Spring 2004

Optimization Techniques for Image Restoration

Melissa Pirolli

Optimization Techniques for Image Restoration

A Thesis

Presented to the Faculty

of the Department of Mathematics and Computer Science

McAnulty College and Graduate School of Liberal Arts

Duquesne University

in partial fulfillment of

the requirements for the degree of

Masters of Science in Computational Mathematics

by

Melissa Anne Pirolli

April 16, 2004

Melissa Anne Pirolli

Optimization Techniques for Image Restoration

Masters of Science in Computational Mathematics

April 16, 2004

APPROVED _____
Stacey Levine, Ph.D., Assistant Professor of Mathematics

APPROVED _____
Eric Rawdon, Ph.D., Assistant Professor of Mathematics

APPROVED _____
Kathleen Taylor, Ph.D., Director
Computational Mathematics Program

APPROVED _____
Constance D. Ramirez, Ph.D., Dean
McAnulty College and Graduate School of Liberal Arts

Contents

List of Figures

Acknowledgments

First, I would like thank my adviser, Dr. Stacey Levine, for all her time and support throughout this process. Dr. Levine's patience and guidance helped me to succeed and produce a work that I am proud of. I would like to thank my committee, Dr. Kathleen Taylor and Dr. Eric Rawdon, for all their helpful suggestions and time. Also, I would like to thank the entire department of mathematics and computer science.

I would to recognize Lisa, Jen, Sara and Joe. Thanks for all your support and positive encouragement. I could not have done this without you.

Finally, I would like to thank my family. Thank you for all your love and support. I could have never accomplished this achievement without you believing in me.

Abstract of the Thesis

Optimization Techniques for Image Restoration

by

Melissa Anne Pirolli

Many fields of study use images to make discoveries about the past, decisions for the present and predictions for the future. Images often acquire degradations such as a blur due to a patient moving during an x-ray or noise picked up through remote sensing imaging equipment. Images may also lose information through compression or transmission. In this thesis, diffusion based models were used to solve the image restoration problem as these models can simultaneously remove noise, preserve edges and restore lost information. Specifically, numerical schemes that are more computationally efficient than the current implementation were developed and tested for denoising via nonstandard diffusion. Furthermore, a new model for digital inpainting is proposed based on the nonstandard diffusion model. Numerical results illustrate the effectiveness of both the denoising and inpainting models in image restoration.

Chapter 1

Introduction

1.1 Image Restoration Problem

Many fields of study use images to make discoveries about the past, decisions for the present and predictions for the future. In medicine, doctors study MRI's and x-rays to diagnose patients with cancer, tumors or bone fractures. Geophysicists use images of the earth's strata to determine the location and accessibility of oil. Images often contain degradations such as a blur due to a patient moving during an x-ray or noise picked up through remote sensing imaging equipment. Images may also lose information through compression or transmission. As technology continues to develop, mathematicians are exploring quicker and more efficient ways to restore images which will help doctors, scientists and engineers better solve these critical problems.

In this project we developed models for image restoration. The image restoration problem attempts to recover μ, a true image, from a degraded image, I (see figure 1.1), where $I = K\mu + e$. Here K is a linear operator which could represent blurring and e is additive Gaussian noise with mean 0. In addition, the image may require 'inpainting' if there are subregions, A, of the true image, μ, that have been completely

lost. In this case, we need to recover μ where we only know that $I|_{A^c} = K\mu|_{A^c} + e$ where A^c is the compliment of the subregions A on the entire image domain (see figure 1.2).

Figure 1.1: Denoising
The image on the left is the noisy image and the right is the true image.

Figure 1.2: Inpainting
The image on the left is the image to be inpainted and the right is the true image.

There are numerous mathematical approaches to image restoration, for example, Fourier methods, statistical methods, regularized least squares, etc. In this thesis, diffusion based models were used to solve the image restoration problem since these models can simultaneously smooth noise and preserve edges while also restoring lost information. First, numerical schemes were developed for nonstandard diffusion that are more computationally efficient than the current implementation for denoising.

There is a comparison of the numerical results for both implementations. Next, this new algorithm was used to expand the model to incorporate inpainting.

The remainder of this thesis is laid out as follows. Chapter 2 describes the behavior of the diffusion based models in denoising images. This chapter includes the model for nonstandard diffusion and its current implementation. The third chapter describes the optimization algorithms which were used to implement the nonstandard diffusion model. Chapter 4 describes the precise numerical implementation and experimental results for the nonstandard diffusion denoising model. In the last chapter, the nonstandard diffusion model is applied to inpainting.

Chapter 2

Diffusion based models for Image Restoration

2.1 Images and edges

A two dimensional grayscale image is a function of two variables, $\mu : \Omega \to \mathbb{R}$ where $\Omega \subseteq \mathbb{R}^2$. The domain is the spatial coordinates (or location in the image) (x_1, x_2) and the range is the intensity value $\mu(x_1, x_2)$. The lowest intensity value often corresponds black. As the intensity values increases, the image displays increasingly lighter shades of gray until the largest intensity value corresponds to white.

In image restoration whether removing degradations or restoring lost information, it is important to preserve edges or object boundaries. An edge is defined as a location where there is a large change in intensity values. More specifically, if $\mu(x_1, x_2)$ is an intensity map, then an edge is defined to be locations where the magnitude of the gradient of μ, $|\nabla\mu(x_1, x_2)|$, achieves a local maximum.

2.2 Diffusion based denoising

2.2.1 Isotropic Diffusion

Since the early 1980s, mathematicians have been using isotropic diffusion as a method for image restoration. Isotropic diffusion diffuses equally in all directions. The standard model used for isotropic diffusion is

$$\min_{\mu} \int_{\Omega} |\nabla \mu|^2 \tag{2.1}$$

where $\Omega \subseteq \mathbb{R}^2$ is the image domain. The main benefit of isotropic diffusion is that it removes noise. However, it has a major drawback in that significant features such as edges are not preserved (see figure 2.1).

2.2.2 Total Variation Based Diffusion

In the early 1990s, mathematicians began to use anisotropic diffusion in image restoration. Anisotropic diffusion does not diffuse equally in all directions and can be used to actually preserves edges. In 1992, Rudin, Osher and Fatemi [9] applied total variation (TV) based diffusion, a type of anisotropic diffusion, to image restoration. TV-based diffusion always diffuses strictly in one direction, that which is orthogonal to the gradient. Since the gradient is directed *across* edges, the TV-model diffuses *tangential* to edges. The model for TV-based diffusion is

$$\min_{\mu} \int_{\Omega} |\nabla \mu| \tag{2.2}$$

where $\Omega \subseteq \mathbb{R}^2$ is the image domain. The benefit of this model is that it preserves edges thus making significant features easily identifiable. However, TV-based diffusion does not as efficiently remove noise and may even mistake noise as a significant feature thus creating unwanted artifacts or false edges. False edges can create significant

problems especially when the ultimate goal of the processing is automatic recognition of an object by a computer (see figure 2.2).

2.2.3 Best of Both Worlds

In the late 1990s, Chambolle and Lions introduced the idea of combining isotropic diffusion (2.1) and total variation based diffusion (2.2) in such a way that noise is removed while edges are preserved [2]. The model they introduced is:

$$\min_{\mu} \int_{\Omega} |\nabla \mu|^p \tag{2.3}$$

where $\Omega \subseteq \mathbb{R}^2$ is the image domain and

$$p = \begin{cases} 1, & |\nabla \mu| > \epsilon \\ 2, & |\nabla \mu| \geq \epsilon \end{cases}$$

The main feature of this model is that wherever the gradient is large, $p \equiv 1$, yielding TV-based diffusion and where the gradient is small, $p \equiv 2$, yielding isotropic diffusion. Chambolle and Lion's model is theoretically sound, i.e. there exists a unique solution. However, this model can be very sensitive to the threshold.

Blomgren introduced another model to calculate the exponent p where $1 \leq p \leq 2$. In [1], he set $p = p(|\nabla \mu|)$ where

- $\lim_{|\nabla \mu| \to 0} p(|\nabla \mu|) = 2$
- $\lim_{|\nabla \mu| \to \infty} p(|\nabla \mu|) = 1$ and
- p is monotonically decreasing.

Thus p depends on the strength of the gradient of the image μ and changes as the image μ is updated. With Blomgren's model, the type of anisotropy varies and the numerical simulations showed good results. However, this model is very difficult to study mathematically and there is no guarantee that a solution exists.

Levine, Chan and Stanich introduced a model where $p = p(x)$ depends on the location, in the image [7]. This is the nonstandard diffusion model. The main feature of the nonstandard diffusion is that where the gradient is large, $p(x) \sim 1$ and only TV-based diffusion is used since the location is likely to be an edge. On the other hand, where the gradient is small, $p(x) \sim 2$, only isotropic diffusion is used since the location is likely to be away from the edges, i.e. where there is a small variation in the image. At all other regions, $p(x)$ falls between 1 and 2 and the image is diffused using a combination of isotropic diffusion and TV-based diffusion, thus the type of anisotropy varies. The nonstandard diffusion model is theoretically sound and has good numerical results (see figure 2.3).

2.3 Nonstandard Diffusion

As usual, I is the degraded image and μ is the updated image. In the thesis, we focus on the nonstandard diffusion model proposed by Levine, Chan and Stanich [7]:

$$\min_{\mu} \int_{\Omega} \Psi\left(x, \nabla\mu\right) + \frac{\lambda}{2}\left(\mu - I\right)^2 \tag{2.4}$$

where Ψ is defined by

$$\Psi\left(x, r\right) = \begin{cases} \frac{1}{p(x)}|r|^{p(x)} & \text{if } |r| \leq \epsilon \\ |r| - \frac{p(x)-1}{p(x)} & \text{if } |r| > \epsilon \end{cases} \tag{2.5}$$

The first term in the minimization problem is the diffusion term. The functional $\Psi(x, r)$ in (2.5) was chosen for several reasons. First, when $\epsilon = 1$, $\Psi\left(x, r\right)$ and $\Psi_r\left(x, r\right)$ are continuous and there exists a unique solution to this problem [7]. Furthermore, when the variation in the image is below a given threshold, ϵ, the exponent varies so the type of anisotropy varies. When the variation is above a given threshold, ϵ, TV-based diffusion is used since it is highly likely that the location in the image is at

an edge.

