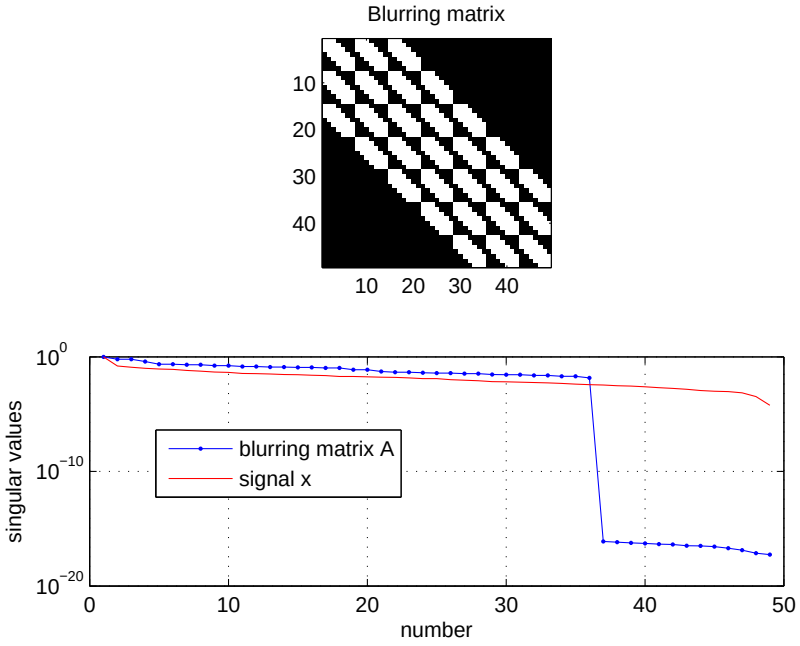

Figure 22: Singular values of PSF matrix A decay gradually and cluster at zero. The condition number of the PSF matrix is 7.975×10^{20}.

Inverse filtering is usually considered an ill-conditioned problem. Regularization techniques must be applied to stabilize the numerical methods.

Any regularization method tries to analyze a well-posed problem whose solution approximates the solution of the original ill-posed problem. The smallest singular value of our PSF matrix A is σ_n, which is shown to be very small. The solution x is ill-conditioned and possibly very large. If x minimizes $||Ax - b||_2$, then $||x||_2 \geq |u_n^T b|/\sigma_n$, where u_n is the last column of matrix U. If we perturbing b, such that, $b + \delta b$ can change x to $x + \delta x$, where $||\delta x||_2$ is as large as $||\delta b||_2/\sigma_n$, see [5] for the Proofs. If the reconstruction is given by $x = A^{\dagger} b = V \Sigma^{\dagger} U^T b$, so $||x||_2 = ||\Sigma^{\dagger} U^T b||_2$. To regularize a rank deficient least squares problem, where rank of A is less than n, we will look for the unique solution of the smallest norm $||Ax - b||_2$ characterized

by the proposition presented in [5]. We write the SVD of A as

$$A = \begin{bmatrix} U_1, U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & O \\ O & O \end{bmatrix} \begin{bmatrix} V_1, V_2 \end{bmatrix}^T = U_1 \Sigma_1 V_1^T,$$

where Σ_1 is $r \times r$ and nonsingular, and U_1 and V_1 have r columns. If we let $\beta = \sigma_n$, the smallest nonzero singular value of A, then

1. All solutions x can be written $x = V_1 \Sigma_1^{\dagger} U_1^T b + V_2 z$, where z is an arbitrary vector.

2. The solution x has minimal norm $||x||_2$ precisely when $z = 0$, in which case $x = V_1 \Sigma_1^{\dagger} U_1^T b$ and $||x||_2 \leq ||b||_2/\beta$.

3. Changing b to $b+\delta b$ can change the minimal norm solution x by at most $||\delta b||_2/\beta$.

The Proofs of this proposition are also given in [5], p.126.

