Research Article

Mathematical Methods in the Applied Sciences

Received 21 June 2015

Published online in Wiley Online Library

(wileyonlinelibrary.com) DOI: 10.1002/mma.3858
MOS subject classification: 35A15; 49M37

Implementation of high-order variational models made easy for image processing

Wenqi Lu[a], Jinming Duan[b], Zhaowen Qiu[c*†], Zhenkuan Pan[a], Ryan Wen Liu[d] and Li Bai[b]

Communicated by S. Wise

High-order variational models are powerful methods for image processing and analysis, but they can lead to complicated high-order nonlinear partial differential equations that are difficult to discretise to solve computationally. In this paper, we present some representative high-order variational models and provide detailed descretisation of these models and numerical implementation of the split Bregman algorithm for solving these models using the fast Fourier transform. We demonstrate the advantages and disadvantages of these high-order models in the context of image denoising through extensive experiments. The methods and techniques can also be used for other applications, such as image decomposition, inpainting and segmentation. Copyright © 2016 John Wiley & Sons, Ltd.

Keywords: image processing; total variation; bounded Hessian; total generalised variation; total curvature; split Bregman algorithm

1. Introduction

Variational methods have played an important role in many disciplines including quantum mechanics, engineering, statistics and economics. The basis of variational methods is the variational principle used in the calculus of variations, a field of mathematical analysis that deals with maximizing or minimizing functionals. These functionals are often expressed as integrals involving functions and their derivatives, and thus, variational methods involve solving partial differential equations (PDEs). In the context of image processing, a functional normally has a data term (or similarity term) and a regulariser term. The former allows the similarity between the restored image and the original image, whilst the latter imposes some kind of smoothness on the restored image involving first or/and second derivative.

In recent two decades, numerous variational models [1–4], have been applied to image processing. One of the most popular variational models for image processing is the total variation (TV) model proposed by Rudin Osher and Fatemi [5]. The energy functional of the model is as follows:

$$E(u) = \frac{1}{2}\int_{\Omega}(u-f)^2 + \alpha\int_{\Omega}|\nabla u|, \tag{1.1}$$

where $f:\Omega \to \mathbb{R}$ is a given noisy image defined on a rectangle space Ω and penalty parameter α imposed on the TV regulariser term controls the level of smoothness of resulting clean image.

This first-order TV model is effective in preserving object edges while removing noise. A desirable property of image-denoising algorithms is to restore images without losing important features (i.e. edges, corners, contrast etc.). However, this model has several undesirable side effects, one of which is the staircase effect, that is, the restored image appears jagged. High-order variational models have been proposed to eliminate the staircase effect.

In this paper, detailed implementation of several high-order variational models for image processing are given. These high-order models are briefly introduced later.

[a] *College of Information Engineering, Qingdao University, China*
[b] *School of Computer Science, University of Nottingham, UK*
[c] *Institute of Information Computer Engineering, NorthEast Forestry University, China*
[d] *School of Navigation, Wuhan University of Technology, China*
[*] *Correspondence to: Zhaowen Qiu, Institute of Information Computer Engineering, NorthEast Forestry University, China.*
[†] *E-mail: qiuzw@nefu.edu.cn*

The simplest high-order models involve only second-order derivatives. For instance, a second-order model [6], named the TL (Total Laplace) model in this paper, used the second-order Laplacian operator:

$$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega |\Delta u|, \tag{1.2}$$

where the laplace is defined as $\Delta u = \partial_x \partial_x u + \partial_y \partial_y u$.

A second-order model uses the Bounded Hessian (BH) regulariser, named BH model in this paper, and was proposed by Lysaker *et al.* [7], Scherzer *et al.* [8,9], Lai *et al.* [10], and Bergounioux *et al.* [11]. The model minimises the following energy functional

$$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega |\nabla^2 u|, \tag{1.3}$$

where the Hessian matrix of u (i.e. $\nabla^2 u$) in the image domain Ω is given as follows:

$$\nabla^2 u = \begin{pmatrix} \partial_x \partial_x u & \partial_y \partial_x u \\ \partial_x \partial_y u & \partial_y \partial_y u \end{pmatrix} \tag{1.4}$$

A model, which combines TV regluriser with Laplacian regluriser, named the CEP2-L^2 model in this paper, was proposed in [12]:

$$E(u_1,u_2) = \frac{1}{2}\int_\Omega (f-u_1-u_2)^2 + \alpha \int_\Omega |\nabla u_1| + \beta \int_\Omega |\Delta u_2|, \tag{1.5}$$

where positive penalty parameters α and β balance the first and second derivatives.

A model, which combines TV and Laplacian reglurisers, named TV and Laplacian model (TVL) in this paper, was proposed in [13–15]:

$$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega |\nabla u| + \beta \int_\Omega |\Delta u|. \tag{1.6}$$

A model, which combines TV and Bounded Hessian reglurisers, named the TVBH in this paper, was proposed in [16,17]:

$$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega |\nabla u| + \beta \int_\Omega \left|\nabla^2 u\right|. \tag{1.7}$$

A model, which uses infimal-convolution of the first-order TV regulariser and BH regulariser, named INFCON in this paper, was proposed in [18]. The TV regulariser preserves the edges and the BH keeps the image smooth. The INFCON model is given as

$$E(u_1,u_2) = \frac{1}{2}\int_\Omega (f-u_1-u_2)^2 + \alpha \int_\Omega |\nabla u_1| + \beta \int_\Omega \left|\nabla^2 u_2\right|. \tag{1.8}$$

More recently, the total generalized variation (TGV) model was proposed in [19] for image denosing. The main feature of TGV is that it can deal with different image characteristics (piecewise constant, piecewise affine, piecewise quadratic etc.). The TGV model is of the following form:

$$E\,(u,p) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega |\nabla u - p| + \beta \int_\Omega |\varepsilon(p)|), \tag{1.9}$$

where $\varepsilon(p)$ is the symmetrised derivative and defined as

$$\varepsilon(p) = \begin{pmatrix} \partial_x p_1 & \frac{\partial_y p_1 + \partial_x p_2}{2} \\ \frac{\partial_y p_1 + \partial_x p_2}{2} & \partial_y p_2 \end{pmatrix}. \tag{1.10}$$

The minimum of (1.9) is taken over all gradients of the deformation field $p = (p_1, p_2)$ on image space Ω.

