A FAST ALGORITHM FOR EDGE-PRESERVING VARIATIONAL MULTICHANNEL IMAGE RESTORATION

JUNFENG YANG*, WOTAO YIN†, YIN ZHANG †, AND YILUN WANG †

Abstract. We generalize the alternating minimization algorithm recently proposed in [32] to efficiently solve a general, edge-preserving, variational model for recovering multichannel images degraded by within- and cross-channel blurs, as well as additive Gaussian noise. This general model allows the use of localized weights and higher-order derivatives in regularization, and includes a multichannel extension of total variation (MTV) regularization as a special case. In the MTV case, we show that the model can be derived from an extended half-quadratic transform of Geman and Yang [14]. For color images with three channels and when applied to the MTV model (either locally weighted or not), the per-iteration computational complexity of this algorithm is dominated by nine fast Fourier transforms. We establish strong convergence results for the algorithm including finite convergence for some variables and fast q-linear convergence for the others. Numerical results on various types of blurs are presented to demonstrate the performance of our algorithm compared to that of the MATLAB deblurring functions. We also present experimental results on regularization models using weighted MTV and higher-order derivatives to demonstrate improvements in image quality provided by these models over the plain MTV model.

Key words. half-quadratic, cross-channel, image deblurring, isotropic total variation, fast Fourier transform

AMS subject classifications. 68U10, 65J22, 65K10, 65T50, 90C25

1. Introduction. The multichannel (e.g., color) image restoration problem has recently attracted much attention in the imaging community (cf. [3, 12, 29, 18, 11]). In this paper, we study an *alternating minimization algorithm* for recovering multichannel images from their blurry and noisy observations.

Blurs in a multichannel image can be more complicated than those in a singlechannel (i.e., grayscale) image because they can exist either within or cross channels. In this paper, we consider both within- and cross-channel blurs. Without loss of generality, we assume that the underlying images have square domains and let an $n \times n$ image with m channels be denoted by $\bar{u} = [\bar{u}^{(1)}; \ldots; \bar{u}^{(m)}] \in \mathbb{R}^{mn^2}$, where $\bar{u}^{(j)} \in \mathbb{R}^{n^2}$ represents the jth channel for $j = 1, \ldots, m$. An observation of $\bar{u}$ is

$$f = K\bar{u} + \omega, \tag{1.1}$$

where $f \in \mathbb{R}^{mn^2}$ has the same size and the number of channels as $\bar{u}$, ω represents the additive noise, and K is a blurring operator in the form of:

$$K = \begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1m} \\ K_{21} & K_{22} & \cdots & K_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ K_{m1} & K_{m2} & \cdots & K_{mm} \end{bmatrix} \in \mathbb{R}^{mn^2 \times mn^2}, \tag{1.2}$$

where $K_{ij} \in \mathbb{R}^{n^2 \times n^2}$, each diagonal submatrix K_{ii} defines the blurring operator within the ith channel, and each off-diagonal matrix K_{ij}, $i \neq j$, defines how the jth channel affects the ith channel.

*Department of Mathematics, Nanjing University, 22 Hankou Road, Nanjing, Jiangsu Province, 210093, P.R. China (jfyang2992@yahoo.com.cn)

†Department of Computational and Applied Mathematics, Rice University, 6100 Main Street, MS-134, Houston, Texas, 77005, U.S.A. (wotao.yin, yin.zhang, yilun.wang@rice.edu)

It is well-known that recovering $\bar{u}$ from f by inverting (1.1) is an ill-posed problem because the solution is highly sensitive to the noise ω. To stabilize the recovery of $\bar{u}$, one must utilize some a prior information. In such a stabilization scheme, $\bar{u}$ is obtained as the solution of

$$\min_u \Phi_{reg}(u) + \mu\Phi_{fid}(u, f), \tag{1.3}$$

where in the objective function, the *regularization term* $\Phi_{reg}(u)$ models the a prior information about $\bar{u}$ in the sense that $\Phi_{reg}(\bar{u})$ tends to be smaller than $\Phi_{reg}(u)$ for most $u \neq \bar{u}$ that satisfies (1.1), the *fidelity term* $\Phi_{fid}(u, f)$ measures the violation of the relation between u and its observation f (e.g., Eq. (1.1)), and the *penalty parameter* $\mu > 0$ weighs the two terms in the minimization. Traditional regularization techniques such as the Tikhonov regularization [30] and the total variation regularization [22] have been carefully studied for grayscale images. In the literature, for the Gaussian noise, the common fidelity term used is

$$\Phi_{fid}(u, f) = \frac{1}{2}\|Ku - f\|_2^2 \tag{1.4}$$

corresponding to the maximum likelihood estimation of $\bar{u}$. For the impulsive noise, e.g., the so-called salt-and-pepper noise, the common fidelity term is based on the 1-norm instead of the square of the 2-norm as in (1.4). This paper studies the case in which ω is Gaussian and the data fidelity term $\Phi_{reg}(u, f)$ is given by (1.4).

In what follows, we give a brief review of the total variation regularization, summarize the contributions of our work, and then describe the organization of this paper.

1.1. Total variation regularization. Among all regularization techniques, the total variation regularization, first introduced in [22], is well-known for preserving discontinuities in recovered images. Let Ω be a square region in $\mathbb{R}^2$. The total variation (TV) of a grayscale image $u(x) : \Omega \to [0, 1]$ can be defined as

$$\mathrm{TV}(u) = \int_\Omega \|\nabla u\| \mathrm{d}x, \tag{1.5}$$

whenever the gradient ∇u exists, where $\|\cdot\|$ is a norm in $\mathbb{R}^2$. For more general functions, the TV is defined using a dual formulation (cf. [34]), which is equivalent to (1.5) when u is differentiable. In practical computation, a discrete form of (1.5) is always used, given by $\mathrm{TV}(u) = \sum_i \|D_i u\|$ in which $u \in \mathbb{R}^{n^2}$ represents an n-by-n grayscale image, $D_i u \in \mathbb{R}^2$ represents certain first-order finite differences of u at pixel i in horizontal or vertical directions, and the summation is taken over all pixels. If $\|\cdot\|$ is the 2-norm, (1.5) defines an isotropic TV, which means that (1.5) is irrelevant to rotation, reflection and changing of positions of an image. If $\|\cdot\|$ is the 1-norm, (1.5) defines an anisotropic TV. Usually, the isotropic discretization is preferred over any anisotropic ones. We use the first-order forward finite differences and the 2-norm throughout this paper, and our algorithm can be easily extended for an anisotropic discretization of TV. Various methods based on TV regularization have been proposed and studied for recovering grayscale images; see e.g., [31, 6, 7, 10].

Since many TV based algorithms have been proven effective for reducing noise and blur without smearing sharp edges for grayscale images, it is natural to extend the TV regularization to multichannel images. Several approaches for this purpose have been proposed. Among them, the simplest one applies the TV regularization to each channel independently. Besides, the authors of [23, 24, 25] extended TV to deal with vector-valued images based on anisotropic diffusion and geometric active contours. Also in [2], the authors proposed the

so-called "color TV" (see (2.2) below) and used an explicit time marching scheme [31] to minimize it. In this paper, we use a different extension of the singlechannel TV, called MTV (see (2.1) below and [4, 8, 9, 28]), which preserves the desirable properties of (1.5) for multichannel image reconstruction and permits very efficient solution for (1.3).

1.2. Contributions. The main contribution of this paper is an efficient algorithm for solving a very general regularization model for multichannel image deblurring. Based upon our previous work in [32] for singlechannel images, we show that this algorithm can be derived from a general *half-quadratic* formulation, and is applicable to a variety of regularization functions for multichannel image restoration beyond TV, as well as to regularization terms with localized weights. In addition, we establish attractive convergence properties for this algorithm such as global convergence with a strong q–linear rate, and finite convergence for some quantities. Under the periodic boundary conditions, its computation can take advantages of fast operations such as *high-dimensional shrinkage* and the fast Fourier transform (FFT); therefore, our MATLAB implementation was much more efficient than the compared solvers in our experiments.

1.3. Organization. The paper is organized as follows. In Section 2, we focus on an extension of TV to vector-valued functions in general and discrete multichannel images in particular, and present the discrete formulation of (1.3) for deblurring. We also compare this extended TV with the "color TV" proposed in [2]. In Section 3, we apply a *half-quadratic transform* to the original discrete formulation to derive our *alternating minimization algorithm*, and present its convergence properties. Numerical results, including a comparison between the proposed algorithm and algorithms in the MATLAB imaging toolbox, are presented in Section 4. In addition, this section demonstrates advantages of regularization models using weighted TV and higher-order derivatives. Finally, conclusion remarks are given in Section 5.

