University of California

Los Angeles

Fast Numerical Algorithms for Total Variation Based Image Restoration

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Mathematics

by

Mingqiang Zhu

2008

The dissertation of Mingqiang Zhu is approved.

Luminita Vese

Lieven Vandenberghe

Achi Brandt

Tony F. Chan, Committee Chair

University of California, Los Angeles

2008

To my wife ...

who is the sunshine in my life

To my parents ...

who always have their faith in me

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my advisor, Professor Tony Chan, for his wise guidance and generous support throughout my graduate studies at UCLA. His enthusiasm and insight provided invaluable help in reaching closure on this thesis topic. I feel paticulary grateful about his effort to meet with me regularly to discuss about my research project despite of his busy schedule in NSF.

I would like to express my gratitude to my other committee members, Professor Achi Brandt, Professor Luminita Vese and Professor Lieven Vandenberghe for their great help, encouragement and valuable discussions on my studies and research.

I am thankful to my coauthor, Professor Stephen Wright for his stimulating discussions. The oppotunity of collabrating with him leads to an important part of my dissertation.

I would like to thank Professor Andrea Bertozz and Professor Stanley Osher for their help in my studies and research. I also appreciate the helpful discussions and suggestions I received from postdoctoral researchers and graduate students in our reserach group and the math department, which includes Jerome Darbon, Xavier Bresson, Sheshadri Thiruvenkadam, Ernie Esser, Ethan Brown and many others.

I am grateful to my friends in IPAM 1129: Connie, Ronald, Igor, Bin, Jian, Yu, and Yunho. I thank them for their cares and all the helpful and insightful discussions during lunch and tea time. That would aslo lead my thanks to IPAM and all the staff working there for providing such a wonderful working environment to all the participants.

I am also thankful to all of the people who work in the mathematics depart-

ment here at UCLA for all of their help, and I would really like to extend my appreciation to Maggie Albert, Robin Krestalude and Babette Dalton.

Vita

1980	Born, Huzhou, Zhejiang, China.
2001	B.S. in Automotive Engineering Tsinghua University, China.
2004	M.A. in Mathematics Arizona State University, Tempe, AZ
2005	M.A. in Mathematics University of California, Los Angeles
2004-2008	Teaching Assistant, Research Assistant Department of Mathematics University of California, Los Angeles

Publications

Abstract of the Dissertation

Fast Numerical Algorithms for Total Variation Based Image Restoration

by

Mingqiang Zhu

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2008

Professor Tony F. Chan, Chair

Image restoration models based on total variation (TV) have been popular and successful since their introduction by Rudin, Osher, and Fatemi (ROF) in 1992. The nonsmooth TV seminorm allows them to preserve sharp discontinuities (edges) in an image while removing noise and other unwanted fine scale detail. On the other hand, the the TV term, which is the L^1 norm of the gradient vector, poses computational challenge in solving those models efficiently. Furthermore, the global coupling of the gradient operator makes the problem extra harder than other L^1 minimization problems where the variables under the L^1 norm are separable. In this paper we propose several new algorithms to tackle these difficulties from different perspectives. Numerical experiments show that they are competitive with the existing popular methods and some of them are significantly faster despite of their simplicity. The first algorithm we introduce is a *primal-dual hybrid gradient descent* method that alternates between the primal and the dual updates. It utilizes the information from both the primal and dual variables and therefore is able to converges faster than the pure primal or pure dual method in the same category. We then proposed gradient projection (GP) methods to solve

the dual problem of the ROF model based on the special structure of the dual constraints. We also test variants of GP algorithms with different step selection strategies, including techniques based on the Barzilai-Borwein method. In this same line, a block co-ordinate descent method is proposed for solving the dual ROF problem. The subproblem at each single block can be solved exactly using different techniques. We also propose a basic multilevel optimization framework for the dual formulation, aiming to speedup our solution process for large scale problems. Finally, we study the connections between some existing methods and give an improvement to CGM method based on the primal-dual interior-point algorithms.

CHAPTER 1

Introduction

1.1 Total Variation Based Image Restoration Models

Image restoration is one of the fundamental problem in digital image processing. Variational models have been extremely successful in a wide variety of image restoration problems and remain one of the most active areas of research in mathematical image processing and computer vision. The most fundamental image restoration problem is perhaps denoising. It forms a significant preliminary step in many machine vision tasks such as object detection and recognition. Total variation (TV)-based image restoration models were first introduced by Rudin, Osher, and Fatemi (ROF) in their pioneering work [60]. It was designed with the explicit goal of preserving sharp discontinuities (edges) in an image while removing noise and other unwanted fine-scale detail. ROF formulated the following minimization problem:

$$\min_{u \in \mathrm{BV}(\Omega)} \quad \int_\Omega |\nabla u| \qquad \text{s.t.} \quad \|u - f\|_2^2 \leq |\Omega|\sigma^2. \tag{1.1}$$

Where $\int_\Omega |\nabla u|$ is the *total variation* (TV) of u and will be denoted as TV$[u]$. Other notations in (1.1) are explained as follows:

- Ω is the image domain, which will be taken to be a bounded domain in $\mathbb{R}^n$ with Lipschitz boundary. Usually Ω is simply a rectangle in $\mathbb{R}^2$, modeling the computer screen.

- $|\Omega|$ is the area of the domain Ω.
- The function $f : \Omega \to \mathbb{R}$ represents the given observed image. $f \in L^2(\Omega)$.
- σ is an estimate of the standard deviation of the noise in the image f.
- The notation $|\cdot|$ represents the Euclidean (ℓ_2) norm on $\mathbb{R}^2$, i.e.

$$|\nabla u| = \big|(u_x;\, u_y)\big| = \sqrt{u_x^2 + u_y^2}$$

- The notation $\|\cdot\|$ represents the usual L^2 norm on $L^2(\Omega)$.
- $\mathrm{BV}(\Omega)$ denotes the collection of all functions (images) in $L^1(\Omega)$ with finite total variations (bounded variations).

The *total variation* of u has a more formalized definition in mathematics:

$$\mathrm{TV}[u] \equiv \max\Big\{ \int_\Omega -u\nabla \cdot w \;\Big|\; w = (w_1, w_2) \in C_c^1(\Omega, \mathbb{R}^2),\ |w| \le 1 \Big\}. \tag{1.2}$$

This definition of TV is more general since it does not require u to be (almost everywhere) differentiable. In fact, we only require u to be in the BV space $\mathrm{BV}(\Omega)$, whose definition uses the above definition of TV:

$$\mathrm{BV}(\Omega) \equiv \Big\{ u \;\Big|\; u \in L^1(\Omega),\ \mathrm{TV}[u] < \infty \Big\}. \tag{1.3}$$

It can be shown that $W^{1,1}(\Omega) \subseteq \mathrm{BV}(\Omega) \subseteq L^1(\Omega)$ and the definition of TV in (1.2) is equivalent to the objective function of (1.1) when $u \in W^{1,1}$. More theoretical properties of *total variation* and BV space can be seen in references [5],[28], [42] and [52]. We also point out that, once we try to solve the problem in the discrete case, all the above technical restriction on u becomes irrelevant. It is clear that any discrete image u in $\mathbb{R}^2$ has finite (discrete) total variations.

The results about existence and uniqueness for the solutions of problem (1.1) can be seen in [17]. The same paper also shows that the constrained ROF model (1.1) is equivalent to the following unconstrained model:

$$\min_{u\in BV} \ \mathrm{TV}[u] + \frac{\lambda}{2}\|u-f\|_2^2, \tag{1.4}$$

for some suitable Lagrange multiplier λ.

In fact, both ROF and their subsequent researchers mostly focus on solving the unconstrained model (1.4) rather than the original constrained model (1.1) since unconstrained optimization is generally comparatively easier to solve.

The original ROF model was introduced to solve image denosing problems. But the methodology can be naturally extended to restore blurred and noisy image by including the known blurring kernal:

$$\min_{u\in BV} \ \mathrm{TV}[u] + \frac{\lambda}{2}\|Ku-f\|_2^2. \tag{1.5}$$

Here K is a given linear blurring operator and every other term is defined the same as in (1.4). In this model, f is formulated as the sum of a Gaussian noise v and a blurry image $K\bar{u}$ resulting from the linear blurring operator K acting on the clean image $\bar{u}$, i.e., $f = K\bar{u} + v$.

Among all linear blurring operators, many are shift-invariant and can be expressed in the form of convolution:

$$(Ku)(x) = (h * u)(x) = \int_\Omega h(x-y)\,u(y)\,dy, \tag{1.6}$$

where h is the given *point spread function* (PSF) associated with K.

Over the years, the ROF model has been extended to many other image processing tasks, including inpaintings, blind-deconvolutions, cartoon-texture decompositions and vector valued images. (See [24] and the reference therein for

recent developments in TV based image processing.) It has also been modified in a variety of ways to improve its performance by using iterative refinement with Bregman distance [61] and the more general inverse scale methods [12] . In this paper, we shall focus on solving the original ROF restoration problem, but we point out that some of our ideas can be naturally extended to other relevant models.

1.2 Duality

The discussion so far, based on the minimization models (1.1) and (1.4), can be viewed as the *primal* approach to solving the TV denoising problem. In this section, we shall discus about the *dual* formulations of the ROF models (1.4) and (1.1) and some related analysis. While the dual formulation is not as well developed as the primal one, it does have some inherent advantages and has been receiving increasing interest recently (see e.g., [13, 25, 14]). It offers an alternative formulation which can lead to effective computational algorithms.

First, we shall adopt the definition (1.2) for the TV semi-norm, which we rewrite here

$$\begin{aligned} \mathrm{TV}[u] &\equiv \max_{w \in W} \int_\Omega -u \nabla \cdot w, \\ \text{where} \quad W &\equiv \left\{ w \;\middle|\; w = (w_1, w_2) \in C_c^1(\Omega, \mathbb{R}^2),\ |w| = \sqrt{w_1^2 + w_2^2} \le 1 \right\} \end{aligned} \tag{1.7}$$

With this definition of TV, the ROF model (1.4) becomes

$$\min_u \max_{w \in C_c^1(\Omega),\, |w| \le 1} \int_\Omega -u \nabla \cdot w + \frac{\lambda}{2} \|u - f\|_2^2, \tag{1.8}$$

where u and w are the primal and dual variables, respectively. The min-max theorem (see e.g., Proposition 2.4 in [40])allows us to interchange the min and

max, to obtain

$$\max_{w \in C_c^1(\Omega), |w| \leq 1} \min_u \int_\Omega -u \nabla \cdot w + \frac{\lambda}{2} \|u - f\|_2^2. \tag{1.9}$$

The inner minimization problem can be solved exactly as follows:

$$u = f + \frac{1}{\lambda} \nabla \cdot w \tag{1.10}$$

leading to the following dual formulation:

$$\max_{w \in C_c^1(\Omega), |w| \leq 1} D(w) := \frac{\lambda}{2} \left[\|f\|_2^2 - \left\| \frac{1}{\lambda} \nabla \cdot w + f \right\|_2^2 \right], \tag{1.11}$$

or, equivalently,

$$\min_{w \in C_c^1(\Omega), |w| \leq 1} \frac{1}{2} \|\nabla \cdot w + \lambda f\|_2^2. \tag{1.12}$$

For a primal-dual feasible pair (u, w), the duality gap $G(u, w)$ is defined to be the difference between the primal and the dual objectives:

$$\begin{aligned} G(u, w) &= P(u) - D(w) \\ &= \int_\Omega |\nabla u| + \frac{\lambda}{2} ||u - f||_2^2 - \frac{\lambda}{2} \left[\|f\|_2^2 - \left\| \frac{1}{\lambda} \nabla \cdot w + f \right\|_2^2 \right] \\ &= \int_\Omega \Big(|\nabla u| - \nabla u \cdot w \Big) + \frac{\lambda}{2} \left\| \frac{1}{\lambda} \nabla \cdot w + f - u \right\|_2^2. \end{aligned} \tag{1.13}$$

The duality gap bounds the distance to optimality of the primal and dual objectives. Specifically, if u and w are feasible for the primal (1.4) and dual (1.11) problems, respectively, we have

$$0 \leq P(u) - O^* \leq G(u, w), \tag{1.14a}$$

$$0 \leq O^* - D(w) \leq G(u, w), \tag{1.14b}$$

where O^* is the (common) primal-dual optimal objective value. We make use of the duality gap in the termination criterion in our numerical experiments.

Optimization theory tells us $G(u, w) \geq 0$, and moreover, (u, w) are optimal if and only if $G(u, w) = 0$.

Notice here $|w| \leq 1$ and it follows $|\nabla u| - \nabla u \cdot w \geq 0 \ \forall x \in \Omega$. Therefore $G(u, w) = 0$ if and only if

$$|\nabla u| + \nabla u \cdot w = 0 \qquad \text{and} \qquad \frac{1}{\lambda}\nabla \cdot w + f - u = 0.$$

It is also clear that $|\nabla u| - \nabla u \cdot w = 0$ is equivalent to $|\nabla u| w - \nabla u = 0$.

Hence, (u, w) is optimal if and only if the following system holds

$$\begin{cases} |\nabla u| w - \nabla u = 0 \\ \nabla \cdot w - \lambda(u - f) = 0 \end{cases} \tag{1.15}$$

The above system (1.15) is referred as the *primal-dual system*.

We can derive the dual formulation of the constrained ROF model (1.1) in exactly the same way. First of all, the constrained ROF model (1.1) can be rewrite into the following min-max problem using the general definition of TV in (1.7):

$$\min_{u \in U} \max_{w \in W} \int_\Omega -u \nabla \cdot w, \tag{1.16}$$

where $U \equiv \{u \in \mathrm{BV}(\Omega) \ : \ \|u - f\|_2^2 \leq |\Omega| \sigma^2\}$ and W is defined as in (1.7).

Again, the min-max theorem allows us to interchange the min and max in the above problem and we obtain the equivalent

$$\max_{w \in W} \min_{u \in U} \int_\Omega -u \nabla \cdot w. \tag{1.17}$$

Now the inner minimization problem in (1.17) can be solved exactly as

$$u = f + \sqrt{|\Omega|}\, \sigma \frac{\nabla \cdot w}{\|\nabla \cdot w\|_2}. \tag{1.18}$$

Substituting the above result back to problem (1.17), we obtain the dual formulation

$$\max_{w \in W} \ D[w] \ \equiv \ -\sqrt{|\Omega|}\, \sigma \|\nabla \cdot w\|_2 - \int_\Omega f \, \nabla \cdot w \tag{1.19}$$

Similarly, the duality gap $G(u, w)$ for any feasible primal-dual pair (u, w) is defined as

$$\begin{aligned} & G(u,w) \equiv P[u] - D[w] \\ = & \int_\Omega \left(|\nabla u| + f \nabla \cdot w \right) + \sqrt{|\Omega|}\, \sigma \|\nabla \cdot w\|_2 \\ = & \int_\Omega \left(|\nabla u| - \nabla u \cdot w \right) + \left[\sqrt{|\Omega|}\, \sigma \|\nabla \cdot w\|_2 + \int_\Omega (f-u)\, \nabla \cdot w \right] \end{aligned} \tag{1.20}$$

Setting $G(u, w) = 0$ gives the following optimality condition

$$\begin{cases} |\nabla u| w - \nabla u = 0 \\ \nabla \cdot w - \frac{\|\nabla \cdot w\|_2}{\sigma\, |\Omega|^{\frac{1}{2}}} (u - f) = 0 \end{cases} \tag{1.21}$$

Comparing (1.18) with (1.10) or comparing (1.21) with (1.15) we find the suitable fidelity parameter λ that leads to equivalent models (1.4) and (1.1) are given as

$$\lambda = \frac{\|\nabla \cdot w^*\|_2}{\sigma\, |\Omega|^{\frac{1}{2}}}, \tag{1.22}$$

where w^* is the dual optimizer of problem (1.19).

1.3 Challenges in Computing TV models

From the computational point of view, the primal and dual formulations pose different challenges for computing their optimality solutions (see Table 1.1). The total variation term in the primal formulation is non-smooth at where $|\nabla u| = 0$, which makes the derivative-based methods impossible without an artificial smoothing parameter. The dual formulation imposes constraints which usually require extra effort compared to unconstrained optimizations. Being quadratic, the dual energy is less nonlinear than the primal energy, but the rank-deficient operator $\nabla\cdot$ makes the dual minimizers possibly non-unique. Finally, they share

the same problem of spatial stiffness due to the global couplings in their energy functions, which presents a challenge to any algorithm in order to control the computational complexity that scales reasonably bounded with the number of pixels.

Primal Problem (1.4)	Dual Problem (1.11)
· Nondifferentiable at $\|\nabla u\| = 0$	· Non-uniqueness due to rank-deficient $\nabla\cdot$
· Highly nonlinear	· Extra constraints
· Spatial stiffness	· Spatial stiffness

Table 1.1: Computational Challenges for Primal and Dual ROF

1.4 Discretization and Notations

Before describing the numerical algorithms, let us choose an appropriate way to discretize the continuous ROF model and fix the main relevant notational conventions.

Often in this paper we need to concatenate vectors and matrices, in both column-wise or row-wise fashion. We follow the MATLAB convention of using "," for adjoining vectors and matrices in a row, and ";" for adjoining them in a column. Thus, for any vectors x, y and z, the following are synonymous:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = (x^T,\, y^T,\, z^T)^T = (x; y; z)$$

For the sake of simplicity, we assume that the domain Ω is squares, and define a regular $n \times n$ grid of pixels, indexed as (i, j), for $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$. We represent images images as two-dimensional matrices of dimension $n \times n$,

where $u_{i,j}$ represents the value of the function u at pixel (i,j). (Adaptation to less regular domains is not difficult in principle.) To define the discrete total variation, we introduce a discrete gradient operator, whose two components at each pixel (i,j) are defined as follows:

$$(\nabla u)^1_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } i < n \\ 0 & \text{if } i = n \end{cases} \tag{1.23a}$$

$$(\nabla u)^2_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } j < n \\ 0 & \text{if } j = n. \end{cases} \tag{1.23b}$$

(Thus $\nabla u \in \mathbb{R}^{n\times n\times 2}$.) The discrete TV of u is then defined by

$$\mathrm{TV}(u) = \sum_{1\le i,j,\le n} |(\nabla u)_{i,j}|,$$

where $|\cdot|$ is the Euclidean (ℓ_2) norm in $\mathbb{R}^2$. Note that this norm is not a smooth function of its argument. It has the classic "ice-cream cone" shape, nondifferentiable when its argument vector is zero.

The discrete divergence operator is defined, by analogy with the continuous setting, as the negative adjoint of the gradient operator, that is, $\nabla\cdot = -\nabla^*$. Defining the inner product of two objects in $\mathbb{R}^{n\times n}$ as follows:

$$\langle u, v\rangle = \sum_{1\le i,j\le n} u_{i,j} v_{i,j},$$

(and similarly for objects in $\mathbb{R}^{n\times n\times 2}$), we have from definition of the discrete divergence operator that for any $u \in \mathbb{R}^{n\times n}$ and $w \in \mathbb{R}^{n\times n\times 2}$, that $\langle \nabla u, w\rangle = \langle u, -\nabla\cdot w\rangle$. It is easy to check that the divergence operator can be defined explicitly as follows:

$$(\nabla\cdot w)_{i,j} = \begin{cases} w^1_{i,j} - w^1_{i-1,j} & \text{if } 1 < i < n \\ w^1_{i,j} & \text{if } i = 1 \\ -w^1_{i-1,j} & \text{if } i = n \end{cases} + \begin{cases} w^2_{i,j} - w^2_{i,j-1} & \text{if } 1 < j < n \\ w^2_{i,j} & \text{if } j = 1 \\ -w^2_{i,j-1} & \text{if } j = n. \end{cases} \tag{1.24}$$

It is worth noticing that the values $\{w^1_{n,j} : j = 1 \cdots, n\}$ and $\{w^2_{i,n} : i = 1, \cdots, n\}$ are not relevant in calculating the divergence $\nabla \cdot w$ in (1.24). Therefore, the discrete dual problem only have essentially $2n(n-1)$ instead of $2n^2$ unknowns. Non-the-less, the boundary points are usually included in w (with values being 0) to make the discrete dual problem consistent in form with the primal problem.

To describe the problem in matrix algebra language, we reorder the image matrix u (resp. f) in row-wise fashion into a vector y (resp. z), associating the (i, j) element of the two-dimensional structure with the element $(j-1)n + i$ of the vector structure, as follows:

$$y_{(j-1)n+i} = u_{i,j}, \quad 1 \le i, j \le n.$$

We have $y \in \mathbb{R}^N$, where $N = n^2$. The (i, j) component of the gradient (1.23) can thus be represented as a multiplication of the vector $y \in \mathbb{R}^N$ by a matrix $A_l \in \mathbb{R}^{2\times N}$, for $l = 1, 2, \ldots, N$:

$$A_l^T y = \begin{cases} (y_{l+1} - y_l;\ y_{l+n} - y_l) & \text{if } l \bmod n \neq 0 \text{ and } l + n \le N \\ (0;\ y_{l+n} - y_l) & \text{if } l \bmod n = 0 \text{ and } l + n \le N \\ (y_{l+1} - y_l;\ 0) & \text{if } l \bmod n \neq 0 \text{ and } l + n > N \\ (0;\ 0) & \text{if } l \bmod n = 0 \text{ and } l + n > N. \end{cases} \tag{1.25}$$

Using this notation, the discrete version of the unconstrained primal ROF model (1.4) can be written as follows:

$$\min_{y\in\mathbb{R}^N} \sum_{l=1}^{N} \|A_l^T y\| + \frac{\lambda}{2}\|y - z\|^2, \tag{1.26}$$

where (and from now on) we use $\|\cdot\|$ do denote the Euclidean norm (ℓ_2 norm) in Euclidean space $\mathbb{R}^m$ with any finite dimension.

Similarly, we restructure the dual variable w row-wisely into a collection of

vectors $x_l \in \mathbb{R}^2$, $l = 1, 2, \ldots, N$, as follows:

$$x_{(j-1)n+i} = \begin{bmatrix} w^1_{i,j} \\ w^2_{i,j} \end{bmatrix}, \quad 1 \le i, j \le n.$$

The complete vector $x \in \mathbb{R}^{2N}$ of unknowns for the discretized dual problem is then obtained by concatenating these subvectors:

$$x = (x_1; x_2; \ldots; x_N).$$

We also form the matrix A by concatenating the matrices A_l, $l = 1, 2, \ldots, N$ defined in (1.25), that is,

$$A = (A_1, \ldots, A_N) \in \mathbb{R}^{N \times 2N}. \tag{1.27}$$

In this notation, the divergence $\nabla \cdot w$ is simply $-Ax$, so the discretization of the dual ROF model (1.12) is

$$\begin{aligned} &\min_{x \in X} \|Ax - \lambda z\|^2 \\ \text{where} \quad &X = \{x : \ x \in \mathbb{R}^{2N}, \ \|x_l\| \le 1 \text{ for } l = 1, 2, \cdots, N\} \end{aligned} \tag{1.28}$$

Similarly, the duality gap in (1.13) has the following discrete form

$$G(y, x) = \sum_{l=1}^{N} \left(\|A_l^T y\| - x_l^T A_l^T y \right) + \frac{\lambda}{2} \left\| y - z + \frac{1}{\lambda} Ax \right\|^2, \tag{1.29}$$

and primal-dual optimality condition (1.15) has the discrete form

$$\begin{cases} \|A_l^T y\| - x_l^T A_l^T y &= 0 \quad \text{for } l = 1, \cdots, N \\ y - z + \frac{1}{\lambda} Ax &= 0 \end{cases} \tag{1.30}$$

By the same token, the constrained ROF model (1.1) has the following discretization for its primal formulation, dual formulation, primal-dual gap and primal-dual optimality conditions:

Primal problem:

$$\begin{aligned} &\min_y \sum_{l=1}^{N} \|A_l^T y\| \\ \text{subject to} \quad &y \in Y \equiv \{y \in \mathbb{R}^N \,:\, \|y - z\| \le n\sigma\} \end{aligned} \tag{1.31}$$

Dual problem:

$$\begin{aligned} &\max_x \; -n\sigma\|Ax\| + z^T Ax \\ \text{subject to} \quad &x \in X \equiv \{x \in \mathbb{R}^{2N} \,:\, \|x_l\|^2 \le 1, \;\; l = 1, 2, \ldots, N\}. \end{aligned} \tag{1.32}$$

Duality Gap:

$$G(y, x) = \sum_{l=1}^{N} \Big(\|A_l^T y\| - x_l^T A_l^T y \Big) + \Big(n\sigma\|Ax\| + (y - z)^T Ax \Big), \tag{1.33}$$

Primal-dual optimality conditions:

$$\begin{cases} \|A_l^T y\|\, x_l - A_l^T y &= 0 \quad \text{for } l = 1, \cdots, N \\ y - z + \frac{\|Ax\|}{n\sigma} Ax &= 0 \end{cases} \tag{1.34}$$

1.5 Organizations of This Paper

In this paper, we try to develop some efficient algorithms to solve the total variation based image restoration models. The paper is organized as follows:

Chapter 1 gives a short introduction to the total variation based image restoration models and BV space. It covers the application of duality theory to the ROF model and derives the dual problem. We then analyze the different computational challenges facing the primal and dual formulation of ROF model and end the chapter by introduction necessary notations and discretization for the study numerical algorithms.

Chapter 2 offers a brief survey of some existing popular methods. At the end of chapter 2, we also give remarks on exiting methods and motivations on developing new algorithms.

Chapter 3 proposes a simple yet efficient *primal-dual hybrid gradient descent* method for solving the ROF model. Results of numerical experiments to demonstrate its fast convergence. Connections to projection type methods in solving variational inequalities and extensions to other models are studied.

Chapter 4 introduces gradient projection methods to solve the dual formulation of the ROF model. We test variants of GP with different step selection and line search strategies, including techniques based on the Barzilai-Borwein method. Global convergence can in most cases be proved by appealing to existing theory. Numerical results are shown in demonstrating performance.

Chapter 5 proposes a block coordinate descent method to solve the dual formulation of the ROF model. Global convergence of the algorithm are proved and results of numerical experiments are shown. Dual formulation of anisotropic ROF model are also studied for its simpler constraints structure.

Chapter 6 tries to speedup the algorithm in solving the dual problem of the ROF model by applying some basic multilevel optimization techniques.

Chapter 7 studies the connections between the CGM method and SOCP method for solving the ROF model. An improvement of CGM algorithm based on *primal-dual interior-point method* is aslo proposed.

Chapter 8 gives an comprehensive comparison of the newly developed methods

in this paper and some existing popular algorithms. It ends with some final conclusions and remarks.