The second term in the minimization problem is the fidelity term. The fidelity term controls how close the updated image, μ, stays to the initial image, I. The parameter λ can vary from image to image, depending on the range of the intensity values.

We will use the $p(x)$ function defined in [7]:

$$p(x) = 1 + \frac{1}{1 + k|\nabla G_\sigma * I|^2} \tag{2.6}$$

where G_σ is the Gaussian filter,

$$G_\sigma(x_1, x_2) = \frac{1}{4\pi\sigma} e^{\frac{\sqrt{x_1^2 + x_2^2}}{4\sigma}}$$

and $k, \sigma > 0$. The convolution of the Gaussian filter and the degraded image I is the solution to the heat equation with initial condition I at a short time σ. This removes a small portion of the noise so it is less likely to be detected as an edge. Since $|\nabla G_\sigma * I|^2$ grows larger near an edge, the second term of $p(x)$ approaches 0 so $\lim_{|\nabla G_\sigma * I| \to \infty} p(x) = 1$. On the other hand, since $|\nabla G_\sigma * I|^2$ grows smaller at locations away from edges, the second term of $p(x)$ approaches 1 so $\lim_{|\nabla G_\sigma * I| \to 0} p(x) = 2$.

2.4 Implementation

There are equivalent formulations of the nonstandard diffusion model defined by equation (2.4). One formulation is to directly solve the minimization problem:

$$\min_{\mu} \int_\Omega I(x, \nabla\mu). \tag{2.7}$$

where $I(x, \nabla\mu) = \Psi(x, \nabla\mu) + \frac{\lambda}{2}(\mu - I)^2$ for the nonstandard diffusion model. A second approach is to solve the Euler-Lagrange equation associated with (2.4). Specifi-

cally, a solution of (2.7) must satisfy

$$\frac{d}{d\epsilon}\frac{I(x,\nabla(\mu+\epsilon v))-I(x,\nabla\mu)}{\epsilon}|_{\epsilon=0}=0$$

Using the calculus of variations, this is equivalent to:

$$divI(x,\nabla\mu)=0. \tag{2.8}$$

Finally, we can approximate the solution using the flow of the Euler-Lagrange equation:

$$\mu_t=divI(x,\nabla\mu). \tag{2.9}$$

In [7], Levine, Chen and Stanich show that as time approaches infinity, $t\to\infty$, the derivative of μ with respect to time approaches 0, $\mu_t\to 0$. So the solution to the flow (2.9) approximates the the solution to the Euler-Lagrange equation (2.8) and the minimization problem (2.7).

2.4.1 Time-marching finite differences schemes

In [7], Levine, Chen and Stanich used finite differences to approximate the flow the Euler-Lagrange equation associated with (2.4). That is,

$$\begin{aligned}\mu_t-\operatorname{div}\left(\Psi_r\left(x,\nabla\mu\right)\right)+\lambda\left(\mu-I\right)=0,\text{ in }\Omega\times[0,T]\\ \frac{\partial\mu}{\partial n}\left(x,t\right)=0,\text{ on }\partial\Omega\times[0,T]\\ \mu(0)=I,\text{ in }\Omega\end{aligned}$$

where Ψ is given in (2.5), $\Psi_r:=\nabla_r\Psi(x,r)$ and $p(x)$ is given in (2.6). To avoid singularities, the diffusion term, $\Psi\left(x,r\right)$, is approximated by

$$\Psi_\beta(x,r) = \begin{cases} \frac{1}{p(x)}\left(\sqrt{|r|^2+\beta^2}\right)^{p(x)} & |r| < \epsilon \\ \sqrt{|r|^2+\beta^2} - \frac{p(x)-1}{p(x)} & |r| \geq \epsilon. \end{cases}$$

The implementation is as follows. Let h represent the spatial step size and Δt represent the time step size. Then, denote $\mu_{ij} = \mu(x_i, y_j)$ and $\mu_{ij}^n = \mu(x_i, y_j, t_n)$ where $x_i = ih$, $y_j = jh$ and $t_n = n\Delta t$. We approximate all of the spacial derivatives using central differences which we denote by:

$$\Delta_x\mu = \frac{\mu_{i+1,j}^n - \mu_{i-1,j}^n}{2h} \qquad \Delta_{xx}\mu = \frac{\mu_{i+1,j}^n - 2\mu_{ij}^n + \mu_{i-1,j}^n}{h^2}$$
$$\Delta_y\mu = \frac{\mu_{i,j+1}^n - \mu_{i,j-1}^n}{2h} \qquad \Delta_{yy}\mu = \frac{\mu_{i,j+1}^n - 2\mu_{ij}^n + \mu_{i,j-1}^n}{h^2}$$
$$\Delta_{xy}\mu = \frac{\mu_{i+1,j+1}^n - \mu_{i+1,j-1}^n - \mu_{i-1,j+1}^n + \mu_{i-1,j-1}^n}{h^2}$$

Using this notation, the diffusion term was approximated by

$$\operatorname{div}((\Psi_\beta)_r(x,\nabla\mu))_{ij}^n =$$
$$\begin{cases} \frac{\left(\Delta_x^2\mu+\Delta_y^2\mu+\beta^2\right)(\Delta_{xx}\mu+\Delta_{yy}\mu)+(p-2)\left(\Delta_x^2\mu\Delta_{xx}\mu+2\Delta_x\mu\Delta_y\mu\Delta_{xy}\mu+\Delta_y^2\mu\Delta_{yy}\mu\right)}{\left(\Delta_x^2\mu+\Delta_y^2\mu+\beta^2\right)^{\frac{4-p}{2}}} & |\nabla\mu| < \epsilon \\ \frac{\beta^2(\Delta_{xx}\mu+\Delta yy\mu)+\Delta_{xx}\mu\Delta_y^2\mu-2\Delta_x\mu\Delta_y\mu\Delta_{xy}\mu+\Delta yy\mu\Delta_x^2\mu}{\left(\Delta_x^2\mu+\Delta_y^2\mu+\beta^2\right)^{\frac{3}{2}}} & |\nabla\mu| \geq \epsilon \end{cases}$$

Using a forward difference to approximate the time derivative ($(\mu_t)_{ij} \approx \frac{\mu_{ij}^{n+1}-\mu_{ij}^n}{\Delta t}$) the scheme can be written

$$\mu_{ij}^{n+1} = \mu_{ij}^n + \Delta t\left(\operatorname{div}((\Psi_\beta)_r(x,\nabla\mu))_{ij}^n + \lambda(\mu_{ij}^n - I_{ij}^n)\right).$$

2.4.2 Optimization Techniques

In this thesis, I implemented the minimization problem (2.4) directly using optimization techniques, specifically the method of steepest descent and the conjugate gradient method. Both techniques are gradient descent methods. The purpose of directly implementing (2.4) was twofold. First, we hoped to develop faster algorithms. Second, (2.4) can be more easily modified to incorporate problems such as inpainting.

Figure 2.1: Isotropic diffusion
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$
(An edge map outlines significant features)

Image plus noise

Image after isotropic diffusion

True image

Figure 2.2: Total variation based diffusion
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$

Image plus noise

Image after total variation based diffusion

True image

Figure 2.3: Nonstandard diffusion
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$

Image plus noise

Image after nonstandard diffusion

True image

Chapter 3

Optimization Techniques

In this chapter, we discuss optimization techniques used to implement (2.4). In general, in order to solve the minimization problem

$$\min_{x \in D} f(x)$$

we need to find

$$x^* = \operatorname*{argmin}_{x \in D} f(x). \tag{3.1}$$

First, we will look at 'line search' methods which are used to minimize one-dimensional functions, i.e. $D = \mathbb{R}$. The search methods we tested were Golden Section method, Fibonacci method, Newton's method and Secant method. Then, we will consider a class of gradient descent methods that minimize more general functions, $f : D \to \mathbb{R}$ where D is a vector space. We will focus on two gradient descent methods for real-valued functions, the method of steepest descent and the conjugate gradient method.

3.1 Line Search Methods

Line search methods are used to locate the value x^* that minimizes the function $f(x)$:

$$x^* = \underset{x \in \mathbb{R}}{\operatorname{argmin}} f(x) \tag{3.2}$$

The Golden Section method and Fibonacci method are the simplest to implement since they do not require any derivatives of f. Newton's method requires the first and second derivative and the Secant method requires the first derivative only. These four methods all assume that the function is unimodal, i.e. there is only one minimizer. Line search methods can be used to determine the optimal step size in the gradient descent methods.

3.2 Gradient Descent Methods

Gradient descent methods follow a general algorithm for minimizing a functional $f : D \to \mathbb{R}$. The goal is to locate the $\mathbf{x}^*$ that minimizes the functional $f(x)$:

$$\mathbf{x}^* = \underset{\mathbf{x} \in D}{\operatorname{argmin}} f(\mathbf{x}) \tag{3.3}$$

The algorithm begins with an initial guess, $\mathbf{x}^{(0)}$. (When applying this to image restoration $\mathbf{x}^{(0)} = I$, the initial degraded image). Then, a descent direction, $\mathbf{d}^{(0)}$, is computed. Using the descent direction and the step size, α, the initial guess moves to a better approximation, $\mathbf{x}^{(1)}$, of the true minimum. Next, the algorithm checks some given stopping criteria, e.g. the squared difference between $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(1)}$. If this difference is 'small enough', then stop. Otherwise, use $\mathbf{x}^{(1)}$ as the new initial guess and repeat the process. Letting the superscript k represent the iteration count, the algorithm is as follows:

1. Start with initial guess, $\mathbf{x}^{(0)} = I$. Then, for $k = 0, 1, 2, ...$

2. Compute a descent direction $\mathbf{d}^{(k)}$ for f at $\mathbf{x}^{(k)}$.

3. Let $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}$, where α_k is either a constant or determined using a line search method where $\alpha_k = \text{argmin}_{\alpha \geq 0} f(\mathbf{x}^{(k)} + \alpha\, \mathbf{d}^{(k)})$.