There are several ideas required to achieve well-posedness or stable solution. Some examples can be, imposing restriction of the data; changing the space or the topology, or modification of the operator itself. For least square model $b = Ax + \eta$, the regularization method constructs the solution as

$$\min_x [u(a, b) + \beta v(a)],$$

where $u(a, b)$ describes how the real image data is related to the degraded data. In other words, this term models the characteristic of the imaging system; $\beta v(a)$ is the regularization term with the regularization operator v operating on the original image x; and the regularization parameter β is used to tune up the weight of the regularization term [20]. The spatial dependence or independence of the noise term A can also help impose regularization on the model.

Figure 18 displays the PSNR and MSE plots after implementing least squares SVD filtering (using the same data in Figure 12) by applying regularization and adjusting the regularization term β ("epsilon in the plot"). As the regularization term increases, we have seen a sharp rise in PSNR and a sharp decline for MSE to a certain point. This means that the regularization term seems to minimize the

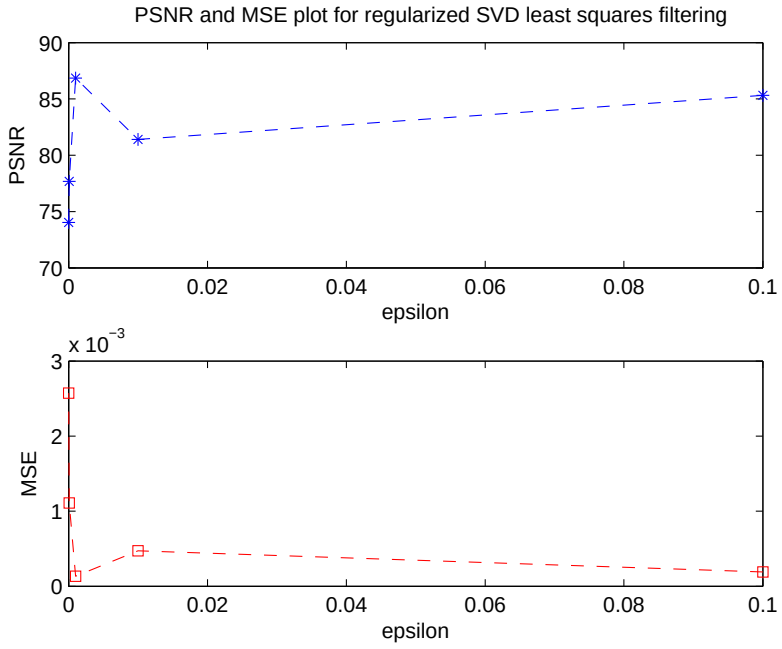

Figure 23: Plots for the SVD inverse filtering with regularization. Results in Figure 12 display the SVD inverse filtering without regularization. Here, the same image and PSF data were used to display PSNR and MSE plots for regularized inverse SVD filtering.

amplified error caused by division with small singular values upto a certain regulazation point. We can apply regularization in the inverse SVD filtering obtained in Figure 13, and Figure 12, and show a better reconstruction. We can see a better reconstruction by visual inspection for Figure 13.

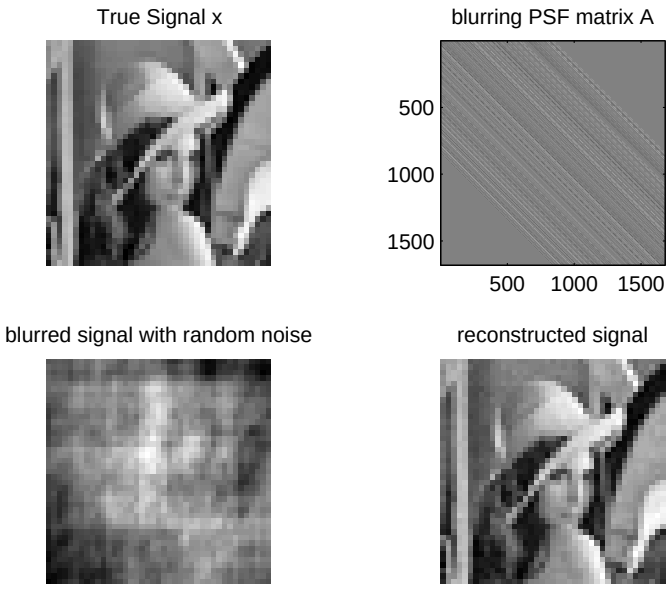

Figure 24: The SVD inverse filtering using regularization $\delta = 0.1$. PSNR = 86.73db and MSE $= 1.38 \times 10^{-4}$.

