Numerical Methods for Edge-Preserving Image Restoration

A dissertation

submitted by

Donghui Chen

In partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

in

Mathematics

TUFTS UNIVERSITY

August 2012

ADVISOR: Misha E. Kilmer

Abstract

Many digital image applications rely on the image quality. Unfortunately, images often are degraded by noise and blur during the formation, transmission, and recording processes. Hence, image restoration is a necessary processing step. Many current image restoration methods lose edge information while removing the defects. This work focuses on developing accurate mathematical models and efficient numerical algorithms for edge-preserving image restoration problems. We first present a new regularization parameter-choice algorithm broadly applicable to several image restoration approaches, which is suitable for large-scale problems. Next, we consider three different types of image restoration scenarios, and present algorithms for each. The first problem we consider is image denoising. In particular, we explore the application of multigrid methods in solving the nonlinear anisotropic diffusion denoising problem. Secondly, we introduce a projection-based algorithm to solve image deblurring problem. For some applications, it is necessary to solve a least squares problem with nonnegative constraints. Therefore, we also present a novel multiplicative nonnegative least squares algorithm for image super-resolution and color image labeling. The convergence analysis of each algorithm is studied. Finally, we demonstrate the applications of the algorithms with many numerical experiments.

Acknowledgements

As I am about to finish this journey through Ph.D. program at Tufts University, I gladly claim this journey to be a pleasant and rewarding one. However, it would not end up this way without the help of many others.

First of all, I would like to thank my advisor Misha Kilmer for the valuable guidance, advice and support. I could not imagine how I survive without her constant support and guidance on research. She also enabled numerous opportunities of teaching, attending conferences, and internships during the past four years.

I would also like to thank Scott MacLaclan for his mentoring, guidance and support. We have been working together since the very first day at Tufts. I feel very privileged to have had the opportunity of working with him and learning from him.

I would also like to thank Matthew Brand at Mitsubishi Electric Research Laboratories and Thomas Höft for being part of the thesis committee. I have benefitted immensely from my collaboration with Matt on the NNLS problem and my time spent at MERL.

I would also like to express my gratitude to Robert Plemmons at Wake Forest University, who lead me to the wonderful numerical optimization world.

In addition, I am grateful to the faculties, staffs, and graduate students in the Department of Mathematics for their help and providing a fun atmosphere to live and work.

I wish to thank my parents, my mother-in-law, and my sister, who have always encouraged me to pursue my education and provided many helping hands. Finally, I wish to thank my wife, Xiao, for her support, friendship, and understanding.

To my family

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Digital images play a more and more important role in human life. Even though there exist vast differences among digital imaging devices, they all share the same underlying physical processes – sampling and quantizing the image from a continuous world. Limited by physical constraints, the obtained digital images are often degraded by defects [25, 106]. Developing efficient numerical image restoration methods has been a challenge for researchers in different areas. This dissertation focuses on developing accurate mathematical models and efficient numerical algorithms for different image restoration problems.

1.1 Image Restoration

The goal of image restoration is to remove defects which degrade image quality. Typically, the degradation can be identified as noise and blur [7, 25, 40, 49, 95]. The process of removing noise is named "denoising", and "deblurring" refers to the process of removing blur. The difficulty of image restoration comes from the ill-posedness of the underlying mathematical models and tremendous computational work involved in the restoration algorithms [58]. In this section, we briefly review the mathematical models that are commonly used in the literature.

1.1.1 Image Denoising

Image denoising is one of the oldest concerns and is still considered a necessary preprocessing step for many applications [7, 25, 71, 95, 112, 113]. Before discussing any existing image denosing technique, we first define the digital image formation model in spatial domain Ω

as

$$\tilde{X}(s,y) = X(s,y) + \mathcal{E}(s,y) \quad \text{for} \quad (s,y) \in \Omega, \tag{1.1}$$

where $\tilde{X}(s,y)$ is the observed noisy (gray scale) intensity value at spatial position (s,y), $X(s,y)$ represents the "true" value, $\mathcal{E}(s,y)$ stands for the noise perturbation at pixel (s,y), which is often assumed to be independent and identically distributed (iid). Many researchers assume that $\mathcal{E}$ is white Gaussian noise [25], which is also our assumption in this work. In practice, the images are stored as matrices. We abuse notation slightly and reuse $\tilde{X}$, X, and $\mathcal{E}$ to denote matrices as well. For example, the (i,j) entry in the matrix X refers to the continuous X sampled at the position (s_i, y_j). Thus,

$$\tilde{X}_{i,j} = X_{i,j} + \mathcal{E}_{i,j}.$$

Figures 1.1a and 1.1b compare the noise-free cameraman image, X, and corresponding noisy image, $\tilde{X}$.

Remark: In this work, most of the images used in experiments are 2-dimensional. In the computation, we transform the 2-D $m \times m$ image into a vector by stacking each column in the image, i.e.

$$\tilde{x} = \text{vec}(\tilde{X}) = [\tilde{X}_{1,1}, \tilde{X}_{2,1}, \cdots, \tilde{X}_{n,1}, \tilde{X}_{1,2} \cdots \tilde{X}_{n,2} \cdots \tilde{X}_{m,m}]^T,$$

where the exponent T means vector transformation. Then in vectorized form, if X is $m \times m$,

$$\tilde{x} = \text{vec}(\tilde{X}) = \text{vec}(X) + \text{vec}(\mathcal{E}) = x + \eta \in \mathbb{R}^n,$$

with $n = m^2$. In the numerical experiments, the noise level is defined as

$$\text{noise level} = \frac{||\eta||_2}{||x||_2}.$$

The methods used to solve the image denoising problem fall into two categories: energy methods [7, 8, 47] and partial differential equation (PDE)-based methods [7, 28]. In this work, we focus on the PDE-based denoising methods. Perhaps the simplest and best studied PDE-based method is the linear diffusion process. Consider the heat equation [7]:

$$\frac{\partial X}{\partial t}(t; s, y) = \Delta X(t; s, y) \quad \text{in} \quad (0, T) \times \Omega, \tag{1.2}$$

with the original noisy image $\tilde{X}(s,y)$ as the initial condition. In the above equation, Δ denotes the Laplace operator on the spatial variables s and y. The solution $X(t; s, y)$ is the restored version of the initial noisy image $\tilde{X}(s,y)$. Recall (1.2) has a unique solution,

$X(t;s,y)$, which is a convolution of the initial noisy image $\tilde{X}(s,y)$ with a Gaussian kernel, $G * \tilde{X}$ where $G(t;s,y) = \frac{1}{4\pi t}\exp\left(-\frac{s^2+y^2}{4t}\right)$. The drawback of this simple model is that it loses edges or textures in the restored image because of the low-pass convolution filter, see Figure 1.1c.

Remark: Throughout this work, we use Matlab command

```
colormap('Hot');
```

to show the grayscale image. Hence the image color varies smoothly from black through shades of red, orange, and yellow, to white, as the intensity value is increasing.

In order to overcome the oversmoothing by the use of Gaussian filters, an anisotropic filter was introduced to the image denoising community by Perona and Malik in [90, 91]. The new anisotropic filter convolves the image only in the smooth area, i.e. the area where the norm of the gradient is small. The nonlinear PDE used to describe the process in the continuous setting is

$$\frac{\partial X}{\partial t}(t;s,y) = \operatorname{div}(c(|\nabla X|^2)\nabla X)(t;s,y) \quad \text{in} \quad (0,T)\times\Omega, \tag{1.3}$$

with noisy image $\tilde{X}(s,y)$ as the initial condition and Neumann boundary conditions. In the above equation, ∇ denotes the gradient operator. The diffusivity function, $c(r^2)$, controls the diffusion direction. In [90], Perona and Malik propose the diffusivity function

$$c(r^2) = \frac{1}{1+r^2/\Gamma}, \quad \Gamma > 0, \tag{1.4}$$

where threshold Γ is determined by the noise level in $\tilde{X}(s,y)$. It is clear that $c(|\nabla X|^2)$ is a decreasing function of the norm of gradient $|\nabla X|$, and that

$$c(r^2) \approx \begin{cases} 1 & \text{if } r \ll 1 \\ 0 & \text{if } r \gg 1 \end{cases}.$$

Therefore, (1.3) restricts the diffusion to within the smooth areas (i.e. the areas where the norm of gradient is relatively small), and limits the amount of cross-edge diffusion in the restored image. The restored image contains sharper edges than that computed by linear Gaussian filter, see Figure 1.1d. While the Perona-Malik (PM) model performs well in practice, it is mathematically ill-posed. This is the so-called Perona-Malik paradox [68]. In Chapter 3, we solve a regularized anisotropic diffusion equation with efficient numerical methods.

1.1.2 Image Deblurring

Image deblurring is a more complicated problem in which seriously attenuated features need to be restored, usually in the presence of small amounts of noise. The continuous image deblurring problem can be modeled by a Fredholm integral equation of first kind [40, 95],

$$\iint \mathcal{K}(s,y;\nu,\tau)X(\nu,\tau)d\nu d\tau = B(s,y) \quad \text{in} \quad \Omega, \tag{1.5}$$

where the kernel function $\mathcal{K}$ and right-hand side B are known, and X is the unknown function. This equation establishes a linear relationship between X and B determined by $\mathcal{K}$. $\mathcal{K}$ is called the point spread function (PSF) of the imaging system (see the Gaussian PSF example in Figure 1.2).

An important special case of (1.5) is when the kernel is spatially invariant, $\mathcal{K}(s,y;\nu,\tau) = \mathcal{K}(s-\nu, y-\tau)$ [40, 54]. Further, the images are often degraded by noise, $\mathcal{E}$. This model is given by

$$\iint \mathcal{K}(s-\nu,y-\tau)X(\nu,\tau)d\nu d\tau \approx \tilde{B}(s,y) = B(s,y) + \mathcal{E}(s,y) \quad \text{in} \quad \Omega, \tag{1.6}$$

where $B(s,y)$ now denotes the true blurred image without any noise.

The discretized version of (1.6) can be written in matrix-vector form as

$$Ax \approx \tilde{b} = b + \eta, \tag{1.7}$$

where square matrix $A \in \mathbb{R}^{n\times n}$, ($n = m^2$ for an $m \times m$ image), is determined by the PSF and corresponding boundary conditions, $\tilde{b} = \text{vec}(\tilde{X}) \in \mathbb{R}^n$ is the observed blurred and noisy image, $x = \text{vec}(X) \in \mathbb{R}^n$ represents the unknown true image, $\eta = \text{vec}(\mathcal{E}) \in \mathbb{R}^n$ denotes the unknown iid noise.

Given a 256×256 image which is relatively small compared to the images taken by digital cameras nowadays, the size of A is $65,536 \times 65,536$ which is large. Depending on the PSF, the blurring matrix A is typically ill-conditioned [54, 58]. Therefore, solving (1.7) in the present of noise is not useful, as we discuss below. We note that in the case A is not full rank, one may replace (1.7) by computing x as

$$\underset{x}{\text{argmin}} \; ||Ax - \tilde{b}||_2^2. \tag{1.8}$$

However, in the presence of noise, this solution will also be worthless. The argument for the rank deficient case is nearly identical to the full rank discussion following in 1.1.2.1.

1.1.2.1 Singular Value Decompsition

Assume $A \in \mathbb{R}^{n \times n}$ has full rank, which is satisfied by the numerical experiments in this work, the singular value decomposition (SVD) of matrix A is formulated as

$$A = USV^T = \sum_{i=1}^{n} s_i u_i v_i^T,$$

with unitary matrices $U = [u_1, u_2, \cdots, u_n] \in \mathbb{R}^{n \times n}$ and $V = [v_1, v_2, \cdots, v_n] \in \mathbb{R}^{n \times n}$, diagonal matrix $S = \text{diag}[s_1, s_2, \cdots, s_n]$, where the s_i are real-valued with $s_1 \geq s_2 \geq \cdots \geq s_n > 0$. In this case, there exists a unique analytical solution of (1.8):

$$x = \sum_{i=1}^{n} \frac{u_i^T \tilde{b}}{s_i} v_i = \underbrace{\sum_{i=1}^{n} \frac{u_i^T b}{s_i} v_i}_{\text{inverted signal}} + \underbrace{\sum_{i=1}^{n} \frac{u_i^T \eta}{s_i} v_i}_{\text{inverted noise}} . \tag{1.9}$$

Unfortunately, the above "naive" SVD solution (1.9) is deteriorated by presence of the inverted noise components, see the "naive" restored image in Figure 1.4b. Here are the underlying reasons [58].

- $|u_i^T \eta|$ is small but nearly constant for white Gaussian noise, while both spectral coefficients, $|u_i^T b|$, and singular values, s_i, decrease as i increases, see Figure 1.3a;
- $|u_i^T b| \gg |u_i^T \eta|$ for small i, $\left|\frac{u_i^T \tilde{b}}{s_i}\right| \approx \left|\frac{u_i^T b}{s_i}\right|$;
- $|u_i^T b| \ll |u_i^T \eta|$ for large i, $\left|\frac{u_i^T \tilde{b}}{s_i}\right| \approx \left|\frac{u_i^T \eta}{s_i}\right|$;
- singular vectors, v_i, corresponding to large s_i represent low frequency information, while v_i corresponding to small s_i represent high-frequency information [58].

Therefore, (1.9) can be rewritten as

$$\begin{aligned} x &= \sum_{i=1}^{k} \frac{u_i^T \tilde{b}}{s_i} v_i + \sum_{i=k+1}^{n} \frac{u_i^T \tilde{b}}{s_i} v_i \\ &\approx \sum_{i=1}^{k} \frac{u_i^T b}{s_i} v_i + \sum_{i=k+1}^{n} \frac{u_i^T \eta}{s_i} v_i, \end{aligned} \tag{1.10}$$

where k is chosen such that $|u_k^T b| \approx |u_k^T \eta|$. The "naive" SVD solution (1.9) is mainly deteriorated by $\sum_{i=k+1}^{n} \frac{u_i^T \eta}{s_i} v_i$ living in the high-frequency space $\mathcal{S}_k^{\perp} = \text{span}\{v_{k+1}, v_{k+2}, \cdots, v_n\}$. This SVD analysis motivates the use of projection space regularization methods. The idea is to keep the solution in the low-frequency space $\mathcal{S}_k = \text{span}\{v_1, v_2, \cdots, v_k\}$ and damp the solution in the high-frequency space $\mathcal{S}_k^{\perp}$.

1.1.2.2 Projection Space Regularization

Motivated by the above SVD analysis, projection space regularization methods can be achieved by introducing a spectral filter factor, ϕ_i, in (1.9). The general form of the filtered solution is given by

$$x = \sum_{i=1}^{n} \phi_i \frac{u_i^T \tilde{b}}{s_i} v_i. \tag{1.11}$$

Some classical examples of regularization methods whose solutions can be written in this form include:

- **Truncated SVD** (TSVD) [51]:

  $$\phi_i \equiv \begin{cases} 1, & i = 1, 2, \cdots, k \\ 0, & i = k+1, k+2, \cdots, n \end{cases}$$

  with k as the regularization parameter. This method simply throws away the information in the high-frequency space $\mathcal{S}_k^{\perp}$, see the restored image in Figure 1.4c and Equation (1.10).

- **Tikhonov regularization** [54, 58]:

  $$\phi_i \equiv \frac{s_i^2}{s_i^2 + \lambda}, \qquad \lambda \in [s_n, s_1].$$

  with λ as the regularization parameter. Employing this spectral filter is equivalent to solving the minimization problem

  $$\underset{x}{\operatorname{argmin}} \; ||Ax - \tilde{b}||_2^2 + \lambda ||x||_2^2. \tag{1.12}$$

  The regularization parameter, λ, controls the tradeoff between two terms in (1.10) and (1.12) and determines the desired smoothness of the solution. It is clear that

  $$\phi_i \approx \begin{cases} 1, & \text{if } s_i \gg \lambda \\ 0, & \text{if } s_i \ll \lambda \end{cases}$$

  Therefore, Tikhonov regularization has an effect similar to TSVD (see the restored image Figure 1.4d).

From Figure 1.4c and 1.4d, we observe both TSVD and Tikhonov regularization reduce the blur. The main drawback of these simple projection space methods is that they only use the low-dimensional smoothing subspace $\mathcal{S}_k$ to approximate the solution, which tends to inhibit the reconstruction of the sharp edges. This is because edges contain high-frequency spectral components, whereas both the TSVD and Tikhonov subspaces $\mathcal{S}_k$ only contain the low-frequency modes. If one simply increases the dimension, k, of $\mathcal{S}_k$ or decreases λ in Tikhonov regularization (1.12) to include higher frequency components, then the solutions will be deteriorated by the inverted noise term.

1.2 Efficient Numerical Algorithms

While an accurate mathematical model is the foundation of image restoration, developing an efficient numerical method is a critical step in practice. In this section, we briefly go over two numerical algorithms used in this work.

1.2.1 Multigrid Methods

Originally introduced as a way to solve elliptic boundary value problems, multigrid methods have been developed and applied to various problems [1, 3, 22, 32, 71, 107, 108, 120]. The optimality of multigrid methods suggests that they are potential good solvers or preconditioners for systems with elliptic operators, which often appear in image restoration problems.

By studying the effects of some simple relaxation schemes, such as the Jacobi and Gauss-Seidel methods, researchers found these algorithms possess the so-called “smoothing property”, i.e. they effectively eliminate oscillatory modes of the error and leave smooth modes of the error undamped [22]. However, these smooth modes become more oscillatory when represented on coarser grids. Based on these facts, a multigrid scheme uses relaxation on the fine grid and, then, adds an interpolated coarse-grid correction which is computed by relaxation on the coarse-grid residual equation. The process is called multigrid cycling [22, 107, 120].

Depending on the application, multigrid methods can be very different. In general, however, multigrid methods involve two steps:

setup phase:

- choosing an appropriate relaxation algorithm;
- choosing suitable coarse-grid variables;
- choosing appropriate grid-transfer operators, i.e. restriction and interpolation operators;
- designing effective coarse-grid equations.

solution phase: uses the above components to perform normal multigrid cycling until a desired stopping criterion is satised.

According to whether the problem associates with any kind of structured physical grids, multigrid methods fall into two categories: geometric multigrid (GMG) methods [22, 107] and algebraic multigrid (AMG) methods [21, 98]. GMG methods use geometrically structured coarse grids within the multigrid process. Traditional GMG methods use no matrix information in defining the grid-transfer operators and, therefore, are not robust to the discontinuous diffusivities considered in (1.3). However, the black box multigrid (BoxMG) algorithm [3, 64, 65], which uses geometrically structured coarse grids in combination with an interpolation operator designed to account for the effects of discontinuous diffusivities, achieves fast multigrid convergence in many situations.

In contrast to GMG methods that work with structured grids, AMG methods make no assumptions other than the given matrix and a right-hand side. They achieve efficiency by tailoring both the coarse-grid structure and interpolation operator to account for jumps in the coefficients in (1.3). This makes AMG methods more robust, and allows them to be applied in a wide variety of applications. However, the cost for this robustness is that they have a more expensive setup process due to the use of unstructured matrix storage approaches.

1.2.2 Nonnegative Constrained Least Squares

In some applications [5, 13, 14, 15, 16, 38, 72, 73, 123], it makes sense to solve the least squares problem (1.8) with nonnegative constraints, referred to as nonnegative least squares (NNLS),

$$\underset{x}{\operatorname{argmin}}\ F_1(x) = \underset{x}{\operatorname{argmin}}\ ||Ax - b||_2^2 \quad \text{s.t} \quad x \geq 0. \tag{1.13}$$

Because

$$\begin{aligned} ||Ax - b||_2^2 &= (Ax-b)^T(Ax-b) \\ &= (Ax)^T(Ax) - (Ax)^T b - b^T(Ax) + b^T b \\ &= x^T(A^TA)x - 2x^T(A^Tb) + b^T b, \end{aligned}$$

define the symmetric positive semi-definite matrix $Q = A^TA$, and vector $h = A^Tb$, then, the NNLS (1.13) is equivalent to the following nonnegative quadratic program (NNQP),

$$\underset{x}{\operatorname{argmin}}\ F(x) = \underset{x}{\operatorname{argmin}}\ \frac{1}{2}x^TQx - x^Th \quad \text{s.t} \quad x \geq 0. \tag{1.14}$$

Since first proposed by Lawson and Hanson [72], there has been much work in the literature on solving (1.13), based on techniques such as active set methods [13, 23, 72], interior point methods [12], iterative approaches [70] etc., (see [33] for a complete review). In this work, we present a new multiplicative algorithm. The new algorithm has a very simple format and can be easily implemented on a parallel machine.

1.3 Contributions

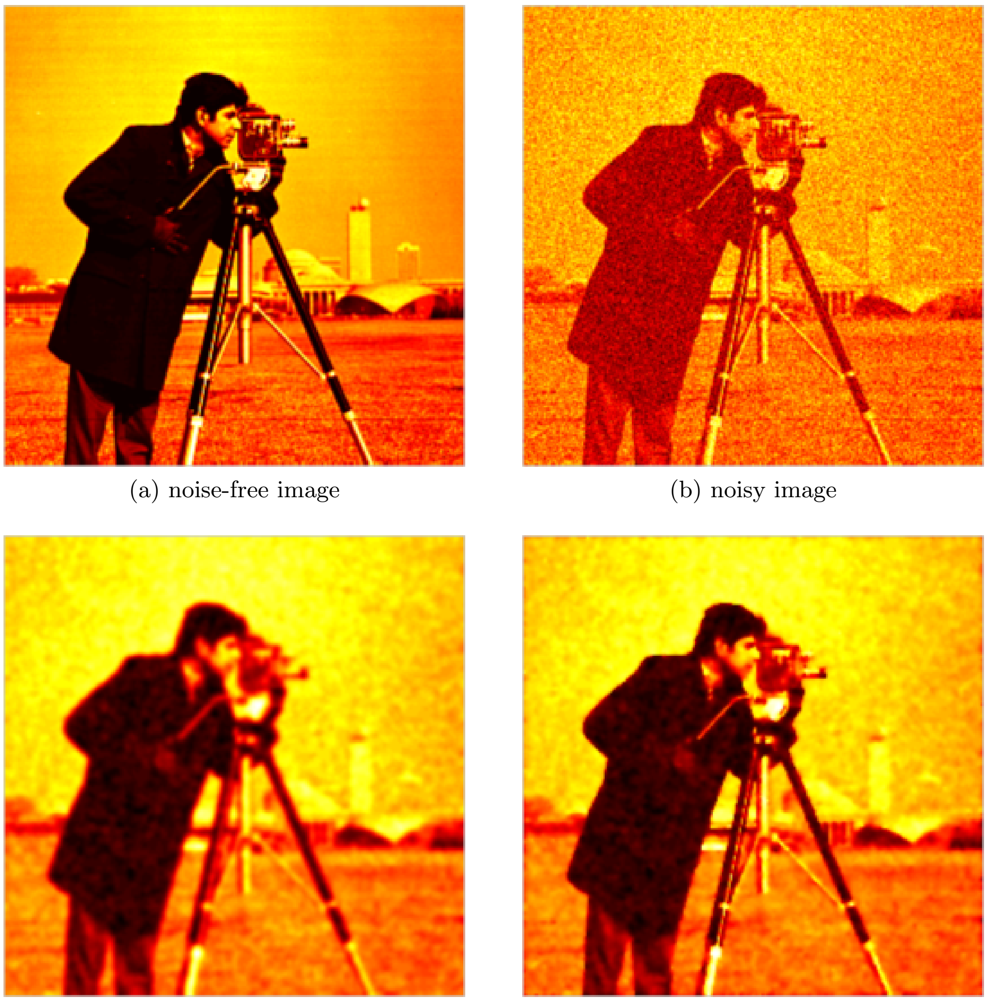
The significant contributions of this work include developing an iterative regularization parameter-choice algorithm, exploring multigrid solvers for anisotropic diffusion denoising, deriving a new mathematical framework for a projection-based edge-preserving deblurring method, and designing a new NNLS algorithm.

- **Regularization parameter-choice method**:
  - A new regularization parameter-choice method based on the normalized cumulative periodogram is proposed. The new algorithm can be applied to a very broad range of parameter-dependent image restoration algorithms.
  - An efficient Matlab implementation with Brent's method is provided, see Appendix A.
- **Multigrid methods for anisotropic diffusion denoising**:
  - A new fixed-point iteration method with multigrid solvers is designed for anisotropic diffusion denoising.
  - A comparison of the performance of BoxMG and AMG in solving anisotropic diffusion equations is provided.
- **Projection-based edge-preserving image deblurring**:
  - A new mathematical framework is developed for projection-based edge-preserving image deblurring. A theoretical justification of the approach is given. A detailed numerical study of the performance, with suggestions for parameter values and tolerances, is given.
  - An efficient Matlab implementation, with Kronecker product SVDs and an AMG preconditioner, is developed; the numerical results presented in this work are obtained from this code, which is available upon request.

- **Nonnegative least squares algorithm**
  - A new multiplicative algorithm is designed for NNLS (1.13). The theoretical convergence analysis is presented.
  - Two image processing applications, super-resolution restoration and color image labeling, with efficient numerical implementations are explored.

1.4 Outline of the Work

The structure of the dissertation is as follows. Chapter 2 illustrates a new regularization parameter-choice method which is based on normalized cumulative periodogram and Brent's method. Chapter 3 introduces the mathematical background of anisotropic diffusion equation and its application in image denoising. A fixed-point iteration with multigrid solvers is proposed. We also present the comparisons with other state-of-arts image denoising techniques. Chapter 4 presents a novel projection-based edge-preserving image deblurring algorithm. Extensive performance testing that gives insight into robust choices for tolerances and parameters is presented. Chapter 5 presents a new multiplicative NNLS algorithm along with its convergence analysis. Efficient implementations for image super-resolution and color image labeling are presented as examples. Final conclusions are given in Chapter 6.


(a) noise-free image

(b) noisy image

(c) restored by Gaussian filter

(d) restored by PM model (1.3)

Figure 1.1: Image denoising example with cameraman image. (a) true noise-free image. (b) noisy image, noise level is 20%. (c) linear Gaussian filter with $t = 2.56$. (d) Perona-Malik anisotropic diffusion (1.3) with stopping time $t = 0.75$ and threshold $\Gamma = 0.2$ in (1.4).

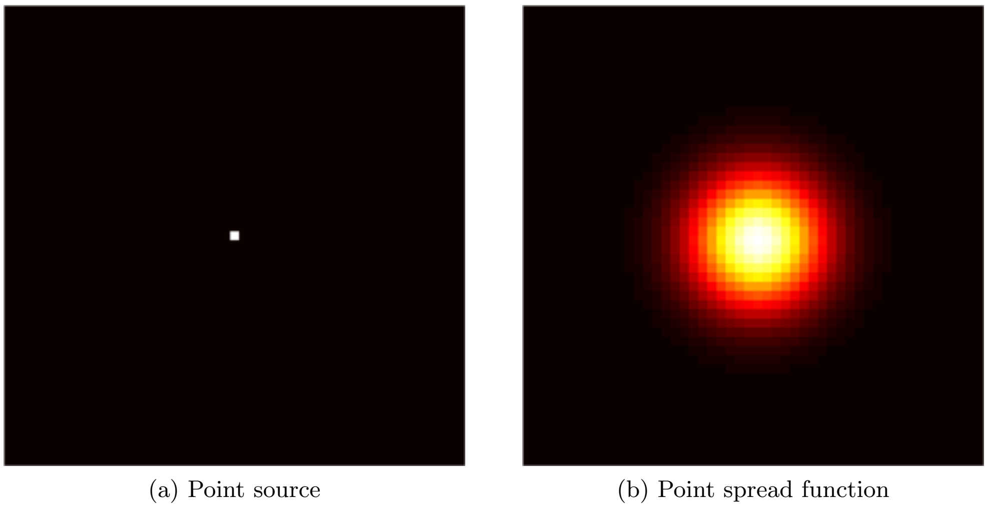
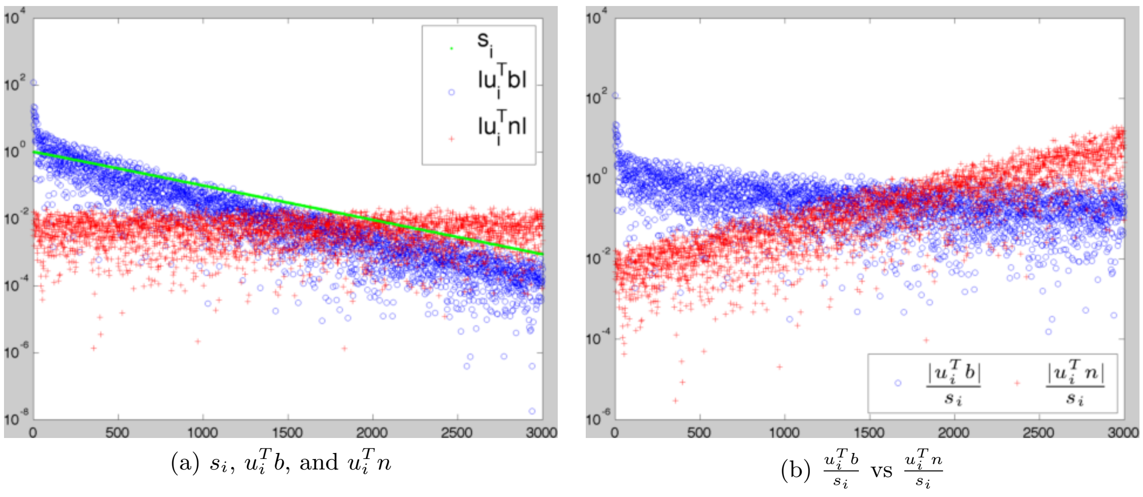
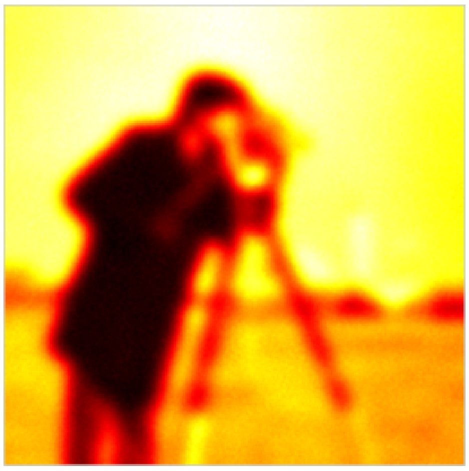

(a) Point source (b) Point spread function

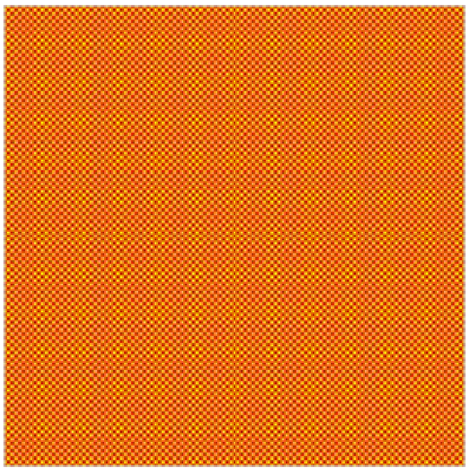
Figure 1.2: Point spread function. (a) single bright pixel. (a) the blurred point source, called the point spread function.

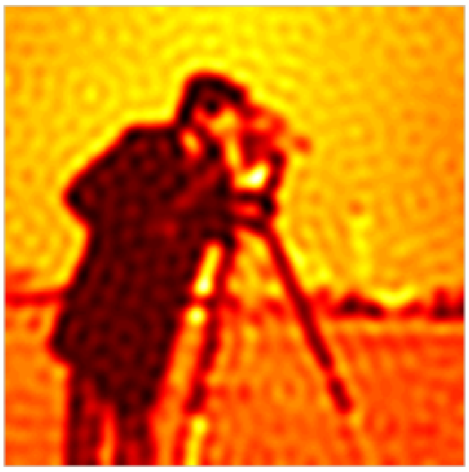


(a) s_i, $u_i^T b$, and $u_i^T n$ (b) $\frac{u_i^T b}{s_i}$ vs $\frac{u_i^T n}{s_i}$

Figure 1.3: Plots of first 3000 singular values, s_i, coefficients, $|u_i^T b|$ and $\frac{|u_i^T b|}{s_i}$, and error coefficients, $|u_i^T \eta|$ and $\frac{|u_i^T \eta|}{s_i}$. (a) plots of decreasing s_i, $u_i^T b$, and nearly constant $u_i^T \eta$. (b) plots of decreasing $\frac{u_i^T b}{s_i}$ and increasing $\frac{u_i^T \eta}{s_i}$.


(a) blurring image


(b) naive restoration


(c) TSVD restoration, $k = 1699$

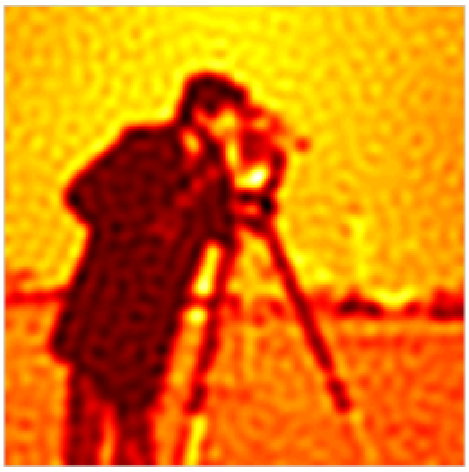

(d) Tikhonov restoration, $\lambda = 0.016$

Figure 1.4: Restored images, the regularization parameters in TSVD and Tikhonov methods are chosen with GCV method. (a) blurred and noisy image. (b) naive restored image. (c) restored image by TSVD method. (d) restored image by Tikhonov regularization method.

Chapter 2

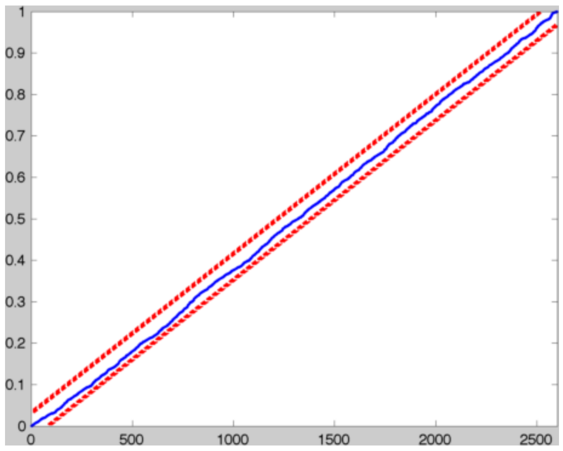
Regularization Parameter-Choice Method

2.1 Introduction

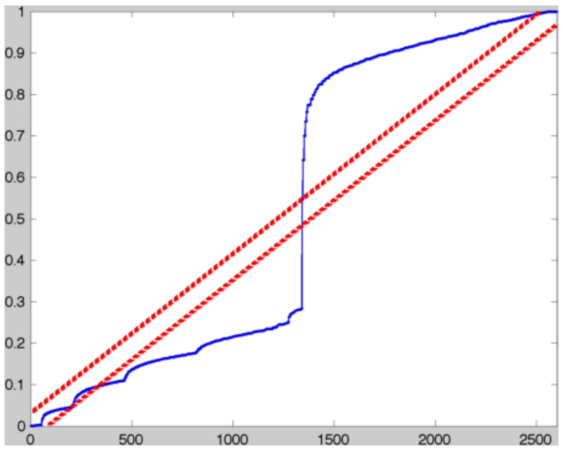
As discussed in Chapter 1, image restoration problems are typically ill-posed, and we have to use regularization techniques, such as TSVD and Tihkonov regularization, to obtain a reasonable approximate solution. In the literature, there exist many different regularization methods [11, 30, 51, 88, 97]. However, no matter which method is used, choosing a suitable regularization parameter is a critical step. Some common parameter-choice methods used in the image restoration community include the discrepancy principle [52, 54, 58], L-curve methods [53, 59], and the generalized cross-validation (GCV) method [48, 89].