4. Check stopping criteria. If it is satisfied, stop. If not, go back to step 2.

The method of steepest descent and conjugate gradient method are classified as gradient methods since the gradient is used in determining the descent direction. Since the gradient is the direction of maximal increase of a function at a particular location, the opposite of the gradient is the direction of maximal decrease. The method of steepest descent and conjugate gradient method differ in how they use the gradient to compute the descent direction $\mathbf{d}^{(k)}$.

3.2.1 Method of Steepest Descent

The first method we used to implement to the nonstandard diffusion model was the method of steepest descent. The method of steepest descent uses the opposite of the gradient of the function, $-\nabla f(x)$, as the descent direction. The step size, α_k, is chosen to achieve the maximum amount of decrease at each iteration using a line search, that is,

$$\alpha_k := \underset{\alpha \geq 0}{\text{argmin}}\, f(\mathbf{x}^{(k)} - \alpha \nabla f(\mathbf{x}^{(k)})).$$

The method of steepest descent only differs from the time marching schemes in the choice of the step size α. The time marching schemes (also referred to as 'gradient descent') use a fixed step size, $\alpha = \Delta t$, where steepest descent updates the step size, $\alpha = \alpha_k$, at each iteration. The method of steepest descent is simpler to implement than the conjugate gradient method and guarantees the minimum will be located in at least a finite number of iterations if the minimum exists.

When we implemented the method of steepest descent, we discovered that the implementation worked better with a fixed α value. The problem was that $f(\mathbf{x}^{(k)} -$

$\alpha \nabla f(\mathbf{x}^{(k)}))$ is not unimodal. Depending on the initial guess for each line search, different α values were found for the minimizer. Therefore, the time-marching finite differences schemes (2.10) was found to be superior to the method of steepest descent.

3.2.2 Nonlinear Conjugate Gradient

The nonlinear conjugate gradient method uses the gradient of the function and the direction from the previous iteration to find the current direction of descent. Our implementation of the conjugate gradient method is motivated by quadratic functionals. In the case that the functional is a quadratic:

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} - \mathbf{x}^{\mathrm{T}}\mathbf{b},$$

then, the best search direction at the jth iteration, $\mathbf{d}^{(j)}$ is the $\mathbf{Q}$-conjugate direction, that is, for all $i < j$, $\mathbf{d}^{(i)^{\mathrm{T}}}\mathbf{Q}\,\mathbf{d}^{(j)} = 0$. The conjugate gradient method differs from gradient descent and the method of steepest descent as follows. The direction, $\mathbf{d}^{(0)}$, at the initial iteration is the same descent direction as gradient descent, i.e. $\mathbf{d}^{(0)} = -\nabla f(\mathbf{x}^{(0)})$. After the initial iteration if the stopping criteria is not satisfied, the descent direction is a combination of gradient and the previous descent direction. In particular, $\mathbf{d}^{(k+1)} = -\nabla f(\mathbf{x}^{(k+1)}) + \beta_k\,\mathbf{d}^{(k)}$ where $\beta_k = \frac{\mathbf{g}^{(k+1)^{\mathrm{T}}}\mathbf{Q}\,\mathbf{d}^{(k)}}{\mathbf{d}^{(k)^{\mathrm{T}}}\mathbf{Q}\,\mathbf{d}^{(k)}}$ and $g = \nabla f$. This guarantees that $\mathbf{x}^{(k+1)} < \mathbf{x}^{(k)}$. This algorithm applied to quadratic functionals with n variables converges in n steps.

For non-quadratic functional such as that used in (2.4), α can either be fixed or determined using a line search and the Hestenes-Stiefel formula and Fletcher-Reeves formula [5] can be used to determine β. Denoting $g = \nabla f$:

The Hestenes-Stiefel formula

$$\beta_k = \frac{g^{(k+1)^{\mathrm{T}}}[g^{(k+1)} - g^{(k)}]}{d^{(k)^{\mathrm{T}}}[g^{(k+1)} - g^{(k)}]} \tag{3.4}$$

The Fletcher-Reeves formula

$$\beta_k = \frac{g^{(k+1)^{\mathrm{T}}} g^{(k+1)}}{g^{(k)^{\mathrm{T}}} g^{(k)}} \tag{3.5}$$

Therefore, $\mathbf{d}^{(k+1)} = -\nabla f(\mathbf{x}^{(k+1)}) + \beta_k \mathbf{d}^{(k)}$. The Hestenes-Stiefel formula is recommended when the line search is known to be inaccurate [6]. In a study by Powell in [8], a global convergence analysis suggests that the Fletcher-Reeves formula for β_k is superior.

For nonquadratic problems, there is no guarantee that the functional will converge in n steps. Therefore, as the algorithm progresses, it is a common practice to reinitialize the direction vector to the opposite of the gradient so the directions continue to be "Q-conjugate". The nonlinear conjugate gradient method is more complex to implement than the gradient descent method but converges faster as we will see experimentally in chapter 4.

Chapter 4

Numerical Implementation

In this chapter, we discuss the numerical implementation of the minimization problem

$$\min_{\mu \in BV \cap L^2(\Omega)} F(\mu)$$

where

$$F(\mu) = \int_\Omega \Phi(\nabla \mu) + \frac{\lambda}{2} (\mu - I)^2 \tag{4.1}$$

where $\int_\Omega \Phi(\nabla \mu)$ is the diffusion term and $\int_\Omega (\mu - I)^2$ is the fidelity term. The domain of the minimization problem is $BV \cap L^2(\Omega) := \{\mu | \int_\Omega |\mu|^2 < \infty \text{ and } \int_\Omega |\nabla \mu| < \infty\}$. The two special cases we will implement are total variation based diffusion where $\Phi(\nabla \mu) = |\nabla \mu|$ and nonstandard diffusion where $\Phi(\nabla \mu) = \Psi(x, \nabla \mu)$ as defined in (2.5). As usual, I is the degraded image and μ represents the updated image.

4.1 One Dimensional Implementation

Suppose that μ is the noisy one dimensional signal that we approximate using $n+1$ data points. Specifically, we approximate μ by $\boldsymbol{\mu} = (\mu_0, \ldots, \mu_n)$ where each μ_i represents the intensity value at the ith location for $i = 0 \ldots n$. Let $\Delta x = 1/n$ and

$D_i = [0, \ldots, 0, -1/\Delta x, 1/\Delta x, 0, \ldots, 0]$. The derivative of $\boldsymbol{\mu}$ can then be approximated by

$$D_i \, \boldsymbol{\mu} = \frac{\mu_i - \mu_{i-1}}{\Delta x} \tag{4.2}$$

.

For simplicity, we will let $J(\mu)$ represent the diffusion term:

$$J(\mu) = \int_\Omega \Phi(\nabla \mu).$$

We then can rewrite $J(\mu)$ in the discretized form

$$J(\boldsymbol{\mu}) \approx \frac{1}{2} \sum_{i=1}^{n} \Phi\left(\left(D_i \boldsymbol{\mu} \right)^2 \right) \Delta x.$$

Minimizing (4.1) using gradient descent methods requires computing the gradient of J. We do so as follows. For any $\mathbf{v} \in \mathbb{R}^n$,

$$\begin{aligned} \frac{d}{d\tau} J(\boldsymbol{\mu} + \tau \mathbf{v}) &= \sum_{i=1}^{n} \Phi'\left([D_i \boldsymbol{\mu}]^2 \right) \, (D_i \boldsymbol{\mu}) \, (D_i \mathbf{v}) \, \Delta x \\ &= \Delta x \, (D\boldsymbol{\mu})^T \, \mathrm{diag}(\Phi'(\boldsymbol{\mu})) \, (D\boldsymbol{\mu})) \\ &= \langle \Delta x \, D^T \, \mathrm{diag}(\Phi'(\boldsymbol{\mu})) \, D\boldsymbol{\mu}, \mathbf{v} \rangle \end{aligned}$$

where

- $\mathrm{diag}(\Phi'(\boldsymbol{\mu}))$ represents the $n \times n$ matrix whose ith diagonal entry is $\mathrm{diag}(\Phi'(D_i \boldsymbol{\mu}))$
- D represents the $n \times (n+1)$ matrix whose ith row is D_i
- $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product

Using (4.3), the gradient of the diffusion term, J is given by.

$$\mathrm{grad} \, J(\boldsymbol{\mu}) = \left(\Delta x \, D^{\mathrm{T}} \, \mathrm{diag}(\Phi'(\boldsymbol{\mu})) \, D \right) \, \boldsymbol{\mu} \tag{4.3}$$

Approximating the functional Φ*:*

The Euclidean norm, $|x|$, is non-differentiable at the origin since $\frac{d}{dx}|x| = \frac{x}{|x|}$, so we must approximate the diffusion term for both the TV-based diffusion and the nonstandard diffusion models. The diffusion term for TV-based diffusion, $\Phi(\nabla \boldsymbol{\mu}) = |\nabla \boldsymbol{\mu}|$ is approximated by

$$\Phi_\beta(\nabla \boldsymbol{\mu}) = \sqrt{|\nabla \boldsymbol{\mu}|^2 + \beta^2}. \tag{4.4}$$