CHAPTER 2

A Brief Survey of Existing Algorithms

2.1 Numerical Algorithms

2.1.1 Time Marching Schemes

If we replace $|\nabla u|$ in (1.4) by $|\nabla u|_\beta = \sqrt{|\nabla u|^2 + \beta}$, we obtain the following smoothed (β-regularized) ROF model

$$\min_u \int_\Omega |\nabla u|_\beta + \frac{\lambda}{2}\|u - f\|_2^2. \tag{2.1}$$

The above problem has a convex differentiable objective function and hence its optimality solution is given by the following Euler-Lagrange equation (first-order condition)

$$\nabla \cdot \Big(\frac{\nabla u}{|\nabla u|_\beta}\Big) - \lambda(u - f) = 0. \tag{2.2}$$

In their original paper [60], Rudin et al. proposed the use of artificial time marching to solve the Euler-Lagrange equation (2.2) which is equivalent to the steepest descent of the energy function. More precisely, consider the image as a function of space and time and seek the steady state of the equation

$$\frac{\partial u}{\partial t} = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|_\beta}\right) - \lambda(u - f) \tag{2.3}$$

In numerical implementation, an explicit time marching scheme with time step Δt and space step Δx(usually is 1) is used. Under this method, the objective value of

the ROF model is guaranteed to be decreasing provided that the time step is small enough and the solution will tend to the unique minimizer as time increases. The method is (asymptotically) slow due to the CFL stability constraints Courant-Friedrichs-Lewy (CFL) condition $\Delta t \leq c|\nabla u|_\beta(\Delta x)^2$ for some constant $c > 0$ (see [59]), which puts a very tight bound on the time step when the solution develop flat regions (where $|\nabla u| \approx 0$). Hence, this scheme is useful in practice only when low-accuracy solutions suffice. Sometimes even the cost of computing a visually satisfactory image is too great.

To relax the CFL condition, Marquina and Osher use, in [59], a "preconditioning" technique to cancel singularities due to the degenerate diffusion coefficient $\frac{1}{|\nabla u|}$:

$$\frac{\partial u}{\partial t} = |\nabla u|\left[\nabla \cdot \left(\frac{\nabla u}{|\nabla u|_\beta}\right) - \lambda(u - f)\right] \tag{2.4}$$

which can also be viewed as mean curvature motion with a forcing term $-\lambda(u-f)$.

2.1.2 Fixed Point Iteration

To bypass the stability constraint in time marching schemes, Vogel and Oman proposed in [66] a "lagged diffusivity" fixed point iteration scheme which solves the stationary Euler-Lagrange equation directly. The main idea is to linearize the Euler-Lagrange equation by lagging the diffusion coefficients $\frac{1}{|\nabla u|_\beta}$ at a previous iteration. The k-th iterate is obtained by solving the sparse linear system (see [66], [67])

$$\nabla \cdot \left(\frac{\nabla u^k}{|\nabla u^{k-1}|_\beta}\right) - \lambda(u^k - f) = 0 \tag{2.5}$$

The above linear system is block tri-diagonal and can be solved by various fast direct/iterative solvers. While global convergence has been proven by different authors(see e.g. [33], [17], [27], [39]), the outer iteration is only linearly convergent and the convergence slows down as β decreases. Moreover, the inner linear

system is difficult to solve efficiently because of the highly varying and possibly degenerate coefficient $\frac{1}{|\nabla u|_\beta}$, as well as spatial stiffness of the ellipticity of the differential operator.

2.1.3 Primal-Dual Newton Method (CGM)

In order to obtain superlinear convergence, a Newton-type algorithm is needed. However, naive exact Newton's method is known to have a very small domain of convergence for equation (2.2) and fails for modest β (see [32], [11]). In [32], Chan et al tried to combine Newton's method with a continuation method in the parameter β, which produced more robustness but still was not overall satisfactory. Ng et al proposed in [56] an nonsmooth Newton's method to solve the original ROF model which does not require the smoothing parameter β. However, the number of Newton iterations it required to achieve convergence is still considered too large.

A much better alternative introduce by Chan, Golub and Mullet(CGM) in [25] is to apply Newton's method to the primal-dual system. By introducing a new dual variable $w := \frac{\nabla u}{|\nabla u|_\beta}$ to equation (2.2), they obtained the following equivalent system to (2.2)

$$\begin{cases} w|\nabla u|_\beta - \nabla u = 0 \\ \nabla \cdot w - \lambda(u - f) = 0 \end{cases} \tag{2.6}$$

It is now clear to us that the above system is a β-regularized version of the true primal-dual system (1.15), which we rewrite here:

$$\begin{cases} |\nabla u| w - \nabla u = 0 \\ \nabla \cdot w - \lambda(u - f) = 0 \end{cases} \tag{2.7}$$

System 2.1.3 and (1.15) are the same except the the norm $|\nabla u|$ in is smoothed

with β. When $\beta \downarrow 0$, the solution of 2.1.3 will converge to the solution of (1.15), i.e., the true optimality solution of the ROF model.

As discussed in [25] and [11], this primal-dual system is better suited for Newton's method than the primal Euler-Lagrange equation (2.2) since the (u, w) system is intuitively more "linear" than the primal system in u only. Furthermore, compared to fixed-point methods and time-marching methods that are at best linearly convergent, the CGM method was shown in [25] to be empirically globally convergent with a quadratic rate. Moreover, the convergent rate deteriorates only moderately with decreasing β and in practice small values of β can be used without too much loss in efficiency.

There is still a need to solve an elliptic linear system at every Newton iteration, but through an elimination of the update on w, the Newton iteration can be implemented by solving only one linear system involving the update for u

$$J(u^k, w^k) \cdot \delta u^k = -r(u^k),$$

where J is positive definite and r is the steepest ascent direction of the primal objective function(i.e. r is the LHS of equation (2.2). This ensures that the Newton update is a decent direction and makes the cost per iteration comparable to that for the fixed point method. For applications where a high accuracy solution is needed, eg. in an automated image processing setting without human intervention, fast convergent methods such as CGM will have a significant advantage over slower linearly convergent methods.

2.1.4 Duality-Based Gradient Descent Method

The main advantage of the dual formulation (1.11) or (1.12) is that the objective function is nicely quadratic and hence there is no issue with non-differentiability,

or the need for numerical regularization, as in the primal formulation. However, this comes with the price of dealing with the constraints on the dual w. In fact, there are as many constraints as there are number of pixels and one does not know in advance where the constraints are active. There are several "standard" numerical approaches to solving nonlinear constrained optimization (1.11), e.g. penalty, barrier, augmented Lagrangian methods and interior-point methods(see e.g. [53]). Carter did some early work in this direction in her thesis [13].

One advance in this direction was made by Chambolle in [14], in which he obtained an explicit analytic formula for the Lagrange multipliers for the constraints. This idea is also extended to high order dual methods by Chan et. al. in [23].

The optimality conditions(Karush-Kuhn-Tucker) for the discretized version of (1.12) is (see [14])

$$-(\nabla(\nabla \cdot w - \lambda f))_{i,j} + \alpha_{i,j} w_{i,j} = 0 \tag{2.8a}$$

$$\alpha_{i,j}(|w_{i,j}|^2 - 1) = 0 \tag{2.8b}$$

$$\alpha_{i,j} \geq 0 \tag{2.8c}$$

where $\alpha_{i,j}$ are the Lagrange multipliers.

Now Chambolle made the following original observation about Lagrange multipliers α. The complementarity conditions (2.8b) are satisfied as either of the following two cases:

$$\alpha_{i,j} = 0 \quad \text{or} \quad |w_{i,j}| = 1 \tag{2.9}$$

Applying (5.15) back to (2.8a), we have

$$\alpha_{i,j} = 0 \;\Rightarrow\; |(\nabla(\nabla \cdot w - \lambda f))_{i,j}| \;=\; \alpha_{i,j}|w_{i,j}| \;=\; 0 \;=\; \alpha_{i,j} \tag{2.10}$$

$$|w_{i,j}| = 1 \;\Rightarrow\; |(\nabla(\nabla \cdot w - \lambda f))_{i,j}| \;=\; \alpha_{i,j}.|w_{i,j}| = \alpha_{i,j} \tag{2.11}$$

In either case, we have

$$\alpha = |\nabla(\nabla \cdot w - \lambda f)| \ , \tag{2.12}$$

which yields the following nonlinear algebraic equation in w only:

$$-\nabla(\nabla \cdot w - \lambda f) + |\nabla(\nabla \cdot w - \lambda f)| w = 0 \tag{2.13}$$

It is important to observe that while (2.13) still inherits the non-differentiability of the primal objective, it is nonetheless "integrated once", i.e. the nonsmoothness is in a set of algebraic equations rather than inside the minimization problem.

He then proposed a semi-implicit gradient descent algorithm (notice the LHS of (2.13) can be viewed as the gradient ascent direction):

$$w^{n+1} = w^n - \tau\Big(- \nabla(\nabla \cdot w^n - \lambda f) + \big|\nabla(\nabla \cdot w^n - \lambda f)\big| w^{n+1}\Big) \tag{2.14}$$

or,

$$w^{n+1} = \frac{w^n + \tau H(w^n)}{1 + \tau |H(w^n)|} \ , \tag{2.15}$$

where $H(w) = \nabla(\nabla \cdot w - \lambda f)$ and τ is a time step chosen suitably small for convergence.

It is clear that the constraints $|w| \leq 1$ are automatically handled in the above algorithm, provided the initial guess satisfies so. Global convergence for $\tau < \frac{1}{8}$ was also proved in [14]. However, since the linearization is through a fixed point method, the convergent rate is at best linear. Also, since nothing is applied to handle the spatial stiffness, the number of iterations will increase with the number of pixels. In practice, a relatively large number of iterations is still required. Of course, the main advantage is that sharp discontinuities can be recovered without numerical regularization and the method is globally convergent and easy to implement.

2.1.5 Second-Order Cone Programming

Recently, Goldfarb and Yin proposed a new approach in [43] for various total variation models based on second-order cone programming(SOCP). They model problem (1.1) as a second-order cone program, which is then solved by modern interior-point methods.

To reform the constrained ROF model

$$\begin{aligned} \min \quad & \sum_{1\le i,j\le n} \|(\nabla u)_{i,j}\| \\ \text{s.t.} \quad & \|u-f\|^2 \le \sigma^2 \end{aligned} \tag{2.16}$$

into a SOCP problem, the authors in [43] introduced new variables p, q, t, v as follows.

They let v to be the noise variable: $v = f - u$ and let $(p_{i,j}, q_{i,j})$ to be the discrete ∇u by forward differentiation:

$$p_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } i < n \\ 0 & \text{if } i = n \end{cases}$$

$$q_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } j < n \\ 0 & \text{if } j = n \end{cases}$$

They then introduce a new variable t as an upper bound of the gradient norm $|\nabla u|$:

$$|(\nabla u)_{i,j}| \;=\; \sqrt{p_{i,j}^2 + q_{i,j}^2} \;\le\; t_{i,j}$$

and this is where the "second-order cone" constraints come from.

As a consequence, problem (7.6) is transformed to

$$\begin{aligned}
\min \quad & \sum_{i,j=1}^{n} t_{i,j} \\
\text{s.t.} \quad & u_{i,j} + v_{i,j} = f_{i,j} && \text{for } i,j = 1,\cdots,n \\
& -p_{i,j} + u_{i+1,j} - u_{i,j} = 0 && \text{for } i = 1,\cdots,n-1,\ j = 1,\cdots,n \\
& -q_{i,j} + u_{i,j+1} - u_{i,j} = 0 && \text{for } i = 1,\cdots,n,\ j = 1,\cdots,n-1 \\
& p_{n,j} = 0 && \text{for } j = 1,\cdots,n, \\
& q_{i,n} = 0 && \text{for } i = 1,\cdots,n, \\
& v_0 = \sigma \\
& (t_{i,j};\ p_{i,j};\ q_{i,j}) \in \mathcal{K}^3 && \text{for } i,j = 1,\cdots,n, \\
& (v_0;\ v) \in \mathcal{K}^{n^2+1}
\end{aligned} \tag{2.17}$$

Eliminating u from formulation (7.7), we obtain the following standard form SOCP:

$$\begin{aligned}
\min \quad & \textstyle\sum_{1\le i,j\le n} t_{i,j} \\
\text{s.t.} \quad & p_{i,j} + v_{i+1,j} - v_{i,j} = f_{i+1,j} - f_{i,j} && \text{for } 1 \le i \le n-1,\ 1 \le j \le n \\
& q_{i,j} + v_{i,j+1} - v_{i,j} = f_{i,j+1} - f_{i,j} && \text{for } 1 \le i \le n,\ 1 \le j \le n-1 \\
& p_{n,j} = 0 && \text{for } 1 \le j \le n \\
& q_{i,n} = 0 && \text{for } 1 \le i \le n \\
& v_0 = \sigma \\
& (t_{i,j};\ p_{i,j};\ q_{i,j}) \in \mathcal{K}^3 && \text{for } 1 \le i,j \le n \\
& (v_0; v) \in \mathcal{K}^{n^2+1}
\end{aligned} \tag{2.18}$$

(7.8) is in standard form of SOCP and there are massive literature about how to solve this class of problems (see e.g. [3]). The dual formulation can also be transformed into SOCP (see [43]). SOCPs can be solved efficiently in practice and, in theory, in polynomial time by interior-point methods. For example, path-following interior-point methods generate a sequence of interior points that follow

the so-called central path toward the optimality. At each iteration, a set of primal-dual equations, which depend on current variable values and a duality gap parameter, are solved by one step of Newton's iteration to yield an improving search direction. Like CGM, SOCP need to solve a linear system at each iteration but can produce highly accurate solutions with low residuals or duality gaps.

In some sense, the SOCP framework bears many similarities with the earlier primal-dual Newton method discussed in section 2.1.3 First of all, they all apply Newton's method to compute the update search direction at each iteration. Secondly, they all use a dual variable and solve the primal-dual system. In the SOCP formulation, we do not need a regularization parameter β. However, when we apply interior-point method to solve SOCP, the primal-dual system that we solve does depend on an artificial parameter that measures the duality gap. This duality parameter is similar to β in CGM, and the only difference is that β is fixed in CGM while the duality parameter dynamically decreases during the solution process of SOCP. Another interesting point of view here is that SOCP minimize t, an upper bound of $|\nabla u|$ instead of minimizing $|\nabla u|$ itself, to get rid of the non-smoothness of the objective function, and this bears similarity to the early idea of minimizing $\sqrt{|\nabla u|^2 + \beta}$, which is also an upper bound of $|\nabla u|$.

2.1.6 Multigrid Methods

In order to obtain an efficient algorithm with scalable computational complexity, some sort of multilevel methods seems necessary. Interested readers can refer to the survey [19] for a more detailed account. One approach is to use multigrid to solve the various linear systems that arise in the methods mentioned above. One of the earliest attempts along this line can be seen in [2], [18], [68], [69] and [64], where a linear multigrid method is used as a fast solver for the inner linear

system (2.5) in the fixed point method. Here we rewrite down the system (2.5)

$$\nabla \cdot \Big(\frac{\nabla u^k}{|\nabla u^{k-1}|_\beta} \Big) - \lambda (u^k - f) = 0 \tag{2.19}$$

However, there are two main drawbacks of this approach. First, the linearized system (2.19) has very "nasty" coefficients (especially when β is small), making it difficult to derive efficient multigrid methods for its solution. Secondly, the outer fixed point iterations is still only linear convergent.

Recently [63] and [41] considered using the nonlinear multigrid method to solve the non-linear Euler-Lagrange equation (2.2). However, here one has to take a relatively large β to achieve convergence.

Another approach introduced in [20, 21] is to apply multilevel algorithms directly to the minimization problem (1.4), and it is generated in [34] to use piecewise linear coarser level corrections. This approach uses coordinate descent relaxation of the primal objective as a smoother. It either does not require the regularization parameter β (in 1D) or works with an extremely small β (in 2D). However, it is a well known fact that coordinate descent method is generally not guaranteed to converge for nonseparable non-smooth optimizations. As shown in [13] and [20, 21], primal relaxation usually gets stuck in a "local minimizer"(in the sense of the algorithm) due to the non-smoothness of the TV term and thus fails to converge to the true solution. Figure (2.1) illustrated how coordinate descent method might get stuck at a wrong solution. The key point here is when a flat region (non-smoothness) develops in an intermediate solution, changing the value of any single pixel might not be sufficient to decrease the objective function. However, the multilevel optimization method based on primal relaxation is somehow more robust because the coarse level correction can help the solution moves away from the "local minimizer". Nevertheless, the multilevel method with standard coarsening can still fail in some circumstances. [20, 21]

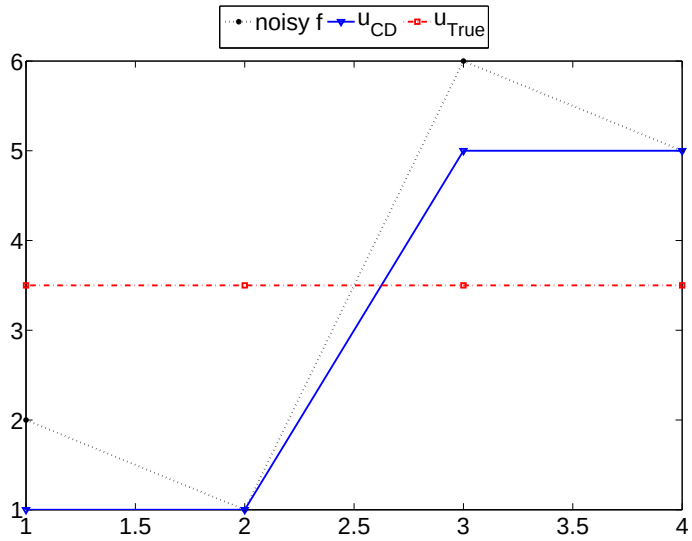

Figure 2.1: Illustration of how coordinate descent might get stuck at a wrong solution.

proposed the multilevel method with a non-standard coarsening scheme based on the "flat" patches of an image. Numerical experiments showed this scheme has better robustness than the standard coarsening and works for most examples. The above multilevel optimization technique has two other potential variants that can handle the difficulty inherited from the non-differentiability. One way is to avoid getting stuck at a wrong solution by using algebraic multigrid(AMG) to automatically find a good coarsening scheme. The other way is to incorporate some randomness in the relaxations to improve robustness, e.g. use simulated annealing as a smoother.

All the existing multigrid methods aim to solve the primal formulations, either the direct minimization problem(1.4) or the associated Euler-Lagrange equation(2.2). One possible alternative is to apply multilevel optimization technique to the dual problem (1.12). The advantage of the dual multilevel method is that the dual relaxation will not get stuck in a "local minimizer" as the formulation is smooth. However, the real challenge here is how to deal with the

constraints. It is not clear that if we can reduce the number of the constraints in a coarser level as we do to the unknowns. So for this technique to work, one has to think creatively about the constraints, not only for our problem, but also for general constrained optimizations.

2.1.7 Graph Cut Algorithm for Anisotropic ROF

Recently graph cut algorithms have been introduced as a fast and exact solver for the discrete anisotropic $\mathrm{TVL}^2/\mathrm{TVL}^1$ models (see [37, 38, 16, 72, 50] and many others).

If the anisotropic TV ($\int_\Omega |u_x| + |u_y|$) is used to replace the isotropic TV ($\int_\Omega |\nabla u|$) and the argument u is restricted to have discrete values, we have the following discrete integer programming problem for the anisotropic ROF model:

$$\min_{u\in\{0,1,\cdots,L-1\}} E(u) = \int_\Omega (|u_x| + |u_y|)\, dxdy + \frac{\lambda}{2}\|u - f\|^2, \tag{2.20}$$

where $L = 2^8$ for 8-bit images and $L = 2^{16}$ for 16-bit images.

The practical use of graph cut algorithm to solve (2.20) relies on the decomposition of an image into its level sets and hence mapping the original problem (2.20) into optimizations of independent binary problems. Using the discrete *co-area formula* (see e.g., [37]), the objective function E in (2.20) can be decomposed into a sum of independent functions E^t which only depends on the sublevel set of u:

$$E(u) = \sum_{t=0}^{L-2} E^t(u^t) + C.$$

Here, u^t is the characteristic functions on the sublevel set: $u^t = \mathbf{1}_{u\le t}$ and C is a constant independent of u.

To minimize $E(\cdot)$ one can minimize all binary problem $E^t(\cdot)$ independently. Thus we get a family $\{\bar{u}^t\}$ which are respectively minimizers of $E^t(\cdot)$. Clearly

the summation will be minimized and thus we have a minimizer of provided this family $\{\bar{u}^t\}$ is monotone:

$$\bar{u}^t \leq \bar{u}^s \quad \forall\, t < s. \tag{2.21}$$

If property (2.21) holds then the optimal solution $\bar{u}$ is given by the reconstruction formula from sublevel sets: $\bar{u}(x) = \min\{t \,:\, \bar{u}^t(x) = 1\}$. If property (2.21) does not hold, then the family $\{\bar{u}^t\}$ is not a function.

Property (2.21) is proved to be true by various authors ([37, 16, 48]). Hence the above optimization strategy is valid. Now note that each $E^t(u^t)$ is a binary MRF with an Ising prior model and be solved efficiently using graph cut algorithms. In [15], the author also did some comparison experiments regarding the efficiency of the discrete graph cuts algorithm and some of methods we discussed in this section.

2.2 Remarks

1. Recently, many variants and extensions to the original ROF model (1.4) have been developed, which include TV and Elastica inpainting ([29], [26]), blind deconvolution ([30]), multi-channel TV ([10]), L^1 fidelity ([57], [22]), multiscale texture decomposition ([52], [65]) and iterative regularization using Bregman distance function. Although the algorithms we listed in this paper are all designed to solve the original ROF model (1.4), it can be naturally adapted to other recently developed models we mentioned above where the main computational challenge is still due to the total variation term.

2. While each of the above methods might have its own comparative advan-

tage in certain situations, some benchmark rules still apply. Independent of the mutigrid technique, there are two general categories: those that need solve a linear system at each iteration and those that need not. To obtain a visually satisfactory solution, the methods in the latter class are preferred for their simplicity and fast initial convergence rate. Out of this category, Chambolle's method becomes very popular because of its simplicity, robustness, no need to use numerical regularization and relative fast convergence. On the other hand, if our goal is to obtain a state-of-art highly accurate solution with low residual/duality gap, the locally superlinear Newton type methods will win eventually even though they have to solve a linear system at every iteration. In this category, The CGM method and SOCP are most widely used for their robustness and fast asymptotic convergence. Finally, if the problem is of very large scale, we need to apply multilevel schemes to reduce complexity.

2.3 Motivations for Developing New Methods

As we have seen, most existing numerical algorithms to solve ROF models (1.4) or (1.12) can be loosely divided into two categories: those that need to solve a linear system of equations at each iteration (implicit) and those that require only a matrix-vector multiplication in the discrete setting (explicit). Generally speaking, the implicit methods (e.g. CGM, HS, and SOCP) have fast asymptotic convergence rates and can provide highly accurate benchmark solutions. However, explicit methods are preferred in many situations for their simplicity and their convergence with relatively little computational effort to medium-accurate and visually satisfactory results. Their low memory requirements make them even more attractive for large-

scale problems. To illustrate the high memory requirements of implicit schemes, we note that an image of size 512×512 is close to the limit of what the SOCP solver MOSEK can handle on a workstation with 2GB of memory.

In this chapter, we shall develop some simple yet efficient algorithms. They are explicit so the memory requirement is low and each iteration only takes $O(N)$ operations. They converge very fast to visually satisfactory solutions and also have much improved asymptotical convergence rate compared with existing explicit methods. In all of our proposed algorithms, we try to exploit the use of the dual variable since a pure primal formulation usually requires some numerical smoothing parameter that would prevent the resulting algorithm from converging to the true optimizer. In other words, our algorithm is either a primal-dual type algorithm or simply a duality based algorithm.

CHAPTER 3

Primal-Dual Hybrid Gradient Descent Method

In this chapter, we propose a simple yet very efficient algorithm for total variation (TV) minimizations with applications in the image processing realm. This descent-type algorithm alternates between the primal and dual formulations and exploit the information from both the primal and dual variables. Therefore, it is able to converge significantly faster than either pure primal/dual gradient descent methods and some other popular existing methods as demonstrated in the numerical experiments. Finally, we show that this idea works for other optimization problems whose objective also involves the non-smooth L^1 norm as in the TV term.

3.1 The Proposed Algorithm

Previous developed gradient descent type methods are the primal time marching method in [60] and Chambolle's duality based semi-implicit gradient descent type method in [14]. In this chapter, we refer to these two methods as the *primal gradient descent* algorithm and *dual gradient descent* algorithm. They are showed briefly as follows.

Primal ROF:

$$\min_{y\in\mathbb{R}^N} \sum_{l=1}^{N} \|A_l^T y\| + \frac{\lambda}{2}\|y - z\|^2, \tag{3.1}$$

Dual ROF:

$$\max_{x\in X} \|Ax - \lambda z\|^2$$
$$\text{where} \quad X = \{x : \ x \in \mathbb{R}^{2N}, \ \|x_l\| \le 1 \text{ for } l = 1, 2, \cdots, N\} \tag{3.2}$$

Primal gradient descent algorithm (smoothed with β):

$$y^{k+1} = y^k - \theta_k\Big(\frac{1}{\lambda}\sum_{l=1}^{N}\frac{A_l A_l^T y^k}{\sqrt{\|A_l^T y^k\|^2 + \beta}} + y^k - z\Big), \tag{3.3}$$

Primal gradient descent algorithm (unsmoothed subgradient):

$$y^{k+1} = y^k - \theta_k\Big(\frac{1}{\lambda}\sum_{l=1}^{N} A_l x_l^k + y^k - z\Big), \tag{3.4}$$

where

$$x_l^k = \begin{cases} \frac{A_l y_l^k}{\|A_l y_l^k\|}, & \text{if } A_l y_l^k \neq 0 \\ \text{any element in the unit ball } B(0,1) \subset \mathbb{R}^2, & \text{else} \end{cases} \tag{3.5}$$

Dual gradient descent algorithm (Chmbolle):

$$x_l^{k+1} \ = \ \frac{x_l^k - \tau_k A_l^T(Ax^k - \lambda z)}{1 + \tau_k\|A_l^T(Ax^k - \lambda z)\|}. \tag{3.6}$$

We notice that the *primal gradient algorithm* (3.3) or (3.4) works exclusively on the primal formulation while the *dual gradient algorithm* focus exclusively on the dual formulation. There is no cross communications between the primal variable and dual variable. Our approach, on the other hand, work on the primal-dual formulation and try to exploit the information of the primal and dual variables simultaneously. We hope the use of both primal and dual information will speed up the gradient descent algorithm.

Our approach can be most effectively illustrated under the setting of the primal-dual formulation (1.8), which has the following discrete form:

$$\min_{y\in\mathbb{R}^N}\max_{x\in X} \Phi(y, x) := \ y^T Ax + \frac{\lambda}{2}\|y - z\|^2 \tag{3.7}$$

where

$$X = \big\{x = (x_1; \cdots; x_N) : \ x_l \in \mathbb{R}^2, \ \|x_l\| \leq 1 \text{ for } l = 1, \cdots, N\big\}$$

Given any intermediatesolution (y^k, x^k) at iteration step k, the proposed algorithm updates the solution as follows.

1. Apply one step of (projected) gradient ascent method to the maximization problem

$$\max_{x \in X} \Phi(y^k, x). \tag{3.8}$$

The ascent direction $\nabla_x \Phi(y^k, x^k) = A^T y^k$, so we update x as

$$x^{k+1} = P_X\big(x^k + \tau_k \lambda A^T y^k\big), \tag{3.9}$$

where τ_k is the (dual) stepsize and P_X denotes the projection onto the set X which can be simply computed in our case (see later remarks). The factor λ is used here so that the stepsize τ_k is sensitive to different problems or different scales of gray levels, which also explains the same situation in (3.11).

2. Apply one step of gradient descent method to the minimization problem

$$\min_{y \in \mathbb{R}^N} \Phi(y, x^{k+1}). \tag{3.10}$$

The ascent direction is $\nabla_y \Phi(y^k, x^{k+1}) = Ax^{k+1} + \lambda(y^k - z)$ and therefore the update is

$$y^{k+1} = y^k - \theta_k(\frac{1}{\lambda} A x^{k+1} + y^k - z), \tag{3.11}$$

where θ_k is the (primal) stepsize.