2. Multichannel TV regularization problem. Before giving the definition of multichannel TV, we introduce some notation. Let $D^{(1)}, D^{(2)} \in \mathbb{R}^{n^2\times n^2}$ be the first-order forward finite difference matrices in horizontal and vertical directions, respectively. As used in the discretized form of (1.5), $D_i \in \mathbb{R}^{2\times n^2}$ is a two-row matrix formed by stacking the ith row of $D^{(1)}$ on that of $D^{(2)}$. For two vectors v_1 and v_2, let $v = (v_1; v_2)$ be the long vector formed by stacking v_1 on the top of v_2. Similarly, $D = (D^{(1)}; D^{(2)}) \triangleq ((D^{(1)})^\top, (D^{(2)})^\top)^\top$. The spectral radius of a matrix is denoted by $\rho(\cdot)$. From here on, the norm $\|\cdot\|$ refers to the 2-norm unless otherwise specified. Additional notation will be introduced as the paper progresses.

To present the definition of multichannel TV and compare it with the "color-TV" (CTV) proposed in [2], we assume temporarily that u is a differentiable function defined on a square region $\Omega \subset \mathbb{R}^2$. Let $u(x) = (u^{(1)}(x); \ldots; u^{(m)}(x)) : \Omega \to \mathbb{R}^m$ be an m–channel image. It is natural to generalize (1.5) to multichannel images as follows,

$$\mathrm{MTV}(u) \triangleq \int_\Omega \|\nabla u\| \mathrm{d}x = \int_\Omega \sqrt{\|\nabla u^{(1)}\|^2 + \cdots + \|\nabla u^{(m)}\|^2}\mathrm{d}x, \tag{2.1}$$

where $\nabla u \in \mathbb{R}^m$ applies $\nabla\cdot$ to all the m channels, namely, $\nabla u \triangleq (\nabla u^{(1)}; \ldots; \nabla u^{(m)})$. $\mathrm{MTV}(u)$ has already been used in the literature for RGB images; see e.g., [4, 8, 9, 28]. For comparison, the "color-TV" proposed in [2] is

$$\mathrm{CTV}(u) = \sqrt{\left(\int_\Omega \|\nabla u^{(1)}\| \mathrm{d}x\right)^2 + \cdots + \left(\int_\Omega \|\nabla u^{(m)}\| \mathrm{d}x\right)^2}. \tag{2.2}$$

It is easy to see from (2.1) and (2.2) that both MTV and CTV preserve the two basic properties of (1.5), namely, i) not overly penalizing discontinuities and ii) rotationally invariant.

Although both MTV and CTV reduce to (1.5) for singlechannel images, they are different in several aspects. First, they treat channels and pixels in different orders. While CTV computes the TV of each channel separately and then combines them, MTV first computes the variation at each pixel with respect to all channels and then sums up the variations over all pixels. Second, MTV and CTV have different geometric interpretations. To illustrate this, let us assume that $\Omega = [a, b] \subset \mathbb{R}$ and $u(t) : \Omega \to \mathbb{R}^m$ be a spatial curve in $\mathbb{R}^m$. Suppose each component of $u(t)$ changes monotonously in Ω. Then, it is easy to see from (2.2) that $\mathrm{CTV}(u) = \|u(a) - u(b)\|$, the Euclidean distance between $u(a)$ and $u(b)$, and from (2.1) that $\mathrm{MTV}(u) = |\widehat{u(a)u(b)}|$, the length of the arc $\widehat{u(a)u(b)}$. Most importantly, the MTV regularization problem allows a fast alternating algorithm as we will show below, while for the CTV regularization the most efficient algorithm so far, to our best of knowledge, is lagged diffusivity (LD) [31], which is much slower when the blurring kernel is relatively large (cf. [32]). This is because LD solves a large linear system at each iteration, which is relatively dense and ill-conditioned unless the blurring kernel is very small. The aforementioned alternating algorithm is much faster than LD because it avoids solving such a linear system. Taking into account that both MTV and CTV regularization give restoration of similar quality (cf. [1]), we prefer MTV to CTV as a regularizer.

The discretized form of (2.1) is given by

$$\mathrm{MTV}(u) = \sum_i \|(I_m \otimes D_i)u\| = \sum_i \sqrt{\|D_i u^{(1)}\|^2 + \cdots + \|D_i u^{(m)}\|^2}, \tag{2.3}$$

where $u = (u^{(1)}; \ldots; u^{(m)}) \in \mathbb{R}^{mn^2}$, I_m is the identity matrix of order m, "$\otimes$" represents the Kronecker product, and $(I_m \otimes D_i)u \in \mathbb{R}^{2m}$ is the forward finite difference at pixel i for all m channels. Given $f = [f^{(1)}; \ldots; f^{(m)}] \in \mathbb{R}^{mn^2}$, which is a blurry and noisy observation, and $K = [K_{k,l}]_{k,l=1}^m \in \mathbb{R}^{mn^2 \times mn^2}$, which is a block convolution matrix, we will recover the true image $\bar{u}$ by solving

$$\min_u \sum_i \|(I_m \otimes D_i)u\| + \frac{\mu}{2}\|Ku - f\|^2. \tag{2.4}$$

For RGB images, we let $m = 3$ in (2.4). In Section 3, we derive an alternating minimization algorithm based on a more general, locally weighted model:

$$\min_u \sum_i \alpha_i \|G_i u\| + \frac{\mu}{2}\|Ku - f\|^2, \tag{2.5}$$

where, at any pixel i, $G_i \in \mathbb{R}^{q \times mn^2}$ for some positive integer q, $\alpha_i > 0$ is a local weight and $\mu > 0$. Although μ can be removed in (2.5) by rescaling the objective function, we keep it for convenience. Clearly, (2.5) reduces to (2.4) when $G_i = I_m \otimes D_i$ and $\alpha_i \equiv 1$. It is known that locally weighted TV can better preserve image textures (cf. [26, 27]), and higher-order derivatives regularization can help reduce the so-called "staircasing" effect sometimes found with the plain TV regularization (cf. [6]). The general model (2.5) incorporates both features.

3. An alternating minimization algorithm. The image-quality advantages of TV over Tikhonov-like regularization is not without a price. The TV-like regularized problem (2.5) is computationally more

difficult to solve due to the nondifferentiability and nonlinearity of the regularization term. Despite efforts over the years, algorithms for solving the isotropic TV-like regularization model (2.5) are still much slower than those for solving Tikhonov-like regularization models. In this section, we develop our alternating minimization algorithm based on a *half–quadratic approximation* of (2.5). This algorithm makes full use of the structure of the blurring and finite-difference operators, and thus is computationally highly efficient. It significantly narrows the gap between TV-like and Tikhonov-like regularizations in terms of computational costs.

To avoid nondifferentiability caused by the regularization term in (2.5), we consider its smooth approximation problem

$$\min_u J(u) \triangleq \sum_i \alpha_i \, \phi_{\alpha_i}(G_i u) + \frac{\mu}{2}\|Ku - f\|^2, \tag{3.1}$$

where, for $\alpha > 0$ and $\beta \gg 0$, $\phi_\alpha(\cdot) : \mathbb{R}^q \to \mathbb{R}$ is an approximation to $\|\cdot\|$ in $\mathbb{R}^q$ defined by

$$\phi_\alpha(\mathbf{t}) = \begin{cases} \frac{\beta}{2\alpha}\|\mathbf{t}\|^2 & \text{if } \|\mathbf{t}\| \le \frac{\alpha}{\beta}; \\ \|\mathbf{t}\| - \frac{\alpha}{2\beta} & \text{otherwise.} \end{cases} \tag{3.2}$$

If α_i is large, which hints that it is supposed to be blocky around pixel i, (3.2) aims to regularize pixel i by quadratic function in a larger area. From the definition of $\phi_\alpha(\cdot)$ in (3.2), problem (2.5) is closely approximated by (3.1) when β is large. We will reformulate (3.1) as a *half-quadratic* problem in Subsection 3.1 and propose to solve it by the *alternating minimization algorithm* in Subsection 3.2.