The diffusion term for nonstandard diffusion,

$$\Phi\left(x, \nabla \boldsymbol{\mu}\right) = \Psi\left(x, \nabla \boldsymbol{\mu}\right) = \begin{cases} \frac{1}{p(x)} |\nabla \boldsymbol{\mu}|^{p(x)} & \text{if } |\nabla \boldsymbol{\mu}| \leq \epsilon \\ |\nabla \boldsymbol{\mu}| - \frac{p(x)-1}{p(x)} & \text{if } |\nabla \boldsymbol{\mu}| > \epsilon \end{cases}$$

is approximated by

$$\Phi_\beta(x, \nabla \boldsymbol{\mu}) = \begin{cases} \frac{1}{p(x)} \left(\sqrt{|\nabla \boldsymbol{\mu}|^2 + \beta^2}\right)^{p(x)} & \text{if } |\nabla \boldsymbol{\mu}| \leq \epsilon \\ \sqrt{|\nabla \boldsymbol{\mu}| + \beta^2} - \frac{p(x)-1}{p(x)} & \text{if } |\nabla \boldsymbol{\mu}| > \epsilon \end{cases} \tag{4.5}$$

where β is a small positive parameter. In each case, Φ_β is differentiable everywhere. From (4.1) and (4.3), we obtain the gradient of $F(\boldsymbol{\mu})$ in one dimension,

$$\operatorname{grad} \mathrm{F}(\boldsymbol{\mu}) = \operatorname{grad} J_\beta(\boldsymbol{\mu}) + \lambda(\boldsymbol{\mu} - \mathbf{I}) \tag{4.6}$$

where

$$J_\beta(\boldsymbol{\mu}) = \int \Phi_\beta(\nabla \boldsymbol{\mu}).$$

The optimization algorithm is then $\boldsymbol{\mu}^{(k+1)} = \boldsymbol{\mu}^{(k)} + \alpha \mathbf{d}^{(k)}$. For gradient descent, $\mathbf{d}^{(k)} = -\text{grad}F^{(k)}$ and for conjugate gradient, $\mathbf{d}^{(k)} = -\text{grad}F^{(k)} + \beta_k \mathbf{d}^{(k-1)}$.

4.2 Results

Using the gradient calculations above for the TV model (2.2) and the nonstandard diffusion model (2.4), figure 4.1 shows an example of one-dimensional noise removal on a piecewise constant image. Figure 4.1(a) is the true image and figure 4.1(b) is the noisy image, i.e. true image plus additive noise. Figure 4.1(b) was restored by the following implementations:

- TV-based diffusion and gradient descent (figure 4.1(c))
- Nonstandard diffusion and gradient descent (figure 4.1(d))
- Nonstandard diffusion and (HSCG) (figure 4.1(e))
- Nonstandard diffusion and (FRCG) (figure 4.1(f)).

For simplicity, HSCG states for Hestenes-Stiefel conjugate gradient and FRCG states for Fletcher-Reeves conjugate gradient. Each implementation used the parameters that gave the optimal results. The number of iterations was chosen by studying the change in the functional values. When this change was 'small enough' i.e. $||\boldsymbol{\mu}^{(k+1)} - \boldsymbol{\mu}^{(k)}|| < \delta$, the algorithm stopped.

TV-based diffusion

As discussed in Chapter 1, the major drawback of TV-based diffusion is that false edges that may be created. Figure 4.1(c) shows an example of noise being detected as false edges and creating 'step edges', sometimes called the 'staircase effect'. This image ran through 1000 iterations.

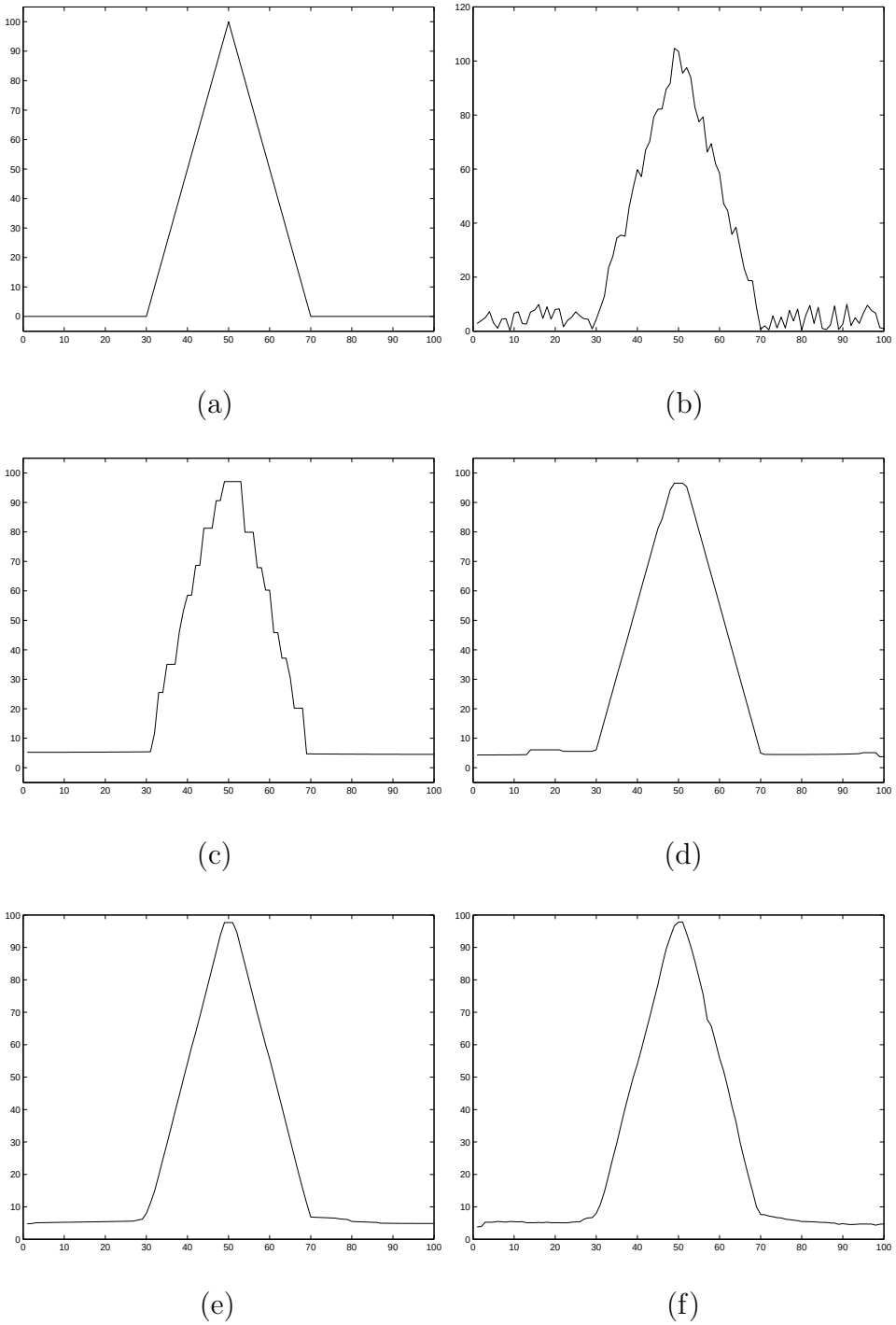
Nonstandard Diffusion

Each of the nonstandard diffusion implementations (figures 4.1(d), (e) and (f)) gave good numerical results. The noise was removed to reconstruct the smooth lines and the three corners in the image are still easily identifiable. For the three nonstandard implementations, each image ran through the following number of iterations:

- Figure 4.1(d): 1000 iterations
- Figure 4.1(e): 350 iterations
- Figure 4.1(f): 100 iterations

The conjugate gradient implementations (4.1(e) and 4.1(f)) converged faster than the gradient descent implementation (4.1(d)). The Fletcher-Reeves implementation converged faster than Hestenes-Stiefel implementation (100 to 350 iterations) for the piecewise constant one-dimensional noisy image.

Figure 4.1: One Dimensional Image



(a) True Image, ; (b) Noisy Image; (c) 1000 iterations of TV based diffusion(gradient descent); (d) 1000 iterations of Nonstandard diffusion(gradient descent); (e) 350 iterations of Nonstandard diffusion (HSCG); (f) 100 iterations of Nonstandard diffusion(FRCG)

4.3 Two Dimensional Implementation

We now consider minimizing this functional (4.1) in two dimensions. Suppose $\boldsymbol{\mu} = [\mu_{ij}]$ is a two-dimensional discretized image. Here, $\mu_{ij} = \mu_{ij}(x_i.y_j)$ where $x_i = i\nabla x$, $y_j = j\nabla y$, $i = 0, \ldots, n_x$, $j = 0, \ldots, n_y$. J can be approximated using the discretized form

$$J(\boldsymbol{\mu}) \approx \frac{1}{2} \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \Phi\left((D_{ij}^x \boldsymbol{\mu})^2 + (D_{ij}^y \boldsymbol{\mu})^2\right)$$

where

$$D_{ij}^x \boldsymbol{\mu} = \frac{\mu_{i.j} - \mu_{i-1.j}}{\Delta x}, \qquad D_{ij}^y \boldsymbol{\mu} = \frac{\mu_{i.j} - \mu_{i.j-1}}{\Delta y}.$$

To calculate the gradient of J, we use a similar computation to that in one dimension:

$$\begin{aligned} \frac{d}{d\tau} J(\boldsymbol{\mu} + \tau \mathbf{v})|_{\tau=0} &= \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \Phi'_{ij} \left[(D_{ij}^x \boldsymbol{\mu})(D_{ij}^x \mathbf{v}) + (D_{ij}^y \boldsymbol{\mu})(D_{ij}^y \mathbf{v})\right] \\ &= \langle \operatorname{diag}\left(\Phi'(\boldsymbol{\mu})\right) D_x \boldsymbol{\mu}, D_x \mathbf{v} \rangle + \langle \operatorname{diag}\left(\Phi'(\boldsymbol{\mu})\right) D_y \boldsymbol{\mu}, D_y \mathbf{v} \rangle \end{aligned}$$

where $\Phi'_{ij} = \Phi'\left((D_{ij}^x \boldsymbol{\mu})^2 + (D_{ij}^y \boldsymbol{\mu})^2\right)$ and D_x and D_y are matrices corresponding to the grid operators in (4.7). Therefore, the gradient of the diffusion term, J, is:

$$\operatorname{grad} J(\boldsymbol{\mu}) = \left(\left(D_x^{\mathrm{T}} \operatorname{diag}(\Phi'_\beta(\boldsymbol{\mu}))\, D_x\right) + \left(D_y^{\mathrm{T}} \operatorname{diag}(\Phi'_\beta(\boldsymbol{\mu}))\, D_y\right)\right) \boldsymbol{\mu} \tag{4.7}$$

where Φ_β is the approximation of Φ in equations (4.4) and (4.5) respectively. From (4.1) and (4.7), the gradient of $F(\boldsymbol{\mu})$ for two dimensions is obtained the same as in one dimension (4.6).