Finally, high-order variational model using the total curvature (TC) regulariser was proposed in [20–23] for image denoising, which uses the mean curvature of the image surface as a regulariser. In fact, the curvature-based models were first introduced by Nitzberg and Mumford [24] in early 1990s and then widely extended for image inpainting [25–27], segmentation with depth [28], construction of illusory shapes [29] etc. The TC model is given as

$$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \left| div\left(\frac{\nabla u}{|\nabla u|}\right)\right|. \tag{1.11}$$

Despite of the effectiveness of these high-order models in removing the staircase effect, it is often a challenging problem to minimise the corresponding functionals. Technically, the Euler equations of the regulariser are fourth order nonlinear partial differential equations (PDEs), which are very difficult to discretise to solve computationally. In this paper, split Bregman algorithm [30–32] is applied to transform the energy minimization problem of given model into several subproblems. These subproblems are then efficiently solved by fast Fourier transform (FFT) [22, 33–36], analytical soft thresholding equation [37] and projection formula [22, 36] without any iteration.

The rest of the paper is organised as follows: Section 2 introduces detailed discretisation of first-order, second-order and fourth-order differential operators using finite difference scheme. Based on Section 2, the fast split Bregman algorithm is designed for all the mentioned models in Section 3. In section 4, we present extensive comparative experiments. Section 5 draws a conclusion about the advantages and disadvantages of these models.

2. Discretisation

In order to solve previous variational models using efficient split Bregman method [30, 38, 39], the discretisation of first-order, second-order and fourth-order derivative operators are required. They are respectively ∂_x^+, ∂_x^-, ∂_y^+, ∂_y^-, $\partial_x^-\partial_x^+$, $\partial_x^+\partial_x^-$, $\partial_y^-\partial_y^+$, $\partial_y^+\partial_y^-$, $\partial_y^+\partial_x^+$, $\partial_x^+\partial_y^+$, $\partial_y^-\partial_x^-$, $\partial_x^-\partial_y^-$, $\partial_x^+\partial_y^-$, $\partial_y^+\partial_x^-$, $\partial_x^+\partial_x^-\partial_x^-\partial_x^+$, $\partial_y^+\partial_y^-\partial_y^-\partial_y^+$, $\partial_y^-\partial_x^-\partial_y^+\partial_x^+$ and $\partial_x^-\partial_y^-\partial_x^+\partial_y^+$. The gradient, Hessian matrix (1.4) and symmetrised derivative (1.10) can be discretised as follows, respectively:

$$\nabla u = \begin{pmatrix} \partial_x^+ u & \partial_y^+ u \end{pmatrix} \tag{2.1}$$

$$\nabla^2 u = \begin{pmatrix} \partial_x^-\partial_x^+ u & \partial_y^+\partial_x^+ u \\ \partial_x^+\partial_y^+ u & \partial_y^-\partial_y^+ u \end{pmatrix} \tag{2.2}$$

$$\varepsilon(p) = \begin{pmatrix} \partial_x^- p_1 & \frac{\partial_y^- p_1 + \partial_x^- p_2}{2} \\ \frac{\partial_y^- p_1 + \partial_x^- p_2}{2} & \partial_y^- p_2 \end{pmatrix}. \tag{2.3}$$

Theoretically, $\partial_x\partial_y u$ should be equal to $\partial_y\partial_x u$ in Hessian matrix (1.4), so it is necessary to define its discrete form such that $\partial_x^+\partial_y^+ u = \partial_y^+\partial_x^+ u$ in (2.2). The issue of boundary conditions is also needed addressing, as it is very important in numerical optimisation. Unlike the Neumann boundary conditions used in [40–42], we apply periodic boundary conditions. By choosing such conditions, FFT can be conveniently utilised for the split Bregman algorithm to improve computational speed.

The first-order forward and backward difference schemes are first given. Let $\Omega \rightarrow \mathbb{R}^{M\times N}$ denote the two-dimensional grey scale image space with size MN. The coordinates x and y are oriented along columns and rows, respectively. So the first-order forward differences of u at point (i, j) along x and y directions are respectively

$$\partial_x^+ u_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } 1 \le i \le M,\ 1 \le j < N \\ u_{i,1} - u_{i,j} & \text{if } 1 \le i \le M,\ j = N \end{cases} \tag{2.4}$$

$$\partial_y^+ u_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } 1 \le i < M,\ 1 \le j \le N \\ u_{1,j} - u_{i,j} & \text{if } i = M,\ 1 \le j \le N \end{cases}. \tag{2.5}$$

The first order backward differences are respectively

$$\partial_x^- u_{i,j} = \begin{cases} u_{i,j} - u_{i,j-1} & \text{if } 1 \le i \le M,\ 1 < j \le N \\ u_{i,j} - u_{i,N} & \text{if } 1 \le i \le M,\ j = 1 \end{cases} \tag{2.6}$$

$$\partial_y^- u_{i,j} = \begin{cases} u_{i,j} - u_{i-1,j} & \text{if } 1 < i \le M,\ 1 \le j \le N \\ u_{i,j} - u_{M,j} & \text{if } i = 1\ ,\ 1 \le j \le N \end{cases}. \tag{2.7}$$

For every $p = (p_1\ \ p_2) \in \left(\mathbb{R}^{M\times N}\right)^2$ and $u \in \mathbb{R}^{M\times N}$, the discrete version of the first-order adjoint divergence operator satisfies

$$\sum_{\substack{1\le i\le M\\ 1\le j\le N}} -div(p)\cdot u = \sum_{\substack{1\le i\le M\\ 1\le j\le N}} p\cdot\nabla u.$$

Therefore, according to the definition of discrete gradient operator, it is easy to check the discrete divergence as

$$div\left(p_{i,j}\right) = \partial_x^- p_{1i,j} + \partial_y^- p_{2i,j}. \tag{2.8}$$

The discrete second-order derivatives $\partial_x^-\partial_x^+ u$, $\partial_x^+\partial_x^- u$, $\partial_y^-\partial_y^+ u$, $\partial_y^+\partial_y^- u$, $\partial_y^+\partial_x^+ u$, $\partial_x^+\partial_y^+ u$, $\partial_y^-\partial_x^-$ and $\partial_x^-\partial_y^-$ at point (i, j) can be written down as follows, respectively. These operators are just the corresponding compositions of the discrete first-order derivative.