Put them all together, we have the following algorithm.

Algorithm *PDHGD*

Step 0. ***Initialization.*** Pick y^0 and a feasible $x^0 \in X$, set $k \leftarrow 0$.

Step 1. Choose stepize τ_k and θ_k.

Step 2. ***Updating.***

$$x^{k+1} = P_X\big(x^k + \tau_k \lambda A^T y^k\big) \tag{3.12a}$$

$$y^{k+1} = (1-\theta_k)y^k + \theta_k(z - \frac{1}{\lambda}Ax^{k+1}) \tag{3.12b}$$

Step 3. Terminate if a stopping criterion is satisfied;
otherwise set $k \leftarrow k+1$ and return to step 1.

3.2 Remarks

1. The hybrid descent algorithm can also be developed as a (primal-dual) proximal method:

$$x^{k+1} = \arg\max_{x \in X} \ \Phi(y^k, x) - \frac{1}{2\lambda\tau_k}\|x - x^k\|^2 \tag{3.13a}$$

$$y^{k+1} = \arg\min_{y \in \mathbb{R}^N} \ \Phi(y, x^{k+1}) + \frac{\lambda(1-\theta_k)}{2\theta_k}\|y - y^k\|^2 \tag{3.13b}$$

2. The projection P_X in (3.13a) can be computed in the following straightforward way:

$$\Big(P_X(x)\Big)_l = \frac{x_l}{\max\{\|x_l\|,\ 1\}}, \quad l = 1, 2, \ldots, N. \tag{3.14}$$

Here, since X is a Cartesian product of unit Euclidean balls, the above operation (4.11) actually projects each 2×1 subvector of x separately onto the unit ball in $\mathbb{R}^2$.

3. Both problem (3.8) and (3.10) can be solved exactly, which would yield the following updating formula (taking $\tau_k = \infty$ and $\theta_k = 1$ in (3.23)):

$$x_l^{k+1} = \frac{A_l^T y^k}{\|A_l^T y^k\|}, \quad \text{for } l = 1, \cdots, N \tag{3.15a}$$

$$y^{k+1} = z - \frac{1}{\lambda} A x^{k+1}. \tag{3.15b}$$

However, we choose not to do so since the above algorithm does not converge.

As a special case, if we only solve subproblem (3.8) exactly (taking $\tau_k = \infty$ in (3.23)), the resulting algorithm would be

$$y^{k+1} = y^k - \theta_k \Big(\frac{1}{\lambda} \sum \frac{A_l A_l^T y^k}{\|A_l^T y^k\|} + y^k - z \Big), \tag{3.16}$$

The above algorithm is exactly a subgradient descent method for the primal formulation (4.1).

Another special case is that we solve subproblem (3.10) exactly (taking $\theta_k = 1$ in (3.23)) and still apply gradient ascent method to (3.8). The resulting algorithm is as follows

$$P_X \Big(x^k - \tau_k A^T (A x^k - \lambda z) \Big), \tag{3.17}$$

which is a (projected) gradient descent method for dual problem (3.2).

Hence, the *primal subgradient descent method* and the *dual projected gradient descent method* are two special cases of our algorithm, which correspond to taking special stepsizes $\tau_k = \infty$ and $\theta_k = 1$ respectively in (3.23).

4. The convergence of the *PDHGD* algorithm is (empirically) observed for a variety of suitable stepsize pairs (τ, θ). Moreover, the choice of stepsizes are insensitive to different problems or scales of gray levels. In some range, the stability constraint on the stepsizes seems more relevant to the product

$\tau\theta$ rather than to τ or θ individually. For example, our experiments show that convergence can be obtained for the stepsize pair (2, 0.2) as well as for (4, 0.1). Finally, the numerical results also reveal that a pair of relatively small τ and large θ gives faster initial convergence rate and the opposite choice gives faster asymptotic convergence. Therefor, we can optimize the performance of the algorithm through some strategy of choosing (τ_k, θ_k), although simple fixed stepsizes might already give satisfactory results.

3.3 Extensions

Our algorithm can be naturally extended to other TV image restoration models without any major modifications.

Constrained ROF Model

The original constrained primal ROF model (1.1) has the following discrete form

$$(\mathbf{P}) \qquad \min_{y \in Y} \sum_{l=1}^{N} \|A_l^T y\|, \tag{3.18}$$

where $Y \equiv \{y \in \mathbb{R}^N : \|y - z\| \leq n\sigma\}$.

The above problem (3.18) has the following primal-dual and dual formulations

$$(\mathbf{PD}) \qquad \min_{y \in Y} \max_{x \in X} \; y^T A x \tag{3.19}$$

$$(\mathbf{D}) \qquad \max_{x \in X} \; -n\sigma\|Ax\| + z^T A x \tag{3.20}$$

Both the primal problem (3.18) and dual problem (3.20) are difficult to solve since their objectives are non-smooth. However, our proposed approach based on

the primal-dual formulation (3.19) are simply

$$x^{k+1} = P_X\big(x^k + \frac{\tau_k}{\sigma} A^T y^k\big) \tag{3.21a}$$

$$y^{k+1} = P_Y\big(y^k + \sigma\theta_k A x^{k+1}\big), \tag{3.21b}$$

where the projection P_Y is given by

$$P_Y(y) = z + \frac{y - z}{\max\big\{\|y - z\|/(n\sigma),\ 1\big\}} \tag{3.22}$$

The full *primal-dual hybrid gradient descent method* for *constrained* ROF model is shown as follows:

Algorithm *PDHGDC*

Step 0. ***Initialization.*** Pick y^0 and a feasible $x^0 \in X$, set $k \leftarrow 0$.

Step 1. Choose stepize τ_k and θ_k.

Step 2. ***Updating.***

$$x^{k+1} = P_X\big(x^k + \frac{\tau_k}{\sigma} A^T y^k\big) \tag{3.23a}$$

$$y^{k+1} = P_Y\big(y^k + \sigma\theta_k A x^{k+1}\big), \tag{3.23b}$$

Step 3. Terminate if a stopping criterion is satisfied; otherwise set $k \leftarrow k + 1$ and return to step 1.

TV Debluring Model

The total variation based image restoration model (1.4) can be extended to

recover blurry and noisy image f by solving the following problem:

$$(\mathbf{P}) \qquad \min_u \int_\Omega |\nabla u| \; + \; \frac{\lambda}{2}\|Ku - f\|_2^2 \tag{3.24}$$

where K is a given linear blurring operator and every other term is defined the same as in (1.4). In this model, f is formulated as the sum of a Gaussian noise v and a blurry image Ku resulting from the linear blurring operator K acting on the clean image $\bar{u}$, i.e., $f = Ku + v$.

Among all linear blurring operators, many are shift-invariant and can be expressed in the form of convolution:

$$(Ku)(x) = (h * u)(x) = \int_\Omega h(x - y)\, u(y)\, dy, \tag{3.25}$$

where h is the given *point spread function* (PSF) associated with K.

The discrete form of model (3.24) is

$$\min_{y \in \mathbb{R}^N} \; \sum_{l=1}^{N} \|A_l^T y\| + \frac{\lambda}{2}\|By - z\|^2, \tag{3.26}$$

where B is the discretization of the blurring operator K.

The primal-dual and dual formulation of (3.26) can be obtained as follows

$$(\mathbf{PD}) \qquad \min_{y \in \mathbb{R}^N} \max_{x \in X} \; y^T Ax + \frac{\lambda}{2}\|By - z\|^2 \tag{3.27}$$

$$(\mathbf{D}) \qquad \max_{x \in X} \; -\frac{1}{2\lambda}\|B^{-1}Ax - \lambda z\| + \frac{\lambda}{2}\|z\|^2. \tag{3.28}$$

However, the blurring matrix B is highly ill-posed (non-invertible in some cases), making it difficult if not impossible to compute the inverse B^{-1}. Therefore the dual formulation (3.28) is of little use in practice and sometimes may not even exist. On the other hand, our *primal-dual hybrid gradient descent* algorithm based on formulation (3.27) still works well. The core part of the algorithm are

given as follows

$$x^{k+1} = P_X\big(x^k + \tau_k \lambda A^T y^k\big) \tag{3.29a}$$

$$y^{k+1} = y^k - \theta_k\Big(\frac{1}{\lambda} A x^{k+1} + B^T(By^k - z)\Big). \tag{3.29b}$$

The full *primal-dual hybrid gradient descent method* for TV debluring is:

Algorithm *PDHGD_Deblur*

Step 0. ***Initialization.*** Pick y^0 and a feasible $x^0 \in X$, set $k \leftarrow 0$.

Step 1. Choose stepize τ_k and θ_k.

Step 2. ***Updating.***

$$x^{k+1} = P_X\big(x^k + \tau_k \lambda A^T y^k\big) \tag{3.30a}$$

$$y^{k+1} = y^k - \theta_k\Big(\frac{1}{\lambda} A x^{k+1} + B^T(By^k - z)\Big). \tag{3.30b}$$

Step 3. Terminate if a stopping criterion is satisfied;
otherwise set $k \leftarrow k + 1$ and return to step 1.

3.4 Theoretical Connections

Our method is related to projection type methods existing in the literature for finding saddle points and, more generally, solutions to variational inequalities. In this section, we shall discuss very briefly about the framework of projection methods for solving variational inequalities and point out the connections and

difference between our method and previous work. We refer interested readers to the survey papers [45] and [71] for the background of this area.

Let H be a real Hilbert space (in our case, $\mathbb{R}^n$), whose inner product and norm are denoted by $\langle\cdot\rangle$, and $\|\cdot\|$ respectively. Let K be a closed convex set in H and F be a mapping from H into itself. We now consider the problem of finding $v^* \in K$ such that

$$\langle v - v^*, F(v^*)\rangle \geq 0, \quad \forall\, v \in K. \tag{3.31}$$

The above problem is called a *variational inequality* problem with v^* being one of its solution. We denote the above variational inequality problem by $\mathrm{VI}(K, F)$. In most real applications, K is convex and F satisfy some monotonicity and Lipschitz continuity properties, which we defined as follows:

Definition 1. *F is said to be*

(*i*) *monotone if* $\langle u - v, F(u) - F(v)\rangle \geq 0 \quad \forall\, u, v \in H.$

(*ii*) *strongly monotone if*

$$\exists\, \nu > 0 \text{ s.t. } \langle u - v, F(u) - F(v)\rangle \geq \nu\|u - v\|^2 \quad \forall\, u, v \in H.$$

(*iii*) *pseudomonotone if* $\langle u - v, F(v)\rangle > 0 \Rightarrow \langle u - v, F(u)\rangle \geq 0 \;\forall\, u, v \in H.$

(*iv*) *Lipschitz continuous if* $\exists\, L > 0$ *s.t.* $\|F(u) - F(v)\| \leq L\|u - v\| \;\forall\, u, v \in H.$

Finding a saddle point (y^*, x^*) to the min-max problem

$$\min_{y \in Y} \max_{x \in X} \Phi(y, x)$$

can be written as a special case of the variational inequality problem:

$$\text{find } v^* \in K \quad \text{s.t.} \quad \langle v - v^*,\, F(v^*)\rangle \geq 0 \quad \forall\, v \in K, \tag{3.32}$$

where

$$v = \begin{bmatrix} y \\ x \end{bmatrix} \quad F(v) = \begin{bmatrix} \Phi_y(x, y) \\ -\Phi_x(x, y) \end{bmatrix} \quad \text{and } K = Y \times X.$$

In particular Our ROF problem (3.19) and (3.7) can both be transformed into a variational inequality problem VI(K, F) in (3.32) with F and K defined as follows.

For unconstrained ROF (3.19):

$$F(v) = \begin{bmatrix} Ax + \lambda(y - z) \\ -A^T y \end{bmatrix} \quad \text{and } K = \mathbb{R}^N \times X.$$

For constrained ROF (3.7):

$$F(v) = \begin{bmatrix} Ax \\ -A^T y \end{bmatrix} \quad \text{and } K = Y \times X.$$

Variational inequality problem is closely related to the fixed-point problem. The fixed-point theory has played an important role in the development of various algorithms for solving variational inequalities. In fact we have the following well-known result (see, e.g. [8, pp. 267]):

Lemma 1. *v^* is a solution of VI(K, F) if and only if*

$$v^* = P_K\big(v^* - \alpha F(v^*)\big) \quad \textit{for any } \alpha > 0.$$

The fixed-point formulation in the above lemma suggests the simple iterative algorithm of solving for u^*.

VI Algorithm 1.

$$v^{k+1} = P_K\big(v^k - \alpha_k F(v^k)\big). \tag{3.33}$$

The convergence of the above algorithm requires F to be strongly monotone and Lipschitz continuous, which is too restrictive in many cases. An alternative approach is to consider the following 'implicit' iterative scheme

VI Algorithm 2.

$$v^{k+1} = P_K\big(v^k - \alpha_k F(v^{k+1})\big). \tag{3.34}$$

The convergence of this new algorithm only requires monotonicity of F but it is often difficulty to solve the implicit update at each iteration, making it less practical.

To overcome the drawbacks of the projection methods defined in (3.33) and (3.34), Korpelevich [51] first proposed a modified method called *extragradient algorithm*. It consists two projections at each iteration: a predictor step and a corrector step.

VI Algorithm 3.

$$\bar{v}^k = P_K\big(v^k - \alpha_k F(v^k)\big) \tag{3.35a}$$

$$v^{k+1} = P_K\big(v^k - \alpha_k F(\bar{v}^k)\big) \tag{3.35b}$$

Global convergence is proved for the above algorithm if F is pseudomonotone or Lipschitz continuous or the problem satisfy some local error bound (see [71]), provided the step size α_k is small enough to satisfy

$$\alpha_k \|F(v^k) - F(\bar{v}^k)\| \leq \mu \|v^k - \bar{v}^k\| \quad \text{for some fixed } \mu \in (0,1),$$

which can be obtained by simple Armijo type line search.

There are many other variants of the original extragradient algorithm with different predictor search rule and corrector step size aiming to improve performance (see [58] and [71]). New related developments in this direction can also be found in [54] and [58], where the final solution is obtained by averaging along the solution path.

Our ROF problem (3.19) and (3.7) can both be transformed into a variational inequality problem VI(K, F) with a monotone and Lipschitz continuous mapping

F. Some of the existing algorithms can be applied directly with proved global convergence. However, numerical experiments show that none of these existing methods has comparable performance to our algorithm. There are many possible explanations for this. First of all, in the variational inequality setting the variables y and x are combined as one variable u and have to be updated in one step with same steplength; while in our approach the primal y and dual x are updated alternatively in a Gauss-seidal type of way with freedom to choose their own step sizes. More importantly, all the existing algorithms are developed to solve variational inequalities as a general class; while our method exploits the particular information of the problem, including the bilinear function F and special structure of the set K, which allow us to choose optimal step size to improve the performance. On the other hand, our approach is lacking of a global convergence proof which would be useful to provide some benchmark rules and can help us better understand how the algorithm works.

3.5 Implementation Details and Numerical Experiments

We report on computational experiments for three test problems in image denoising. All the programs are run in an IBM T60 Notebook PC with 1.83 GHz Intel Core Duo CPU and 1G RAM. All methods are coded in MATLAB. It is expected that the performance can be improved by recoding in C or C++, but we believe that improvements would be fairly uniform across all the algorithms.

We test three problems on image denoising. The original clean images and the input noisy images are shown in Figure 7.1. The size of the two test problems are $256 \times 256, 512 \times 512$ and 512×512 respectively. The noisy images are generated by adding Gaussian noises with standard deviation $\sigma = 20$ to the original clean images.

The parameter λ in the unconstrained ROF model (3.1) is inverse related to the noise level σ and usually need to be tuned for each individual image. In our case, λ is chosen in the following way. We first compute the constrained ROF model by algorithm *PDHGDC* for the optimality solution (y^*, x^*). Then the particular λ that will make the unconstrained ROF model (3.1) equivalent to the constrained model (3.18) is given by

$$\lambda = \frac{\|Ax^*\|}{n\sigma}. \tag{3.36}$$

For our test problems, the parameters λ obtained in the above way are $0.053, 0.037$ and 0.049 respectively.

We tested the following algorithms for denoising problems:

- Chambolle's semi-implicit gradient descent method [14];
- Primal-dual hybrid descent methods proposed in Section 3.1;
- The CGM method of [25].

Although suitable constant stepsizes will give good convergence results, the power of our proposed algorithm shall be most exploit with some optimal strategy of choosing stepsizes $(\tau_k,\ \theta_k)$. Throughout the experiments, we use the following stepsize strategy:

Algorithm *PDHGD* $\quad \tau_k = 0.2 + 0.08k, \quad \theta_k = (0.5 - \frac{5}{15+k})/\tau_k;$

Algorithm *PDHGDC* $\quad \tau_k = 0.2 + 0.08k \quad \theta_k = 0.5/\tau_k.$

In Chambolle's method, we take the time step to be 0.248 for its near optimal performance. In the CGM implementation, we used a direct solver for the linear system at each iteration (the conjugate gradient iterative solver was slower on

these examples). The smooth parameter β is dynamically updated based on duality gap rather than fixed. In particular we take $\beta^{(0)} = 100$ and let $\beta^{(k)} = \beta^{(k-1)} \cdot \left(\frac{G^{(k)}}{G^{(k-1)}}\right)^2$. We noticed that this simple strategy of updating β borrowed from interior-point methods outperforms the classical CGM measured by the decrease of duality gap.

The decision about when an approximate solution is of sufficiently high quality to terminate the algorithm can be difficult for general constrained optimization problems. Often, we wish the approximate solution x to be close to a global minimizer x^* and/or the function value $F(x)$ be close to $F(x^*)$. In the denoising case, the duality gap provides a reliable and easily calculated stopping criterion. We terminate our program whenever the relative duality gap $\frac{G(y^k,x^k)}{D(x^k)}$ reaches a pre-specified tolerance threshold TOL.

Tables 6.1 and 6.2 report numbers of iterations and CPU times required by two $PDHD$ algorithms as well as by Chambolle's algorithm and CGM method for the relative duality gap to achieve certain threshold. In all codes, we used the same starting point $(y^0, x^0) = (z, 0)$. (Convergence does not depend on initial conditions though.) We vary the threshold TOL from 10^{-2} to 10^{-6}, producing results of increasingly high accuracy as TOL is decreased.

The results in the tables demonstrates that our proposed approaches is very competitive to existing methods. They are the winners for all tests with different stopping criterions TOL $= 10^{-2}, 10^{-4}, 10^{-6}$. It is significantly faster than Chambolle's method to obtain medium-high accurate solutions and significantly faster than CGM method to obtain low-medium accurate solutions.

Figure 3.2 shows the denoised images obtained at different values of TOL. Note that visually there is little difference between the results obtained with two tolerance values 10^{-2} and 10^{-4}. Smaller values of TOL do not produce further

visual differences.

Figure 7.2 plots the relative duality gap against the number of iterations for Chambolle's method as well as for the *PDHGD* algorithm. It shows that the *PDHGD* converges much faster than Chambolle's method, especially when a medium-high accurate solution is required.

3.6 Conclusion

We have proposed a primal-dual hybrid gradient descent method to solve the total variation based image restoration model of Rudin, Osher and Fatemin (ROF) [60]. The algorithm tries to improve performance by alternating between the primal and dual variable and exploit information from both variables. We compare our method with two popular existing approaches proposed by Chambolle [14] and Chan, Golub, and Mulet [25] and show our method is consistently faster than earlier approaches in all experiments with different stopping criterions. The complexity of our method is $O(N)$ at each iteration, in fact its cost is basically the same as Chamboll's method at each iteration; while the cost of the CGM method at each iteration is $O(N^{\frac{3}{2}})$ for solving the block tri-diagonal linear system.

Our algorithm can be applied to solve both the unconstrained ROF and constrained ROF model and, in theory, it can be applied to solve other TV minimization model or L^1 minimization problem by transforming it to a min-max form. We also pointed out that our algorithm is related to existing projection type methods for solving variational inequalities.

Figure 3.1: Denoising test problems. Left: original clean image. Right: noisy image with Gaussian noise ($\sigma = 20$). First row: test problem 1, 256×256 *cameraman*. Middle row: test problem 2, 512×512 *barbara*. Bottom row: test problem 3, 512×512 *boat*

Table 3.1: Number of iterations and CPU times (in seconds) for problem 1.

	TOL $= 10^{-2}$		TOL $= 10^{-4}$		TOL $= 10^{-6}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	45	1.77	1213	61.3	22597	1159
PDHD	14	0.45	73	2.28	328	10.4
PDHDC	14	0.46	70	2.27	308	9.98
CGM	6	20.3	14	49.8	19	68.0

Table 3.2: Number of iterations and CPU times (in seconds) for problem 2.

	TOL $= 10^{-2}$		TOL $= 10^{-4}$		TOL $= 10^{-6}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	141	26.5	3083	578	47935	8988
PDHD	25	3.66	117	23.7	541	110
PDHDC	24	3.66	113	22.9	519	108
CGM	7	191	15	417	23	642

Table 3.3: Number of iterations and CPU times (in seconds) for problem 3.

	TOL $= 10^{-2}$		TOL $= 10^{-4}$		TOL $= 10^{-6}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	61	11.4	1218	228	22235	4168
PDHD	16	2.36	72	10.6	320	47.2
PDHDC	16	2.38	71	10.7	316	47.5
CGM	7	189	14	382	20	547

Figure 3.2: The denoised images with different level of termination criterions. left column: $\text{TOL} = 10^{-2}$, right column: $\text{TOL} = 10^{-4}$.

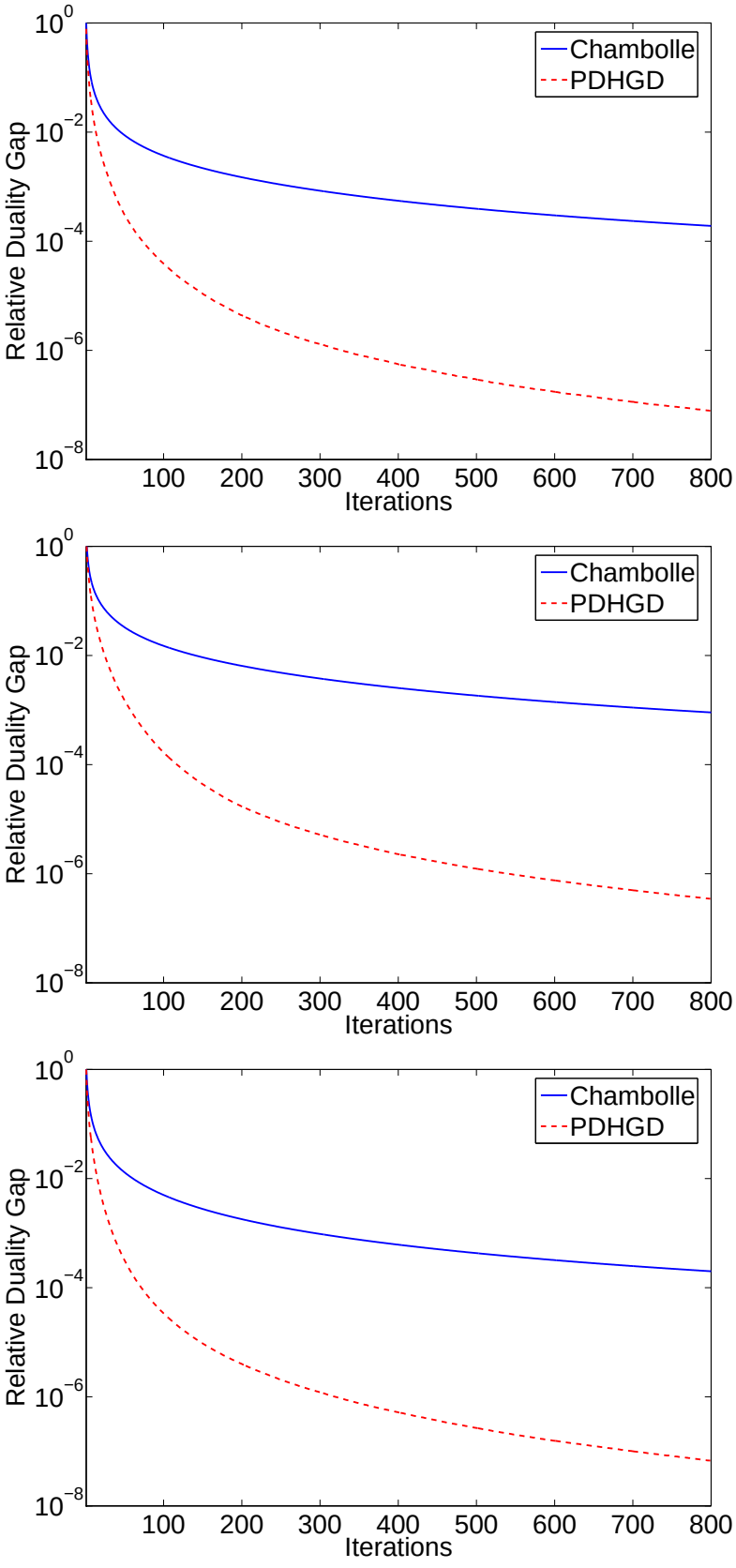

Figure 3.3: Plot of relative duality gap $\frac{D(y^k)-D(x^k)}{D(x^k)}$ v.s. Iterations. Top left: test problem 1. Top right: test problem2. Bottom: test problem 3.

CHAPTER 4

Duality Based Gradient Projection Method

The discrete primal and dual formulations of the unconstrained ROF model is given as

$$(\mathbf{P}) \quad \min_{y} \sum_{l=1}^{N} \|A_l^T y\|_2 + \frac{\lambda}{2}\|y - z\|_2^2 \tag{4.1}$$

$$(\mathbf{D}) \quad \min_{x \in X} \frac{1}{2}\|Ax - \lambda z\|^2$$

$$\text{where } X \equiv \big\{x = (x_1; \cdots; x_N) : \ x_l \in \mathbb{R}^2, \|x_l\| \leq 1 \text{ for } l = 1, \cdots, N\big\}, \tag{4.2}$$

In this chapter, we report on the development, implementation, and testing of some simple but fast explicit algorithms. These algorithms are based on the following dual formulation (4.2) so they do not require any numerical smoothing parameters that would prevent them from converging to the true optimizer.

4.1 A Fundamental Convergence Result

Before we discuss about the proposed algorithm we make several remarks on the discretized problems (4.1), (4.2) and prove a general convergence result that will be useful later. It is easy to verify that both problems can be obtained from the function $\ell : \mathbb{R}^N \times X \to \mathbb{R}$ defined as follows:

$$\ell(y, x) := x^T A^T v + \frac{\lambda}{2}\|y - z\|_2^2. \tag{4.3}$$

The primal problem (4.1) is simply

$$\min_{v \in \mathbb{R}^N} \max_{x \in X} \ell(y, x),$$

while the dual problem (4.2) is equivalent to

$$\max_{x \in X} \min_{v \in \mathbb{R}^N} \ell(y, x).$$

It is easy to verify that the conditions (H1), (H2), (H3), and (H4) of [47, pp. 333-334] are satisfied by this setting. Thus, it follows from [47, Chapter VII, Theorem 4.3.1] that ℓ has a nonempty convex set of saddle points $(\bar{y}, \bar{x}) \in \mathbb{R}^N \times X$. Moreover, from [47, Chapter IV, Theorem 4.2.5] and compactness of X, the point $(\bar{y}, \bar{x}) \in \mathbb{R}^N \times X$ is a saddle point if and only if $\bar{y}$ solves (4.1) and $\bar{x}$ solves (4.2).