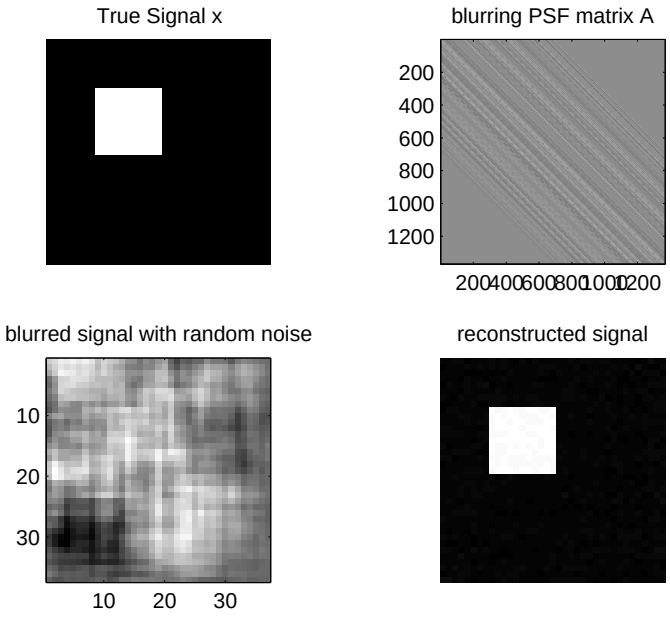

Figure 25: Inverse SVD filtering of the binary object x. The restored object by regularization $\delta = 0.5$, at the bottom right, gives a better result compared to the reconstructed signal in Figure 13. PSNR $= 89.54$ and MSE $= 7.23 \times 10^{-5}$.

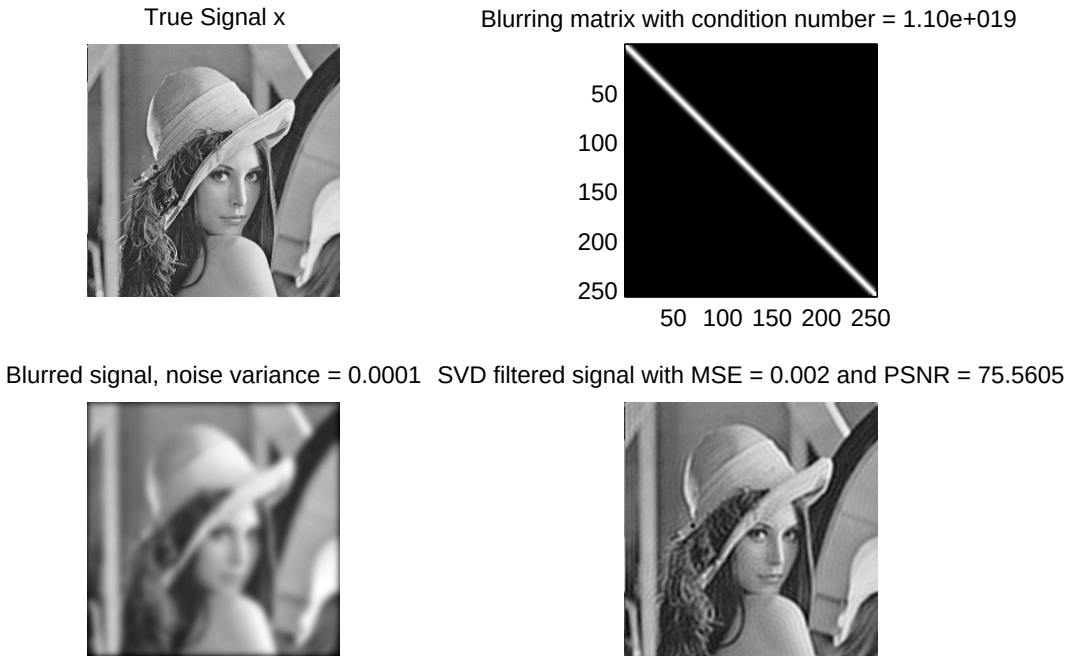

Figure 26: The Figure shows SVD least squares filtering using the Kronecker products for the PSF matrix and regularization delta = 0.01.