3.1. Half-quadratic formulation of (3.1)**.** In this subsection, we transform (3.1) into (3.12) below using the half-quadratic technique originally introduced by Geman and Yang in [14]. Consider the following general framework of recovering an image u from its corrupted measurements f:

$$\min_u \sum_i \phi(g_i^\top u) + \frac{\mu}{2}\|Ku - f\|^2, \tag{3.3}$$

where $g_i^\top u \in \mathbb{R}$ is a local finite difference of u, $\phi(g_i^\top \cdot)$ is convex and edge-preserving, and K is a convolution operator. Instead of solving (3.3) directly, the authors of [14] (and also [13]) proposed to solve an equivalent problem

$$\min_{u,b} \sum_i \big(\psi(b_i) + Q(g_i^\top u, b_i)\big) + \frac{\mu}{2}\|Ku - f\|^2, \tag{3.4}$$

where $Q(t, s)$ and $\psi(s)$ are chosen such that $Q(t, s)$ is quadratic in t and

$$\phi(t) = \min_{s\in\mathbb{R}}(\psi(s) + Q(t, s)), \ \forall t \in \mathbb{R}. \tag{3.5}$$

The objective function in the right-hand side of (3.5) is called "half-quadratic" because it is quadratic in t, but not in s. The same applies to the objective function of (3.4) with respect to u and b. In addition, (3.4) is separable in each b_i. Since (3.4) can be solved by minimizing with respect to u and b alternately, it is important to select Q and ψ that give rise to fast minimizations with respect to u and b, respectively and separately. For this purpose, two forms of half-quadratic formulations have been widely studied: the *multiplicative* form $Q(t, s) = \frac{1}{2}t^2 s$ from [13] and the *additive* form $Q(t, s) = \frac{1}{2}(t - s)^2$ from [14]. However,

in the half-quadratic model based on the multiplicative form, the computation cost is higher because the Hessian of (3.4) with respect to u depends on b, and thus may vary from one iteration to another.

In the following, we transform (3.1) into a half-quadratic problem based on the additive form but in a generalized manner that allows b_i in (3.4) (or s in (3.5)) to be vectors . The Hessian with respect to u of the new formulation is independent of b and has a block circulant structure, which is important because such a matrix can be diagonalized by discrete Fourier transforms. As such, we need a function $\psi_\alpha(\cdot) : \mathbb{R}^q \to \mathbb{R}$ that satisfies

$$\alpha\, \phi_\alpha(\mathbf{t}) = \min_{\mathbf{s}\in\mathbb{R}^q} \left\{ \alpha\, \psi_\alpha(\mathbf{s}) + \frac{\beta}{2}\|\mathbf{s}-\mathbf{t}\|^2 \right\}. \tag{3.6}$$

Let

$$\theta_\alpha(\mathbf{t}) = \frac{1}{2}\|\mathbf{t}\|^2 - \frac{\alpha}{\beta}\phi_\alpha(\mathbf{t}) \;\text{ and }\; \vartheta_\alpha(\mathbf{s}) = \frac{\alpha}{\beta}\psi_\alpha(\mathbf{s}) + \frac{1}{2}\|\mathbf{s}\|^2. \tag{3.7}$$

Simple manipulation shows that for ϕ_α and ψ_α to satisfy (3.6), it is necessary and sufficient to have $\theta_\alpha = \vartheta_\alpha^*$ where ϑ_α^* is the conjugate of ϑ_α defined as

$$\vartheta_\alpha^*(\mathbf{s}) = \sup_{\mathbf{t}\in\mathbb{R}^q} \{\mathbf{s}^\top\mathbf{t} - \vartheta_\alpha(\mathbf{t})\}, \tag{3.8}$$

which shows how to construct ψ_α from ϕ_α through computing ϑ_α from θ_α.

LEMMA 3.1. *For $x \in \mathbb{R}^q$ and $A \in \mathbb{R}^{p\times q}$, the subdifferential of $f(x) \triangleq \|Ax\|$ is*

$$\partial f(x) = \begin{cases} \{A^\top Ax/\|Ax\|\} & \text{if } Ax \neq 0; \\ \left\{A^\top h : \|h\| \le 1, h \in \mathbb{R}^p\right\} & \text{otherwise.} \end{cases} \tag{3.9}$$

A simple proof of Lemma 3.1 can be found in [32].

LEMMA 3.2. *For $\phi_\alpha(\mathbf{t})$ defined in* (3.2) *and $\theta_\alpha(\mathbf{t})$ defined in* (3.7)*, we have*

$$\theta_\alpha^*(\mathbf{s}) = \frac{\alpha}{\beta}\|\mathbf{s}\| + \frac{1}{2}\|\mathbf{s}\|^2. \tag{3.10}$$

Proof. According to (3.8), (3.7) and (3.2), we have

$$\begin{aligned} \theta_\alpha^*(\mathbf{s}) = \sup_{\mathbf{t}} \left\{\mathbf{s}^\top\mathbf{t} - \theta_\alpha(\mathbf{t})\right\} &= \max\left\{ \sup_{\|\mathbf{t}\|\le\alpha/\beta} \mathbf{s}^\top\mathbf{t},\; \sup_{\|\mathbf{t}\|>\alpha/\beta} \left\{ \mathbf{s}^\top\mathbf{t} - \frac{1}{2}\|\mathbf{t}\|^2 + \frac{\alpha}{\beta}\|\mathbf{t}\| - \frac{\alpha^2}{2\beta^2} \right\} \right\} \\ &= \max\left\{ \frac{\alpha}{\beta}\|\mathbf{s}\|,\; \sup_{\|\mathbf{t}\|>\alpha/\beta} \left\{ \mathbf{s}^\top\mathbf{t} - \frac{1}{2}\left(\|\mathbf{t}\| - \frac{\alpha}{\beta}\right)^2 \right\} \right\} \end{aligned} \tag{3.11}$$

If $\mathbf{s} = 0$, it is obvious $\theta_\alpha^*(0) = 0$ and (3.10) holds. In what follows, we assume $\mathbf{s} \neq 0$. In light of (3.9) with A being the identity, the above supreme is attained for $\|\mathbf{t}\| > \alpha/\beta$ and $\mathbf{s} = \mathbf{t} - \alpha\mathbf{t}/(\beta\|\mathbf{t}\|)$, from which we get $\|\mathbf{s}\| = \|\mathbf{t}\| - \alpha/\beta$. Since $\mathbf{s} \neq 0$, $\|\mathbf{t}\| > \alpha/\beta$ is satisfied. Thus,

$$\mathbf{t} = \|\mathbf{t}\| \cdot \frac{\mathbf{t}}{\|\mathbf{t}\|} = \|\mathbf{t}\| \cdot \frac{\mathbf{s}}{\|\mathbf{s}\|} = \left(\|\mathbf{s}\| + \frac{\alpha}{\beta}\right)\frac{\mathbf{s}}{\|\mathbf{s}\|}.$$

Plugging the above into the inside supreme in (3.11), we get (3.10). ∎

Given Lemma 3.2, we are able to find $\psi_\alpha(\cdot)$ that satisfies (3.6) in a simple way. It is easy to check from the definitions of $\phi_\alpha(\mathbf{t})$ and $\theta_\alpha(\mathbf{t})$ in (3.2) and (3.7) that $\theta_\alpha(\mathbf{t})$ is convex. Therefore, by considering the fact that $(\theta_\alpha^*)^* = \theta_\alpha$, the requirement (3.6) or equivalently $\theta_\alpha = \vartheta_\alpha^*$ is satisfied by letting $\vartheta_\alpha = \theta_\alpha^*$. Comparing the definition of $\vartheta_\alpha(\cdot)$ with (3.10), letting $\psi_\alpha(\mathbf{s}) \equiv \|\mathbf{s}\|$ will satisfy (3.6). As such, the approximation problem (3.1) becomes an equivalent augmented problem

$$\min_{u,\mathbf{w}} \mathcal{J}(u,\mathbf{w}) = \sum_i \left\{ \alpha_i \|\mathbf{w}_i\| + \frac{\beta}{2}\|\mathbf{w}_i - G_i u\|^2 \right\} + \frac{\mu}{2}\|Ku - f\|^2, \tag{3.12}$$

where $\mathbf{w}_j \in \mathbb{R}^q$, $j = 1, \ldots, n^2$, and $\mathbf{w} = [\mathbf{w}_1;\ \mathbf{w}_2; \ldots; \mathbf{w}_{n^2}] \in \mathbb{R}^{qn^2}$. Although derived from the half-quadratic framework and the theory of conjugate functions, (3.12) is simply a splitting and penalty formulation of the original problem (2.5) which can be explained as follows. By introducing a collection of auxiliary variables $\{\mathbf{w}_i : i = 1, \ldots, n^2\}$, problem (2.5) is easily transformed into an equivalent constrained problem

$$\min_{u,\mathbf{w}} \left\{ \sum_i \alpha_i \|\mathbf{w}_i\| + \frac{\mu}{2}\|Ku - f\|^2 :\ \mathbf{w}_i = G_i u,\ i = 1, \ldots, n^2 \right\}. \tag{3.13}$$

By comparing (3.13) with (3.12), it is easy to see that (3.12) is nothing but quadratic penalty method for (3.13). Below our analysis focuses on (3.12).