$$\partial_x^+\partial_x^- u_{i,j} = \partial_x^-\partial_x^+ u_{i,j} = \begin{cases} u_{i,N} - 2u_{i,j} + u_{i,j+1} & \text{if } 1 \le i \le M,\ j = 1 \\ u_{i,j-1} - 2u_{i,j} + u_{i,j+1} & \text{if } 1 \le i \le M,\ 1 < j < N \\ u_{i,j-1} - 2u_{i,j} + u_{i,1} & \text{if } 1 \le i \le M,\ j = N \end{cases} \tag{2.9}$$

$$\partial_y^+\partial_y^- u_{i,j} = \partial_y^-\partial_y^+ u_{i,j} = \begin{cases} u_{M,j} - 2u_{i,j} + u_{i+1,j} & \textit{if } i = 1,\ 1 \le j \le N \\ u_{i-1,j} - 2u_{i,j} + u_{i+1,j} & \textit{if } 1 < i < M,\ 1 \le j \le N \\ u_{i-1,j} - 2u_{i,j} + u_{1,j} & \textit{if } i = M,\ 1 \le j \le N \end{cases} \tag{2.10}$$

$$\partial_x^+\partial_y^+ u_{i,j} = \partial_y^+\partial_x^+ u_{i,j} = \begin{cases} u_{i,j} - u_{i+1,j} - u_{i,j+1} + u_{i+1,j+1} & \textit{if } 1 \le i < M,\ 1 \le j < N \\ u_{i,j} - u_{1,j} - u_{i,j+1} + u_{1,j+1} & \textit{if } i = M,\ 1 \le j < N \\ u_{i,j} - u_{i+1,j} - u_{i,1} + u_{i+1,1} & \textit{if } 1 \le i < M,\ j = N \\ u_{i,j} - u_{1,j} - u_{i,1} + u_{1,1} & \textit{if } i = M,\ j = N \end{cases} \tag{2.11}$$

$$\partial_x^-\partial_y^- u_{i,j} = \partial_y^-\partial_x^- u_{i,j} = \begin{cases} u_{i,j} - u_{i,N} - u_{M,j} + u_{M,N} & \textit{if } i = 1,\ j = 1 \\ u_{i,j} - u_{i,j-1} - u_{M,j} + u_{M,j-1} & \textit{if } i = 1,\ 1 < j \le N \\ u_{i,j} - u_{i,N} - u_{i-1,j} + u_{i-1,N} & \textit{if } 1 < i \le M,\ j = 1 \\ u_{i,j} - u_{i,j-1} - u_{i-1,j} + u_{i-1,j-1} & \textit{if } 1 < i \le M,\ 1 < j \le N \end{cases} . \tag{2.12}$$

Note that the constraint $\partial_x\partial_y u = \partial_y\partial_x u$ in Hessian matrix (1.4) is satisfied as now the discrete form $\partial_x^+\partial_y^+ u_{i,j} = \partial_y^+\partial_x^+ u_{i,j}$ in (2.11).

Based on (2.9) and (2.10), the definition of discrete Laplace operator is given as

$$\Delta u_{i,j} = div(\nabla u_{i,j}) = \partial_x^-\partial_x^+ u_{i,j} + \partial_y^-\partial_y^+ u_{i,j}. \tag{2.13}$$

In addition, we need the discrete second-order divergence operator that also has adjointness property: for every $q = \begin{pmatrix} q_1 & q_2 \\ q_3 & q_4 \end{pmatrix} \in \left(\mathbb{R}^{M\times N}\right)^4$ and $u \in \mathbb{R}^{M\times N}$, we have $\sum_{\substack{1\le i\le M \\ 1\le j\le N}} div^2(q)\cdot u = \sum_{\substack{1\le i\le M \\ 1\le j\le N}} q \cdot \nabla^2 u$. Then the discrete second-order divergence reads

$$div^2\left(q_{i,j}\right) = \partial_x^+\partial_x^- q_{1i,j} + \partial_y^-\partial_x^- q_{2i,j} + \partial_x^-\partial_y^- q_{3i,j} + \partial_y^+\partial_y^- q_{4i,j}. \tag{2.14}$$

In order to implement these high-order models, we give another two definitions of the discrete second-order derivatives

$$\partial_x^+\partial_y^- u_{i,j} = \begin{cases} u_{i.j+1} - u_{i,j} - u_{M,j+1} + u_{M,j} & \textit{if } i = 1,\ 1 \le j < N \\ u_{i.1} - u_{i,j} - u_{M,1} + u_{M,j} & \textit{if } i = 1,\ j = N \\ u_{i,j+1} - u_{i,j} - u_{i-1,j+1} + u_{i-1,j} & \textit{if } 1 < i \le M,\ 1 \le j < N \\ u_{i,1} - u_{i,j} - u_{i-1,1} + u_{i-1,j} & \textit{if } 1 < i \le M,\ j = N \end{cases} \tag{2.15}$$

$$\partial_y^+\partial_x^- u_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} - \ u_{i,N} + u_{i,N} & \textit{if } 1 \le i < M,\ j = 1 \\ u_{1,j} - u_{i,j} - u_{1,N} + u_{i,N} & \textit{if } i = M,\ j = 1 \\ u_{i+1,j} - u_{i,j} - u_{i+1,j-1} + u_{i,j-1} & \textit{if } 1 \le i < M,\ 1 < j \le N \\ u_{1,j} - u_{i,j} - u_{1,j-1} + u_{i,j-1} & \textit{if } i = M,\ 1 < j \le N \end{cases} \tag{2.16}$$

and the discrete fourth order derivatives

$$\partial_x^-\partial_x^+\partial_x^-\partial_x^+ u_{i,j} = \partial_x^+\partial_x^-\partial_x^-\partial_x^+ u_{i,j} = 6u_{i,j} + u_{i,j+2} + u_{i,j-2} - 4u_{i,j+1} - 4u_{i,j-1} \tag{2.17}$$

$$\partial_y^-\partial_y^+\partial_y^-\partial_y^+ u_{i,j} = \partial_y^+\partial_y^-\partial_y^-\partial_y^+ u_{i,j} = 6u_{i,j} + u_{i+2,j} + u_{i-2,j} - 4u_{i+1,j} - 4u_{i-1,j} \tag{2.18}$$