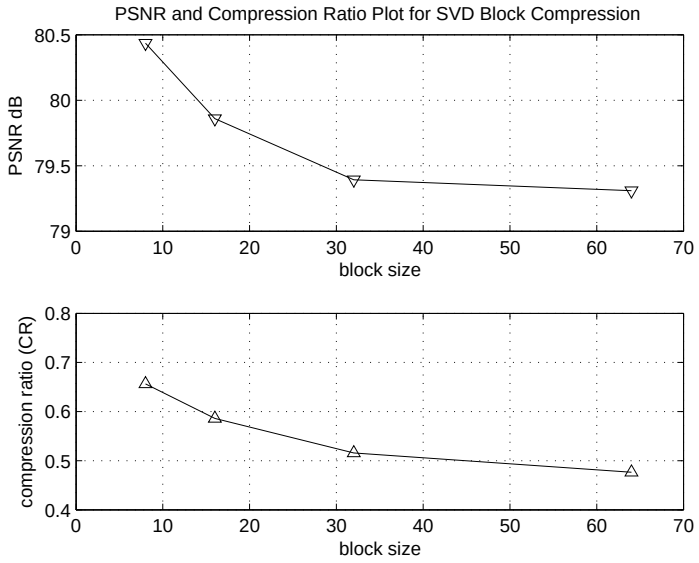

Figure 27: The Figure shows the overall PSNR and compression ratio plots for several SVD blocks. The PSNR decreases as the block size increases. The compression ratio using SVD blocks also decreases with an increase in block size.

6. CONCLUSION

The SVD can be shown to improve the appearance of distorted or noised images by implementing it in several filtering techniques. We have discussed the SVD block denoising technique, where we divide a sample image matrix into square blocks and calculate the SVD of each block to reduce the changes in singular values and singular vectors. Filtering noise is performed through eliminating changes in singular values and singular vectors that resulted from additive white Gaussian noise. Although, denoising using the block operation resulted with high mean square error and low peak signal-to-noise ratio, it has been shown to work well for compression. The block SVD algorithm effectively compresses the image, while reducing computation time.

The widely known SVD application to image compression is discussed. SVD compression method allows us to reduce the size of the matrix representing the image by taking only few singular values of it. This led to the compression of images without losing image quality. The SVD compression is similar to the noise filtering method with the notion that insignificant singular values can be avoided to obtain a compressed or restored image. Compression using the SVD blocks was much more efficient than SVD compression of the entire matrix.

We have also shown an implemented SVD method for deconvolving blurred images by a given blur parameter PSF. We have modeled the filtering operation using least squares and computed the solution that minimizes the norm of the least square equation. Using this technique, an inverse operation is done as the blur element enters the reconstructed image in the form of an inverted noise. In this approach, we have shown how to damp the effects caused by division by the small singular values. The SVD filtering in images can yield better results compared to other methods of filtering, such as median and mean filtering techniques.

The SVD computation is not recommended for large matrices and any algorithms that address ill-posed problems. Because of large computation that results from vectorizing input signal, SVD deconvolution is not a reliable approach for noise filtering. We can avoid large computation by using Kronecker products provided by Hansen et. al. (2006) for our blurring matrix. Stability for the SVD least squares solution was achieved by imposing certain regularity or modification of the filter operator itself. After applying regularization using SVD deconvolution, we were able to restore a corrupted signal with higher PSNR values.

References

[1] R. Ashin, A. Morimoto, M. Nagase, R. Vaillancourt, "Image Compression with Multiresolution Singular Value Decomposition and Other Methods," Mathematical and Computer Modeling, 41: 773-790, 2005.

[2] S. Belkasim, Selected from power point silde presentations and lecture notes in Digital Image Processing course, June 2007, Aug. 2007.

[3] J. Chung, "Filtering Methods for Image Restoration," Honors Thesis, Department of Mathematics and Computer Science, Emory University, May 2004.