3.2. Alternating minimization. Now, we are ready to apply the alternating minimization technique from [32] to (3.12). On the one hand, there is no interaction between different $\mathbf{w}_i$'s in (3.12), so minimizing with respect to $\mathbf{w}$ for fixed u reduces to solving a collection of low dimensional problems

$$\min_{\mathbf{w}_i \in \mathbb{R}^m} \alpha_i \|\mathbf{w}_i\| + \frac{\beta}{2}\|\mathbf{w}_i - G_i u\|^2,\ i = 1, \ldots, n^2. \tag{3.14}$$

On the other hand, minimizing with respect to u for fixed $\mathbf{w}$ becomes

$$\min_u \frac{\beta}{2} \sum_i \|\mathbf{w}_i - G_i u\|^2 + \frac{\mu}{2}\|Ku - f\|^2,$$

which is a least squares problem equivalent to

$$\left(\sum_i G_i^\top G_i + \frac{\mu}{\beta} K^\top K \right) u = \sum_i G_i^\top \mathbf{w}_i + \frac{\mu}{\beta} K^\top f. \tag{3.15}$$

A similar technique with Bregman iterations [21, 33] recently proposed in [15] can be applied in the same way. To simplify analysis, we introduce some additional notation. For $j = 1, \ldots, q$, let $G^{(j)} \in \mathbb{R}^{n^2 \times mn^2}$ be the matrix formed by staking the jth rows of $G_1, G_2, \ldots, G_{n^2}$. Denote

$$G \triangleq \begin{pmatrix} G^{(1)} \\ \vdots \\ G^{(q)} \end{pmatrix} \in \mathbb{R}^{qn^2 \times mn^2} \ \text{ and } \ \mathbf{W} \triangleq \begin{pmatrix} \mathbf{w}_1^\top \\ \vdots \\ \mathbf{w}_{n^2}^\top \end{pmatrix} \triangleq [w_1, w_2, \ldots, w_q] \in \mathbb{R}^{n^2 \times q}, \tag{3.16}$$

namely, $w_j \in \mathbb{R}^{n^2}$ is the jth column of $\mathbf{W}$ and formed by stacking the jth components of $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{n^2}$. Furthermore, let $w = (w_1; \ldots; w_q) = \mathbf{W}(:) \in \mathbb{R}^{qn^2}$ be the vectorization of $\mathbf{W}$ (which is just a reordering of $\mathbf{w}$). Given these notation, (3.15) can be rewritten as

$$\left(G^\top G + \frac{\mu}{\beta} K^\top K \right) u = \sum_{j=1}^q (G^{(j)})^\top w_j + \frac{\mu}{\beta} K^\top f = G^\top w + \frac{\mu}{\beta} K^\top f. \tag{3.17}$$

In what follows, we argue that the problems in (3.14) admit closed form solutions and (3.17) can also be solved easily as long as G defined in (3.16) has certain special structures.

LEMMA 3.3. *For any $\alpha, \beta > 0$ and $\mathbf{t} \in \mathbb{R}^q$, the minimizer of*

$$\min_{\mathbf{s}\in\mathbb{R}^q} \alpha\|\mathbf{s}\| + \frac{\beta}{2}\|\mathbf{s}-\mathbf{t}\|^2 \tag{3.18}$$

is

$$\mathbf{s}(\mathbf{t}) = \max\left\{\|\mathbf{t}\| - \frac{\alpha}{\beta}, 0\right\}\frac{\mathbf{t}}{\|\mathbf{t}\|}, \tag{3.19}$$

where we follow the convention $0\cdot(0/0) = 0$.

Proof. Since the objective function in (3.18) is convex, bounded below and coercive, problem (3.18) has at least one minimizer $\mathbf{s}$. According to the optimality condition for convex optimization, the subdifferential of the objective function at a minimizer should contain the origin. In light of (3.9) with $A = I$, $\mathbf{s}$ must satisfy

$$\begin{cases} \alpha\mathbf{s}/\|\mathbf{s}\| + \beta(\mathbf{s}-\mathbf{t}) = 0 & \text{if } \mathbf{s} \neq 0; \\ \beta\|\mathbf{t}\| \leq \alpha & \text{otherwise.} \end{cases} \tag{3.20}$$

If $\mathbf{s} \neq 0$, it holds $\mathbf{t} = \mathbf{s} + \alpha\mathbf{s}/(\beta\|\mathbf{s}\|)$. Hence, $\|\mathbf{t}\| = \|\mathbf{s}\| + \alpha/\beta$ and

$$\mathbf{s} = \|\mathbf{s}\|\cdot\frac{\mathbf{s}}{\|\mathbf{s}\|} = \|\mathbf{s}\|\cdot\frac{\mathbf{t}}{\|\mathbf{t}\|} = \left(\|\mathbf{t}\| - \frac{\alpha}{\beta}\right)\frac{\mathbf{t}}{\|\mathbf{t}\|}. \tag{3.21}$$

Furthermore, $\mathbf{s} = 0$ if and only if $\|\mathbf{t}\| \leq \alpha/\beta$. Combining this with (3.21), we obtain (3.19). ∎

According to Lemma 3.3, the problems in (3.14) have unique solutions given explicitly by

$$\mathbf{w}_i = \max\left\{\|G_i u\| - \frac{\alpha_i}{\beta}, 0\right\}\frac{G_i u}{\|G_i u\|},\ i = 1,\ldots,n^2. \tag{3.22}$$

Now, we show that (3.17) can also be easily solved under many circumstances. Without loss of generality, we assume that u is a RGB color image. In this case, we have $u = (u^r; u^g; u^b) \in \mathbb{R}^{3n^2}$ and $K = [K_{k,l}]_{k,l=1}^3 \in \mathbb{R}^{3n^2\times 3n^2}$. For simplicity, we assume that $G_i = I_3 \otimes D_i \in \mathbb{R}^{6\times 3n^2}$, namely, $G_i u$ only contains the first-order finite differences of u at pixel i. For higher-order finite differences, the following argument also applies. Recall that $D = (D^{(1)}; D^{(2)})$ and D_i is a two-row matrix formed by stacking the ith rows of $D^{(1)}$ and $D^{(2)}$. Following the notation in (3.16), the normal equations (3.17) reduce to

$$\left(I_3 \otimes (D^\top D) + \frac{\mu}{\beta}K^\top K\right)u = (I_3 \otimes D)^\top w + \frac{\mu}{\beta}K^\top f. \tag{3.23}$$

Under the periodic boundary conditions for u, $D^{(1)}$, $D^{(2)}$, $(D^{(1)})^\top(D^{(1)})$, $(D^{(2)})^\top(D^{(2)})$ and all block matrices $\{K_{i,j}^\top K_{k,l}, i,j,k,l = 1,2,3\}$ in $K^\top K$ are block circulant (see [19], for example). Therefore, each block in the coefficient matrix of (3.23) can be diagonalized by the two-dimensional discrete Fourier transform $\mathcal{F}$. Applying $\mathcal{F}$ to both sides of (3.23) yields

$$\begin{bmatrix} \Lambda_{11} & \Lambda_{12} & \Lambda_{13} \\ \Lambda_{21} & \Lambda_{22} & \Lambda_{23} \\ \Lambda_{31} & \Lambda_{32} & \Lambda_{33} \end{bmatrix}\begin{pmatrix} \mathcal{F}(u^r) \\ \mathcal{F}(u^g) \\ \mathcal{F}(u^b) \end{pmatrix} = \mathcal{F}(D^{(1)})^* \circ \begin{pmatrix} \mathcal{F}(w_1) \\ \mathcal{F}(w_3) \\ \mathcal{F}(w_5) \end{pmatrix} + \mathcal{F}(D^{(2)})^* \circ \begin{pmatrix} \mathcal{F}(w_2) \\ \mathcal{F}(w_4) \\ \mathcal{F}(w_6) \end{pmatrix} + \frac{\mu}{\beta}\xi, \tag{3.24}$$

where Λ_{ij}, for $i,j=1,2,3$, are all diagonal matrices, "$*$" represents the complex conjugate, "$\circ$" represents the pointwise product and ξ is a constant vector given by

$$\xi = \begin{bmatrix} \mathcal{F}(K_{11}) & \mathcal{F}(K_{12}) & \mathcal{F}(K_{13}) \\ \mathcal{F}(K_{21}) & \mathcal{F}(K_{22}) & \mathcal{F}(K_{23}) \\ \mathcal{F}(K_{31}) & \mathcal{F}(K_{32}) & \mathcal{F}(K_{33}) \end{bmatrix}^* \begin{pmatrix} \mathcal{F}(f^r) \\ \mathcal{F}(f^g) \\ \mathcal{F}(f^b) \end{pmatrix},$$

where $\mathcal{F}(K_{ij})$ is a diagonal matrix formed by the Fourier transform of the vector representing the convolution matrix K_{ij} under the periodic boundary conditions.