Note that by strict convexity of the objective in (4.1), the solution $\bar{y}$ of (4.1) is in fact uniquely defined. For any saddle point $(\bar{y}, \bar{x})$, we have that $\ell(\bar{y}, \bar{x}) \leq \ell(y, \bar{x})$ for all $v \in \mathbb{R}^N$, that is, $\bar{y}$ is a minimizer of $\ell(\cdot, \bar{x})$. Thus, from optimality conditions for $\ell(\cdot, \bar{x})$, the following relationship is satisfied for the unique solution $\bar{y}$ of (4.1) and for any solution $\bar{x}$ of (4.2):

$$A\bar{x} + \lambda(\bar{y} - z) = 0. \tag{4.4}$$

By uniqueness of $\bar{y}$, it follows that $A\bar{x}$ is constant for all solutions $\bar{x}$ of (4.2).

The following general convergence result will be useful in our analysis of algorithms in Section 4.2.

Proposition 1. *Let $\{x^k\}$ be any sequence with $x^k \in X$ for all $k = 1, 2, \ldots$ such that all accumulation points of $\{x^k\}$ are stationary points of (4.2). Then the sequence $\{v^k\}$ defined by*

$$v^k = g - \frac{1}{\lambda} A x^k \tag{4.5}$$

converges to the unique solution $\bar{y}$ of (4.1).

Proof. Note first that all stationary points of (4.2) are in fact (global) solutions of (4.2), by convexity.

Suppose for contradiction that $v^k \not\to \bar{y}$. Then we can choose $\epsilon > 0$ and a subsequence $\mathcal{S}$ such that $\|v^k - \bar{y}\|_2 \geq \epsilon$ for all $k \in \mathcal{S}$. Since all x^k belong to the bounded set X, the sequence $\{x^k\}$ is bounded, so $\{v^k\}$ is bounded also. In particular, the subsequence $\{v^k\}_{k\in\mathcal{S}}$ must have an accumulation point $\hat{v}$, which must satisfy $\|\hat{v} - \bar{y}\|_2 \geq \epsilon > 0$. By restricting $\mathcal{S}$ if necessary, we can assume that $\lim_{k\in\mathcal{S}} v^k = \hat{v}$. By boundedness of $\{x^k\}$, we can further restrict $\mathcal{S}$ to identify a point $\hat{x} \in X$ such that $\lim_{k\in\mathcal{S}} x^k = \hat{x}$. By (4.5), we thus have

$$A\hat{x} + \lambda(\hat{v} - z) = 0 = \lim_{k\in\mathcal{S}} Ax^k + \lambda(v^k - z) = 0, \tag{4.6}$$

Since $\hat{x}$ is an accumulation point of the whole sequence, we have by assumption that $\hat{x}$ is a stationary point and hence a solution of (4.2). By our observation following (4.4), we thus have that $A\hat{x}+\lambda(\bar{y}-z) = 0$, where $\bar{y}$ is the unique solution of (4.1). By comparing this expression with (4.6), we obtain the contradiction $\hat{v} = \bar{y}$, proving the result. □

4.2 Gradient Projection Algorithms

4.2.1 Introduction

Our proposed approaches are essentially gradient projection algorithms applied to (4.2), in which the search path from each iterate is obtained by projecting negative-gradient (steepest descent) directions onto the feasible set. Various enhancements involving different step-length rules and different line-search strategies are important in making the method efficient.

For the general problem of optimizing a smooth function over a closed convex

set, that is,

$$\min_{x \in X} F(x) \tag{4.7}$$

(where $F : \mathbb{R}^m \to \mathbb{R}$ is smooth and X is a closed convex subset of $\mathbb{R}^m$), gradient projection methods set

$$x^{k+1} = x^k + \gamma_k (x^k(\alpha_k) - x^k), \tag{4.8}$$

for some parameter $\gamma_k \in [0, 1]$, where

$$x^k(\alpha_k) := P_X(x^k - \alpha_k \nabla F(x^k)), \tag{4.9}$$

for some $\alpha_k > 0$. Here, P_X denotes the projection onto the set X. Since X is closed and convex, the operator P_X is uniquely defined, but in order for the gradient projection approach to make practical sense, this operator must also be easy to compute. For this reason, gradient projection approaches have been applied most often to problems with separable constraints, where X can be expressed as a Cartesian product of low-dimensional sets. In our case (4.2), X is a cross product of unit balls in $\mathbb{R}^2$, so computation of P_X requires only $O(N)$ operations. Bertsekas [7] gives extensive background on gradient projection algorithms.

From now on, we will focus on the solution of problem (4.2), which we restate here:

$$\min_{x \in X} F(x) := \frac{1}{2} \|Ax - \lambda g\|_2^2, \tag{4.10}$$

where the compact set $X \subset \mathbb{R}^{2N}$ is defined in (4.2). In this section, we discuss GP techniques for solving this problem. Our approaches move from iterate x^k to the next iterate x^{k+1} using the scheme (4.8)-(4.9).

Projection P_X on the set X, a Cartesian product of unit Euclidean balls, can be computed straightforwardly as follows.

$$\Big(P_X(x)\Big)_l = \frac{x_l}{\max\{\|x_l\|,\ 1\}}, \qquad l = 1, 2, \ldots, N. \tag{4.11}$$

This operation projects each 2×1 subvector of x separately onto the unit ball in $\mathbb{R}^2$. It is worth pointing out here that this structure of the dual constraints, which makes the gradient projection approach practical, also enables Chambolle to develop an analytical formula for the Lagrange multipliers in [14].

Our approaches below differ in their rules for choosing the step parameters α_k and γ_k in (4.8) and (4.9).

4.2.2 Three Frameworks

We next consider three gradient projection frameworks that encompass our gradient projection algorithms, and present convergence results for methods in these frameworks.

Framework GP-NoLS chooses α_k in some predetermined range and sets $\gamma_k \equiv 1$.

Framework GP-NoLS

Step 0. ***Initialization.*** Choose parameters $\alpha_{\min}$, $\alpha_{\max}$ with $0 < \alpha_{\min} < \alpha_{\max}$. Choose x^0 and set $k \leftarrow 0$.

Step 1. Choose steplength $\alpha_k \in [\alpha_{\min},\ \alpha_{\max}]$.

Step 2. Set $x^{k+1} = x^k(\alpha_k)$.

Step 3. Terminate if a stopping criterion is satisfied; otherwise set $k \leftarrow k + 1$ and go to Step 1.

Framework GP-ProjArc also sets $\gamma_k \equiv 1$, but chooses α_k by a backtracking line search to satisfy an Armijo criterion, which enforces monotonic decrease of F. This approach is referred to by Bertsekas [7, p. 236] as *Armijo Rule Along the Projection Arc*.

Framework GP-ProjArc

Step 0. ***Initialization.*** Choose parameters $\alpha_{\min}$, $\alpha_{\max}$ with $0 < \alpha_{\min} < \alpha_{\max}$, and choose $\rho \in (0,1)$ and $\mu \in (0, \frac{1}{2})$. Choose x^0 and set $k \leftarrow 0$.

Step 1. Choose initial steplength $\bar{\alpha}_k \in [\alpha_{\min},\ \alpha_{\max}]$.

Step 2. ***Backtracking Line Search.*** Choose m to be the smallest nonnegative integer such that

$$F(x^k(\rho^m \bar{\alpha}_k)) \leq F(x^k) - \mu \nabla F(x^k)^T (x^k - x^k(\rho^m \bar{\alpha}_k)),$$

where $x^k(\alpha)$ is defined as in (4.9);
Set $\alpha_k = \rho^m \bar{\alpha}_k$ and $x^{k+1} = x^k(\alpha_k)$.

Step 3. Terminate if a stopping criterion is satisfied; otherwise set $k \leftarrow k+1$ and go to Step 1.

Framework GP-LimMin fixes α_k at the start of each iteration (possibly using information gathered on previous steps), but then performs a "limited minimization" procedure to find γ_k, again ensuring decrease of F at every step.

Framework GP-LimMin

Step 0. ***Initialization.*** Choose parameters $\alpha_{\min}$, $\alpha_{\max}$ with $0 < \alpha_{\min} < \alpha_{\max}$. Choose x^0 and set $k \leftarrow 0$.

Step 1. Choose steplength $\alpha_k \in [\alpha_{\min},\ \alpha_{\max}]$. Compute $x^k(\alpha_k)$ and set $\delta^k := (x^k(\alpha_k) - x^k)$.

Step 2. ***Limited Minimizing Line Search.*** Set $x^{k+1} = x^k + \gamma_k \delta^k$, with $\gamma_k = \text{mid}(0,\ \gamma_{k,\text{opt}},\ 1)$ and

$$\gamma_{k,\text{opt}} = \arg\min\ F(x^k + \gamma \delta^k) = \frac{-(\delta^k)^T \nabla F(x^k)}{\|A\delta^k\|_2^2} \tag{4.12}$$

Step 3. Terminate if a stopping criterion is satisfied; otherwise set $k \leftarrow k + 1$ and go to Step 1.

The first algorithm we consider — Algorithm GPCL — is obtained from Framework GP-NoLS by setting α_k equal to the fixed value $\alpha > 0$ at every step. Convergence is obtained for all α sufficiently small, as we now show.

Theorem 1. *Let $\{x^k\}$ be a sequence generated by Algorithm GPCL. Then if $0 < \alpha < .25$, the sequence v^k obtained from (4.5) converges to the unique solution $\bar{y}$ of (4.1).*

Proof. Given any two vectors x' and x'' we have that

$$\nabla F(x') - \nabla F(x'') = A^T A(x' - x''),$$

so the Lipschitz constant for ∇F is $\|A^T A\|_2$, which is bounded by 8 (see [14, p. 92]). It follows immediately from [7, Proposition 2.3.2] that every accumulation point of $\{x^k\}$ is stationary for (4.10) provided that $0 < \alpha < .25$. The result now follows immediately from Proposition 1. □

The upper bound of .25 in Theorem 1 is tight; we observe in practice that the method is unstable evn for $\tau = .251$.

We consider other algorithms in Framework GP-NoLS below, in which α_k takes on different values at each iteration that may violate the bound of .25. These methods may be non-monotone (the function F may increase on some iterations) and convergence results cannot be proven in general without the addition of step acceptance criteria, such as the requiring a significant improvement over the worst function value at the last M points visited for some positive integer M; see [9]. Strategies are also required for modifying steps that fail these criteria.

For algorithms in Framework GP-ProjArc , we have the following convergence result.

Theorem 2. *Let $\{x^k\}$ be a sequence generated by an algorithm in Framework GP-ProjArc . Then the sequence v^k obtained from (4.5) converges to the unique solution $\bar{y}$ of (4.1).*

Proof. Proposition 2.3.3 of Bertsekas [7], with minor modifications for the variable choice of $\bar{\alpha}_k$ within the range $[\alpha_{\min}, \alpha_{\max}]$, shows that all limit points of $\{x^k\}$ are stationary. The result then follows from Proposition 1. □

An identical result holds for algorithms in Framework GP-LimMin .

Theorem 3. *Let $\{x^k\}$ be a sequence generated by an algorithm in Framework GP-LimMin . Then the sequence v^k obtained from (4.5) converges to the unique solution $\bar{y}$ of (4.1).*

Proof. Proposition 2.3.1 of Bertsekas [7], with minor modifications for the variable choice of $\bar{\alpha}_k$ within the range $[\alpha_{\min}, \alpha_{\max}]$, shows that all limit points of $\{x^k\}$ are stationary. The result then follows from Proposition 1. □

4.2.3 Barzilai-Borwein Strategies

We discuss strategies that choose α_k using approaches first proposed by Barzilai and Borwein [6] (BB) and subsequently elaborated by other authors. For the problem $\min_{x\in\mathbb{R}^{2N}} F(x)$, the basic BB strategy sets $x^{k+1} \leftarrow x^k - \alpha_k \nabla F(x^k)$, where α_k is chosen so that $\alpha_k^{-1} I$ mimics the behavior of the Hessian $\nabla^2 F$ over the previous step. By Taylor's theorem, we have

$$\nabla^2 F(x^k)\Delta x^{k-1} \approx \Delta g^{k-1}, \qquad \Delta x^{k-1} \approx (\nabla^2 F(x^k))^{-1}\Delta g^{k-1},$$

where

$$\Delta x^{k-1} := x^k - x^{k-1}, \qquad \Delta g^{k-1} := \nabla F(x^k) - \nabla F(x^{k-1}),$$

so our desired property on α is that $\alpha^{-1}\Delta x^{k-1} \approx \Delta g^{k-1}$. Note that for the F we consider here (4.10), we have $\Delta g^{k-1} = A^T A \Delta x^{k-1}$.

One formula for α is obtained by performing a least-squares fit in one variable, as follows:

$$\alpha_{k,1} = \left[\arg\min_{\tau\in\mathbb{R}} \|\tau\Delta x^{k-1} - \Delta g^{k-1}\|_2^2\right]^{-1},$$

which yields

$$\alpha_{k,1} = \frac{\|\Delta x^{k-1}\|_2^2}{\langle \Delta x^{k-1}, \Delta g^{k-1}\rangle} = \frac{\|\Delta x^{k-1}\|_2^2}{\|A\Delta x^{k-1}\|_2^2}. \tag{4.13}$$

An alternative formula is obtained similarly, by doing a least-squares fit to α rather than α^{-1}, to obtain

$$\alpha_{k,2} = \arg\min_{\alpha\in\mathbb{R}} \|\Delta x^{k-1} - \alpha\Delta g^{k-1}\|_2^2 = \frac{\langle \Delta x^{k-1}, \Delta g^{k-1}\rangle}{\|\Delta g^{k-1}\|_2^2} = \frac{\|A\Delta x^{k-1}\|_2^2}{\|A^T A\Delta x^{k-1}\|_2^2}. \tag{4.14}$$

These step lengths were shown in [6] to be effective on simple problems; a partial analysis explaining the behavior was given. Numerous variants have been proposed recently, and subject to with theoretical and computational evaluation. The BB stepsize rules have also been extended to constrained optimization, particularly to bound-constrained quadratic programming; see, for example [36] and [62]. The same formulae (4.13) and (4.14) can be used in these cases.

Other variants of Barzilai-Borwein schemes have been proposed by other authors in other contexts. The cyclic Barzilai-Borwein (CBB) method proves to have better performance than the standard BB in many cases (see for example [35] and the references therein). In this approach, we recalculate the BB stepsize from one of the formulae (4.13) or (4.14) at only every mth iteration, for some integer m. At intervening steps, we simply use the last calculated value of α_k. There are alternating Barzilai-Borwein (ABB) schemes that switch between the definitions (4.13) and (4.14), either adaptively or by following a fixed schedule.

4.2.4 Implemented Variants of Gradient Projection

We discuss here the variants of gradient projection that were implemented in our computational testing.

Algorithm GPLS. This algorithm falls into GP-PA, where we choose the initial steplength $\bar{\alpha}_k$ at each iteration by predicting what the steplength would be if no new constraints were to become active on this step. Specifically, we define the vector g^k by

$$g_i^k = \begin{cases} (\nabla F(x^k))_l, & \text{if } \|x_l^k\|_2 < 1 \text{ or } (\nabla F(x^k))_l^T x_l^k > 0, \\ \left[I - x_l^k (x_l^k)^T\right] (\nabla F(x^k))_l, & \text{otherwise.} \end{cases}$$

We then choose the initial guess to be

$$\bar{\alpha}_k = \arg\min_{\alpha} F(x^k - \alpha g^k),$$

which can be computed explicitly as

$$\bar{\alpha}_k = \frac{(g^k)^T \nabla F(x^k)}{\|Ag^k\|_2^2} = \frac{\|g^k\|_2^2}{\|Ag^k\|_2^2}.$$

In practice, we find that using $\frac{1}{2}\bar{\alpha}_k$ as the initial value gives better performance, and backtracking is not necessary in any of our numerical experiments.

Algorithm GPBB-NM. This is a nonmonotone Barzilai-Borwein method in GP-NoLS, in which we obtain the step α_k via the formula (4.13), projected if necessary onto the interval $[\alpha_{\min}, \alpha_{\max}]$.

Algorithm GPBB-M. A monotone Barzilai-Borwein method in GP-LM, in which α_k is obtained as in Algorithm GPBB-NM.

Algorithm GPCBB-m. A monotone cyclic Barzilai-Borwein algorithm in GP-LM, in which α_k is recalculated from (4.13) at every mth iteration. Formally, we set

$$\alpha_{ml+i} = \alpha_{ml+1}^{BB} \quad \text{for } l = 0, 1, 2, \ldots \text{ and } i = 1, 2, \ldots, m-1,$$

where α_{ml+1}^{BB} is obtained from (4.13) with $k = ml + 1$, restricted to the interval $[\alpha_{\min}, \alpha_{\max}]$.

Algorithm GPABB. A monotonic alternating Barzilai-Borwein method in GP-LM, in which the technique of Serafini, Zanghirati, and Zanni [62, Section 2.2], is used to switch between the rules (4.13) and (4.14). This technique makes use of two positive integer parameters $n_{\min}$ and $n_{\max}$ with $0 < n_{\min} \leq n_{\max}$. Let n_α be the number of consecutive iterations that use the same steplength selection rule, (4.13) or (4.14). We switch from one rule to the other at the next iteration $k+1$ if either (i) $n_\alpha \geq n_{\max}$ or (ii) $n_\alpha \geq n_{\min}$ and α_k is either a *separating steplength* or a *bad descent generator*. The current step α_k is a *separating steplength* if it lies between the values generated by the two rules at the next iteration, that is, $\alpha_{k+1,2} < \alpha_k < \alpha_{k+1,1}$. Given two constants γ_l and γ_u with $0 < \gamma_l \leq 1 \leq \gamma_u$, we say that α_k is a *bad descent generator* if one of the following conditions holds:

(a) $\gamma_{k,\text{opt}} < \gamma_l$ and $\alpha_k = \alpha_{k,1}$; or

(b) $\gamma_{k,\text{opt}} > \gamma_u$ and $\alpha_k = \alpha_{k,2}$.

where $\gamma_{k,\text{opt}}$ is obtained from the limited minimization rule (4.12). We refer interested readers to [62] for the rationale of the criterion. In any case, the chosen α_k is adjusted to ensure that it lies in the interval $[\alpha_{\min}, \alpha_{\max}]$.

4.3 A Sequential Quadratic Programming Algorithm

We describe here a variation on the techniques of the previous section in which the curvature of the boundary of the constraint set X is accounted for in computing the search direction. The method can be viewed as a sequential quadratic programming (SQP) method applied to the dual formulation (4.10). The KKT optimality conditions for this formulation can be written as follows:

$$\begin{aligned} A_l^T(Ax - \lambda g) + 2z_l x_l = 0, \qquad & l = 1, 2, \ldots, N, \\ 0 \le z_l \ \perp \ \|x_l\|^2 - 1 \le 0, \qquad & l = 1, 2, \ldots, N, \end{aligned}$$

where the scalars z_l are Lagrange multipliers for the constraints $\|x_l\|_2^2 \le 1$, $l = 1, 2, \ldots, N$, and the operator $\perp$ indicates that at least one of its two operands must be zero. At iteration k, we compute an estimate of the active set $\mathcal{A}_k \subset \{1, 2, \ldots, N\}$, which are those indices for which we believe that $\|x_l\|_2^2 = 1$ at the solution. In our implementation, we choose this set as follows:

$$\begin{aligned} \mathcal{A}_k &= \{l \mid \|x_l^k\|_2 = 1 \ \text{ and } \ (x_l^k)^T[\nabla F(x^k)]_l \le 0\} \\ &= \{l \mid \|x_l^k\|_2 = 1 \ \text{ and } \ (x_l^k)^T A_l^T(Ax^k - \lambda g) \le 0\}. \end{aligned} \tag{4.15}$$

The SQP step is a Newton-like step for the following system of nonlinear equations, from the current estimates x^k and z_l^k, $l = 1, 2, \ldots, N$:

$$A_l^T(Ax - \lambda g) + 2x_l z_l = 0, \qquad l = 1, 2, \ldots, N, \tag{4.16a}$$

$$\|x_l\|_2^2 - 1 = 0, \qquad l \in \mathcal{A}_k, \tag{4.16b}$$

$$z_l = 0, \qquad l \notin \mathcal{A}_k. \tag{4.16c}$$

Using $\tilde{z}_l^{k+1}$ to denote the values of z_l at the next iterate, and $\tilde{d}^k$ to denote the step in x^k, a "second-order" step can be obtained from (4.16) by solving the following system for $\tilde{d}^k$ and $\tilde{z}_l^{k+1}$, $l = 1, 2, \ldots, N$:

$$A_l^T A\tilde{d}^k + 2\tilde{z}_l^{k+1}\tilde{d}_l^k = -A_l^T[Ax^k - \lambda g] - 2x_l^k\tilde{z}_l^{k+1}, \;\; l = 1, 2, \ldots, N, \tag{4.17a}$$

$$2(x_l^k)^T\tilde{d}_l^k = 0, \;\; l \in \mathcal{A}_k, \tag{4.17b}$$

$$\tilde{z}_l^{k+1} = 0, \;\; l \notin \mathcal{A}_k. \tag{4.17c}$$

We now define Newton-like steps d^k in x, and new iterates z^{k+1} in z, by replacing A^TA by $\alpha_k^{-1}I$ in (4.17a) and solving the following linear system:

$$\alpha_k^{-1}d_l^k + 2z_l^{k+1}d_l^k = -A_l^T[Ax^k - \lambda g] - 2x_l^k z_l^{k+1}, \;\; l = 1, 2, \ldots, N, \tag{4.18a}$$

$$2(x_l^k)^T d_l^k = 0, \;\; l \in \mathcal{A}_k, \tag{4.18b}$$

$$z_l^{k+1} = 0, \;\; l \notin \mathcal{A}_k. \tag{4.18c}$$

Considering indices $l \in \mathcal{A}_k$, we take the inner product of (4.18a) with x_l^k and use (4.18b) and (4.15) to obtain:

$$z_l^{k+1} = -(1/2)(x_l^k)^T A_l^T(Ax^k - \lambda g), \;\; l \in \mathcal{A}_k.$$

We obtain the steps d_l^k for these indices by substituting this expression in (4.18a):

$$d_l^k = -(\alpha_k^{-1} + 2z_l^{k+1})^{-1}\left[A_l^T(Ax^k - \lambda g) + 2x_l^k z_l^{k+1}\right], \;\; l \in \mathcal{A}_k.$$

In fact, because of (4.18c), this same formula holds for $l \notin \mathcal{A}_k$, when it reduces to the usual negative-gradient step

$$d_l^k = -\alpha_k A_l^T (Ax^k - \lambda g), \;\; l \notin \mathcal{A}_k.$$

We define the (nonmonotone) Algorithm SQPBB-NM by making an initial choice of α_k at each iteration according to the formula (4.13), and calculating $x^{k+1} = x^k + d^k$ and z^{k+1} as described above. In the monotone variant of this method, known as SQPBB-M, we successively decrease α_k by a factor of $\rho \in (0, 1)$, as in Framework GP-ProjArc , and recalculate x^{k+1} and z^{k+1} as above, until a decrease in the objective function is obtained.

We also tried versions of these methods in which α_k was recalculated only on every mth iteration; these are referred to as SQPBB-NM(m) and SQPBB-M(m), respectively.

4.4 Computational Experiments

We report on computational experiments for three test problems in image denoising. The original clean images and the input noisy images are shown in Figures 5.2 and 5.3, respectively. The sizes of the discretizations for the three test problems are 128×128, 256×256, and 512×512, respectively. The noisy images are generated by adding Gaussian noise to the clean images using the MATLAB function `imnoise`, with variance parameter set to 0.01. The fidelity parameter λ is taken to be 0.045 throughout the experiments. This parameter is inversely related to the noise level σ and usually needs to be tuned for each individual image to get an optimal visual result.

We tested the following algorithms:

- Chambolle's semi-implicit gradient descent method [14];
- many variants of gradient projection proposed in Section 4.2;
- the SQP method of Section 4.3;
- the CGM method of [25].

We report on a subset of these tests here, including the gradient projection variants that gave consistently good results across the three test problems.

In Chambolle's method, we take the step to be 0.248 for near optimal performance, although global convergence is proved in [14] only for steps in the range $(0, .125)$. We use the same value $\alpha_k = 0.248$ in Algorithm GPCL, as it appears to be near optimal in this case as well.

For all gradient projection variants, we set $\alpha_{\min} = 10^{-5}$ and $\alpha_{\max} = 10^5$. (Performances are insensitive to these choices, as long as $\alpha_{\min}$ is sufficiently small and $\alpha_{\max}$ sufficiently large.) In Algorithm GPLS, we used $\rho = 0.5$ and $\mu = 10^{-4}$. In Algorithm GPABB, we set $\gamma_l = 0.1$ and $\gamma_u = 5$.

We also tried variants of the GPBB methods in which the initial choice of α_k was scaled by a factor of 0.5 at every iteration. We found that this variant often enhanced performance. This fact is not too surprising, as we can see from Section 4.3 that the curvature of the boundary of constraint set X suggests that it is appropriate to add positive diagonal elements to the Hessian approximation, which corresponds to decreasing the value of α_k.

In the CGM implementation, we used a direct solver for the linear system at each iteration, as the conjugate gradient iterative solver (which is an option in the CGM code) was slower on these examples. The smooth parameter β is dynamically updated based on duality gap from iteration to iteration. In

particular, we take $\beta_0 = 100$ and let $\beta_k = \beta_{k-1} \left(G_k / G_{k-1}\right)^2$, where G_k and G_{k-1} are the duality gaps for the past two iterations. This simple strategy for updating β, which is borrowed from interior-point methods, outperforms the classical CGM approach, producing faster decrease in the duality gap.

All methods are coded in MATLAB. It is likely the performance can be improved by recoding in C or C++, but we believe that improvements would be fairly uniform across all the algorithms.

Tables 6.1, 6.2, and 6.3 report number of iterations and average CPU times over ten runs, where each run adds a different random noise vector to the true image. In all codes, we used the starting point $x^{(0)} = 0$ in each algorithm and the relative duality gap stopping criterion. We vary the threshold TOL from 10^{-2} to 10^{-6}, producing results of increasingly high accuracy as TOL is decreased.

Figure 5.4 shows the denoised images obtained at different values of TOL. Note that visually there is little difference between the results obtained with two tolerance values 10^{-2} and 10^{-4}. Smaller values of TOL do not produce further visual differences.

The tables show that on all problems, the proposed gradient projection algorithms are competitive to Chambolle's method, and that some variants are significantly faster, especially when moderate accuracy is required for the solutions. Two variants stood out as good performers: the GPBB-NM variant and the GPBB-M(3) variant in which the initial choice of α_k was scaled by 0.5 at each iteration. For all tests with TOL $= 10^{-2}$, TOL $= 10^{-3}$, and TOL $= 10^{-4}$, the winner was one of the gradient-projection Barzilai-Borwein strategies.

For these low-to-moderate accuracy requirements, CGM is generally slower than the gradient-based methods, particularly on the larger problems. The picture changes considerably, however, when high accuracy (TOL $= 10^{-6}$) is required.

Figure 4.1: The original clean images for our test problems. Left: 128×128 "shape"; middle: 256×256 "cameraman"; right: 512×512 "Barbara".