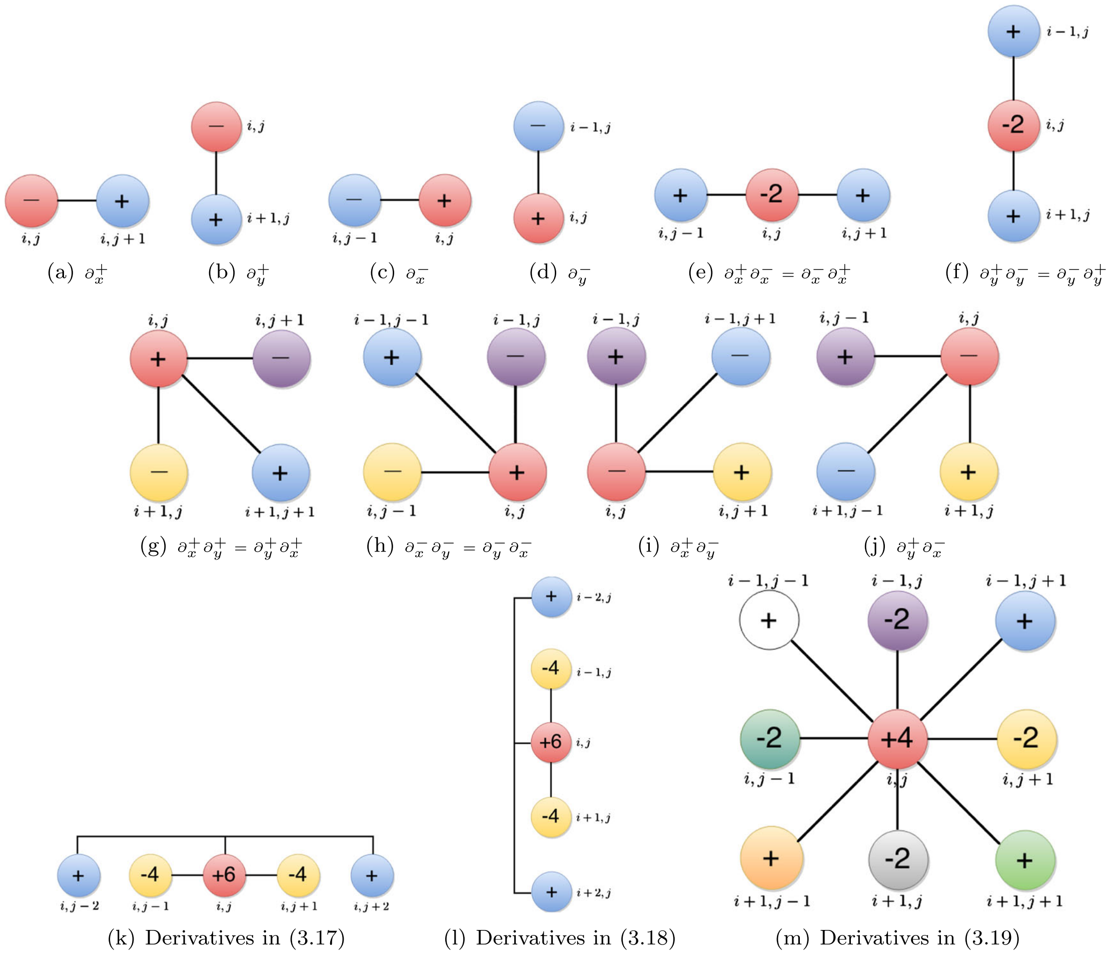
$$\begin{aligned} \partial_x^-\partial_x^+\partial_y^-\partial_y^+ u_{i,j} &= \partial_y^-\partial_y^+\partial_x^-\partial_x^+ u_{i,j} = \partial_x^-\partial_y^-\partial_x^+\partial_y^+ u_{i,j} = \partial_y^-\partial_x^-\partial_y^+\partial_x^+ u_{i,j} \\ &= 4u_{i,j} + u_{i+1,j+1} + u_{i-1,j+1} + u_{i+1,j-1} + u_{i-1,j-1} \\ &\quad - 2\left(u_{i,j+1} + u_{i,j-1} + u_{i+1,j} + u_{i-1,j}\right) \end{aligned} . \tag{2.19}$$

For simplicity, the finite differences on the boundaries for the discrete fourth-order derivatives are not listed. One can easily compose the already-defined second-order derivatives in (2.9)–(2.12) for the boundary conditions of the fourth-order derivatives in (2.17)–(2.19). Based on (2.17)–(2.19), we can obtain following two discrete fourth-order differential operators:

$$div^2\left(\nabla^2 u_{i,j}\right) = \partial_x^+\partial_x^-\partial_x^-\partial_x^+ u_{i,j} + \partial_y^-\partial_x^-\partial_y^+\partial_x^+ u_{i,j} + \partial_x^-\partial_y^-\partial_x^+\partial_y^+ u_{i,j} + \partial_y^+\partial_y^-\partial_y^-\partial_y^+ u_{i,j} \tag{2.20}$$

$$\Delta\left(\Delta u_{i,j}\right) = \partial_x^-\partial_x^+\partial_x^-\partial_x^+ u_{i,j} + \partial_x^-\partial_x^+\partial_y^-\partial_y^+ u_{i,j} + \partial_y^-\partial_y^+\partial_x^-\partial_x^+ u_{i,j} + \partial_y^-\partial_y^+\partial_y^-\partial_y^+ u_{i,j}. \tag{2.21}$$

Note that $div^2\left(\nabla^2 u_{i,j}\right) = \Delta\left(\Delta u_{i,j}\right)$. Figure 1 describes the behaviour of all the previous discrete differential operators. Having defined all necessary discrete quantities, the numerical optimisation of the variational models in section 1 can be implemented in the next section.



(a) ∂_x^+ (b) ∂_y^+ (c) ∂_x^- (d) ∂_y^- (e) $\partial_x^+\partial_x^- = \partial_x^-\partial_x^+$ (f) $\partial_y^+\partial_y^- = \partial_y^-\partial_y^+$

(g) $\partial_x^+\partial_y^+ = \partial_y^+\partial_x^+$ (h) $\partial_x^-\partial_y^- = \partial_y^-\partial_x^-$ (i) $\partial_x^+\partial_y^-$ (j) $\partial_y^+\partial_x^-$

(k) Derivatives in (3.17) (l) Derivatives in (3.18) (m) Derivatives in (3.19)

Figure 1. Illustration of the discrete first-order, second-order and fourth-order derivative approximations.

3. Fast numerical algorithm based on Split Bregman Method

3.1. Split Bregman for total variation model (1.1)

First, we introduce an auxiliary splitting vector variable $\boldsymbol{w} = (w_1\ w_2)$, a Bregman iterative parameter $\boldsymbol{b} = (b_1\ b_2)$ and a positive penalty parameter θ, transforming the functional (1.1) into following form

$$E(u,\boldsymbol{w};\boldsymbol{b}) = \frac{1}{2}\int_\Omega (u-f)^2+\alpha\int_\Omega |\boldsymbol{w}|+\frac{\theta}{2}\int_\Omega (\boldsymbol{w}-\nabla u-\boldsymbol{b})^2. \tag{3.1}$$

In order to find the minimiser of functional (1.1), an alternating optimization method should be used to search all the saddle points of functional (3.1). To do so, we first fix variable $\boldsymbol{w}$, deriving the Euler-equation with respect to u

$$u-\theta div(\nabla u) = f-\theta div(\boldsymbol{w}-\boldsymbol{b}). \tag{3.2}$$