For a fixed $\beta > 0$, the coefficients matrix in (3.23) is a constant. Therefore, $\{\Lambda_{ij} : i,j=1,2,3\}$, $\mathcal{F}(D^{(1)})$, $\mathcal{F}(D^{(2)})$ and ξ only need to be computed once before the iterations. At each iteration, (3.23) is solved in three steps. First, two dimensional FFTs are applied to w_j, for $j=1,\ldots,6$. Second, a block diagonal system in the form of (3.24) is solved to obtain $\mathcal{F}(u^\sigma)$, for $\sigma = r,g,b$, which can be easily done by the Gaussian elimination method. Third, the inverse two dimensional FFT, $\mathcal{F}^{-1}$, is applied to $\mathcal{F}(u^\sigma)$, for $\sigma = r,g,b$, to get the updated u. The total number of FFTs required is 9.

Alternatively, under the Neumann boundary conditions and assuming that all the blurring kernels are symmetric, the forward and inverse FFTs shall be replaced by the forward and inverse Discrete Cosine Transforms, respectively (cf. [19]). In our numerical experiments, we assumed the periodic boundary conditions and used FFTs.

Now, we are ready to formally present our algorithm. For a fixed β, (3.12) is solved by an alternating minimization scheme given below.

ALGORITHM 1. *Input f, K, $\mu > 0$, $\beta > 0$ and $\alpha_i > 0$, $i = 1,\ldots,n^2$. Initialize $u = f$.*

While *"not converged,"* **Do**

1) Compute $\mathbf{w}$ *by* (3.22) *for given* u.

2) Solve (3.17) *to get* u *for given* $\mathbf{w}$.

End Do

The stopping criterion is specified in the next subsection. A practical implementation of Algorithm 1 including the setting of parameters and a continuation scheme is presented in Section 4.

3.3. Optimality conditions and stopping criterion. Since the objective function in (3.12) is convex, $(\mathbf{w},u)$ solves (3.12) if and only if the subdifferential of the objective function at $(\mathbf{w},u)$ contains the origin. This gives rise to the following optimality conditions in light of Lemma 3.1,

$$\begin{cases} \alpha_i \mathbf{w}_i/\|\mathbf{w}_i\| + \beta(\mathbf{w}_i - G_i u) = 0 & i \in I_1 \triangleq \{i : \mathbf{w}_i \neq \mathbf{0}\}, \\ \beta\|G_i u\| \leq \alpha_i & i \in I_2 \triangleq \{i : \mathbf{w}_i = \mathbf{0}\}, \end{cases} \tag{3.25}$$

$$\beta G^\top(Gu - w) + \mu K^\top(Ku - f) = 0. \tag{3.26}$$

Our stopping criterion for Algorithm 1 is based on the optimality conditions (3.25) and (3.26). Let

$$\begin{cases} r_1(i) \triangleq (\alpha_i \mathbf{w}_i/\|\mathbf{w}_i\|)/\beta + \mathbf{w}_i - G_i u & i \in I_1, \\ r_2(i) \triangleq \|G_i u\| - \alpha_i/\beta & i \in I_2, \\ r_3 \triangleq \beta G^\top(Gu - w) + \mu K^\top(Ku - f), \end{cases} \tag{3.27}$$

where I_1 and I_2 are defined in (3.25). Algorithm 1 is terminated once

$$Res \triangleq \max\left\{\max_{i\in I_1}\{\|r_1(i)\|\}, \max_{i\in I_2}\{r_2(i)\}, \|r_3\|_\infty\right\} \le \epsilon \tag{3.28}$$

is met, where Res measures the total residual and $\epsilon > 0$ is a prescribed tolerance. In (3.27), condition (3.25) is scaled by $1/\beta$, but (3.26) is not, because in practice the latter can be met quickly even without this scaling. Combining (3.25) and (3.26) to eliminate w, we can derive

$$\sum_{i\in I_1} \alpha_i G_i^\top \frac{G_i u}{\|G_i u\|} + \sum_{i\in I_2} G_i^\top h_i + \mu K^\top (Ku - f) = 0, \tag{3.29}$$

where $h_i \triangleq \beta G_i u$ satisfies $\|h_i\| \le \alpha_i$. Let u^* be any solution of (2.5). Define $I_1^* = \{i, G_i u^* \neq 0\}$ and $I_2^* = \{1, \ldots, n^2\} \setminus I_1^*$. Then, from Lemma 3.1, there exist $h_i^* \in \mathbb{R}^2$ satisfying $\|h_i^*\| \le \alpha_i$, for all $i \in I_2^*$, such that

$$\sum_{i\in I_1^*} \alpha_i G_i^\top \frac{G_i u^*}{\|G_i u^*\|} + \sum_{i\in I_2^*} G_i^\top h_i^* + \mu K^\top (Ku^* - f) = 0. \tag{3.30}$$

Equation (3.29) differs from (3.30) only in the index sets over which the summations are taken. As β increases, I_1 will approach I_1^*. In subsection 3.4, we present the convergence properties of Algorithm 1.

3.4. Convergence results. The convergence of the quadratic penalty method as the penalty parameter goes to infinity is well known (see Theorem 17.1 in [20] for example). That is, as $\beta \to \infty$, the solution of (3.1) or (3.12) converges to that of (2.5). However, in practice a sufficiently large value for β should be adequate. We will specify how to choose β empirically in Section 4. In this subsection, we present convergence results of Algorithm 1 for a fixed β without proofs since these results are rather straightforward generalizations of the results in [32] to higher dimensions. First, we present a technical assumption and necessary notation.

ASSUMPTION 1. *$\mathcal{N}(K) \cap \mathcal{N}(G) = \{0\}$, where $\mathcal{N}(\cdot)$ is the null space of a matrix.*

Define

$$M = G^\top G + \frac{\mu}{\beta} K^\top K \quad \text{and} \quad T = GM^{-1}G^\top. \tag{3.31}$$

Assumption 1 ensures that M is nonsingular, and therefore T is well defined. Clearly $\rho(T) \le 1$.

We will make use of the following two index sets:

$$L = \left\{ i \ : \ \|G_i u^*\| < \frac{\alpha_i}{\beta} \right\} \quad \text{and} \quad E = \{1, \ldots, n^2\} \setminus L. \tag{3.32}$$

Denote $w_E = ((w_1)_E; \ldots; (w_q)_E)$ where $(w_j)_E$ is the subvector of w_j with components in E. For $k, l = 1, \ldots, q$, let $B^{(k,l)} = G^{(k)} M^{-1} (G^{(l)})^\top$ and $B_{EE}^{(k,l)} = [B_{i,j}^{(k,l)}]_{i,j\in E}$ be the minor of $B^{(k,l)}$ with indices in E. From the definition of T, $T = [B^{(k,l)}]_{k,l=1}^q$. Let $T_{EE} = [B_{EE}^{(k,l)}]_{k,l=1}^q$ be a minor of T. Similar notation applies to $(T^2)_{EE}$. Now, we are ready to present the convergence results.

THEOREM 3.4 (Convergence). *Under Assumption 1, for any fixed $\beta > 0$, the sequence $\{(w^k, u^k)\}$ generated by Algorithm 1 from any starting point (w^0, u^0) converges to a solution (w^*, u^*) of (3.12).*

THEOREM 3.5 (Finite convergence). *Under Assumption 1, the sequence $\{(w^k, u^k)\}$ generated by Algorithm 1 from any starting point (w^0, u^0) satisfies $\mathbf{w}_i^k = \mathbf{w}_i^* = 0$ for all $i \in L$, for all but a finite number of iterations that does not exceed $\|w^0 - w^*\|^2/\omega^2$, where*

$$\omega \triangleq \min_{i\in L} \left\{ \frac{\alpha_i}{\beta} - \|h_i(w^*)\| \right\} > 0.$$

THEOREM 3.6 (q-linear convergence). *Let M and T be defined as in* (3.31). *Under Assumption 1, the sequence $\{(w^k, u^k)\}$ generated by Algorithm 1 satisfies*

1. $\|w_E^{k+1} - w_E^*\| \le \sqrt{\rho((T^2)_{EE})}\|w_E^k - w_E^*\|$;
2. $\|u^{k+1} - u^*\|_M \le \sqrt{\rho(T_{EE})}\|u^k - u^*\|_M$;

for all k sufficiently large, where $\|v\|_M^2 \triangleq v^\top M v$.

Theorem 3.6 states that Algorithm 1 converges q-linearly at a rate depending on the spectral radii of the submatrices T_{EE} and $(T^2)_{EE}$ rather than on that of the whole matrix T. Since $\rho(T) \le 1$ and T_{EE} is a minor of T, it holds that $\rho(T_{EE}) \le \rho(T) \le 1$. Similarly, $\rho((T^2)_{EE}) \le \rho(T^2) \le 1$. As pointed out in [32], $\rho(T_{EE})$ is often much smaller than $\rho(T)$ in practice, so our results are sharper than the general convergence results for the quadratic penalty method.