4.4 Results

Using the gradient calculations above for the TV model (2.2) and the nonstandard diffusion model (2.4), we tested both models on a piecewise constant image (see figure 4.2) and a real piecewise smooth image (see figure 4.4). Nonstandard diffusion is successful in restoring both piecewise constant and piecewise smooth regions. Our hypothesis was that directly implementing the minimization problem (2.4) with optimization techniques will restore a degraded image faster than implementing the flow of the Euler-Lagrange equation associated with the minimization problem. The results included here are purely experimental. In the future, based on the conclusions from our numerical simulations, the convergence times can be studied and compared in general.

In the remainder of this thesis, for each implementation the parameters chosen were the ones which gave the optimal results and the stopping criteria was when $||\boldsymbol{\mu}^{(k+1)} - \boldsymbol{\mu}^{(k)}|| < \delta$. We found that the difference in CPU time per iteration for the different methods was negligible. For the conjugate gradient implementation, the results displayed are for the Fletcher-Reeves implementation of β. The Fletcher-Reeves implementation performed better than the Hestenes-Stiefel implementation in that it consistently gave more efficient convergence. The Fletcher-Reeves implementation in two dimensions reinitialize the direction vector $\mathbf{d}^{(k)}$ to the opposite of the gradient, $-\nabla F$, at any iteration where the calculation of the β_k value was negative.

4.4.1 Piecewise constant image

Gradient Descent

In figure 4.2, the gradient descent implementation was used to denoise a piecewise constant image. The initial noisy image is in 4.2(a), the restored image using TV-based diffusion is in 4.2(b), the restored image using nonstandard diffusion is in 4.2(c) and

the true image is in 4.2(d). When testing gradient descent with nonstandard diffusion on the piecewise constant image (4.2(c)), the image converged after 85 iterations. The numerical results were very promising since this final image shows smooth non-noisy regions as well as sharp edges. Using the same algorithm but now implementing TV-based diffusion, we ran the noisy image for 85 iterations. Figure 4.2(b) shows that the convergence time for TV-based diffusion is slower than the nonstandard diffusion, since the image reconstructed with the TV-model still appears noisy.

Conjugate Gradient

In figure 4.3, the conjugate gradient implementation was used to reconstruct the same image. The initial noisy image is in 4.3(a), the restored image using TV-based diffusion is in 4.3(b), the restored image using nonstandard diffusion is in 4.3(c) and the true image is in 4.3(d). The initial noisy image is the same initial noisy image used in figure 4.2. For the conjugate gradient algorithm, the numerical results are given for nonstandard diffusion and TV based diffusion. TV-based diffusion (4.3(b)) and nonstandard diffusion (4.3(c)) implementations were run for 24 iterations because at this point, the nonstandard diffusion model converged to the true image. Figure 4.3(c) appears to be almost identical to the true image, 4.3(d), after 24 iterations versus 4.3(b) which still appears noisy.

When comparing the gradient descent and conjugate gradient implementations for nonstandard diffusion for the piecewise constant image, the conjugate gradient implementation (figure 4.3(c)- 24 iterations) converged faster than the gradient descent implementation (figure 4.2(c)-85 iterations) which supports our original hypothesis.

4.4.2 Piecewise smooth image

Figures 4.4, 4.5, 4.6 and 4.7 demonstrate reconstructions of a piecewise smooth image, Lenna. The stopping criteria was based on the edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$, and gradient

map, $\nabla\mu$. The models continued running iterations until the edge map was not displaying noise and the gradient map appeared to represent the outline of the objects in the image. In this case, the stopping criteria could not be based on the updated image alone because of the loss of textures, i.e. Lenna's hair and scarf. The final gradient map does not represent the gradient map of the true image because the textures merged as the noisy image was updated. Each figure displays the initial degraded image (a), the updated image at about 1/3 of the total iterations (b), the updated image at about 2/3 the total iterations (c), the final restored image (e) and the true image (f).

Gradient Descent

The gradient descent implementation of TV-based diffusion (see figure 4.4) required 450 iterations until the image converged. The most noticeable change in the updated images are between 150 and 300 iterations (4.4(b) and (c), respectively). After 300 iterations, the updated images have a very small change at each iteration. In the final edge map (4.4(d)), there is some noise still appearing. Since the change between iterations is small, it can be assumed that the TV-model has detected this noise to be an edge thus has created and is preserving false edges. The TV-model kept sharp edges and significant features such as Lenna's lips and eyes which could have been easily lost or blurred.

For the gradient descent implementation of nonstandard diffusion, (see figure 4.5), the noisy image, 4.5(a), ran through 300 iterations until the image was restored (4.5(d)). At this point, 4.5(d) appears to represent the true image, 4.5(e), with only a difference in the appearance of the textures. The gradient in 4.5(d) represents a good outline of the objects and boundaries in Lenna and the edge map shows that the noise has been removed. The edge map shows good results since it is still defining significant little features such as Lenna's lips and eyes which could have been easily

lost. From studying the three outputs at 100, 200 and 300 iterations, figures 4.5(b), (c) and (d), respectively, it seems that the percent of the image restored at each output is about the same. This is different from what was seen with the TV-based diffusion where there was a large change between 4.4(b) and (c).

The gradient descent implementation of nonstandard diffusion (see figure 4.5) restored the same noisy image faster than TV-based diffusion (see figure 4.4), i.e. 300 iterations versus 450 iterations. This again supports our hypothesis.

Conjugate Gradient

For the conjugate gradient implementation of TV-based diffusion, (see figure 4.6), the noisy image, 4.6(a), ran for 425 iterations until the image was restored (4.6(d)). At this point, there was sufficiently small change in the images and the edge map between iterations. After 150 iterations, figure 4.6(b) appears to have little to no change. The most noticeable change in the updated images are between 275 and 425 iterations (4.6(c) and (d), respectively). After 425 iterations, the updated images have a very small change at each iteration. In the final edge map (4.6(d)), there is some noise still appearing. As with the gradient descent implementation of conjugate gradient (4.4), it appears that the TV-model still preserves true edges, but has also detected some noise as edges and created false edges.

For the conjugate gradient implementation of nonstandard diffusion, (see figure 4.7), the noisy image, 4.7(a), ran for 140 iterations until the image was restored (4.7(d)). At this point, figure 4.7(d) appears to represent the true image, 4.7(e), with only a difference in the appearance of the textures. The gradient in 4.7(d), just like the gradient descent implementation of nonstandard diffusion (figure 4.5), represents a good outline of the objects in Lenna and the edge map shows that noise has been removed while still defining significant little features. Again, similar to the gradient descent, at each of the three outputs, i.e 50, 100 and 140 iterations, figures 4.7(b),

(c) and (d), respectively, the same portion of the image is restored.

The conjugate gradient implementation of nonstandard diffusion (see figure 4.7) restored the same noisy image faster than TV-based diffusion (see figure 4.6), i.e. 140 iterations versus 425 iterations. This again supports our hypothesis.

Comparison

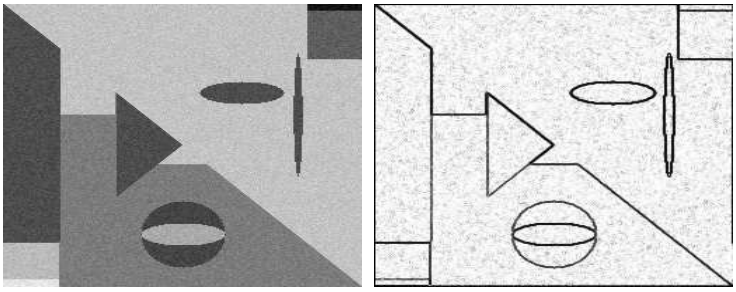
Figure 4.8 compares the four implementations. In figure 4.8, the first column is the gradient descent implementation of TV-based diffusion, the second column is the gradient descent implementation of nonstandard diffusion, the third column is the conjugate gradient implementation of TV-based diffusion and the fourth column is the conjugate gradient implementation of nonstandard diffusion. The same initial noisy image was used and the updated images are displayed for iteration 0, 50, 100 and 140. These outputs were chosen since they yielded the optimal results for the conjugate gradient implementation of the nonstandard model. From figure 4.8, both images implemented using nonstandard diffusion (columns two and four) provide better and faster results for noise removal while preserving edges than TV-based diffusion (columns one and three). These results again support our hypothesis that the conjugate gradient implementation (column four) restores noisy images faster than the gradient descent implementation (column two).