More recently, a method based on the normalized cumulative periodogram (NCP) has been proposed [32, 56, 99]. Compared to traditional methods, such as L-curve approaches, which only use the norm of the residual vector, the NCP method seeks to use more information available in the residual vector. The key idea of this method is to choose the regularization parameter for which the residual vector changes from being dominated by the remaining signal to being like white noise. By employing statistical tools, such as the Komolgorov-Smirnov (KS) test [105], and fast Fourier transforms, this method leads to a parameter-choice rule which is particularly well-suited for large-scale problems. See [99] and the references therein for more details of the NCP regularization parameter-choice method. In this section, we illustrate a new parameter-choice method based on NCP and Brent's method [94].

The remainder of this chapter is organized as follows. Section 2.2 introduces NCP and its



(a) Normal distributed random numbers



(b) Sum of Cosines

Figure 2.1: Examples of normalized cumulative periodogram for different vectors, x. Left: vector x contains elements which are Matlab generated normal distributed random numbers; Right: vector x contains elements which are sum of cosine functions with different periods. Blue line is the NCP plots. Two red dot-lines are the 95% Kolmogorov-Smirnov test confidence limits. The NCP of the normal random numbers stays in the limits (left).

application in image processing. An efficient algorithm based on NCP and Brent's method is presented in Section 2.3.

2.2 Normalized Cumulative Periodogram

The periodogram is essentially a discrete Fourier transform of the input data.

Definition 2.1. *Given 1D signal $x \in \mathbb{R}^{n\times 1}$, denote $\lfloor \frac{n}{2} \rfloor$ as the maximum integer less than $\frac{n}{2}$, the periodogram [18] is defined as the squared-magnitude of the discrete Fourier transform (DFT)*

$$p(w_j) = |DFT(x)_j|^2 = \frac{1}{n}\left|\sum_{i=1}^{n} x(i)e^{-2\pi \imath w_j i}\right|^2,$$

where $w_j = \frac{j-1}{n}$ for any $j = 1, 2, \cdots, q = \lfloor \frac{n}{2} \rfloor + 1$. Further, $p(w_j) = p(w_{n-j+1})$ for $j = q+1, \cdots, n$.

The idea of the periodogram is to split the input data, x, into low- and high-frequency components. If the input data $x(t)$ appears to be very smooth (wiggly), then the values of the periodogram for low (high) frequencies will be large relative to other values. We say that the data set has an excess of low (high) frequency. For a purely random series, all of the frequencies should be of equal importance and thus the periodogram will vary randomly around a constant.

A useful tool for describing the overall behavior of the periodogram is the normalized cumulative periodogram

$$(\text{ncp}(x))_k = \frac{\sum_{j=1}^{k} p(w_j)}{\sum_{j=1}^{q} p(w_j)}, \quad k = 1, 2, \cdots, q = \left\lfloor \frac{n}{2} \right\rfloor + 1 \tag{2.1}$$

Note that if x is white Gaussian noise, the plot of ncp(x) versus w should follow along a straight line from $(0, 0)$ to $(0.5, 1)$. A test of the hypothesis that a signal is white noise can be achieved by the Kolmogorov-Smirnov (KS) test [105]. Thus, we choose the largest regularization parameter λ such that the residual vector looks like white noise. See Figure 2.1 for the example NCP plots of different data sets. As shown in Figure 2.1a, the NCP plot of the normal random data stays in the 95% confidence band. While the data used in Figure 2.1b are sum of several cosine functions. The NCP doesn't stay in the confidence band.

For a 2D image represented by the $m \times m$ matrix, X, the NCP is defined in a similar manner [56]. Let the $q \times q$ matrix $P_{i,j} = |\text{DFT}(X)_{i,j}|^2$ be the power spectrum of X where $q = \left\lfloor \frac{m}{2} \right\rfloor + 1$. The elements of P need to be reordered in order of increasing spatial frequency, see the details in [56] and Appendix A. After reordering, the vector

$$\hat{p} = \Pi \text{vec}(P), \quad \text{where } \Pi \text{ is a permutation matrix,}$$

holds all the power spectrum elements in order of increasing spatial frequency. Then, the NCP of X, which is a vector of length $q^2 - 1$, is defined as in Equation (2.1)

$$(\text{ncp}(X))_k = \frac{\sum_{j=1}^{k} \hat{p}(w_j)}{\sum_{j=1}^{q^2-1} \hat{p}(w_j)}, \quad k = 1, 2, \cdots, q^2 - 1.$$

In practice, in order to find a near optimal regularization parameter, Hansen *et al.* suggest to solve the following minimization problem to select the regularization parameter [56]

$$\underset{\lambda}{\text{argmin}} \;\; \mathcal{N}(\lambda) := ||v - \text{ncp}(r_\lambda)||_1, \tag{2.2}$$

where r_λ is the computed residual vector using regularization parameter λ, i.e.

$$r_\lambda = \text{vec}(\tilde{X} - X_{\text{restored}}),$$

and v is the straight line connecting $(0, 0)$ and $(0.5, 1)$.

Algorithm 1 Brent-NCP Algorithm

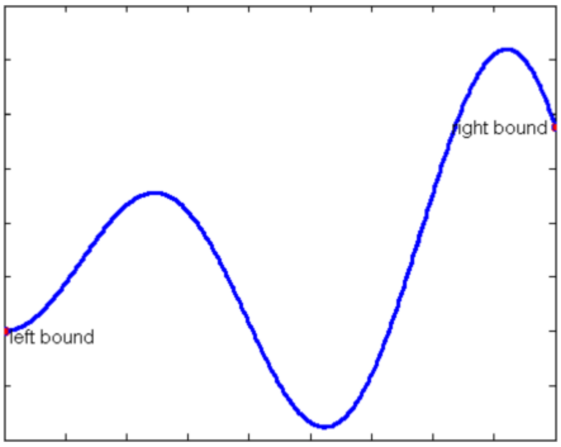
```
1: Set $a < c$, $b \leftarrow a + \frac{3-\sqrt{5}}{2} \times (c-a)$
2: $\lambda \leftarrow (a+c)/2$
3: Compute the residual $r_\lambda := x^0 - x(\lambda)$, where $x(\lambda)$ is the computed solution using
   regularization parameter $\lambda$
4: Compute $\mathcal{N}(\lambda) = ||v - \text{ncp}(r_\lambda)||_1$
5: while $|\lambda - b| > \text{tol}$ do
6:     Construct a trial parabolic fit
7:     if parabolic fit is acceptable then
8:         Take the parabolic step
9:     else
10:        Take a golden section step
11:    end if
12:    Update the values $a, b, c, \lambda$, compute $\mathcal{N}(\lambda) = ||v - \text{ncp}(r_\lambda)||_1$.
13: end while
14: return $\lambda$
```

2.3 Brent-NCP Algorithm

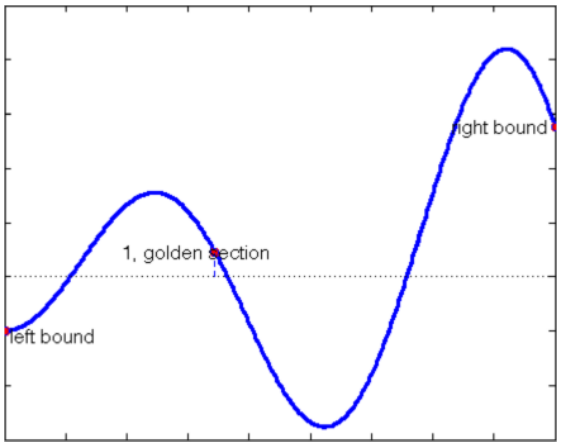
Efficiently solving the minimization problem (2.2) is essential to the success of the parameter-choice algorithm. Because $\mathcal{N}(\lambda)$ is not differentiable, solving (2.2) requires solving the regularization problem for different regularization parameters, λ, which is a time consuming process. Since Brent's method is characterized by quadratic convergence in case of smooth functions and guaranteed linear convergence in case of nonsmooth or oscillatory functions, we propose Algorithm 1 to minimize the number of search steps for (2.2).

Brent's method combines the golden section search method with a parabolic interpolation method to minimize $\mathcal{N}(\lambda)$ [6, 94]. Starting with 2 boundary points, a and b, the third point, c, is computed with a golden section search step. At each iteration, Brent's method approximates $\mathcal{N}(\lambda)$ using an interpolating parabola through the existing 3 points. The minimum of the parabola is taken as a guess for the minimum of $\mathcal{N}(\lambda)$. If it lies within the bounds of the current interval, then the interpolating point is accepted, and is used to generate a smaller interval. If the interpolating point is not accepted, then the algorithm falls back to a golden section step. The details of Brent's method, including some additional checks to improve convergence, can be found in [6, 94]. For convenience, we summarize Brent's method applied to (2.2) as Algorithm 1. The Malab code is provided in Appendix A.

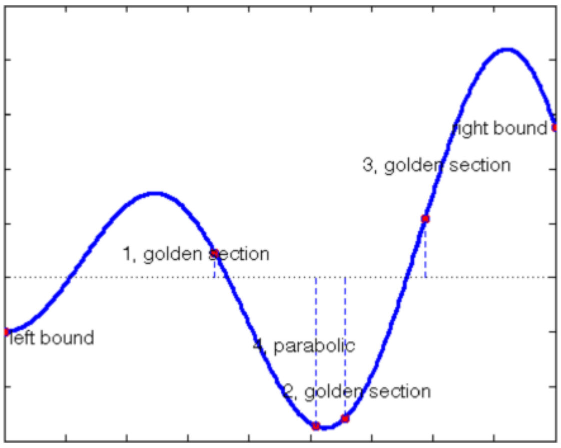
Figure 2.2 shows an example illustrating the convergence of Brent's method. Starting with left and right bounds in Figure 2.2a, the Brent's method gets the third points with



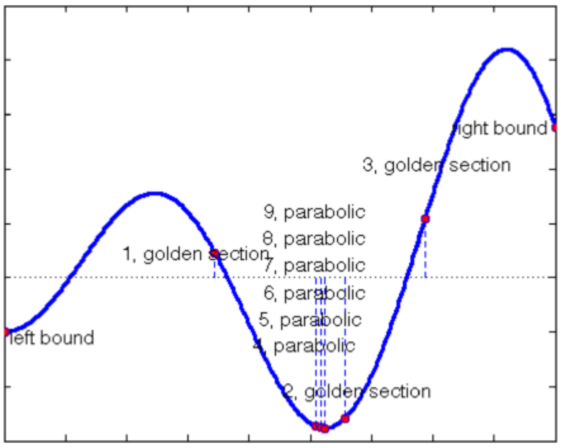
(a) Initial two input bounds



(b) Step 1: golden section search



(c) Step 4: parabolic interpolation



(d) Converges at 9th iteration

Figure 2.2: Example illustrating the convergence of Brent's method. The actual method (golden section search or parabolic interpolation) used in Brent's method at each iteration is shown in 2.2d. In this example, the algorithm converges in 9 steps.

golden section step, see Figure 2.2b. Then it constructs a parabolic fit with these three points. In the next step, it checks whether the fit is acceptable. In this example, the parabolic fit is not acceptable. It takes another golden section step with the updated two bounds. The algorithm stops at the 9th iteration when the interval become smaller than a tolerance 2.2d.

In practice, the above Brent-NCP algorithm converges very quickly for all the regularization methods considered in Chapter 3. Depending on the method used in the evaluation of $\mathcal{N}(\lambda)$ in Steps 4 and 12, Algorithm 1 has many potential applications, see the numerical experiments in Chapter 3.

Chapter 3

Multigrid Anisotropic Diffusion Denoising

3.1 Introduction

As discussed in Section 1.1.1, the Perona-Malik model,

$$\frac{\partial X}{\partial t}(t;s,y) = \operatorname{div}(c(|\nabla X|^2)\nabla X)(t;s,y) \quad \text{in} \quad (0,T)\times\Omega, \tag{3.1}$$

with the noisy image, $\tilde{X}$, as the initial value and Neumann boundary conditions, performs well in image denoising in practice. Theoretically, however, it is an ill-posed problem. This is the so-called Perona-Malik paradox [68]. The underlying reason for this paradox is that the forward-backward diffusion equation (3.1) is not well-posed. Defining the flux function $\Phi(r) = rc(r^2)$, the forward-backward diffusion process depends on the sign of first derivative of $\Phi(r)$,

$$\Phi'(r) = c(r^2) + 2r^2c'(r^2),$$

- if $\Phi'(r) > 0$, the PM model is a forward parabolic equation, and all edges are blurred;
- if $\Phi'(r) < 0$, the PM model is a backward parabolic equation, and the edges are sharpened.

In [90], Perona and Malik choose the diffusivity to be $c(r^2) = \frac{1}{1+r^2/\Gamma^2}$, where Γ is a threshold determined by the noise level [27, 91]. Given a threshold, Γ, the PM approach shows the desirable result of removing noise in an image in regions where $|\nabla X(t;s,y)| < \Gamma$, and sharpening edges in an image in regions where $|\nabla X(t;s,y)| \geq \Gamma$.

In [68], Kichenassamy proves that if the initial image, $\tilde{X}(s,y)$, is not infinitely differentiable, the weak solution of (1.3) does not exist. Consequently, he introduces the notion of a "generalized solution", which is piecewise linear and contains jumps. However, one should neither expect uniqueness nor stability with respect to the initial image. Examples of significantly differing solutions with nearly identical initial data have been reported [68].

The remainder of this chapter is organized as follows. Section 3.2 goes over the anisotropic diffusion regularization methods and introduces our objective equation. A semi-implicit discretization technique is discussed in Section 3.3. Section 3.4 discusses several related regularization parameter-choice algorithms. Section 3.5 presents numerical experiments of the new approach and comparisons with other image denoising methods.

3.2 Anisotropic Diffusion with Regularization

Although the ill-posedness of the PM model can be handled by applying an implicit spatial discretization [113], in order to make the numerical implementation more predictable, it is more natural to introduce regularization into the continuous PM model (3.1).

3.2.1 Spatial Regularization

Catté *et al.* introduce a spatial regularization that makes the forward-backward diffusion process (3.1) become well-posed [29]. The idea is to replace the image gradient, ∇X by a smoothed version, $G_\sigma * \nabla X$, in the diffusivity coefficient $c(|\nabla X|^2)$, where G_σ can be any "low-pass filter". In this work, we assume that G_σ is a Gaussian kernel with standard deviation $\sigma = 0.5$. Since $G_\sigma * \nabla X = \nabla(G_\sigma * X)$, the spatially regularized PM model becomes

$$\frac{\partial X}{\partial t}(t;s,y) = \operatorname{div}(c(|\nabla(G_\sigma * X)|^2)\nabla X)(t;s,y) \quad \text{in} \quad (0,T)\times\Omega. \tag{3.2}$$

Catté *et al.* prove that there exists a unique solution for the regularized PM equation (3.2) with corresponding initial and boundary conditions [29]. Furthermore, this spatial regularization makes the filter insensitive to noise. This avoids the shortcoming of the original PM model, which cannot distinguish between "true" edges and "false" edges created by the noise. Weickert *et al.* [116] propose an additive operator splitting (AOS) scheme to solve (3.2). The diffusion stopping time, T, is the parameter controlling the restored image quality.

3.2.2 Reaction Diffusion Equation

While this spatial regularization makes the PM model well-posed, it leads to a process where the solution always converges to a constant steady-state solution [113]. In order to get a nontrivial result, it is then required to specify a stopping time, T_0, such that the restored image, $X(T_0)$, is a good representation of the denoised image. Sometimes, it is attempted to circumvent this task by adding an additional reaction term [84],

$$\frac{\partial X}{\partial t}(t;s,y) = \operatorname{div}(c(|\nabla X|^2)\nabla X)(t;s,y) + \lambda(\tilde{X}(s,y) - X(t;s,y)) \quad \text{in} \quad (0,T)\times\Omega. \tag{3.3}$$

The reaction term, $(\tilde{X}(s,y) - X(t;s,y))$, keeps the steady-state solution close to the original image, $\tilde{X}(s,y)$. In practice, such a modification shifts the problem of specifying a stopping time, T_0, to the problem of determining the non-negative regularization parameter, λ.

Combining the spatial regularization (3.2) and reaction anisotropic diffusion (3.3) approaches, we get the anisotropic diffusion equation

$$\frac{\partial X}{\partial t}(t;s,y) = \operatorname{div}(c(|\nabla(G_\sigma * X)|^2)\nabla X)(t;s,y) + \lambda(\tilde{X}(s,y) - X(t;s,y)) \quad \text{in} \quad (0,T)\times\Omega, \tag{3.4}$$

where we assume that G_σ is fixed and known. As discussed above, this PDE is not only well-posed, but also has a non-trivial steady-state solution satisfying

$$0 = \operatorname{div}(c(|\nabla(G_\sigma * X)|^2)\nabla X)(s,y) + \lambda(\tilde{X}(s,y) - X(s,y)) \quad \text{in} \quad \Omega. \tag{3.5}$$

Thus, for a fixed regularization parameter λ, two approaches are possible: iterating the time step in (3.4) until the steady state is reached, or solving (3.5)) using a fixed point iterative approach. In either case, finding a near optimal regularization parameter is critical to obtaining a good restored image, and requires that either (3.4) or (3.5) be solved for different values of λ. We discuss methods for choosing such a value in Section 3.4.

In the case of solving (3.4) for a fixed λ, the issues to consider are the time step size and how accurately to solve the resulting linear system at each time step. Weickert's AOS scheme is one method that can also be employed to solve (3.4) for each fixed λ. Unlike using AOS to solve (3.2), the stopping time T is not considered a regularization parameter – rather, the stopping time is given by the time step at which the solution appears to have reached steady-state. The case of solving (3.5) by fixed point iteration for a given value of λ, in contrast to the AOS approach for (3.4), requires a more accurate linear solve at each iteration. The potential comparative upside, however, is that fewer overall iterations will needed, and unintentional smoothing that may occur as a result of inaccurate solves during timestepping to solve (3.4) are avoided. In this work, we consider two solvers for the linear systems that must be solved accurately with the fixed point approach: BoxMG and AMG. Numerical results that compare AOS for solving (3.4) vs. fixed point iteration with BoxMG to solve (3.5) vs. fixed point iteration with AMG to solve (3.5) are presented in Section 3.5. In all three cases, in order to do a fair comparison, the same regularization parameter selection scheme is employed.

3.3 Discretization and Multigrid Solvers

In [115], the authors compare three discretization schemes: both explicit and semi-implicit schemes based on a 3×3 stencil, and an explicit scheme based on a 5×5 stencil. The conclusion is that the 5×5 stencil explicit discretization scheme is superior than the explicit scheme based on 3×3 stencils in terms of rotation invariance, accuracy, and avoidance of blurring artifacts. However, results for the 3×3 AOS-stabilized semi-implicit approach are comparable to those for the 5×5 explicit stencil. Therefore, we use a semi-implicit discretization technique to discretize (3.5), which retains the memory advantage of using a 3×3 stencil. Instead of using a simple central difference scheme as in [116], we use more points in computing $c(|\nabla(G_\sigma * X)|^2)$ to increase the stability [7].

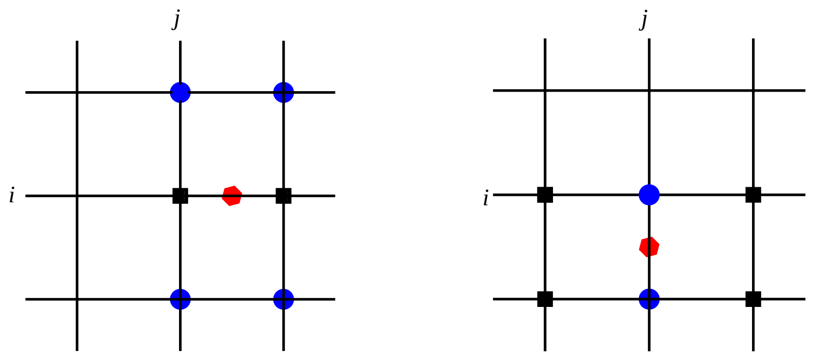
Assuming the regularization parameter, λ, is known, the following fixed-point iteration is used to compute the solution of (3.5),

$$\lambda(X^{k+1} - \tilde{X})(s, y) = \operatorname{div}(c(|\nabla(G_\sigma * X^k)|^2)\nabla X^{k+1})(s, y), \tag{3.6}$$

where the superscript, k, denotes a numerical approximation taken at the kth iteration. We use central differences to approximate the derivatives of the image, X. In digital images, the distance between adjacent grid points, h, is constant. For simplicity, we omit the distance h in the following discretization formulas. The value of the divergence operator at grid point (i, j) can then be written as

$$\begin{aligned}
\operatorname{div}(c\nabla X)_{i,j} &= c_{i+\frac{1}{2},j}(X_{i+1,j} - X_{i,j}) - c_{i-\frac{1}{2},j}(X_{i,j} - X_{i-1,j}) \\
&\quad + c_{i,j+\frac{1}{2}}(X_{i,j+1} - X_{i,j}) - c_{i,j-\frac{1}{2}}(X_{i,j} - X_{i,j-1}) \\
&= c_{i+\frac{1}{2},j}X_{i+1,j} + c_{i-\frac{1}{2},j}X_{i-1,j} + c_{i,j+\frac{1}{2}}X_{i,j+1} + c_{i,j-\frac{1}{2}}X_{i,j-1} \\
&\quad - (c_{i+\frac{1}{2},j} + c_{i-\frac{1}{2},j} + c_{i,j+\frac{1}{2}} + c_{i,j-\frac{1}{2}})X_{i,j}
\end{aligned}$$

Notice that interpolation is needed to evaluate the diffusivity, $c = c(|\nabla\hat{X}|^2)$, at locations $(i \pm \frac{1}{2}, j)$ and $(i, j \pm \frac{1}{2})$. This can be done as follows, see also Figure 3.1. Denoting $\hat{X} = G_\sigma * X^n$, we use central differences and linear interpolation with a compact stencil to



(a) $(\nabla\hat{x})_{i,j+\frac{1}{2}}$ (b) $(\nabla\hat{x})_{i+\frac{1}{2},j}$

Figure 3.1: Grid points involved in the approximation of the diffusivities $c(|\nabla\hat{X}|^2)$ at grid points $(i, j+\frac{1}{2})$ and $(i+\frac{1}{2}, j)$ (marked by hexagons). The grid points represented by dots are used to compute derivatives in the vertical direction, while the grid points represented by squares are used to compute derivatives in the horizontal direction.

compute the diffusivity

$$c_{i,j+\frac{1}{2}} := c\left((\hat{X}_{i,j+1}-\hat{X}_{i,j})^2+\left(\frac{\hat{X}_{i+1,j+1}-\hat{X}_{i-1,j+1}+\hat{X}_{i+1,j}-\hat{X}_{i-1,j}}{4}\right)^2\right),$$

and

$$c_{i+\frac{1}{2},j} := c\left(\left(\frac{\hat{X}_{i+1,j+1}-\hat{X}_{i+1,j-1}+\hat{X}_{i,j+1}-\hat{X}_{i,j-1}}{4}\right)^2+(\hat{X}_{i+1,j}-\hat{X}_{i,j})^2\right).$$

Because of the Gaussian filter, G_σ, and the average used in computing the diffusivity, c, the above discretization is less sensitive to noise than without the filter. In practice, this also makes the discretization have a rotation-invariance property [7]. On the other hand, using a compact stencil provides a good balance between accuracy and computational time.

The discretization of (3.6) is, then,

$$a_{i,j}X_{i,j}^{k+1}-\left(c_{i+\frac{1}{2},j}X_{i+1,j}^{k+1}+c_{i-\frac{1}{2},j}X_{i-1,j}^{k+1}+c_{i,j+\frac{1}{2}}X_{i,j+1}^{k+1}+c_{i,j-\frac{1}{2}}X_{i,j-1}^{k+1}\right)=\lambda\tilde{X}_{i,j},$$

where $\tilde{X}=I_0$, $a_{i,j}=\left(\lambda+\left(c_{i+\frac{1}{2},j}+c_{i-\frac{1}{2},j}+c_{i,j+\frac{1}{2}}+c_{i,j-\frac{1}{2}}\right)\right)$. In matrix-vector notation, the above discrete form can be written as

$$A(x^k,\lambda)x^{k+1}=\lambda\tilde{x}. \tag{3.7}$$

For fixed λ, the linearized diffusion equation (3.7) represents a fixed-point linearization of the nonlinear PDE described by (3.5). This, naturally, leads to the fixed-point iteration given as Algorithm 2, whose convergence proof can be found in [4]. However, for the sake of

Algorithm 2 Fixed-Point Iteration for Fixed λ

```
1: while $\mathrm{norm}(x^{n+1} - x^n)/\mathrm{norm}(x^n) > \mathrm{tol}_{\mathrm{fp}}$ do
2:     Compute the matrix $A_n = A(x^n, \lambda)$ in (3.7) with approximation $x^n$
3:     Compute the solution $x^{n+1}$ of $A_n x^{n+1} = \lambda\tilde{x}$
4: end while
5: return $I^{n+1}$
```

completeness, we restate the theorem as follows. The proof is by constructing a contractive function, Q,

$$(Q(X^k))_{i,j} = \left(\frac{c_{i+\frac{1}{2},j}}{a_{i,j}}X^{k+1}_{i+1,j} + \frac{c_{i-\frac{1}{2},j}}{a_{i,j}}X^{k+1}_{i-1,j} + \frac{c_{i,j+\frac{1}{2}}}{a_{i,j}}X^{k+1}_{i,j+1} + \frac{c_{i,j-\frac{1}{2}}}{a_{i,j}}X^{k+1}_{i,j-1}\right) + \lambda\tilde{X}_{i,j}.$$

Note that if the coefficients of X^{k+1} are strictly less than 1 for $\lambda > 0$, Q is a contraction function in terms of ∞-norm, see [4] for the detailed information.

Theorem 3.1. *For any positive regularization parameter, i.e.* $\lambda > 0$*, the fixed-point scheme* (3.7) *has a unique bounded solution.*

Note $A(x^n, \lambda)$ is an M-matrix, i.e. the off-diagonal entries of matrix $A(x^n, \lambda)$ are less than or equal to zero, while the diagonal entries are positive, and are strictly diagonally dominant. Therefore, multigrid methods can be effectively used to solve the linearized problem [98, 107]. In this work, we use the BoxMG and AMG methods as black-box solvers and compare their performance in solving the anisotropic diffusion equation (3.5).

3.4 Choosing Regularization Parameters

We will compare the performance of the AOS scheme for (3.2), (3.4) and fixed-point iterations with multigrid solvers for (3.5). The remaining outstanding issue in solving the anisotropic diffusion equations given in (3.2), (3.4) and (3.5) is the choice of the regularization parameters: the diffusion stopping time, T, and/or the parameter, λ.

In [114], Weickert points out that since the variance of the continuum solution of the anisotropic diffusion equation at time t, $\mathrm{var}(X(t))$, is monotonously decreasing, the relative variance

$$\frac{\mathrm{var}(X(t))}{\mathrm{var}(\tilde{X})}, \tag{3.8}$$

decreases monotonically from 1 to 0. This ratio measures the distance of $X(t)$ from the initial state $\tilde{X}$ and final state $X(t)$ when $t = \infty$. Prescribing a certain value for the above ratio provides a criterion for the stopping time. Assuming that the signal-to-noise ratio (SNR) is known, [114] proposes to choose the stopping time, T_0, to satisfy the relation

$$\frac{\mathrm{var}(X(T_0))}{\mathrm{var}(\tilde{X})} = \frac{1}{1 + \frac{1}{\mathrm{SNR}}}. \tag{3.9}$$

However, one of the drawbacks of this method is that it requires that the SNR be known for the noisy image. Otherwise, the user must specify a threshold for the ratio (3.8). This shifts the problem of choosing a stopping time to that of choosing a threshold. Moreover, as pointed out by Weickert, criterion (3.9) tends to underestimate the optimal stopping time, as even a well-tuned filter cannot avoid influencing the signal before eliminating the noise. For these reasons, in the numerical results section we will use methods other than (3.9) for selecting the stopping time for AOS.

In [79], Mrázek proposed a decorrelation criterion to choose the diffusion stopping time, which is claimed to outperform (3.9). Given the assumption that the noise is uncorrelated with the unknown true image, the decorrelation method for choosing the diffusion stopping time, T, is to minimize the correlation coefficient (CC),

$$T = \underset{t \geq 0}{\mathrm{argmin}} \ \frac{\mathrm{cov}(X(t) - \tilde{X}, X(t))}{\sqrt{\mathrm{var}(X(t) - \tilde{X}) \cdot \mathrm{var}(X(t))}}.$$

Note that this CC idea can be modified to choose λ in (3.5) according to

$$\lambda = \underset{\lambda}{\mathrm{argmin}} \ \frac{\mathrm{cov}(X_\lambda - \tilde{X}, X_\lambda)}{\sqrt{\mathrm{var}(X_\lambda - \tilde{X}) \cdot \mathrm{var}(X_\lambda)}}, \tag{3.10}$$

where X_λ is the restored image computed with the regularization parameter, λ.

The Brent-NCP method in previous chapter provides a viable alternative to above approaches and is flexible enough that it can be used to find either the stopping time in (3.2) or the λ in (3.5). In practice, Algorithm 1 converges very quickly for all the regularization methods considered. In the following experiments, we use Algorithm 1 to find three regularization parameters.

- stopping time for (3.2) with AOS scheme
- regularization parameter λ in (3.5) with the fixed-point iteration and Multgird solver;
- regularization parameter in total variation regularization.

3.5 Numerical Results

This section is devoted to presenting the results obtained with the proposed algorithm. Comparisons with the AOS scheme for (3.2) [116] and (3.4), TV denoising [30, 97] and the block matching 3D (BM3D) method [37] are also presented.

3.5.1 Comparison Measures

After getting the restored image, we compute the mean structure similarity (MSSIM) of the restored and noise-free images as a measurement of the restored image quality [110]. Given any two discrete images, X and Z, the structure similarity (SSIM) measure is defined as

$$\mathrm{SSIM}(X, Z) = \frac{(2\mu_X\mu_Z + c_1)(2\mathrm{cov}(X, Z) + c_2)}{(\mu_X^2 + \mu_Z^2 + c_1)(\sigma_X^2 + \sigma_Z^2 + c_2)} \tag{3.11}$$

where μ_X and μ_Z are the means of images X and Z respectively, σ_X and σ_Z are the variances of images X and Z, $\mathrm{cov}(X, Z)$ is the covariance of the two images, and c_1 and c_2 are two parameters to stabilize the division with small denominators, the defaults are $c_1 = 0.0001, c_2 = 0.0009$.

In practice, SSIM is calculated on local windows rather than over the whole image. As in [110], we use a normalized 11×11 circular-symmetric Gaussian weighting matrix w, with standard deviation of 1.5. As the result, for kth local window, X^k and Z^k, local mean, μ_{X^k}, variance, σ_{X^k}, and covariance, $\mathrm{cov}(X^k, Z^k)$, in the SSIM measure (3.11) are modified as

$$\mu_{X^k} = \sum_{i,j=1}^{11} w_{i,j} X_{i,j}^k, \quad \sigma_{X^k} = \left(\sum_{i,j=1}^{11} w_{i,j} (X_{i,j}^k - \mu_{X^k})^2 \right),$$

$$\mathrm{cov}(X^k, Z^k) = \sum_{i,j=1}^{11} w_{i,j} (X_{i,j}^k - \mu_{X^k})(Z_{i,j}^k - \mu_{Z^k}).$$

In order to get a single overall similarity measure of the two images, the MSSIM is the mean of the SSIM of the local windows,

$$\mathrm{MSSIM}(X, Z) = \frac{1}{n} \sum_{k=1}^{n} \mathrm{SSIM}(X^k, Z^k).$$

We also consider the traditional image-quality measurement, the peak signal-to-noise ratio (PSNR) [109],

$$\mathrm{PSNR} = 20 \times \log_{10} \left(\frac{\max(X)}{\sqrt{\mathrm{MSE}}} \right),$$

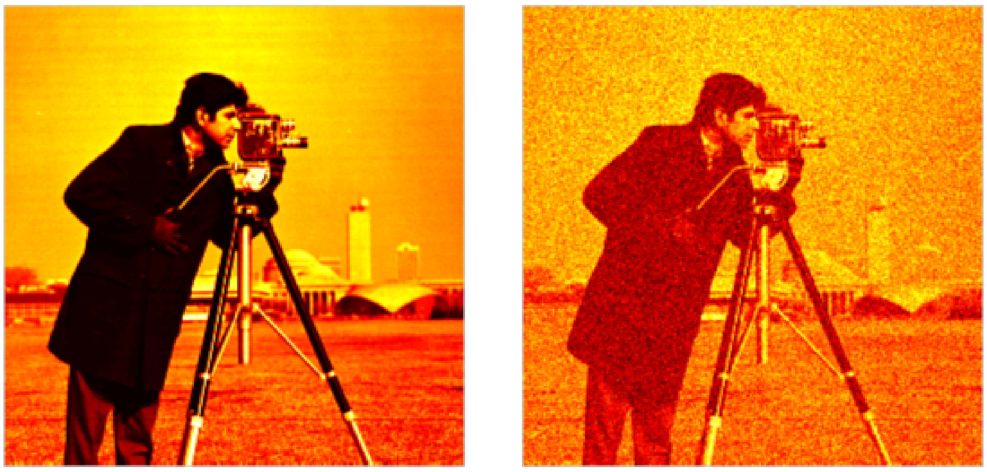

(a) Luminance shift image, PSNR = 19.5, MSSIM = 0.91 (b) White Gaussian noise, PSNR = 19.5, MSSIM = 0.31.

Figure 3.2: Comparison of MSSIM and PSNR. (a) luminance-shift image. (b) white Gaussian noise. Note (a) and (b) have the same PSNR index, 19.5, the MSSIM of (a) is 0.91, which is much larger than that for (b), 0.31.

where $\max(X)$ is the maximum possible intensity value of the clean image, and MSE is the mean squared error between the clean and noisy images. Our experiments show that MSSIM is more suitable for measuring the quality of denoised images, see Figure 3.2. In this figure, both the luminance-shift image and the one with white noise have the same PSNR index, 19.5; however, by visual quality, the luminance-shift image is noise-free, and is much better than the one with white noise. This is shown from the MSSIM index: the MSSIM of the luminance-shift image is 0.91, while the MSSIM for the image with white noise is 0.31.