According to the definition of discrete gradient (2.1), divergence (2.8) and Laplace (2.13) in section 2, the discretisation form of (3.2) can be written as

$$u_{i,j}-\theta\left(\partial_x^-\partial_x^+u_{i,j}+\partial_y^-\partial_y^+u_{i,j}\right) = G_{i,j} \tag{3.3}$$

with $G_{i,j} = f_{i,j}-\theta\left(\partial_x^-\left(w_{1i,j}-b_{1i,j}\right)+\partial_y^-\left(w_{2i,j}-b_{2i,j}\right)\right)$. As periodic boundary conditions have been imposed on the discrete derivatives in section 2, the discrete Fourier transform can be directly applied to the both sides of Eqn (3.3)

$$\mathcal{F}\left(u_{i,j}-\theta\left(\partial_x^-\partial_x^+u_{i,j}+\partial_y^-\partial_y^+u_{i,j}\right)\right) = \mathcal{F}\left(G_{i,j}\right),$$

where $\mathcal{F}$ denotes the discrete Fourier transform. For discrete frequencies r and s, we have

$$\underbrace{\left(1-2\theta\left(\cos\frac{2\pi s}{N}+\cos\frac{2\pi r}{M}-2\right)\right)}_{\xi}\mathcal{F}\left(u_{i,j}\right)=\mathcal{F}\left(G_{i,j}\right). \tag{3.4}$$

$i \in [1, M]$ and $j \in [1, N]$ are the indexes in discrete time domain. $r \in [0, M)$ and $s \in [0, N)$ are the frequencies in the discrete frequency domain. (3.4) provides us with a closed-form solution of u as

$$u_{i,j}=\Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{i,j}\right)}{\xi}\right)\right), \tag{3.5}$$

where $\mathcal{F}^{-1}$ denotes the discrete inverse Fourier transform. $\Re$ is the real part of a complex number. "—" stands for pointwise division of matrices.

Fixing variables u for vector $\boldsymbol{w}$ next, we can obtain its Euler-equation as follows

$$\alpha\frac{\boldsymbol{w}}{|\boldsymbol{w}|}+\theta\left(\boldsymbol{w}-\nabla u-\boldsymbol{b}\right)=0,$$

which can be solved component wisely through following discrete 2D analytical generalised soft thresholding equation with the convention that $0/0 = 0$

$$\boldsymbol{w}_{i,j}=\max\left(\left|\nabla u_{i,j}+\boldsymbol{b}_{i,j}\right|-\frac{\alpha}{\theta},0\right)\frac{\nabla u_{i,j}+\boldsymbol{b}_{i,j}}{\left|\nabla u_{i,j}+\boldsymbol{b}_{i,j}\right|}. \tag{3.6}$$

Finally, updating the Bregman iterative parameter $\boldsymbol{b}$, which can be seen from step 6 in the **Algorithm** 3.1. The split Bregman algorithm for the TV denoising problem (1.1) reads as follows:

Algorithm 3.1 : Split Bregman algorithm for TV model (1.1)

```
1: function TV_Denoising(f)
2:   Initialization: Set $(\boldsymbol{w};\boldsymbol{b})=0$, $(\alpha,\theta)>0$
3:   repeat
4:     Compute $u$ according to (3.5)
5:     Compute $\boldsymbol{w}$ according to (3.6)
6:     Update Bregman iterative parameter $\boldsymbol{b}\leftarrow\boldsymbol{b}+\nabla u-\boldsymbol{w}$
7:   until convergence of $u$
8:   return $u$
9: end function
```

3.2. *Split Bregman for total Laplace (TL) model (1.2)*

In order to minimise TL model (1.2), an auxiliary splitting scalar variable w, a Bregman iterative parameter b and a penalty parameter θ are introduced, transforming (1.2) into

$$E(u,w;b)=\frac{1}{2}\int_{\Omega}(u-f)^2+\alpha\int_{\Omega}|w|+\frac{\theta}{2}\int_{\Omega}(w-\Delta u-b)^2.$$

Applying alternating optimization technique, we first fix w for u and obtain following Euler equation

$$u+\theta\Delta\left(\Delta u\right)=f+\theta\Delta\left(w-b\right). \tag{3.7}$$

Referring to definition of (2.13) and (2.21) in section 2, (3.7) is discretised as

$$u_{i,j}+\theta\left(\partial_x^-\partial_x^+\partial_x^-\partial_x^+u_{i,j}+\partial_x^-\partial_x^+\partial_y^-\partial_y^+u_{i,j}+\partial_y^-\partial_y^+\partial_x^-\partial_x^+u_{i,j}+\partial_y^-\partial_y^+\partial_y^-\partial_y^+u_{i,j}\right)=G_{i,j} \tag{3.8}$$

with $G_{i,j}=f_{i,j}+\theta\left(\partial_x^-\partial_x^+\left(w_{i,j}-b_{i,j}\right)+\partial_y^-\partial_y^+\left(w_{i,j}-b_{i,j}\right)\right)$. Applying discrete Fourier transform to the both sides of Eqn (3.8) gives

$$\mathcal{F}\left(u_{i,j}+\theta\left(\partial_x^-\partial_x^+\partial_x^-\partial_x^+u_{i,j}+\partial_x^-\partial_x^+\partial_y^-\partial_y^+u_{i,j}+\partial_y^-\partial_y^+\partial_x^-\partial_x^+u_{i,j}+\partial_y^-\partial_y^+\partial_y^-\partial_y^+u_{i,j}\right)\right)=\mathcal{F}\left(G_{i,j}\right) \tag{3.9}$$

For discrete frequencies r and s, we have the following equivalent of Eqn (3.9)

$$\underbrace{\left(1+4\theta\left(\cos\frac{2\pi s}{N}+\cos\frac{2\pi r}{M}-2\right)^{2}\right)}_{\xi}\mathcal{F}\left(u_{i,j}\right)=\mathcal{F}\left(G_{i,j}\right). \tag{3.10}$$

(3.11) provides us with a closed-form solution of u as

$$u_{i,j}=\Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{i,j}\right)}{\xi}\right)\right). \tag{3.11}$$

Next fixing variable u to derive following Euler equation with respect to scalar variable w

$$\alpha\frac{w}{|w|}+\theta\left(w-\Delta u-b\right)=0,$$

which can be solved component wisely through the following discrete 1D analytical generalised soft thresholding equation with the convention that $0/0=0$

$$w_{i,j}=\max\left(\left|\Delta u_{i,j}+b_{i,j}\right|-\frac{\alpha}{\theta},0\right)sign\left(\Delta u_{i,j}+b_{i,j}\right). \tag{3.12}$$

Lastly, we give the overall implementation of TL model (1.2), where Bregman iterative parameter b is updated.

