Relaxation Methods for Image Denoising Based on Difference Schemes

Rong-Qing Jia†, Hanqing Zhao† and Wei Zhao†
Department of Mathematical and Statistical Sciences
University of Alberta
Edmonton, Canada T6G 2G1
rjia@ualberta.ca, hzhao@math.ualberta.ca and wazhao@math.ualberta.ca

Abstract

In this paper, we propose some relaxation methods that can be used to design very fast iteration schemes for image denoising based on the total variation model. By using certain techniques from convex optimization, we establish the convergence of the iteration schemes based on these relaxation methods. Furthermore, we provide some empirical formulas for the parameters needed in the denoising model. As a result, we are able to construct automatic algorithms for image denoising that produce nearly optimal results. Finally, we apply the relaxation methods to image denoising based on high-order difference schemes. The resulting iteration scheme is fast and yields significantly better image quality than the numerical schemes based on the total variation model.

Key words and phrases: image denoising, difference schemes, total variation, relaxation methods, optimization.

AMS Mathematics Subject Classification: 94A08, 68U10, 49M20, 65K10.

Abbreviated Title: Relaxation Methods for Image Denoising.

†Supported in part by NSERC Canada under Grant OGP 121336

Relaxation Methods for Image Denoising Based on Difference Schemes

§1. Introduction

An image is regarded as a function from $\{1,\dots,N\}\times\{1,\dots,N\}$ to $\mathbb{R}$, where $N\ge 2$. Suppose $u\in\mathbb{R}^{J_N\times J_N}$, where $J_N:=\{1,\dots,N\}$. For $1\le p<\infty$, let

$$\|u\|_p:=\Big(\sum_{1\le i,j\le N}|u(i,j)|^p\Big)^{1/p},$$

and let $\|u\|_\infty:=\max_{1\le i,j\le N}|u(i,j)|$. The inner product of two vectors $u,v\in\mathbb{R}^{J_N\times J_N}$ is defined to be

$$\langle u,v\rangle:=\sum_{1\le i,j\le N}u(i,j)v(i,j). \tag{1.1}$$

Clearly, $\|u\|_2^2=\langle u,u\rangle$. For a linear operator A on $\mathbb{R}^{J_N\times J_N}$, we use A^T to denote the adjoint operator of A with respect to the inner product given in (1.1). If $A=A^T$, then we say that A is **symmetric**. A symmetric linear operator A is said to be **positive definite** if $\langle Av,v\rangle>0$ for every nonzero vector v in $\mathbb{R}^{J_N\times J_N}$. We use I to denote the identity operator on $\mathbb{R}^{J_N\times J_N}$.

Let $f\in\mathbb{R}^{J_N\times J_N}$ be an observed image with noise. We wish to recover a target image u from f by denoising. The TV (Total Variation) model of Rudin, Osher, and Fatemi [21] for image denoising is considered to be one of the best denoising models. The anisotropic TV model for denoising can be formulated as the following minimization problem with an appropriately chosen positive parameter μ:

$$\min_u\Big[\|\nabla_x u\|_1+\|\nabla_y u\|_1+\frac{\mu}{2}\|u-f\|_2^2\Big], \tag{1.2}$$

where ∇_x denotes the difference operator given by $\nabla_x u(1,j)=0$ for $j=1,\dots,N$ and

$$\nabla_x u(i,j)=u(i,j)-u(i-1,j),\quad i=2,\dots,N,\ j=1,\dots,N,$$

and ∇_y is the difference operator given by $\nabla_y u(i,1)=0$ for $i=1,\dots,N$ and

$$\nabla_y u(i,j)=u(i,j)-u(i,j-1),\quad i=1,\dots,N,\ j=2,\dots,N.$$

This motivates us to consider the general minimization problem of a convex function on the n-dimensional Euclidean space $\mathbb{R}^n$. Let $E:\mathbb{R}^n\to\mathbb{R}$ be a convex function. A vector h in $\mathbb{R}^n$ is called a **subgradient** of E at a point $v\in\mathbb{R}^n$ if

$$E(u)-E(v)-\langle h,u-v\rangle\ge 0\quad\forall\, u\in\mathbb{R}^n.$$

The **subdifferential** $\partial E(v)$ is the set of subgradients of E at v. It is known that the subdifferential of a convex function at any point is nonempty. If $g\in\partial E(u)$ and $h\in\partial E(v)$, then

$$E(u)-E(v)-\langle h,u-v\rangle\ge 0\quad\text{and}\quad E(v)-E(u)-\langle g,v-u\rangle\ge 0.$$

It follows that $\langle g-h, u-v\rangle \geq 0$. Clearly, v is a minimal point of E if and only if $0 \in \partial E(v)$. If this is the case, we write

$$v = \arg\min_u \{E(u)\}.$$

If E is given by $E(u) = |u| + \frac{\lambda}{2}(u-c)^2$, $u \in \mathbb{R}$, where $\lambda > 0$ and $c \in \mathbb{R}$, then $0 \in \partial E(v)$ if and only if $v = shrink(c, 1/\lambda)$, where

$$shrink(c, 1/\lambda) := \begin{cases} c - 1/\lambda & \text{for } c > 1/\lambda, \\ 0 & \text{for } -1/\lambda \leq c \leq 1/\lambda, \\ c + 1/\lambda & \text{for } c < -1/\lambda. \end{cases}$$

This soft thresholding operator was introduced by Donoho in [11].

For $\lambda > 0$ and $c \in \mathbb{R}$, we define

$$cut(c, 1/\lambda) := \begin{cases} 1/\lambda & \text{for } c > 1/\lambda, \\ c & \text{for } -1/\lambda \leq c \leq 1/\lambda, \\ -1/\lambda & \text{for } c < -1/\lambda. \end{cases}$$

Clearly, $shrink(c, 1/\lambda) + cut(c, 1/\lambda) = c$. Let $v = (v_1, \ldots, v_n)$ and $c = (c_1, \ldots, c_n)$ be two vectors in $\mathbb{R}^n$. We write $v = shrink(c, 1/\lambda)$ if $v_i = shrink(c_i, 1/\lambda)$, $i = 1, \ldots, n$. Analogously, we write $v = cut(c, 1/\lambda)$ if $v_i = cut(c_i, 1/\lambda)$, $i = 1, \ldots, n$.

Suppose E is the function on $\mathbb{R}^n$ given by

$$E(u) = \|u\|_1 + \frac{\lambda}{2}\|u - c\|_2^2, \quad u \in \mathbb{R}^n,$$

where $\lambda > 0$ and $c = (c_1, \ldots, c_n) \in \mathbb{R}^n$. Given $v = (v_1, \ldots, v_n) \in \mathbb{R}^n$, we see that $0 \in \partial E(v)$ if and only if $v = shrink(c, 1/\lambda)$.