Figure 4.2: The input noisy images for our test problems. Gaussian noise is added using MATLAB function `imnoise` with variance 0.01.

The rapid asymptotic convergence of CGM is seen to advantage in this situation, and it outperforms the gradient-based methods in all cases. This result is observed again in figure 4.4, which plots the duality gap again the CPU time cost for Chambolle's method, CGM method and the GPBB-NM variant of the gradient projection algorithm.

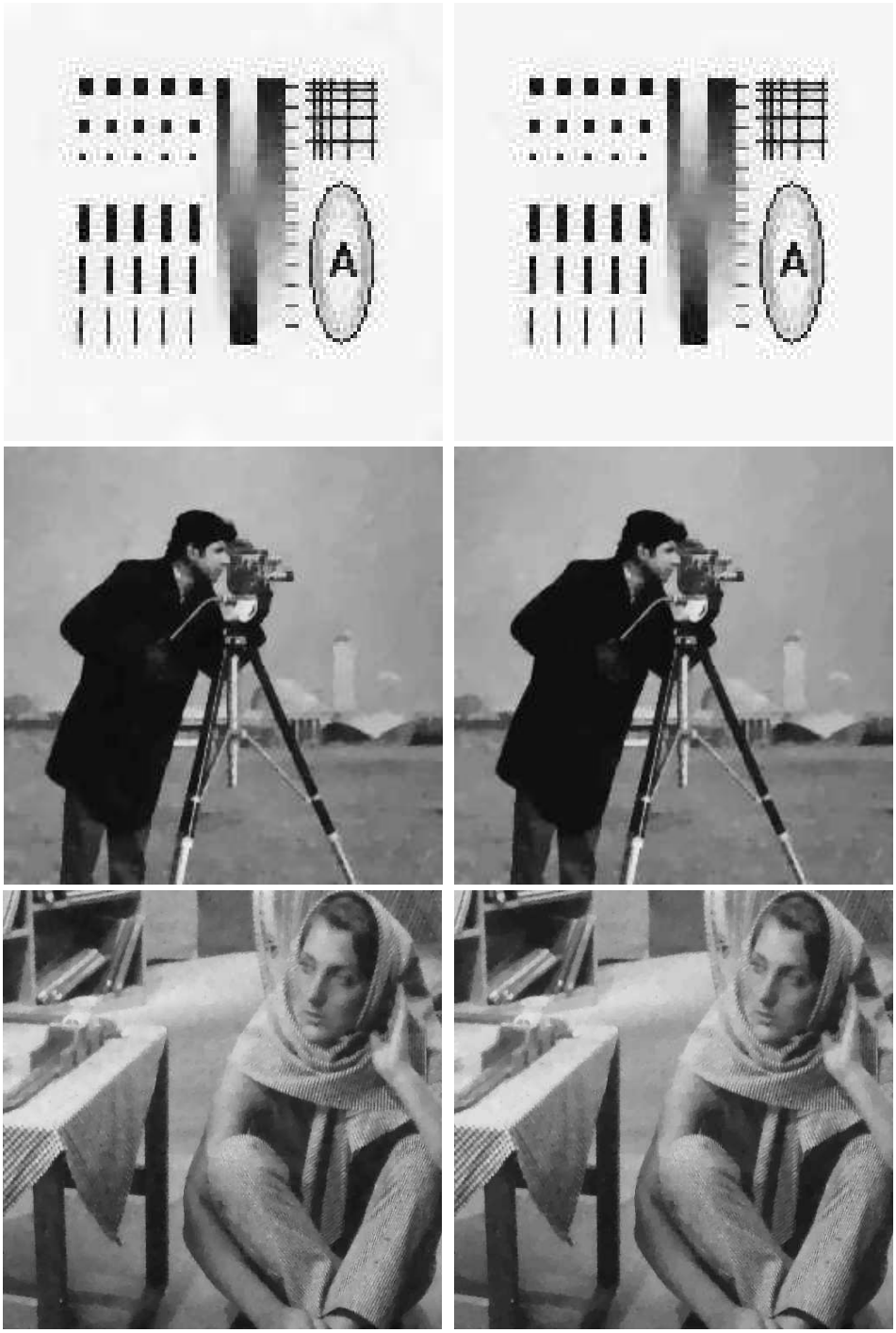

Figure 4.3: The denoised images with different level of termination criterions. left column: $\text{TOL} = 10^{-2}$, right column: $\text{TOL} = 10^{-4}$.

Table 4.1: Number of iterations and CPU times (in seconds) for problem 1. $*$ = initial α_k scaled by 0.5 at each iteration.

	TOL $= 10^{-2}$		TOL $= 10^{-3}$		TOL $= 10^{-4}$		TOL $= 10^{-6}$	
Algorithms	Iter	Time	Iter	Time	Iter	Time	Iter	Time
Chambolle	18	0.12	169	1.15	1082	7.23	23884	154
GPCL	39	0.25	136	0.90	736	4.71	16196	110
GPLS	23	0.33	181	2.66	845	12.8	17056	260
GPBB-M	12	0.16	152	1.82	836	10.1	17464	167
GPBB-M (2)	18	0.17	187	1.80	941	9.00	21146	193
GPBB-M(3)	13	0.11	92	0.78	287	2.44	3749	32.2
GPBB-M(3)*	11	0.09	49	0.41	190	1.60	2298	19.7
GPBB-NM	10	0.09	48	0.46	217	2.10	3857	32.3
GPABB	13	0.16	58	0.66	245	2.80	2355	24.0
SQPBB-M	13	0.17	47	0.66	196	2.81	3983	61.0
CGM	6	4.05	9	5.90	12	8.00	18	12.1

Table 4.2: Number of iterations and CPU times (in seconds) for problem 2. $*$ = initial α_k scaled by 0.5 at each iteration.

	TOL $= 10^{-2}$		TOL $= 10^{-3}$		TOL $= 10^{-4}$		TOL $= 10^{-6}$	
Algorithms	Iter	Time	Iter	Time	Iter	Time	Iter	Time
Chambolle	27	1.05	164	6.40	815	31.80	15911	628
GPCL	32	1.26	112	4.31	540	21.0	11434	452
GPLS	20	1.85	132	14.8	575	66	12892	1531
GPBB-M	20	1.14	124	7.01	576	32.7	11776	674
GPBB-M (2)	20	1.12	72	4.05	245	13.9	4377	251
GPBB-M(3)	20	1.12	77	4.33	345	19.5	3522	200
GPBB-M(3) fudge	17	0.95	47	2.65	162	9.17	1766	100
GPBB-NM	16	0.85	48	2.53	178	9.52	2802	150
GPABB	16	1.06	47	3.16	168	11.4	1865	127
SQPBB-M	14	1.10	41	3.44	152	13.5	2653	245
CGM	6	22.30	10	37.5	13	48.8	19	71.0

Table 4.3: Number of iterations and CPU times (in seconds) for problem 3. $*$ = initial α_k scaled by 0.5 at each iteration.

	TOL $= 10^{-2}$		TOL $= 10^{-3}$		TOL $= 10^{-4}$		TOL $= 10^{-6}$	
Algorithms	Iter	Time	Iter	Time	Iter	Time	Iter	Time
Chambolle	27	5.46	131	28.3	534	112	8314	1781
GPCL	24	4.65	80	17.2	328	69.1	5650	1212
GPLS	34	10.9	86	32.5	322	128	5473	2258
GPBB-M	20	5.50	84	24.9	332	98.2	5312	1576
GPBB-M (2)	20	5.40	56	16.3	160	46.6	4408	1290
GPBB-M(3)	20	5.38	67	19.5	174	50.5	2533	738
GPBB-M(3) fudge	17	4.58	41	11.9	131	38.0	1104	321
GPBB-NM	15	3.92	40	11.3	115	32.4	1371	388
GPABB	14	4.57	35	12.3	122	42.8	1118	394
SQPBB-M	14	4.91	40	15.8	109	44.6	1556	646
CGM	7	168	10	216	14	302	21	441

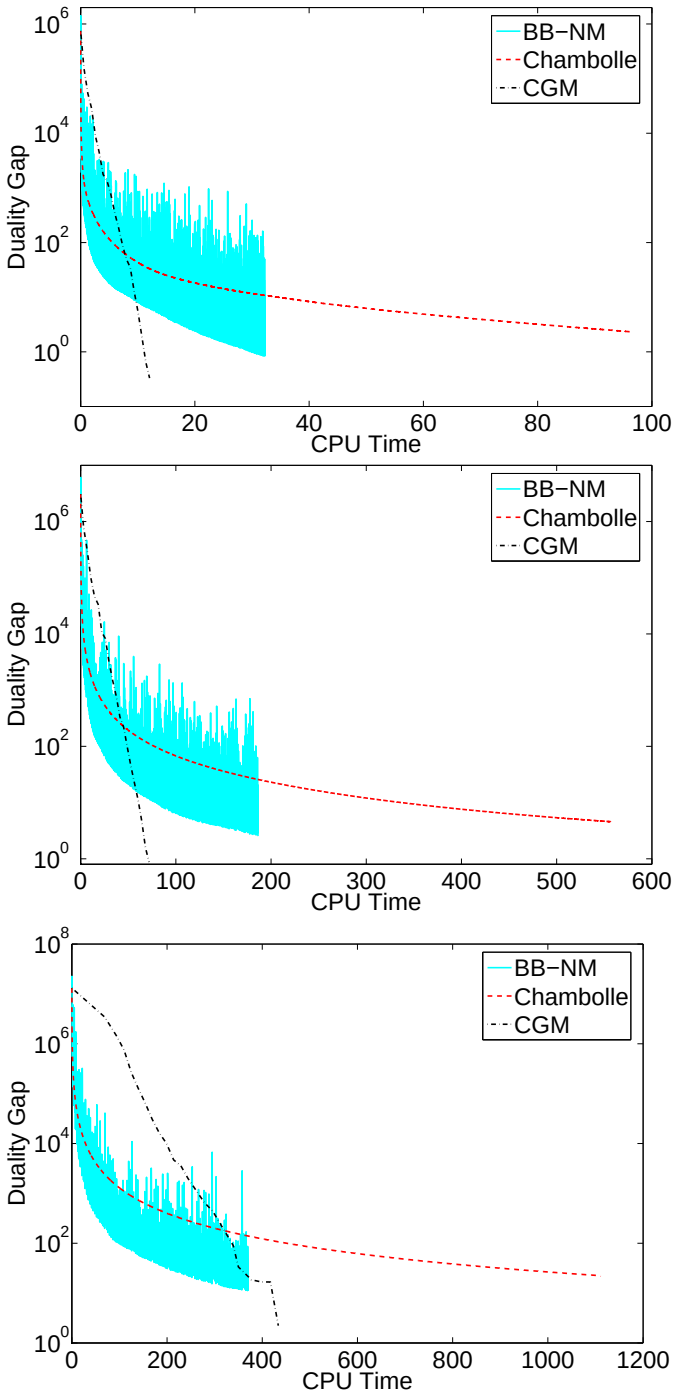

Figure 4.4: Plots of duality gap vs. CPU time cost. Problem 1-3 are shown in the order of top to bottom.

CHAPTER 5

Block Coordinate Descent (BCD) Method

5.1 Framework

In this section, we shall apply block coordinate descent (BCD) or block nonlinear Gauss-Seidal method to solve the dual ROF problem (1.12).

Consider a general optimization problem

$$\begin{aligned} &\text{minimize} \quad F(x) \\ &\text{subject to} \quad x \in X = X_1 \times X_2 \times \cdots \times X_n \in \mathbb{R}^m, \end{aligned} \tag{5.1}$$

where $f : \mathbb{R}^m \to \mathbb{R}$ is a continuously differentiable function and the feasible set X is the Cartesian product of closed, nonempty and convex subsets $X_i \in R^{m_i}$, for $i = 1, \cdots, n$ with $\sum_{i=1}^{n} m_i = m$.

If the vector $x \in R^m$ is partitioned into n component vectors $x_i \in R^{m_i}$, then the minimization version of the block coordinate descent method for the solution of (5.1) is defined by

$$x_i^{k+1} = \arg\min_{y_i \in X_i} F(x_1^{k+1}, \cdots, x_{i-1}^{k+1},\ y_i,\ x_{i+1}^{k}, \cdots, x_n^{k}), \tag{5.2}$$

which update cyclicly for each component of x, starting from a given initial point $x_0 \in X$ and generates a sequence of $\{x^k\}$ with $x^k = (x_1^k \cdots, x_n^k)$.

The discrete dual formulation for TV restoration is

$$\begin{aligned} &\min \ \phi(w) = \|\nabla \cdot w + z\|^2 \\ \text{subject to} \quad &|w_{i,j}|^2 \le 1 \text{ for } i,j = 1 \cdots, n, \end{aligned} \tag{5.3}$$

where $z = \lambda f,\ w = (w_{i,j}) \in \mathbb{R}^{2\times n\times n}$ with each component $w_{i,j} \in \mathbb{R}^2$, i.e.

$$w_{i,j} = \begin{bmatrix} w^1_{i,j} \\ w^2_{i,j} \end{bmatrix}, \quad 1 \le i,j \le n. \tag{5.4}$$

The discrete divergence operator $\nabla\cdot$ is defined in (1.24) which we rewrite it here:

$$(\nabla \cdot w)_{i,j} = \begin{cases} w^1_{i,j} - w^1_{i-1,j} & \text{if } 1 < i < n \\ w^1_{i,j} & \text{if } i = 1 \\ -w^1_{i-1,j} & \text{if } i = n \end{cases} + \begin{cases} w^2_{i,j} - w^2_{i,j-1} & \text{if } 1 < j < n \\ w^2_{i,j} & \text{if } j = 1 \\ -w^2_{i,j-1} & \text{if } j = n. \end{cases} \tag{5.5}$$

Following the idea of the block coordinate descent method (5.2), we fix w at all other pixels except at some interior point (i,j) $(i,j \ne 1 \text{ or } n)$, which lead to a local minimization problem in $\mathbb{R}^2$:

$$\begin{aligned} \min_{\|w_{i,j}\|\le 1} \phi(w_{i,j}) &= [w^1_{i,j} + w^2_{i,j} - (w^1_{i-1,j} + w^2_{i,j-1} - z_{i,j})]^2 \\ &+ [w^1_{i,j} - (w^1_{i+1,j} + w^2_{i+1,j} - w^2_{i+1,j-1} + z_{i+1,j})]^2 \\ &+ [w^2_{i,j} - (w^2_{i,j+1} + w^1_{i,j+1} - w^1_{i-1,j+1} + z_{i,j+1})]^2 \\ &+ \text{ other terms not dependent on } w_{i,j} \end{aligned} \tag{5.6}$$

The local minimization problems on the boundary points can be modified accordingly as in the definition of the divergence operator (5.5). We shall describe them briefly here.

The dual ROF model (5.3) has separable simple Euclidean ball constraints which makes each sub minimization problem (5.6) fairly easy to solve at each

'coordinate'. This simple trait of the constraints that makes the BCD algorithm practical here is also the reason that makes gradient projection practical and enables Chambolle to derive the analytical formula for the Lagrange multipliers.

When $i = 1$:

$$\begin{aligned}\min_{\|w_{i,j}\|\leq 1} \phi(w_{i,j}) &= [w^1_{i,j} + w^2_{i,j} - (w^2_{i,j-1} - z_{i,j})]^2 \\ &+ [w^1_{i,j} - (w^1_{i+1,j} + w^2_{i+1,j} - w^2_{i+1,j-1} + z_{i+1,j})]^2 \\ &+ [w^2_{i,j} - (w^2_{i,j+1} + w^1_{i,j+1} + z_{i,j+1})]^2 \\ &+ \text{ other terms not dependent on } w_{i,j}\end{aligned} \tag{5.7}$$

When $j = 1$:

$$\begin{aligned}\min_{\|w_{i,j}\|\leq 1} \phi(w_{i,j}) &= [w^1_{i,j} + w^2_{i,j} - (w^1_{i-1,j} - z_{i,j})]^2 \\ &+ [w^1_{i,j} - (w^1_{i+1,j} + w^2_{i+1,j} + z_{i+1,j})]^2 \\ &+ [w^2_{i,j} - (w^2_{i,j+1} + w^1_{i,j+1} - w^1_{i-1,j+1} + z_{i,j+1})]^2 \\ &+ \text{ other terms not dependent on } w_{i,j}\end{aligned} \tag{5.8}$$

Following (5.5), the boundary values $\{w^1_{n,j} : j = 1 \cdots, n\}$ and $\{w^2_{i,n} : i = 1, \cdots, n\}$ are not relevant in our problem and we simply set them to be 0. Hence the sub minimization problems on boundary $i = n$ or $j = n$ degenerate to the following 1-D problems.

When $i = n$:

$$\begin{aligned}\min_{|w^2_{i,j}|\leq 1} \phi(w_{i,j}) &= [w^2_{i,j} - (w^2_{i,j-1} + w^1_{i-1,j} - z_{i,j})]^2 \\ &+ [w^2_{i,j} - (w^2_{i,j+1} - w^1_{i-1,j+1} + z_{i,j+1})]^2 \\ &+ \text{ other terms not dependent on } w_{i,j}\end{aligned} \tag{5.9}$$

When $j = n$:

$$\begin{aligned}\min_{|w^1_{i,j}|\le 1} \phi(w_{i,j}) &= [w^1_{i,j} - (w^1_{i-1,j} + w^2_{i,j-1} - z_{i,j})]^2 \\ &+ [w^1_{i,j} - (w^1_{i+1,j} - w^2_{i+1,j-1} + z_{i+1,j})]^2 \\ &+ \text{ other terms not dependent on } w_{i,j}\end{aligned} \tag{5.10}$$

5.2 Implementation

The sub optimization problems (5.6)- (5.10) generated by BCD algorithm are ball constrained quadratic minimization problems with at most two unknowns. We shall now give a generic algorithm to solve each of these subproblems. We shall solve subproblem (5.6- 5.8) in details in the following discussion using (5.6) as an example. The subproblems (5.9) and (5.10) are scalar minimizations and can be solved very easily.

Problem (5.9):

$$\begin{aligned} v &= \frac{1}{2}(w^2_{i,j-1} + w^2_{i,j+1} + w^1_{i-1,j} - w^1_{i-1,j+1} + z_{i,j+1} - z_{i,j}) \\ w^2_{i,j} &= \min\big\{\max\{-1, v\}, 1\big\}. \end{aligned}$$

Problem (5.10):

$$\begin{aligned} v &= \frac{1}{2}(w^1_{i-1,j} + w^1_{i+1,j} + w^2_{i,j-1} - w^2_{i+1,j-1} + z_{i+1,j} - z_{i+1,j}) \\ w^1_{i,j} &= \min\big\{\max\{-1, v\}, 1\big\}. \end{aligned}$$

Problem (5.6- 5.8): First problem (5.6) can be rewritten as

$$\min_{p^2+q^2\le 1} \phi(p, q) = (p - a)^2 + (q - b)^2 + (p + q - c)^2 \tag{5.11}$$

where $w_{i,j} = (p, q)$ and

$$\begin{aligned} a &= w^1_{i+1,j} + w^2_{i+1,j} - w^2_{i+1,j-1} + z_{i+1,j} \\ b &= w^2_{i,j+1} + w^1_{i,j+1} - w^1_{i-1,j+1} + z_{i,j+1} \\ c &= w^1_{i-1,j} + w^2_{i,j-1} - z_{i,j} \end{aligned}$$

To solve problem (5.11), we first solve its unconstrained problem and obtain by first-order conditions

$$\overline{p} = \frac{1}{3}(2a - b + c) \tag{5.12}$$

$$\overline{q} = \frac{1}{3}(2b - a + c) \tag{5.13}$$

The above $(\overline{p}, \overline{q})$ is the solution to (5.11) if $\overline{p}^2 + \overline{q}^2 \leq 1$. Otherwise the constraint is active and we solve the equality-constrained problem by the KKT conditions

$$\begin{aligned} (\lambda + 2)p \ + \ q &= a + c \\ p \ + \ (\lambda + 2)q &= b + c \\ p^2 \ + \ q^2 &= 1 \\ \lambda &> 0 \ , \end{aligned}$$

where λ is the Lagrange multiplier. The first two equations in KKT system yield

$$\begin{aligned} p + q &= \frac{a + b + 2c}{\lambda + 3} \\ p - q &= \frac{a - b}{\lambda + 1} \end{aligned} \tag{5.14}$$

Substituting the above equations to $(p+q)^2 + (p-q)^2 = 2$ we obtain the following equation for the Lagrange multiplier

$$f(\lambda) = \frac{A}{(\lambda + 3)^2} + \frac{B}{(\lambda + 1)^2} - 1 = 0 \tag{5.15}$$

where $A = \frac{1}{2}(a+b+2c)^2$ and $B = \frac{1}{2}(a-b)^2$.

Equation (5.15) yields an a fourth-order polynomial which can be solved analytically. However, it is more effective to solve it numerically by Newton's method. Newton's method is convergent for solving 5.15 with initial condition $\lambda^0 = 0$. Instead of giving a rigorous proof, we shall illustrate our argument with the following figure (5.1). First, there exists a $\lambda^* > 0$ with $f(\lambda^*) = 0$. Then it follows that $f(0) > 0$ by the monotonicity of function $f(\lambda)$ on $(-1, \infty)$. The plot of $f(\lambda)$ looks like the one in figure (5.1). Then it is straightforward to see that the iterates λ^k converges to λ^* with monotonically decreasing $|\lambda^k - \lambda^*|$. In practice it only take couple of steps to obtain an approximate λ^*.

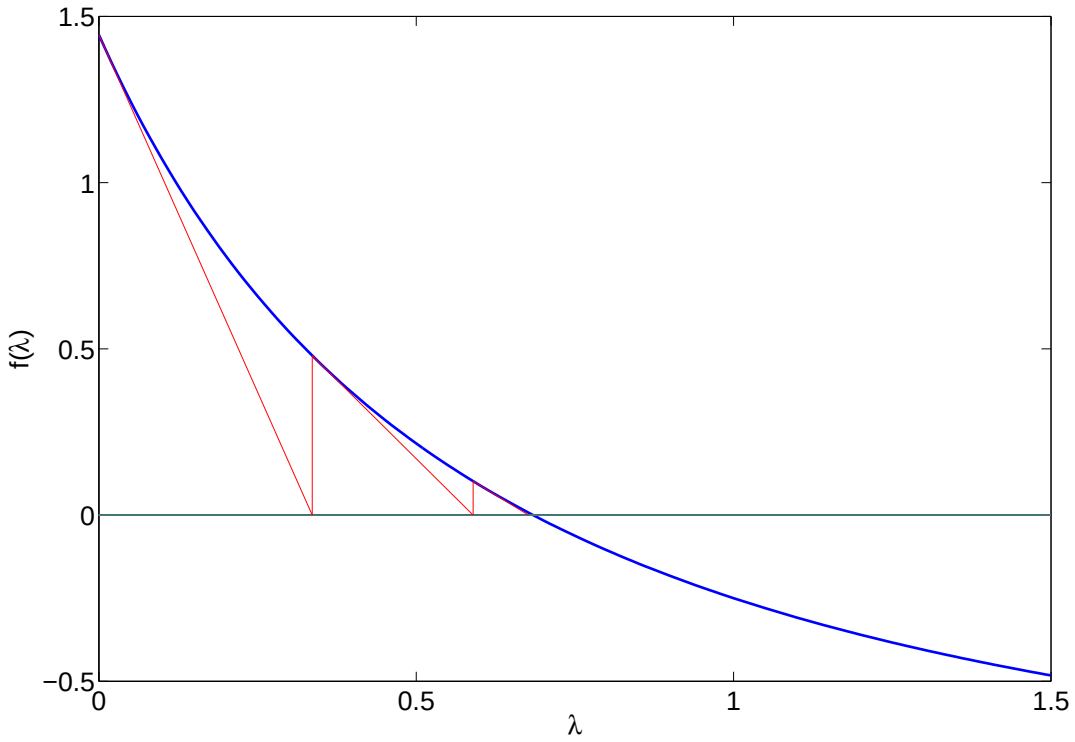

Figure 5.1: Convergence of Newton's method

Substituting the obtained solution of (5.15) λ^* to system (5.14) then gives the

optimal $w_{i,j}$.

The full *BCD* algorithm are given as follows

Algorithm *BCD*

Step 0. ***Initialization.*** Pick initial feasible $w^{(0)}$, set $k \leftarrow 0$.

Step 1. ***Updating.*** For $i, j = 1, \cdots, n$, update $w_{i,j}^{(k+1)}$ as

$$w_{i,j}^{(k+1)} = \arg \min_{\|w_{i,j}\| \leq 1} \left\| \nabla \cdot \left(w_{1,1}^{(k+1)}, \cdots, w_{i,j}, \cdots w_{n,n}^{(k)} \right) - z \right\|^2$$

by the method described earlier in section (5.2).

Step 2. Terminate if a stopping criterion is satisfied;
otherwise set $k \leftarrow k + 1$ and return to step 1.

5.3 BCD for Anisotropic TV

The discrete dual formulation for the anisotropic TV denoising model is

$$\begin{aligned} &\min \ D[w] = \|\nabla \cdot w + z\| \\ \text{subject to} \quad &|w_{i,j}^1| \leq 1 \text{ and } |w_{i,j}^2| \leq 1 \text{ for } i, j = 1 \cdots, n, \end{aligned} \tag{5.16}$$

where w and the $\nabla \cdot w$ are the same as defined in (5.4) and (5.5).

The constraints in problem (5.16) are separable in the scalar level, which makes the subproblem (5.2) of the BCD algorithm even easier to solve. In fact, the algorithm is simply just coordinate descent method. We shall briefly describe the coordinate descent algorithm for the anisotropic dual ROF model as follows.

Suppose (i,j) is not on the boundary. Fix any other arguments except $w^1_{i,j}$, we have the following sub minimization problem

$$\begin{aligned}\min_{|w^1_{i,j}|\le 1} \quad & [w^1_{i,j} - (w^1_{i-1,j} - w^2_{i,j} + w^2_{i,j-1} - z_{i,j})]^2 \\ + \quad & [w^1_{i,j} - (w^1_{i+1,j} + w^2_{i+1,j} - w^2_{i+1,j-1} + z_{i+1,j})]^2 \\ + \quad & \text{other terms not dependent on } w^1_{i,j},\end{aligned} \tag{5.17}$$

which can be easily solved as

$$\begin{aligned} v &= \frac{1}{2}(w^1_{i-1,j} + w^1_{i+1,j} - w^2_{i,j} + w^2_{i,j-1} + w^2_{i+1,j} - w^2_{i+1,j-1} + z_{i+1,j} - z_{i+1,j}) \\ w^1_{i,j} &= \min\big\{\max\{-1, v\}, 1\big\}. \end{aligned} \tag{5.18}$$

Similarly, fixing any other arguments except $w^1_{i,j}$ gives

$$\begin{aligned}\min_{|w^2_{i,j}|\le 1} \quad & [w^2_{i,j} - (w^2_{i,j-1} + w^1_{i-1,j} - w^1_{i,j} - z_{i,j})]^2 \\ + \quad & [w^2_{i,j} - (w^2_{i,j+1} + w^1_{i,j+1} - w^1_{i-1,j+1} + z_{i,j+1})]^2 \\ + \quad & \text{other terms not dependent on } w^1_{i,j},\end{aligned} \tag{5.19}$$

which can be solved as

$$\begin{aligned} v &= \frac{1}{2}(w^2_{i,j-1} + w^2_{i,j+1} + w^1_{i,j+1} - w^1_{i,j} + w^1_{i-1,j} - w^1_{i-1,j+1} + z_{i,j+1} - z_{i,j}) \\ w^2_{i,j} &= \min\big\{\max\{-1, v\}, 1\big\}. \end{aligned} \tag{5.20}$$

The boundary points can be treated the same way as in the isotropic case. We can also unify all the points into one framework by setting $w^1_{0,j}, w^1_{n,j}, w^2_{i,0}, w^2_{i,n}$ to be 0 for $i,j = 1,\cdots n-1$ and iterates through $i = 1,\cdots,n-1; j = 1,\cdots,n$ for $w^1_{i,j}$ and iterates through $i = 1,\cdots,n; j = 1,\cdots,n-1$ for $w^2_{i,j}$.