Algorithm 3.2 : Split Bregman algorithm for TL model (1.2)

```
function TL_Denoising(f)
Initialization: Set $(w;b)=0$, $(\alpha,\theta)>0$
repeat
Compute $u$ according to (3.11)
Compute $w$ according to (3.12)
Update Bregman iterative parameter $b \leftarrow b+\Delta u-w$
until convergence of $u$
return $u$
end function
```

3.3. *Split Bregman for bounded Hessian model (1.3)*

To solve BH model (1.3) using split Bregman method, we transform it into following multivariable energy functional

$$E\left(u,\boldsymbol{w};\boldsymbol{b}\right)=\frac{1}{2}\int_{\Omega}(u-f)^{2}+\alpha\int_{\Omega}|\boldsymbol{w}|+\frac{\theta}{2}\int_{\Omega}\left(\boldsymbol{w}-\nabla^{2}u-\boldsymbol{b}\right)^{2}, \tag{3.13}$$

where $\boldsymbol{w}=\begin{pmatrix}w_1 & w_2\\ w_3 & w_4\end{pmatrix}$ is a matrix-valued function related to the Hessian of the function u. $|\boldsymbol{w}|=\sqrt{\sum_{1\le n\le 4}\left(\boldsymbol{w}_n\right)^2}$ represents the Frobenius norm of matrix $\boldsymbol{w}$. By introducing Bregman iteration parameter $\boldsymbol{b}=\begin{pmatrix}b_1 & b_2\\ b_3 & b_4\end{pmatrix}$ and applying Bregman distance technique, constraint $\boldsymbol{w}=\nabla^2 u$ can be effectively enforced when functional (3.13) reaches convergent.

Using the same manner to optimise (3.13), we first fix the 4D vector variable $\boldsymbol{w}$ for u

$$u+\theta div^{2}\left(\nabla^{2}u\right)=f+\theta div^{2}\left(\boldsymbol{w}-\boldsymbol{b}\right) \tag{3.14}$$

The discretisation form of (3.14) is given as follows according to the definition (2.14) and (2.20) in section 2:

$$u_{i,j}+\theta\left(\partial_x^{+}\partial_x^{-}\partial_x^{-}\partial_x^{+}u_{i,j}+\partial_y^{-}\partial_x^{-}\partial_y^{+}\partial_x^{+}u_{i,j}+\partial_x^{-}\partial_y^{-}\partial_x^{+}\partial_y^{+}u_{i,j}+\partial_y^{+}\partial_y^{-}\partial_y^{-}\partial_y^{+}u_{i,j}\right)=G_{i,j} \tag{3.15}$$

with $G=f+\theta\left(\partial_x^{+}\partial_x^{-}\left(w_1-b_1\right)+\partial_y^{-}\partial_x^{-}\left(w_2-b_2\right)+\partial_x^{-}\partial_y^{-}\left(w_3-b_3\right)+\partial_y^{+}\partial_y^{-}\left(w_4-b_4\right)\right)$. Applying discrete Fourier transform to the both sides of equation (3.15) gives

$$\mathcal{F}\left(u_{i,j}+\theta\left(\partial_x^{+}\partial_x^{-}\partial_x^{-}\partial_x^{+}u_{i,j}+\partial_y^{-}\partial_x^{-}\partial_y^{+}\partial_x^{+}u_{i,j}+\partial_x^{-}\partial_y^{-}\partial_x^{+}\partial_y^{+}u_{i,j}+\partial_y^{+}\partial_y^{-}\partial_y^{-}\partial_y^{+}u_{i,j}\right)\right)=\mathcal{F}\left(G_{i,j}\right). \tag{3.16}$$

The equivalent of (3.16) is

$$\underbrace{\left(1+4\theta\left(\cos\frac{2\pi s}{N}+\cos\frac{2\pi r}{M}-2\right)^2\right)}_{\xi}\mathcal{F}\left(u_{i,j}\right)=\mathcal{F}\left(G_{i,j}\right). \tag{3.17}$$

Note that the left-hand side of (3.17) is the same as that of (3.11). (3.17) gives following closed-form

$$u_{i,j}=\Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{i,j}\right)}{\xi}\right)\right). \tag{3.18}$$

Fixing variable u to derive the following Euler equation with respect to 4D vector $\boldsymbol{w}$

$$\alpha\frac{\boldsymbol{w}}{|\boldsymbol{w}|}+\theta\left(\boldsymbol{w}-\nabla^2u-\boldsymbol{b}\right)=0,$$

which can be analytically solved by following discrete 4D generalised soft thresholding equation with the convention that $0/0=0$

$$\boldsymbol{w}_{i,j}=\max\left(\left|\nabla^2u_{i,j}+\boldsymbol{b}_{i,j}\right|-\frac{\alpha}{\theta},0\right)\frac{\nabla^2u_{i,j}+\boldsymbol{b}_{i,j}}{\left|\nabla^2u_{i,j}+\boldsymbol{b}_{i,j}\right|}. \tag{3.19}$$

Algorithm 3.3 : Split Bregman algorithm for BH model (1.3)

```
function BH_Denoising(f)
Initialization: Set $(\boldsymbol{w};\boldsymbol{b})=0$ and $(\alpha,\theta)>0$
repeat
Compute $u$ according to (3.18)
Compute $\boldsymbol{w}$ according to (3.19)
Update Bregman iterative parameter $\boldsymbol{b}\leftarrow\boldsymbol{b}+\nabla^2u-\boldsymbol{w}$
until convergence of $u$
return $u$
end function
```

3.4. *Split Bregman for CEP2-L^2 model (1.5)*

As CEP2-L^2 model (1.5) includes both first-order and second-order derivatives, we should introduce two splitting variables $(\boldsymbol{w},v)$, two Bregman iterative parameters $(\boldsymbol{b},d)$ and two penalty parameters (θ_1,θ_2), transforming the original functional into

$$\begin{aligned}E\left(u_1,u_2,\boldsymbol{w},v;\boldsymbol{b},d\right)=&\frac{1}{2}\int_\Omega\left(f-u_1-u_2\right)^2\\&+\alpha\int_\Omega|\boldsymbol{w}|+\frac{\theta_1}{2}\int_\Omega\left(\boldsymbol{w}-\nabla u_1-\boldsymbol{b}\right)^2.\\&+\beta\int_\Omega|v|+\frac{\theta_2}{2}\int_\Omega\left(v-\Delta u_2-d\right)^2\end{aligned}$$

To do optimisation, we first fix variables $(u_2,\boldsymbol{w},v;\boldsymbol{b},d)$ for u_1 and obtain following Euler equation

$$u_1-\theta_1 div\left(\nabla u_1\right)=\left(f-u_2\right)+\theta_1 div\left(\boldsymbol{b}-\boldsymbol{w}\right). \tag{3.20}$$

This equation is second-order linear PDE, which has similar form to (3.2). We can directly write the solution of u_1 as

$$u_{1i,j}=\Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{1i,j}\right)}{\xi_1}\right)\right), \tag{3.21}$$

where G_1 is the right-hand side of (3.20) and

$$\xi_1=1-2\theta_1\left(\cos\frac{2\pi s}{N}+\cos\frac{2\pi r}{M}-2\right). \tag{3.22}$$

Fixing variables $(u_1,\boldsymbol{w},v;\boldsymbol{b},d)$ for u_2, we have

$$u_2+\theta_2\Delta\left(\Delta u_2\right)=\left(f-u_1\right)+\theta_2\Delta\left(v-d\right). \tag{3.23}$$