Suppose u^* is the unique solution to the minimization problem (1.2). In order to find the solution u^*, Goldstein and Osher in [13] proposed to use the Bregman method, as introduced in [3]. Following [13], we introduce new vectors $v_x, v_y \in \mathbb{R}^{J_N \times J_N}$ and consider the minimization problem

$$\min_{v_x, v_y, u} \Big[E(v_x, v_y, u) + \frac{\lambda}{2}\|v_x - \nabla_x u\|_2^2 + \frac{\lambda}{2}\|v_y - \nabla_y u\|_2^2 \Big],$$

where $\lambda > 0$ and $E(v_x, v_y, u) := \|v_x\|_1 + \|v_y\|_1 + (\mu/2)\|u - f\|_2^2$. Choose $b_x^0 = b_y^0 = 0$ and $v_x^0 = v_y^0 = 0$. For $k = 0, 1, 2, \ldots$, let

$$(v_x^{k+1}, v_y^{k+1}, u^{k+1}) := \operatorname*{argmin}_{v_x, v_y, u} [E(v_x, v_y, u) + H^k(v_x, v_y, u)], \tag{1.3}$$

where $H^k(v_x, v_y, u) := (\lambda/2)\|v_x - \nabla_x u - b_x^k\|_2^2 + (\lambda/2)\|v_y - \nabla_y u - b_y^k\|_2^2$, and let

$$b_x^{k+1} := b_x^k - (v_x^{k+1} - \nabla_x u^{k+1}), \quad b_y^{k+1} := b_y^k - (v_y^{k+1} - \nabla_y u^{k+1}). \tag{1.4}$$

It was proved in [16] that $\lim_{k\to\infty} u^k = u^*$.

The minimization problem in (1.3) is still difficult to solve. To overcome the difficulty, the split Bregman method was introduced in [13]. This method separates the variables u and v in (1.3) as follows:

$$u^{k+1} := \arg\min_u [E(v_x^k, v_y^k, u) + H^k(v_x^k, v_y^k, u)],$$

$$(v_x^{k+1}, v_y^{k+1}) := \operatorname*{argmin}_{v_x, v_y} [E(v_x, v_y, u^{k+1}) + H^k(v_x, v_y, u^{k+1})].$$

The above two minimization problems can be solved in the following way:

$$(\mu - \lambda\Delta)u^{k+1} = \mu f + \lambda\nabla_x^T(v_x^k - b_x^k) + \lambda\nabla_y^T(v_y^k - b_y^k), \tag{1.5}$$

where $\Delta := -\nabla_x^T\nabla_x - \nabla_y^T\nabla_y$, and

$$v_x^{k+1} = shrink(\nabla_x u^{k+1} + b_x^k, 1/\lambda), \quad v_y^{k+1} = shrink(\nabla_y u^{k+1} + b_y^k, 1/\lambda). \tag{1.6}$$

The convergence of the iteration scheme given by (1.5), (1.6), and (1.4) could be established by using the proximal forward-backward splitting algorithm based on the Moreau-Yosida regularization. See the work [9] of Combettes and Wajs. Also, see the recent paper [4] of Cai, Osher, and Shen for more general discussions on split Bregman methods. The actual implementation in [13] was to use the Gauss-Seidel method once in each iteration step to solve the linear system of equations in (1.5). As far as we know, however, the convergence of this iteration scheme has not been established.

An alternative method for the minimization problem in (1.3) was proposed in [17]. Suppose u^k is known. Solve (v_x, v_y) in (1.3):

$$(v_x^k, v_y^k) := \operatorname*{argmin}_{v_x, v_y} [E(v_x, v_y, u^k) + H^{k-1}(v_x, v_y, u^k)].$$

Suppose that the linear operator $\mu I + \lambda\Delta$ is positive definite. Let B be the square root of $\mu I + \lambda\Delta$. Using the Bregman method, we let u^{k+1} be the solution of the following minimization problem:

$$u^{k+1} := \arg\min_u \Big\{\frac{1}{2}\|B(u-f)\|_2^2 - \langle B^2(u^k - f), u - u^k\rangle + \frac{\lambda}{2}\|v_x^k - \nabla_x u\|_2^2 + \frac{\lambda}{2}\|v_y^k - \nabla_y u\|_2^2\Big\}.$$

After solving the above two minimization problems, we obtain the following iteration scheme: Set $b_x^0 := 0$, $b_y^0 := 0$, and $u^1 := f$. For $k = 1, 2, \ldots$, let

$$b_x^k := cut(\nabla_x u^k + b_x^{k-1}, 1/\lambda), \quad b_y^k := cut(\nabla_y u^k + b_y^{k-1}, 1/\lambda), \tag{1.7}$$

$$u^{k+1} := f - \frac{\lambda}{\mu}(\nabla_x^T b_x^k + \nabla_y^T b_y^k). \tag{1.8}$$

For details, see Lemmas 1 and 2 of [17]. It was proved there that $\lim_{k\to\infty} u^k = u^*$, provided $\lambda/\mu \le 1/8$.

The above iteration scheme could also be derived from the Uzawa algorithm (see [1, Chap. 10]). The Uzawa algorithm can be viewed as a subgradient method applied to the dual problem. By considering the problem dual to the minimization problem (1.2), Chambolle in [5] and [6] developed (sub)gradient-based algorithms. However, he did not establish convergence for the iteration scheme given by (1.7) and (1.8). Instead, he gave a proof of convergence for a different iteration scheme, as given in [5]. Recently, Beck and Teboulle in [2] also considered the dual problem and established the sublinear rate of convergence in function values under the condition $\lambda/\mu \le 1/8$.

Let $\tau := \lambda/\mu$. Then τ can be viewed as the step size of the iteration scheme. If τ is too small, then the iteration scheme may converge very slowly. If τ is too large, the iteration scheme may diverge. In this paper, we propose certain relaxation methods which admit a wider range of step sizes. In this way we can speed up the iteration process significantly.

Here is the outline of the paper. In §2 we introduce two relaxation methods for the minimization problem (1.2). Moreover, we establish the convergence of the iteration schemes based on these relaxation methods. In §3 we report the numerical results of image denoising by using the relaxation methods. For the anisotropic model (1.2) for denoising, we compare the performance of our algorithm with the GO algorithm of Goldstein and Osher as given in [13]. To achieve the same or slightly better image quality, our algorithm only requires 10% – 20% of the time needed for the GO algorithm. In §4, we design some automatic algorithms for image denoising based on the total variation model. We provide an empirical formula to estimate the parameter μ and demonstrate that our automatic algorithm produces nearly optimal results. In §5, we extend our results to image denoising based on high-order difference schemes. Our relaxation method gives fast iteration schemes that yield significantly better image quality than the numerical schemes based on the total variation model. Finally, in §6, we make concluding remarks and discuss topics for possible future research.

§2. Relaxation Methods for the Total Variation Model

Relaxation methods are frequently used in numerical linear algebra. A linear system of equations has the form $Ax = y$, where A is an invertible $n \times n$ matrix, y is a given n-vector, and x is the unknown n-vector. A simple iteration scheme for solving the linear system $Ax = y$ may be described as follows. Let F be an invertible $n \times n$ matrix. Starting with an initial guess x^0, we perform the iteration scheme

$$x^{k+1} := x^k + F(y - Ax^k), \quad k = 0, 1, 2, \ldots.$$

If $\lim_{k\to\infty} x^k = x^*$, then $Ax^* = y$. A relaxation method employs a weighted average of the update $x^k + F(y - Ax^k)$ and the previous value x^k:

$$x^{k+1} := (1 - t)x^k + t[x^k + F(y - Ax^k)] = x^k + tF(y - Ax^k),$$

where the weight factor $t > 0$ is called the relaxation parameter. Relaxation methods often accelerate the convergence of the classical Jacobi and Gauss-Seidel iterations. See [18, §4.2] for more details of the Jacobi method with relaxation and the Gauss-Seidel method with relaxation.