4. Numerical experiments. In this section, we present numerical results to show the efficiency of the proposed alternating minimization algorithm in recovering multichannel images. Specifically, we experimented on recovering several RGB images with different blurs and noise levels and using different regularization approaches. The first class of results concentrates on different blurs including both within-channel and cross-channel blurs. The second class shows that the proposed algorithm can solve models with different regularizations. We illustrate this by solving two additional regularization problems: the weighted TV regularization problem and the higher-order derivatives regularization problem.

4.1. Test images, platform and practical implementation. We tested several images including Lena (512×512), Rose (303×250), and Sunset (338×460). Image Lena has a nice mixture of flat regions, shading area, textures, and other details. Rose and Sunset were used in weighted TV regularization and higher-order derivatives regularization problems. We implemented Algorithm 1 in MATLAB and generated blurring effects using the MATLAB function "`imfilter`" with the periodic boundary conditions. The experiments were performed under Windows Vista Premium and MATLAB v7.0 (R14) running on a Lenovo laptop with an Intel Core 2 Duo CPU at 1.8 GHz and 1 GB of memory.

As is usually done, the quality of restoration is measured by the signal-to-noise ratio (SNR)

$$\text{SNR} \triangleq 10 * \log_{10} \frac{\|\bar{u} - \mathbf{E}(\bar{u})\|^2}{\|\bar{u} - u\|^2},$$

where $\bar{u}$ is the original image, $\mathbf{E}(\bar{u})$ is the mean intensity value of $\bar{u}$ and u is the restored image. We tested three blurring kernels: motion, Gaussian, and average. For simplicity, let the motion blur with a motion length `len` and an angle `theta` be denoted by (M, `len`, `theta`). Similarly, the Gaussian blur with a blurring size `hsize` and a standard deviation `sigma` is denoted by (G, `hsize`, `sigma`), and the average blur with a blurring size `hsize` by (A, `hsize`). Throughout the experiments, we added Gaussian noise with zero mean and different standard deviations, which are denoted by `std`. To determine a good pair of parameters (β, μ) in (3.12), we fixed $\{\alpha_i = 1, \forall i\}$ and tested our code on a series of combinations of different blurring effects. Specially, for `std`$= 10^{-3}$, we tested on the color image Rose (which has a relatively small size) with different blurs and a relatively strict stopping criterion: $\epsilon = 10^{-3}$ in (3.28). Fig. 4.1 gives the recovered SNRs for different (β, μ) values. The left-hand plot in Fig. 4.1 gives the SNRs with respect to within-channel blurs in

which we set the diagonal kernels as follows

$$\{H_{11} = (\mathrm{M}, 41,\ 135), H_{22} = (\mathrm{G}, 11,\ 9), H_{33} = (\mathrm{A}, 15)\},$$

and off-diagonal kernels $\{H_{ij} = 0 :\ i, j = 1, 2, 3;\ i \neq j\}$. The right-hand plot in Fig. 4.1 gives the SNRs with respect to cross-channel blurs. The cross-channel kernels were generated in three steps:

1. Generate 9 kernels:

$$\{(\mathrm{M}, 11,\ 45), (\mathrm{M}, 21,\ 90), (\mathrm{M}, 41,\ 135); (\mathrm{G}, 7,\ 5), (\mathrm{G}, 9,\ 5), (\mathrm{G}, 11,\ 5); (\mathrm{A}, 13), (\mathrm{A}, 15), (\mathrm{A}, 17)\};$$

2. Randomly assign the above 9 kernels to $\{H_{11}, H_{12}, H_{13}; H_{21}, H_{22}, H_{23}; H_{31}, H_{32}, H_{33}\}$;
3. Multiply 0.8 to diagonal kernels and 0.1 to off-diagonal kernels.

As can be seen from Fig. 4.1, β does not need to be extremely large for any μ in practice. Specifically, a β value of 2^7 is sufficiently large to get a SNR that is very close to the highest one for each μ (which is also the case for other noise levels). For $\beta \geq 2^7$, the SNRs almost remain constant. Hence, letting β to be larger than 2^7 will only increase computational cost but not solution quality. Therefore, we set $\beta = 2^7$ by default in Algorithm 1.

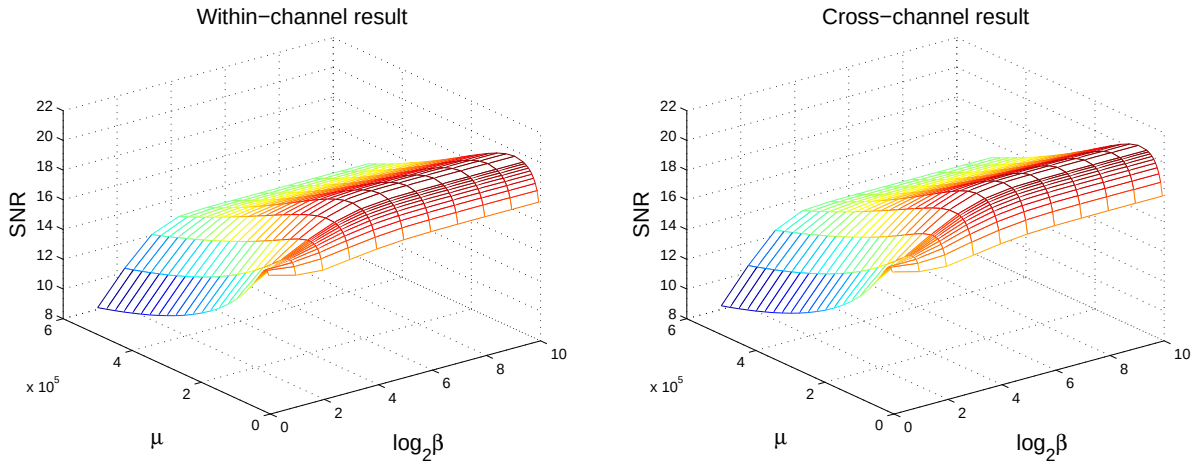

FIG. 4.1. *Within-channel (left) and Cross-channel (right) results.* $\mu = [0.1 : 10] \times 5 \times 10^4$ *and* $\beta = \{2^0, 2^1, \ldots, 2^{10}\}$.

The above approach of generating cross-channel kernels was used in all of our experiments involving cross-channel blurs. In step 2, randomness is introduced to avoid possible bias. We point out that the same set of kernels generated in step 1 may result in blurry images of different degrees because kernels may be assigned to different cross-channel positions in step 2. We have observed that the performance of our algorithm seems indifferent to such kernel configurations. In addition, the speed of our algorithm remains essentially unchanged as the kernel sizes vary (though recovery quality is clearly a function of kernel sizes). In our experiments presented in the following subsections, we tested kernels of different sizes and at different locations to validate the above assertions. In step 3, we choose weights added to the kernels to be diagonally dominant, considering that within-channel blurs are usually stronger than cross-channel ones. Similar methods for choosing kernel weights are used in the literature; see e.g., [11, 12]. Using such

diagonally dominant weights also helps avoid numerical singularity in the matrix K that had been sometimes observed when equal weights were used.

To accelerate convergence, we also implemented a continuation scheme for β in which β changes from a small value step by step to the default value as the algorithm proceeds. This continuation approach accelerates convergence according to Theorem 3.6. For more information on how this continuation scheme improves the overall convergence speed, see e.g., [32, 16]. The best choice of μ depends on the noise level `std`. For too large values of μ, the restored images will remain noisy. On the other hand, for too small values of μ, some fine details in the images will get lost. In our tests with noise level `std` being 10^{-3}, $\mu = 5 \times 10^4$ seems suitable, as can be seen from Fig. 4.1. Thus, we set $\mu = 5 \times 10^4$ by default for $\texttt{std} = 10^{-3}$. For larger noise, we first determined a good μ for the standard TV/L^2 model (2.4) based on experience and then tested it with different regularizations.

The implementation of Algorithm 1 includes two loops: the outer loop increases β from 1 to 2^7 and the inner loop solves (3.12) to a prescribed accuracy for each fixed β. We simply doubled β after every outer iteration and stopped the inner iteration when $Res \leq 0.05$ was satisfied in (3.28). Of course this algorithm framework can be more flexible in terms of updating β and stopping the inner iterations, but the above simple implementation already worked very well. Following [32], we give the name fast total variation deconvolution, or FTVd, to Algorithm 1 with the above continuation strategy.