3.5.2 Experiments

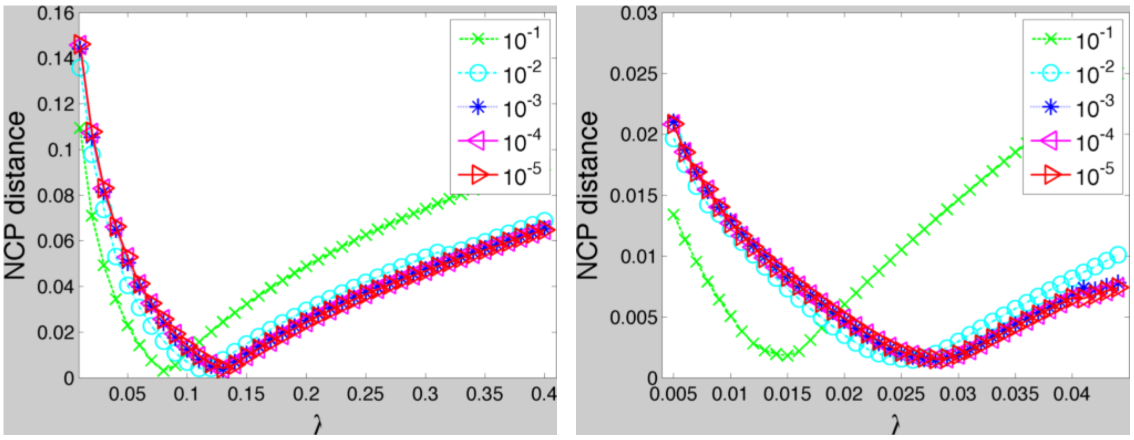
We consider 5 common test images, also used in [37], and add various levels of Gaussian noise to these images to test the algorithms. We include both the MSSIM and PSNR as measures of the restored image quality for the convenience of comparison with other papers, but note the results in Figure 3.2.

As mentioned previously, the quality of the restored image depends on the value of the regularization parameter. Thus, no matter which regularization scheme is employed, there will be an outer loop over the regularization parameter values. We initially consider two possibilities – choosing the λ that minimizes the CC functional (3.10) vs. choosing the λ that minimizes the NCP functional (2.2). We show experimentally in the next section that each of these has a well-defined minimum, and use Brent's method to solve for that minimum (i.e. Brent-CC and Brent-NCP). For AOS applied to solve (3.2), the regularization parameter is the stopping time T, and the CC or the NCP approach can be used to determine this value. If solving either (3.4) (with AOS) or (3.5) (fixed point with BoxMG or AMG), the undetermined regularization parameter is λ.

The numerical experiments are outlined as follows

- In Subsection 3.5.2.1, we fix the regularization approach as that of solving (3.5) using fixed point iteration with BoxMG as the linear solver. Then we investigate the use of the CC functional vs. the NCP functional to choose λ.
- In Subsection 3.5.2.2, we fix the selection approach (i.e. outer iteration) as Brent-NCP and compare results given using three algorithms: AOS to solve (3.4) for each λ (we refer to this as AOS-R); solving (3.5) with fixed point iteration and BoxMG as the linear solver; solving (3.5) with fixed point iteration and AMG as the linear solver.
- To illustrate the applicability of our Brent-NCP regularization parameter selection approach, in Subsection 3.5.2.3, we fix the selection approach as Brent-NCP and apply it to finding the regularization parameter (stopping time, T) for AOS applied to (3.2) and to finding the regularization parameter λ in Total Variation, a well-known denoising scheme. We compare these results with the traditional AOS scheme as presented in [79] and against the BoxMG results obtained in the previous subsection.
- We conclude with a comparison of the diffusion-based denoising techniques to the block-based denoising technique known as BM3D in Subsection 3.5.2.4.

For the anisotropic diffusion approach based on (3.5), two important technical parameters need to be fixed: the outer stopping tolerance for the fixed-point iteration, tol_{fp}, and the inner stopping tolerance for the multigrid solvers for each linearization, tol_{mg}. In Figure 3.3, we investigate the effects of varying tol_{fp} with tol_{mg} fixed at 0.1. Note that while the optimal λ changes significantly with tol_{fp}, the quality of the restored image, as measured by the NCP distance is much less sensitive. Similarly, Table 3.1 compares the effects of the inner stopping tolerance, tol_{mg}, with tol_{fp} fixed at 10^{-3}. Note that the computed regularization parameter is essentially unchanged by choosing larger values of tol_{mg}. Thus, we take $\text{tol}_{\text{mg}} = 0.1$ in the experiments here to improve the overall efficiency of the approach, in combination with $\text{tol}_{\text{fp}} = 10^{-3}$.



(a) BoxMG, noise level = 0.3 (b) BoxMG, noise level = 1.0

Figure 3.3: Test of the tolerance of the fixed-point iteration, $\mathrm{tol_{fp}}$, NCP-λ plots for cameraman image. For given regularization parameter, λ, on the x-axis, the y-axis shows the NCP distance measurements of the restored images computed using the corresponding λ. As $\mathrm{tol_{fp}}$ decreases from 10^{-1} to 10^{-5}, the NCP-λ plot converges to a limiting curve for different noise levels, 0.3 (left) and 1.0 (right).

3.5.2.1 Optimizing CC vs NCP for Choosing λ

We first show experimentally that there are unique minimizers for both (2.2) and (3.10), and that employing Brent's method achieves these minimizers. We solve our objective function (3.5) using the fixed-point iteration with BoxMG as the inner linear solver, where the regularization parameters are computed using Brent's method. For the CC method, instead of computing $\mathcal{N}(\lambda)$ in Steps 4 and 12, we calculate the correlation coefficient as in Equation (3.10). Figures 3.4 and 3.5 show the results for two test images: cameraman and fingerprint, with noise level varying from 0.3 to 0.6. The dots shown in Figures 3.4 and 3.5 are the results computed using Brent's method. It is clear that the computed solutions are the minimizers of the NCP distance and correlation coefficient functions.

Figures 3.4 and 3.5 also show the MSSIM and PSNR measurements of the restored images. Compared to the CC method, one of the advantages of the NCP approach is that the MSSIM and PSNR measurements of the restored images corresponding to the minimizers of the NCP distance are close to the maximum values of MSSIM and PSNR that are achieved. While this is also nearly true for the CC method applied to the cameraman image, it is obviously not the case for the fingerprint image, which has more texture information than the cameraman image. This experiment shows that the NCP method is more stable than the CC method and that minimizing the NCP correlates nicely and consistently across images with achieving large MSSIM or PSNR measures of the restored images.

Next, we compare the restored image quality computed using these regularization parameter-choice methods. The results are shown in Table 3.2. For the images with little texture information, such as the cameraman image, the results using CC are comparable to those using NCP, especially for low noise levels. However, for large noise levels, or images with lots of texture information, NCP clearly outperforms CC. This is especially noticeable in the fingerprint image; with a noise level of 1.0, the regularization parameter chosen based on the NCP criterion yields a significant improvement on that chosen based on the CC criterion, particularly when considering their MSSIM measures of 0.40 and 0.11.

3.5.2.2 Comparison of AMG, BoxMG, and AOS for regularized diffusion

We first compare the restored images computed using two different multigrid solvers, BoxMG and AMG, using both solvers in the computation of the optimal regularization parameters using Brent's method for the cameraman image with different noise levels. The results are shown in Table 3.3, where we see that the number of Brent's method and linearization steps are nearly the same for both solvers. This is also true for the computed regularization parameters. Furthermore, from the results in Table 3.4, we can see that the MSSIM and PSNR measurements of restored images when using BoxMG and AMG to solve the anisotropic diffusion equation (3.5) are almost the same. Note that while the iteration counts for AMG look better than those for BoxMG, BoxMG is computationally faster. As shown in Table 3.3, BoxMG is about 6 times faster than AMG in terms of computational time, due to the combination of the more expensive setup phase within AMG and its use of unstructured storage and indirect addressing. While AMG is known to be very robust and effective for highly discontinuous diffusivities, these results show that the added costs required for AMG do not pay off in this situation.

As mentioned in Section 3.2, the AOS scheme can be applied directly to solve the time-dependent regularized anisotropic diffusion equation (3.4). We call this scheme AOS-R. We use an outer iteration to find λ by the Brent-NCP Algorithm 1. In each iteration with fixed λ, we solve for the steady-state solution of the regularized diffusion equation with AOS-R, with $\tau = 0.5$. This inner iteration stops when the relative difference between two consecutive steps measured in the Frobenius norm becomes less than 10^{-5}. Numerical experiments indicate that smaller timesteps or a more accurate stopping criterion do not improve these results. The quality measures of the results are shown in Table 3.4. From this table, we can see that both the AMG and BoxMG solvers for (3.5) (reported results are for λ values chosen using Brent-NCP as the outer wrapper in each case as well) outperform the AOS-R for (3.4) in terms of accuracy. The results suggest that solving somewhat inaccurately over a possibly large number of time steps permits some unintentional smoothing into the process of solving for the steady state solution, whereas solving (3.5) by fewer, and more accurate, fixed point steps, prevents this problem (see also [9]). Another interesting feature to note is that the PSNR values for the AOS-R results are very static with respect to noise level for each test problem. We consider this further evidence of the pitfalls of using PSNR to measure restored image quality (see also examples in [109]).

3.5.2.3 Comparison with Unregularized AOS and Total Variation Denoising

We test the applicability of the Brent-NCP Algorithm 1 by computing the diffusion stopping time in (3.2) for the AOS scheme and the regularization parameter, λ, for TV denoising.

For the AOS scheme, in the experiments of [79, 116], the authors use fixed time steps, τ, to compute the restored images. If we use a fixed time step, say $\tau = 0.5$, the computational time for the high noise level images will become very large compared with the other methods. Moreover, in [116], Weickert *et al.* point out that AOS with semi-implicit time-stepping is stable for all time steps. Therefore, instead of fixing the time step, τ, we fix the number of time steps to be 10, i.e. the time step, $\tau = \frac{T}{10}$, where T is the stopping time returned by the Brent's method. The reason we choose $\tau = \frac{T}{10}$ is illustrated later in the discussion.

The objective function for TV is

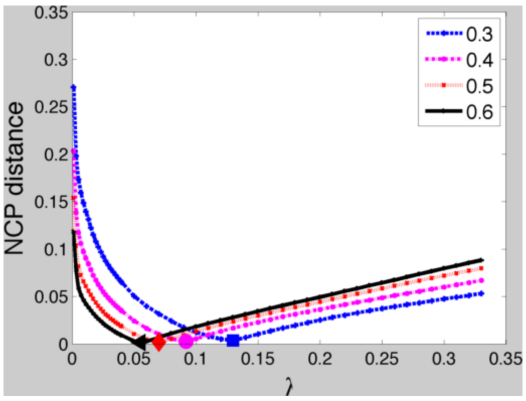
$$\operatorname*{argmin}_{x} \, \text{TV}(X) + \frac{\lambda}{2}\|X - \tilde{X}\|_f^2, \tag{3.12}$$

where $\text{TV}(X) = \iint_\Omega |\nabla X|$, and λ is the regularization parameter [97], $\|\cdot\|_f$ represents the Frobenius norm. Here, we use the algorithm proposed by Chambolle [30] to solve (3.12). Different regularization parameters, λ, have tremendous impact on the restored image quality. Traditionally, the regularization parameter is chosen by trial and error. Recent research into better ways to choose the regularization parameter has included methods based on image geometry or local variance estimators [41, 103]. Figure 3.6 shows the MSSIM and PSNR measurement of the restored images computed using TV denoising, where the regularization parameters are computed using the Brent-NCP method replacing the anisotropic diffusion solves in Step 4 and 12 with the TV minimization in (3.12). Note that minimizing the NCP correlates nicely with achieving large MSSIM or PSNR measures of the restored images.

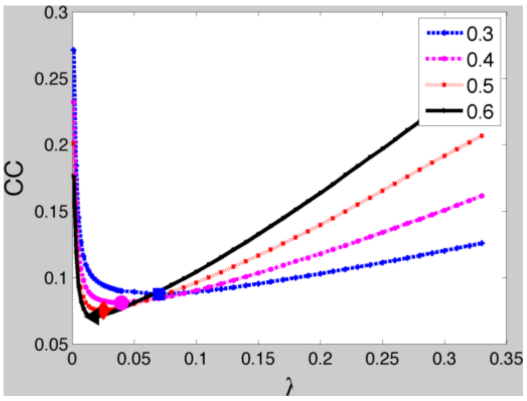
Table 3.5 gives comparisons of the unregularized AOS scheme for (3.2) (AOS-U) with diffusion time, T, chosen by the Brent-NCP algorithm, TV denoising with regularization parameter, λ, chosen by the Brent-NCP algorithm, and the traditional AOS scheme (AOS-T) where the diffusion time, T, is chosen as in [79], with $\tau = 0.5s$. For the AOS scheme with the Brent-NCP method for choosing the stopping time in (3.2), we found that 10 discrete time steps were sufficient to give restored images of quality comparable to the other two methods in our results. The quality of the restored images using anisotropic diffusion with the proposed algorithm is comparable to that of the images computing using the AOS-U and TV denoising. They all are clearly better than the results by AOS-T, especially in the MSSIM measures. As shown in Figure 3.7, it is difficult to see any difference in the detail regions shown for the restored Barbara images. One possible way to enhance the restored image quality using the AOS-T scheme is to use a very small time step, τ. However, due to the added cost of the additional time steps, the execution time will become much longer than the other schemes. Since all these methods are greatly dependent on the choice of regularization parameters, which are all computed using the Brent-NCP algorithm, this table also shows the broad applicability of the Brent-NCP algorithm in choosing regularization parameters.

3.5.2.4 Comparison with Block-Based Denoising

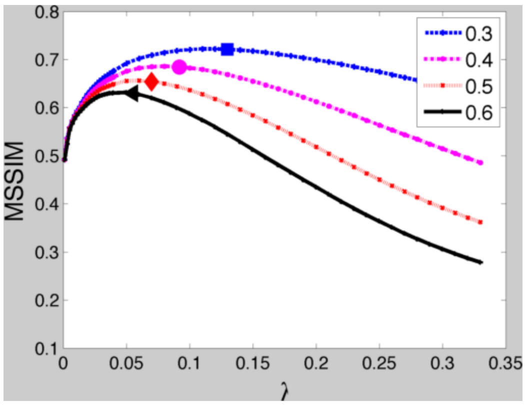
While diffusion-based denoising techniques are common in the literature, many other approaches are also possible. Among them, BM3D proves to be very effective and is one of the best denoising methods in the literature [37]. BM3D is a block-based approach that collects the local information in a noisy image and groups similar 2D image fragments together into 3D data arrays. Then, a collaborative filtering technique is used to deal with these 3D groups. Table 3.6 compares BM3D with the anisotropic diffusion approach presented here. One of the disadvantages of the BM3D algorithm is that it requires the user to input the estimated noise level. Given an accurate estimated noise level, the restored images from BM3D have better quality than those generated by anisotropic diffusion in terms of both MSSIM and PSNR. However, noise estimation itself is a difficult research area [75]. If the input noise level is not accurate, the restored images can have a bad quality, especially if the noise level is underestimated, see Figure 3.7. This is also shown in Table 3.6. We use these different noise level inputs: the exact noise level, n_0, an underestimated noise level, $0.7 \times n_0$, and an overestimated noise level, $1.3 \times n_0$. The MSSIM and PSNR measurements decrease substantially for the underestimated case. Here, we point out that, in practice, it is very difficult to accurately estimate the true noise level. For this reason, it is not fair to compare between anisotropic diffusion denoising and BM3D by simply looking at the measurements without considering the algorithms' requirements.



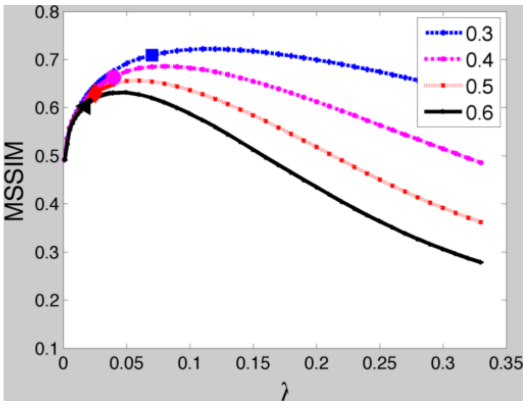
(a) NCP-λ for NCP parameter choice



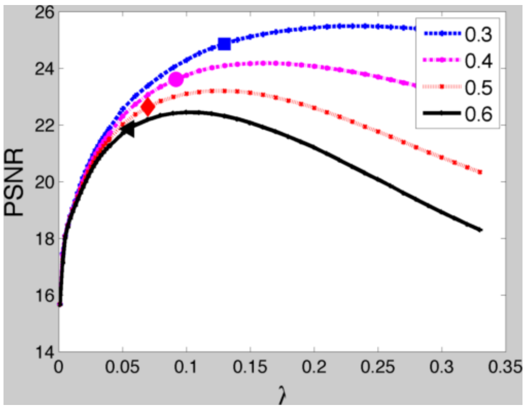
(b) CC-λ for CC parameter choice



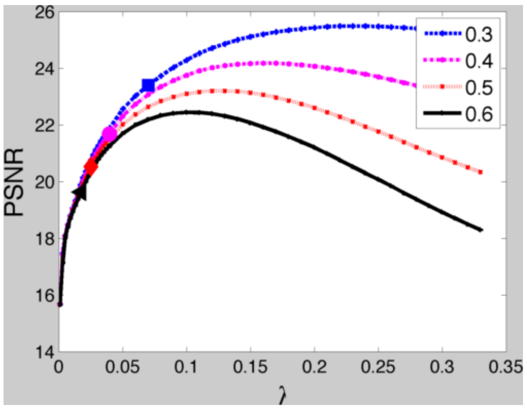
(c) MSSIM-λ for NCP parameter choice



(d) MSSIM-λ for CC parameter choice

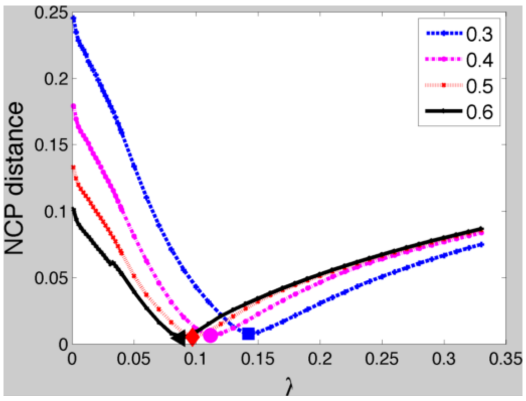


(e) PSNR-λ for NCP parameter choice

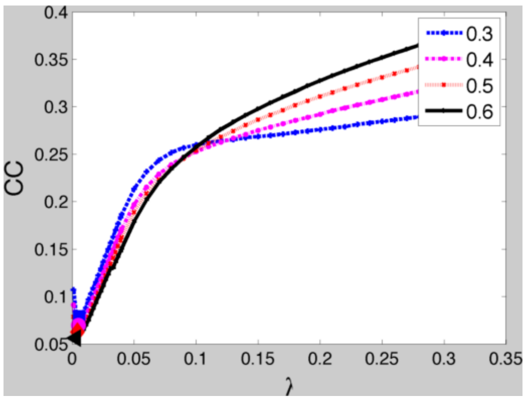


(f) PSNR-λ for CC parameter choice

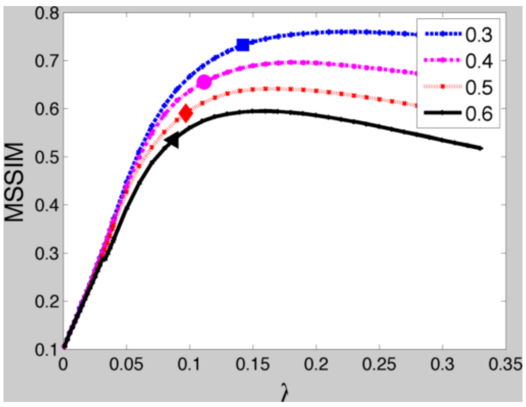
Figure 3.4: Anisotropic diffusion restored image quality for cameraman image, where the noise level varies from 0.3 to 0.6. The markers are the restored image measurements computed using the optimal regularization parameters, λ, which are computed using Brent-NCP (left) and Brent-CC (right). As shown in the plots, the restored images using the regularization parameters chosen by Brent-NCP (left) are near optimal, i.e. near maximum of the MSSIM and PSNR measures. But the results using the regularization parameter chosen by Brent-CC (right) are not so close to the near optima.



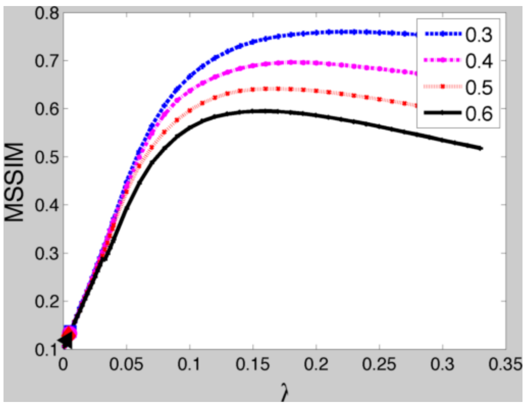
(a) NCP-λ for NCP parameter choice



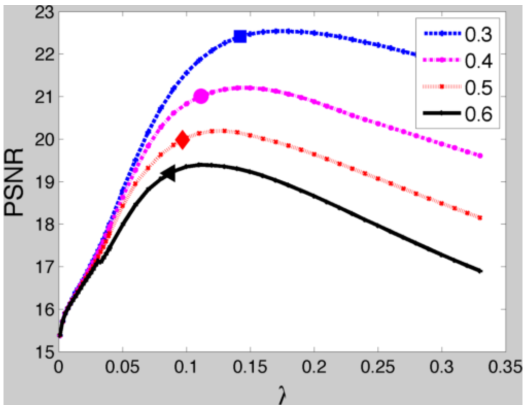
(b) CC-λ for CC parameter choice



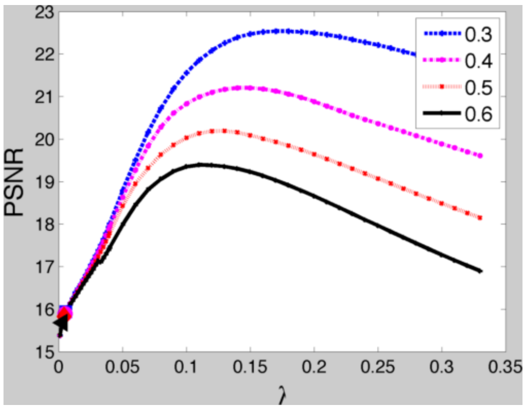
(c) MSSIM-λ for NCP parameter choice



(d) MSSIM-λ for CC parameter choice

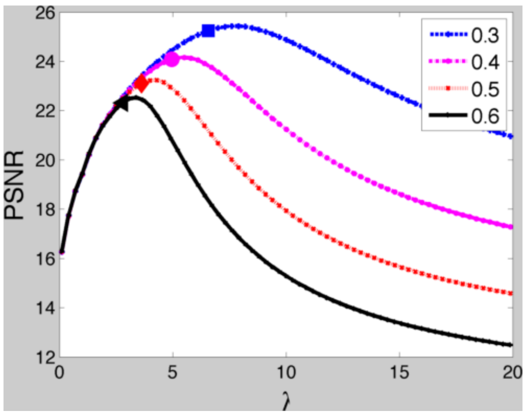


(e) PSNR-λ for NCP parameter choice

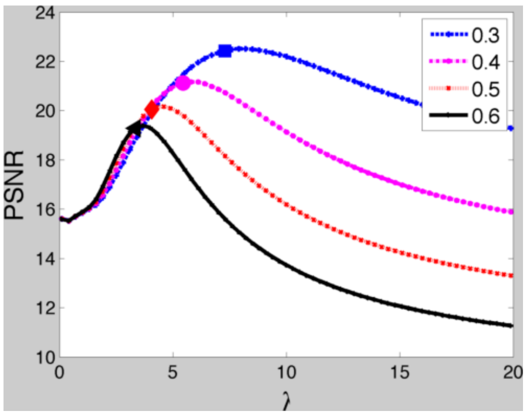


(f) PSNR-λ for CC parameter choice

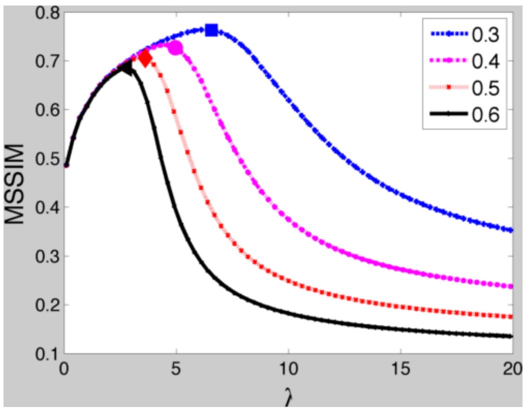
Figure 3.5: Anisotropic diffusion restored image quality for fingerprint image, where the noise level varies from 0.3 to 0.6. The markers are the restored image measurements computed using the optimal regularization parameters, λ, which are computed using Brent-NCP (left) and Brent-CC (right). As shown in the plots, the restored images using the regularization parameters chosen by Brent-NCP (left) are near optimal, i.e. near maximum of the MSSIM and PSNR measures. But the results using the regularization parameter chosen by Brent-CC (right) are not so close to the near optima.



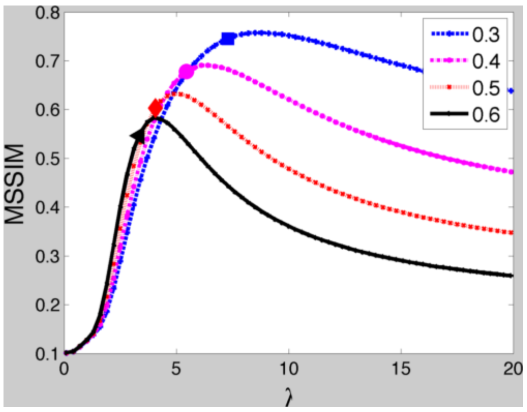
(a) NCP-λ



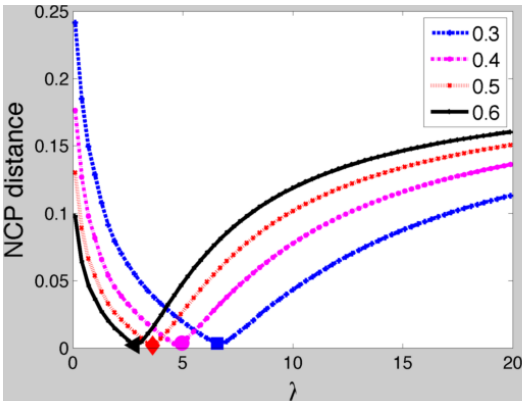
(b) NCP-λ



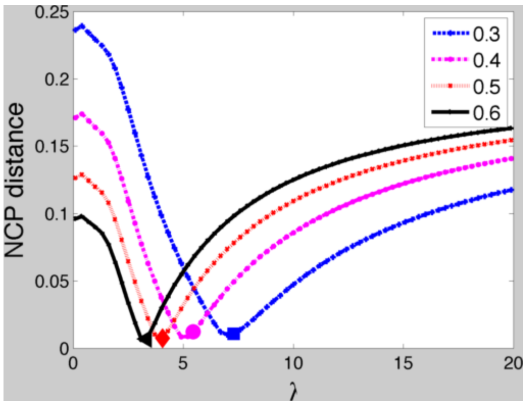
(c) MSSIM-λ



(d) MSSIM-λ

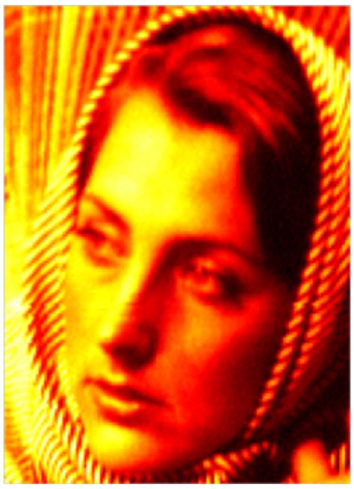


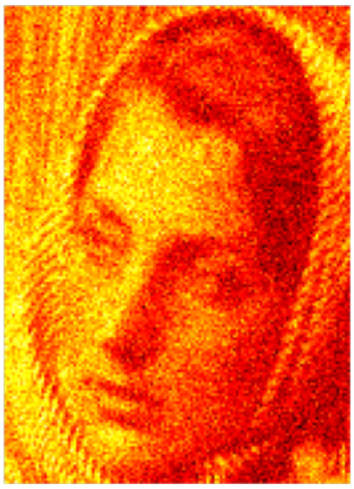
(e) PSNR-λ

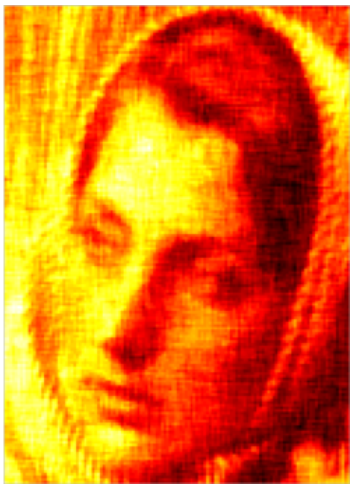


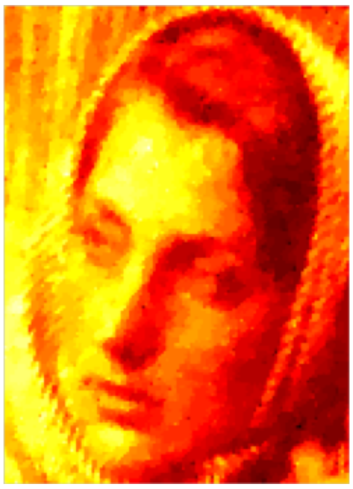
(f) PSNR-λ

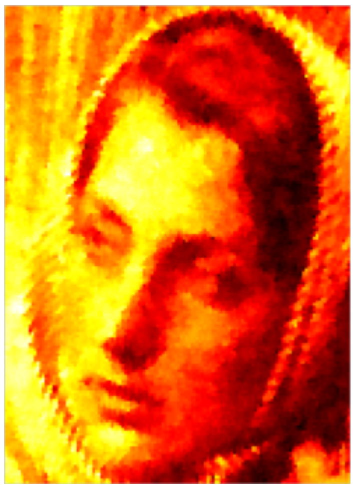
Figure 3.6: TV restored image quality for cameraman (left) and fingerprint (right), where the noise level varies from 0.3 to 0.6. The markers are the restored image measurements computed using the optimal regularization parameters, λ, which are computed using Brent-NCP and TV denoising.

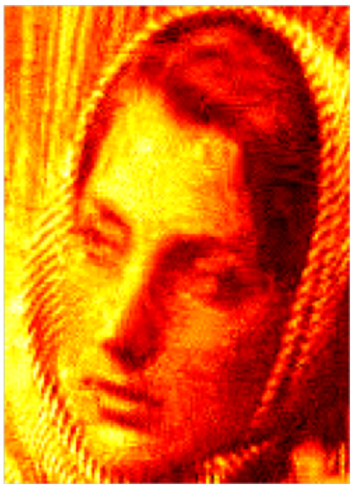

(a) noise-free

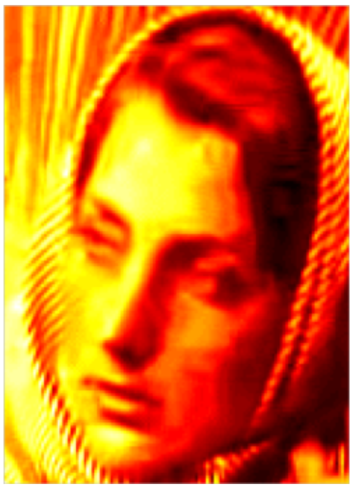

(b) noisy


(c) AOS-U


(d) TV


(e) AD


(f) BM3D with under-estimated noise level


(g) BM3D with exact noise level

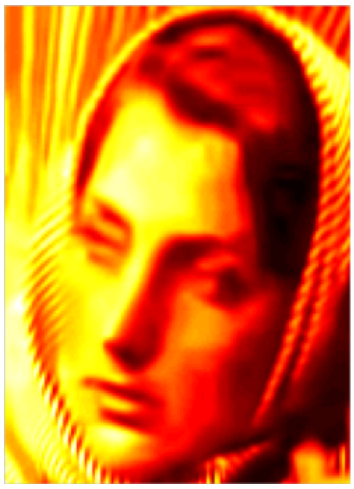

(h) BM3D with over-estimated noise level

Figure 3.7: Detailed restored Barbara image by different methods. The noise level in the noisy image (b) is 0.2. The regularization parameters for AOS-U, TV, and AD are computed use Brent-NCP method. The MSSIM and PSNR measurements for the restored images are shown in Tables 3.4, 3.5, and 3.6.

Table 3.1: Comparison of the computed regularization parameters and the computational work in the Brent-NCP algorithm using BoxMG to solve the linearized systems. Given fixed $\text{tol}_{\text{fp}} = 10^{-3}$, this table compares the effects of the inner stopping tolerance, tol_{mg}. Note that the computed regularization parameter is essentially unchanged by choosing larger values of tol_{mg}. Thus, we take $\text{tol}_{\text{mg}} = 0.1$ in the experiments here to improve the overall efficiency of the approach.

noise level	tol_{mg}	Brent steps	linearization steps	V-cycles	time(s)	λ_{opt}	relative change(%)
0.3	10^{-6}	16	153	1351	21.5	1.30e-01	0
	10^{-5}	16	153	1052	16.2	1.30e-01	2.51e-06
	10^{-4}	16	153	787	13.6	1.30e-01	4.89e-04
	10^{-3}	16	154	510	11.1	1.30e-01	2.49e-02
	10^{-2}	16	154	322	9.2	1.30e-01	2.72e-03
	10^{-1}	15	146	158	7.3	1.30e-01	2.38e-01
1.0	10^{-6}	16	213	1926	26.7	2.80e-02	0
	10^{-5}	16	213	1511	22.7	2.80e-02	1.81e-05
	10^{-4}	16	213	1098	18.6	2.80e-02	2.12e-04
	10^{-3}	16	213	737	15.2	2.80e-02	3.49e-03
	10^{-2}	16	215	451	12.6	2.80e-02	4.51e-01
	10^{-1}	16	221	233	10.8	2.80e-02	4.95e-01

Table 3.2: Comparison of NCP and CC criteria for choosing the regularization parameter, λ. All the results are computed using Brent's algorithm and fixed-point iteration with the BoxMG solver. The highlighted numbers show the measurements, CC or NCP, that gave the highest value of measure (MSSIM and PSNR, respectively) for each experiment. One experiment is one row of the table.

images	noise level	MSSIM			PSNR		
		noisy	CC	NCP	noisy	CC	NCP
cameraman	0.1	0.52	0.84	**0.84**	25.5	30.2	**29.8**
	0.2	0.31	0.76	**0.77**	19.5	25.7	**26.7**
	0.3	0.21	0.71	**0.72**	16.0	23.4	**24.9**
	0.4	0.16	0.66	**0.68**	13.5	21.7	**23.6**
	0.5	0.12	0.63	**0.65**	11.5	20.5	**22.6**
	0.6	0.10	0.60	**0.63**	10.0	19.6	**21.9**
	0.7	0.08	0.59	**0.61**	8.6	19.3	**21.2**
	0.8	0.06	0.58	**0.59**	7.5	19.1	**20.7**
	0.9	0.05	**0.58**	**0.58**	6.4	18.9	**20.3**
	1.0	0.04	0.57	**0.57**	5.5	18.8	**20.0**
house	0.1	0.44	0.84	**0.84**	24.3	31.8	**31.5**
	0.2	0.23	0.78	**0.78**	18.3	28.2	**28.2**
	0.3	0.14	0.73	**0.74**	14.8	25.4	**26.3**
	0.4	0.10	0.68	**0.71**	12.3	22.7	**25.0**
	0.5	0.07	0.64	**0.69**	10.3	20.9	**24.1**
	0.6	0.05	0.64	**0.67**	8.8	20.5	**23.3**
	0.7	0.04	0.63	**0.65**	7.4	19.8	**22.6**
	0.8	0.03	0.62	**0.64**	6.3	19.7	**22.1**
	0.9	0.03	0.62	**0.63**	5.2	19.5	**21.6**
	1.0	0.02	0.62	**0.63**	4.3	19.4	**21.1**
barbara	0.1	0.62	0.72	**0.84**	25.6	24.5	**28.7**
	0.2	0.38	0.66	**0.71**	19.6	23.5	**24.9**
	0.3	0.26	0.62	**0.64**	16.0	22.9	**23.5**
	0.4	0.19	0.59	**0.60**	13.5	22.4	**22.8**
	0.5	0.14	0.57	**0.58**	11.6	21.9	**22.3**
	0.6	0.10	0.55	**0.56**	10.0	21.4	**21.8**
	0.7	0.08	0.53	**0.54**	8.7	20.9	**21.5**
	0.8	0.07	0.51	**0.52**	7.5	20.5	**21.2**
	0.9	0.05	0.50	**0.51**	6.5	20.1	**20.9**
	1.0	0.04	0.49	**0.50**	5.6	19.6	**20.6**
boat	0.1	0.57	0.80	**0.82**	25.3	30.0	**30.4**
	0.2	0.31	0.67	**0.72**	19.3	25.9	**27.4**
	0.3	0.19	0.60	**0.66**	15.8	23.9	**25.8**

Continued on Next Page. . .