Motivated by the preceding discussion, we derive from (1.7) and (1.8) the following relaxation scheme: Set $b_x^0 := 0$, $b_y^0 := 0$, and $u^1 := f$. For $k = 1, 2, \ldots$, let

$$b_x^k := (1-t)b_x^{k-1} + t(cut(\nabla_x u^k + b_x^{k-1}, 1/\lambda)), \tag{2.1}$$

$$b_y^k := (1-t)b_y^{k-1} + t(cut(\nabla_y u^k + b_y^{k-1}, 1/\lambda)), \tag{2.2}$$

$$u^{k+1} := f - \frac{\lambda}{\mu}(\nabla_x^T b_x^k + \nabla_y^T b_y^k). \tag{2.3}$$

Clearly, the iteration scheme given in (1.7) and (1.8) is the special case of the current iteration scheme when $t = 1$.

In the following theorem we establish convergence of the relaxation scheme. Our proof is motivated by the techniques employed in [8] and [4].

Theorem 1. *Let $(u^k)_{k=1,2,\ldots}$ be the sequence produced by the iteration scheme given in (2.1), (2.2), and (2.3). If $0 < t < 2$ and $\lambda/\mu \le (2-t)/4$, then $\lim_{k\to\infty} u^k = u^*$.*

Proof. Let G denote the function given by $G(v) := \|v\|_1$ for $v \in \mathbb{R}^{J_N \times J_N}$. Recall that u^* is the unique solution to the minimization problem (1.2). Let $v_x^* := \nabla_x u^*$ and $v_y^* := \nabla_y u^*$. It follows from (1.2) that there exist some $p_x^* \in \partial G(v_x^*)$ and $p_y^* \in \partial G(v_y^*)$ such that

$$\nabla_x^T p_x^* + \nabla_y^T p_y^* + \mu(u^* - f) = 0.$$

Let $b_x^* := p_x^*/\lambda$ and $b_y^* := p_y^*/\lambda$. Consequently,

$$\mu(u^* - f) + \lambda(\nabla_x^T b_x^* + \nabla_y^T b_y^*) = 0. \tag{2.4}$$

For $k = 1, 2, \ldots$, let

$$v_x^k := \arg\min_v \Big\{ \|v\|_1 + \frac{\lambda}{2}\|v - \nabla_x u^k - b_x^{k-1}\|_2^2 \Big\}, \tag{2.5}$$

$$v_y^k := \arg\min_v \Big\{ \|v\|_1 + \frac{\lambda}{2}\|v - \nabla_y u^k - b_y^{k-1}\|_2^2 \Big\}. \tag{2.6}$$

It follows that $v_x^k = shrink(\nabla_x u^k + b_x^{k-1}, 1/\lambda)$ and $v_y^k = shrink(\nabla_y u^k + b_y^{k-1}, 1/\lambda)$. This together with (2.1) and (2.2) gives

$$b_x^k = b_x^{k-1} + t(\nabla_x u^k - v_x^k) \quad \text{and} \quad b_y^k = b_y^{k-1} + t(\nabla_y u^k - v_y^k). \tag{2.7}$$

Moreover, let $p_x^k := -\lambda(v_x^k - \nabla_x u^k - b_x^{k-1})$ and $p_y^k := -\lambda(v_y^k - \nabla_y u^k - b_y^{k-1})$. Then by (2.5) and (2.6) we have $p_x^k \in \partial G(v_x^k)$ and $p_y^k \in \partial G(v_y^k)$.

For $k = 1, 2, \ldots$, denote the errors by

$$u_e^k := u^k - u^*, \quad v_{x,e}^k := v_x^k - v_x^*, \quad v_{y,e}^k := v_y^k - v_y^*, \quad b_{x,e}^k := b_x^k - b_x^*, \quad b_{y,e}^k := b_y^k - b_y^*. \tag{2.8}$$

It follows from (2.7) that

$$b_{x,e}^{k+1} = b_{x,e}^k + t(\nabla_x u_e^{k+1} - v_{x,e}^{k+1}) \quad \text{and} \quad b_{y,e}^{k+1} = b_{y,e}^k + t(\nabla_y u_e^{k+1} - v_{y,e}^{k+1}). \tag{2.9}$$

Moreover, by (2.3) we have $\mu(u^{k+1} - f) + \lambda(\nabla_x^T b_x^k + \nabla_y^T b_y^k) = 0$. Subtracting (2.4) from this equation, we obtain

$$\mu u_e^{k+1} + \lambda(\nabla_x^T b_{x,e}^k + \nabla_y^T b_{y,e}^k) = 0.$$

Taking the inner product of both sides of the above equation with u_e^{k+1}, we get

$$\mu\langle u_e^{k+1}, u_e^{k+1}\rangle + \lambda\langle b_{x,e}^k, \nabla_x u_e^{k+1}\rangle + \lambda\langle b_{y,e}^k, \nabla_y u_e^{k+1}\rangle = 0. \tag{2.10}$$

Recall that $p_x^* = \lambda b_x^*$, $v_x^* = \nabla_x u^*$, and $p_x^{k+1} = -\lambda(v_x^{k+1} - \nabla_x u^{k+1} - b_x^k)$. Hence,

$$p_x^{k+1} - p_x^* + \lambda(v_{x,e}^{k+1} - \nabla_x u_e^{k+1} - b_{x,e}^k) = 0.$$

Since $p_x^{k+1} \in \partial G(v_x^{k+1})$ and $p_x^* \in \partial G(v_x^*)$, we have $\langle p_x^{k+1} - p_x^*, v_x^{k+1} - v_x^*\rangle \geq 0$. Taking the inner product of both sides of the above equation with $v_{x,e}^{k+1} = v_x^{k+1} - v_x^*$, we get

$$\langle p_x^{k+1} - p_x^*, v_x^{k+1} - v_x^*\rangle + \lambda\langle v_{x,e}^{k+1}, v_{x,e}^{k+1}\rangle - \lambda\langle \nabla_x u_e^{k+1}, v_{x,e}^{k+1}\rangle - \lambda\langle b_{x,e}^k, v_{x,e}^{k+1}\rangle = 0. \tag{2.11}$$

Similarly,

$$\langle p_y^{k+1} - p_y^*, v_y^{k+1} - v_y^*\rangle + \lambda\langle v_{y,e}^{k+1}, v_{y,e}^{k+1}\rangle - \lambda\langle \nabla_y u_e^{k+1}, v_{y,e}^{k+1}\rangle - \lambda\langle b_{y,e}^k, v_{y,e}^{k+1}\rangle = 0. \tag{2.12}$$

We also have $\langle p_y^{k+1} - p_y^*, v_y^{k+1} - v_y^*\rangle \geq 0$. Adding (2.10), (2.11), and (2.12) together, we obtain

$$\begin{aligned} &\mu\langle u_e^{k+1}, u_e^{k+1}\rangle + \lambda\|v_{x,e}^{k+1}\|_2^2 + \lambda\|v_{y,e}^{k+1}\|_2^2 - \lambda\langle \nabla_x u_e^{k+1}, v_{x,e}^{k+1}\rangle - \lambda\langle \nabla_y u_e^{k+1}, v_{y,e}^{k+1}\rangle \\ &+\lambda\langle b_{x,e}^k, \nabla_x u_e^{k+1} - v_{x,e}^{k+1}\rangle + \lambda\langle b_{y,e}^k, \nabla_y u_e^{k+1} - v_{y,e}^{k+1}\rangle + \gamma_{k+1} = 0, \end{aligned} \tag{2.13}$$

where $\gamma_k := \langle p_x^k - p_x^*, v_x^k - v_x^*\rangle + \langle p_y^k - p_y^*, v_y^k - v_y^*\rangle \geq 0$. We deduce from (2.9) that

$$\begin{aligned} \lambda\langle b_{x,e}^k, \nabla_x u_e^{k+1} - v_{x,e}^{k+1}\rangle &= \tfrac{\lambda}{2t}(\|b_{x,e}^{k+1}\|_2^2 - \|b_{x,e}^k\|_2^2) - \tfrac{t\lambda}{2}\|\nabla_x u_e^{k+1} - v_{x,e}^{k+1}\|_2^2, \\ \lambda\langle b_{y,e}^k, \nabla_y u_e^{k+1} - v_{y,e}^{k+1}\rangle &= \tfrac{\lambda}{2t}(\|b_{y,e}^{k+1}\|_2^2 - \|b_{y,e}^k\|_2^2) - \tfrac{t\lambda}{2}\|\nabla_y u_e^{k+1} - v_{y,e}^{k+1}\|_2^2. \end{aligned}$$