Note that variable u_1 has been calculated from (3.21) and should be used for (3.23). (3.23) has the similar form to (3.7), and its closed-form solution can be derived as

$$u_{2i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{2i,j}\right)}{\xi_2}\right)\right), \tag{3.24}$$

where G_2 is the right-hand side of (3.23) and

$$\xi_2 = 1 + 4\theta_2\left(\cos\frac{2\pi s}{N} + \cos\frac{2\pi r}{M} - 2\right)^2. \tag{3.25}$$

Fixing variable $(u_1, u_2, v; \boldsymbol{b}, d)$ for vector $\boldsymbol{w}$, we have

$$\alpha\frac{\boldsymbol{w}}{|\boldsymbol{w}|} + \theta_1\left(\boldsymbol{w} - \nabla u_1 - \boldsymbol{b}\right) = 0.$$

Its analytical solution is as follows:

$$\boldsymbol{w}_{i,j} = \max\left(\left|\nabla u_{1i,j} + \boldsymbol{b}_{i,j}\right| - \frac{\alpha}{\theta_1}, 0\right)\frac{\nabla u_{1i,j} + \boldsymbol{b}_{i,j}}{\left|\nabla u_{1i,j} + \boldsymbol{b}_{i,j}\right|} \tag{3.26}$$

After $\boldsymbol{w}$ is updated, the following formulation with respect to scalar variable v is derived

$$\beta\frac{v}{|v|} + \theta_2\left(v - \Delta u_2 - d\right) = 0.$$

The solution is similar to (3.12), which is a 1D analytical generalised soft thresholding equation

$$v_{i,j} = \max\left(\left|\Delta u_{2i,j} + d_{i,j}\right| - \frac{\beta}{\theta_2}, 0\right) sign\left(\Delta u_{2i,j} + d_{i,j}\right). \tag{3.27}$$

Finally, the update of two Bregman iterative parameters are shown from step 08 and 09 in the overall **Algorithm** 3.4

Algorithm 3.4 : Split Bregman algorithm for CEP2-L^2 model (1.5)

```
01: function CEP2-L$^2_{Denoising}(f)$
02: Initialization: Set $u_2 = f$, $(\boldsymbol{w}, v; \boldsymbol{b}, d) = 0$, $(\alpha, \beta, \theta_1, \theta_2) > 0$
03: repeat
04: Compute $u_1$ according to (3.21)
05: Compute $u_2$ according to (3.24)
06: Compute $\boldsymbol{w}$ according to (3.26)
07: Compute $v$ according to (3.27)
08: Update Bregman iterative parameter $\boldsymbol{b} \leftarrow \boldsymbol{b} + \nabla u_1 - \boldsymbol{w}$
09: Update Bregman iterative parameter $d \leftarrow d + \Delta u_2 - v$
10: until convergence of $u = u_1 + u_2$
11: return $u = u_1 + u_2$
12: end function
```

3.5. *Split Bregman for TV and Laplace model (1.6)*

Similarly to the optimisation produces of CEP2-L^2 functional, we first introduce same numbers of auxiliary variables and obtain the following energy functional

$$\begin{aligned} E\left(u, \boldsymbol{w}, v; \boldsymbol{b}, d\right) = \frac{1}{2}\int_\Omega (f-u)^2 \\ + \alpha\int_\Omega |\boldsymbol{w}| + \frac{\theta_1}{2}\int_\Omega \left(\boldsymbol{w} - \nabla u - \boldsymbol{b}\right)^2. \\ + \beta\int_\Omega |v| + \frac{\theta_2}{2}\int_\Omega \left(v - \Delta u - d\right)^2 \end{aligned}$$

The Euler equation with respect to u is

$$u - \theta_1 div\left(\nabla u\right) + \theta_2\Delta\left(\Delta u\right) = f - \theta_1 div\left(\boldsymbol{w} - \boldsymbol{b}\right) + \theta_2\Delta\left(v - d\right). \tag{3.28}$$

This formulation is a combination of (3.2) and (3.7), which can be also solved by discrete Fourier transform. Its solution is given as

$$u_{i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{i,j}\right)}{\xi}\right)\right), \tag{3.29}$$

where G is the right-hand side of (3.28). $\xi = \xi_1 + \xi_2 - 1$ where ξ_1 and ξ_2 are (3.22) and (3.25), respectively.

The discretisation problems of $\boldsymbol{w}$ and v are given by following two equations, respectively:

$$\boldsymbol{w}_{i,j} = \max\left(\left|\nabla u_{i,j} + \boldsymbol{b}_{i,j}\right| - \frac{\alpha}{\theta_1}, 0\right) \frac{\nabla u_{i,j} + \boldsymbol{b}_{i,j}}{\left|\nabla u_{i,j} + \boldsymbol{b}_{i,j}\right|} \tag{3.30}$$

$$v_{i,j} = \max\left(\left|\Delta u_{i,j} + d_{i,j}\right| - \frac{\beta}{\theta_2}, 0\right) sign\left(\Delta u_{i,j} + d_{i,j}\right). \tag{3.31}$$

Algorithm 3.5 : Split Bregman algorithm for TVL model (1.6)

```
01: function TVL_{Denoising}(f)
02: Initialization: Set $(\boldsymbol{w}, v; \boldsymbol{b}, d) = 0$, $(\alpha, \beta, \theta_1, \theta_2) > 0$
03: repeat
04: Compute $u$ according to (3.29)
05: Compute $\boldsymbol{w}$ according to (3.30)
06: Compute $v$ according to (3.31)
07: Update Bregman iterative parameter $\boldsymbol{b} \leftarrow \boldsymbol{b} + \nabla u - \boldsymbol{w}$
08: Update Bregman iterative parameter $d \leftarrow d + \Delta u - v$
09: until convergence of $u$
10: return $u$
11: end function
```

3.6. *Split Bregman for infimal-convolution model (1.8)*

The INFCON model (1.8) can be written as the following multivariable functional:

$$\begin{aligned} E\left(u_1, u_2, \boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d}\right) = & \frac{1}{2}\int_\Omega (f - u_1 - u_2)^2 \\ & + \alpha \int_\Omega |\boldsymbol{w}| + \frac{\theta_1}{2}\int_\Omega (\boldsymbol{w} - \nabla u_1 - \boldsymbol{b})^2, \\ & + \beta \int_\Omega |\boldsymbol{v}| + \frac{\theta_2}{2}\int_\Omega \left(\boldsymbol{v} - \nabla^2 u_2 - \boldsymbol{d}\right)^2 \end{aligned}$$

where variables $\boldsymbol{w}$ and $\boldsymbol{b}$ are 2D vector, while $\boldsymbol{v}$ and $\boldsymbol{d}$ are 4D vector. To minimise the functional, we first fix $(u_2, \boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d})$ for u_1. The solution form of u_1 is exactly same as formulation (3.21).