More experimental results

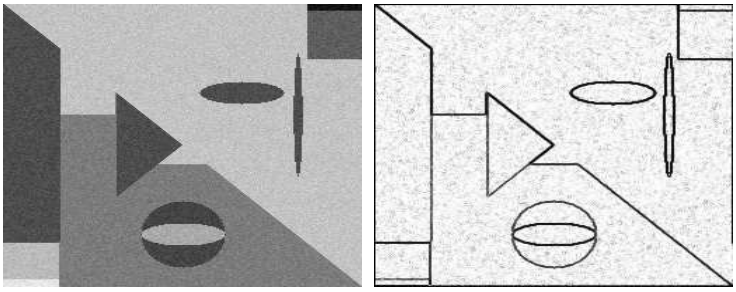
Since the conjugate gradient implementation for nonstandard diffusion had the faster convergence for denoising, I have included three more images restored using this implementation (figures 4.9, 4.10 and 4.11). Figure 4.9(a) is a MRI image of a brain where additive noise was added using the Gaussian distribution with mean 0. After 300 iterations (4.9(b)), the algorithm converged. Figure 4.9(b) has good edge detection, especially outlining the tumor in the center of the brain. Figure 4.10(a)

is a PET image of a lung. After 250 iterations (4.10(b)), the algorithm converged. Finally, figure 4.11(a) is an image of a lake with additive noise. As 225 iterations (4.11(b)), the algorithm converged. In figure 4.11(b), some of the textures in the trees were lost but the outline of the clouds in the sky and the boat in the water can still be seen thus the edges were preserved while the noise was removed.

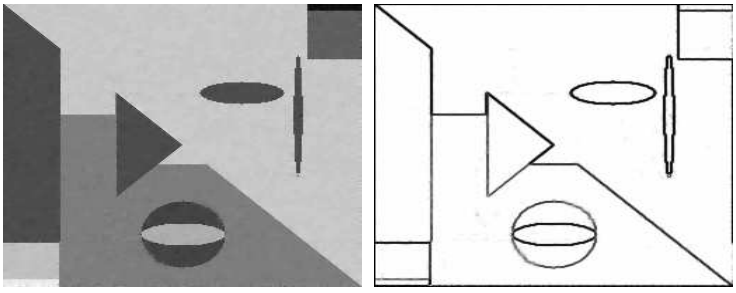
Figure 4.2: Gradient Descent on a piecewise constant image
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$



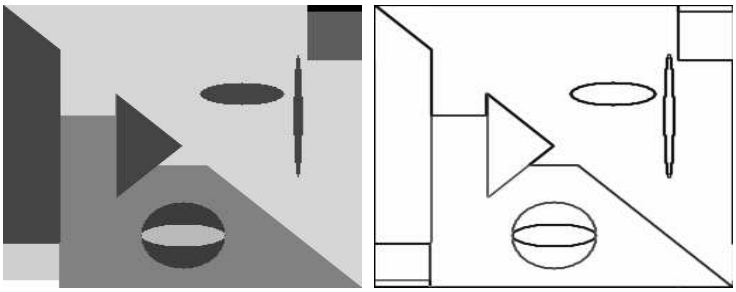
(a) Image plus noise



(b) TV-based diffusion

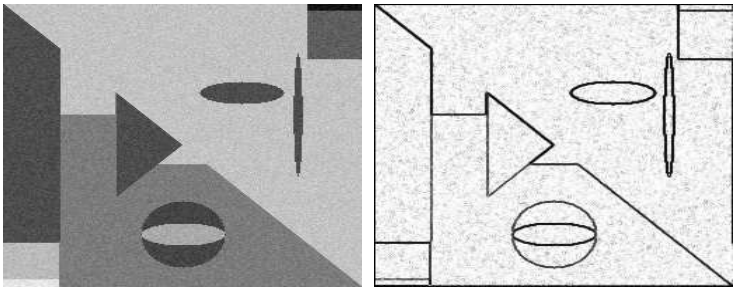


(c) Nonstandard Diffusion

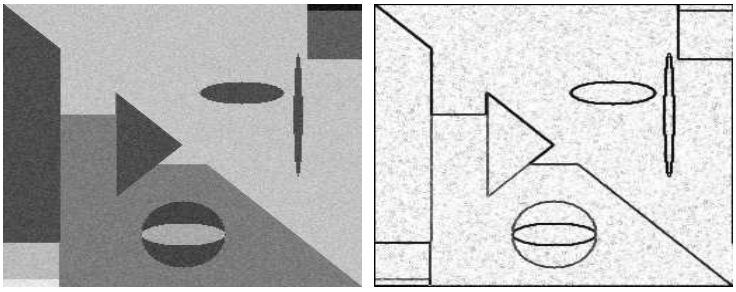


(d) True image

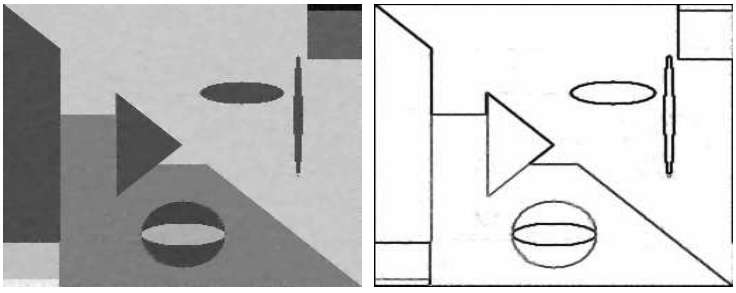
Figure 4.3: Conjugate Gradient on a piecewise constant image
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$



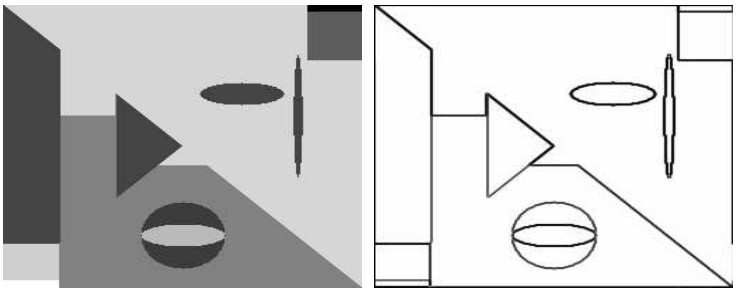
(a) Image plus noise



(b) TV-based Diffusion



(c) Nonstandard Diffusion



(d) True image

Figure 4.4: TV-based diffusion (gradient descent) on Lenna
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$; gradient, $\nabla\mu$

(a) 0 iterations

(b) 150 iterations

(c) 300 iterations

(d) 450 iterations

(e) True image

Figure 4.5: Nonstandard diffusion (gradient descent) on Lenna
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$, gradient $\nabla\mu$

(a) 0 iterations

(b) 100 iterations

(c) 200 iterations

(d) 300 iterations

(e) True image

Figure 4.6: TV-based diffusion (conjugate gradient) on Lenna
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$, gradient $\nabla\mu$

(a) 0 iterations

(b) 150 iterations

(c) 275 iterations

(d) 425 iterations

(e) True image

Figure 4.7: Nonstandard diffusion (conjugate gradient) on Lenna
Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$, gradient $\nabla\mu$

(a) 0 iterations

(b) 50 iterations

(c) 100 iterations

(d) 140 iterations

(e) True image

Figure 4.8: Comparison of implementations
Left to right: TV (gradient descent); NS (gd); TV (cg); NS (cg)

0 iterations

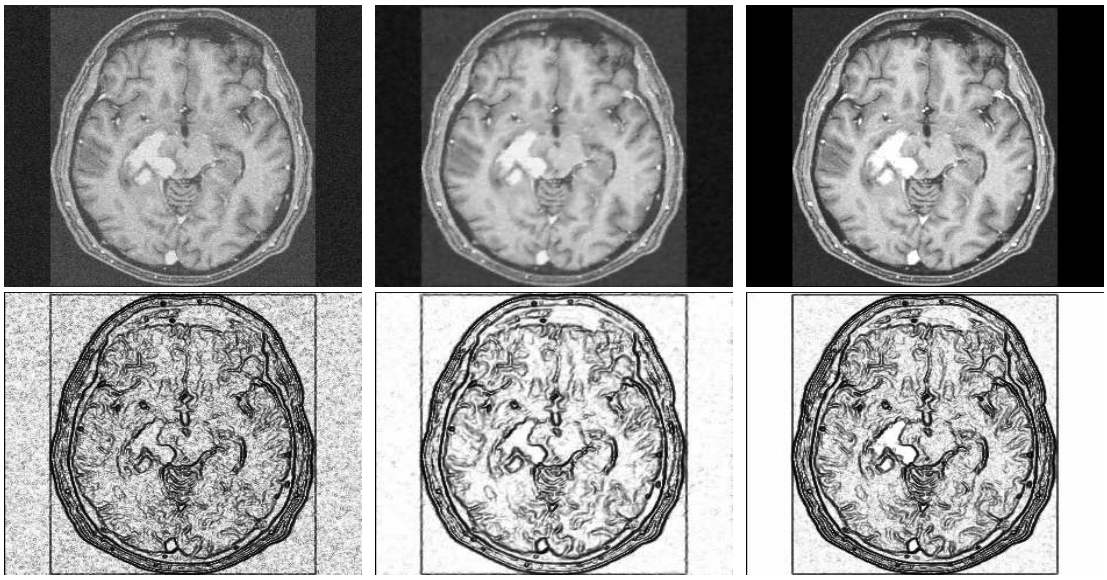
0 iterations

50 iterations

100 iterations

140 iterations

Figure 4.9: Nonstandard diffusion (conjugate gradient) on MRI image
Top to bottom: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$