It is interesting to observe from Fig. 4.1 that numerically there is a local maximum SNR value for $\mu = 10^4$ and $\beta = 1$. Note that here μ is only 0.2 times of the default value 5×10^4 corresponding to $\texttt{std} = 10^{-3}$. Since we changed β from 1 to a prefixed value, there will be only one outer iteration if we set the final (target) value of β to 1. The inner iteration number used is also one which can be explained as follows. From (3.22), the shrinkage quantity $1/\beta$ is large when β is small. Since we rescaled the intensity value of u into $[0, 1]$, this results to $\{\|G_i u\| \leq 1, \forall i\}$ in all our tests and therefore $\mathbf{w}_i \equiv 0$ when $\beta = 1$. As such, (3.12) reduces to the Tikhonov regularization problem

$$\min_u \|G_i u\|^2 + \mu \|Ku - f\|^2,$$

which is solved exactly by FFTs. Although the resulting SNR is a little lower than the highest SNR reachable, it is much better than the initial SNR of the blurry and noisy image. Most importantly, the CPU time consumed is much less in this case. Therefore, for $\texttt{std} = 10^{-3}$ we set $\mu = 10^4$, $\beta = 1$ and name the corresponding algorithm instance as the fast mode of FTVd or FTVd–FM.

4.2. Comparisons with MATLAB functions. In this subsection, we compare the images recovered by FTVd and FTVd–FM with those recovered by functions "`deconvreg`", "`deconvwnr`" and "`deconvlucy`" from the MATLAB Image Processing Toolbox. Currently, these are the only MATLAB functions that can be used to restore color images, and they require the within-channel blurring kernels to be identical over all channels, namely, $H_{11} = H_{22} = H_{33} \equiv H$, and that there be no cross-channel blurs. For this simple case, we tested $H =$(G, 21, 11) with $\texttt{std} = 10^{-3}$. The results on image Lena are given in Fig. 4.2.

From Fig. 4.2, FTVd gave better results in terms of both SNR and visible quality. The results of "`deconvreg`" and "`deconvwnr`" have obvious ripples. Generally, MATLAB functions were faster than FTVd because they solved much simpler models. However, FTVd–FM was not only faster than "`deconvreg`" and "`deconvlucy`" but also gave a better result. We did not present the result of "`deconvlucy`" here because

FIG. 4.2. *Comparison results with MATLAB deblurring functions.*

the algorithm spent much longer CPU time than "`deconvreg`" and "`deconvwnr`" just to yield an image of a similar quality.

4.3. Cross-channel results. In this subsection, we present the experimental results recovered by FTVd from cross-channel blurred images. The cross-channel kernels were generated in exactly the same way as in subsection 4.1 except the 9 kernels used here are:

$$\{(\mathrm{M}, 21,\ 45), (\mathrm{M}, 41,\ 90), (\mathrm{M}, 61,\ 135); (\mathrm{G}, 11,\ 9), (\mathrm{G}, 21,\ 11), (\mathrm{G}, 31,\ 13); (\mathrm{A}, 13), (\mathrm{A}, 15), (\mathrm{A}, 17)\}.$$

Observe that the above blurs in all three channels are quite severe. The noise level was still $\texttt{std} = 10^{-3}$. To the best of our knowledge, problem (2.4) with the MTV replaced by the "color-TV" is usually solved by the LD method. As pointed out in [32], for yielding similar or better results, FTVd is faster than LD especially for larger blurring kernels. Therefore, we did not compare with LD. The recovered Lena images are given in Fig. 4.3.

As can be seen from Fig. 4.3, FTVd with the default settings produced an image with an SNR value of 21.07dB. The image recovered by FTVd–FM with much shorter time, although has a lower SNR, is also visually acceptable.

4.4. Weighted TV restoration results. We tested our algorithm on a weighted TV/L^2 model, which enhances the reconstruction of (2.4) by preserving sharp edges better. At pixels near color discontinuities, we penalize the relevant pixels less by assigning relatively small weights in TV. Specifically, for RGB images we set $G_i = I_3 \otimes D_i$ and determined α_i in (2.5) by

$$\gamma_i = \frac{1}{1 + \tau \|G_i \tilde{u}\|} \quad \text{and} \quad \alpha_i = \frac{n^2 \gamma_i}{\sum_j \gamma_j}, \tag{4.1}$$

FIG. 4.3. *Results recovered by FTVd–FM and FTVd from cross-channel blurring.*

where $\tilde{u}$ is an estimation of the original image and $\tau > 0$ is determined based on experiments. As an empirical formula, (4.1) may be far from optimal but is sufficient for our purpose here which is to illustrate that FTVd can efficiently solve the weighted model (2.5). In this test, we set $\texttt{std} = 10^{-2}$ and $\mu = 10^3$. The cross-channel kernels were generated in the same manner as in subsection 4.1 except that the 9 kernels used here were:

$$\{(\mathrm{M}, 21,\ 45), (\mathrm{M}, 41,\ 90), (\mathrm{M}, 61,\ 135); (\mathrm{G}, 7,\ 5), (\mathrm{G}, 9,\ 5), (\mathrm{G}, 11,\ 5); (\mathrm{A}, 13), (\mathrm{A}, 15), (\mathrm{A}, 17)\}.$$

Compared with the 9 kernels used in subsection 4.3, we used Gaussian blurs of smaller sizes because the noise level was higher, while using motion and average blurs of the same sizes as in subsection 4.3. We solved the unweighted (original MTV) model (2.4) followed by the weighted model (2.5) in which the weights $\{\alpha_i, \forall\, i\}$ are determined by formula (4.1) using the solution $\tilde{u}$ of the unweighted model and $\tau = 15$ in this test. The results on image Rose are given in Fig. 4.4.

FIG. 4.4. *Numerical results of weighted TV/L^2. Original (left, 303×250); Blurry and noisy (middle left); Results of unweighted TV/L^2 (middle right); Results of weighted TV/L^2 (right).*

From Fig. 4.4, the image reconstructed by the weighted TV/L^2 model has both higher SNR and better visual quality than that by the unweighted model. The little drops of water on the flower are clearer in the weighted restoration result. When the "correct" weights were used, namely, the weights computed by (4.1) using the original image as $\tilde{u}$, we obtained an image (which was not depicted in Fig. 4.4) with SNR=16.72dB. Therefore, a better choice of weights can help improve the restoration quality, as is expected.

4.5. Higher-order derivatives regularization. In this subsection, we present images reconstructed by solving regularization models based on higher-order derivatives. We tested with the RGB image Sunset (338×460) and set $G_i = I_3 \otimes \mathcal{D}_i$, where $\mathcal{D}_i$ is the matrix that computes first and second order forward finite differences at pixel i, namely, $\mathcal{D}_i u^{(j)} \in \mathbb{R}^6$, $j = r, g, b$, is the vector consisting of the two first-order forward finite differences (approximating u_x and u_y, respectively) and the four second-order finite differences (approximating u_{xx}, u_{xy}, u_{yx} and u_{yy}, respectively, where $u_{xy} = u_{yx}$) of $u^{(j)}$ at pixel i. Since the weighted model gives better results than the unweighted one, we computed the weighted higher-order model, called HTV/L^2. In all, the problem we solved is

$$\min_u \sum_i \alpha_i \|(I_3 \otimes \mathcal{D}_i)u\| + \frac{\mu}{2}\|Ku - f\|^2.$$

In order to make the staircasing effect of TV model visible, in this test we used even larger additive noise with `std` $= 0.1$. On the other hand, to maintain a good recovery quality, we chose to use smaller sizes for blurring kernels. The nine kernels used in this experiment were

$$\{(\mathrm{M}, 3, 0), (\mathrm{M}, 3, 0), (\mathrm{M}, 3, 0); (\mathrm{G}, 3, 0.5), (\mathrm{G}, 3, 0.5), (\mathrm{G}, 3, 0.5); (\mathrm{A}, 3), (\mathrm{A}, 3), (\mathrm{A}, 3)\}.$$

The weights were generated according to (4.1) using the original image $\tilde{u}$. We set $\mu = 12.5$ and $\tau = 15$. The results on image Sunset are given in Fig. 4.5.

Comparing the middle two images in Fig. 4.5, the right-hand image produced by the HTV/L^2 model has cleaner sky and less staircasing in the clouds. The reduction of staircasing effects can be seen by comparing the two zoom-in images on the bottom of Fig. 4.5.