Table 3.2 – Continued

images	noise level	MSSIM			PSNR		
		noisy	CC	NCP	noisy	CC	NCP
	0.4	0.13	0.56	**0.62**	13.3	22.6	**24.7**
	0.5	0.10	0.53	**0.59**	11.4	21.9	**23.9**
	0.6	0.07	0.52	**0.57**	9.8	21.5	**23.3**
	0.7	0.06	0.47	**0.55**	8.4	19.5	**22.8**
	0.8	0.04	0.47	**0.53**	7.3	19.3	**22.3**
	0.9	0.04	0.46	**0.52**	6.3	19.1	**21.9**
	1.0	0.03	0.46	**0.51**	5.3	18.8	**21.6**
fingerprint	0.1	0.84	0.91	**0.92**	24.6	27.5	**28.2**
	0.2	0.61	0.83	**0.82**	18.6	24.3	**24.5**
	0.3	0.43	0.14	**0.73**	15.0	15.9	**22.4**
	0.4	0.31	0.13	**0.66**	12.5	15.9	**21.0**
	0.5	0.23	0.13	**0.59**	10.6	15.8	**20.0**
	0.6	0.18	0.12	**0.53**	9.0	15.7	**19.2**
	0.7	0.14	0.12	**0.49**	7.7	15.7	**18.6**
	0.8	0.11	0.12	**0.46**	6.5	15.6	**18.2**
	0.9	0.09	0.11	**0.43**	5.5	15.6	**17.9**
	1.0	0.08	0.11	**0.40**	4.6	15.6	**17.6**

Table 3.3: Comparison of computational time and results in Brent-NCP algorithm using two different multigrid solvers: AMG and BoxMG for the cameraman image. This table shows that the number of Brent's method and linearization steps are nearly the same for both solvers, as are the computed regularization parameters. However, the computational time for BoxMG is only about 1/6 of that of AMG due the expensive setup phase and indirect addressing used within AMG. The highlighted timing results correspond to the faster of the two methods.

noise level	optimal λ		time		brent steps		linearization step		total V-cycles	
	AMG	BoxMG	AMG	BoxMG	AMG	BoxMG	AMG	BoxMG	AMG	BoxMG
0.1	4.61e-1	4.61e-1	17.3	**2.7**	10	10	53	53	53	94
0.2	2.04e-1	2.05e-1	25.0	**4.9**	11	12	87	96	107	198
0.3	1.30e-1	1.30e-1	44.0	**7.4**	16	15	153	146	246	322
0.4	9.16e-2	9.16e-2	42.2	**7.5**	14	14	151	151	250	314
0.5	6.98e-2	6.98e-2	40.8	**7.2**	13	13	144	144	233	299
0.6	5.49e-2	5.51e-2	41.2	**7.0**	12	12	142	141	229	290
0.7	4.47e-2	4.48e-2	51.6	**8.7**	14	14	179	177	296	401
0.8	3.74e-2	3.75e-2	60.2	**9.9**	15	15	205	200	355	389
0.9	3.19e-2	3.20e-2	64.4	**10.7**	15	15	218	214	378	435
1.0	2.79e-2	2.80e-2	65.8	**10.8**	15	16	220	221	392	451

Table 3.4: Comparison of restored images using different methods: BoxMG and AMG solvers for (3.5), and regularized AOS, AOS-R. For the AOS-R scheme, we take time step, $\tau = 0.5$, and use a relative difference stopping criterion of $relerr = 10^{-5}$. The highlighted numbers are the proposed anisotropic diffusion denoising with BoxMG solver. The restored results are better than AOS-R method in terms of both MSSIM and PSNR measures.

images	noise level	MSSIM				PSNR			
		noisy	BoxMG	AMG	AOS-R	noisy	BoxMG	AMG	AOS-R
	0.1	0.52	**0.84**	0.84	0.55	25.5	**29.8**	29.8	11.2
	0.2	0.31	**0.77**	0.77	0.56	19.5	**26.7**	26.7	11.3
	0.3	0.21	**0.72**	0.72	0.53	16.0	**24.9**	24.9	11.3
	0.4	0.16	**0.68**	0.68	0.48	13.5	**23.6**	23.6	11.3
	0.5	0.12	**0.65**	0.65	0.41	11.5	**22.6**	22.6	11.3
cameraman	0.6	0.10	**0.63**	0.63	0.33	10.0	**21.9**	21.9	11.3
	0.7	0.08	**0.61**	0.61	0.26	8.6	**21.2**	21.2	11.2
	0.8	0.06	**0.59**	0.59	0.22	7.5	**20.7**	20.7	11.1
	0.9	0.05	**0.58**	0.58	0.20	6.4	**20.3**	20.3	11.1
	1.0	0.04	**0.57**	0.57	0.21	5.5	**20.0**	20.0	11.1
	0.1	0.44	**0.84**	0.84	0.59	24.3	**31.5**	31.5	10.2
	0.2	0.23	**0.78**	0.78	0.58	18.3	**28.2**	28.2	10.3
	0.3	0.14	**0.74**	0.74	0.52	14.8	**26.3**	26.3	10.3
	0.4	0.10	**0.71**	0.71	0.41	12.3	**25.0**	25.0	10.3
	0.5	0.07	**0.69**	0.69	0.32	10.3	**24.1**	24.1	10.2
house	0.6	0.05	**0.67**	0.67	0.30	8.8	**23.3**	23.3	10.2
	0.7	0.04	**0.65**	0.65	0.34	7.4	**22.6**	22.6	10.2
	0.8	0.03	**0.64**	0.64	0.37	6.3	**22.1**	22.1	10.2
	0.9	0.03	**0.63**	0.63	0.38	5.2	**21.6**	21.6	10.2
	1.0	0.02	**0.63**	0.63	0.40	4.3	**21.1**	21.1	10.2
	0.1	0.62	**0.84**	0.84	0.48	25.6	**28.7**	28.7	11.4
	0.2	0.38	**0.71**	0.71	0.48	19.6	**24.9**	24.9	11.4

Continued on Next Page...

barbara

Table 3.4 – Continued

images	noise level	MSSIM				PSNR			
		noisy	BoxMG	AMG	AOS-R	noisy	BoxMG	AMG	AOS-R
	0.3	0.26	**0.64**	0.64	0.46	16.0	**23.5**	23.5	11.4
	0.4	0.19	**0.60**	0.60	0.43	13.5	**22.8**	22.8	11.4
	0.5	0.14	**0.58**	0.58	0.37	11.6	**22.3**	22.3	11.4
	0.6	0.10	**0.56**	0.56	0.31	10.0	**21.8**	21.8	11.3
	0.7	0.08	**0.54**	0.54	0.26	8.7	**21.5**	21.5	11.2
	0.8	0.07	**0.52**	0.52	0.22	7.5	**21.2**	21.2	11.1
	0.9	0.05	**0.51**	0.51	0.21	6.5	**20.9**	20.9	11.1
	1.0	0.04	**0.50**	0.50	0.23	5.6	**20.6**	20.6	11.2
boat	0.1	0.57	**0.82**	0.82	0.51	25.3	**30.4**	30.4	11.2
	0.2	0.31	**0.72**	0.72	0.51	19.3	**27.4**	27.4	11.2
	0.3	0.19	**0.66**	0.66	0.49	15.8	**25.8**	25.8	11.2
	0.4	0.13	**0.62**	0.62	0.43	13.3	**24.7**	24.7	11.2
	0.5	0.10	**0.59**	0.59	0.35	11.4	**23.9**	23.9	11.2
	0.6	0.07	**0.57**	0.57	0.29	9.8	**23.3**	23.3	11.1
	0.7	0.06	**0.55**	0.55	0.27	8.4	**22.8**	22.8	11.1
	0.8	0.04	**0.53**	0.53	0.29	7.3	**22.3**	22.3	11.1
	0.9	0.04	**0.52**	0.52	0.30	6.3	**21.9**	21.9	11.1
	1.0	0.03	**0.51**	0.51	0.31	5.3	**21.6**	21.6	11.1
fingerprint	0.1	0.84	**0.92**	0.92	0.51	24.6	**28.2**	28.2	10.4
	0.2	0.61	**0.82**	0.82	0.51	18.6	**24.5**	24.5	10.4
	0.3	0.43	**0.73**	0.73	0.50	15.0	**22.4**	22.4	10.4
	0.4	0.31	**0.66**	0.66	0.47	12.5	**21.0**	21.0	10.4
	0.5	0.23	**0.59**	0.59	0.44	10.6	**20.0**	20.0	10.3
	0.6	0.18	**0.53**	0.53	0.40	9.0	**19.2**	19.2	10.2
	0.7	0.14	**0.49**	0.49	0.38	7.7	**18.6**	18.6	10.2
	0.8	0.11	**0.46**	0.46	0.37	6.5	**18.2**	18.2	10.2

Continued on Next Page…

Table 3.4 – Continued

images	noise level	MSSIM				PSNR			
		noisy	BoxMG	AMG	AOS-R	noisy	BoxMG	AMG	AOS-R
	0.9	0.09	**0.43**	0.43	0.35	5.5	**17.9**	17.9	10.2
	1.0	0.08	**0.40**	0.40	0.33	4.6	**17.6**	17.6	10.1

Table 3.5: Comparison of restored images using different methods: unregularized AOS, AOS-U, the traditional AOS scheme, AOS-T, where the stopping time is found by minimizing the CC measure, and TV as in (3.12). The highlighted numbers are the proposed anisotropic diffusion denoising with BoxMG solver. In terms of both MSSIM and PSNR measures, the restored results are better than AOS-U and AOS-T methods, and performs slightly worse than TV denoising. However, the regularization parameters used in TV denoising is chosen by our Brent-NCP algorithm.

images	noise level	MSSIM					PSNR				
		noisy	BoxMG	AOS-U	AOS-T	TV	noisy	BoxMG	AOS-U	AOS-T	TV
cameraman	0.1	0.52	**0.84**	0.82	0.73	0.88	25.5	**29.8**	28.2	29.0	30.5
	0.2	0.31	**0.77**	0.73	0.69	0.81	19.5	**26.7**	25.1	25.7	27.1
	0.3	0.21	**0.72**	0.67	0.62	0.76	16.0	**24.9**	23.8	24.1	25.2
	0.4	0.16	**0.68**	0.64	0.57	0.73	13.5	**23.6**	22.7	23.1	24.1
	0.5	0.12	**0.65**	0.61	0.53	0.71	11.5	**22.6**	22.0	22.3	23.1
	0.6	0.10	**0.63**	0.58	0.50	0.69	10.0	**21.9**	21.5	21.7	22.3
	0.7	0.08	**0.61**	0.58	0.47	0.67	8.6	**21.2**	20.9	21.2	21.7
	0.8	0.06	**0.59**	0.57	0.45	0.65	7.5	**20.7**	20.5	20.7	21.2
	0.9	0.05	**0.58**	0.56	0.44	0.64	6.4	**20.3**	20.2	20.3	20.9
	1.0	0.04	**0.57**	0.54	0.42	0.63	5.5	**20.0**	19.9	20.0	20.4
house	0.1	0.44	**0.84**	0.83	0.68	0.85	24.3	**31.5**	30.6	29.2	32.1
	0.2	0.23	**0.78**	0.76	0.67	0.80	18.3	**28.2**	27.6	27.7	29.0
	0.3	0.14	**0.74**	0.72	0.60	0.77	14.8	**26.3**	26.0	25.9	27.1
	0.4	0.10	**0.71**	0.69	0.60	0.75	12.3	**25.0**	24.9	25.1	25.8
	0.5	0.07	**0.69**	0.67	0.56	0.73	10.3	**24.1**	24.0	24.1	24.8
	0.6	0.05	**0.67**	0.66	0.53	0.71	8.8	**23.3**	23.4	23.3	23.8
	0.7	0.04	**0.65**	0.64	0.51	0.69	7.4	**22.6**	22.8	22.6	23.1
	0.8	0.03	**0.64**	0.63	0.51	0.68	6.3	**22.1**	22.2	22.3	22.5
	0.9	0.03	**0.63**	0.62	0.53	0.67	5.2	**21.6**	21.7	22.1	22.0
	1.0	0.02	**0.63**	0.62	0.53	0.66	4.3	**21.1**	21.3	21.7	21.6

Continued on Next Page…

Table 3.5 – Continued

images	noise level	MSSIM					PSNR				
		noisy	BoxMG	AOS-U	AOS-T	TV	noisy	BoxMG	AOS-U	AOS-T	TV
barbara	0.1	0.62	**0.84**	0.74	0.77	0.84	25.6	**28.7**	27.8	26.1	29.1
	0.2	0.38	**0.71**	0.66	0.66	0.72	19.6	**24.9**	24.2	24.3	25.2
	0.3	0.26	**0.64**	0.61	0.59	0.64	16.0	**23.5**	23.1	23.3	23.7
	0.4	0.19	**0.60**	0.58	0.54	0.61	13.5	**22.8**	22.5	22.6	22.8
	0.5	0.14	**0.58**	0.56	0.51	0.58	11.6	**22.3**	22.1	22.1	22.3
	0.6	0.10	**0.56**	0.55	0.48	0.56	10.0	**21.8**	21.8	21.6	21.9
	0.7	0.08	**0.54**	0.54	0.44	0.54	8.7	**21.5**	21.5	21.0	21.6
	0.8	0.07	**0.52**	0.53	0.42	0.53	7.5	**21.2**	21.3	20.7	21.3
	0.9	0.05	**0.51**	0.52	0.41	0.52	6.5	**20.9**	21.0	20.4	21.0
	1.0	0.04	**0.50**	0.51	0.42	0.51	5.6	**20.6**	20.8	20.3	20.8
boat	0.1	0.57	**0.82**	0.80	0.81	0.82	25.3	**30.4**	29.1	30.0	30.8
	0.2	0.31	**0.72**	0.71	0.70	0.73	19.3	**27.4**	26.7	27.2	27.7
	0.3	0.19	**0.66**	0.66	0.65	0.67	15.8	**25.8**	25.4	25.7	26.0
	0.4	0.13	**0.62**	0.61	0.60	0.63	13.3	**24.7**	24.3	24.7	24.9
	0.5	0.10	**0.59**	0.59	0.56	0.60	11.4	**23.9**	23.8	23.9	24.1
	0.6	0.07	**0.57**	0.56	0.54	0.57	9.8	**23.3**	23.2	23.3	23.5
	0.7	0.06	**0.55**	0.55	0.51	0.56	8.4	**22.8**	22.7	22.8	23.0
	0.8	0.04	**0.53**	0.53	0.50	0.54	7.3	**22.3**	22.4	22.4	22.5
	0.9	0.04	**0.52**	0.52	0.49	0.53	6.3	**21.9**	22.1	22.1	22.2
	1.0	0.03	**0.51**	0.51	0.48	0.52	5.3	**21.6**	21.8	21.8	21.9
fingerprint	0.1	0.84	**0.92**	0.93	0.17	0.92	24.6	**28.2**	28.8	16.5	28.2
	0.2	0.61	**0.82**	0.85	0.17	0.83	18.6	**24.5**	25.1	16.4	24.5
	0.3	0.43	**0.73**	0.77	0.17	0.75	15.0	**22.4**	23.0	16.4	22.4
	0.4	0.31	**0.66**	0.70	0.17	0.68	12.5	**21.0**	21.5	16.4	21.1
	0.5	0.23	**0.59**	0.64	0.17	0.60	10.6	**20.0**	20.6	16.4	20.0
	0.6	0.18	**0.53**	0.61	0.17	0.55	9.0	**19.2**	20.0	16.3	19.3

Continued on Next Page...

Table 3.5 – Continued

images	noise level	MSSIM					PSNR				
		noisy	BoxMG	AOS-U	AOS-T	TV	noisy	BoxMG	AOS-U	AOS-T	TV
	0.7	0.14	**0.49**	0.55	0.16	0.49	7.7	**18.6**	19.2	16.3	18.6
	0.8	0.11	**0.46**	0.49	0.16	0.42	6.5	**18.2**	18.5	16.3	18.0
	0.9	0.09	**0.43**	0.47	0.16	0.42	5.5	**17.9**	18.2	16.2	17.8
	1.0	0.08	**0.40**	0.42	0.15	0.38	4.6	**17.6**	17.7	16.1	17.4

Table 3.6: Comparison of anisotropic diffusion denoising and BM3D method. We also test the sensitivity of BM3D with respect to the input noise levels. The input noise levels are the exact noise level, n_0, an underestimated value, $0.7 \times n_0$, and an overestimated value, $1.3 \times n_0$. The highlighted numbers are the proposed anisotropic diffusion denoising with BoxMG solver. In terms of both MSSIM and PSNR measures, if known noise level ahead, the restored results by BM3D are better than the proposed method. In practice, however, it is very difficulty to determine the true noise level. If the input noise level is not accurate, the restored image qualities are degraded sharply.

images	noise level	MSSIM					PSNR				
		noisy	BoxMG	BM3D			noisy	BoxMG	BM3D		
				n_0	$0.7n_0$	$1.3n_0$			n_0	$0.7n_0$	$1.3n_0$
cameraman	0.1	0.53	**0.84**	0.91	0.83	0.88	25.5	**29.8**	32.3	30.8	31.4
	0.2	0.31	**0.77**	0.84	0.65	0.83	19.5	**26.7**	29.0	26.7	28.4
	0.3	0.22	**0.72**	0.80	0.52	0.79	16.0	**24.9**	27.1	24.1	26.2
	0.4	0.16	**0.68**	0.77	0.38	0.76	13.5	**23.6**	25.8	20.9	25.1
	0.5	0.13	**0.65**	0.74	0.40	0.74	11.5	**22.6**	24.9	21.9	24.2
	0.6	0.10	**0.63**	0.72	0.35	0.71	10.0	**21.9**	24.0	20.9	23.5
	0.7	0.08	**0.61**	0.69	0.32	0.69	8.6	**21.2**	23.3	20.0	22.8
	0.8	0.07	**0.59**	0.67	0.28	0.67	7.5	**20.7**	22.7	19.3	22.3
	0.9	0.06	**0.58**	0.65	0.26	0.65	6.4	**20.3**	22.2	18.6	21.8
	1.0	0.05	**0.57**	0.62	0.22	0.63	5.5	**20.0**	21.6	17.5	21.2
house	0.1	0.45	**0.84**	0.88	0.79	0.87	24.3	**31.5**	34.6	31.5	34.1
	0.2	0.23	**0.78**	0.84	0.59	0.83	18.3	**28.2**	31.7	27.1	31.2
	0.3	0.15	**0.74**	0.81	0.42	0.81	14.8	**26.3**	29.8	23.4	29.4
	0.4	0.10	**0.71**	0.78	0.39	0.79	12.3	**25.0**	28.3	23.2	28.0
	0.5	0.07	**0.69**	0.76	0.33	0.76	10.3	**24.1**	27.1	21.7	26.7
	0.6	0.05	**0.67**	0.73	0.28	0.74	8.8	**23.3**	26.0	20.6	25.8
	0.7	0.04	**0.65**	0.70	0.25	0.72	7.4	**22.6**	25.2	19.7	25.0
	0.8	0.03	**0.64**	0.68	0.22	0.70	6.3	**22.1**	24.4	18.9	24.4

Continued on Next Page...

Table 3.6 – Continued

images	noise level	MSSIM					PSNR				
		noisy	BoxMG	BM3D			noisy	BoxMG	BM3D		
				n_0	$\mathbf{0.7}n_0$	$\mathbf{1.3}n_0$			n_0	$\mathbf{0.7}n_0$	$\mathbf{1.3}n_0$
	0.9	0.03	**0.63**	0.63	0.18	0.66	5.2	**21.6**	23.3	17.3	23.2
	1.0	0.02	**0.63**	0.53	0.14	0.56	4.3	**21.1**	21.9	15.5	22.0
barbara	0.1	0.62	**0.84**	0.93	0.86	0.92	25.6	**28.7**	33.5	30.8	33.1
	0.2	0.39	**0.71**	0.88	0.71	0.87	19.6	**24.9**	30.2	26.6	29.7
	0.3	0.26	**0.64**	0.83	0.57	0.81	16.0	**23.5**	28.2	23.7	27.6
	0.4	0.19	**0.60**	0.78	0.41	0.76	13.5	**22.8**	26.7	20.4	26.1
	0.5	0.14	**0.58**	0.74	0.45	0.72	11.6	**22.3**	25.5	21.6	25.0
	0.6	0.11	**0.56**	0.70	0.39	0.68	10.0	**21.8**	24.6	20.6	24.1
	0.7	0.08	**0.54**	0.66	0.35	0.64	8.7	**21.5**	23.8	19.7	23.3
	0.8	0.07	**0.52**	0.63	0.31	0.61	7.5	**21.2**	23.1	19.0	22.6
	0.9	0.05	**0.51**	0.59	0.28	0.58	6.5	**20.9**	22.4	18.4	22.1
	1.0	0.04	**0.50**	0.54	0.24	0.54	5.6	**20.6**	21.6	17.4	21.4
boat	0.1	0.57	**0.82**	0.86	0.80	0.84	25.3	**30.4**	32.5	30.5	31.8
	0.2	0.31	**0.72**	0.79	0.64	0.76	19.3	**27.4**	29.4	26.6	28.8
	0.3	0.20	**0.66**	0.73	0.50	0.70	15.8	**25.8**	27.5	23.9	26.9
	0.4	0.13	**0.62**	0.68	0.35	0.66	13.3	**24.7**	26.2	20.6	25.8
	0.5	0.10	**0.59**	0.65	0.37	0.64	11.4	**23.9**	25.3	21.7	25.0
	0.6	0.07	**0.57**	0.62	0.31	0.61	9.8	**23.3**	24.6	20.6	24.4
	0.7	0.06	**0.55**	0.60	0.27	0.59	8.4	**22.8**	24.0	19.8	23.8
	0.8	0.04	**0.53**	0.57	0.24	0.57	7.3	**22.3**	23.5	19.1	23.3
	0.9	0.04	**0.52**	0.55	0.21	0.56	6.3	**21.9**	23.0	18.4	22.9
	1.0	0.03	**0.51**	0.52	0.18	0.53	5.3	**21.6**	22.4	17.2	22.3
	0.1	0.85	**0.92**	0.95	0.92	0.94	24.6	**28.2**	30.3	28.1	29.8
	0.2	0.62	**0.82**	0.89	0.83	0.87	18.6	**24.5**	26.8	24.0	26.3
	0.3	0.44	**0.73**	0.84	0.71	0.82	15.0	**22.4**	25.0	20.8	24.4

Continued on Next Page…

fingerprint

Table 3.6 – Continued

images	noise level	MSSIM					PSNR				
		noisy	BoxMG	BM3D			noisy	BoxMG	BM3D		
				n_0	$\mathbf{0.7}n_0$	$\mathbf{1.3}n_0$			n_0	$\mathbf{0.7}n_0$	$\mathbf{1.3}n_0$
	0.4	0.32	**0.66**	0.80	0.70	0.77	12.5	**21.0**	23.7	20.8	23.3
	0.5	0.24	**0.59**	0.77	0.64	0.74	10.6	**20.0**	22.8	19.6	22.5
	0.6	0.18	**0.53**	0.73	0.59	0.71	9.0	**19.2**	22.0	18.7	21.8
	0.7	0.14	**0.49**	0.70	0.54	0.67	7.7	**18.6**	21.3	18.0	21.2
	0.8	0.11	**0.46**	0.66	0.50	0.63	6.5	**18.2**	20.7	17.3	20.5
	0.9	0.09	**0.43**	0.60	0.43	0.54	5.5	**17.9**	19.7	16.1	19.3
	1.0	0.08	**0.40**	0.49	0.32	0.44	4.6	**17.6**	18.3	14.4	18.0

Chapter 4

Edge-Preserving Projection-Based Deblurring

4.1 Introduction

As discussed in Section 1.1.2, the continuous spatial-invariant image deblurring model can be written as a Fredholm integral equation of the first kind,

$$\iint K(s-\nu, y-\tau)X(\nu,\tau)d\nu d\tau \approx \tilde{B}(s,y) = B(s,y) + \mathcal{E}(s,y),$$

with the corresponding discretized model in matrix-vector form

$$Ax \approx \tilde{b} = b + \eta.$$

As discussed in Chapter 1, the least squares solution will be deteriorated by the noise term on the right-hand side.

Projection-based methods employ the idea of approximating the solution within a subspace, $\mathcal{S}_k$, with smaller dimension [57, 60, 78]. Besides TSVD and Tikhonov regularization, most of the current iterative regularization algorithms for large-scale algorithms achieve this by projecting the problem onto a subspace which has similar spectral properties as the subspace spanned by singular vectors. The subspace $\mathcal{S}_k$ is very often the Krylov subspace associated with applying, for example, CGLS [46, 61, 87], MINRES [86] or GMRES [100], to the original problem. These projection methods, and in particular those based on Krylov subspaces, have proven themselves useful as efficient computational tools in many applications.

The main disadvantage of the projection methods is that the use of a low-dimensional subspace $\mathcal{S}_k$ for the regularized solution tends to inhibit the reconstruction of the sharp edges in the solution. This is because such features require a large number of high-frequency spectral components, whereas the TSVD or Krylov subspace $\mathcal{S}_k$ typically contains the low-frequency components. Unfortunately, if one simply increase the dimension, k, of $\mathcal{S}_k$ in order to include components with higher frequency, e.g., by performing more iterations of the iterative method, then more inverted noise enters the solution [58].

In this chapter, we presents a new edge-preserving projection (EPP) algorithm which is based on the powerful projection paradigm, but augmented in such a way that we include a controlled amount of the high-frequency components in the computed solution. First we use $\mathcal{S}_k$ to compute a smooth regularized solution and, then, we add components from the orthogonal complement $\mathcal{S}_k^\perp$ in a controlled way, in order to better represent the desired features in the solution. This approach is inspired by the PP-TSVD algorithm [57] for computing piecewise constant (or polynomial) solutions.

The remainder of this chapter is organized as follows. Section 4.2 presents the new edge-preserving algorithm and the convergence analysis. Section 4.3 discusses the efficient numerical implementation issues. Section 4.4 presents numerical experiments of the new deblurring algorithm and comparisons with other state-of-art deblurring algorithms.

4.2 The Projection-Based Edge-Preserving Algorithm

This section presents the main ideas of the algorithm, while the implementation details for large-scale problems are discussed in the next section.

4.2.1 Mathematical Model

Assume $W_k \in \mathbb{R}^{n\times k}$ is a matrix with orthonormal columns that span the subspace $\mathcal{S}_k$, and let W_0 be the matrix containing the orthonormal basis vectors for the complementary space $\mathcal{S}_k^\perp$. The matrix $W_k W_k^T$ is the L_2 orthogonal projection matrix associated with $\mathcal{S}_k$. The fundamental assumption here is that the columns of W_k represent "smooth" modes in which it is possible to distinguish the signal, b, from the noise, η, in (1.6). In other words, considering $W_k^T\tilde{b} = W_k^T b + W_k^T\eta$ and $W_0^T\tilde{b} = W_0^T b + W_0^T\eta$, then, the assumptions are equivalent to

$$||W_k^T b|| > ||W_k^T\eta|| \quad \text{and} \quad ||W_0^T b|| < ||W_0^T\eta||.$$

Our strategy is then to compute the solution of the following modified projection problem

$$\min_{x\in B} ||Lx||_p \quad \text{s.t.} \qquad B = \{x : \operatorname*{argmin}_y ||(AW_kW_k^T)y - b||_2\} \tag{4.1}$$

where L and p define a (semi-)norm suited for the problem, typically $1 < p < 2$.

The choice of the combination of L and p is important and, of course, somewhat problem dependent. However, if L approximates a gradient operator, and $p < 2$, then the norm $||Lx||_p$ allows us to compute regularized solutions that are less smooth than the solutions computed by $p = 2$ [57, 58]. Compared to 2-norm minimization, minimization of the 1-norm is known to be more robust to outliers in that a small number of isolated large errors do not usually change the solution. However, if $p = 1$, the solution may not be unique. A similar edge enhancing effect is also achieved with p greater than but close to 1. In this work, L is set as the discrete approximation to the first-order gradient operator. For an $m \times m$ image, denote $I \in \mathbb{R}^{m\times m}$ as the identity matrix, and define

$$L = \begin{pmatrix} L_0 \otimes I \\ I \otimes L_0 \end{pmatrix}, \quad \text{where} \quad L_0 = \begin{pmatrix} -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 1 \end{pmatrix} \in \mathbb{R}^{m\times m},$$

where $\otimes$ is the Kronecker product. Given $r \times c$ matrix P and $p \times q$ matrix Q, the Kronecker product of $P \otimes Q$ is the $rp \times cq$ matrix

$$P \otimes Q = \begin{pmatrix} P_{1,1}Q & P_{1,2}Q & \cdots & P_{1,c}Q \\ \vdots & \vdots & \ddots & \vdots \\ P_{r,1}Q & P_{r,2}Q & \cdots & P_{r,c}Q \end{pmatrix}. \tag{4.2}$$

Note that the null space of L, $\mathcal{N}(L) = \text{span}\{\mathbf{1}\}$, where all entries of $\mathbf{1} \in \mathbb{R}^n$ are 1, and $n = m^2$.

4.2.2 Uniqueness Analysis

In [44], Eldén *et al.* provide an explicit solution of (4.1) for the case of $p = 2$. The authors also prove the uniqueness condition for the minimizer. In [60], Hansen *et al.* propose an algorithm, MTSVD, for the case $p = 2$ where W_k consists of the first k singular vectors. They further develop an algorithm, PP-TSVD, for the case $p = 1$ with the same W_k [57]. In this work, we extend these results by solving (4.1) for $1 < p < 2$ and for different choices of W_k. We first present the following lemma before showing analysis of (4.1).

Lemma 4.1. *The minimization problem*

$$\operatorname*{argmin}_x \; ||Ax - b||_p, \quad p > 1, \tag{4.3}$$

has a unique minimizer, x^*, if and only if A has full column rank.

Proof. (Proof by contradiction)

Suppose there exists a unique minimizer, x^*, for the p-norm minimization problem (4.3), and A does not have full column rank. Then, for any nonzero $y \in \mathcal{N}(A)$,

$$||A(x^* + y) - b||_p = ||Ax^* - b||_p$$

Hence, $x^* + y \neq x^*$ is another minimizer. This is a contradiction to that x^* is the unique minimizer.

On the other hand, suppose A has full column rank, and there are two minimizers, $x_1 \neq x_2$ for (4.3), such that

$$||Ax_1 - b||_p = ||Ax_2 - b||_p = \beta \geq 0.$$

- If $\beta = 0$, then, $Ax_1 - b = Ax_2 - b = 0 \Rightarrow A(x_1 - x_2) = 0$. A having full column rank implies that $x_1 = x_2$. This is a contradiction of the assumption $x_1 \neq x_2$.

- If $\beta > 0$, by Minkowski's Inequality [62],

$$\begin{aligned} \left\| A\left(\frac{x_1 + x_2}{2}\right) - b \right\|_p &= \left\| \frac{1}{2}(Ax_1 - b) + \frac{1}{2}(Ax_2 - b) \right\|_p \\ &\leq \frac{1}{2}||Ax_1 - b||_p + \frac{1}{2}||Ax_2 - b||_p \\ &= \frac{1}{2}\beta + \frac{1}{2}\beta = \beta, \end{aligned} \tag{4.4}$$

  with equality if and only if $(Ax_1 - b)$ and $(Ax_2 - b)$ are linearly dependent. However, since β is the minimum of $||Ax - b||_p$ for all $x \in \mathbb{R}^n$, it is a contradiction to have

$$\left\| A\left(\frac{x_1 + x_2}{2}\right) - b \right\|_p < \beta.$$

  Therefore, we must have equality in (4.4), meaning that $(Ax_1 - b)$ and $(Ax_2 - b)$ are linearly dependent, i.e. there exists $\alpha \geq 0$ such that

$$(Ax_1 - b) = \alpha(Ax_2 - b). \tag{4.5}$$

  - If $\alpha = 1$, $Ax_1 - b = Ax_2 - b \Rightarrow A(x_1 - x_2) = 0$. A having full column rank implies that $x_1 = x_2$. This is a contradiction with assumption $x_1 \neq x_2$.
  - If $\alpha \neq 1$, (4.5) implies

$$A(x_1 - \alpha x_2) - (1 - \alpha)b = 0.$$

Hence,

$$\left\| A\left(\frac{x_1-\alpha x_2}{1-\alpha}\right) - b \right\|_p = 0 < \beta,$$

which means there exists $\left(\frac{x_1-\alpha x_2}{1-\alpha}\right) \in \mathbb{R}^n$ such that $\left\| A\left(\frac{x_1-\alpha x_2}{1-\alpha}\right) - b \right\|_p = 0 < \beta$. This contradicts that with β is the minimum of (4.3).

□

The following theorem shows the existence and uniqueness of the solution of (4.1).

Theorem 4.2. *The minimization problem*

$$\min_{x\in B} ||Lx||_p \quad s.t. \quad B = \{x : \operatorname*{argmin}_y ||(AW_kW_k^T)y - b||_2\}$$

has a unique minimizer, x^*, if and only if

$$\mathcal{N}(AW_kW_k^T) \cap \mathcal{N}(L) = \{0\}.$$

Proof. From [44], the constraint set in (4.1), B, can be written as

$$B = \{x : x = (AW_kW_k^T)^\dagger b + Px', \ x' \text{ arbitrary}\},$$

where $\dagger$ represents the Moore-Penrose pseudoinverse [2], $P = I - (AW_kW_k^T)^\dagger(AW_kW_k^T)$ is the projection onto $\mathcal{N}(AW_kW_k^T)$.