The full *BCDaniso* algorithm is given as follows. We here eliminate the iteration supscript k and use one w to describe the algorithm.

Algorithm *BCDaniso*

Step 0. ***Initialization.*** Pick initial feasible w. Set the boundary points $w^1_{0,j}$, $w^1_{n,j}$, $w^2_{i,0}$ and $w^2_{i,n}$ to be 0 for $i, j = 1, \cdots n-1$.

Step 1. ***Updating.*** For $i = 1, \cdots, n-1$ and $j = 1, \cdots, n$, set

$$\begin{aligned} v &= \frac{1}{2}(w^1_{i-1,j} + w^1_{i+1,j} - w^2_{i,j} + w^2_{i,j-1} + w^2_{i+1,j} - w^2_{i+1,j-1} + z_{i+1,j} - z_{i+1,j}) \\ w^1_{i,j} &= \min\big\{\max\{-1, v\},\, 1\big\}. \end{aligned} \tag{5.21}$$

For $i = 1, \cdots, n$ and $j = 1, \cdots, n-1$, set

$$\begin{aligned} v &= \frac{1}{2}(w^2_{i,j-1} + w^2_{i,j+1} + w^1_{i,j+1} - w^1_{i,j} + w^1_{i-1,j} - w^1_{i-1,j+1} + z_{i,j+1} - z_{i,j}) \\ w^2_{i,j} &= \min\big\{\max\{-1, v\},\, 1\big\}. \end{aligned} \tag{5.22}$$

Step 2. Terminate if w satisfies the stopping criterion;

otherwise return to step 1.

5.4 Convergence Theory

The convergence of the block coordinate descent method follows directly from [44] or sec 2.7 of [7]. We quote the relevant results from [44] as follows.

Definition 2. *Following the convention in (5.1) and let $i \in \{1, 2, \cdots, n\}$; we say that f is strictly quasiconvex with respect to $x_i \in X_i$ on X if for every $x \in X$ and $y_i \in X_i$ with $y_i \neq x_i$ we have*

$$f(x_1, \cdots, tx_i + (1-t)y_i, \cdots, x_n) < \max\{f(x),\, f(x_1, \cdots, y_i, \cdots, x_n)\}$$

for all $t \in (0, 1)$.

Proposition 2. *Suppose that the function f is strictly quasiconvex with respect to x_i on X for each $i = 1, 2, \cdots, n-2$ in the sense of Definition 2 and that the sequence $\{x^k\}$ generated by the BCD method (5.2) has limit points. Then every limit point $\bar{x}$ of $\{x^k\}$ is a critical point of Problem (5.1).*

Theorem 4. *The BCD and BCDaniso algorithms described in the above sections are global convergent.*

Proof. The proof follows directly from Proposition 2. First of all, it is easy to check that the energy function of the dual ROF model is strictly quasi-convex with respect to each corresponding 'coordinate' for both the isotropic and anisotropic case. Hence, Proposition 2 applies. Secondly, X is compact for both cases. Hence every sequence of iterates $\{w^{(k)}\}$ has a limit point $\bar{w}$. It follows from Proposition 2 that the limit point $\bar{w}$ is a stationary point and hence a global minimizer of the given convex problem. Finally, by monotonicity of the algorithm: $F(w^{(k+1)}) \leq F(w^{(k)})$, we have that $F(w^{(k)}) \downarrow F^* =$ global minimum energy. □

5.5 Numerical Experiments

We report on computational experiments for three test problems in image denoising. All the programs are run in an IBM T60 Notebook PC with 1.83 GHz Intel Core Duo CPU and 1G RAM. All methods are coded in MATLAB. The original clean images and the input noisy images are shown in Figures 5.2 and 5.3, respectively. The sizes of the discretizations for the three test problems are 128×128, 256×256, and 512×512, respectively. The noisy images are generated by adding Gaussian noise to the clean images using the MATLAB function `imnoise`, with variance parameter set to 0.01. The fidelity parameter λ is taken to be 0.045 for the isotropic ROF model and 0.05 for the anisotropic ROF model throughout the

experiments. This parameter is inversely related to the noise level σ and usually needs to be tuned for each individual image to get an optimal visual result. For the BCD algorithm for isotropic ROF model, the subproblem at each coordinate block are solved by the approach described in section 5.2. The inner Newton' method to compute λ^* at each coordinate block is stopped when $|f'(\lambda)| < 10^{-12}$.

We tested the following algorithms:

- Chambolle's semi-implicit gradient descent method [14] for isotropic ROF model;
- BCD algorithm for isotropic ROF model as described in section 5.2;
- Chambolle's semi-implicit gradient descent method [14] for anisotropic ROF model;
- BCD algorithm for anisotropic ROF model as described in section 5.3.

Figure 5.4 shows the denoised images obtained by different models: the isotropic ROF model and the anisotropic ROF model.

Tables 6.1, 6.2, and 6.3 report number of iterations and average CPU times over ten runs, where each run adds a different random noise vector to the true image. In all codes, we used the starting point $w^{(0)} = 0$ in each algorithm and the relative duality gap stopping criterion. We vary the threshold TOL from 10^{-2} to 10^{-4}, producing results of increasingly high accuracy as TOL is decreased.

The tables show that on all problems, the proposed Block Coordinate Descent algorithms are competitive to Chambolle's method, and that they are consistently faster in most situations. The tables also show that the advantage of BCD algorithm over Chambolle's algorithm is more profound for anisotropic TV due to the simpler constraints.

Figure 5.2: The original clean images for our test problems. Left: 128×128 "shape"; middle: 256×256 "cameraman"; right: 512×512 "Barbara".

Figure 5.3: The input noisy images for our test problems. Gaussian noise is added using MATLAB function `imnoise` with variance 0.01.

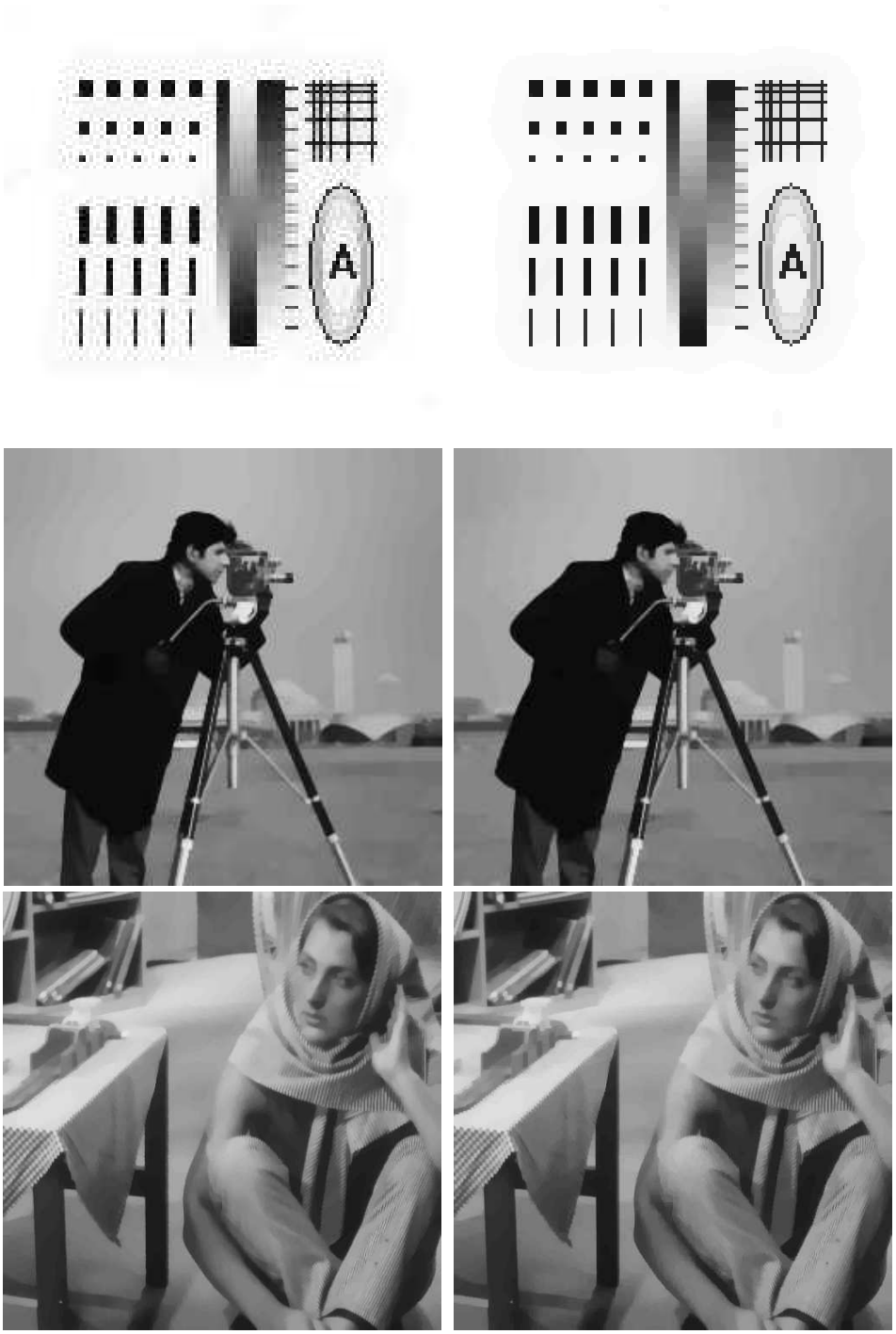

Figure 5.4: The denoised images with TV terms. left column: isotropic TV, right column: anisotropic TV.

Table 5.1: Iterations & CPU costs, isotropic TV, problem 1.

	$\text{TOL} = 10^{-2}$		$\text{TOL} = 10^{-3}$		$\text{TOL} = 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	35	0.23	323	2.19	1779	11.7
BCD	11	0.22	81	1.83	403	9.50

Table 5.2: Iterations & CPU costs, isotropic TV, problem 2.

	$\text{TOL} = 10^{-2}$		$\text{TOL} = 10^{-3}$		$\text{TOL} = 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	47	2.02	276	11.0	1285	50.2
BCD	14	1.59	66	7.47	278	32.3

Table 5.3: Iterations & CPU costs, isotropic TV, problem 3.

	$\text{TOL} = 10^{-2}$		$\text{TOL} = 10^{-3}$		$\text{TOL} = 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	44	11.1	202	49.6	814	200
BCD	12	6.38	45	24.3	166	92.3

Table 5.4: Iterations & CPU costs, anisotropic TV, problem 1.

	TOL $= 10^{-2}$		TOL $= 10^{-3}$		TOL $= 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	48	0.39	430	4.41	2042	20.6
BCD	7	0.06	49	0.39	247	1.97

Table 5.5: Iterations & CPU costs, anisotropic TV, problem 2.

	TOL $= 10^{-2}$		TOL $= 10^{-3}$		TOL $= 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	63	2.70	345	15.1	1279	54.0
BCD	8	0.38	41	1.75	150	6.59

Table 5.6: Iterations & CPU costs, anisotropic TV, problem 3.

	TOL $= 10^{-2}$		TOL $= 10^{-3}$		TOL $= 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
Chambolle	61	12.8	252	51.2	805	164
BCD	8	1.70	32	6.67	99	19.6

CHAPTER 6

Multilevel Optimizations

As we have seen in section 2.1.6, various multigrid (multilevel) methods have been developed, trying to solve the total variation denoising problem efficiently with scalable computational complexity. Most existing multilevel methods are based on the primal formulation (1.4) and/or its associated Euler-Lagrange equation. The non-differentiability of the primal energy usually makes it difficult to develop efficient multigrid algorithms. In most of the times, finding a good smoother for the primal problem is already challenging enough.

In this chapter, we shall derive the multilevel optimization algorithms based on the dual formulation of the ROF model (1.12). We shall see that in this case the main challenge in developing efficient multilevel methods is the nonlinear constraints. In other words, we need to find a correct way to interpolate the constraints in coarser level such that the coarser level correction will not violate the finer level constraints.

6.1 Multilevel Optimization for 1D Dual TV Model

As a starting point, we shall describe the multilevel optimization algorithm applied on dual formulation of the one dimensional total variation based signal

denoising model, which is ($z = \lambda f$) :

$$\begin{aligned} \min \quad & \phi(w) = \|\nabla \cdot w + z\|^2 \\ \text{subject to} \quad & L_j \leq w_j \leq R_j \quad \text{for } j = 1, 2, \cdots n, \end{aligned} \tag{6.1}$$

where $L_j = -1,\ R_j = 1$ and the one dimensional divergence operator $\nabla\cdot$ is

$$(\nabla \cdot w)_j = \begin{cases} w_1, & \text{if } i = 1 \\ w_j - w_{j-1} & \text{if } 2 \leq j \leq n \\ -w_n & \text{if } j = n+1 \end{cases} \tag{6.2}$$

We shall now apply multilevel technique to solve the above problem (6.1). Let k be the level of our multilevel algorithm with $k = 1$ being the finest level and $k = log_2^n + 1$ being the coarsest level. At level k, the uniform block size is $b = 2^{k-1}$ and number of blocks is $\tau_k = n/b$.

Suppose w is an intermediate solution (at the finest level) and its modification at the next coarser level is given by $\delta w = P_m^n\, c = [c_1,\ c_1,\ c_2,\ c_2, \cdots\cdots, c_m,\ c_m]$, where $m = \frac{n}{2}$, $c = [c_1, c_2, \cdots, c_m]$ is the unknowns on the coarser level and P is the (piecewise constant) prolongation operator from coarser level to finer level.

The objective function with the coarser level correction is then given as

$$
\begin{aligned}
& \phi(w + Pc) \\
= \quad & \|\nabla \cdot (w + Pc) - z\|^2 \\
= \quad & (w_1 + c_1 + z_1)^2 + [(w_2 + c_1) - (w_1 + c_1) + z_2]^2 \\
& + [(w_3 + c_2) - (w_2 + c_1) + z_3]^2 + [(w_4 + c_2) - (w_3 + c_2) + z_4]^2 \\
& + \cdots\cdots\cdots\cdots \\
& + [(w_{n-1} + c_m) - (w_{n-2} + c_{m-1})]^2 + [(w_n + c_m) - (w_{n-1} + c_m)]^2 \\
& + [-(w_n + c_{\tau_k}) + z_{n+1}]^2 \\
= \quad & (c_1 + w_1 + z_1)^2 \\
& + (c_2 - c_1 + w_3 - w_2 + z_3)^2 \\
& + \cdots\cdots \\
& + (c_m - c_{m-1} + w_{n-1} - w_{n-2} + z_{n-1})^2 \\
& + (-c_m - w_n + z_{n+1})^2 \\
& + \text{other terms independent of c} \\
= \quad & \|\nabla \cdot c + \tilde{z}\|^2 + \text{other terms independent of c,}
\end{aligned}
$$

where $\tilde{z}$ is given by

$$
\begin{aligned}
\tilde{z}_1 &= w_1 + z_1 \\
\tilde{z}_{j+1} &= w_{2j+1} - w_{2j} + z_{2j+1} \qquad \text{for } j = 1, \cdots, m-1 \\
\tilde{z}_{m+1} &= -w_n + z_{n+1},
\end{aligned}
$$

which is equivalent to $\tilde{z} = R\,(\nabla \cdot w + z)$. Here R is the restriction operator from finer level to coarser level, i.e.,

$$R\,(x_1,\, x_2,\, x_3,\, \cdots, x_{2l+1}) = x(1:2:2l+1) = (x_1,\, x_3, x_5\, \cdots, x_{2l+1}).$$

The finer level constraints $-1 \leq (w + Pc)_j \leq 1$ can be expressed as

$$\begin{array}{rcccl} L_{2j-1} & \leq & c_j + w_{2j-1} & \leq & R_{2j-1} \\ L_{2j} & \leq & c_j + w_{2j} & \leq & R_{2j} \end{array} \qquad \text{for } j = 1, \cdots, m.$$

The above constraints is equivalent to the following coarser level constraints

$$\tilde{L}_j \leq c_j \leq \tilde{R}_j \qquad \text{for} \quad j = 1, \cdots, m, \tag{6.3}$$

where

$$\begin{array}{rcl} \tilde{L}_j & = & \max(L_{2j-1} - w_{2j-1},\, L_{2j} - w_{2j}) \\ \tilde{R}_j & = & \min(R_{2j-1} - w_{2j-1},\, R_{2j} - w_{2j}). \end{array} \tag{6.4}$$

Put the above information all together, the problem of solving for the coarser level correction is the following minimization:

$$\begin{array}{rl} \min & \tilde{\phi}(c) = \|\nabla \cdot c + \tilde{z}\|^2 \\ \text{subject to} & \tilde{L}_j \leq c_j \leq \tilde{R}_j \qquad \text{for} \quad j = 1, \cdots, m. \end{array} \tag{6.5}$$

Now, problem (6.5) has the exact same form as the original problem (6.1) except it is on a coarser level. Hence we can apply the same smoother to get an approximate solution of (6.5) and then pass to the even coarser grid for corrections. We can adopt the same technique recursively until we get to the coarsest level and then we add the correction back level by level until reach the finest level. Each cycle of getting from finest level to coarsest level and then back to finest level is called a *V-Cycle* in multigird jargon. The complexity for each *V-Cycle* is $O(n)$.

The full multilevel algorithm *ML1D* using the recursive *V-Cycle* is shown below. Here we use

$$w = \text{Smoother}(w^0,\, z,\, L,\, R,\, iter)$$

to denote an approximate intermediate solution of (6.1) that we obtained by apply a specified smoother(relaxation) *iter* times with initial guess w^0. The actual level of the problem is implied by the context.

We then use the recursive function

$$w = \text{Vcycle}(w^0,\, z,\, L,\, R)$$

to denote outcome of applying one *V-cycle* to solve the original problem (6.1).

The recursive function can be defined as follows:

function $w = \text{Vcycle}(w^0,\, z,\, L,\, R)$
$n = \text{length}(w^0)$;
if $n == 1$
$w_1 = \frac{1}{2}(z_2 - z_1)$;
return;
else
$w = \text{Smoother}(w^0,\, z,\, L,\, R,\, iter1)$;
$m = \frac{n}{2}$;
for $j = 1 : m$
$\tilde{L}_j = \max(L_{2j-1} - w_{2j-1},\, L_{2j} - w_{2j})$;
$\tilde{R}_j = \min(R_{2j-1} - w_{2j-1},\, R_{2j} - w_{2j})$;
end
$\tilde{z} = R(\nabla \cdot w + z)$;
$c = \text{Vcycle}(0,\, \tilde{z},\, \tilde{L},\, \tilde{R})$;
$w = w + Pc$;
$w = \text{Smoother}(w,\, z,\, L,\, R,\, iter2)$;
return;
end

The full multilevel algorithm *ML1D* is shown as below.

Algorithm *ML1D*

Step 0. ***Initialization.*** Set $z = \lambda f,\ L_j = -1, R_j = 1,$ for $j = 1, \cdots n.$

Pick initial feasible w^0. Set $k = 0$.

Step 1. ***Updating Using V-Cycle.***

$$w^{k+1} = \text{Vcycle}(w^k,\ z,\ L,\ R).$$

Step 2. Terminate if w^{k+1} satisfies the stopping criterion;

otherwise set $k \leftarrow k + 1$ and return to step 1.

6.2 Multilevel Optimization for 2D Dual TV Model

We now generalize the idea of multilevel optimization to 2D TV models.

The discrete dual formulation for TV restoration is

$$\begin{aligned} &\min\ \phi(w) = \|\nabla \cdot w + z\|^2 \\ \text{subject to}\quad &\|w_{i,j}\| \le 1 \text{ for } i, j = 1 \cdots, n, \end{aligned} \tag{6.6}$$

where $z = \lambda f,\ w = (w_{i,j}) \in \mathbb{R}^{2\times n\times n}$ with each component $w_{i,j} \in \mathbb{R}^2$, i.e.

$$w_{i,j} = \begin{bmatrix} w^1_{i,j} \\ w^2_{i,j} \end{bmatrix}, \quad 1 \le i, j \le n. \tag{6.7}$$

The discrete divergence operator $\nabla\cdot$ is defined in (1.24) which we rewrite it here:

$$(\nabla \cdot w)_{i,j} = \begin{cases} w^1_{i,j} - w^1_{i-1,j} & \text{if } 1 < i < n \\ w^1_{i,j} & \text{if } i = 1 \\ -w^1_{i-1,j} & \text{if } i = n \end{cases} + \begin{cases} w^2_{i,j} - w^2_{i,j-1} & \text{if } 1 < j < n \\ w^2_{i,j} & \text{if } j = 1 \\ -w^2_{i,j-1} & \text{if } j = n. \end{cases} \tag{6.8}$$

The multilevel formulation for the 2D problem is more complicated and tedious than the 1D problem. First let us fixed some conventions. We set the size of matrices w^1, w^2 and z to be $n \times (n+1)$, $(n+1) \times n$ and $(n+1) \times (n+1)$ respectively with n being a power of 2. Let $m = \frac{n}{2}$.

We follow the same setup in the 1D case and let w^1, w^2 be the intermediate solution on the finest level. We then use c^1, c^2 to denote the corrections in the next coarser level. We again use the piecewise constant interpolation to map P the coarser level correction to the finer level, i.e.

$$C = \begin{pmatrix} c_{1,1} & c_{1,2} & \dots \\ c_{2,1} & c_{2,2} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \quad \Rightarrow \quad PC = \begin{pmatrix} c_{1,1} & c_{1,1} & c_{1,2} & c_{1,2} & \dots \\ c_{1,1} & c_{1,1} & c_{1,2} & c_{1,2} & \dots \\ c_{2,1} & c_{2,1} & c_{2,2} & c_{2,2} & \dots \\ c_{2,1} & c_{2,1} & c_{2,2} & c_{2,2} & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

The objective function with the coarser level correction is

$$\phi(w + Pc) = \|\nabla \cdot (w + Pc) + z\|^2 \tag{6.9}$$

To rearrange the above objective function, we realized that the indices $(2i, 2j), (2i-1, 2j), (2i, 2j-1)$ and $(2i-1, 2j-1)$ should be treated differently. If we work our

every detail, we will obtain the following formulation for energy function (6.9):

$$\begin{aligned}\phi(w+Pc) &= \|\nabla\cdot c+\tilde{z}\|^2 \\ &+ \sum_{j=1}^{m}\|\nabla\cdot c^1(:,j)F(:,j)\|^2 \\ &+ \sum_{i=1}^{m}\|\nabla\cdot c^2(i,:)+G(i,:)\|^2 \\ &+ \text{ other terms independent of } c. \end{aligned} \tag{6.10}$$

In (6.10) and later equations, We use the same notation $\nabla\cdot$ for both the 2D divergence operator defined in (6.8) and 1D divergence operator defined in (6.2). It is easily to figure out their real meaning in the context. Here, the first is for 2D and the rest are for 1D.

In (6.10), c_1 and G are $m\times(m+1)$ matrices, c_2 and F are $(m+1)\times m$ matrices and $\tilde{z}$ has dimension $(m+1)\times(m+1)$. F, G and $\tilde{z}$ are given as follows.

$$\begin{aligned} S &= \nabla\cdot w+z \\ \tilde{z} &= S(1:2:n+1,1:2:n+1) \\ F &= S(1:2:n+1,2:2:n) \\ G &= S(2:2:n,1:2:n+1) \end{aligned} \tag{6.11}$$

Motivated by equation (6.10), we rewrite the original dual ROF model (6.6) to facilitate the recursive multigrid V-cycle algorithm as follows.

$$\begin{aligned}\min\ \phi(w) &= \|\nabla\cdot w+z\|^2 \\ &+\gamma\sum_{j=1}^{n}\|\nabla\cdot w^1(:,j)+X(:,j)\|^2 \\ &+\gamma\sum_{i=1}^{n}\|\nabla\cdot w^2(i,:)+Y(i,:)\|^2. \\ \text{subject to}\quad &\|w_{i,j}\|\le R_{i,j}\quad \text{for } i,j=1\cdots,n, \end{aligned} \tag{6.12}$$

where initially in the finest level $\gamma=0$, X, Y are $(n+1)\times n$ and $n\times(n+1)$ zero matrices and R is a $(n+1)\times(n+1)$ matrix with all ones.

Then the objective function with coarser level corrections can be written as

$$\begin{aligned}
\phi(w+Pc) &= \|\nabla\cdot c+\tilde{z}\|^2 \\
&\quad + \sum_{j=1}^{m}\|\nabla\cdot c_1(:,j)+F(:,j)\|^2 \\
&\quad + \sum_{i=1}^{m}\|\nabla\cdot c_2(i,:)+G(i,:)\|^2 \\
&\quad + 2\gamma\sum_{j=1}^{m}\|\nabla\cdot c_1(:,j)+Q(:,j)\|^2 \\
&\quad + 2\gamma\sum_{i=1}^{m}\|\nabla\cdot c_2(i,:)+T(i,:)\|^2 \\
&\quad + \text{ other terms independent of } c. \qquad (6.13)
\end{aligned}$$

or

$$\begin{aligned}
\phi(w+Pc) &= \|\nabla\cdot c+\tilde{z}\|^2 \\
&\quad + \tilde{\gamma}\sum_{j=1}^{m}\|\nabla\cdot c_1(:,j)+\tilde{X}(:,j)\|^2 \\
&\quad + \tilde{\gamma}\sum_{i=1}^{m}\|\nabla\cdot c_2(i,:)+\tilde{Y}(i,:)\|^2 \\
&\quad + \text{ other terms independent of } c. \qquad (6.14)
\end{aligned}$$

where F, G and $\tilde{z}$ are given in (6.11), and $Q, T, \tilde{X}, \tilde{Y}$ and $\tilde{\gamma}$ are given as follows:

$$\begin{aligned}
\tilde{\gamma} &= 1+2\gamma \\
Q &= \tfrac{1}{2}\big[U(1:2:n+1,1:2:n-1)+U(1:2:n+1,2:2:n)\big] \\
T &= \tfrac{1}{2}\big[V(1:2:n-1,1:2:n+1)+V(2:2:n,1:2:n+1)\big] \qquad (6.15) \\
\tilde{X} &= (F+2\gamma Q)/\tilde{\gamma} \\
\tilde{Y} &= (G+2\gamma T)/\tilde{\gamma}
\end{aligned}$$

The objective function (6.14) is exactly in the same form with the original energy in (6.12) except it is on the coarser level.