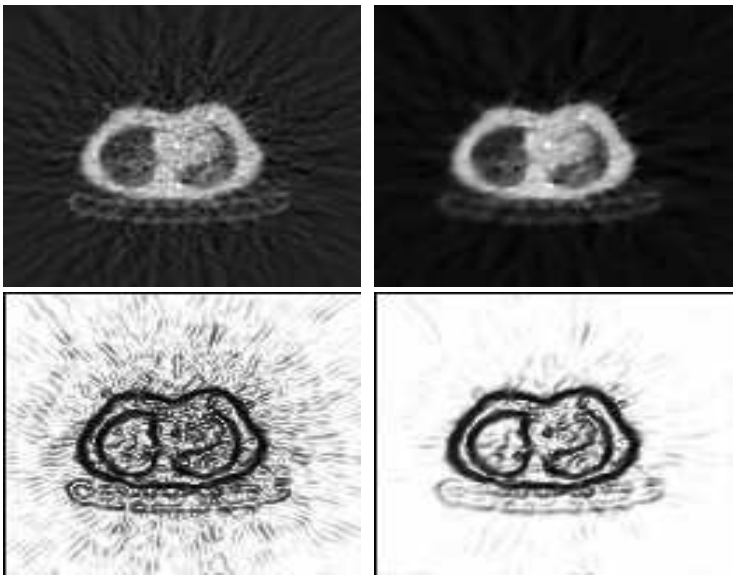


(a) Noisy image

(b) 300 iterations

(c) True image

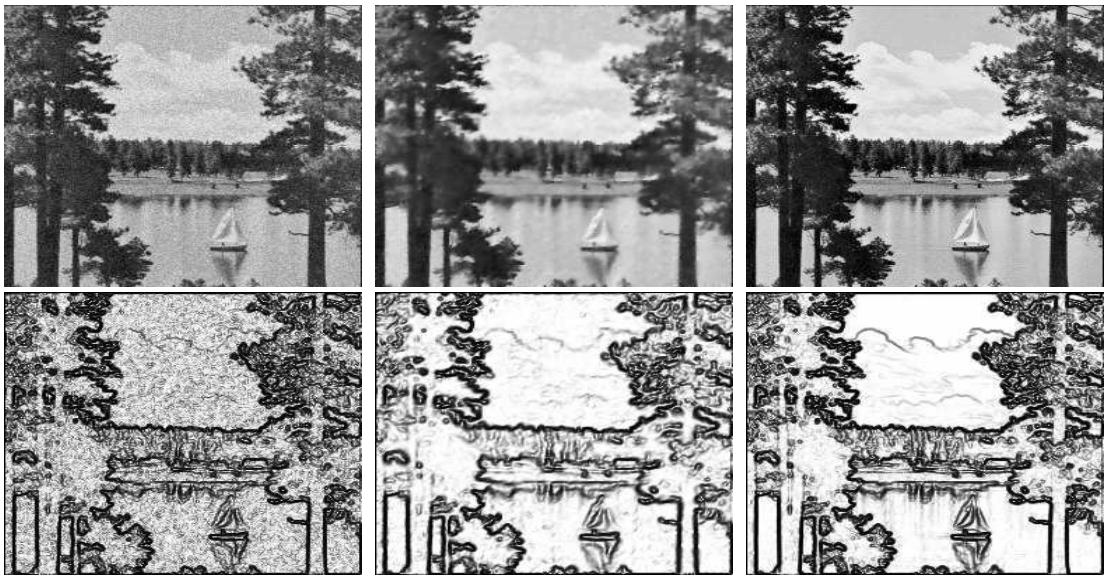
Figure 4.10: Nonstandard diffusion (conjugate gradient) on a lung
Top to bottom: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$



(a) (b)

Noisy image 250 iterations

Figure 4.11: Nonstandard diffusion (conjugate gradient) on a lake
Top to bottom: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$



(a) Noisy image

(b) 225 iterations

(c) True image

Chapter 5

Inpainting

5.1 The Inpainting Model

The term 'inpainting' comes from art restoration and is the process of restoring ancient paintings which have been damaged, aged or flawed by some other factor. This inspired the term digital inpainting, which is the process of filling in missing information or 'lost packets' within an image where the domain of these lost packets is known. Chan and Shen [3] developed the inpainting model, Curvature-Driven Diffusion (CDD), based on the TV based diffusion model by Rudin, Osher and Fatemi [9]. Chan and Shen make note that one of the major drawbacks of inpainting models are the inability to restore edges when 'packets' lie across these large intensity value changes or jumps. We use the same idea as Chan and Shen in [3] and [4] for inpainting using the nonstandard diffusion model (2.4). Let Ω be the image domain and A be the domain where information is missing. We propose the following inpainting model:

$$\min_{\mu} \int_{\Omega} G_A(x, |\kappa|)\Psi\left(x, \nabla\mu\right) + \frac{\lambda_A}{2}\left(\mu - I\right)^2 \tag{5.1}$$

where Ψ is the diffusion term defined in (2.5). The factor in front of the diffusion term is

$$G_A(x, |\kappa|) := \begin{cases} a & \text{if } x \in \Omega - A \\ |\kappa| & \text{if } x \in A \end{cases}$$

where

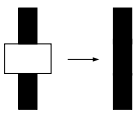
$$\kappa = \text{div}\frac{\nabla\mu}{|\nabla\mu|}$$

is the mean curvature of μ. The coefficient of the fidelity term is

$$\lambda_A := \begin{cases} \lambda & \text{if } x \in \Omega - A \\ 0 & \text{if } x \in A. \end{cases}$$

In $\Omega - A$, if denoising is required, we set $a \equiv 1$ and (5.1) is just the nonstandard diffusion model (2.4). If denoising is not required, we can set $a \equiv 0$. In the inpainting regions, A, G_A is equal to the absolute value of curvature, $|\kappa|$. This encourages inpainting in missing regions which lie across an edge, since curvature is very large at corners (see figure 5.1). Furthermore, curvature is also extremely high at noise, which is the initial random guess inside the inpainting regions. Thus, the diffusion

Figure 5.1: Inpainting
The left image requires inpainting and the right is the true image.



is stronger inside A since Ψ is multiplied by $|\kappa|$. As the region is inpainted, the curvature will begin to decrease, eventually slowing the diffusion. Finally, the fidelity term is not needed in the inpainting region, A, and is thus removed.

5.2 Results

Since the conjugate gradient implementation of nonstandard diffusion yields the best results, the inpainting model 5.1 was only tested using this implementation. For the rest of the thesis, all outputs are tested only using the Fletcher-Reeves conjugate gradient implementation of nonstandard diffusion. There are four images that inpainting was applied to:

- Piecewise constant image - figure 5.2
- Piecewise smooth image - figure 5.3
- Piecewise constant noisy image - figure 5.4
- Piecewise smooth noisy image - figure 5.5.

Inpainting

In figures 5.2 and 5.3, (a) is the image with missing regions, (b) is the restored image and (c) is the true image.

In figure 5.2(a), almost all of the missing regions have been restored after 229 iterations using $a = 1$. These results are best seen in the edge maps corresponding to the restored image and the true image, figures 5.2(b) and 5.2(c), respectively. There are two missing regions which covered an edge. We had good results with inpainting on these regions since the edges were restored.

The piecewise smooth image, Lenna, in figure 5.3(a) required 5000 iterations before the inpainting was complete. The convergence was much slower, since in this case we set $a = 0$ in order to preserve the more intricate details such as textures. It ap-

pears from figure 5.3(b) that inpainting smooth regions was very successful. However, inpainting fine geometric structures such as edges is much more difficult.

In both the piecewise constant and piecewise smooth images, the use of nonstandard diffusion instead of TV-based diffusion helps to prevent false edges from being created in or around the inpainting regions.

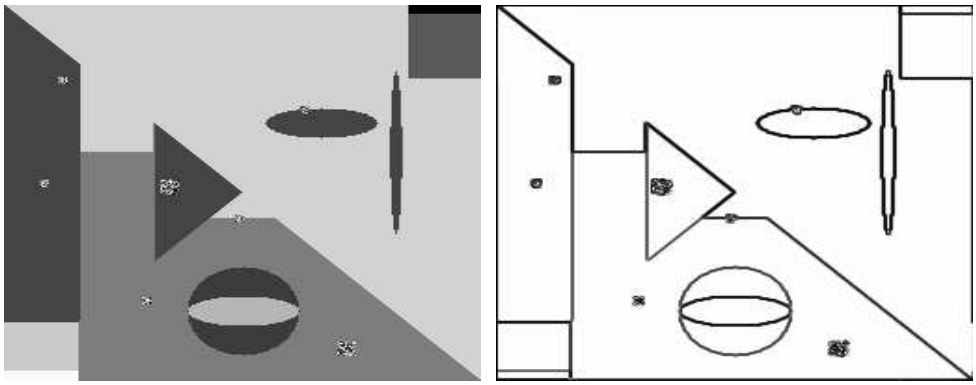
Inpainting and Denoising

In figures 5.4 and 5.5, (a) is the noisy image with missing regions, (b) is the restored image and (c) is the true image.

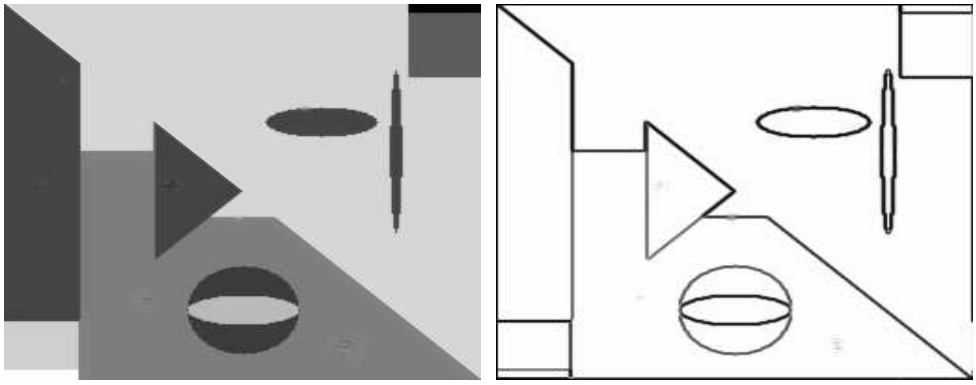
Figure 5.4(a) ran for 100 iterations using denoising only (2.4) and then ran for 11000 iterations using inpainting (5.1) with $a = 0$ until the image was optimal (figure 5.4(b)). In figure 5.4(b), the noise has been removed and the missing regions have been completed. The inpainting for figure 5.4 was not as successful as the results in figure 5.2 due to the edge reconstructions but in figure 5.4 the noise was removed, the missing regions are correctly filled in and false edges have not been created. Thus, it appears that denoising and inpainting a piecewise constant image was successful.