Let $\tilde{b} = (AW_kW_k^T)^\dagger b$. Solving the constrained minimization (4.1) is equivalent to solving the following unconstrained problem

$$\operatorname*{argmin}_{x'} ||LPx' - (-L\tilde{b})||_p.$$

By Lemma 4.1, the above minimization problem has a unique solution if and only if $\mathcal{N}(LP) = \{0\}$. This is true for $P = I - (AW_kW_k^T)^\dagger(AW_kW_k^T)$, the projection onto $\mathcal{N}(AW_kW_k^T)$, if and only if $\mathcal{N}(AW_kW_k^T) \cap \mathcal{N}(L) = \{0\}$. □

4.2.3 Algorithm

From the proof of Theorem 4.2, we can solve the constrained minimization problem (4.1) by a two-step algorithm. We suppose the restored image contains the smooth components in $\mathcal{S}_k$, represented by W_kx_{lf}, and the edge-correction components in the orthogonal complement $\mathcal{S}_k^\perp$, represented by W_0x_{hf}, so that the image, x, is represented as

$$x = W_kx_{\text{lf}} + W_0x_{\text{hf}}.$$

Further, x_{lf} and x_{hf} can be computed independently as shown in Algorithm 3.

Algorithm 3 Edge-Preserving Projection Algorithm

```
1: Compute the smooth component, $W_k x_{\text{lf}}$, using the 2-norm minimization problem
       $x_{\text{lf}} = \underset{x}{\operatorname{argmin}} ||(AW_k)x - b||_2,$    (4.6)
2: Compute the correction component, $W_0 x_{hf}$, using the $p$-norm minimization problem
       $x_{\text{hf}} = \underset{x}{\operatorname{argmin}} ||(LW_0)x - (-LW_k x_{\text{lf}})||_p.$    (4.7)
3: The restored solution is
       $x = W_k x_{\text{lf}} + W_0 x_{\text{hf}}.$
```

4.2.4 Choosing Projection Spaces

From Lemma 4.1, a sufficient condition for the uniqueness of x is that both AW_k and LW_0 have full column rank. Hence, (4.6) and (4.7) in Algorithm 3 have unique solutions x_k and x_0, correspondingly. Theoretically, we can choose any subspace $\mathcal{S}_k$ and its orthogonal compliment $\mathcal{S}_k^{\perp}$ with corresponding W_k and W_0. In practice, however, in order to get an efficient numerical implementation, we have to choose suitable basis vectors for $\mathcal{S}_k$ and its orthogonal complement $\mathcal{S}_k^{\perp}$ with the following requirements:

- $||W_k^T b|| > ||W_k^T \eta||$ and $||W_0^T b|| < ||W_0^T \eta||$;
- AW_k and LW_0 have full column rank;
- there exist efficient numerical algorithms to compute the matrix-vector multiplications for A, W_k, W_0 and their transpose.

4.2.4.1 Singular Vectors

In [57, 60], Hansen *et al.* propose MTSVD and PP-TSVD algorithms using singular vectors as the basis vectors. Therefore, it comes naturally to use singular vectors as the basis vectors for $\mathcal{S}_k$ and $\mathcal{S}_k^{\perp}$.

Theorem 4.3. *Let L be the discrete approximation to the gradient operator, A be an $n \times n$ blurring matrix, and assume $W_k = [v_1, v_2, \cdots, v_k]$ where v_i are the first k right singular vectors of A with corresponding nonzero singular values $\sigma_i \neq 0$. Then the minimization problem*

$$\min_{x \in B} ||Lx||_p, \quad s.t. \quad B = \{x : \underset{y}{\operatorname{argmin}} ||(AW_k W_k^T)y - b||_2\}$$

has a unique solution if $\mathbf{1} \notin \mathcal{S}_k^{\perp}$.

Proof. $W_k W_k^T \mathbf{1}$ is the orthogonal projection of $\mathbf{1}$ onto subspace $\mathcal{S}_k$, and since $\mathbf{1} \notin \mathcal{S}_k^{\perp}$, there exists $\{\alpha_1, \alpha_2, \cdots, \alpha_k\}$ which is not all zero, such that

$$W_k W_k^T \mathbf{1} = \sum_{i=1}^{k} \alpha_i v_i.$$

Hence,

$$AW_k W_k^T \mathbf{1} = A\left(\sum_{i=1}^{k} \alpha_i v_i\right) = \sum_{i=1}^{k} \alpha_i \sigma_i v_i.$$

$$||AW_k W_k^T \mathbf{1}||_2^2 = \sum_{i=1}^{k} \alpha_i^2 \sigma_i^2 > 0 \quad \Rightarrow \quad AW_k W_k^T \mathbf{1} \neq 0.$$

Therefore, $\mathbf{1} \notin \mathcal{N}(AW_k W_k^T)$, and $\mathcal{N}(AW_k W_k^T) \cap \mathcal{N}(L) = \{0\}$ because $\mathcal{N}(L) = \text{span}\{\mathbf{1}\}$. $\square$

Note that for many blurring operators, A, $\mathcal{S}_k$ contains low-frequency components, and $\mathcal{S}_k^{\perp}$ contains relatively high-frequency components. It is therefore likely that the projection of $\mathbf{1}$ onto $\mathcal{S}_k$ is not zero.

4.2.4.2 Discrete Cosine Transform

Another suitable set of basis vectors for this approach are those associated with spectral transforms such as the discrete sine or cosine transforms (DST or DCT) and their multi-dimensional extensions [55, 58]. Recall for 1D m-by-1 signals, the DCT matrix is defined as

$$C_{ij}^{1d} = \begin{cases} \frac{1}{\sqrt{m}} & \text{if } i = 0 \\ \sqrt{\frac{2}{m}} \cos(\frac{(2j+1)i\pi}{2m}) & \text{if } i > 0 \end{cases} \qquad \text{for i, j} = 0, 1, 2, \cdots, m-1$$

The rows of the above matrix are orthonormal. The 2-dimensional DCT matrix is the Kronecker product of the above matrix [76], $C = C^{1d} \otimes C^{1d}$ with $\otimes$ defined as in Equation (4.2). These basis vectors, which are the rows of the DCT matrix, have the desired spectral properties. The multiplications with W_k, W_0, and their transposes are equivalent to computing either a fast transform or its inverse, which can be implemented efficiently [58].

Theorem 4.4. *Let L be the discrete approximation to the gradient operator, $A \in \mathbb{R}^{n \times n}$ be an blurring matrix, $W_k = [w_1, w_2, \cdots, w_k]$ where w_i are rows of the 2-D DCT matrix, transposed. Then the minimization problem*

$$\min_{x \in B} ||Lx||_p, \quad s.t. \quad B = \{x : \underset{y}{\text{argmin}} \; ||(AW_k W_k^T)y - b||_2\}$$

has a unique solution if and only if $\mathbf{1} \notin \mathcal{N}(A)$.

Proof. If $\mathbf{1} \notin \mathcal{N}(A)$, from the definition of DCT matrix and W_k, we have $AW_kW_k^T\mathbf{1} = A\mathbf{1} \neq 0$. Because $\mathcal{N}(L) = \text{span}\{\mathbf{1}\}$, it is clear that $\mathcal{N}(AW_kW_k^T) \cap \mathcal{N}(L) = \{0\}$. From Theorem 4.2, Equation (4.1) has a unique solution.

On the other hand, if Equation (4.1) has a unique solution, from Theorem 4.2, $\{\mathbf{1}\} \notin \mathcal{N}(AW_kW_k^T)$, which implies $AW_kW_k^T\mathbf{1} = A\mathbf{1} \neq 0$, i.e. $\mathbf{1} \notin \mathcal{N}(A)$. □

4.3 Computational Issues and Numerical Implementations

While the above analysis guarantees the existence and uniqueness of the solution to Equation (4.1), it is critical to develop efficient numerical implementation for large-scale problems, which must take the following three issues into account:

- choose the optimal dimension of the smooth subspace, $\mathcal{S}_k$;
- choose suitable basis vectors for $\mathcal{S}_k$ and $\mathcal{S}_k^{\perp}$;
- solve the p-norm minimization problem (4.7) efficiently.

The optimal subspace dimension, k, can be computed with the methods mentioned in Chapter 1. In our experiments in Section 4.4, we use the GCV method to select a suitable k. The reason for this choice of GCV method is explained later in Section 4.4.

4.3.1 Choosing Projection Spaces

As discussed in previous section, singular vectors and the 2-D DCT matrix can be used as the basis vectors for $\mathcal{S}_k$ and $\mathcal{S}_k^{\perp}$. In this section, we will address numerical implementation issues with these choices.

4.3.1.1 Kronecker Product of Singular Vectors

For large-scale deblurring problems, it is impossible to get $W_k = [v_1, \cdots, v_k]$ by computing the SVD of the blurring matrix A without utilizing its structure. Fortunately, in most realistic problems, the point spread function in Equation (1.6) is separable, or can be approximated by a separable one [58, 66, 76, 81]. Hence, the blurring matrix A can be represented as a Kronecker product defined as in Equation (4.2),

$$A \approx A_1 \otimes A_2.$$

Given the SVDs of the two matrices A_1 and A_2,

$$A_1 = U_1S_1V_1^T, \quad A_2 = U_2S_2V_2^T,$$

the SVD of the matrix A is (approximately)

$$A \approx USV^T = ((U_1 \otimes U_2)\Pi)(\Pi^T(S_1 \otimes S_2)\Pi)((V_1 \otimes V_2)\Pi)^T, \tag{4.8}$$

where the permutation matrix Π ensures that the diagonal elements of $\Pi^T(S_1 \otimes S_2)\Pi$ appear in descending order. Then, the first k columns in $(V_1 \otimes V_2)\Pi$ consist of the basis vectors v_i in $\mathcal{S}_k$, while $\mathcal{S}_k^\perp$ contains the rest of the singular vectors.

If the Kronecker product approximation is reasonably accurate, the solution of Equation (4.6), x_{lf}, can be approximated using the explicit formula

$$x_{\text{lf}} \approx \sum_{i=1}^{k} \frac{u_i^T \tilde{b}}{s_i} v_i, \tag{4.9}$$

where the singular vectors are those in the approximation. However, if the Kronecker product approximation is not accurate, we can solve (4.6) with W_k coming from the approximate iteratively. In our experiments, we use the above approximation (4.9) as x_{lf} in (4.6). Future research includes replacing the approximation by an iterative algorithm to solve Equation (4.6) in the case of singular vectors as basis in W_k.

4.3.1.2 Discrete Cosine Transform

For the DCT basis, there is no such explicit solution formula for Equation (4.6). However, the DCT of a $m \times m$ matrix can be implemented in an very efficient way using an FFT algorithm that costs $O(m \log(m))$ operations. The matrix-vector multiplications with W_k and its transpose are equivalent to computing either the DCT or its inverse. Therefore, it is unnecessary to form the matrix AW_k explicitly to solve Equation (4.6). Some fast implementation techniques can be found in [58].

4.3.2 Iteratively Reweighted Least Squares and AMG Preconditioner

The key to the success of the EPP Algorithm 3 is an efficient solver for the p-norm minimization problem (4.7), where the iteratively reweighted least squares (IRLS) method [17, 72, 85, 117] is widely used. IRLS is identical to Newton's method with line search. This approach reduces the p-norm problem to the solution of a sequence of weighted least squares problems, which can be solved using standard least squares algorithms. In [85], Osborne shows that the IRLS method is convergent for $1 < p < 3$.

For convenience, we briefly summarize the IRLS algorithm for solving the following general p-norm problem,

$$\underset{x}{\operatorname{argmin}} \; ||\hat{A}x - \hat{b}||_p^p.$$

Algorithm 4 Iterative Reweighted Least Squares

```
1: $\hat{x}_0 = 0$ (starting vector)
2: for $j = 0, 1, 2, \dots$ do
3:     $r^j = \hat{A}\hat{x}^j - \hat{b}$
4:     $D_j = \text{diag}(|r^j|^{(p-2)/2})$
5:     $y^j = \underset{x}{\text{argmin}}\ ||D_j(\hat{A}x - (-r^j))||_2$ (determined iteratively)
6:     $y^j = \frac{1}{p-1}y^j$
7:     $\alpha_j = \underset{\alpha}{\text{argmin}}\ f(\hat{x}^j + \alpha y^j)$ (line search)
8:     $\hat{x}^{j+1} = \hat{x}^j + \alpha^j y^j$
9: end for
```

Denote the jth iteration vector by $\hat{x}^j$, the diagonal matrix, D_j, is determined by jth residual vector $r^j = \hat{A}\hat{x}^j - \hat{b}$

$$D_j = \text{diag}\left(\left|\hat{A}\hat{x}^j - \hat{b}\right|^{\frac{p-2}{2}}\right).$$

The Newton search direction, except for a scaling, is identical to the solution of the weighted least squares problem

$$\underset{x}{\text{argmin}}\ ||D_j(\hat{A}x - (-r^j))||_2. \tag{4.10}$$

However, as the solution, $\hat{x}^j$, gets close to the optimal solution, $\hat{x}^*$, for $1 < p < 2$, the diagonal elements in D^j increase to infinity (leading to ill-conditioning) and this tendency increases as p approach 1. Hence, the matrix $D_j\hat{A}$ in Equation (4.10) becomes increasingly ill-conditioned as the iterations converge. It is very difficulty to find a suitable preconditioner if trying to solve the least squares problem (4.10) directly.

Consider the corresponding normal equations,

$$\hat{A}^T D_j^2 \hat{A}x = -\hat{A}^T D_j^2 r^j = -\hat{A}^T D_j^2(\hat{A}\hat{x}^j - \hat{b})$$

Define $q^{j+1} = x + \hat{x}^j$. The normal equations can be rewritten as

$$\hat{A}^T D_j^2 \hat{A} q^{j+1} = \hat{A}^T D_j^2 \hat{b}. \tag{4.11}$$

The benefit of the above transformation is that the right-hand side in the new Equation (4.11) depends on iteration j only through D^j, which is known in jth iteration.[1]

Note that in Equation (4.7), $\hat{A} = LW_0$ and $\hat{b} = -LW_k x_{\text{lf}}$, so Equation (4.11) can be rewritten as

$$W_0^T(L^T D_j^2 L)W_0 q_j = -W_0^T(L^T D_j^2 L)W_k x_{\text{lf}}. \tag{4.12}$$

[1]Thanks to Eric de Sturler for pointing this out.

Since the condition number of the diagonal matrix D_j^2 increases as the algorithm converging to the optimal minimizer, preconditioning is helpful in solving Equation (4.12). Recall that L is a gradient operator. Hence, $L^T D_j^2 L$ represents a diffusion operator with large discontinuities in the diffusion coefficient. As discussed in Chapter 1, AMG methods are robust when the diffusion coefficients are discontinuous and vary widely [98, 107]. Therefore, we employ a AMG method to develop a right preconditioner, M, to solve Equation (4.12). The right preconditioned problem is

$$[W_0^T(L^T D_j^2 L)W_0 M]\tilde{q}_j = -W_0^T(L^T D_j^2 L)W_k x_{\mathrm{lf}}, \tag{4.13}$$

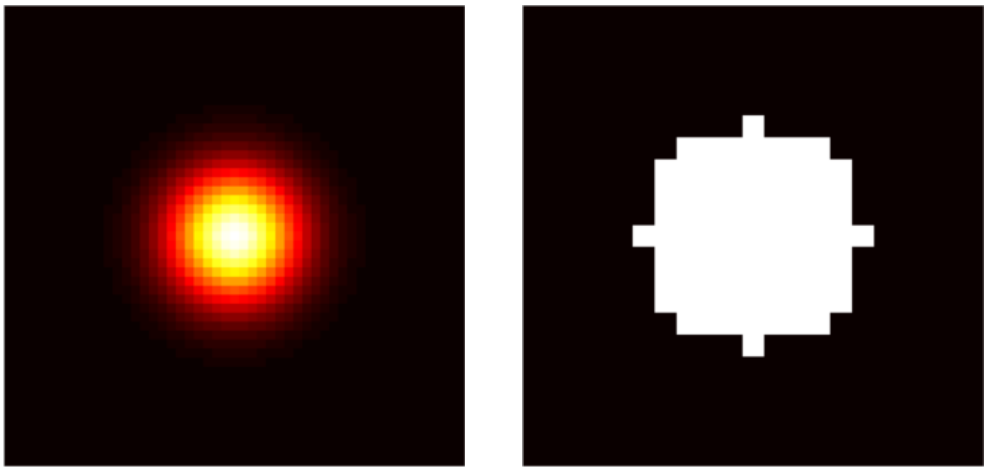
where $q_j = M\tilde{q}_j$. In our implementation, given a vector, z, the matrix-vector multiplication, Mz, is implemented in three steps:

1. Multiply by W_0, $\tilde{z} = W_0 z$
2. Use AMG method to solve $(L^T D_j^2 L)u = \tilde{z}$, get the intermediate vector, u.
3. Multiply by W_0^T, $z = W_0^T u$

Note the matrix $W_0^T(L^T D_j^2 L)W_0$ is symmetric positive definite if D_j^2 is positive definite. Otherwise, the positive definiteness of D_j^2 is guaranteed by adding a small positive number, δ, to the diagonal elements. The first thought is to solve Equation (4.13) with conjugate gradient (CG) method [61]. However, this requires the preconditioner, M, is also symmetric positive definite. In our experiments, we use Gauss-Seidel method in the pre- and post-relaxations. Hence, the AMG residual reduction operator is not symmetric [98], and, thus, the preconditioner is not symmetric. Therefore, we have to solve Equation (4.10) with GMRES algorithm with right AMG preconditioner [100]. Future research includes replacing the Gauss-Seidel iteration by red-black Gauss-Seidel in pre-relaxation and black-red Gauss-Seidel in post-relaxation. In this case, the new AMG preconditioner will be symmetric. Hence, it is possible to solve the preconditioned normal equations with the CG method.

4.4 Numerical Results

In this section, we present numerical experiments using the EPP algorithm and a comparison with total variation deblurring.



(a) Gaussian PSF, $\rho = 0$, $\sigma_1 = \sigma_2 = 5$ (b) Out-of-focus PSF, $r = 5$

Figure 4.1: Point spread functions.

4.4.1 Image Quality Measures and PSFs

The "noise level" of a test image is defined as

$$\text{noise level} = \frac{||\eta||_2}{||b||_2}.$$

The quality of restored images are measured by the MSSIM (see Chapter 3), and relative error, which is defined as

$$\text{relative error} = \frac{||x_{\text{restored}} - x||_2}{||x||_2}.$$

In the following experiments, the test images are generated with two common types of PSFs: Gaussian blur and out-of-focus blur, with reflexive boundary condition.

- Gaussian PSF

$$p_{ij} = \exp\left(-\frac{1}{2}\left[\begin{smallmatrix} i-k \\ j-l \end{smallmatrix}\right]^T \left[\begin{smallmatrix} \sigma_1^2 & \rho^2 \\ \rho^2 & \sigma_2^2 \end{smallmatrix}\right] \left[\begin{smallmatrix} i-k \\ j-l \end{smallmatrix}\right]\right),$$

  where the parameters σ_1, σ_2 and ρ determine the width and the orientation of the PSF, (k, l) is the center pixel of the PSF. If $\rho = 0$, then the horizontal and vertical components of the blur is separable. In Figure 4.1a, in this example, $\rho = 0$, $\sigma_1 = \sigma_2 = 5$.

- out-of-focus PSF

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

where (k, l) is the center pixel of PSF, and r is the radius of the blur. In Figure 4.1b, $r = 5$. The PSF is symmetric but not separable. Therefore, it is impossible to efficiently compute an accurate SVD. In the numerical experiments shown later, the use of approximate singular vectors dramatically degrades the restored image quality.

In order to compute the truncation parameter, k, we can either use Brent-NCP method in Chapter 2, or use GCV method [36]. However, GCV method can be implemented very efficiently if the singular vectors or DCT basis are known ahead of time, which is true in this problem. Therefore, it is convenient to compute the truncation parameter, k, with GCV method. The GCV functional is defined as

$$\text{GCV}(j) = \frac{\sum_{j+1}^{n}(\hat{b})^2}{(n-j)^2}, \quad \text{for} \quad j = 1, 2, \cdots, n-1,$$

where $\hat{b} = u_i\tilde{b}$ for singular vectors u_i, or $\hat{b} = \text{DCT}(b)$ for DCT basis, see the implementation details in [36]. However, as noted in [36], the GCV method always provides a parameter that is too large. Further, in our experiments, we assume the singular vectors are approximated by a Kronecker product, which might be not accurate. Hence, we choose k to be equal to 2/3 of the output from GCV algorithm, where the factor of 2/3 is chosen by experiments. Two stopping criteria, which are used in the numerical experiments, are chosen by trial-and-error. Experimental results computed with smaller tolerances are qualitatively similar to those computed with the following tolerances, but the computational time is much longer.

- Stopping criteria in IRLS Algorithm 4: 10^{-3}.
- Stopping criteria in GMRES in Step 5 of Algorithm 4: 10^{-2}.

4.4.2 Choosing Norm Parameter: p

In Algorithm 3, p can be any number between 1 and 2. For smaller p, the solution tends to have sharper edges. On the other hand, as p gets closer to 1, the p-norm minimization in Equation (4.7) becomes more ill-conditioned requiring more computational work.

Table 4.1 shows the results of the restored out-of-focus blurred images using DCT-EPP algorithm. The first column shows test images which are commonly used in testing image restoration algorithms: cameraman, clock, house, resolution. The size of these images are 256×256 pixels. The second column is the radius of blur, r, which varies from 5 to 15 pixels. The noise levels in the test images, varying from 1% to 10%, are shown in the third column. The fourth column presents the computed truncation parameter k. The relative errors and MSSIMs of the restored images of Equation (4.6), x_k, are shown in Columns 5 and 6. The relative errors and MSSIMs for the final restored images computed using different p-norms, $p = 1.01, 1.05, 1.1, 1.2$, are shown in the last 8 columns.

As shown in the table, compared to the restored quality of x_k, the final restored images, x, have larger MSSIM and smaller relative errors. This means the correction step (4.7) improves the restored image quality. This is also illustrated in Figures 4.2 and 4.3. The corrections, x_0, contain edge information. In terms of the p-norms, the restored images computed using $p = 1.01$ are better than the results using larger p.

In the case of Gaussian blur, the restored results are shown in Table 4.2 and Figures 4.4 and 4.5. As in the above discussion for the case of out-of-focus blur, the correction step Equation (4.7) in DCT-EPP algorithm with $p = 1.01$ improves the restored image quality.

For Guassian blur, we get similar conclusions when using singular vectors as the basis vectors for $\mathcal{S}_k$ and $\mathcal{S}_k^{\perp}$, see Table 4.4 and Figures 4.6 and 4.7. However, if the blur is out-of-focus blur, which is not separable, the SVD-EPP algorithm performs poorly in terms of the MSSIM measure and relative error, see Table 4.3. As discussed in previous section, if the PSF is non-separable, the singular vectors of the blurring matrix, A, are approximated by Kronecker product of two Toeplitz matrices. In the experiments, the smooth component, x_k, is approximated by the SVD solution (4.9). Compared to the separable Gaussian blur case, the smooth components, x_k, have larger relative error and smaller MSSIM measures, see Table 4.3. Therefore, the final restored images are degraded in this step. Future research includes replacing the direct approximation by solving Equation (4.6) with some iterative algorithms, such as LSQR [87], LSMR [46] etc.

4.4.3 Comparison with Total Variation Deblurring

In this section, we compare the performance of the EPP algorithm with the TV deblurring algorithm proposed in [63]. The TV method has been used to solve image restoration problems since it was introduced by Rudin, Osher, and Fatemi [97]. The TV method has the ability to preserve edges in the object image [30, 63, 97]. The objective function of TV deblurring can be written as

$$\underset{x}{\operatorname{argmin}}\ ||Ax - b||_2^2 + \lambda ||x||_{TV}, \tag{4.14}$$

where $||\cdot||_{TV}$ is the TV regularization term, $||x||_{TV} = \sum_{i=1}^{n} |\nabla x_i|_2$, and λ is an undetermined positive regularization parameter. In [63], the authors propose an iterative algorithm for Equation (4.14) by solving the following objective minimization problem

$$\underset{y}{\operatorname{argmin}}\ \left\{ \underset{x}{\operatorname{argmin}}\ ||Ax - b||_2^2 + \lambda_1 ||x - y||_2^2 \right\} + \lambda_2 ||y||_{\mathrm{TV}}$$

in two steps:

- Deblurring Step:

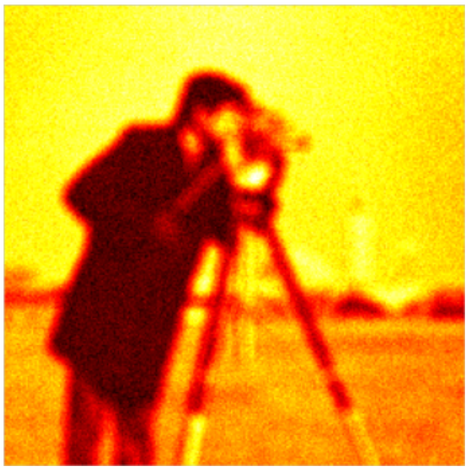
$$x_{\text{deblur}} = \underset{x}{\operatorname{argmin}}\ ||Ax - b||_2^2 + \lambda_1 ||x||_2^2.$$

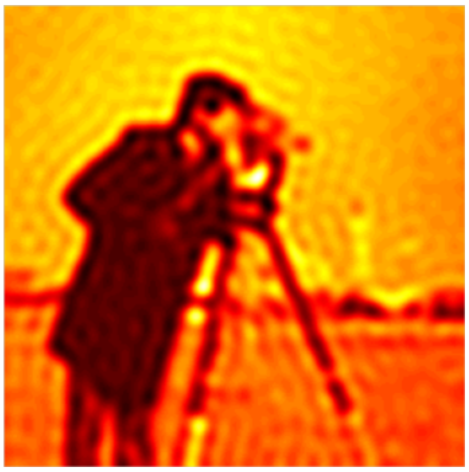
- Denoising Step:

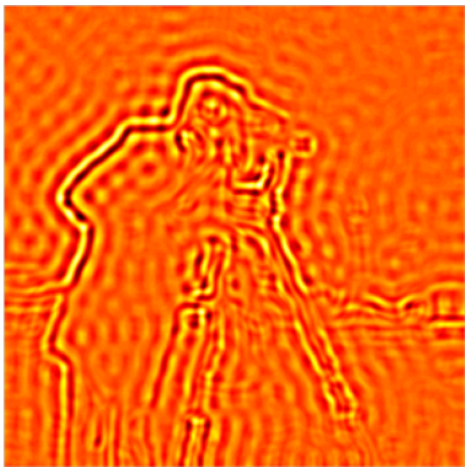
$$x_{\text{tv}} = \underset{y}{\operatorname{argmin}}\ ||x_{\text{deblur}} - y||_2^2 + \lambda_2 ||y||_{TV}.$$

One drawback of above method is that, instead of one undetermined regularization parameter, there are two, λ_1 and λ_2, to be determined. In [74], the authors use the GCV method to find the optimal λ_1 given a fixed λ_2. The regularization parameters, however, have to be updated in each iteration, which is time-consuming. In our experiments, we select λ_1 with the GCV method, and compute the solutions to the restoration problem with several different λ_2 in a reasonable range with small gaps. The optimal λ_2 is chosen according to the MSSIM measure, which requires the true images are known. In comparison, for the EPP algorithm, the only truncation parameter, k, is computed using the GCV method.

As shown in Tables 4.5 and 4.6, the restored results by the TV method qualitatively have similar image quality as those by EPP algorithm. However, the EPP algorithm outperforms the TV method in the case of large levels of blur and noise. For small levels of blur and noise, the TV method performs better in terms of MSSIM. However, as shown in the tables, in the case of small noise level, the TV results have better image quality in terms of MSSIM without the denoising step. In terms of relative error, most of the TV results have smaller relative error without the denoising step, which raises the question about the effectiveness of the denoising step.


(a) blurred-noisy image


(b) x_k


(c) x_0

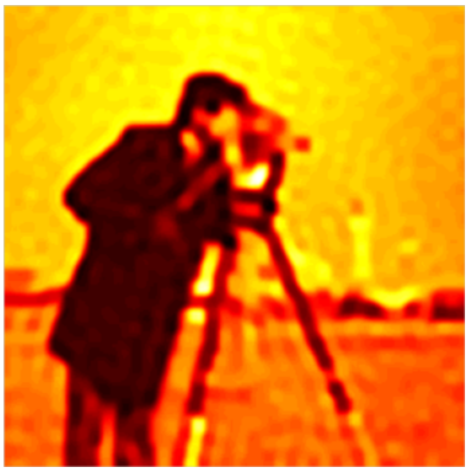
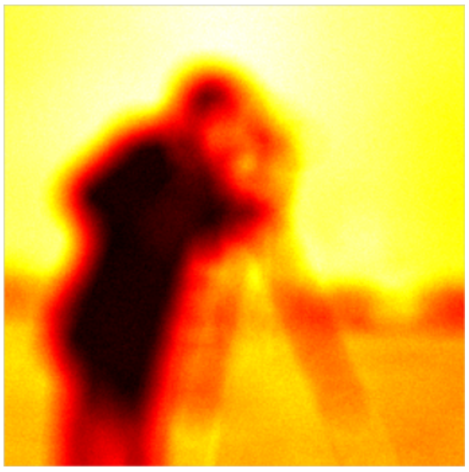
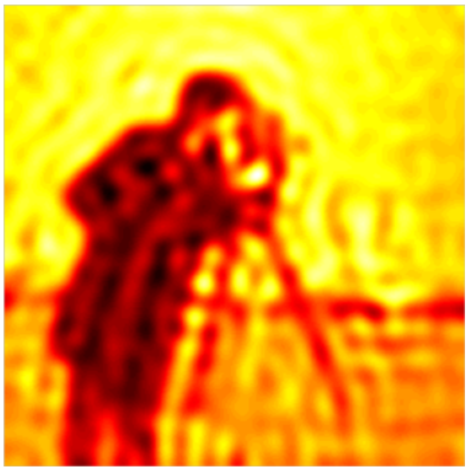
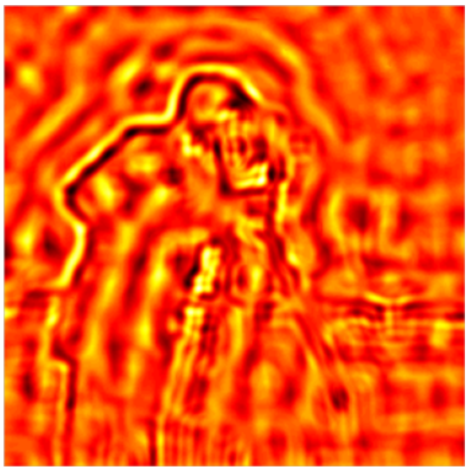

(d) x

Figure 4.2: DCT-EPP restored cameraman image: out-of-focus blur, $r = 5$, noise level is 5%. The correction (c) clearly contains edge information, and the final restored image (d) is visually better than (b).


(a) blurred-noisy image


(b) x_k


(c) x_0

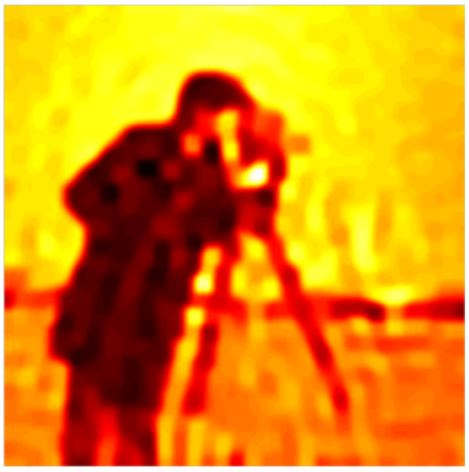

(d) x

Figure 4.3: DCT-EPP restored cameraman image: out-of-focus blur, $r = 15$, noise level is 1%. The correction (c) clearly contains edge information, and the final restored image (d) is visually better than (b).

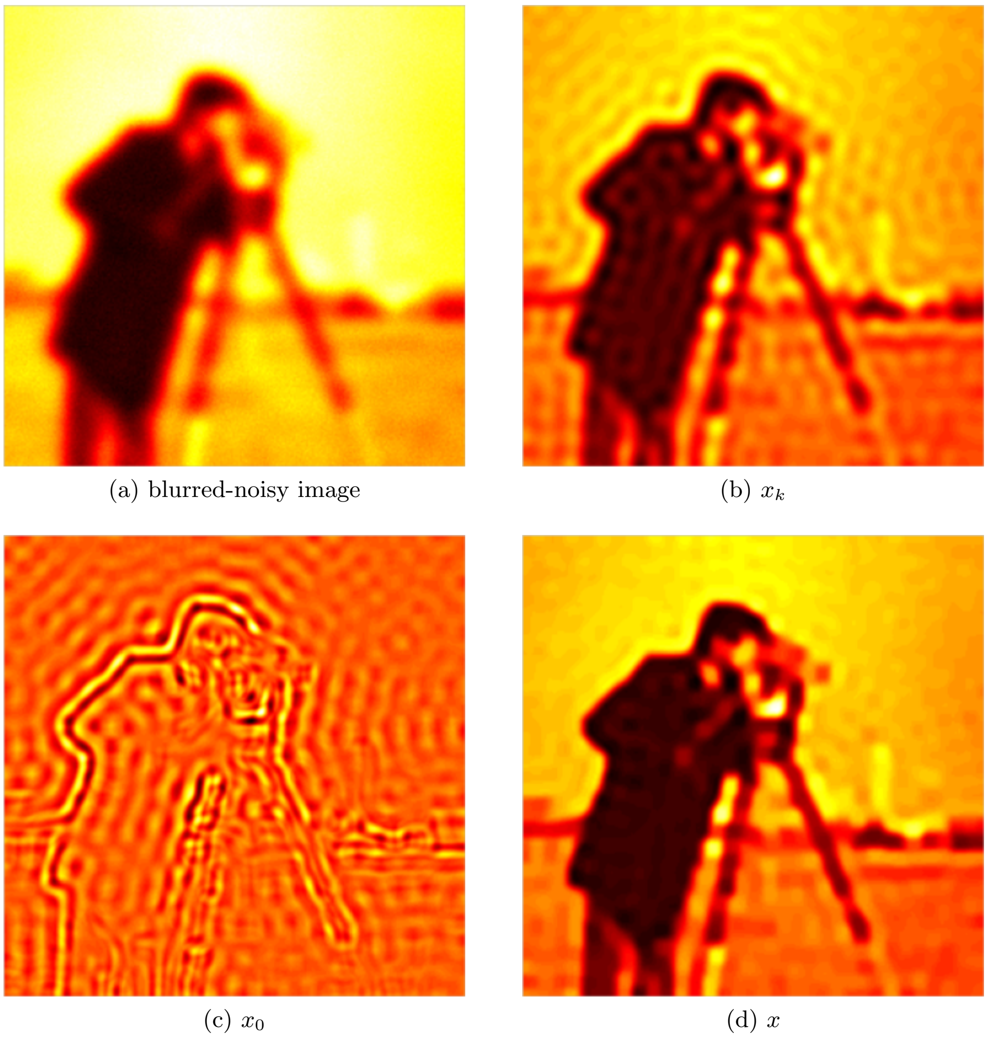
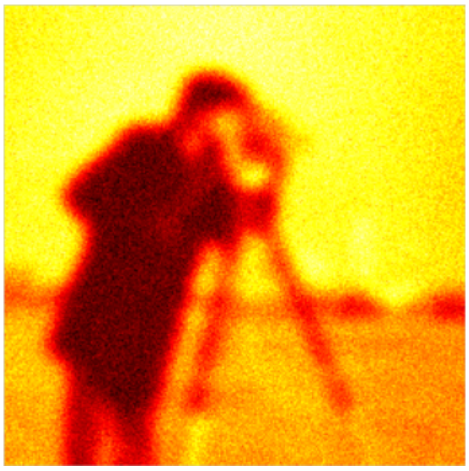
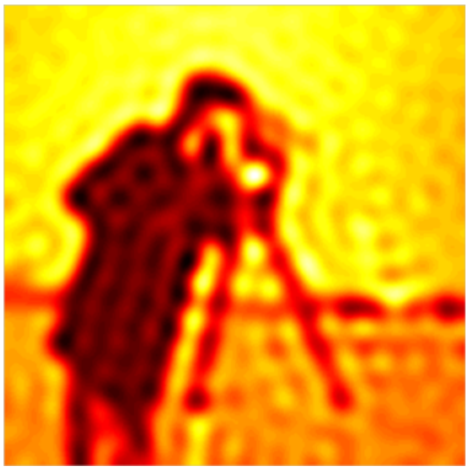
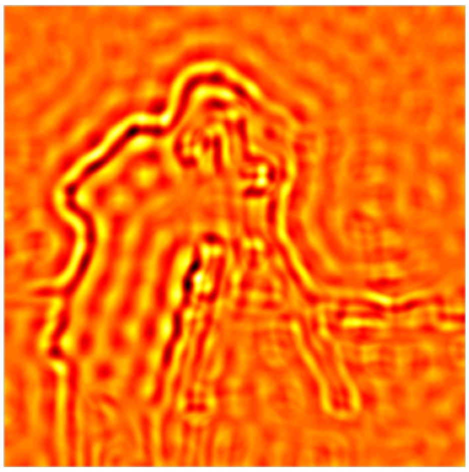

(a) blurred-noisy image (b) x_k (c) x_0 (d) x

Figure 4.4: DCT-EPP restored cameraman image: Gaussian blur, $\sigma = 5$, noise level is 1%. The correction (c) clearly contains edge information, and the final restored image (d) is visually better than (b).