Substituting the above two equations into (2.13), we get

$$\mu\|u_e^{k+1}\|_2^2 + \lambda[g_t(v_{x,e}^{k+1}, \nabla_x u_e^{k+1}) + g_t(v_{y,e}^{k+1}, \nabla_y u_e^{k+1})] + \frac{\lambda}{2t}(\alpha_{k+1} - \alpha_k) + \gamma_{k+1} = 0, \tag{2.14}$$

where $\alpha_k := \|b_{x,e}^k\|_2^2 + \|b_{y,e}^k\|_2^2$ and g_t is the function given by

$$g_t(v,w) := (1-t/2)\|v\|_2^2 - (1-t)\langle v,w\rangle - (t/2)\|w\|_2^2, \quad v,w \in \mathbb{R}^{J_N\times J_N}.$$

Note that

$$\frac{2-t}{2}\|v\|_2^2 - (1-t)\langle v,w\rangle + \frac{(1-t)^2}{2(2-t)}\|w\|_2^2 \geq 0 \quad \forall\, v,w \in \mathbb{R}^{J_N\times J_N}.$$

It follows that

$$g_t(v,w) \geq -\frac{(1-t)^2}{2(2-t)}\|w\|_2^2 - \frac{t}{2}\|w\|_2^2 = -\frac{1}{2(2-t)}\|w\|_2^2. \tag{2.15}$$

This together with (2.14) implies the following inequality:

$$\mu\|u_e^{k+1}\|_2^2 - \frac{\lambda}{4-2t}(\|\nabla_x u_e^{k+1}\|_2^2 + \|\nabla_y u_e^{k+1}\|_2^2) \leq \frac{\lambda}{2t}(\alpha_k - \alpha_{k+1}).$$

Recall that $\Delta = -\nabla_x^T\nabla_x - \nabla_y^T\nabla_y$. Hence,

$$\mu\|u_e^{k+1}\|_2^2 - \frac{\lambda}{4-2t}(\|\nabla_x u_e^{k+1}\|_2^2 + \|\nabla_y u_e^{k+1}\|_2^2) = \langle(\mu I + \lambda\Delta/(4-2t))u_e^{k+1}, u_e^{k+1}\rangle.$$

By our assumption, $\lambda/\mu \leq (2-t)/4$, and hence the linear operator $\mu I + \lambda\Delta/(4-2t)$ is positive definite. Thus, there exists some $\rho > 0$ such that $\langle(\mu I + \lambda\Delta/(4-2t))u,u\rangle \geq \rho\|u\|_2^2$ for all $u \in \mathbb{R}^{J_N\times J_N}$. Consequently,

$$\rho\|u_e^{k+1}\|_2^2 \leq \frac{\lambda}{2t}(\alpha_k - \alpha_{k+1}), \quad k = 1,2,\ldots.$$

Summing this inequality over $k = 1,\ldots,K$ gives

$$\sum_{k=1}^{K}\rho\|u_e^{k+1}\|_2^2 \leq \sum_{k=1}^{K}\frac{\lambda}{2t}(\alpha_k - \alpha_{k+1}) \leq \frac{\lambda\alpha_1}{2t} = \frac{\lambda}{2t}(\|b_{x,e}^1\|_2^2 + \|b_{y,e}^1\|_2^2).$$

This shows that the series $\sum_{k=1}^{\infty}\|u_e^{k+1}\|_2^2$ converges. Consequently, $\lim_{k\to\infty} u_e^k = 0$, that is, $\lim_{k\to\infty} u^k = u^*$. □

According to Theorem 1, the relaxation method admits a wider range of step sizes. Recall that the step size $\tau = \lambda/\mu$. In particular, for $t = 1/2$, the iteration scheme converges, provided $0 < \tau \leq 3/8$. Our numerical experiments show that the iteration scheme usually converges if $0 < \tau \leq 1/2$.

We are in a position to introduce the second relaxation method to find the unique solution u^* to the minimization problem (1.2). This method gives rise to the following iteration scheme: Set $b_x^0 := 0$, $b_y^0 := 0$, and $u^1 := f$. For $0 < s \leq 1$ and $k = 1,2,\ldots$, let b_x^k and b_y^k be given by (1.7), and let

$$u^{k+1} := (1-s)u^k + s[f - (\lambda/\mu)(\nabla_x^T b_x^k + \nabla_y^T b_y^k)]. \tag{2.16}$$

Theorem 2. *Let $(u^k)_{k=1,2,\ldots}$ be the sequence produced by the iteration scheme given in (1.7) and (2.16). If $0 < s \leq 1$, and if the linear operator*

$$(2-s)^2 I - (\lambda/\mu)(\nabla_x^T \nabla_x + \nabla_y^T \nabla_y)/2$$

is positive definite, then $\lim_{k\to\infty} u^k = u^*$.

Proof. The proof is analogous to that of Theorem 1. For $k = 1, 2, \ldots$, let v_x^k and v_y^k be the same as given by (2.5) and (2.6), respectively. Moreover, let u_e^k, $v_{x,e}^k$, $v_{y,e}^k$, $b_{x,e}^k$, and $b_{y,e}^k$ be the errors as defined in (2.8). It follows from (2.16) that

$$\mu u^{k+1} - \mu(1-s)u^k - \mu s f + s\lambda(\nabla_x^T b_x^k + \nabla_y^T b_y^k) = 0.$$

We deduce from (2.4) that

$$\mu u^* - \mu(1-s)u^* - \mu s f + s\lambda(\nabla_x^T b_x^* + \nabla_y^T b_y^*) = 0.$$

Subtracting the second equation from the first one, we obtain

$$\mu u_e^{k+1} - \mu(1-s)u_e^k + s\lambda(\nabla_x^T b_{x,e}^k + \nabla_y^T b_{y,e}^k) = 0. \tag{2.17}$$

Taking the inner product of both sides of the above equation with $u_e^{k+1} + (1-s)u_e^k$, we get

$$\mu[\|u_e^{k+1}\|_2^2 - (1-s)^2\|u_e^k\|_2^2] + \lambda s \eta_k = 0, \tag{2.18}$$

where

$$\eta_k := \langle b_{x,e}^k, \nabla_x u_e^{k+1}\rangle + \langle b_{y,e}^k, \nabla_y u_e^{k+1}\rangle + (1-s)\langle b_{x,e}^k, \nabla_x u_e^k\rangle + (1-s)\langle b_{y,e}^k, \nabla_y u_e^k\rangle.$$

By summing both sides of equation (2.18) over $k = 1, \ldots, K$, we obtain

$$\mu[\|u_e^{K+1}\|_2^2 - \|u_e^1\|_2^2] + \sum_{k=1}^{K} \mu s(2-s)\|u_e^k\|_2^2 + \lambda s \sum_{k=1}^{K} \eta_k = 0. \tag{2.19}$$