4.6. Computational cost of FTVd and a note. In the framework of Algorithm 1, there are two steps at each iteration. The first step applies the weighted shrinkage operation in (3.22) and has a linear computational complexity. The second step solves a system of linear equations by calling FFT and Gaussian elimination. Note that the computational cost of Gaussian elimination in solving block diagonalized equations such as (3.24) is very small compared with that of FFTs since no fillings occur and no pivoting is necessary. Therefore, the per-iteration computation cost of Algorithm 1 is dominated by that on the FFTs, each costing $O(n^2 \log(n))$. The remaining question is what is the total number of inner iterations needed for Algorithm 1 to attain a required accuracy and how this number varies with the image size. In our experiments, this number for the current implementation of FTVd using the default parameters is almost always around 12. For each $\beta > 1$, since Algorithm 1 has a good starting point from the previous outer iteration, it only takes 1 or 2 inner iterations on average to reach the prescribed accuracy. Given that nine FFTs are required at each inner iteration, the total numbers of FFTs taken by FTVd is around 120 for MTV regularized problem. If higher-order derivatives are involved in regularization, more FFTs will be needed for additional auxiliary vectors. Furthermore, since Algorithm 1 does not require any matrix operations, it is numerically stable and insensitive to ill-conditioning.

Recently, a new splitting algorithm is proposed in [17], which also solves (2.4). The augmented problem formed in [17] can be written in the form of

$$\min_u \sum_i \|(I_3 \otimes D_i)v\| + \alpha\|v - u\|^2 + \frac{\mu}{2}\|Ku - f\|^2, \tag{4.2}$$

FIG. 4.5. *Numerical results of weighted high order regularization. Original (upper left); Blurry and noisy (upper right); Weighted TV/L^2 result (middle left); Weighted higher-order result (middle right); Zoom in the results of weighted TV/L^2 (bottom left) and HTV/L^2 (bottom right).*

and was solved by alternating minimizations with respect to u and v. For a fixed v, the minimization with respect to u involves six FFTs. However, for a fixed u, the minimization problem with respect to v is a TV denoising problem that does not have a closed-form solution. In [17], the TV denoising problem is solved iteratively by extended Chambolle's projection algorithm [5]. While the per-iteration computational complexity of our method is dominated by nine FFTs, that of [17] is dominated by the cost of solving a TV denoising problem in addition to six FFTs. According to the reported numerical results in [17], their algorithm appears to require at least as many outer iterations as ours.

5. Conclusion remarks. In this paper, we derived an alternating minimization algorithm for deblurring multichannel (color) images based on a general variational model that includes total variation regularization as a special case. In this model, the blurs can take place both within and cross channels, and the edge-preserving regularization terms can be locally weighted and use higher-order derivatives of images. The algorithm possesses strong convergence properties and, more importantly, is practically efficient. The fast speed of the algorithm is the result of exploiting problem structures to enable the use of multi-dimensional shrinkage and fast transforms in solving subproblems. Our numerical experiments confirm that, with the help of a continuation scheme, a simple MATLAB implementation of our algorithm already achieves a remarkable practical performance.

Acknowledgments. The work of J. Yang has been supported by the Chinese Scholarship Council during his visit to Rice University. The work of W. Yin has been supported in part by NSF CAREER Grant DMS-0748839. The work of Y. Zhang has been supported in part by NSF Grant DMS-08yyyyy. The work of Y. Wang has been supported by the second author's internal faculty research grant from the Dean of Engineering at Rice University.

REFERENCES

[1] L. Bar, A. Brook, N. Sochen and N. Kiryati, *Deblurring of color images corrupted by impulsive noise*, IEEE Trans. Image Process., vol. 16, no. 4, pp. 1101–1110, 2007.

[2] P. Blomgren and T. F. Chan, *Color TV: Total variaiton methods for restoration of vector-valued images*, IEEE Trans. Image Process., vol. 7, no. 3, pp. 304–309, 1998.

[3] K. Boo and N. K. Bose, *Multispectral image restoration with multisensors*, IEEE Trans. Geosci. Remote Sensing, vol. 35, pp. 1160–1170, 1997.

[4] A. Brook, R. Kimmel, and N. Sochen, *Variational restoration and edge detection for color images*, J. Math. Imag. Vis., vol. 18, pp. 247–268, 2003.

[5] A. Chambolle, *An algorithm for total variation minimization and applications*, J. Math. Imag. Vis., vol. 20, pp. 89–97, 2004.

[6] A. Chambolle and P. L. Lions, *Image recovery via total variation minimization and related problems*, Numer. Math., vol. 76, pp. 167–188, 1997.

[7] T. F. Chan, S. Esedoglu, F. Park, and A. Yip, *Recent developments in total variation image restoration*, CAM Report 05-01, Department of Mathematics, UCLA, 2004.

[8] T. F. Chan, S. H. Kang, and J. Shen, *Total variation denoising and enhancement color images based on the CB and HSV color models*, Journal of Visual Communication and Image Representation, vol. 12, no. 4, 422–435 June 2001.

[9] T. F. Chan, and J. Shen, *Variational restoration of non-flat image features: Models and algorithms*, SIAM J. Appl. Math. vol. 61, pp. 1338-1361, 2000.

[10] D. C. Dobson and F. Santosa, *Recovery of blocky images from noisy and blurred data*, SIAM J. Appl. Math., vol. 56, pp. 1181–1198, 1996.

[11] H. Y. Fu, M. K. Ng, and J. L. Barlow, *Structured total least squares for color image restoration*, SIAM J. Sci. Comput., vol. 28, no. 3, pp. 1100–1119, 2006.

[12] N. P. Galatsanos, A. K. Katsaggelos, R. T. Chan, and A. D. Hillery, *Least squares restorations of multichannel images*, IEEE Trans. Signal Process., vol. 39, pp. 2222–2236, 1991.

[13] D. Geman and G. Reynolds, *Constrained restoration and the recovery of discontinuities*, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 14, no. 3, pp. 367–383, 1992.

[14] D. Geman and C. Yang, *Nonlinear image recovery with half-quadratic regularization*, IEEE Trans. Image Process., vol. 4, pp. 932–946, 1995.

[15] T. Goldstein and S. Osher, *The split Bregman algorithm for L1 regularized problems*, UCLA CAM Report 08-29, 2008.

[16] E. Hale, W. Yin, and Y. Zhang, *Fixed-point continuation for L1 minimization: methodology and convergence*, Tech. Report 07-07, CAAM, Rice University, 2007.

[17] Y. M. Huang, M. K. Ng, and Y. W. Wen, *Efficient total variation minimization methods for color image restoration*, submitted.

[18] M. K. Ng, and N. K. Bose, *Fast color image restoration with multisensors*, International Journal of Imaging Systems and Technology, vol. 12, pp. 189–197, 2002.

[19] M. K. Ng, R. H. Chan, and W. Tang, *A fast algorithm for deblurring models with neumann boundary conditions*, SIAM J. Sci. Comput., vol. 21, no. 3, pp. 851–866, 1999.

[20] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer, 2000.

[21] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, *An iterative regularization method for total variation based image restoration*, SIAM J. Multiscale Model. Simul., 4 (2005), pp. 460–489.

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

[23] G. Sapiro, *Color snakes*, Tech. Rep. HPL-95-113, Hewlett Packard Comput. periph. Lab., 1995.

[24] G. Sapiro, *Vector-valued active contours*, Proc. Conf. Computer Vision and Pattern Recognition, IEEE Computer Society, pp. 520–525, 1996.

[25] G. Sapiro and D. L. Ringach, *Anisotropic diffusion of multivalued images with applications to color filtering*, IEEE Trans. Image Process., vol. 5, pp. 1582–1586, 1996.

[26] D. Strong and T. F. Chan, *Relation of regularization parameter and scale in total variation based image denoising*, UCLA CAM Report 96–7, University of California, Los Angeles, CA, 1996.

[27] D. Strong, P. Blomgren and T. F. Chan, *Spatially adaptive local feature-driven total variation minimizing image restoration*, UCLA CAM Report 97–32, University of California, Los Angeles, CA, 1997.

[28] B. Tang, G. Sapiro, and V. Caselles, *Color image enhancement via chromaticity diffusion*, Tech. Report, Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN, 1999.

[29] A. Tekalp, and G. Pavlovic, *Multichannel image modeling and kalman filtering for multispectral image restoration*, Signal Process., vol. 19, pp. 221–232, 1990.

[30] A. Tikhonov, and V. Arsenin, *Solution of ill-posed problems*, Winston, Washington, DC, 1977.

[31] C. R. Vogel and M. E. Oman, *Iterative methods for total variation denoising*, SIAM J. Sci. Comput., vol. 17, no. 1, pp. 227–238, 1996.

[32] Y. Wang, J. Yang, W. Yin and Y. Zhang, *A new alternating minimization algorithm for total variation image reconstruction*, Accepted by SIAM Journal on Imaging Sciences, 2008.

[33] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, *Bregman iterative algorithms for compressed sensing and related problems*, SIAM J. Imag. Sciences 1 (2008), pp. 143–168.

[34] W. P. Ziemer, *Weakly differentiable functions: Sobolev spaces and functions of bounded variation, graduate texts in mathematics*, Springer, 1989.