(a) blurred-noisy image


(b) x_k


(c) x_0

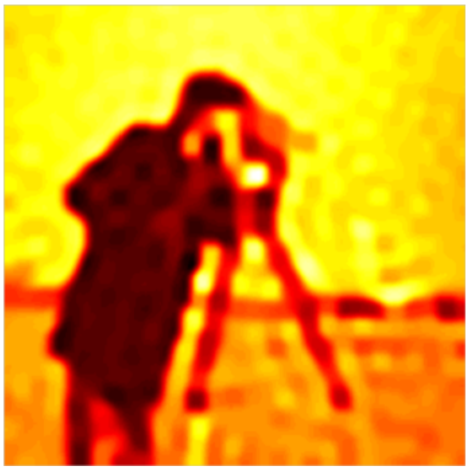
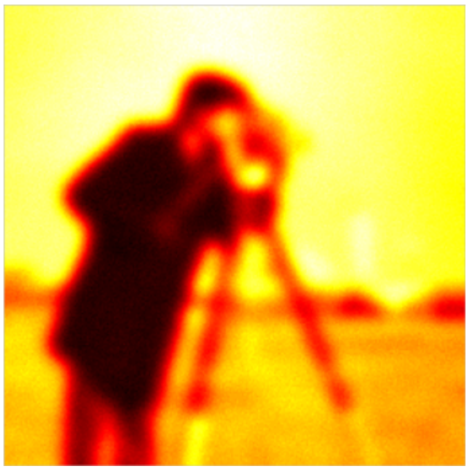
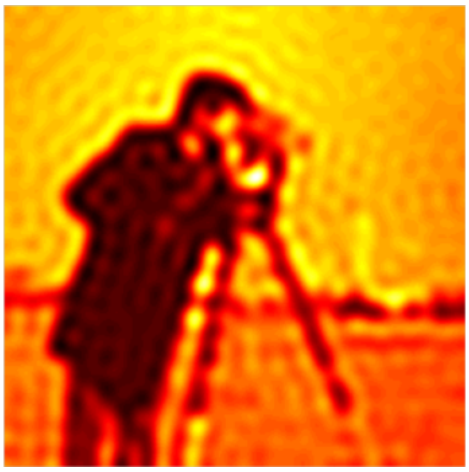
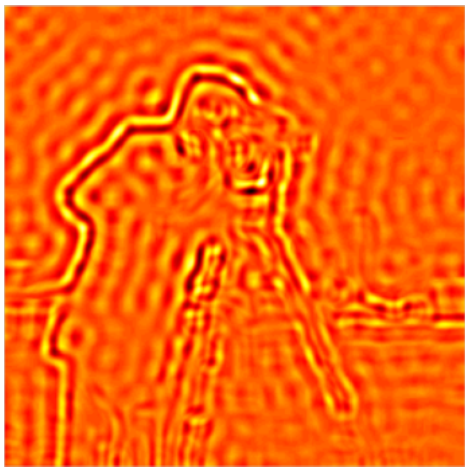

(d) x

Figure 4.5: DCT-EPP restored cameraman image: Gaussian blur, $\sigma = 5$, noise level is 5%. The correction (c) clearly contains edge information, and the final restored image (d) is visually better than (b).


(a) blurred-noisy image


(b) x_k


(c) x_0

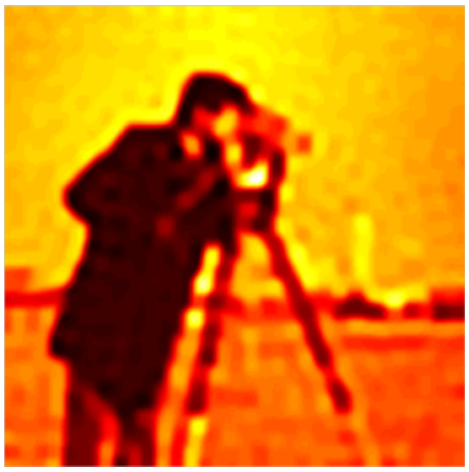
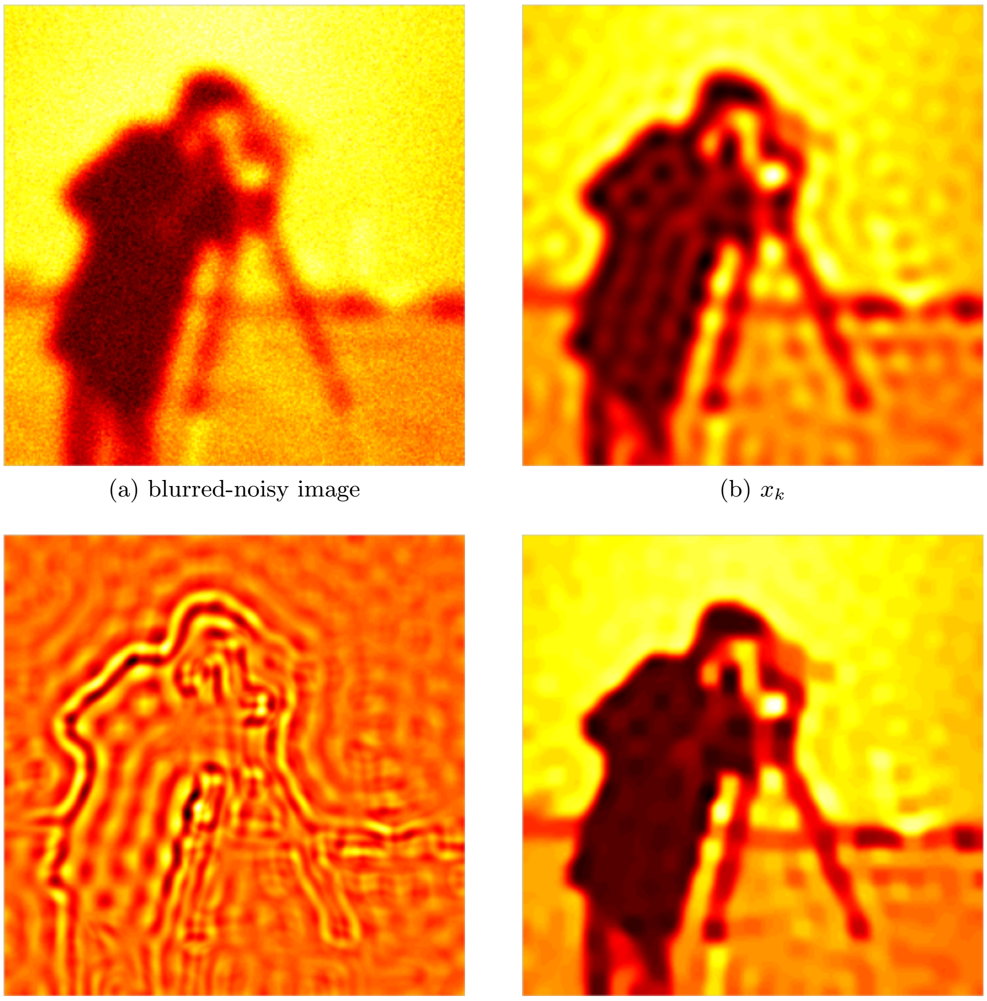

(d) x

Figure 4.6: SVD-EPP restored cameraman image: Gaussian blur, $\sigma = 5$, noise level is 1%. The correction (c) clearly contains edge information, and the final restored image (d) is visually better than (b).



(a) blurred-noisy image

(b) x_k

(c) x_0

(d) x

Figure 4.7: SVD-EPP restored cameraman image: Gaussian blur, $\sigma = 5$, noise level is 5%. The correction (c) clearly contains edge information, and the final restored image (d) is visually better than (b).

Table 4.1: Out-of-focus PSF: comparison of the restored images by DCT-EPP Algorithm 3 with different p-norms, $p = 1.01, 1.05, 1.1, 1.2$. The highlighted numbers show the chosen p-norm that gave the highest value of measure (relative error and MSSIM, respectively) for each experiment. One experiment is one row of the table. As shown in the table, as expected, $p = 1.01$ gave the best results.

images	PSF	noise level	k	x_{lf}		x							
	(r)	(%)		relative	MSSIM	relative error				MSSIM			
				error		1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
cameraman	5	1	2519	0.148	0.617	**0.138**	0.138	0.138	0.139	**0.694**	0.693	0.691	0.688
	5	5	1570	0.160	0.598	**0.153**	0.153	0.153	0.154	**0.656**	0.656	0.654	0.651
	5	10	1254	0.168	0.569	**0.161**	0.161	0.161	0.161	**0.629**	0.628	0.627	0.625
	10	1	1321	0.177	0.524	**0.162**	0.162	0.163	0.164	**0.619**	0.618	0.616	0.611
	10	5	427	0.204	0.493	**0.195**	0.196	0.196	0.197	**0.556**	0.555	0.554	0.548
	10	10	424	0.205	0.494	**0.197**	0.197	0.197	0.198	**0.553**	0.553	0.551	0.546
	15	1	774	0.195	0.491	**0.178**	0.178	0.179	0.180	**0.586**	0.585	0.581	0.575
	15	5	216	0.231	0.468	**0.224**	0.224	0.224	0.225	**0.514**	0.512	0.511	0.508
	15	10	196	0.235	0.470	**0.227**	0.227	0.227	0.228	**0.512**	0.511	0.510	0.508
clock	5	1	2191	0.081	0.740	**0.075**	0.075	0.075	0.075	**0.797**	0.796	0.795	0.793
	5	5	1463	0.094	0.695	**0.089**	0.089	0.089	0.090	**0.739**	0.738	0.737	0.735
	5	10	1152	0.103	0.666	**0.099**	0.099	0.099	0.100	**0.699**	0.699	0.698	0.697
	10	1	676	0.113	0.654	**0.109**	0.109	0.109	0.109	**0.694**	0.694	0.693	0.691
	10	5	424	0.121	0.658	**0.118**	0.118	0.118	0.118	**0.679**	0.679	0.678	0.677
	10	10	381	0.124	0.649	**0.120**	0.120	0.120	0.121	**0.670**	0.670	0.670	0.669
	15	1	772	0.113	0.640	**0.106**	0.106	0.106	0.107	**0.692**	0.691	0.690	0.688
	15	5	205	0.132	0.651	**0.130**	0.130	0.130	0.130	**0.666**	0.666	0.666	0.666
	15	10	174	0.135	0.647	**0.132**	0.132	0.132	0.132	**0.660**	0.660	0.659	0.659
	5	1	2428	0.079	0.693	**0.067**	0.067	0.067	0.068	**0.765**	0.764	0.762	0.759
	5	5	1496	0.093	0.660	**0.084**	0.085	0.085	0.085	**0.716**	0.715	0.714	0.711
	5	10	1272	0.100	0.637	**0.092**	0.092	0.092	0.093	**0.687**	0.687	0.686	0.683

Continued on Next Page...

house

Table 4.1 – Continued

images	PSF	noise level	k	x_{lf}		x							
	(r)	(%)		relative	MSSIM	relative error				MSSIM			
				error		1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
	10	1	601	0.123	0.596	**0.115**	0.115	0.116	0.116	**0.649**	0.649	0.647	0.643
	10	5	439	0.137	0.578	**0.130**	0.131	0.130	0.131	**0.609**	0.608	0.610	0.608
	10	10	385	0.141	0.575	**0.136**	0.136	0.136	0.137	**0.604**	0.603	0.603	0.600
	15	1	752	0.124	0.585	**0.110**	0.110	0.110	0.112	**0.651**	0.650	0.649	0.644
	15	5	198	0.158	0.565	**0.153**	0.153	0.153	0.154	**0.587**	0.587	0.586	0.585
	15	10	187	0.161	0.562	**0.156**	0.156	0.157	0.157	**0.580**	0.579	0.579	0.578
	5	1	2560	0.168	0.501	**0.157**	0.157	0.158	0.158	**0.660**	0.659	0.653	0.645
	5	5	1616	0.196	0.476	**0.188**	0.188	0.188	0.189	**0.601**	0.599	0.597	0.591
	5	10	1539	0.202	0.448	**0.196**	0.196	0.197	0.197	**0.536**	0.534	0.531	0.528
	10	1	810	0.226	0.403	**0.221**	0.221	0.221	0.221	**0.506**	0.505	0.502	0.500
resolution	10	5	569	0.248	0.464	**0.243**	0.243	0.243	0.243	**0.522**	0.521	0.519	0.515
	10	10	426	0.253	0.454	**0.249**	0.249	0.249	0.250	**0.503**	0.503	0.502	0.498
	15	1	810	0.235	0.400	**0.227**	0.227	0.227	0.228	**0.499**	0.498	0.495	0.490
	15	5	538	0.250	0.396	**0.245**	0.245	0.246	0.246	**0.457**	0.457	0.455	0.450
	15	10	297	0.258	0.410	**0.256**	0.256	0.256	0.256	**0.459**	0.458	0.457	0.454

Table 4.2: Gaussian PSF: comparison of the restored images by DCT-EPP Algorithm 3 with different p-norms, $p = 1.01, 1.05, 1.1, 1.2$. The highlighted numbers show the chosen p-norm that gave the highest value of measure (relative error and MSSIM, respectively) for each experiment. One experiment is one row of the table. As shown in the table, as expected, $p = 1.01$ gave the best results.

images	PSF	noise level	k	x_{lf}		x							
	(σ)	(%)		relative	MSSIM	relative error				MSSIM			
				error		1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
	5	1	1118	0.170	0.573	**0.162**	0.162	0.162	0.163	**0.642**	0.642	0.640	0.635
	5	5	739	0.185	0.529	**0.176**	0.176	0.177	0.177	**0.597**	0.596	0.594	0.589
	5	10	593	0.194	0.519	**0.186**	0.186	0.186	0.187	**0.574**	0.573	0.572	0.569
	10	1	338	0.213	0.488	**0.204**	0.205	0.205	0.205	**0.546**	0.545	0.543	0.539
cameraman	10	5	250	0.226	0.467	**0.218**	0.218	0.219	0.219	**0.512**	0.511	0.510	0.506
	10	10	178	0.239	0.480	**0.231**	0.231	0.231	0.232	**0.519**	0.518	0.517	0.514
	15	1	166	0.242	0.476	**0.234**	0.234	0.234	0.235	**0.517**	0.516	0.514	0.512
	15	5	132	0.248	0.476	**0.240**	0.240	0.240	0.241	**0.513**	0.512	0.511	0.509
	15	10	123	0.251	0.471	**0.243**	0.243	0.243	0.244	**0.513**	0.512	0.511	0.507
	5	1	1106	0.102	0.686	**0.098**	0.098	0.098	0.099	**0.728**	0.728	0.727	0.724
	5	5	646	0.115	0.664	**0.112**	0.112	0.112	0.112	**0.688**	0.688	0.688	0.686
	5	10	501	0.120	0.649	**0.116**	0.116	0.116	0.117	**0.673**	0.673	0.672	0.671
	10	1	300	0.126	0.655	**0.123**	0.123	0.123	0.123	**0.672**	0.671	0.671	0.670
clock	10	5	200	0.133	0.650	**0.130**	0.130	0.130	0.130	**0.665**	0.665	0.665	0.664
	10	10	152	0.136	0.647	**0.134**	0.134	0.134	0.134	**0.658**	0.658	0.658	0.658
	15	1	146	0.136	0.646	**0.133**	0.133	0.133	0.134	**0.658**	0.658	0.658	0.657
	15	5	109	0.140	0.643	**0.138**	0.138	0.138	0.138	**0.652**	0.652	0.652	0.651
	15	10	78	0.145	0.642	**0.142**	0.142	0.142	0.142	**0.649**	0.649	0.649	0.648
	5	1	1062	0.106	0.636	**0.097**	0.097	0.098	0.098	**0.690**	0.689	0.688	0.685
	5	5	718	0.118	0.605	**0.112**	0.112	0.112	0.113	**0.650**	0.649	0.648	0.645
	5	10	519	0.132	0.579	**0.126**	0.126	0.126	0.126	**0.614**	0.614	0.613	0.610

Continued on Next Page...

house

Table 4.2 – Continued

images	PSF	noise level	k	x_{lf}		x							
	(σ)	(%)		relative error	MSSIM	relative error				MSSIM			
						1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
	10	1	356	0.143	0.577	**0.137**	0.138	0.138	0.138	**0.603**	0.603	0.602	0.600
	10	5	236	0.154	0.565	**0.149**	0.150	0.150	0.150	**0.587**	0.587	0.586	0.585
	10	10	190	0.161	0.559	**0.156**	0.156	0.156	0.156	**0.579**	0.579	0.579	0.577
	15	1	162	0.166	0.561	**0.161**	0.161	0.162	0.162	**0.576**	0.575	0.575	0.574
	15	5	104	0.181	0.564	**0.177**	0.178	0.178	0.178	**0.574**	0.574	0.573	0.573
	15	10	90	0.184	0.566	**0.181**	0.181	0.181	0.181	**0.575**	0.575	0.575	0.574
	5	1	1350	0.208	0.465	**0.202**	0.202	0.202	0.203	**0.565**	0.564	0.561	0.552
	5	5	932	0.233	0.421	**0.229**	0.229	0.229	0.229	**0.490**	0.489	0.487	0.481
	5	10	754	0.242	0.424	**0.239**	0.239	0.239	0.239	**0.478**	0.477	0.476	0.472
	10	1	330	0.257	0.466	**0.255**	0.255	0.255	0.255	**0.502**	0.501	0.501	0.499
resolution	10	5	214	0.263	0.471	**0.261**	0.261	0.261	0.262	**0.494**	0.493	0.493	0.492
	10	10	167	0.267	0.471	**0.265**	0.265	0.265	0.265	**0.492**	0.492	0.491	0.490
	15	1	160	0.267	0.478	**0.265**	0.265	0.265	0.266	**0.506**	0.505	0.504	0.503
	15	5	118	0.274	0.478	**0.271**	0.271	0.271	0.271	**0.499**	0.499	0.498	0.497
	15	10	97	0.277	0.473	**0.273**	0.273	0.273	0.274	**0.490**	0.489	0.489	0.488

Table 4.3: Out-of-focus PSF: comparison of the restored images by SVD-EPP Algorithm 3 with different p-norms, $p = 1.01, 1.05, 1.1, 1.2$. The highlighted numbers show the computed smooth components, x_{lf}. Compared to Tables 4.1, 4.2 and 4.4, in terms of both relative error and MSSIM, the quality of x_{lf} is worse. The possible reasons are discussed in Section 4.3.1.1.

images	PSF (r)	noise level (%)	k	x_{lf}		x							
				relative error	MSSIM	relative error				MSSIM			
						1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
cameraman	5	1	5646	**0.239**	**0.449**	0.243	0.242	0.242	0.242	0.523	0.522	0.521	0.519
	5	5	2426	**0.221**	**0.458**	0.222	0.222	0.222	0.222	0.515	0.514	0.514	0.512
	5	10	1238	**0.220**	**0.492**	0.213	0.213	0.213	0.212	0.567	0.566	0.566	0.564
	10	1	2624	**0.264**	**0.357**	0.268	0.267	0.267	0.267	0.410	0.410	0.409	0.408
	10	5	832	**0.235**	**0.452**	0.235	0.235	0.234	0.234	0.509	0.508	0.507	0.505
	10	10	443	**0.235**	**0.454**	0.231	0.231	0.231	0.231	0.520	0.519	0.517	0.513
	15	1	2150	**0.320**	**0.207**	0.324	0.324	0.324	0.323	0.237	0.237	0.237	0.235
	15	5	504	**0.255**	**0.424**	0.252	0.252	0.252	0.252	0.481	0.480	0.479	0.475
	15	10	338	**0.251**	**0.427**	0.247	0.247	0.247	0.247	0.478	0.477	0.476	0.474
clock	5	1	4932	**0.135**	**0.520**	0.135	0.135	0.134	0.134	0.590	0.589	0.589	0.587
	5	5	1489	**0.131**	**0.593**	0.124	0.124	0.124	0.124	0.671	0.671	0.671	0.670
	5	10	1311	**0.134**	**0.576**	0.124	0.124	0.124	0.124	0.642	0.642	0.642	0.641
	10	1	2629	**0.155**	**0.419**	0.156	0.156	0.156	0.155	0.471	0.471	0.471	0.469
	10	5	777	**0.140**	**0.547**	0.138	0.138	0.138	0.138	0.599	0.598	0.598	0.596
	10	10	467	**0.138**	**0.601**	0.133	0.133	0.133	0.133	0.651	0.651	0.650	0.649
	15	1	2068	**0.201**	**0.238**	0.205	0.205	0.205	0.205	0.280	0.280	0.279	0.276
	15	5	410	**0.145**	**0.586**	0.143	0.143	0.143	0.143	0.623	0.623	0.623	0.622
	15	10	261	**0.145**	**0.603**	0.143	0.143	0.143	0.143	0.639	0.639	0.640	0.639
house	5	1	4865	**0.138**	**0.562**	0.138	0.138	0.138	0.138	0.621	0.621	0.620	0.619
	5	5	1794	**0.135**	**0.575**	0.131	0.131	0.131	0.131	0.632	0.633	0.632	0.631
	5	10	1198	**0.142**	**0.564**	0.130	0.130	0.130	0.130	0.627	0.627	0.626	0.625
	10	1	2491	**0.166**	**0.449**	0.166	0.166	0.166	0.165	0.503	0.503	0.502	0.499

Continued on Next Page...

Table 4.3 – Continued

images	PSF	noise level	k	x_{lf}		x							
	(r)	(%)		relative error	MSSIM	relative error				MSSIM			
						1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
	10	5	674	**0.153**	**0.529**	0.150	0.150	0.150	0.150	0.576	0.575	0.574	0.573
	10	10	467	**0.153**	**0.547**	0.148	0.148	0.149	0.149	0.590	0.589	0.589	0.587
	15	1	1888	**0.212**	**0.320**	0.215	0.215	0.215	0.214	0.359	0.359	0.358	0.356
	15	5	436	**0.164**	**0.539**	0.163	0.163	0.163	0.163	0.569	0.569	0.568	0.567
	15	10	236	**0.169**	**0.543**	0.167	0.167	0.167	0.167	0.570	0.569	0.569	0.568
resolution	5	1	6532	**0.269**	**0.289**	0.271	0.271	0.270	0.270	0.397	0.396	0.393	0.390
	5	5	2664	**0.259**	**0.282**	0.260	0.260	0.260	0.260	0.373	0.372	0.370	0.366
	5	10	1821	**0.253**	**0.331**	0.251	0.251	0.251	0.251	0.417	0.416	0.415	0.413
	10	1	3741	**0.305**	**0.178**	0.310	0.310	0.310	0.309	0.233	0.232	0.231	0.227
	10	5	1036	**0.277**	**0.275**	0.275	0.275	0.275	0.275	0.378	0.377	0.375	0.370
	10	10	868	**0.277**	**0.265**	0.278	0.278	0.278	0.278	0.351	0.350	0.347	0.345
	15	1	2542	**0.333**	**0.123**	0.339	0.338	0.338	0.337	0.165	0.164	0.163	0.160
	15	5	746	**0.283**	**0.245**	0.283	0.283	0.283	0.283	0.325	0.324	0.323	0.317
	15	10	531	**0.277**	**0.291**	0.274	0.274	0.274	0.274	0.357	0.355	0.354	0.351

Table 4.4: Gaussian PSF: comparison of the restored images by SVD-EPP Algorithm 3 with different p-norms, $p = 1.01, 1.05, 1.1, 1.2$. The PSF is Gaussian blur. The highlighted numbers show the chosen p-norm that gave the highest value of measure (relative error and MSSIM, respectively) for each experiment. One experiment is one row of the table. As shown in the table, as expected, $p = 1.01$ gave the best results.

images	PSF (σ)	noise level (%)	k	x_{lf}		x							
				relative error	MSSIM	relative error				MSSIM			
						1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
	5	1	1119	0.170	0.574	**0.162**	0.162	0.162	0.163	**0.642**	0.641	0.640	0.637
	5	5	740	0.185	0.529	**0.176**	0.176	0.176	0.177	**0.597**	0.596	0.595	0.591
	5	10	594	0.194	0.519	**0.186**	0.186	0.186	0.187	**0.574**	0.573	0.572	0.568
	10	1	339	0.213	0.491	**0.204**	0.204	0.205	0.205	**0.548**	0.547	0.546	0.542
cameraman	10	5	250	0.226	0.473	**0.219**	0.219	0.220	0.220	**0.518**	0.517	0.515	0.512
	10	10	180	0.238	0.475	**0.231**	0.231	0.231	0.232	**0.514**	0.513	0.512	0.508
	15	1	166	0.242	0.476	**0.234**	0.234	0.234	0.235	**0.517**	0.516	0.514	0.512
	15	5	134	0.248	0.480	**0.240**	0.240	0.240	0.241	**0.516**	0.515	0.514	0.511
	15	10	124	0.251	0.472	**0.243**	0.243	0.243	0.244	**0.511**	0.510	0.509	0.507
	5	1	1106	0.102	0.687	**0.098**	0.098	0.098	0.098	**0.728**	0.728	0.727	0.726
	5	5	646	0.115	0.664	**0.112**	0.112	0.112	0.112	**0.688**	0.688	0.688	0.686
	5	10	502	0.120	0.649	**0.116**	0.116	0.116	0.117	**0.673**	0.672	0.672	0.671
	10	1	300	0.126	0.655	**0.123**	0.123	0.123	0.123	**0.672**	0.671	0.671	0.670
clock	10	5	202	0.133	0.650	**0.130**	0.130	0.130	0.130	**0.665**	0.665	0.665	0.664
	10	10	146	0.137	0.646	**0.134**	0.134	0.134	0.134	**0.657**	0.657	0.657	0.656
	15	1	146	0.136	0.646	**0.133**	0.133	0.133	0.134	**0.658**	0.658	0.658	0.658
	15	5	109	0.140	0.643	**0.138**	0.138	0.138	0.138	**0.652**	0.652	0.652	0.651
	15	10	78	0.145	0.642	**0.142**	0.142	0.142	0.142	**0.649**	0.649	0.649	0.649
	5	1	1063	0.106	0.636	**0.097**	0.097	0.098	0.098	**0.690**	0.689	0.688	0.685
	5	5	719	0.118	0.606	**0.112**	0.112	0.112	0.112	**0.650**	0.650	0.649	0.646
	5	10	520	0.131	0.579	**0.126**	0.126	0.126	0.126	**0.614**	0.614	0.613	0.610

Continued on Next Page…

house

Table 4.4 – Continued

images	PSF	noise level	k	x_{lf}		x							
	(σ)	(%)		relative error	MSSIM	relative error				MSSIM			
						1.01	1.05	1.1	1.2	1.01	1.05	1.1	1.2
	10	1	356	0.143	0.576	**0.137**	0.138	0.138	0.138	**0.603**	0.602	0.602	0.600
	10	5	236	0.154	0.565	**0.149**	0.150	0.150	0.150	**0.587**	0.587	0.586	0.585
	10	10	190	0.161	0.559	**0.155**	0.155	0.156	0.156	**0.581**	0.581	0.580	0.578
	15	1	162	0.166	0.561	**0.161**	0.161	0.162	0.162	**0.576**	0.575	0.575	0.574
	15	5	115	0.177	0.563	**0.172**	0.172	0.173	0.173	**0.576**	0.576	0.576	0.575
	15	10	90	0.184	0.566	**0.181**	0.181	0.181	0.181	**0.575**	0.575	0.575	0.574
	5	1	1316	0.208	0.460	**0.202**	0.202	0.202	0.203	**0.566**	0.564	0.561	0.553
	5	5	934	0.233	0.427	**0.229**	0.229	0.229	0.229	**0.490**	0.489	0.488	0.483
	5	10	754	0.242	0.424	**0.239**	0.239	0.239	0.239	**0.478**	0.477	0.476	0.472
	10	1	329	0.257	0.466	**0.255**	0.255	0.255	0.255	**0.502**	0.501	0.500	0.498
resolution	10	5	214	0.263	0.471	**0.261**	0.261	0.261	0.262	**0.494**	0.493	0.493	0.492
	10	10	170	0.266	0.463	**0.263**	0.263	0.263	0.263	**0.491**	0.491	0.490	0.489
	15	1	161	0.267	0.479	**0.265**	0.265	0.265	0.266	**0.505**	0.505	0.504	0.503
	15	5	118	0.274	0.477	**0.271**	0.271	0.271	0.272	**0.499**	0.499	0.498	0.497
	15	10	98	0.277	0.473	**0.273**	0.273	0.273	0.274	**0.489**	0.489	0.489	0.488

Table 4.5: Out-of-focus PSF: comparison of the restored images by the TV Algorithm with different regularization parameters λ_1 and λ_2. The highlighted the numbers show the defects of the TV deblurring in Section 4.4.3, i.e. the results of first step solution $x_{\rm lf}$ are better than the final restored solution $x_{\rm tv}$. Compare to the restored results by the proposed EPP method in Tables 4.1, 4.2 and 4.4, the TV deblurring performs better in the small blurring case, for example, $r = 5$ in PSF. However, if the blurring is large, such as $r = 10, 15$, the EPP restored results in Tables 4.1, 4.2 and 4.4 are better.

images	PSF (r)	noise level (%)	λ_1	λ_2	relative error		MSSIM	
					$x_{\rm lf}$	$x_{\rm tv}$	$x_{\rm lf}$	$x_{\rm tv}$
cameraman	5	1	0.079	0.5	**0.138**	**0.150**	**0.700**	**0.677**
	5	5	0.113	0.5	**0.145**	**0.153**	0.612	0.656
	5	10	0.222	0.5	**0.154**	**0.159**	0.566	0.626
	10	1	0.047	0.5	**0.165**	**0.172**	**0.612**	**0.606**
	10	5	0.092	0.5	**0.176**	**0.180**	0.544	0.571
	10	10	0.097	0.5	0.188	0.187	0.422	0.495
	15	1	0.033	0.5	**0.179**	**0.184**	**0.563**	**0.562**
	15	5	0.069	0.5	**0.192**	**0.195**	0.501	0.520
	15	10	0.091	0.5	0.206	0.205	0.422	0.462
clock	5	1	0.089	0.5	**0.074**	**0.083**	**0.799**	**0.788**
	5	5	0.146	0.5	**0.081**	**0.086**	0.682	0.755
	5	10	0.262	0.5	**0.090**	**0.092**	0.610	0.707
	10	1	0.064	0.5	**0.097**	**0.102**	**0.732**	**0.725**
	10	5	0.097	0.5	**0.104**	**0.107**	0.636	0.676
	10	10	0.150	0.5	0.112	0.112	0.561	0.627
	15	1	0.033	0.5	**0.103**	**0.106**	0.702	0.702
	15	5	0.071	0.5	**0.113**	**0.114**	0.613	0.646
	15	10	0.150	0.5	0.122	0.122	0.583	0.617
house	5	1	0.082	0.5	**0.072**	**0.082**	**0.758**	**0.738**
	5	5	0.134	0.5	**0.081**	**0.086**	0.677	0.714
	5	10	0.214	0.5	**0.090**	**0.092**	0.612	0.684
	10	1	0.067	0.5	**0.101**	**0.107**	**0.680**	**0.669**
	10	5	0.081	0.5	**0.109**	**0.111**	0.602	0.634
	10	10	0.145	0.5	**0.119**	**0.120**	0.549	0.596
	15	1	0.035	0.5	**0.111**	**0.115**	**0.653**	**0.648**
	15	5	0.088	0.5	**0.125**	**0.127**	0.595	0.608
	15	10	0.112	0.5	0.138	0.138	0.528	0.559
resolution	5	1	0.078	0.5	**0.156**	**0.175**	**0.667**	**0.660**
	5	5	0.111	0.5	**0.165**	**0.180**	0.531	0.619
	5	10	0.122	0.5	**0.173**	**0.183**	0.392	0.548
	10	1	0.060	0.5	**0.198**	**0.209**	**0.573**	**0.569**

Continued on Next Page...