The constraints on the coarser level variable c are given as

$$\begin{aligned} \|c_{i,j} + w_{2i-1,2j-1}\| &\leq R_{2i-1,2j-1} \\ \|c_{i,j} + w_{2i-1,2j}\| &\leq R_{2i-1,2j} \\ \|c_{i,j} + w_{2i,2j-1}\| &\leq R_{2i,2j-1} \\ \|c_{i,j} + w_{2i,2j}\| &\leq R_{2i,2j} \end{aligned} \qquad \text{for} \quad 1 \leq i,j \leq m \tag{6.16}$$

Now we shall tried to simplify the above constraints on the coarser level so that we can reduce the overall complexity of the problem by a factor of 2×2. Unfortunately, there is no simple way to achieve that as we have done for the 1D problem in (6.3). It is easily to see from 6.16 that the feasible set for each $c_{i,j}$ is the intersection of 4 unit disks, which in general cannot be expressed using a single equation.

There are several plausible approaches for this problem. For example, in stead of trying to simply the constraints (6.16) to an **equivalent** single constraint, i.e. the old constraints (6.16) are satisfied iff. the new constraint is satisfied, we can relax this equivalence to a sufficient but not necessary new constraint as follows:

$$\|c_{i,j}\| \leq \tilde{R}_{i,j} \tag{6.17}$$

$$\text{where} \qquad \tilde{R}_{i,j} = \min\left\{R_{k,l} - \|w_{k,l}\| : 2i-1 \leq k \leq 2i,\ 2j-1 \leq l \leq 2j\right\}.$$

If we choose the new constraints on the coarser level as above, then the optimization problem of finding coarser level correction is exactly in the same form as the original problem in the finest level. We can apply some smoother (e.g. BCD algorithm in Chapter 5) several iterations to solve an intermediate solution on the coarser level and pass it to the next even coarser level recursively. However, the problem is that the constraints (6.17) may be too restrictive for the algorithm to make any useful corrections on the coarser level. An extreme case is when $\tilde{R}_{i,j} = 0$ and no correction will be allowed on that block.

We notice that the anisotropic TV model does not have this problem and therefore is more suitable for applying our multilevel optimization algorithm in this sense. The dual formulation of the anisotropic ROF model has the exact same objective function with the isotropic ROF model (6.12) but its constraints are simple bound constraints. The dual anisotropic ROF model is shown as follows

$$\begin{aligned}
\min\ \phi(w) \quad = \quad & \|\nabla \cdot w + z\|^2 \\
& + \gamma \sum_{j=1}^{n} \|\nabla \cdot w^1(:,j) + X(:,j)\|^2 \\
& + \gamma \sum_{i=1}^{n} \|\nabla \cdot w^2(i,:) + Y(i,:)\|^2. \qquad (6.18) \\
\text{subject to} \quad & L^1_{i,j} \le w^1_{i,j} \le R^1_{i,j} \quad \text{for } 1 \le i \le n,\ 1 \le j \le n+1 \\
& L^2_{i,j} \le w^2_{i,j} \le R^2_{i,j} \quad \text{for } 1 \le i \le n+1,\ 1 \le j \le n
\end{aligned}$$

where X, Y and γ are the same as in (6.12). L^1, L^2 are matrices with all entries being -1 and R^1, R^2 are matrices with all entries being 1.

From the above analysis, it is not hard to obtain that the problem of finding coarser level correction c is the following minimization problem.

$$\begin{aligned}
\min \quad & \|\nabla \cdot c + \tilde{z}\|^2 \\
& + \tilde{\gamma} \sum_{j=1}^{m} \|\nabla \cdot c_1(:,j) + \tilde{X}(:,j)\|^2 \\
& + \tilde{\gamma} \sum_{i=1}^{m} \|\nabla \cdot c_2(i,:) + \tilde{Y}(i,:)\|^2 \qquad (6.19) \\
\text{subject to} \quad & \tilde{L}^1_{i,j} \le c^1_{i,j} \le \tilde{R}^1_{i,j} \quad \text{for } 1 \le i \le m,\ 1 \le j \le m+1 \\
& \tilde{L}^2_{i,j} \le c^2_{i,j} \le \tilde{R}^2_{i,j} \quad \text{for } 1 \le i \le m+1,\ 1 \le j \le m
\end{aligned}$$

where $\tilde{X}, \tilde{Y}, \tilde{\gamma}$ are the same as in (6.15) and $\tilde{L}, \tilde{R}$ are given as follows:

$$\begin{aligned}
\tilde{L}^1_{i,j} &= \max\left\{L^1_{k,l} - w^1_{k,l} \ : \ 2i-1 \le k \le 2i, \ 2j-1 \le l \le 2j\right\};\\
\tilde{L}^2_{i,j} &= \max\left\{l^2_{k,l} - w^2_{k,l} \ : \ 2i-1 \le k \le 2i, \ 2j-1 \le l \le 2j\right\};\\
\tilde{R}^1_{i,j} &= \min\left\{R^1_{k,l} - w^1_{k,l} \ : \ 2i-1 \le k \le 2i, \ 2j-1 \le l \le 2j\right\};\\
\tilde{R}^2_{i,j} &= \min\left\{R^2_{k,l} - w^2_{k,l} \ : \ 2i-1 \le k \le 2i, \ 2j-1 \le l \le 2j\right\};
\end{aligned}$$

Now coarser level correction problem (6.19) has the exact same formulation with the original problem (6.18). So we can apply the same strategy recursively and obtain the multilevel V-cycle algorithm as in the 1D problem.

6.3 Numerical Experiments and Comments

We test our multilevel (ML) optimization algorithm on the anisotropic ROF model using the block coordinate descent (BCD) method developed in Chapter 5 as the smoother. We also compared the performance of the ML algorithm and the uni-level BCD algorithm. The test problems and parameters are exactly the same as those in Chapter 5. At each level of the V-cycle inside the ML algorithm, we apply 3 relaxations before pass to the coarser level and apply 2 more relaxations after adding the coarser level corrections.

Tables 6.1, 6.2, and 6.3 report number of iterations and average CPU times over ten runs, where each run adds a different random noise vector to the true image. In all codes, we used the starting point $x^{(0)} = 0$ in each algorithm and the relative duality gap stopping criterion. We vary the threshold TOL from 10^{-2} to 10^{-4}, producing results of increasingly high accuracy as TOL is decreased.

These tables show that the improvement over the BCD algorithm by adopting the multilevel optimization framework is not significant. This might due to

Table 6.1: Iterations & CPU costs, anisotropic TV, problem 1.

	$\text{TOL} = 10^{-2}$		$\text{TOL} = 10^{-3}$		$\text{TOL} = 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
ML BCD	2	0.06	11	0.30	50	1.42
BCD	7	0.06	49	0.39	247	1.97

Table 6.2: Iterations & CPU costs, anisotropic TV, problem 2.

	$\text{TOL} = 10^{-2}$		$\text{TOL} = 10^{-3}$		$\text{TOL} = 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
ML BCD	2	0.31	8	1.22	29	4.13
BCD	8	0.38	41	1.75	150	6.59

Table 6.3: Iterations & CPU costs, anisotropic TV, problem 3.

	$\text{TOL} = 10^{-2}$		$\text{TOL} = 10^{-3}$		$\text{TOL} = 10^{-4}$	
Algorithms	Iter	CPU (s)	Iter	CPU (s)	Iter	CPU (s)
ML BCD	2	1.47	6	4.35	19	13.4
BCD	8	1.70	32	6.67	99	19.6

the characteristics of this particular problem. In particular, the solution and residuals are not smooth with any available relaxation algorithms because of the constraints. For future work, some more intelligent coarsening and interpolation schemes must be adopted to fully exploit the advantage of multigrid algorithms.

CHAPTER 7

Connections and Improvements of Some Existing Methods

7.1 Connections between CGM and SOCP

In Chapter 2, we give a survey of some existing algorithms for solving the total variation based image restoration models, which includes the CGM method and SOCP approach. We also point out that SOCP and CGM bear many similarities. Both methods apply Newton's method to some primal-dual system to compute the update at each iteration. In this section, we shall investigate the connections between these two methods in more details.

Preliminaries of SOCP

We remind the reader that we follow the MATLAB convention of using " , " for adjoining vectors and matrices in a row, and " ; " for adjoining them in a column. Thus, for any vectors x, y and z, the following are synonymous:

$$(x; y; z) = (x^T,\, y^T,\, z^T)^T = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

We also use $\oplus$ denote joining matrices diagonally, that is, for two matrices A and B,

$$A \oplus B = \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}$$

Now we use the notation $\mathcal{K}^n$ to denote the Lorenz cone (ice cream cone) in $\mathbb{R}^n$, i.e.

$$\mathcal{K}^n \equiv \{x = (x_0; \bar{x}) \in \mathbb{R}^n : \|\bar{x}\| \leq x_0\}$$

Second-Order Cone Programming (SOCP) problem has the following standard form

$$\begin{array}{llll} \textbf{Primal} & & \textbf{Dual} & \\ \min & c^T x & \min & b^T y \\ \text{s. t.} & Ax = b, & \text{s. t.} & A^T y + c = z, \\ & x \in \mathcal{K}. & & z \in \mathcal{K}. \end{array} \tag{7.1}$$

where

$$\begin{array}{ll} x = (x_1; \cdots ; x_r) & \text{with } x_i \in \mathbb{R}^{n_i} \\ c = (c_1; \cdots ; c_r) & \text{with } c_i \in \mathbb{R}^{n_i} \\ z = (z_1; \cdots ; z_r) & \text{with } z_i \in \mathbb{R}^{n_i} \\ A = (A_1, \cdots , A_r) & \text{with } A_i \in \mathbb{R}^{m \times n_i} \\ y, b \in \mathbb{R}^m & \end{array}$$

and $\mathcal{K}$ is the Cartesian product of several cones:

$$\mathcal{K} = \mathcal{K}^{n_1} \times \mathcal{K}^{n_2} \times \cdots \times \mathcal{K}^{n_r}.$$

Associated with each vector $x = (x_0; \bar{x}) \in \mathbb{R}^n$ there is an arrow-shaped matrix $\mathrm{Arw}(x)$ defined as:

$$\mathrm{Arx}(x) \equiv \begin{pmatrix} x_0 & \bar{x}^T \\ \bar{x} & x_0 I \end{pmatrix}$$

Observe that $x \in \mathcal{K}$ ($x \in \operatorname{int}\mathcal{K}$) iff. $\operatorname{Arw}(x)$ is positive semidefinite (positive definite).

We also use $\operatorname{Arw}(\cdot)$ in the block sense; that is if $x = (x_1; \cdots; x_r)$ such that $x_i \in \mathbb{R}^{n_i}$ for $i = 1, \cdots, r$, then

$$\operatorname{Arw}(x) = \operatorname{Arw}(x_1) \oplus \cdots \oplus \operatorname{Arw}(x_r).$$

There is a particular algebra associated with second-order cones, the understanding of which sheds light on all aspects of the SOCP problem, from duality and complementarity properties, to conditions of non-degeneracy and ultimately to the design and analysis of interior-point algorithms. This algebra is well-known and is a special case of a so-called *Euclidean Jordan algebra*. (see for a complete study.) Here we only borrow some notations and definitions to facilitate our explanations.

For now we assume that all vectors consist of a single block $x = (x_0; \bar{x})$. For two vectors x and y define the following multiplication

$$x \circ y \equiv (x^T y; x_0\bar{y} + y_0\bar{x}) = \begin{pmatrix} x^T y \\ x_0 y_1 + y_0 x_1 \\ \vdots \\ x_0 y_n + y_0 x_n \end{pmatrix}$$

Let $e = (1; 0; \cdots 0)$. We define the inverse of x, x^{-1} by

$$y \equiv x^{-1} \quad \text{iff.} \quad x \circ y = y \circ x = e,$$

and define the determinant of x, $\det(x)$ by

$$\det(x) = x_0^2 - \|\bar{x}\|^2.$$

It is clear that

$$x^{-1} = \frac{1}{\det(x)} (x_0; -\bar{x}).$$

SOCP problems are usually solved by primal-dual path following interior-point methods. Note that for $x \in \text{int } \mathcal{K}$ the function `-ln(det(x))` is a convex barrier function for $\mathcal{K}$. if we replaced the second-order cone constraints by $x_i \in \text{int } \mathcal{K}^{n_i}$ and add the logarithm barrier term $-\mu \sum \texttt{ln}(\texttt{det}(x_i))$ to the objective function in the primal problem we get

$$\begin{aligned} (\mathbf{P}_\mu) \quad \min \quad & \sum_{i=1}^{r} c_i^T x_i - \mu \sum_{i=1}^{r} \texttt{ln}(\texttt{det}(x_i)) \\ \text{s.t.} \quad & \sum_{i=1}^{r} A_i x_i = b, \\ & x_i \in \text{int } \mathcal{K}^{n_i} \quad \text{for } i = 1, \cdots, r. \end{aligned} \tag{7.2}$$

The Karush-Kuhn-Tucker (KKT) optimality conditions for (7.2) are:

$$\begin{aligned} & \sum_{i=1}^{r} A_i x_i = b, \\ & c_i - A_i^T y - 2\mu x_i^{-1} = 0, \quad \text{for } i = 1, \cdots, r, \\ & x_i \in \text{int } \mathcal{K}^{n_i} \quad \text{for } i = 1, \cdots, r. \end{aligned} \tag{7.3}$$

If we set $z_i = c_i - A_i^T y$, the any solution of problem $\mathbf{P}_\mu$ satisfies:

$$\begin{aligned} & \sum_{i=1}^{r} A_i x_i = b, \\ & A_i^T y + z_i = c_i, \quad \text{for } i = 1, \cdots, r, \\ & x_i \circ z_i = 2\mu e, \quad \text{for } i = 1, \cdots, r, \\ & x_i, z_i \in \text{int } \mathcal{K}^{n_i} \quad \text{for } i = 1, \cdots, r. \end{aligned} \tag{7.4}$$

For every $\mu > 0$ we can show that the above system (7.4) has a unique solution (x_μ, y_μ, z_μ). The trajectory points (x_μ, y_μ, z_μ) satisfying (7.4) is defined as the

primal-dual central path or simply the *central path* associated with the SOCP problem (7.1).

Generally speaking the strategy of path-following primal-dual interior-point method can be sketched as follows. We start with a point near or on the central path. First, we apply Newton's method to the system (7.4) to get a direction $(\Delta x, \Delta y, \Delta y)$ that reduces the duality gap, and then take a step in this direction making sure that the new point is still feasible and in the interior of $\mathcal{K}$. Then we reduce μ by some factor and repeat the process. With judicious choices of initial point, step length and reduction schedule for μ we are able to show convergence in a polynomial number of iterations.

Applying Newton' method to (7.4) will result a linear system for the update direction $(\Delta x, \Delta y, \Delta y)$:

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ \mathrm{Arw}(x) & 0 & \mathrm{Arw}(x) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = \begin{pmatrix} b - Ax \\ c - A^T y - z \\ 2\mu e - x \circ z \end{pmatrix} \tag{7.5}$$

SOCP for ROF and Connections with CGM

To reform the constrained ROF model

$$\begin{aligned} \min \quad & \sum_{1 \le i,j \le n} \|(\nabla u)_{i,j}\| \\ \text{s.t.} \quad & \|u - f\|^2 \le \sigma^2 \end{aligned} \tag{7.6}$$

into a SOCP problem, the authors in [43] introduced new variables p, q, t, v as follows.

They let v to be the noise variable: $v = f - u$ and let $(p_{i,j}, q_{i,j})$ to be the

discrete ∇u by forward differentiation:

$$p_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } i < n \\ 0 & \text{if } i = n \end{cases}$$

$$q_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } j < n \\ 0 & \text{if } j = n \end{cases}$$

They then impose the second-order cone constraints $\sqrt{p_{i,j}^2 + q_{i,j}^2} \leq t_{i,j}$.

As a consequence, problem (7.6) is transformed to

$$\begin{array}{lll} \min & \displaystyle\sum_{i,j=1}^{n} t_{i,j} & \\ \text{s.t.} & u_{i,j} + v_{i,j} = f_{i,j} & \text{for } i,j = 1,\cdots,n \\ & -p_{i,j} + u_{i+1,j} - u_{i,j} = 0 & \text{for } i = 1,\cdots,n-1,\ j = 1,\cdots,n \\ & -q_{i,j} + u_{i,j+1} - u_{i,j} = 0 & \text{for } i = 1,\cdots,n,\ j = 1,\cdots,n-1 \\ & p_{n,j} = 0 & \text{for } j = 1,\cdots,n, \\ & q_{i,n} = 0 & \text{for } i = 1,\cdots,n, \\ & v_0 = \sigma & \\ & (t_{i,j};\ p_{i,j};\ q_{i,j}) \in \mathcal{K}^3 & \text{for } i,j = 1,\cdots,n, \\ & (v_0;\ v) \in \mathcal{K}^{n^2+1} & \end{array} \tag{7.7}$$

Eliminating u from formulation (7.7), we obtain the following standard form

SOCP:

$$
\begin{array}{lll}
\min & \sum_{1\le i,j\le n} t_{i,j} & \\
\text{s.t.} & p_{i,j} + v_{i+1,j} - v_{i,j} = f_{i+1,j} - f_{i,j} & \text{for } 1 \le i \le n-1,\ 1 \le j \le n \\
& q_{i,j} + v_{i,j+1} - v_{i,j} = f_{i,j+1} - f_{i,j} & \text{for } 1 \le i \le n,\ 1 \le j \le n-1 \\
& p_{n,j} = 0 & \text{for } 1 \le j \le n \\
& q_{i,n} = 0 & \text{for } 1 \le i \le n \\
& v_0 = \sigma & \\
& (t_{i,j};\ p_{i,j};\ q_{i,j}) \in \mathcal{K}^3 & \text{for } 1 \le i,j \le n \\
& (v_0; v) \in \mathcal{K}^{n^2+1} &
\end{array}
\tag{7.8}
$$

Problem 7.8 is in the standard form of a SOCP problem (7.1). To see that, we introduce the following notations and variables.

$n \times n$ **:** Image size

$N + 1$ **:** Number of cones $n^2 + 1$, $(N = n^2)$

M **:** Number of equality constraints (number of rows in $\bar{A}$), $M = 2N + 1$

k **:** Index of blocks, $k \equiv k(i,j) = (j-1)n + i$

$$
\begin{aligned}
x_k &= (t_{i,j}\,;\ p_{i,j}\,;\ q_{i,j}), \quad \text{for } k = 1, 2, \cdots, N. \\
c_k &= (1;\ 0;\ 0), \quad \text{for } k = 1, \cdots, N. \\
x_{N+1} &= (v_0;\ v) \in \mathbb{R}^{N+1}. \\
c_{N+1} &= (0;\ \cdots;\ 0) \in \mathbb{R}^{N+1}. \\
\bar{A}_k &= \begin{pmatrix} 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{M\times 3}, \text{ i.e., } \bar{A}_k(2k-1:2k,\ 2:3) = I_2, \quad \text{for } k = 1, \cdots, N. \\
\bar{A}_{N+1} &= \begin{pmatrix} 0 & A^T \\ 1 & 0 \end{pmatrix}, \text{ where } A \text{ is defined in Chapter 1 in (1.25) and (1.27).} \\
\bar{A} &= (\bar{A}_1,\ \bar{A}_2,\ \cdots,\ \bar{A}_{N+1}) \\
x &= (x_1;\ x_2;\ \cdots;\ x_{N+1}) \\
c &= (c_1;\ \cdots;\ c_{N+1}). \\
b &= (A^T f;\ \sigma). \\
\mathcal{K} &= \mathcal{K}^3 \times \cdots \times \mathcal{K}^3 \times \mathcal{K}^{N+1}.
\end{aligned}
$$

Using the above notation and variables, we can rewrite problem (7.8) into the following standard form of SOCP:

$$
\begin{aligned}
\min \quad & c^T x \\
\text{subject to} \quad & \bar{A}x = b, \\
& x \in \mathcal{K}.
\end{aligned}
$$

We now follow the general procedure of the primal-dual path following interior-

point methods to solve 7.8. We shall show the KKT system of (7.8) that we solve at each iteration of the interior-point method are equivalent to the CGM system developed in [25].

First, the constraints that $\|v\| \leq \sigma$ is always active at optimality in practice and therefor we can replace it by equality constraints $\|v\| = \sigma$ in (7.8). Replacing the second-order cone constraints $p_{i,j}^2 + q_{i,j}^2 \leq t_{i,j}^2$ by $p_{i,j}^2 + q_{i,j}^2 < t_{i,j}^2$ and adding the logarithm barrier term $-\frac{\mu}{2} \sum \mathtt{ln}(t_{i,j}^2 - (p_{i,j}^2 + q_{i,j}^2))$ to the objective function in the problem (7.8) we get a problem $(\mathbf{P}_\mu)$ as in (7.2):

$$\min \sum_{1 \leq i,j \leq n} t_{i,j} - \mu \sum_{1 \leq i,j \leq n} \ln(t_{i,j}^2 - p_{i,j}^2 - q_{i,j}^2) \tag{7.9a}$$

$$\text{s.t.} \quad p_{i,j} + v_{i+1,j} - v_{i,j} = f_{i+1,j} - f_{i,j} \quad \text{for } 1 \leq i \leq n-1,\ 1 \leq j \leq n \tag{7.9b}$$

$$q_{i,j} + v_{i,j+1} - v_{i,j} = f_{i,j+1} - f_{i,j} \quad \text{for } 1 \leq i \leq n,\ 1 \leq j \leq n-1 \tag{7.9c}$$

$$\sum v_{i,j}^2 - \sigma^2 = 0 \tag{7.9d}$$

$$(t_{i,j};\ p_{i,j};\ q_{i,j}) \in \operatorname{int} \mathcal{K}^3 \qquad \text{for } 1 \leq i,j \leq n \tag{7.9e}$$

The KKT system for problem (7.9) is

$$1 - \frac{2\mu t_{i,j}}{t_{i,j}^2 - (p_{i,j}^2 + q_{i,j}^2)} = 0 \tag{7.10a}$$

$$-\alpha_{i,j} + \frac{2\mu p_{i,j}}{t_{i,j}^2 - (p_{i,j}^2 + q_{i,j}^2)} = 0 \tag{7.10b}$$

$$-\gamma_{i,j} + \frac{2\mu q_{i,j}}{t_{i,j}^2 - (p_{i,j}^2 + q_{i,j}^2)} = 0 \tag{7.10c}$$

$$(\alpha_{i,j} - \alpha_{i-1,j}) + (\gamma_{i,j} - \gamma_{i,j-1}) + \lambda v_{i,j} = 0 \tag{7.10d}$$

$$p_{i,j} + v_{i+1,j} - v_{i,j} - f_{i+1,j} + f_{i,j} = 0 \tag{7.10e}$$

$$q_{i,j} + v_{i,j+1} - v_{i,j} - f_{i,j+1} + f_{i,j} = 0 \tag{7.10f}$$

$$\sum v_{i,j}^2 - \sigma^2 = 0 \tag{7.10g}$$

where α, γ and λ are Lagrange multipliers for the constraints (7.9b), (7.9c) and

(7.9d) respectively. Note in both (7.9) and (7.10), we omit the special case for the points on the boundary of the image domain ($i = n$ or $j = n$) only for simplicity.

Equation (7.10a) is same as

$$t_{i,j}^2 - 2\mu t_{i,j} - (p_{i,j}^2 + q_{i,j}^2) = 0,$$

which is equivalent to

$$t_{i,j} = \sqrt{p_{i,j}^2 + q_{i,j}^2 + \mu^2} + \mu$$

since $t_{i,j} \geq 0$.

Hence, The first three equations in the above KKT system, equations (7.10a-7.10c) are equivalent to

$$\begin{aligned} t_{i,j} &= \sqrt{p_{i,j}^2 + q_{i,j}^2 + \mu^2} + \mu \\ t_{i,j}\,\alpha_{i,j} - p_{i,j} &= 0 \\ t_{i,j}\,\gamma_{i,j} - q_{i,j} &= 0 \end{aligned}$$

If we eliminate v by $v = f - u$, replace $(p_{i,j},\, q_{i,j})$ by $(\nabla u)_{i,j}$ and introduce w as $w_{i,j} = (\alpha_{i,j}, \gamma_{i,j})$, equations (7.10a-7.10c) are equivalent to

$$\sqrt{|(\nabla u)_{i,j}|^2 + \beta}\; w_{i,j} - (\nabla u)_{i,j} = 0,$$

and equation (7.10d) simply becomes

$$(\nabla \cdot w)_{i,j} - \lambda(u_{i,j} - f_{i,j}) = 0.$$

Hence system (7.10) can be reformed to the following equivalent system:

$$\begin{aligned} \left(\sqrt{|(\nabla u)_{i,j}|^2 + \mu^2} + \mu\right) w_{i,j} - (\nabla u)_{i,j} &= 0 \\ (\nabla \cdot w)_{i,j} - \lambda(u_{i,j} - f_{i,j}) &= 0 \\ \sum |u_{i,j} - f_{i,j}|^2 - \sigma^2 &= 0 \end{aligned} \tag{7.11}$$

From [25], we have the CGM system for the constraint ROF model is

$$\begin{aligned} \sqrt{|(\nabla u)_{i,j}|^2 + \beta}\; w_{i,j} - (\nabla u)_{i,j} &= 0 \\ (\nabla \cdot w)_{i,j} - \lambda(u_{i,j} - f_{i,j}) &= 0 \\ \sum |u_{i,j} - f_{i,j}|^2 - \sigma^2 &= 0 \end{aligned} \tag{7.12}$$

Note system (7.11) is exactly the same as (7.12) except $\sqrt{|\nabla u|^2 + \mu^2} + \mu$ is replaced by $\sqrt{|\nabla u|^2 + \beta}$. In other words, the SOCP problem and CGM method actually solve the similar system at each iteration. The difference is the barrier parameter μ in SOCP KKT system decreases to 0 following some reduction schedule while the smoothing parameter β in CGM is a fixed small number.

7.2 An improvement of CGM Method

The primal-dual optimality condition for the unconstrained ROF model is

$$\|A_i^T y\|\; x_i - A_i^T y = 0, \qquad i = 1, \ldots, N. \qquad \text{(feaibility condition)} \tag{7.13a}$$

$$Ax + \lambda y - \lambda y_0 = 0. \qquad \text{(complimentary condition)} \tag{7.13b}$$

In CGM method [25], the feasibility condition is modified with a parameter $\beta > 0$

$$\sqrt{\|A_i^T y\|^2 + \beta^2}\; x_i - A_i^T y = 0, \quad i = 1, \ldots, N,$$

so that the system become smooth and the point x will be enforced to stay inside the interior of the feasible set

$$X = \big\{ x = (x_1; \cdots; x_N) \;:\; \|x_i\| \le 1 \text{ for } i = 1, \cdots, N \big\}.$$

CGM method [25] then solves the following primal-dual system using New-

ton's method.

$$\sqrt{\|A_i^T y\|^2 + \beta^2}\, x_i - A_i^T y = 0, \qquad i = 1, \ldots, N. \tag{7.14a}$$

$$Ax + \lambda y - \lambda y_0 = 0. \tag{7.14b}$$

In [25] the parameter β is fixed, preventing the solution of the above system from converging to the true optimizer, which is a solution of system (7.13). Moreover, if we pick a very small $\beta > 0$ to reduce this effect, the convergence of the method can become slow.

To overcome this drawback of CGM, we propose a method that dynamically reduces β during the solution process. In such a scheme, β can decrease to a very small number to make the solution arbitrary close to the true optimality solution. Furthermore, the warm start of β from a relative large number will improve the convergence over picking a very small β initially.