Figure 5.5(a) ran for 250 iterations using denoising only (2.4) and then ran for 5000 iterations using inpainting (5.1) with $a = 0$ until the image was optimal (figure 5.5(b)). In figure 5.5(b), the noise has been removed and the missing regions have been completed. Thus, it appears that denoising and inpainting a piecewise smooth image was successful. From the edge map corresponding with 5.5(b), some of the significant features, i.e. lenna's hair, are being lost due to the denoising. Similar to the results for figure 5.3, the edges have not been reconstructed in the inpainted regions but at the same time false edges have not been created.

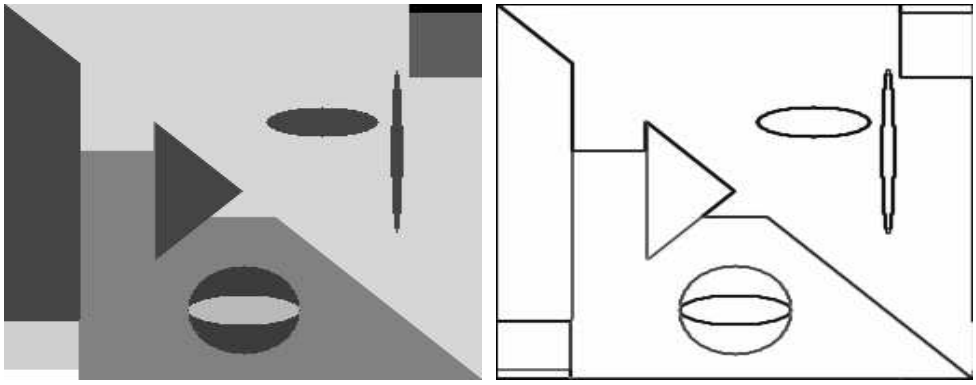
Figure 5.2: Inpainting on a piecewise constant image



(a)



(b)



(c)

Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$
(a) Image with missing regions;
(b) Conjugate Gradient: 229 iterations;
(c) True Image

Figure 5.3: Inpainting on Lenna

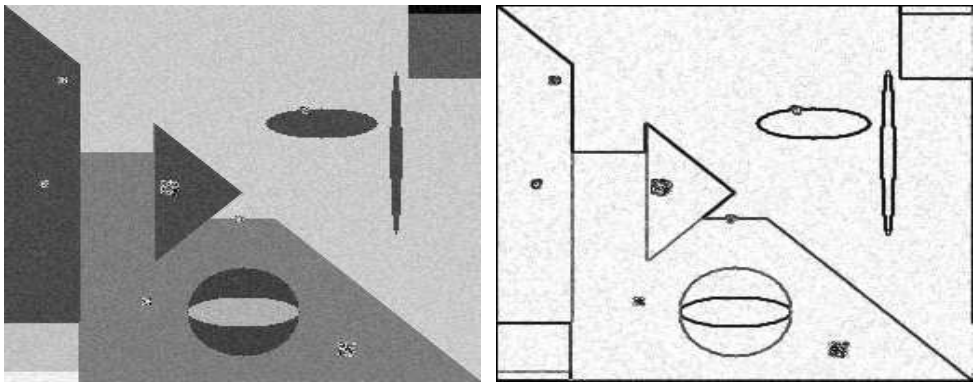
(a)

(b)

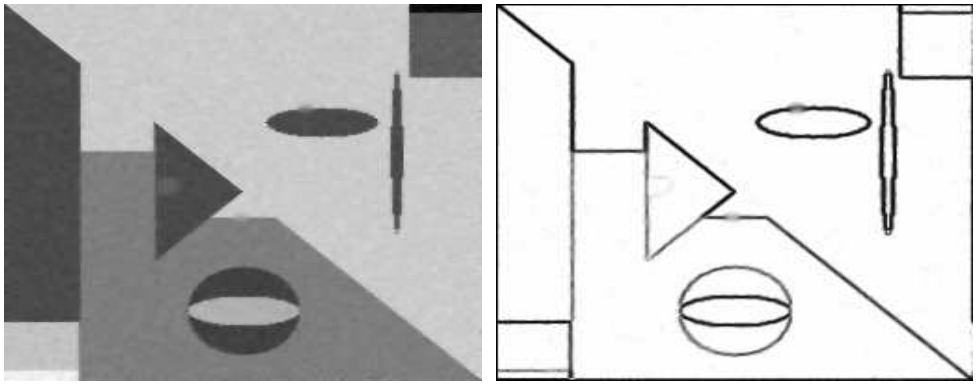
(c)

Left to right: image $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$
(a) Image with missing regions;
(b) Conjugate Gradient: 5000 iterations;
(c) True Image

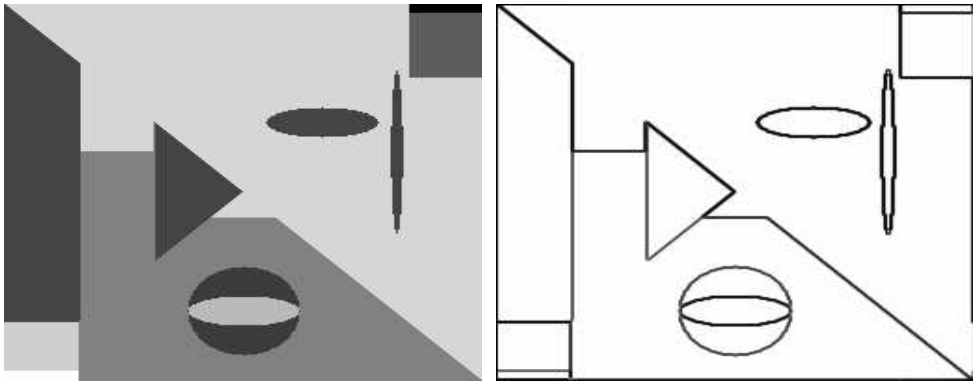
Figure 5.4: Inpainting and denoising on a piecewise constant image



(a)



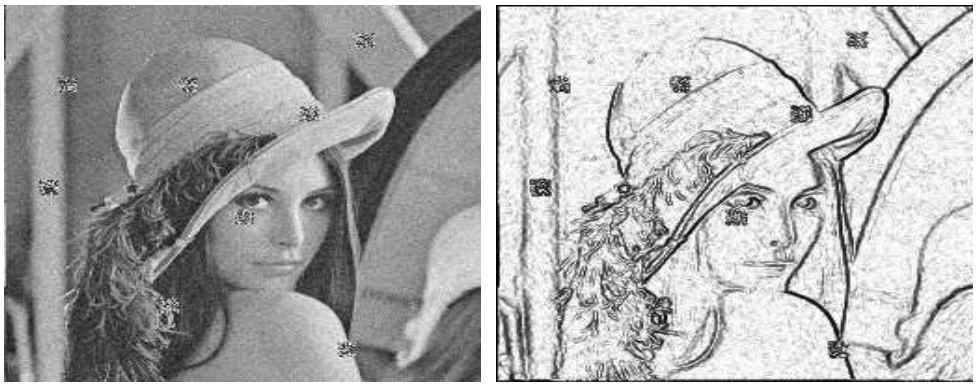
(b)



(c)

Left to right: image $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$
(a) Noisy image with missing regions;
(b) Conjugate Gradient: 11000 iterations;
(c) True Image

Figure 5.5: Inpainting and denoising on Lenna



(a)

(b)

(c)

Left to right: image, $\mu(x_1, x_2)$; edge map, $\frac{1}{1+k|\nabla G_\sigma * \mu|^2}$
(a) Noisy image with missing regions;
(b) Conjugate Gradient: 5000 iterations;
(c) True Image

Chapter 6

Conclusion

We presented a new implementation using the conjugate gradient algorithm for nonstandard diffusion for image restoration. This allowed us to denoise piecewise constant and piecewise smooth images faster than the current time-marching finite difference implementation. We included experimental results of images to support this claim. The nonstandard diffusion model was also modified to incorporate inpainting. This modification successfully inpainted regions on a piecewise constant image such that the regions appeared natural after restoration. In the future, numerical schemes developed here can be expanded to incorporate deblurring as well as texture extraction.

Bibliography

[1] P. Blomgren. Total variation methods for restoration of vector valued images. *Ph.D. Thesis*, pages 384–387, 1998.

[2] A. Chambolle and P.-L. Lions. Image recovery via total variation minimization and related problems. *Numer. Math.*, 76(2):167–188, 1997.

[3] T. Chan and S. J. Nontexture inpainting by curvature driven diffusions (cdd). *J. Visual Comm. Image Rep.*, 12(4):436–449, 2001.

[4] T. F. Chan, S. H. Kang, and J. Shen. Euler's elastica and curvature-based inpainting. *SIAM J. Appl. Math.*, 63(2):564–592 (electronic), 2002.

[5] E. K. P. Chong and S. H. Żak. *An Introduction to Optimization*. Discrete Mathematics and Optimization. John Wiley and Sons, Inc., New York City, NY, 2001.

[6] E. M. Johansson, F. U. Dowla, and D. M. Goodman. Backpropagation learning for multi-layer-feed-forward neural networks using the conjugate gradient method. *Int. J. Neural Systems*, 2(4):291–301, 1992.

[7] S. Levine, Y. Chen, and J. Stanich. Nonstandard diffusion in image restoration. *submitted to Computer Vision and Pattern Recognition*, 2003.

[8] M. J. D. Powell. Convergence properties of algorithms for nonlinear optimization. *SIAM Rev.*, 28(4):487–500, 1986.

[9] S. Rudin, L. Osher and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.