In order to estimate $\sum_{k=1}^K \eta_k$, we derive from (2.11),(2.12), and (2.9) that

$$\langle b_{x,e}^k, \nabla_x u_e^{k+1}\rangle + \langle b_{y,e}^k, \nabla_y u_e^{k+1}\rangle = (\alpha_{k+1} - \alpha_k)/2 + \gamma_{k+1}/\lambda + \beta_{k+1}/2 - \delta_{k+1}/2.$$

where $\alpha_k := \|b_{x,e}^k\|_2^2 + \|b_{y,e}^k\|_2^2$, $\beta_k := \|v_{x,e}^k\|_2^2 + \|v_{y,e}^k\|_2^2$, $\delta_k := \|\nabla_x u_e^k\|^2 + \|\nabla_y u_e^k\|^2$, and $\gamma_k := \langle p_x^k - p_x^*, v_x^k - v_x^*\rangle + \langle p_y^k - p_y^*, v_y^k - v_y^*\rangle$. Furthermore,

$$\langle b_{x,e}^k, \nabla_x u_e^k\rangle + \langle b_{y,e}^k, \nabla_y u_e^k\rangle = (\alpha_k - \alpha_{k-1})/2 + \gamma_k/\lambda + \beta_k/2 + \delta_k/2 - \theta_k,$$

where $\theta_k := \langle v_{x,e}^k, \nabla_x u_e^k\rangle + \langle v_{y,e}^k, \nabla_y u_e^k\rangle$. Since $\gamma_k \geq 0$ for all $k \geq 1$, we obtain

$$\eta_k \geq \frac{1}{2}(\alpha_{k+1} - \alpha_k) + \frac{1-s}{2}(\alpha_k - \alpha_{k-1}) + \frac{1}{2}\beta_{k+1} + \frac{1-s}{2}\beta_k - \frac{1}{2}\delta_{k+1} + \frac{1-s}{2}\delta_k - (1-s)\theta_k.$$

We have

$$\sum_{k=1}^{K}\Big[\frac{1}{2}(\alpha_{k+1}-\alpha_k)+\frac{1-s}{2}(\alpha_k-\alpha_{k-1})\Big]=\frac{1}{2}(\alpha_{K+1}-\alpha_1)+\frac{1-s}{2}(\alpha_K-\alpha_0),$$

$$\sum_{k=1}^{K}\Big[\frac{1}{2}\beta_{k+1}+\frac{1-s}{2}\beta_k\Big]=\frac{1}{2}[\beta_{K+1}-\beta_1]+\frac{2-s}{2}\sum_{k=1}^{K}\beta_k,$$

$$\sum_{k=1}^{K}\Big[-\frac{1}{2}\delta_{k+1}+\frac{1-s}{2}\delta_k\Big]=-\frac{1}{2}[\delta_{K+1}-\delta_1]-\frac{s}{2}\sum_{k=1}^{K}\delta_k.$$

By the construction of $b_{x,e}^k$ and $b_{y,e}^k$ we have $\|\lambda b_{x,e}^k\|_\infty \le 2$ and $\|\lambda b_{y,e}^k\|_\infty \le 2$. In light of (2.17), the sequence $(\|u_e^k\|_\infty)_{k=1,2,\dots}$ is bounded. Consequently, the sequence $(\delta_k)_{k=1,2,\dots}$ is bounded. Taking (2.19) into account, we see that there exists a positive constant M independent of K such that

$$\sum_{k=1}^{K}\mu(2-s)\|u_e^k\|_2^2+\sum_{k=1}^{K}\lambda\Big[\frac{2-s}{2}\beta_k-(1-s)\theta_k-\frac{s}{2}\delta_k\Big]\le M.$$

By (2.15) we have

$$\frac{2-s}{2}\beta_k-(1-s)\theta_k-\frac{s}{2}\delta_k\ge-\frac{1}{2(2-s)}[\|\nabla_x u_e^k\|_2^2+\|\nabla_y u_e^k\|_2^2].$$

Note that

$$\|\nabla_x u_e^k\|_2^2+\|\nabla_y u_e^k\|_2^2=\langle-\Delta u_e^k,u_e^k\rangle.$$

Consequently,

$$\sum_{k=1}^{K}\langle(\mu(2-s)I+\lambda\Delta/(4-2s))u_e^k,u_e^k\rangle\le M.$$

By our assumption, the linear operator $\mu(2-s)^2I+\lambda\Delta/2$ is positive definite. Hence, the series $\sum_{k=1}^{\infty}\|u_e^k\|_2^2$ converges. Consequently, $\lim_{k\to\infty}u_e^k=0$, that is, $\lim_{k\to\infty}u^k=u^*$. □

By Theorem 2, we have $\lim_{k\to\infty}u^k=u^*$, provided the step size $\tau=\lambda/\mu\le(2-s)^2/4$. In particular, for $s=1/2$, this is true if $\tau\le 9/16$.

§3. Numerical Experiments

In this section we report the numerical results of image denoising by using the relaxation method as described in the iteration scheme given in (2.1), (2.2), and (2.3). We fix the relaxation parameter $t=0.5$ and the step size $\tau=0.5$. Thus, for given μ, $\lambda=0.5\mu$.

In what follows, all the images considered have the size 512×512 and the grey-scale in the range between 0 and 255. A Gaussian noise with the normal distribution $N(0,\sigma^2)$ is

added to the original image. Let u be the original image, and let f be the noised image. By u^{k+1} we denote the result after k iterations.

For image processing, the image quality is usually measured in terms of the Peak Signal-to-Noise Ratio (PSNR), which is defined by $PSNR = 20\log_{10} M/\sqrt{E}$, where M is the maximum possible pixel value of the image and E is the mean squared error. In our case, $M = 255$ and $E = \|u^{k+1} - u\|_2^2/N^2$ with $N = 512$.

We test our (JZZ) algorithm and compare our algorithm with the GO algorithm of Goldstein and Osher [13] on four images: *Lena*, *Boat*, *Goldhill*, and *Bridge*.

In this section, for each image and each σ, we assume that the optimal value of μ is known. Automatic estimation of μ will be discussed in the next section.

All the computation is conducted on a Lenovo desktop with 2 GB memory and an Intel Core 2 CPU 6400 at 2.13 GHz. We use gcc to write a C code to implement our algorithm.

We first deal with image *Lena*. In the following table, the first row gives the value of σ, and the second row gives the PSNR value of the noisy image obtained by adding a Gaussian noise with the normal distribution $N(0, \sigma^2)$ to the original image. The denoising results by the GO algorithm are shown in the third, fourth, and fifth rows. The third row indicates the PSNR value of the denoised image after N_{it} iterations. For the GO algorithm, N_{it} is chosen to be $\sigma + 10$. The fifth row records the CPU time in seconds needed for the iterations performed. The denoising results by the JZZ algorithm are shown in the sixth, seventh, and eighth rows. The sixth row indicates the PSNR value of the denoised image after N_{it} iterations. For the JZZ algorithm, N_{it} is chosen to be $0.6\sigma - 2$. The eighth row records the CPU time in seconds needed for the iterations conducted. From the table we see that the PSNR values of our algorithm are slightly better than the corresponding PSNR values of the GO algorithm. But our algorithm only requires 10% – 20% of the time needed for the GO algorithm.