Remember the parameter β in CGM is similar to the barrier parameter μ in the KKT system for the SOCP formulation. However, the *primal-dual interior-point* method adopted to solve SOCP has a reduction schedule for μ to decrease to 0. Hence, in reducing β to 0 in CGM, we are just borrowing the same idea from the primal-dual interior-point method.

Analogically to SOCP, we can define the *central path* as the trajectory of points

$$\mathcal{C} = \{(x_\beta, y_\beta) \,|\, \beta > 0\},$$

where (x_β, y_β) solves the equations (7.14) for each $\beta > 0$. As $\beta \to 0$, (x_β, y_β) will converge to the optimality solution. Different from [25], where β is a constant, the β in our algorithm is a measure of the duality gap and updated accordingly at each iteration.

Let us define the following notations:

$$\rho_i^\beta = \sqrt{\|A_i^T y\|^2 + \beta^2}\ ; \tag{7.15}$$

$$E_\beta = \mathrm{Diag}(\rho_i^\beta \mathbf{I}_2), \quad F_\beta = \mathrm{Diag}(\mathbf{I}_2 - \frac{1}{\rho_i^\beta} x_i y^T A_i)\,. \tag{7.16}$$

Then the feasibility and complementary condition (7.14) can be written as

$$\begin{pmatrix} Ax + \lambda y - \lambda y_0 \\ E_\beta\, x - A^T y \end{pmatrix} = 0 \tag{7.17}$$

Applying Newton's method to the above equations gives the following linear system for the updates $(\Delta x, \Delta y)$:

$$\begin{bmatrix} A & \lambda \mathbf{I}_N \\ E_\beta & -F_\beta A^T \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_d \\ r_c \end{bmatrix}, \tag{7.18}$$

where

$$r_d = \lambda(y_0 - y) - Ax \quad \text{and} \quad r_c = A^T y - E_\beta\, x. \tag{7.19}$$

The system can be solved as follows:

$$G_\beta\, \Delta y \;=\; -AE_\beta^{-1}A^T y - \lambda(y - y_0)\,; \tag{7.20}$$

$$\Delta x \;=\; E_\beta^{-1}A^T y + E_\beta^{-1}F_\beta A^T \Delta y - x\,, \tag{7.21}$$

where $G_\beta = AE_\beta^{-1}F_\beta A^T + \lambda \mathbf{I}_N$.

We choose different steplength rule to update the primal y and the dual x. For the dual, to ensure strict feasibility (i.e. $\|x_i\| < 1$ for $i = 1, \ldots, N$), we define the updating rule as

$$\tilde{x} = x + \min(1, 0.99\alpha_{\max})\Delta x \tag{7.22}$$

where $\alpha_{\max} = \max\{\alpha : \|x_i + \alpha \Delta x_i\| \le 1, \quad i = 1, \ldots, N\}$

Predictor-Corrector Enhancement We now discuss how to adapt Mehrotra's predictor-corrector method to our problem. One motivation of this method is to estimate the centering parameter β by the performance of the predictor step (i.e. affine scaling step in LP). The other important idea is to take into account the second-order terms in the corrector step as to compensate for some of the nonlinearity in the system. A key point is that both predictor and corrector share the same matrix factorization, so the extra work in the corrector step is little. We refer to [70] for an excellent discussion of Mehrotra's method.

Let $\tilde{y}$ and $\tilde{x}$ be the primal and dual variables updated in the predictor step. As a natural generalization of Mehrotra's method in LP, a heuristic value of the centering parameter β to be used in the corrector step can be defined by

$$\tilde{\beta} = \sigma \, \frac{\mathrm{gap}(x, y)}{N}, \quad \sigma = \Big(\frac{\mathrm{gap}(\tilde{x}, \tilde{y})}{\mathrm{gap}(x, y)}\Big)^3, \tag{7.23}$$

where

$$\mathrm{gap}(x, y) = \sum_{i=1}^{N} \Big(\|A_i^T y\| - x_i^T (A_i^T y) \Big) \tag{7.24}$$

To compute the second-order correction, we replace x and y in the centering condition (**??**) by $x + \Delta x$ and $y + \Delta y$ respectively and obtain, for $i = 1, \ldots, N$,

$$\sqrt{\|A_i^T (y + \Delta y)\|^2 + \tilde{\beta}^2} \; (x_i + \Delta x_i) - A_i^T (y + \Delta y) = 0,$$

i.e.,

$$\rho_i^{\tilde{\beta}} \sqrt{1 + 2\frac{y^T A_i A_i^T \Delta y}{(\rho_i^{\tilde{\beta}})^2} + \frac{\|A_i^T \Delta y\|^2}{(\rho_i^{\tilde{\beta}})^2}} \; (x_i + \Delta x_i) - (A_i^T y + A_i^T \Delta y) = 0$$

Apply Taylor series and neglecting higher order terms gives

$$\rho_i^{\tilde{\beta}} \Big(1 + \frac{y^T A_i A_i^T \Delta y}{(\rho_i^{\tilde{\beta}})^2} + \frac{\|A_i^T \Delta y\|^2}{2(\rho_i^{\tilde{\beta}})^2} - \frac{(y^T A_i A_i^T \Delta y)^2}{2(\rho_i^{\tilde{\beta}})^4} \Big) (x_i + \Delta x_i) - (A_i^T y + A_i^T \Delta y) = 0$$

Rearranging terms, we obtain

$$E_{\tilde{\beta}}\Delta x - F_{\tilde{\beta}}A^T\Delta y = (\tilde{r}_c)_i + h_i^{(1)}, \tag{7.25}$$

where $\tilde{r}_c = A^T y - E_{\tilde{\beta}}\, x$ and

$$h_i^{(1)} = -\frac{y^T A_i A_i^T \Delta y}{\rho_i^{\tilde{\beta}}}\Delta x_i - \frac{\|A_i^T\Delta y\|^2}{2\rho_i^{\tilde{\beta}}}x_i + \frac{(y^T A_i A_i^T \Delta y)^2}{2(\rho_i^{\tilde{\beta}})^3}x_i. \tag{7.26}$$

It is not practical to solve (7.25) directly with $\tilde{\beta}$ on the left-hand side, since the factorization of G_β has already been computed in the predictor step using the previous value β. Alternatively, we modify (7.25) to be

$$E_\beta\Delta x - F_\beta A^T\Delta y = (\tilde{r}_c)_i + h_i^{(1)} + h_i^{(2)}, \tag{7.27}$$

with extra corrector term

$$h_i^{(2)} = (\rho_i^\beta - \rho_i^{\tilde{\beta}})\Delta x_i + \left(\frac{1}{\rho_i^\beta} - \frac{1}{\rho_i^{\tilde{\beta}}}\right)(y_i^T A_i A_i^T y_i)x_i. \tag{7.28}$$

There is no correction for the feasibility updating equation

$$A\Delta x + \lambda\Delta y = r_d \tag{7.29}$$

Combining (7.27) and (7.29), we have the corrector system

$$\begin{bmatrix} A & \lambda\mathbf{I}_N \\ E_\beta & -F_\beta A^T \end{bmatrix}\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_d \\ r_c^c \end{bmatrix}, \tag{7.30}$$

where

$$r_c^c = (\tilde{r}_c)_i + h_i^{(1)} + h_i^{(2)}. \tag{7.31}$$

It gives the corrector step as

$$G_\beta\,\Delta y = -A(E_\beta^{-1}r_c^c + x) - \lambda(y - y_0)\,; \tag{7.32}$$

$$\Delta x = E_\beta^{-1}(F_\beta A^T\Delta y + r_c^c). \tag{7.33}$$

7.2.1 Numerical Experiments

We test our prima-dual interior point algorithm on two image denoising problem. We also compared the performance of the interior point method with CGM method. The original clean images and the input noisy images are shown in Figure 7.1. The size of the two test problems are 128×128 and 512×512 respectively. The noisy images are generated by adding Gaussian noise to the clean images using the MATLAB function `imnoise`, with variance parameter set to 0.01. The fidelity parameter λ is taken to be 0.045 throughout the experiments.

Figure 7.2 plots the relative duality gap against the number of iterations for CGM method as well as for the primal-dual interior point method.

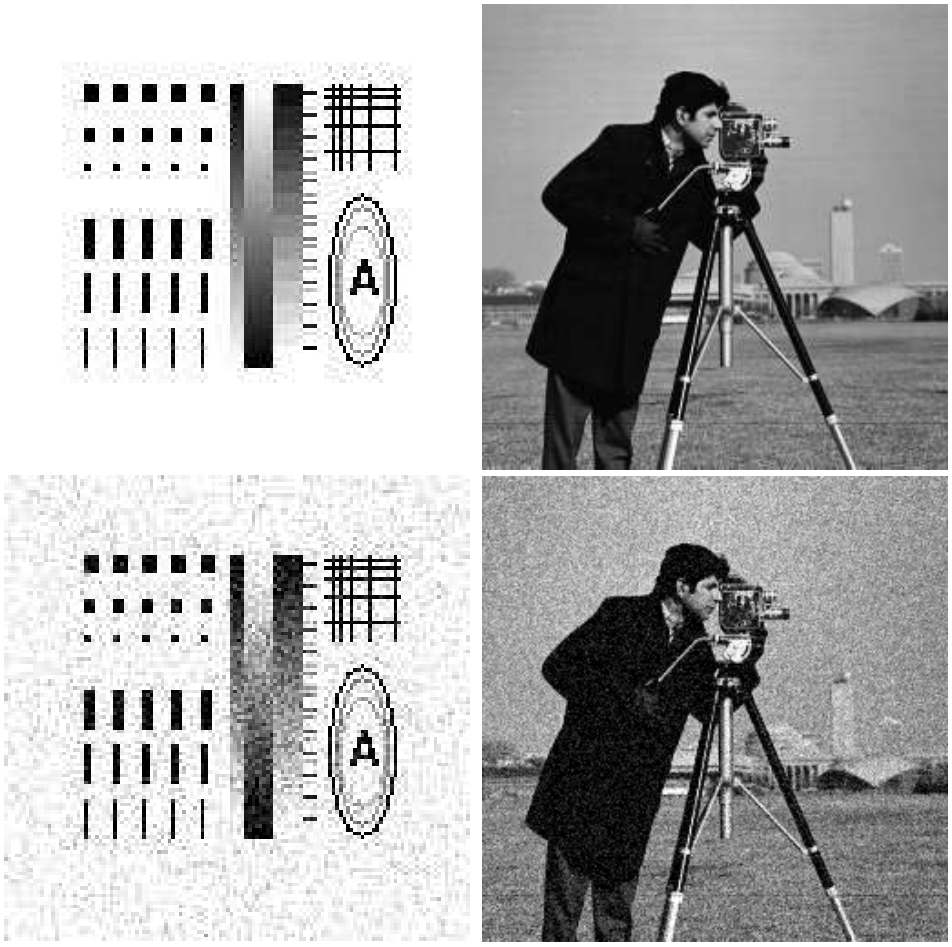

Figure 7.1: The original clean images and noisy images for our test problems. Left: 128×128 "shape"; right: 256×256 "cameraman".

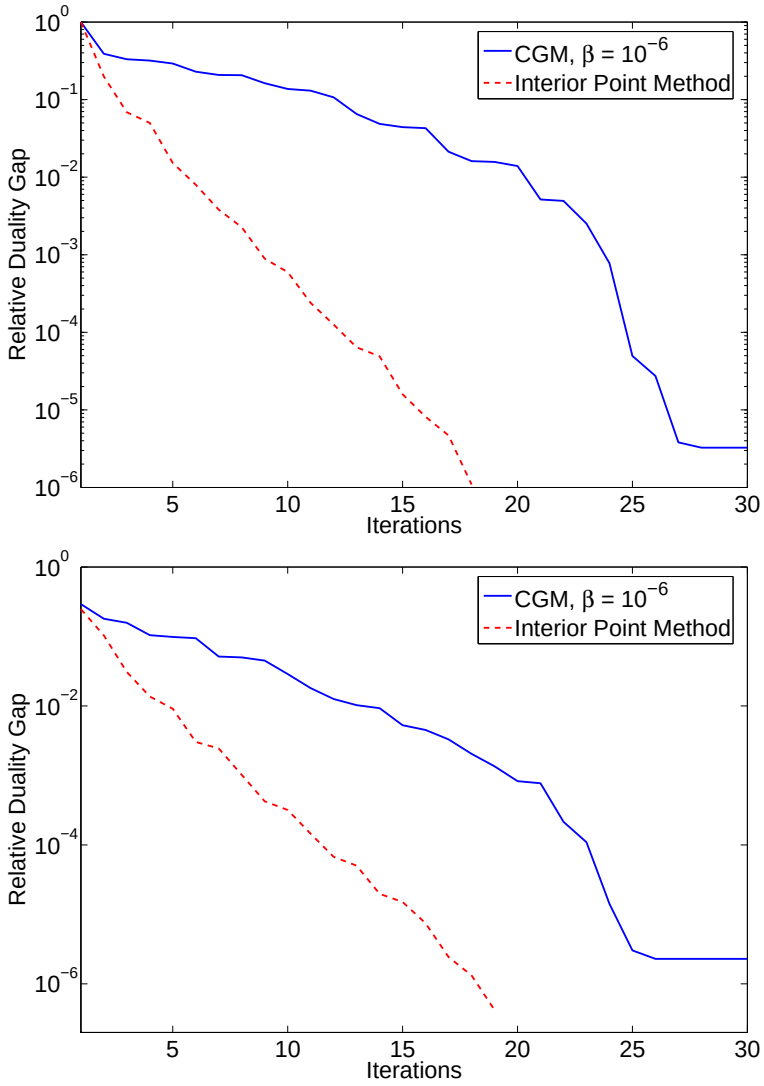

Figure 7.2: Plot of relative duality gap v.s. Iterations. Top : test problem 1, Bottom: test problem 2.

CHAPTER 8

Conclusions and Discussions

We have proposed some new algorithms to solve the total variation based image restoration models, including *primal-dual hybrid gradient descent method*, duality based *gradient projection* method and the dual *block coordinate descent method*. All of them are very efficient and competitive to the existing popular methods. These proposed methods are either based on the dual formulation or the primal-dual formulation so they do not need any smoothing parameter for the total variation term which will prevent any algorithm from converging to the true optimality solution. They are explicit first-order methods, which are simple to implement, only take a few sweeps at each iteration, do not require a huge amount of memory and hence are very suitable for solving large-scale image restoration problems.

Of all the methods we proposed, the primal-dual hybrid gradient method are the most efficient in all situations. It costs roughly 20 times less CPU time to compute an visually-satisfactory medium-accuracy solution compared with Chambolle's semi-implicit gradient-descent method. This advantage get more profound when higher-accuracy solution are required. In fact, this method remains competitive in computing high-accuracy solutions even compared with high-order methods like CGM.

We proposed a basic multilevel optimization framework for the dual formulation of the anisotropic ROF model. The improvement of the multilevel scheme

over the uni-levlel scheme is limited, implies that more sophisticated coarsening and interpolation schemes are needed to exploit the advantage of multigrid algorithms.

We also studied the connection between CGM method and the SOCP approach for solving ROF model and proposed an improvement of the CGM algorithm based on the primal-dual interior-point methods.

References

[1] R. Acar & C. R. Vogel. Analysis of total variation penalty methods for ill-posed problems, *Inverse Problems* , 10 (1994) 1217 - 1229.

[2] S. T. Acton, Multigrid anisotropic difustion, IEEE Trans. Imag. Proc., 3(1998), 280-291.

[3] F. Alizadeh & D. Goldfarb. Second-order cone programming, Mathematical Programming, 95 (2003), 3-51.

[4] K. ANDERSEN, E. CHRISTIANSEN, A. CONN, AND M. OVERTON, *An efficient primal-dual interior-point method for minimizing a sum of Euclidean norms*, SIAM J. Sci. Comput., 22 (2000), pp. 243-262.

[5] G. Aubert & P. Kornprobst. Mathematical Problems in Image Processing - Partial Differential Equations and the Calculus of Variations, 2nd edition. Springer, Applied Mathematical Sciences, Vol 147, 2006.

[6] J. Barzilai and J. Borwein. Two point step size gradient methods, IMA Journal of Numerical Analysis 8 (1988), pp. 141–148.

[7] D. P. Bertsekas. Nonlinear Programming, 2nd ed., Athena Scientific, Boston, 1999.

[8] Bertsekas, D. P. & Tsitsiklis, J. N., *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[9] E. G. Birgin, J. M. Martinez & M. Raydan. Nonmonotone spectral projected gradient methods on convex sets, SIAM Journal on Optimization 10 (2000), pp. 1196–1121.

[10] P. Blomgren & T. Chan. Color TV: Total Variation Methods for Restoration of Vector-Valued Images, *IEEE Trans. Image Proc.*, 7(1998), 304-309.

[11] P. Blomgren, T. F. Chan, P. Mulet, L. Vese & W. L. Wan. Variational PDE models and methods for image processing, in: Research Notes in Mathematics, 420 (2000), 43-67, Chapman & hall/CRC.

[12] M. Burger, S. Osher, J. Xu, & G. Gilboa, Nonlinear inverse scale space methods for image restoration. *Lecture Notes in Computer Science*, Vol. 3752, pp. 25-36, 2005.

[13] J. L. Carter. Dual method for total variation-based image restoration, UCLA CAM Report 02-13, 2002.

[14] A. Chambolle. An algorithm for total variation minimization and applications, J. Math. Imag. Vis. 20 (2004), pp. 89–97.

[15] A. Chambolle. Total variation minimization and a class of binary MRF models. In: *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 136-152, Springer Berlin, 2005.

[16] A. Chambolle, Total Variation minimization and a class of binary MRF models, In Springer-Verlag, (ed.), *5th International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition(EMMCVPR)*, Vol. LNCS 3757, pp. 136152, 2005.

[17] A. Chambolle & P. L. Lions, Image recovery via total variation minimization and related problmes, Numer. Math. 76 (1997), pp. 167–188.

[18] R. H. Chan, T. F. Chan & W. L. Wan, Multigrid for differential convolution problems arising from image processing, *UCLA CAM Report* 97-20, 1997.

[19] T. F. Chan, K. Chen, & X. C. Tai. Nonlinear multilevel scheme for solving the total variation image minimization problem, in: *Image Processing Based on Partial Differential Equations*, Springer Berlin Heidelberg, 2005.

[20] T. F. Chan & K. Chen. On a nonlinear multigrid algorithm with primal relaxation for the image total variation minimisation, *Numerical Algorithms*, 41 (2006), 387-411.

[21] T. F. Chan & K. Chen, An optimization based total variation image denoising. SIAM J. Multiscale Modeling and Simulation, Vol 5(2), pp.615-645, 2006.

[22] T. F. Chan & S. Esedoglu. Aspects of total variation regularized L^1 function approximation. *SIAM Journal on Applied Mathematics*, 65:5 (2005), 1817-1837.

[23] T. Chan, S. Esedoglu, & F. E. Park. A Fourth Order Dual Method for Staircase Reduction in Texture Extraction and Image Restoration Problems, UCLA CAM Report 05-28, 2005.

[24] T. F. Chan, S. Esedoglu, F. Park & A. Yip. Total variation image restoration: overview and rescent developments. In Handbook of Mathematical Models in Computer Vision. Springer Verlag, 2005. Edt. by: N. Paragios, Y. Chen, O. Faugeras.

[25] T. F. Chan, G. H. Golub & P. Mulet. A nonlinear primal dual method for total variation based image restoration, SIAM J. Sci. Comput. 20 (1999), pp. 1964–1977.

[26] T. Chan, S. Kang, & J. Shen. Euler's Elastica and Curvature-based Image Inpainting. *SIAM J. Appl. Math.*, 63(2):564-592, 2002.

[27] T. F. Chan & P. Mulet. On the Convergence of the Lagged Diffusivity Fixed Point Method in Total Variation Image Restoration, SIAM J. Numer. Analysis, 36 (1999), 354-367.

[28] T. F. Chan & J. Shen. Image processing and analysis, variational, PDE, wavelet and wtochastic methods. SIAM, Philadelphia, 2005.

[29] T. Chan & J. Shen. Mathematical Models of Local Non-texture Inpaintings, *SIAM J. Appl. Math.*, 62 (1019-1043), 2001.

[30] T. Chan & C. Wong. Total Variation Blind Deconvolution, *IEEE Trans. Image Process.*, 7(1998), 370-375.

[31] T. F. Chan, M. Zhu. Fast Algorithms for Total Variation-Based Image Processing. To appear in: The Proceedings of 4th ICCM, Hangzhou, China 2007.

[32] T. F. Chan, H. M. Zhou & R. H. Chan, Continuation Method for Total Variation Denoising Problems, UCLA CAM Report 95-28, 1995.

[33] P. Charbonnier, L. Blanc-Feraud, G. Aubert & M.Barluad. Deterministic edge-preserving regularization in computed imaging. IEEE Trans. Image Processing 6 (1997), 298-311.

[34] K. Chen & X-C Tai, A Nonlinear Multigrid Method For Total Variation Minimization From Image Restoration. *Journal of Scientific Computing*, Vol. 33 (2), pp.115-138, 2007.

[35] Y.-H. Dai, W. W. Hager, K. Schittkowski, & H Zhang. The cyclic Barzilai-Borwein method for unconstrained optimization, IMA J. Num. Ana. 26 (2006), pp. 604–627.

[36] Y.-H. Dai & R. Fletcher. Projected Barzilai-Borwein methods for large-scale box constrained quadratic programming, Numerische Mathematik 100 (2005), pp. 21–47.

[37] J. Darbon & M. Sigelle. Exact optimization of discrete constrained total variation minimization problems. In: R. Klette and J. Zunic, editors, *Tenth International Workshop on Combinatorial Image Analysis*, volume 3322 of LNCS (2004), 548-557.

[38] J. Darbon & M. Sigelle. A fast and exact algorithm for total variation minimization. In: J. S. Marques, N. Prez de la Blanca, and P. Pina, editors, *2nd Iberian Conference on Pattern Recognition and Image Analysis*, volume 3522 of LNCS (2005) 351-359.

[39] D. C. Dobson & C. R. Vogel. Convergence of an iterative method for total variation denoising SIAM J. Numer. Analysis, 34 (1997), 1779-1791.

[40] I. Ekeland & R. Témam. Convex Analysis and Variational Problems. SIAM Classics in Applied Mathematics, 1999.

[41] C. Frohn-Schauf, S. Henn & K. Witsch. Nonlinear multigrid methods for total variation image denoising, *Comput. Visual Sci.*, 7 (2004), 199-206.

[42] E. Giusti. Minimal Surfaces and Functions of Bounded Variation. Birkhäuser, Boston, 1984.

[43] D. Goldfarb & W. Yin. Second-order cone programming methods for total variation-based image restoration, SIAM J. Sci. Comput. 27 (2005), pp. 622–645.

[44] L. Grippoa & M. Sciandroneb. On the convergence of the block nonlinear Gauss Seidel method under convex constraints, *Operations Research Letters* 26 (2000) 127-136.

[45] P.T. Harker & J.S. Pang. Finite-dimensional variational inequality and nonlinear complementarity problems: a survey of theory, algorithms and applications. *Math. Programming* 48 (1990), 161-220.

[46] M. Hintermüller & G. Stadler. An infeasible primal-dual algorithm for TV-based inf-convolution-type image restoration, SIAM J. Sci. Comput. 28 (2006), pp. 1–23.

[47] J.-B. Hiriart-Urruty & C. Lemaréchal. *Convex Analysis and Minimization Algorithms, Volume I*, Springer-Verlag, Berlin & Heidelberg, 1993.

[48] D. Hochbaum, An efficient algorithm for image segmentation, Markov Random Fields and related problems, *Journal of the ACM*, Vol. 48, No. 2, pp. 686701, 2001.

[49] A. N. Iusem. On the convergence properties of the projected gradient method for convex optimization, Computational and Applied Mathematics 22 (2003), pp. 37–52.

[50] H. Ishikawa, Exact optimization for Markov random fields with convex priors, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 10, pp. 13331336, 2003.

[51] G. Korpelevich. The extragradient method for finding saddle points and other poblems. *Ekonomika i Matematicheskie Metody* 12 (1976), 747-756.

[52] Y. Meyer. Oscillating Patterns in Image Processing and Nonlinear Evolution Equations, Uiversity Lecture Series, vol 22. AMS, Providence, 2001.

[53] J. Nocedal, S. J. Wright. Nemerical Optimization, Springer, 1999.

[54] A. Nemirovski. Prox-Method with Rate of Convergence $O(1/t)$ for Variational Inequalities with Lipschitz Continuous Monotone Operators and Smooth Convex-Concave Saddle Point Problems. *SIAM Journal on Optimization* 15 (2005), 229-251.

[55] Yu. Nesterov. Dual extrapolation and its applications for solving variational inequalities and related problems. *Math. Program., Ser. B* 109 (2007), 319344.

[56] M. Ng , L. Qi , Y. Yang & Y. Huang, On Semismooth Newtons Methods for Total Variation Minimization. *Journal of Mathematical Imaging and Vision*, 3(27), 2007.

[57] M. Nikolova. Minimizers of cost-functions involving nonsmooth data fidelity terms, *SIAM J. Numer. Anal.*, 40 (2002), 965-994.

[58] M. A. Noor. New extragradient-type methods for general variational inequalities. *Journal of Math. Anal. Appl.* 277 (2003), 379-394.

[59] S. Osher & A. Marquina. Explicit algorithms for a new time dependent model based on level set motion for nonlinear deblurring and noise removal, SIAM J. Sci. Comput. 22 (2000), pp. 387-405.

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

[61] S. Osher, M. Burger, D. Goldfarb, J. Xu & W. Yin, An Iterative Regularization Method for Total Variation Based Image Restoration. *SIAM J. Multiscale Modeling and Simulation* 4(2), 460-489, 2005.

[62] T. Serafini, G. Zanghirati, & L. Zanni. Gradient projection methods for large quadratic programs and applications in training support vector machines, Optimzation Methods and Software 20 (2004), pp. 353–378.

[63] J. Savage & K. Chen. An improved and accelerated nonlinear multigrid method for total-variation denoising, *Int. J. Comput. Math.*, 82 (2005), 1001-1005.

[64] P. S. Vassilevski & J. G. Wade. A comparison of multilevel methods for total variation regularization, *Electronic Trans. On Numer. Anal.*, 6 (1997), 255-270.

[65] L. Vese & S. Osher. Modeling Textures with Total Variation Min- imization and Oscillating Patterns In Image Processing, *J. Math. Imaging Vision*, 20 (2004), 7-18.

[66] C. R. Vogel & M. E. Oman. Iterative methods for total variation denoising, SIAM J. Sci. Stat. Comput. 17 (1996), pp. 227–238.

[67] C. R. Vogel & M. E. Oman. Fast, robust total variation-based reconstruction of noisy, blurred images, IEEE Trans. Image Proc., 7 (1998), 813-824.

[68] C. R. Vogel. A multigrid method for total variation-based image denoising, in: *Computation and control IV, 20, Progress in systems and control theory*, Birkhauser, 1995.

[69] C. R. Vogel. Negative resutls for multilevel preconditioners in image deblurring, in: *Scale-space theroies in computer vision*, Springer-Verlag, 1999.

[70] S. WRIGHT, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997.

[71] N. Xiu & J. Zhang. Some recent advances in projection-type methods for variational inequalities. *J. of Comp. and Applied Math.* 152 (2003), 559-585.

[72] B. Zalesky, Network Flow Optimization for Restoration of Images. *Journal of Applied Mathematics*, Vol. 2, No. 4, pp. 199218, 2002.