[4] L. David, Linear Algebra and Its Applications, Addison-Wesley, NY, ed. 7, 2000.

[5] J. Demmel, "Applied Numerical Linear Algebra," SIAM Society for Industrial and Applied Mathematics, 1997.

[6] R. J. DeSa and I. B. C. Matheson, "A Practical Approach to Interpretation of Singular Value Decomposition Results," Methods in Enzymology, Vol. 385.

[7] Z. Devcic and S. Loncaric, "Blind Restoration of Space-Invariant Image Degredations in the Singular Value Decomposition Domain," IEEE, 2001.

[8] Z. Devcic and S. Loncaric, "Non-linear Image Noise Filtering Algorithm Based on SVD Block Processing," University of Zagreb, Department of Electroic Systems and Information Processing.

[9] Diffraction Limited, Glossary, *http://www.cyanogen.com/help/maximdl/MaxIm-DL.htm kernel Filter.htm*

[10] S. Gilbert, "Introduction to Linear Algebra," SIAM, Wellesly-Cambridge, 3rd ed., 1993.

[11] R. C. Gonzalez, "Digital Image Processing," Second Ed Pearson Education, 2004.

[12] P. C. Hansen, "Truncated Singular Value Decomposition Solutions to Discrete Ill-posed Problems Ill-determined Numerical Rank," SIAM J, 1990.

[13] P. C. Hansen, J. G. Nagy, D. P. O'Leary, "Deblurring Image: Matrices, Spectra, and Filtering," SIAM Fundamentals of Algorithms 3, 2006.

[14] R. A. Horn and C. R. Johnson, Matrix Analysis, Cambridge University Press, Cambridge, 1991.

[15] R. A. Horn and C. R. Johnson, Matrix Analysis, Cambridge University Press, Cambridge, 1994.

[16] K. Y. Issa and A. V. Gopal, "Method for Removing Streak Artifacts in Medical Images," United States Patent 5987347, Gen. Electric (US), 16 Nov. 1999, *http://www.freepatentsonline.com/5987347.html*

[17] K. Konstantinides, B. Natrajan, and G. S. Yovanof, "Noise Estimation and Filtering Using Block-Based Singular Value Decomposition," IEEE Transactions on Image Processing, Vol. 6, NO. 3, March 1997.

[18] S. Leach, "Singular Value Decomposition -A Primer," Learning Dynamical Systems: A Tutorial, Department of Computer Science, Brown University, 1995.

[19] J. S. Leon, "Linear Algebra with Applications," ed. 6, 2002.

[20] C. C. MacDuffee, The Theory of Matrices, J. Springer, Berlin, 1933; Chealsea, New York, 1975.

[21] P. M. Mai, Singular Value Decomposition, Institute of Geophysics, Zuerich, Switzerland, 2007, *http://www.seismo.ethz.ch/staff/martin/courses/inv/SVDaddendum.pdf*

[22] J. G. Nagy and D. P. O'Leary, "Fast Iterative Image Restoration with a Spatially-Varying PSF," Scientific Literature Digital Library, pp. 388-399, 1997.

[23] B. R. Payne, "Accelerating Scientific Computation in Bioinformatics by Using Graphics Processing Units as Parallel Vector Processors," PhD Dissertation, 2004.

[24] S. Qureshi, Embedded Image Processing on the TMS320C6000TM: Examples in Code Composer StudioTM and MATLAB, 2005.

[25] E. Steve, Steve on Image Processing: Image Deblurring Introduction, Matlab central, 2007 *http://blogs.mathworks.com/steve/2007/08/13/image-deblurring-introduction/*.

[26] K. A. Syed, "The Discrete Cosine Transform (DCT): Theory and Application," 2003.

[27] O. O. Veragara Villegas, R. P. Elias, and V. G. Cruz Sanchez, "Singular Value Decomposition Image Compression System for Automatic Object Recognition," From Proceeding (505) Advances in Computer Science and Technology, 2006.

[28] R. Vio, J. Nagy, and W. Wamsteker, "Multiple-image composition and deblurring with spatially-variant PSFs, Astronomy and Astrophysics," 2003.