Table 1: Denoising Results of *Lena* for Optimal μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
optimal	μ	0.17	0.11	0.075	0.058	0.046	0.039	0.034
denoised	PSNR	34.33	32.46	31.17	30.16	29.45	28.83	28.21
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.31	0.41	0.47	0.53	0.61	0.69	0.78
denoised	PSNR	34.47	32.54	31.23	30.19	29.48	28.85	28.23
by the JZZ	N_{it}	4	7	10	13	16	19	22
Algorithm	time	0.03	0.08	0.08	0.09	0.13	0.14	0.16

In Tables 2, 3, and 4, we list denoising results for images *Boat*, *Goldhill*, and *Bridge*, respectively. For these images, we choose $N_{it} = 0.4\sigma$ for the JZZ algorithm.

Table 2: Denoising Results of *Boat* for Optimal μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
optimal	μ	0.20	0.12	0.088	0.066	0.052	0.043	0.038
denoised	PSNR	32.45	30.53	29.15	28.14	27.33	26.69	26.13
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.33	0.47	0.48	0.55	0.64	0.69	0.77
denoised	PSNR	32.56	30.61	29.20	28.18	27.36	26.71	26.15
by the JZZ	N_{it}	4	6	8	10	12	14	16
Algorithm	time	0.03	0.06	0.06	0.08	0.09	0.09	0.11

Table 3: Denoising Results of *Goldhill* for Optimal μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
optimal	μ	0.20	0.12	0.083	0.063	0.051	0.041	0.035
denoised	PSNR	32.56	30.71	29.52	28.62	27.95	27.39	26.88
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.33	0.47	0.48	0.55	0.61	0.69	0.75
denoised	PSNR	32.66	30.81	29.60	28.69	28.01	27.46	26.95
by the JZZ	N_{it}	4	6	8	10	12	14	16
Algorithm	time	0.03	0.06	0.07	0.08	0.08	0.09	0.11

Table 4: Denoising Results of *Bridge* for Optimal μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
optimal	μ	0.29	0.16	0.11	0.081	0.065	0.052	0.045
denoised	PSNR	30.52	28.18	26.72	25.69	24.90	24.27	23.74
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.38	0.42	0.50	0.58	0.66	0.72	0.80
denoised	PSNR	30.57	28.23	26.76	25.73	24.95	24.31	23.79
by the JZZ	N_{it}	4	6	8	10	12	14	16
Algorithm	time	0.03	0.05	0.06	0.06	0.08	0.09	0.12

A very different approach for the total variation minimization problem in image restoration was proposed by Darbon and Sigelle in [10]. Their method relied on the decomposition of an image into its level sets. Solutions of minimization problems at each level were obtained by using the technique of graph cuts. We downloaded the latest version of the executable codes from Darbon's website and tested the codes. From our numerical experiments we found that the GO algorithm outperformed the DS (Darbon and Sigelle) algorithm. First, the GO algorithm is faster than the DS algorithm. It was reported in [10] that their algorithm took about 3 second on a Pentium4 3GHz for denoising an image of size 512×512. The latest version of their codes took between 1 and 1.25 second on our computer for denoising an image of size 512×512. Second, the quality of the denoised image produced by the GO algorithm is better than the one generated by the DS algorithm as the PSNR values are at least 0.5 dB higher.

§4. Automatic Algorithms for Image Denoising

In this section we design some automatic algorithms for image denoising based on the total variation model.

First, we need to estimate the standard deviation σ for the noise. For natural images, Donoho in [11], and Wang and Shang in [22] gave good estimates for σ. Suppose that f is the observed image of size $N \times N$. Express

$$\left\{\sqrt{(|\nabla_x f(i,j)|^2 + |\nabla_y f(i,j)|^2)/2} : 1 \le i, j \le N\right\}$$

as a sequence. Let θ be the median value of this sequence. In light of the results in [22], we use the following estimation for σ:

$$\sigma = 1.0482\,\theta.$$

Assuming σ is known, we employ the following procedure to estimate μ. For each σ, let

$$\mu_\sigma^0 := \frac{2.15}{\sigma} - 0.02.$$

Then we perform the iteration scheme as given in (2.1), (2.2), and (2.3) with $\mu = \mu_\sigma^0$, $t = 0.5$, and $\lambda = 0.5\mu$. After 0.4σ iterations we get $u = u^{0.4\sigma+1}$. Then we compute

$$\tau_\sigma := \frac{\|\nabla_x u\|_1 + \|\nabla_y u\|_1}{2N^2} - (5.9943 - 0.0566\sigma).$$

Furthermore, we set

$$\mu_\sigma := \mu_\sigma^0 + 0.0088|\tau_\sigma|\tau_\sigma + 0.0023. \tag{4.1}$$

With the automatic choice of $\mu = \mu_\sigma$, we test the GO algorithm and the JZZ algorithm on four images: *Lena*, *Boat*, *Goldhill*, and *Bridge*. For the GO algorithm, the number N_{it} of iterations is chosen to be $\sigma + 10$. For the JZZ algorithm, N_{it} is chosen to be $0.4\sigma + 2$. The

following tables describe the numerical results. We see that the PSNR values with automatic μ are close to the PSNR values with optimal μ.

Table 5: Denoising Results of *Lena* for Automatic μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
automatic	μ	0.18	0.11	0.080	0.061	0.049	0.041	0.035
denoised	PSNR	34.35	32.45	31.15	30.14	29.44	28.83	28.21
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.31	0.39	0.45	0.53	0.59	0.66	0.73
denoised	PSNR	34.45	32.52	31.20	30.16	29.45	28.83	28.21
by the JZZ	N_{it}	6	8	10	12	14	16	18
Algorithm	time	0.03	0.05	0.06	0.08	0.09	0.11	0.12

Table 6: Denoising Results of *Boat* for Automatic μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
automatic	μ	0.20	0.13	0.090	0.068	0.054	0.044	0.036
denoised	PSNR	32.45	30.53	29.14	28.13	27.34	26.70	26.11
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.31	0.41	0.47	0.53	0.59	0.67	0.73
denoised	PSNR	32.51	30.59	29.18	28.16	27.36	26.71	26.14
by the JZZ	N_{it}	6	8	10	12	14	16	18
Algorithm	time	0.05	0.06	0.06	0.08	0.09	0.11	0.12

Table 7: Denoising Results of *Goldhill* for Automatic μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
automatic	μ	0.19	0.12	0.085	0.064	0.050	0.041	0.035
denoised	PSNR	32.53	30.72	29.52	28.62	27.95	27.39	26.87
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.33	0.41	0.47	0.53	0.59	0.66	0.73
denoised	PSNR	32.60	30.78	29.58	28.68	28.01	27.45	26.94
by the JZZ	N_{it}	6	8	10	12	14	16	18
Algorithm	time	0.03	0.05	0.08	0.08	0.09	0.11	0.12

Table 8: Denoising Results of *Bridge* for Automatic μ

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
automatic	μ	0.28	0.17	0.12	0.084	0.063	0.048	0.039
denoised	PSNR	30.51	28.19	26.72	25.69	24.90	24.23	23.65
by the GO	N_{it}	20	25	30	35	40	45	50
Algorithm	time	0.33	0.41	0.50	0.58	0.64	0.70	0.77
denoised	PSNR	30.54	28.21	26.75	25.72	24.93	24.28	23.70
by the JZZ	N_{it}	6	8	10	12	14	16	18
Algorithm	time	0.05	0.06	0.06	0.08	0.09	0.11	0.12