Table 4.5 – Continued

images	PSF	noise level	λ_1	λ_2	relative error		MSSIM	
	(r)	(%)			x_{lf}	x_{tv}	x_{lf}	x_{tv}
	10	5	0.069	0.5	**0.204**	**0.213**	0.466	0.517
	10	10	0.095	0.5	**0.215**	**0.221**	0.376	0.452
	15	1	0.030	0.5	**0.214**	**0.222**	0.527	0.531
	15	5	0.052	0.5	**0.225**	**0.231**	0.440	0.477
	15	10	0.064	0.5	**0.235**	**0.237**	0.337	0.397

Table 4.6: Gaussian PSF: comparison of the restored images by the TV Algorithm with different regularization parameters λ_1 and λ_2. The highlighted the numbers show the defects of the TV deblurring in Section 4.4.3, i.e. the results of first step solution x_{lf} are better than the final restored solution x_{tv}. Compare to the restored results by the proposed EPP method in Tables 4.1, 4.2 and 4.4, the TV deblurring performs better in the small blurring case, for example, $\sigma = 5$ in PSF. However, if the blurring is large, such as $\sigma = 10, 15$, the EPP restored results in Tables 4.1, 4.2 and 4.4 are better.

images	PSF	noise level	λ_1	λ_2	relative error		MSSIM	
	(σ)	(%)			x_{lf}	x_{tv}	x_{lf}	x_{tv}
	5	1	0.020	0.5	**0.165**	**0.167**	0.621	0.624
	5	5	0.076	0.5	**0.174**	**0.176**	0.577	0.587
	5	10	0.128	0.5	**0.181**	**0.182**	0.543	0.559
	10	1	0.010	0.5	**0.204**	**0.205**	0.522	0.528
cameraman	10	5	0.034	0.5	**0.212**	**0.213**	0.502	0.507
	10	10	0.090	0.5	**0.218**	**0.219**	0.490	0.496
	15	1	0.007	0.5	0.232	0.232	0.485	0.489
	15	5	0.020	0.5	0.236	0.236	0.476	0.479
	15	10	0.025	0.5	0.249	0.249	0.450	0.455
	5	1	0.021	0.5	**0.096**	**0.098**	**0.731**	**0.730**
	5	5	0.107	0.5	**0.104**	**0.106**	0.686	0.693
	5	10	0.178	0.5	0.110	0.110	0.637	0.654
	10	1	0.017	0.5	**0.122**	**0.123**	0.667	0.668
clock	10	5	0.066	0.5	0.126	0.126	0.654	0.656
	10	10	0.132	0.5	0.129	0.129	0.643	0.645
	15	1	0.012	0.5	0.133	0.133	0.655	0.655
	15	5	0.038	0.5	0.136	0.136	0.643	0.644
	15	10	0.111	0.5	0.138	0.138	0.639	0.640
	5	1	0.025	0.5	**0.098**	**0.101**	**0.678**	**0.676**
	5	5	0.083	0.5	**0.108**	**0.110**	0.635	0.641
	5	10	0.166	0.5	**0.115**	**0.116**	0.605	0.615
	10	1	0.007	0.5	**0.136**	**0.137**	0.594	0.595
house	10	5	0.041	0.5	0.145	0.145	0.577	0.579
	10	10	0.078	0.5	0.152	0.151	0.554	0.558
	15	1	0.007	0.5	0.157	0.157	0.573	0.574
	15	5	0.047	0.5	0.165	0.165	0.570	0.571
	15	10	0.075	0.5	0.171	0.171	0.558	0.560
	5	1	0.009	0.5	**0.200**	**0.206**	**0.558**	**0.566**
	5	5	0.039	0.5	**0.216**	**0.220**	0.481	0.503
	5	10	0.074	0.5	**0.226**	**0.228**	0.424	0.453
	10	1	0.011	0.5	0.254	0.254	0.488	0.492

resolution

Continued on Next Page...

Table 4.6 – Continued

images	PSF	noise level	λ_1	λ_2	relative error		MSSIM	
	(σ)	(%)			x_{lf}	x_{tv}	x_{lf}	x_{tv}
	10	5	0.055	0.5	0.257	0.257	0.475	0.479
	10	10	0.110	0.5	0.259	0.259	0.460	0.465
	15	1	0.007	0.5	0.263	0.263	0.484	0.486
	15	5	0.031	0.5	0.266	0.266	0.481	0.483
	15	10	0.057	0.5	0.268	0.268	0.478	0.480

Chapter 5

Nonnegative Least Squares Algorithm

5.1 Introduction

Numerical problems with nonnegativity constraints on solutions are pervasive throughout science, engineering and business. In order to preserve inherent characteristics of solutions corresponding to amounts and measurements, associated with, for instance, image restoration and reconstruction [26, 73, 82, 122], or chemical concentrations [13], it makes sense to respect the nonnegativity so as to avoid physically absurd and unpredictable results. This viewpoint has both computational as well as philosophical underpinnings.

In numerical linear algebra, nonnegativity constraints very often arise in least squares problems, called nonnegative least squares (NNLS), which were first presented in the book by Lawson and Hanson [72]. The first widely used NNLS algorithm was designed by Lawson and Hanson. A variation of their algorithm is available as *lsqnonneg* in Matlab. More recently, NNLS computations have been generalized to approximate nonnegative matrix or tensor factorizations [16, 69, 73, 122, 124], or to obtain low-dimensional representations of nonnegative data, see [33] for a review of NNLS problems and algorithms. Similar multiplicative iteration of this form is shown in [19] and its parallel application for model predictive control in [20].

In this section, we present a new iterative NNLS algorithm along with its convergence analysis. At each iteration of our algorithm, the new value of x in (1.14) is computed by multiplying the current value by a factor that depends on the quality of the approximation in (1.14). We prove that the quality of the approximation improves monotonically, and the iteration is guaranteed to converge to a locally optimal solution.

The remainder of this chapter is organized as follows. Section 5.2 presents the new multiplicative NNLS algorithm and the convergence analysis. Section 5.3 explores the applications of the algorithm in two image processing problems, image super-resolution and color image labelling.

5.2 Multiplicative Iteration and Its Convergence Analysis

5.2.1 Multiplicative Iteration

Recall the NNLS problem

$$\operatorname*{argmin}_x F_1(x) = \operatorname*{argmin}_x ||Ax - b||_2^2 \quad s.t. \quad x \geq 0, \tag{5.1}$$

is equivalent to minimizing the following nonnegative constrained quadratic programming problem:

$$\operatorname*{argmin}_x F(x) = \operatorname*{argmin}_x \frac{1}{2} x^T Q x - x^T h \quad s.t. \quad x \geq 0, \tag{5.2}$$

where $Q = A^T A \in \mathcal{R}^{n \times n}$ is positive semi-definite, $h = A^T b \in \mathbb{R}^n$. In this analysis, we assume there is no zero column in A. Hence, the diagonal elements in Q are all positive. Note that $F(x)$ is bounded from below, i.e. $F(x) \geq -b^T b$.

Our proposed multiplicative update for solving (5.2) is

$$x_i \leftarrow x_i \left[\frac{2(Q^- x)_i + h_i^+ + \delta}{(|Q|x)_i + h_i^- + \delta} \right], \tag{5.3}$$

where $Q^+ = \max(Q, 0)$, $Q^- = Q^+ - Q$, $|Q| = \text{abs}(Q) = Q^+ + Q^-$, $h^+ = \max(h, 0)$, $h^- = h^+ - h$, "max" and "abs" are element-wise comparison, and constant $0 < \delta \ll 1$ keeps the iteration monotonically convergent. Notice the multiplicative factor,

$$\frac{2(Q^- x)_i + h_i^+ + \delta}{(|Q|x)_i + h_i^- + \delta},$$

is always nonnegative. Hence, given a nonnegative initial guess, the updates obey the nonnegativity constraints. In terms of computational wise, the algorithm involves only two matrix-vector multiplications in each iteration.

Because

$$\begin{aligned}
x_i^{k+1} - x_i^k &= \left[\frac{2(Q^- x)_i + h_i^+ + \delta}{(|Q|x)_i + h_i^- + \delta}\right] x_i^k - x_i^k \\
&= \left[\frac{2(Q^- x)_i + h_i^+ - (|Q|x)_i - h_i^-}{(|Q|x)_i + h_i^- + \delta}\right] x_i^k \\
&= -\left[\frac{(Qx)_i - h_i}{(|Q|x)_i - h_i^- + \delta}\right] x_i^k \\
&= -\left[\frac{x_i^k}{(|Q|x)_i - h_i^- + \delta}\right] ((Qx)_i - h_i) \\
&= -\gamma_k \nabla(F(x^k)),
\end{aligned}$$

where the step-size $\gamma_k = \left[\frac{x_i^k}{(|Q|x)_i - h_i^- + \delta}\right]$, and $\nabla(F(x)) = Qx - h$. Therefore, our multiplicative update is an element-wise iterative gradient descent method.

Remark: If both Q and h contain nonnegative elements only, the multiplicative update (5.3) reduce to

$$x_i \leftarrow x_i \left[\frac{h_i}{(Qx)_i}\right]. \tag{5.4}$$

The reduced iteration (5.4) is called the image space reconstruction algorithm (ISRA) [38]. The convergence analysis of ISRA can be found in the literature [42, 92]. Recently, the theoretical upper limit of the convergence rate of ISRA has been shown in [5, 50]. The applications of ISRA can be found in computed tomography [38], image deblurring [14], image super-resolution [93]. In [73], Lee and Seung generalize the idea of ISRA to the problem of non-negative matrix factorization (NMF).

Remark: Other similar research can be found in the literature [101, 102]. Instead of requiring Q to have nonnegative elements, the authors propose a multiplicative algorithm for symmetric positive definite matrix Q and positive vector h.

In [19] and [20], the authors proposed similar multiplicative iteration. The advantage of our algorithm is that (5.3) guarantees the iteration converges to local minimum monotonically.

5.2.2 Convergence Analysis

In this analysis, we first show convergence analysis with the assumption that the optimal solution of (5.2), x^*, is strictly positive. The general case will be discussed later. Similarly to [73, 101, 102], this convergence analysis is based on construction of an auxiliary function for $F(x)$ in (5.2).

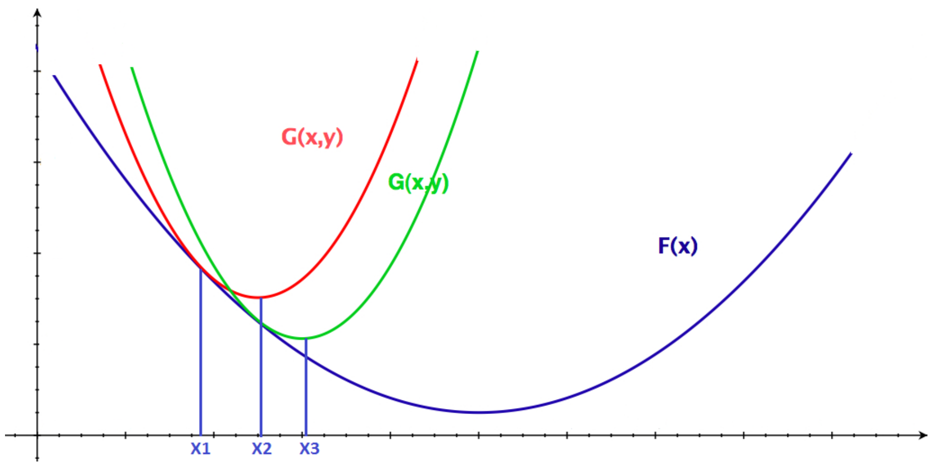


Figure 5.1: Plot illustrating auxiliary function $G(x, y)$ and objective function $F(x)$. The new multiplicative NNLS algorithm (5.3) is constructed by searching the minimum of auxiliary function $G(x, y)$ in each iteration.

Definition 5.1. *For positive vectors, x, y, an auxiliary function, $G(x, y)$, of $F(x)$, has the following two properties*

- $F(x) < G(x, y)$ *if* $x \neq y$;
- $F(x) = G(x, x)$.

Figure 5.1 illustrates the relationship between auxiliary function, $G(x, y)$, and objective function, $F(x)$. In each iteration, the updated x is computed by minimizing the auxiliary function. The iteration stops when it converges to a stationary point, i.e. a local minimum, of the objective function. The following lemma is presented in [73, 101, 102], for the sake of completeness, we include it here.

Lemma 5.2. *Assume $G(x, y)$ is an auxiliary function of $F(x)$, then $F(x)$ is strictly decreasing under the update*

$$x^{k+1} = \underset{x}{\operatorname{argmin}}\ G(x, x^k),$$

as long as $x^{k+1} \neq x^k$.

Proof. By the definition of an auxiliary function $G(x, y)$, if $x^{k+1} \neq x^k$, we have

$$F(x^{k+1}) < G(x^{k+1}, x^k) \leq G(x^k, x^k) = F(x^k).$$

□

In order to derive an auxiliary function for $F(x)$, we first present the following lemma, which is a generalization of Lemma 1 in [101].

Lemma 5.3. *Let P be a $n \times n$ nonnegative real symmetric matrix, x be a $n \times 1$ positive vector. Define the diagonal matrix D*

$$D_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ \frac{(Px)_i}{x_i} & \text{otherwise} \end{cases}$$

Then, the matrices, $(D \pm P)$, are positive semi-definite.

Proof. Consider the matrices,

$$M_1 = \text{diag}(x_i)(D+P)\text{diag}(x_i), \quad M_2 = \text{diag}(x_i)(D-P)\text{diag}(x_i),$$

where $\text{diag}(x_i)$ represents the diagonal matrix with vector x on the main diagonal. Since x is a positive vector, $\text{diag}(x_i)$ is invertible. Hence, $D \pm P$ are congruent with M_1, M_2, respectively. The matrices $D \pm P$ are positive semi-definite if and only if M_1 and M_2 are positive semi-definite [62]. For any nonzero vector z,

$$\begin{aligned} z^T M_1 z &= \sum_{ij} (D_{ij} + P_{ij}) x_i x_j z_i z_j \\ &= \sum_{ij} D_{ij} x_i x_j z_i z_j + \sum_{ij} P_{ij} x_i x_j z_i z_j \\ &= \sum_{i} D_{ii} x_i^2 z_i^2 + \sum_{ij} P_{ij} x_i x_j z_i z_j \\ &= \sum_{i} (Px)_i x_i z_i^2 + \sum_{ij} P_{ij} x_i x_j z_i z_j \\ &= \sum_{ij} P_{ij} x_i x_j z_i^2 + \sum_{ij} P_{ij} x_i x_j z_i z_j \\ &= \frac{1}{2} \sum_{ij} P_{ij} x_i x_j (z_i + z_j)^2 \geq 0 \end{aligned}$$

Similarly,

$$z^T M_2 z = \frac{1}{2} \sum_{ij} P_{ij} x_i x_j (z_i - z_j)^2 \geq 0.$$

Therefore, $D \pm P$ are positive semi-definite. $\square$

The following lemma constructs an auxiliary function for (5.2).

Lemma 5.4. *For any positive vectors, x, y, define the diagonal matrix, $D(y)$, with diagonal element*

$$D_{ii} = \frac{(|Q|y)_i + h_i^- + \delta}{y_i}, \quad i = 1, 2, \cdots, n$$

where $\delta > 0$. The function

$$G(x,y) = F(y) + (x-y)^T \nabla F(y) + \frac{1}{2}(x-y)^T D(y)(x-y)$$

is an auxiliary function for

$$F(x) = \frac{1}{2}x^T Q x - x^T h.$$

Proof. The second property of auxiliary function, $G(x,x) = F(x)$, is obviously true. All we need to show is the first property, $G(x,y) > F(x)$ for $x \neq y$. Notice that Q is the Hesssian matrix of $F(x)$. The Taylor expansion of $F(x)$ at y is

$$F(x) = F(y) + (x-y)^T \nabla F(y) + \frac{1}{2}(x-y)^T Q(x-y)$$

hence,

$$G(x,y) - F(x) = \frac{1}{2}(x-y)^T (D(y) - Q)(x-y).$$

Therefore, $G(x,y) > F(x)$ if and only if $(D(y) - Q)$ is positive definite.

Recall that $|Q| = Q^+ + Q^-$, $|Q|y = Q^+y + Q^-y$,

$$\begin{aligned} D(y) - Q &= \operatorname{diag}\left(\frac{(|Q|y)_i + h_i^- + \delta}{y_i}\right) - Q \\ &= \operatorname{diag}\left(\frac{(|Q|y)_i + h_i^- + \delta}{y_i}\right) - (Q^+ - Q^-) \\ &= \operatorname{diag}\left(\frac{(Q^+y)_i}{y_i}\right) - Q^+ + \operatorname{diag}\left(\frac{(Q^-y)_i}{y_i}\right) + Q^- + \operatorname{diag}\left(\frac{h_i^- + \delta}{y_i}\right) \end{aligned}$$

From Lemma 5.3, $\left(\operatorname{diag}\left(\frac{(Q^+y)_i}{y_i}\right) - Q^+\right)$ and $\left(\operatorname{diag}\left(\frac{(Q^-y)_i}{y_i}\right) + Q^-\right)$ are positive semi-definite. Further, $\operatorname{diag}\left(\frac{h_i^- + \delta}{y_i}\right)$ is positive definite. Thus, $(D(y) - Q)$ is positive definite. Therefore, $G(x,y) > F(x)$ for any vectors $x \neq y$. □

We now prove the convergence of multiplicative iteration (5.3).

Theorem 5.5. *The objective function $F(x)$ in (5.2) is monotonically decreasing under the multiplicative update*

$$x_i^{k+1} = x_i^k \left[\frac{2(Q^-x^k)_i + h_i^+ + \delta}{(|Q|x^k)_i + h_i^- + \delta}\right],$$

with $\delta > 0$. It attains a local minimum at the limit point of the iteration.

Proof. Lemma 5.2 and 5.4 show that the objective function $F(x)$ is monotonically decreasing under the update

$$x^{k+1} = \underset{x}{\operatorname{argmin}} \;\; G(x, x^k) \quad \text{if} \quad x^{k+1} \neq x^k.$$

It remains to show that the iteration (5.3) approaches a local minimum of $G(x, x^k)$. This can be shown by taking the first partial derivative of $G(x, y)$ with respect to x, and setting it to 0,

$$\nabla_x G(x, x^k) = \nabla F(x^k) + D(x^k)(x - x^k) = 0 \tag{5.5}$$

Hence, for $0 < \delta \ll 1$,

$$\begin{aligned}
x &= x^k - (D(x^k))^{-1}\nabla F(x^k) \\
&= x^k - (D(x^k))^{-1}(Qx^k - h + \delta - \delta) \\
&= x^k - (D(x^k))^{-1}(|Q|x^k + h^- + \delta - 2Q^-x^k - h^+ - \delta) \\
&= x^k - (D(x^k))^{-1}(|Q|x^k + h^- + \delta) + (D(x^k))^{-1}(2Q^-x^k + h^+ + \delta) \\
&= (D(x^k))^{-1}(2Q^-x^k + h^+ + \delta) \\
&= \operatorname{diag}\left(\frac{2Q^-x^k + h^+ + \delta}{|Q|x^k + h^- + \delta}\right) x^k
\end{aligned} \tag{5.6}$$

The second-to-last equality is because $(D(x^k))^{-1}(|Q|x^k + h^- + \delta) = x^k$.

On the other hand, the decreasing sequence $\{F(x^{k+1})\}$ is bounded below $-b^T b$. Therefore, it converges to its lower bound F^*. Because $F(x)$ is continuous, given any closed finite interval $[a, b]$, there exists $x^* \in [a, b]$ such that $F(x^*) = F^*$. Since $F(x^*)$ is a local minimum of $F(x)$, the gradient of $F(x)$ at x^* is zero, i.e.

$$\nabla F(x^*) = Qx^* - h = 0.$$

Equivalently,

$$\frac{2(Q^-x^*)_i + h_i^+ + \delta}{(|Q|x^*)_i + h_i^- + \delta} = 1,$$

which means x^* is a stationary vector. Thus, the sequence $\{F(x^k)\}$ converges to its minimum $F(x^*)$ as $\{x^k\}$ approaches its limit point x^*. □

Remark: The Lagrangian function of (5.2) is

$$L(x, \mu) = \frac{1}{2}x^T Qx - x^T h - \mu x, \tag{5.7}$$

with Lagrangian multiplier $\mu_i \geq 0$. Assuming x^* is the optimal solution of (5.7), the Karush-Kuhn-Tucker (KKT) conditions [83] are

$$\begin{aligned}
x^* \circ (Qx^* - h - \mu) &= 0 \quad (5.8) \\
\mu \circ x^* &= 0, \quad (5.9)
\end{aligned}$$

with $\circ$ denoting the Hadamard product. If $x_i^* > 0$, from above two equations, $\mu_i = 0$ and $(Qx^*)_i - h_i = 0$, gives

$$\frac{2(Q^- x^*)_i + h_i^+ + \delta}{(|Q|x^*)_i + h_i^- + \delta} = 1.$$

Because the multiplicative factor in (5.3) equals to one, the stationary limit of the iteration is the optimal solution x^*.

Remark: The multiplicative iteration (5.3) requires a positive constant δ. Without the positive δ, the iteration can't be guaranteed to monotonically converge. Theoretically, δ can be any positive number.

5.2.3 Nonnegative Sparse Optimum and Acceleration

In the previous discussion, we restricted the optimal solution, x^*, to have all positive elements. In practice, however, x^* may contain zero components. As shown in the previous lemmas and theorem, starting with a positive initial guess, x^0, the iteration sequence defined by (5.3), $\{x^k\}$, is always positive. Hence, the computed iteration can't get to an optical solution with a zero component exactly. Nevertheless, the sequence $\{x^k\}$ monotonically converges to the optimal solution.

However, if we know ahead of time that the solution is sparse ahead, we can add a regularization term to (5.1),

$$\operatorname*{argmin}_x \hat{F}(x) = \operatorname*{argmin}_x ||Ax - b||_2^2 + \lambda ||x||_1, \quad x \geq 0, \lambda > 0 \tag{5.10}$$

with nonnegative λ as the regularization parameter. Note that for nonnegative x, $||x||_1 = \sum_{i=1}^n x_i$. The corresponding quadratic programming problem is

$$\operatorname*{argmin}_x \hat{F}(x) = \operatorname*{argmin}_x \frac{1}{2} x^T Q x - x^T h + \lambda \sum_{i=1}^n x_i, \quad x \geq 0, \lambda > 0 \tag{5.11}$$

The new multiplicative iteration for (5.11) is

$$x_i \leftarrow x_i \left[\frac{2(Q^- x)_i + h_i^+}{(|Q|x)_i + h_i^- + \lambda} \right]. \tag{5.12}$$

The convergence analysis for (5.12) is similar to the previous analysis for (5.3), with the auxiliary function

$$\hat{G}(x, y) = \hat{F}(y) + (x - y)^T \nabla \hat{F}(y) + \frac{1}{2}(x - y)^T \hat{D}(y)(x - y)$$

and diagonal matrix, $\hat{D}(y)$,

$$\hat{D}_{ii} = \frac{(|Q|y)_i + h_i^- + \lambda}{y_i}, \quad y_i > 0, i = 1, 2, \cdots, n.$$

Theorem 5.6. *The objective function $\hat{F}(x)$ in* (5.11) *is monotonically decreasing under the multiplicative update*

$$x_i^{k+1} = x_i^k \left[\frac{2(Q^- x^k)_i + h_i^+}{(|Q|x^k)_i + h_i^- + \lambda} \right],$$

with $\lambda > 0$. It attains the minimum at the limit point of the iteration.

The proof of Theorem 5.6 is similar to the proof for Theorem 5.5 (replace δ by λ). In the case of a sparse optimal solution, x^*, the regularized multiplicative iteration converges to a solution with zero components faster than the iteration without regularization (5.3),

5.3 Image Processing Applications

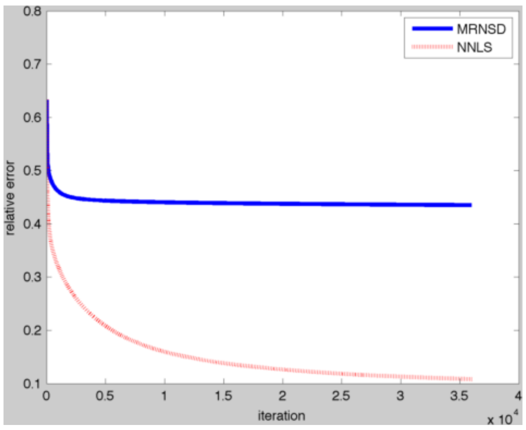
In this section, we show the use of the proposed NNLS algorithm by applying it to image super-resolution and color image labeling problems. Our purpose here is to illustrate the application of the proposed NNLS algorithm. We leave the detailed comparison of this algorithm with other NNLS algorithms for future research.

5.3.1 Numerical Convergence Example

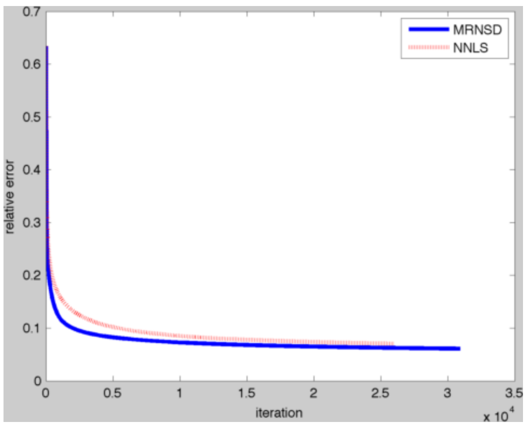
The goal of this set of experiments is to show that the multiplicative iteration 5.3 converges monotonically and compare the convergence rate with MRNSD [82]. The two real-world datasets can be downloaded from the University of Florida Sparse Matrix Collection [39].

- HB/bcsstk21 dataset. Harwell-Boeing 3600×3600 sparse symmetric positive definite matrix with condition number $1.76e + 07$ and 26600 nonzero entries (approximately 0.2% sparsity).
- Gset/G61 dataset. 7000×7000 sparse symmetric matrix with condition number $1.70e + 18$ and 34296 nonzero entries(approximately 0.07% sparsity). The nonzero elements are uniformly distributed in matrix.

We generate an absolute value of sine function vector, x, on interval $[0, 20]$ and the right-hand side vector, $b = A * x$. Hence, we can compute the relative error at each iteration. The iteration stops either the tolerance criteria are satisfied or max number of iterations is reached. Because, typically, both algorithms don't converge quickly. We set the max number of iterations to be 10 times the length of vector x. The tolerance, which is 10^{-6}, is set for the relative difference between current and previous iterations. Our initial guess is set to be a random positive vector generate with Matlab *rand* command.



(a) HB/bcsstk21 dataset



(b) Gset/G61 dataset

Figure 5.2: Relative errors at each iteration for algorithm (5.3) and MRNSD.

Figure 5.2a and 5.2b show the iteration history of Algorithm 5.3 and MRNSD. Note in both our algorithm and MRNSD, there are 2 matrix-vector multiplications in each iteration. Hence, we don't show the computational time, which is affected by the implementation details. Here we see that both algorithms reduces the relative errors dramatically in the first several iterations. The reductions become slower as iteration increasing. In addition, we see that our algorithm (5.3) produces solutions with lower relative error than MRNSD in the first test. In the second test, both algorithms have the similar performance. A more detailed comparison leaves for the future research.

5.3.2 Image Super-Resolution

Image super-resolution (SR) refers to the process of combining a set of low resolution images into a single high resolution image [31, 35]. Typically, it is a accomplished in two steps: registration and reconstruction. In the registration step, the relative displacement of each low-resolution image from a reference image is computed. Then the high resolution image is computed by solving a linear inverse problem.

Image registration, which plays a central role in the context of SR reconstruction, refers the process of spatially matching the reference and target images, so that the pixels in the two images correspond to the same physical region of the scene being imaged. Many methods have been developed for various types of problems [24, 125]. In this experiment, we assume that the transformations between low-resolution images are purely translational [43], which can be computed by a transform model estimation, such as that proposed by Chung *et al.* [35].

Once the low-resolution images have been registered, all the pixels can be combined to form a composite image. This image is no longer sampled on a uniform rectangular grid. Reconstructing the high-resolution image requires interpolation and re-sampling. In the experiments, we use bilinear interpolation, which is commonly used because of the trade-off between performance and computational complexity [121].

The underlying SR forward mathematical model is written as the following linear system [35]

$$y_k = D_k S_k x + n_k, \quad k = 1, 2, \cdots, K \tag{5.13}$$

where

- x is the unknown high resolution image;
- y_k is kth the low resolution image;
- S_k is the bilinear interpolation matrix depending on the displacement computed by registration step;
- D_k is the sparse decimation matrix, see [31] for more information;
- n_k represents unknown errors in the observed data, such as discretization errors and noise.

Equation (5.13) is solved by computing the following NNLS problem

$$\underset{x}{\operatorname{argmin}} \frac{1}{2} \sum_{k=1}^{K} ||D_k S_k x - y_k||^2, \quad \text{s.t.} \quad x \geq 0$$

The mathematical framework used here is the inexact Gauss-Newton method for super-resolution in [35] [1]. We use our NNLS algorithm and MRNSD to solve the above minimization problem. In these experiments, D_k, S_k and y_k are all nonnegative. Hence, the proposed algorithm is in the reduced form of ISRA. Our test data-set is taken from the Multi-Dimensional Signal Processing Research Group (MDSP) [45]. Numerical experiments show that if the NNLS algorithms stop in $25 - 30$ iterations, the results are visually better. Hence, we set the max iteration to be 25 in both our NNLS algorithm 5.3 and MRNSD algorithm.

Figures 5.3a and 5.3b show 2 of the 30 uncompressed 57×49 grayscale low-resolution text image. The high-resolution computed by our algorithm is shown in Figure 5.3d. Figure 5.3c

[1]Thanks to Julianne Chung and James Nagy for providing the Matlab code.

shows the computed high-resolution image by MRNSD algorithm. The high-resolution image has the size of 285×245. Figures 5.4a and 5.4b show 2 of 16 low-resolution EIA images with size 90×90. Figures 5.4c and 5.4d show the reconstructed 360×360 high-resolution images by MRNSD and our algorithm, correspondingly. As shown in the figures, the high-resolution images are visually much better than the corresponding low-resolution images.

Remark: The purpose of these experiments is to show our new NNLS algorithm (5.3) can be employed in super-resolution problem. The reconstructed high-resolution image quality depends heavily on the particular model being used, especially image registration model [31]. The least square solvers play a less important role in this problem.

5.3.3 Color Image Labeling

The goal of image labeling is to divide the image into several constituent parts. There are many existing methods, including the Mumford-Shah functional method [80], the level set method [111], and the snake method [67, 118]. Among these approaches, Markov random fields (MRF) based interactive image segmentation techniques have become popular because of their robustness [96]. These techniques require users to impose hard constraints by indicating certain pixels (seeds) that absolutely have to be part of the labeling k. Intuitively, the hard constraints provide clues as to what the user intends to segment. Denote X as the m-by-n test RGB images, X_{ij} represent a 3-by-1 vector at pixel (i, j).

The class set is denoted by $\mathcal{C} = \{1, 2, \cdots, K\}$. The probabilistic labeling approaches compute a probability measure field for each pixel (i, j),

$$\mathcal{X} = \{X_{ij}^k : k \in \mathcal{C}, i = 1, 2, \cdots, m, j = 1, 2, \cdots, n\}$$

with the constraints

$$\sum_{k=1}^{K} X_{ij}^k = 1, \quad X_{ij}^k \geq 0, \quad \forall k \in \mathcal{C}. \tag{5.14}$$

Denoting $\mathcal{N}_{ij} = \{(i', j') : \min\{|i' - i|, |j' - j|\} = 1\}$ as the set of neighbors of pixel (i, j), the cost function are in the following quadratic form

$$\operatorname*{argmin}_{x} \sum_{k=1}^{K} \sum_{i=1}^{m} \sum_{j=1}^{n} \left(\frac{\alpha}{2} \sum_{(i',j') \in \mathcal{N}_{ij}} \omega_{iji'j'} (X_{i'j'}^k - X_{ij}^k)^2 + D_{ij}^k X_{ij}^k \right), \tag{5.15}$$

with the constraints (5.14). D_{ij}^k is the cost of assigning label k to pixel (i, j). The first term, $\sum_{(i',j') \in \mathcal{N}_{ij}} \omega_{iji'j'} (X_{i'j'}^k - X_{ij}^k)^2$, which controls the granularity of the regions, promotes

Algorithm 5 NNLS Algorithm MRF Image Segmentation

```
1: while norm($(X^k)^{\text{new}} - (X^k)^{\text{old}}$)/norm($(X^k)^{\text{old}}$) > tol do
2:     Update the probability measure field
           $(X^k_{ij})^{\text{new}} = (X^k_{ij})^{\text{old}} * \frac{2\alpha \sum_{(i',j')\in\mathcal{N}_{ij}} \omega_{iji'j'} X^k_{ij} + (D^k_{ij})^- + \lambda_{ij}}{\alpha X^k_{ij}\left(\sum_{(i',j')\in\mathcal{N}_{ij}} \omega_{iji'j'}\right) + \alpha \sum_{(i',j')\in\mathcal{N}_{ij}} \omega_{iji'j'} X^k_{ij} + (D^k_{ij})^+}$
3:     Update the Lagrangian parameter
           $\lambda^{\text{new}}_{ij} = \lambda^{\text{old}}_{ij} * \frac{1}{\sum_k X^k_{ij}}$
4: end while
5: return $(X^k)^{\text{new}}$
```

smooth regions. The spatial smoothness is controlled by the positive parameter, α, and weight, ω, which is chosen such that $\omega_{iji'j'} \approx 1$ if the neighbouring pixels (i,j) and (i',j') are likely to belong to the same class and $\omega_{iji'j'} \approx 0$ otherwise. In these experiments, ω is defined to be the cosine of the angle between two neighbouring pixels,

$$\omega_{iji'j'} = \frac{X^T_{ij} X_{i'j'}}{|X_{ij}| \cdot |X_{i'j'}|}.$$

The cost of labeling k at each pixel (i,j), D^k_{ij}, is trained with a Gaussian mixture model [119] using seeds labeled by the user. Given sample mean, μ^k, and variance, σ^k, for the seeds with labeling k, D^k_{ij} is computed as the Mahalanobis distance [77] between each pixel of the image and the seeds,

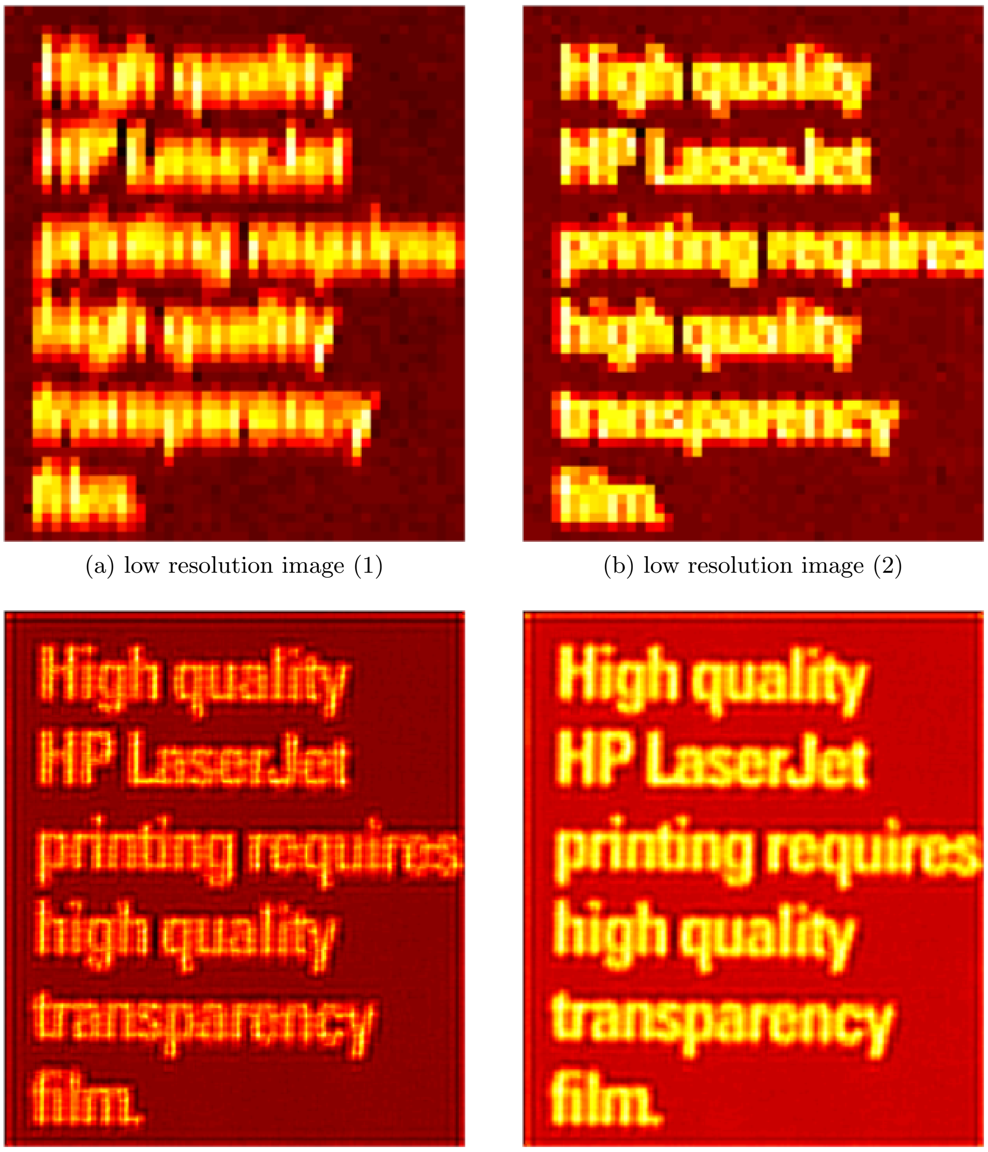
$$D^k_{ij} = \frac{1}{2}\sum_{k=1}^{K}(X_{ij} - \mu^k)^T(\Sigma^k)^{-1}(X_{ij} - \mu^k) + \frac{1}{2}\log(\Sigma^k).$$

The KKT optimality conditions

$$\begin{aligned} \alpha \sum_{(i',j')\in\mathcal{N}_{ij}} \omega_{iji'j'}(X^k_{ij} - X^k_{i'j'}) + D^k_{ij} - \lambda_{ij} &= 0 \\ \alpha X^k_{ij}\left(\sum_{(i',j')\in\mathcal{N}_{ij}} \omega_{iji'j'}\right) - \alpha \sum_{(i',j')\in\mathcal{N}_{ij}} \omega_{iji'j'} X^k_{ij} + D^k_{ij} - \lambda_{ij} &= 0 \end{aligned}$$

yield a two-step Algorithm 5.