Automatic algorithms for image denoising were also considered by Chambolle in [5]. Following [5, §4], we have the following iteration scheme: Set $b_x^0 := 0$, $b_y^0 := 0$, and $u^1 := f$. Choose an initial $\mu^1 > 0$. For $k = 1, 2, \ldots$, let

$$b_x^k := cut(\nabla_x u^k + b_x^{k-1}, 8/\mu^k), \quad b_y^k := cut(\nabla_y u^k + b_y^{k-1}, 8/\mu^k).$$

$$u^{k+1} := f - \frac{1}{8}(\nabla_x^T b_x^k + \nabla_y^T b_y^k).$$

Then update μ_{k+1}:

$$\mu^{k+1} := \mu^k \frac{\|f - u^{k+1}\|_2}{N\sigma}.$$

Our numerical experiments tell us that the sequence $(\mu^k)_{k=1,2,\ldots}$ converges. However, its limit μ often deviates far from the optimal one. See the following table:

Table 9: Denoising Results of *Lena* by Chambolle's Automatic Algorithm

	σ	10	15	20	25	30	35	40
limit	μ	0.11	0.066	0.046	0.034	0.025	0.018	0.013
denoised by	PSNR	33.52	31.58	30.23	29.13	28.10	27.02	25.78
Chambolle's	N_{it}	100	120	150	170	190	210	230
Algorithm	time	1.26	1.33	1.53	1.78	1.94	2.12	2.25

Comparing Table 9 with Table 5, we see that Chambolle's automatic algorithm gives considerably lower PSNR values. In particular, for $\sigma = 40$, the difference could be as large as 2.4 dB. Moreover, it takes 100 – 230 iterations to achieve the stated PSNR values.

§5. Image Denoising Based on High-order Difference Schemes

We have demonstrated that our relaxation methods give rise to very fast schemes for image denoising based on the anisotropic model (1.2). Our relaxation methods can also be extended to the isotropic model and other more sophisticated models for image denoising.

In [15] Lysaker, Lundervold, and Tai introduced an image denoising scheme based on pure fourth order PDEs (partial differential equations). The model of fourth order PDEs is more suitable for smoother images. In [7] Chang, Tai, and Xing proposed a combination model of second order and fourth order PDEs for image denoising. They claimed that their combination model was better than the models of either pure second order or fourth order PDEs.

In this section we put forward an image denoising model based on a combination of the second and fourth order *difference schemes*. By using our relaxation methods we give a denoising scheme which runs faster and performs better than many other known numerical schemes.

For $u \in \mathbb{R}^{J_N \times J_N}$, define

$$\Delta_x u(i,j) := \begin{cases} u(1,j) - u(2,j) & \text{if } i = 1, \\ 2u(i,j) - u(i-1,j) - u(i+1,j), & \text{if } 1 < i < N, \\ u(N,j) - u(N-1,j), & \text{if } i = N, \end{cases}$$

and

$$\Delta_y u(i,j) := \begin{cases} u(i,1) - u(i,2) & \text{if } j = 1, \\ 2u(i,j) - u(i,j-1) - u(i,j+1), & \text{if } 1 < j < N, \\ u(i,N) - u(i,N-1), & \text{if } j = N. \end{cases}$$

Then Δ_x and Δ_y are difference operators on $\mathbb{R}^{J_N \times J_N}$. Evidently, $\Delta_x^T = \Delta_x$ and $\Delta_y^T = \Delta_y$.

Let $f \in \mathbb{R}^{J_N \times J_N}$ be an observed image. Our model of denoising is to find the target image u as the unique solution of the following minimization problem with appropriately chosen positive parameters μ and ν:

$$\min_u \Big\{ \frac{1}{\mu}(\|\nabla_x u\|_1 + \|\nabla_y u\|_1) + \frac{1}{\nu}(\|\Delta_x u\|_1 + \|\Delta_y u\|_1) + \frac{1}{2}\|u - f\|_2^2 \Big\}.$$

Let s be a real number such that $0 < s \le 1$. In order to find the unique solution u^* to the above minimization problem, we propose the following iteration scheme: Set $b_x^0 := 0$, $b_y^0 := 0$, $c_x^0 := 0$, $c_y^0 := 0$, and $u^1 := f$. For $k = 1, 2, \ldots$, let

$$\begin{aligned} b_x^k &:= cut(\nabla_x u^k + b_x^{k-1}, 1/\lambda), \\ b_y^k &:= cut(\nabla_y u^k + b_y^{k-1}, 1/\lambda), \\ c_x^k &:= cut(\Delta_x u^k + c_x^{k-1}, 1/\lambda), \\ c_y^k &:= cut(\Delta_y u^k + c_y^{k-1}, 1/\lambda), \end{aligned}$$

and

$$u^{k+1} := (1-s)u^k + s\Big[f - \frac{\lambda}{\mu}(\nabla_x^T b_x^k + \nabla_y^T b_y^k) - \frac{\lambda}{\nu}(\Delta_x^T c_x^k + \Delta_y^T c_y^k)\Big].$$

An argument analogous to the proof of Theorem 2 gives the following theorem.

Theorem 3. *Let $(u^k)_{k=1,2,\ldots}$ be the sequence produced by the above iteration scheme. If $0 < s \leq 1$, and if the linear operator*

$$(2-s)^2 I - (\lambda/\mu)(\nabla_x^T \nabla_x + \nabla_y^T \nabla_y)/2 - (\lambda/\nu)(\Delta_x^T \Delta_x + \Delta_y^T \Delta_y)/2$$

is positive definite, then

$$\lim_{k\to\infty} u^k = u^*.$$

We test the HD (High-order Difference) model for image denoising on images *Lena* and *Boat*. The parameters are fixed to be $s = 0.2$, $\mu = \nu = 2.4\mu_\sigma$, and $\lambda = 0.4\mu$, where μ_σ is given by (4.1). We adopt the following stopping criterion:

$$\frac{\|u^{k+1} - u^k\|_2}{N} < 0.1.$$

For an image of size 512×512, $N = 512$.

We also consider the following denoising model based on pure fourth order difference scheme with optimal ν:

$$\min_u \Big\{\frac{1}{\nu}(\|\Delta_x u\|_1 + \|\Delta_y u\|_1) + \frac{1}{2}\|u - f\|_2^2\Big\}.$$

The numerical results are listed in Tables 9 and 10. We see that the HD model produces significantly better results than the TV model for images with large smooth parts such as *Lena*. Moreover, the combination HD model outperforms the pure fourth order model.

Table 10: Denoising Results of *Lena* Based on the HD Model

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
TV Model	PSNR	34.35	32.45	31.15	30.14	29.44	28.83	28.21
optimal	ν	0.26	0.14	0.09	0.07	0.05	0.04	0.04
4th Order	PSNR	34.78	32.83	31.47	30.45	29.61	28.92	28.29
automatic	μ and ν	0.43	0.27	0.19	0.15	0.12	0.098	0.084
denoised	PSNR	35.05	33.20	31.90	30.86	30.11	29.49	28.83
by the HD	N_{it}	19	23	26	29	31	33	35
Model	time	0.78	0.95	1.04	1.17	1.22	1.30	1.36

Table 11: Denoising Results of *Boat* Based on the HD Model

	σ	10	15	20	25	30	35	40
noisy	PSNR	28.14	24.62	22.10	20.22	18.68	17.42	16.21
TV Model	PSNR	32.45	30.53	29.14	28.13	27.34	26.70	26.11
optimal	ν	0.34	0.19	0.13	0.09	0.07	0.06	0.05
4th Order	PSNR	32.63	30.65	29.24	28.21	27.39	26.74	26.13
automatic	μ and ν	0.47	0.30	0.22	0.16	0.13	0.11	0.087
denoised	PSNR	32.82	30.97	29.60	28.57	27.78	27.10	26.48
by the HD	N_{it}	18	22	25	28	30	32	35
Model	time	0.73	0.86	1.06	1.17	1.22	1.25	1.36

Figure 1 and Figure 2 are attached at the end of this paper. Figure 1 shows the four images *Lena, Boat, Goldhill*, and *Bridge* used for our numerical experiments. In Figure 2, we compare the denoising effects for image *Lena* with noise level $\sigma = 40$ by using the ROF model and the HD model. We see that the HD model produces higher image quality. In particular, the HD model preserves the smooth part of the image better and reduces the staircase effect.

Recently, Beck and Teboulle in [2] considered the GP (Gradient Projection) method for the dual problem to the denoising problem based on total variation. For the fixed step size $\tau = 1/8$, they extended the method of Nesterov introduced in [19] and developed in [20] that achieves a rate of convergence of $O(1/k^2)$ for values of the objective function. As a result, they proposed the FGP (Fast Gradient Projection) algorithm for image denoising. We tested the FGP algorithm for *Lena* and compared it with the GO (Goldstein and Osher) algorithm for the isotropic model and the HD model. The results are listed in the following table.

Table 12: Comparison of Various Denoising Schemes for *Lena*

	σ	10	15	20	25	30	35	40
denoised	PSNR	34.50	32.60	31.32	30.40	29.61	28.98	28.40
by the GO	N_{it}	20	25	30	35	40	45	50
isotropic	time	0.35	0.45	0.53	0.59	0.68	0.77	0.86
denoised	PSNR	34.50	32.60	31.32	30.40	29.61	28.98	28.40
by the FGP	N_{it}	13	13	14	22	29	29	33
Algorithm	time	0.30	0.30	0.33	0.49	0.64	0.64	0.72
denoised	PSNR	34.49	32.74	31.33	30.50	29.67	29.04	28.42
by the HD	N_{it}	6	8	8	11	12	13	13
Model	time	0.21	0.30	0.31	0.39	0.43	0.48	0.48

From the above table we see that the FGP Algorithm is slightly faster than the GO algorithm for the isotropic model. For $10 \leq \sigma \leq 20$, the HD model and the FGP algorithm have almost the same speed. But, for $25 \leq \sigma \leq 40$, the HD model outperforms the FGP algorithm. Moreover, if we increase the number of iterations, the PSNR values produced by the GO algorithm and the FGP algorithm will be almost the same as shown in Table 12. For the HD model, if we increase the number of iterations, the PSNR values will be increased considerably, as indicated by Table 10.

§6. Conclusion

In this paper we introduce relaxation methods for image denoising based on difference schemes. Our emphasis is placed on the speed of the algorithm. In this aspect, the iteration schemes based on Theorems 1 and 2 for the TV anisotropic model are much faster than the corresponding GO algorithm. In addition to image denoising, our schemes can often be employed in preprocessing of images. Moreover, these denoising schemes can be used to solve subproblems in certain deblurring algorithms as discussed in [2].

In order to increase the quality of denoised images, we extend the relaxation method to an image denoising model based on high order difference schemes, which produces higher image quality. In particular, the HD model preserves the smooth part of the image better and reduces the staircase effect. Our numerical experiments demonstrate that our iteration scheme based on the HD model outperforms both the GO algorithm and the FGP algorithm for the isotropic model.

Another approach to better image denoising is to choose the parameter μ to be location dependent. See the work [12] of Duval, Aujol and Gousseau on non-local means, and the work [14] of Le, Chartrand and Asaki on applications to reconstruction of images corrupted by Poisson noise. But non-local means require computationally demanding algorithms. For more sophisticated models of image denoising, it will be worthwhile to consider more advanced acceleration techniques from optimization. These will be interesting topics for future research. We believe that the relaxation methods will play a useful role in the study.

References

[1] K. J. Arrow, L. Hurwicz, and H. Uzawa, Studies in Linear and Non-linear Programming, Stanford University Press, Stanford, 1958.

[2] A. Beck and M. Teboulle, Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems, IEEE Transactions on Image Processing, published online July 24, 2009.

[3] L. M. Brègman, A relaxation method of finding a common point of convex sets and its application to the solution of problems in convex optimization, USSR Computational Mathematics and Mathematical Physics **7** (1967), 200–217.

[4] J.-F. Cai, S. Osher, and Z. W. Shen, Split Bregman methods and frame-based image restoration, UCLA CAM Report (09-28). (Multiscale Modeling and Simulation, to appear)

[5] A. Chambolle, An algorithm for total variation minimization and applications, J. Math. Imaging Vis. **20** (2004), 89–97.
[6] A. Chambolle, Total variation minimization and a class of binary MRF models, Lecture Notes Comput. Sci, vol. **3757** (2005), pp. 136–152.
[7] Q. Chang, X. Tai, and L. Xing, A compound algorithm of denoising using second-order and fourth-order partial differential equations, Numerical Mathematics: Theory, Methods and Applications **2** (2009), 353–376.
[8] X.-J. Chen, Global and superlinear convergence of inexact Uzawa methods for saddle point problems with nondifferentiable mappings, SIAM J. Numerical Analysis **35** (1998), 1130–1148.
[9] P. Combettes and V. Wajs, Signal recovery by proximal forward-backward splitting, Multiscale Model. Simul. **4** (2005), 1168–1200.
[10] J. Darbon and M. Sigelle, Image restoration with discrete total variation, Part I: fast and exact optimization, J. Math. Imaging Vis. **26** (2006), 261–276.
[11] D. L. Donoho, De-noising by soft-thresholding, IEEE Transactions on Information Theory **41** (1995), 613–627.
[12] V. Duval, J.-F. Aujol and Y. Gousseau, On the parameter choice of the Non-Local Means, CMLA Preprint 2010-06.
[13] T. Goldstein and S. Osher, The split Bregman method for L1 regularized problems, SIAM Journal on Imaging Sciences **2** (2009), 323–343.
[14] T. Le, R. Chartrand and T. Asaki, A Variational approach to constructing images corrupted by Poisson noise, J. Math. Imaging Vis. **27** (2007), 257–263.
[15] M. Lysaker, A. Lundervold, and X.-C. Tai, Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time, IEEE Transactions on Image Processing **12** (2004) 1579 - 1590.
[16] R. Q. Jia, H. Q. Zhao, and W. Zhao, Convergence analysis of the Bregman method for the variational model of image denoising, Applied and Computational Harmonic Analysis **27** (2009), 367–379.
[17] R. Q. Jia and H. Q. Zhao, A fast algorithm for the total variation model of image denoising, Advances in Computational Mathematics, published online May 21, 2009.
[18] R. Kress, Numerical Analysis, Springer-Verlag, New York, 1998.
[19] Y. E. Nesterov, A method of solving a convex programming problem with convergence rate $O(1/k^2)$, Soviet Math. Dokl. **27** (1983), 372–376.
[20] Y. E. Nesterov, Gradient methods for minimizing composite objective function, Tech. Report, CORE, 2007.
[21] L. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms, Physics D **60** (1992), 259–268.
[22] J. Z. Wang and X. Q. Shang, Adaptive smoothing of anisotropic diffusion equations in image denoising, preprint.

(a) Lena

(b) Boat

(c) Goldhill

(d) Bridge

Figure 1: Images for numerical experiments.

(a) The original image

(b) Contaminated with $\sigma = 40$

(c) Denoised by the ROF model

(d) Denoised by the HD model

(e) Denoised by the ROF model: face

(f) Denoised by the HD model: face

Figure 2: *Lena* denoised by the ROF and HD models.