We implement our NNLS algorithm without explicitly construct the matrix Q. Figure 5.5 shows the results of the labeled images. The images are segmented into different areas nicely using this new algorithm. This example shows another application of our NNLS algorithm (5.3).



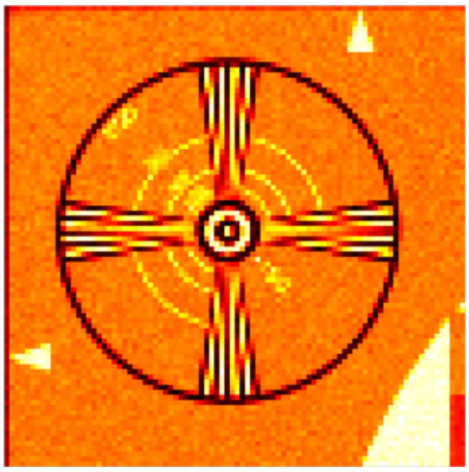
(a) low resolution image (1)

(b) low resolution image (2)

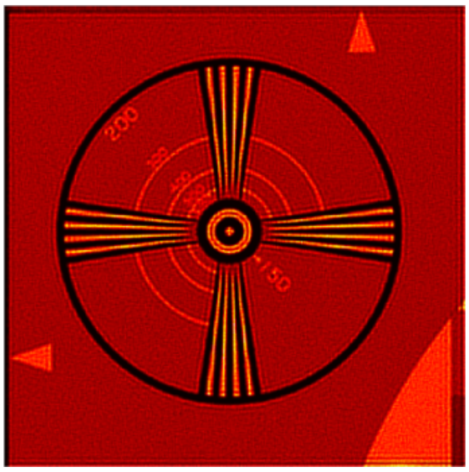
(c) restored high resolution image: MRNSD

(d) restored high resolution image: Algorithm 5

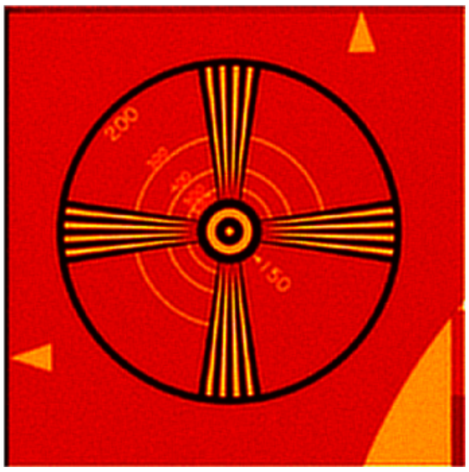
Figure 5.3: 5.3a and 5.3b are 2 sample 57×49 low-resolution images out of 30. 5.3c and 5.3d are the reconstructed high-resolution image with size 285×245, by MRNSD and proposed algorithm correspondingly. Note that both algorithms stop at 25 iterations, and the cost of each iteration of the proposed algorithm is about the same as the cost of MRNSD (2 matrix-vector multiplications are the major cost)..


(a) low resolution image (1)

(b) low resolution image (2)



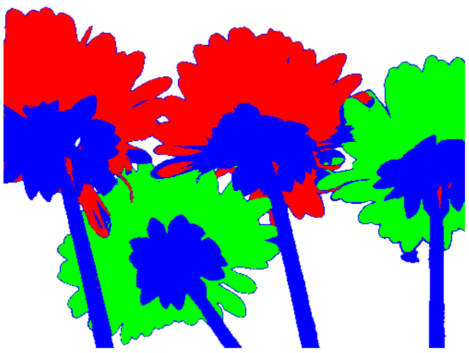
(c) Restore high resolution image: MRNSD



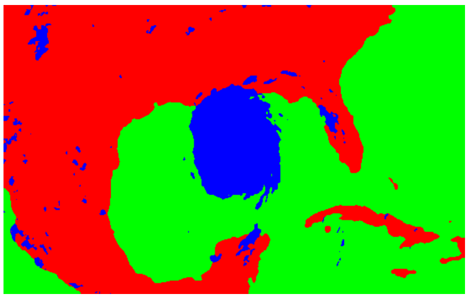
(d) Restored high resolution image: Algorithm 5

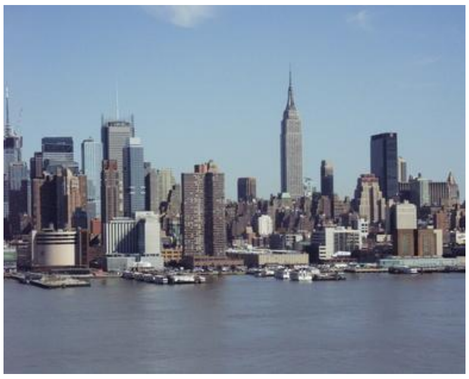
Figure 5.4: 5.4a and 5.4b are 2 sample 90×90 low-resolution images out of 16. 5.4c and 5.4d are the reconstructed high-resolution image with size 360×360, by MRNSD and proposed algorithm correspondingly. Note that both algorithms stop at 25 iterations, and the cost of each iteration of the proposed algorithm is about the same as the cost of MRNSD (2 matrix-vector multiplications are the major cost).

(a) flowers


(b) segmented image

(c) Satellite image


(d) segmented image


(e) Manhattan skyline

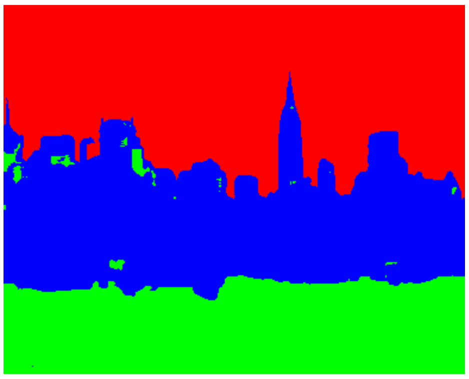

(f) segmented image

Figure 5.5: Image labeling results using MRF model solved by the proposed NNLS algorithm

Chapter 6

Concluding Remarks

In this work, we discuss some mathematical schemes and numerical implementations for large-scale image restoration problems. We develop two edge-preserving algorithms for image restoration and a novel multiplicative NNLS algorithm. Although this work focuses on image restoration problems, ill-posed inverse problems arise in many other scientific applications, and the developed numerical methods can be used in other fields as well.

The mathematical contributions of this work include developing an efficient Brent-NCP parameter-selection algorithm for large-scale regularization problems. For image deblurring problem, we develop the theory for a new edge-preserving deblurring framework (4.1) based on the projection space idea. In Chapter 5, we design a new multiplicative NNLS algorithm (5.3) and give theoretical convergence analysis for the algorithm.

In terms of computational contributions, we explore the applications of multigrid methods and compare the performance of BoxMG and AMG in anisotropic diffusion denoising problem, show by examples the broad range of applications of Brent-NCP parameter-selection algorithm in several existing parameter-dependent algorithms. In the edge-preserving deblurring problem, we implement an efficient and competitive algorithm, including developing the Kronecker product SVDs and AMG preconditioner, and running extensive numerical tests to determine several parameters. We also illustrate the potential applications of the new multiplicative NNLS algorithm in the context of image super-resolution and color image labeling.

Some future work includes:

Implementing Symmetric AMG Preconditioners As discussed in Chapter 4, because we use Gauss-Seidel method in the pre- and post-relaxations, which causes our AMG residual reduction operator is not symmetric. Hence, we can't use preconditioned CG method to solve the normal equation (4.12). However, if we replace the Gauss-Seidel method by red-black Gauss-Seidel method in pre-relaxation and black-red Gauss-Seidel method in post-relaxation, the new AMG conditioner is symmetric. Therefore, it is possible to solve the normal equation with CG-type algorithms, which might be faster than current GMRES solver.

Studying the Singular Vectors for the Non-separable PSF in EPP Algorithm As discussed in Chapter 4, if the PSF is non-separable, we have to approximate the singular vectors with Kronecker products of two smaller matrices. In our experiments, we compute the low-frequency components, x_{lf}, by the explicit approximation

$$x_{\text{lf}} = \sum_{i=1}^{k} \frac{u_i^T \tilde{b}}{s_i} v_i.$$

However, the results in Table 4.3 show that the MSSIM measures are smaller and relative errors are larger than DCT-EPP results in Table 4.1. Another approach to compute the x_{lf} is to solve the minimization problem (4.6)

$$\underset{x}{\operatorname{argmin}} \; ||(AW_k)x - b||_2,$$

with iterative methods. It would be interesting to see the restored results using iterative algorithms, such as LSQR [87], LSMR [46] etc. to compute x_{lf}.

Multiple Subspaces Projection Edge-Preserving Deblurring Recently, Chung *et al.* developed a windowed approach for spectral regularization of deblurring problem [34]. By providing a regularization parameter for each spectral subspace, this windowed regularization approach effectively restored blurred images, particularly for high-noise data. In Chapter 4, we split the spectral space into two subspaces, and computed components of the solution living in the two subspaces with different methods. A natural extension is partitioning the spectral space into more subspaces, and computing components in each subspaces separately. We have tried to partition the spectral space into 3 subspaces, and compute components living in first two subspaces while ignoring the last subspace, which is totally dominated by noise; however, there remain some unsolved numerical implementation issues.

Theoretical Properties of Projection-Based Regularization In [104], the authors studied two fundamental properties of TV methods:

- edge locations of image features tend to be preserved and under certain conditions, are preserved exactly;
- intensity change experienced by individual features is inversely proportional to the scale of each feature.

These results help explain how and why TV image restoration can remove noise while leaving relatively intact larger-scaled image features. The goal of our projection regularization is the same as TV regularization. It would be interesting to investigate the theoretical properties of projection regularization.

Systematic Comparing the NNLS Algorithm with Others In [82, 10], Nagy *et al.* proposed another NNLS algorithm – modified residual norm steepest descent algorithm (MRNSD) and its extension to weighted least squares problems, WMRNSD. As our algorithm (5.3), there are two matrix-vector multiplications in each iteration. Their numerical experiments show that MRNSD, especially when used with a preconditioner, is competitive to the unconstrained Krylov subspace methods. It would be interesting to compare the performance of our NNLS algorithm, MRNSD, WMRNSD, and other existing NNLS algorithms, using some benchmark NNLS problems.

Exploring the Parallel Implementation of the NNLS Algorithm In our multiplicative NNLS algorithm, the components in each new iteration do not depend on each other, so they can be computed simultaneously, hence, can be easily implemented on parallel machines. Exploring the parallel implementation of the new algorithm with different applications is an interesting problem.

Appendix A

Brent-NCP Matlab Code

```
function xmin = brentNCP(ax, cx, tol, denoisingMethod, im_n)
% Brent-NCP method: compute the near optimal regularization parameter in [ax, cx]
% Inputs:
% ax: left bound
% cx: right bound
% tol: tolerance, iterations stops when interval becomes less than tol
% denoisingMethod: function handle, user input denoising method,
%               (can be replace by other parameter-dependent method)
% im_n: noisy image, argument for denoisingMethod
% Output: xmin: regularization parameter
% Donghui Chen, 2012

% initialization
gold = 1 - (-1+sqrt(5))/2;
ITMAX = 100;
ZEPS = 10^(-10);
e = 0;

% let a = min(ax, cx), b = max(ax, cx)
if ax < cx, a = ax; b = cx; else a = cx; b = ax; end;

bx = a + gold*(b-a); % golden section step
```

```
x = bx; w = bx; v = bx;
fx = compNCPdist(denoisingMethod, im_n, x); % compute the NCP distance

fw = fx; fv = fw;

% iteration
for iter = 1:ITMAX
    xm = 0.5*(a+b);
    tol1 = tol*abs(x) + ZEPS;
    tol2 = 2*tol1;
    if (abs(x-xm) <= (tol2-0.5*(b-a)))
        xmin = x;
        fprintf('succeeded after %d steps\n', iter);
        return;
    end

    %     fit a parabola
    r = 0; q = r; p = q;
    if (abs(e) > tol1)
        r = (x-w)*(fx-fv);
        q = (x-v)*(fx-fw);
        p = (x-v)*q - (x-w)*r;
        q = 2*(q-r);
        if (q > 0)
            p = -p;
        end
        q = abs(q);
        r = e;
        e = d;
    end

    if (~(abs(p) >= abs(0.5*q*r) || p <= q*(a-x) || p >= q*(b-x)))
%         parabolic fit is good
        d = p/q;
        u = x+d;
        if (u-a < tol2 || b-u < tol2), d = tol1*sign(xm-x); end;
    else
%          golden-section step.
        if x >= xm
            e = (a-x);
        else
            e = (b-x);
        end
        d = gold*e;
    end

%       update the new point, the step size is either d or tol1
    if abs(d) >= tol1
        u = x+d;
    else
        u = x+tol1*sign(d);
    end
    fu = compNCPdist(denoisingMethod, im_n, u); % compute the NCP distance

%      update a, b, v, w, x
    if (fu <= fx)
        if u >= x, a = x; else b = x; end;
        [v, w, x] = shift(v, w, x, u);
        [fv, fw, fx] = shift(fv, fw, fx, fu);
    else
        if u < x, a = u; else b = u; end;
        if (fu <= fw || w == x)
            v = w; w = u; fv = fw; fw = fu;
        elseif (fu <= fv || v == x || v == w)
            v = u; fv = fu;
        end
    end
end

disp('failed requirements after %d steps\n', ITMAX);
```

```
%-----------------------------------------------------------------
function [a, b, c] = shift(a, b, c, d)
a = b;
b = c;
c = d;

%-----------------------------------------------------------------
function dist = compNCPdist(denoisingMethod, im_n, mu)
% Compute the NCP distance
% Inputs:
% denoisingMethod: function handle, user input denoising method,
%                  (can be replace by other parameter-dependent method)
%  im_n: noisy image
%  mu:  current regularization parameter
% Output: dist: the NCP distance between the residual and white Gaussian noise
% Donghui Chen 2012

im_c = denoisingMethod(im_n, mu); % compute restored image
res = im_c - im_n; % residual image
dist = NCPdist(res); % the NCP distance between the residual and white Gaussian noise


%-----------------------------------------------------------------
function dist = NCPdist(X)
% Compute the cumulative periodogram of the 2D image X
% Per Christian Hansen, IMM, Nov. 17, 2004.

[n1, n2] = size(X);

q1 = (floor(n1/2));
q2 = (floor(n2/2));
for i=1:q1
    for j=1:q2
        R(i,j) = i^2+j^2;
    end
end

[dummy, perm] = sort(reshape(R,q1*q2,1));

Z = abs(fft2(X)).^2;
d = reshape(Z(1:q1,1:q2), q1*q2, 1);
D = d(perm);
D = D(2:end);  % Get rid of DC component.

v = linspace(0, 1, size(D,1))'; % NCP of the white Gaussian noise
cp = cumsum(D)/sum(D);
dist = norm(v-cp, 1); % NCP distance between input image and white Gaussian noise
```

Bibliography

[1] S. Acton. Multigrid anisotropic diffusion. *IEEE Transactions on Image Processing*, 7(3):280–290, 1998.

[2] A. Albert. *Regression and the Moore-Penrose pseudoinverse.* Mathematics in Science and Engineering. Elsevier, Burlington, MA, 1972.

[3] R. Alcouffe, A. Brandt, J. Dendy Jr., and J. Painter. The multigrid method for the diffusion equation with strongly discontinuous coefficients. *SIAM J. on Scientific and Statistical Computing*, 2(4):430–454, 1981.

[4] L. Alvarez and L. Mazorra. Signal and image restoration using shock filters and anisotropic diffusion. *SIAM J. Numer. Anal.*, 31(2):590–605, April 1994.

[5] G.E.B. Archer and D.M. Titterington. The iterative image space reconstruction algorithm as an alternative to the EM algorithm for solving positive linear inverse problems. *Statistica Sinica*, 5:77–96, 1995.

[6] K. Atkinson. *An Introduction to Numerical Analysis.* Wiley, 2 edition, 1989.

[7] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations (2nd edition)*, volume 147 of *Applied Mathematical Sciences.* Springer-Verlag, 2006.

[8] G. Aubert and L. Vese. A variational method in image recovery. *SIAM J. Numerical Analysis*, 34:1948–1979, 1997.

[9] D. Barash, M. Israeli, and R. Kimmel. An accurate operator splitting scheme for nonlinear diffusion filtering. In *Scale-Space Theories in Computer Vision*, pages 281–289, 2001.

[10] J. Bardsley and J. Nagy. Covariance-preconditioned iterative methods for nonnegatively constrained astronomical imaging. *SIAM J. Matrix Anal. Appl.*, 27(4):1184–1197, 2006.

[11] F. Bauer and M. Lukas. Comparing parameter choice methods for regularization of ill-posed problems. *Mathematics and Computers in Simulation*, 81(9):1795–1841, May 2011.

[12] S. Bellavia, M. Macconi, and B. Morini. An interior point Newton-like method for nonnegative least-squares problems with degenerate solution. *Numerical Linear Algebra with Applications*, 13(10):825–846, 2006.

[13] M. Van Benthem and M. Keenan. Fast algorithm for the solution of large-scale nonnegativity-constrained least squares problems. *J. of Chemometrics*, 18(10):441–450, 2004.

[14] F. Benvenuto, R. Zanella, L. Zanni, and M. Bertero. Nonnegative least-squares image deblurring: improved gradient projection approaches. *Inverse Problems*, 26, 2010.

[15] A. Berman and R. Plemmons. *Nonnegative Matrices in the Mathematical Sciences.* SIAM, January 1987.

[16] M. Berry, M. Browne, A. Langville, V. Pauca, and R. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics & Data Analysis*, 52(1):155–173, September 2007.

[17] A. Björck. *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia, 1996.

[18] P. Bloomfield. *Fourier Analysis of Time Series: An Introduction.* John Wiley and Sons, New York, NY, 1976.

[19] M. Brand and D. Chen. Parallel quadratic programming for image processing. In *18th IEEE International Conference on Image Processing (ICIP)*, pages 2261 –2264, September 2011.

[20] M. Brand, V. Shilpiekandula, Y. Chen, and S. Bortoff. A parallel quadratic programming algorithm for model predictive control. In *Proceedings of the 18th IFAC World Congress*, volume 18, August 2011.

[21] A. Brandt, S. McCormick, and J. Ruge. Algebraic multigrid (AMG) for sparse matrix equations. In D. Evans, editor, *Sparsity and its Applications*, pages 257–284. Cambridge, 1984.

[22] W. Briggs, V. Henson, and S. McCormick. *A Multigrid Tutorial (2nd ed.)*. SIAM, Philadelphia, PA, USA, 2000.

[23] R. Bro and S. De Jong. A fast non-negativity-constrained least squares algorithm. *J. of Chemometrics*, 11(5):393–401, 1997.

[24] L. Brown. A survey of image registration techniques. *ACM Comput. Surv.*, 24:325–376, December 1992.

[25] A. Buades, B. Coll, and J.M. Morel. On image denoising methods. Technical report, CMLA (Centre de Mathematiques et de Leurs Applications), 2004.

[26] D. Calvetti, G. Landi, L. Reichel, and F. Sgallari. Non-negativity and iterative methods for ill-posed problems. *Inverse Problems*, 20(6):1747, 2004.

[27] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986.

[28] V. Caselles and J.M. Morel. Introduction to the special issue on partial differential equations and geometry-driven diffusion in image processing and analysis. *IEEE Transactions on Image Processing*, 7(3):269–273, 1998.

[29] F. Catté, P.L. Lions, J.M. Morel, and T. Coll. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J. Numerical Analysis*, 29(1):182–193, 1992.

[30] A. Chambolle. An algorithm for total variation minimization and applications. *J. of Mathematical Imaging and Vision*, 20(1-2):89–97, 2004.

[31] D. Chen. Comparisons of multiframe super-resolution algorithms for pure translation motion. Master's thesis, Wake Forest University, Winston-Salem, NC, August 2008.

[32] D. Chen, S. MacLachlan, and M. Kilmer. Iterative parameter-choice and multigrid methods for anisotropic diffusion denoising. *SIAM J. Scientific Computing*, 33(5):2972–2994, 2011.

[33] D. Chen and R. Plemmons. Nonnegativity constraints in numerical analysis. In *A. Bultheel and R. Cools (Eds.), Symposium on the Birth of Numerical Analysis, World Scientific*, 2009.

[34] J. Chung, G. Easley, and D. O'Leary. Windowed spectral regularization of inverse problems. *SIAM J. on Scientific Computing*, 33(6):3175–3200, 2011.

[35] J. Chung, E. Haber, and J. Nagy. Numerical methods for coupled super-resolution. *Inverse Problem*, 22:1261–1272, 2006.

[36] J. Chung, J. Nagy, and D. O'Leary. A weighted-GCV method for Lanczos-hybrid regularization. *Electronic Transactions on Numerical Analysis*, 28:149–167, 2008.

[37] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising with block-matching and 3D filtering. In *Proc. SPIE Electronic Imaging: Algorithms and Systems V*, volume 6064, 2006.

[38] M. E. Daube-Witherspoon and G. Muehllehner. An iterative image space reconstruction algorthm suitable for volume ECT. *Medical Imaging, IEEE Transactions on*, 5(2):61 –66, June 1986.

[39] T. Davis and Y. Hu. The university of florida sparse matrix collection. ACM Transactions on Mathematical Software (to appear), http://www.cise.ufl.edu/research/sparse/matrices.

[40] G. Demoment. Image reconstruction and restoration: overview of common estimation structures and problems. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(12):2024 –2036, December 1989.

[41] Y. Dong and M. Hintermüller. Multi-scale total variation with automated regularization parameter selection for color image restoration. In *Scale Space and Variational Methods in Computer Vision*, volume 5567 of *Lecture Notes in Computer Science*, pages 271–281. Springer Berlin / Heidelberg, 2009.

[42] P. P. B. Eggermont. Multiplicative iterative algorithms for convex programming. *Linear Algebra and its Applications*, 130:25–42, 1990.

[43] M. Elad and Y. Hel-Or. A fast super-resolution reconstruction algorithm for pure translational motion and common space invariant blur. *IEEE Transactions on Image Processing*, 10(8):1187–1193, 2001.

[44] L. Eldén. A weighted pseudoinverse, generalized singular values, and constrained least squares problems. *BIT Numerical Mathematics*, 22:487–502, 1982.

[45] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar. Fast and robust multi-frame super-resolution. *IEEE Transactions on Image Processing*, 13:1327–1344, 2003.

[46] D. Fong and M. Saunders. LSMR: An iterative algorithm for sparse least-squares problems. *SIAM J. Scientific Computing*, 33(5):2950–2971, 2011.

[47] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6(6):721 –741, November 1984.

[48] G. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[49] R. Gonzalez and R. Woods. *Digital Image Processing, Second Edition.* Prentice Hall, 2002.

[50] J. Han, L. Han, M. Neumann, and U. Prasad. On the rate of convergence of the image space reconstruction algorithm. *Operators and Matrices*, 3(1):41–58, 2009.

[51] P.C. Hansen. The truncated SVD as a method for regularization. *BIT*, 27:534–553, October 1987.

[52] P.C. Hansen. The discrete Picard condition for discrete ill-posed problems. *BIT Numerical Mathematics*, 30:658–672, 1990.

[53] P.C. Hansen. Analysis of discrete ill-posed problems by means of the L-curve. *SIAM Review*, 34(4):561–580, 1992.

[54] P.C. Hansen. *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion.* SIAM, 1998.

[55] P.C. Hansen and T.K. Jensen. Noise propagation in regularizing iterations for image deblurring. *Electron. Trans. Numer. Anal.*, 31:204–220, 2008.

[56] P.C. Hansen, M. Kilmer, and R.H. Kjeldsen. Exploiting residual information in the parameter choice for discrete ill-posed problems. *BIT*, 46:41–59, 2006.

[57] P.C. Hansen and K. Mosegaard. Piecewise polynomial solutions to linear inverse problems. In B. Jacobsen, K. Mosegaard, and P. Sibani, editors, *Inverse Methods*, volume 63 of *Lecture Notes in Earth Sciences*, pages 284–294. Springer Berlin / Heidelberg, 1996.

[58] P.C. Hansen, J. Nagy, and D. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering.* SIAM, Philadelphia, PA, USA, 2006.

[59] P.C. Hansen and D. O'Leary. The use of the L-curve in the regularization of discrete ill-posed problems. *SIAM Journal on Scientific Computing*, 14(6):1487–1503, 1993.

[60] P.C. Hansen, T. Sekii, and H. Shibahashi. The modified truncated SVD method for regularization in general form. *SIAM J. on Scientific and Statistical Computing*, 13(5):1142–1150, 1992.

[61] M. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. of Research of the National Bureau of Standards*, 49(6):409–436, December 1952.

[62] R.A. Horn and C.R. Johnson. *Matrix Analysis.* Cambridge University Press, 1990.

[63] Y. Huang, M. Ng, and Y. Wen. A fast total variation minimization method for image restoration. *J. Multiscale Model. Simul.*, 7:774–795, 2008.

[64] J. Dendy Jr. Black box multigrid. *Journal of Computational Physics*, 48(3):366–386, 1982.

[65] J. Dendy Jr. Black box multigrid for nonsymmetric problems. *Applied Mathematics and Computation*, 13(3-4):261–283, 1983.

[66] J. Kamm and J. Nagy. Optimal Kronecker product approximation of block Toeplitz matrices. *SIAM J. Matrix Anal. Appl*, 22:2000, 1999.

[67] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[68] S. Kichenassamy. The Perona-Malik paradox. *SIAM J. Applied Mathematics*, 57(5):1328–1342, 1997.

[69] M. Kilmer and C. Martin. Factorization strategies for third-order tensors. *Linear Algebra and its Applications*, 435(3):641–658, 2011.

[70] D. Kim, S. Sra, and I. Dhillon. A new projected quasi-newton approach for the non-negative least squares problem. Technical report, The University of Texas at Austin, 2006.

[71] H. Köstler. *A Multigrid Framework for Variational Approaches in Medical Image Processing and Computer Vision.* PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, 2008.

[72] C. Lawson and R. Hanson. *Solving Least Squares Problems.* SIAM, 3rd edition, 1995.

[73] D. Lee and S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, April 2001.

[74] H. Liao, F. Li, and M. Ng. Selection of regularization parameter in total variation image restoration. *J. Opt. Soc. Am. A*, 26(11):2311–2320, November 2009.

[75] C. Liu, W. Freeman, R. Szeliski, and S. Kang. Noise estimation from a single image. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:901–908, 2006.

[76] C. Van Loan and N. Pitsianis. Approximation with Kronecker products. In *Linear Algebra for Large Scale and Real Time Applications*, pages 293–314. Kluwer Publications, 1993.

[77] P. Mahalanobis. On the generalised distance in statistics. In *Proceedings National Institute of Science, India*, volume 2, pages 49–55, April 1936.

[78] S. Morigi, L. Reichel, and F. Sgallari. Orthogonal projection regularization operators. *Numerical Algorithms*, 44(2):99–114, February 2007.

[79] P. Mrázek. Selection of optimal stopping time for nonlinear diffusion filtering. In *Proceedings of the Third International Conference on Scale-Space and Morphology in Computer Vision*, pages 290–298. Springer-Verlag, 2001.

[80] D. Mumford and J. Shah. Boundary detection by minimizing functionals. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 22–26, 1985.

[81] J. Nagy, M. Ng, and L. Perrone. Kronecker product approximations for image restoration with reflexive boundary conditions. *SIAM J. Matrix Anal. Appl.*, 25:829–841, March 2003.

[82] J. Nagy and Z. Strakoš. Enforcing nonnegativity in image reconstruction algorithms. In *SPIE Conference Series*, volume 4121 of *SPIE Conference Series*, pages 182–190, October 2000.

[83] J. Nocedal and S. Wright. *Numerical Optimization.* Springer, New York, 2nd edition, 2006.

[84] K. Nordström. Biased anisotropic diffusion – a unified regularization and diffusion approach to edge detection. Technical report, EECS Department, University of California, Berkeley, May 1989.

[85] M. Osborne. *Finite algorithms in optimization and data analysis.* Wiley, Chichester, New York, 1985.

[86] C. Paige and M. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12:617–629, 1975.

[87] C. Paige and M. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.*, 8:43–71, March 1982.

[88] R. Palm. *Numerical comparison of regularization algorithms for solving ill-posed problems.* PhD thesis, University of Tartu, Estonia, 2010.

[89] G. Papandreou and P. Maragos. A cross-validatory statistical approach to scale selection for image denoising by nonlinear diffusion. In *Computer Vision and Pattern Recognition*, pages 625–630, 2005.

[90] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *Proceedings of IEEE Computer Society Workshop on Computer Vision*, pages 16–22, 1987.

[91] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, July 1990.

[92] A. De Pierro. On the convergence of the iterative image space reconstruction algorithm for volume ECT. *Medical Imaging, IEEE Transactions on*, 6(2):174 –175, June 1987.

[93] Y. Pirogov, A. Timanovsky, and V. Gladun. Image acquisition and processing in passive radiovision systems. *Radiophysics and Quantum Electronics*, 49:597–604, 2006.

[94] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 2 edition, October 1992.

[95] R. Puetter, T. Gosnell, and A. Yahil. Digital image reconstruction: Deblurring and denoising. *Astronomy and Astrophysics*, 43(1):139, 2005.

[96] M. Rivera, O. Dalmau, and J. Tago. Image segmentation by convex quadratic programming. In *19th International Conference on Pattern Recognition, 2008*, pages 1–5, 2008.

[97] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992.

[98] J. Ruge and K. Stüben. Algebraic multigrid. In S. McCormick, editor, *Multigrid Methods*, pages 73–130, Philidelphia, Pennsylvania, 1987. SIAM.

[99] B. Rust and D. O'Leary. Residual periodograms for choosing regularization parameters for ill-posed problems. *Inverse Problems*, 24(3):034005 (30pp), 2008.

[100] Y. Saad and M. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, July 1986.

[101] F. Sha, Y. Lin, L. Saul, and D. Lee. Multiplicative updates for nonnegative quadratic programming. *Neural Comput.*, 19(8):2004–2031, 2007.

[102] F. Sha, L. Saul, and D. Lee. Multiplicative updates for nonnegative quadratic programming in support vector machines. In *Advances in Neural Information Processing Systems 15*, pages 1041–1048. MIT Press, 2002.

[103] D. Strong, J.-F. Aujol, and T. Chan. Scale recognition, regularization parameter selection, and Meyer's G norm in total variation regularization. *Multiscale Modeling and Simulation*, 5(1):273–303, 2006.

[104] D. Strong and T. Chan. Edge-preserving and scale-dependent properties of total variation regularization. *Inverse Problems*, 19(6):S165, 2003.

[105] B. Tomasik, I. Melo, G. Torrieri, S. Vogel, and M. Bleicher. The use of Kolmogorov-Smirnov test in event-by-event analysis. *Nucl. Phys.*, A830:195c–198c, 2009.

[106] M. Tompsett, G. Amelio, and G. Smith. Charge coupled 8-bit shift register. *Applied Physics Letters*, 17(3):111–115, 1970.

[107] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid.* Academic Press, San Diego, U.S.A., 2001.

[108] C. Vogel. A multigrid method for total variation-based image denoising. *Conference Proceedings in Computation and Control IV*, 20, 1995.

[109] Z. Wang and A. Bovik. Mean squared error: Love it or leave it? A new look at signal fidelity measures. *IEEE Signal Processing Magazine*, 26(1):98–117, January 2009.

[110] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.

[111] M. Wasilewski. Active contours using level sets for medical image segmentation. Technical report, University of Waterloo, 2004.

[112] J. Weickert. *Anisotropic Diffusion in Image Processing.* PhD thesis, Universität Kaiserslautern, Germany, 1996.

[113] J. Weickert. A review of nonlinear diffusion filtering. In *SCALE-SPACE '97: Proceedings of the First International Conference on Scale-Space Theory in Computer Vision*, pages 3–28, London, UK, 1997. Springer-Verlag.

[114] J. Weickert. Coherence-enhancing diffusion of colour images. *Image and Vision Computing*, 17(3-4):201–212, 1999.

[115] J. Weickert and H. Scharr. A scheme for coherence-enhancing diffusion filtering with optimized rotation invariance. *J. of Visual Communication and Image Representation*, 13(1-2):103–118, 2002.

[116] J. Weickert, B. ter Haar Romeny, and M. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, 7:398–410, 1998.

[117] R. Wolke and H. Schwetlick. Iteratively reweighted least squares: Algorithms, convergence analysis, and numerical comparisons. *SIAM J. on Scientific and Statistical Computing*, 9(5):907–921, 1988.

[118] C. Xu and J. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transaction on Image Processing*, 7(3):359–369, 1998.

[119] L. Xu and M. Jordan. On convergence properties of the EM algorithm for gaussian mixtures. *Neural Computation*, 8:129–151, 1995.

[120] I. Yavneh. Why multigrid methods are so efficient. *Computing in Science and Engineering*, 8:12–22, 2006.

[121] G. Ye. *Image registration and super-resolution mosaicing.* PhD thesis, School of Information Technology and Electrical Engineering, University of New South Wales at Australian Defence Force Academy, 2005.

[122] R. Zdunek and A. Cichocki. Nonnegative matrix factorization with quadratic programming. *Neurocomput.*, 71:2309–2320, June 2008.

[123] Q. Zhang, H. Wang, R. Plemmons, and P. Pauca. Tensor methods for hyperspectral data analysis: a space object material identification study. *J. of the Optical Society of America A*, 25(12):3001–3012, 2008.

[124] T. Zhang and G. Golub. Rank-one approximation to high order tensors. *SIAM J. on Matrix Analysis and Applications*, 23(2):534–550, 2001.

[125] B. Zitová and J. Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, October 2003.