After u_1 is solved, we fix $(u_1, \boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d})$ and derive the Euler equation with respect to u_2

$$u_2 + \theta_2 div^2\left(\nabla^2 u_2\right) = (f - u_1) + \theta_2 div^2\left(\boldsymbol{v} - \boldsymbol{d}\right) \tag{3.32}$$

This formulation is similar to (3.14) and can be solved via following FFT under the assumption of circulant boundary condition

$$u_{2i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{2i,j}\right)}{\xi_2}\right)\right), \tag{3.33}$$

where G_2 is the right-hand side of Eqn (3.32) and ξ_2 is defined in (3.25).

$(u_1, u_2, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d})$ are fixed next to obtain the solution of the variable $\boldsymbol{w}$, which is same as (3.26).

Fixing $(u_1, u_2, \boldsymbol{w}; \boldsymbol{b}, \boldsymbol{d})$ for vector $\boldsymbol{v}$, we have following Euler equation:

$$\beta \frac{\boldsymbol{v}}{|\boldsymbol{v}|} + \theta_2\left(\boldsymbol{v} - \nabla^2 u_2 - \boldsymbol{d}\right) = 0.$$

Its solution is the following 4D analytical equation with the convention that $0/0 = 0$

$$\boldsymbol{v}_{i,j} = \max\left(\left|\nabla^2 u_{2i,j} + \boldsymbol{d}_{i,j}\right| - \frac{\beta}{\theta_2}, 0\right) \frac{\nabla^2 u_{2i,j} + \boldsymbol{d}_{i,j}}{\left|\nabla^2 u_{2i,j} + \boldsymbol{d}_{i,j}\right|}. \tag{3.34}$$

The overall **Algorithm** 3.6 for INFCON model is as follows:

Algorithm 3.6 : Split Bregman algorithm for INFCON model (1.8)

```
function INFCON_Denoising(f)
Initialization: Set $u_2 = f$, $(\boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d}) = 0$, $(\alpha, \beta, \theta_1, \theta_2) > 0$
repeat
Compute $u_1$ according to (3.21)
Compute $u_2$ according to (3.33)
Compute $\boldsymbol{w}$ according to (3.26)
Compute $\boldsymbol{v}$ according to (3.34)
Update Bregman iterative parameter $\boldsymbol{b} \leftarrow \boldsymbol{b} + \nabla u_1 - \boldsymbol{w}$
Update Bregman iterative parameter $\boldsymbol{d} \leftarrow \boldsymbol{d} + \nabla^2 u_2 - \boldsymbol{v}$
until convergence of $u = u_1 + u_2$
return $u = u_1 + u_2$
end function
```

3.7. *Split Bregman for combined TV and BH (TVBH) model (1.7)*

Using the same manner as before, we first transform original (1.7) model into following multivaiable formulation

$$\begin{aligned} E\left(u, \boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d}\right) = {} & \frac{1}{2}\int_{\Omega} (f-u)^2 \\ & + \alpha \int_{\Omega} |\boldsymbol{w}| + \frac{\theta_1}{2}\int_{\Omega} (\boldsymbol{w} - \nabla u - \boldsymbol{b})^2 \\ & + \beta \int_{\Omega} |\boldsymbol{v}| + \frac{\theta_2}{2}\int_{\Omega} \left(\boldsymbol{v} - \nabla^2 u - \boldsymbol{d}\right)^2 \end{aligned}$$

To minimise the functional, we first fix $(\boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d})$ for u and obtain its Euler equation

$$u - \theta_1 div\left(\nabla u\right) + \theta_2 div^2\left(\nabla^2 u\right) = f - \theta_1 \nabla\left(\boldsymbol{w} - \boldsymbol{b}\right) + \theta_2 div^2\left(\boldsymbol{v} - \boldsymbol{d}\right). \tag{3.35}$$

This formulation is a combination of (3.2) and (3.14), which can be solved via discrete fast Fourier transform. Its solution is given as

$$u_{i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{i,j}\right)}{\xi}\right)\right), \tag{3.36}$$

where G is the right-hand side of (3.35). $\xi = \xi_1 + \xi_2 - 1$ where ξ_1 and ξ_2 are (3.22) and (3.25), respectively.

The solution of variable $\boldsymbol{w}$ is same as (3.30), and vector $\boldsymbol{v}$ can be updated by

$$\boldsymbol{v}_{i,j} = \max\left(\left|\nabla^2 u_{i,j} + \boldsymbol{d}_{i,j}\right| - \frac{\beta}{\theta_2}, 0\right)\frac{\nabla^2 u_{i,j} + \boldsymbol{d}_{i,j}}{\left|\nabla^2 u_{i,j} + \boldsymbol{d}_{i,j}\right|}. \tag{3.37}$$

Algorithm 3.7 : Split Bregman algorithm for TVBH model (1.7)

```
function TVBH_Denoising(f)
Initialization: Set $(\boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d}) = 0$, $(\alpha, \beta, \theta_1, \theta_2) > 0$
repeat
Compute $u$ according to (3.36)
Compute $\boldsymbol{w}$ according to (3.30)
Compute $\boldsymbol{v}$ according to (3.37)
Update Bregman iterative parameter $\boldsymbol{b} \leftarrow \boldsymbol{b} + \nabla u - \boldsymbol{w}$
Update Bregman iterative parameter $\boldsymbol{d} \leftarrow \boldsymbol{d} + \nabla^2 u - \boldsymbol{v}$
until convergence of $u$
return $u$
end function
```

3.8. *Split Bregman for total generalised variation model (1.9)*

In order to solve TGV model (1.9) using split Bregman method, we introduce auxiliary variables $(\boldsymbol{w}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d})$ and positive penalty parameters (θ_1, θ_2), transforming it into following form:

$$\begin{aligned} E\left(u,\boldsymbol{p},\boldsymbol{w},\boldsymbol{v};\boldsymbol{b},\boldsymbol{d}\right) = {} & \frac{1}{2}\int_{\Omega}(u-f)^2 \\ & +\alpha\int_{\Omega}|\boldsymbol{w}| + \frac{\theta_1}{2}\int_{\Omega}(\boldsymbol{w}-\nabla u+\boldsymbol{p}-\boldsymbol{b})^2. \\ & +\beta\int_{\Omega}|\boldsymbol{v}| + \frac{\theta_2}{2}\int_{\Omega}(\boldsymbol{v}-\varepsilon(\boldsymbol{p})-\boldsymbol{d})^2 \end{aligned} \tag{3.38}$$

The meaning of the primal and dual variables in (3.38) is listed as follows:

- u denotes the denoised image we need to find;
- $\boldsymbol{p} = (p_1\ p_2)$ is the symmetrised gradient of the deformation field;
- $\boldsymbol{w} = (w_1\ w_2)$ is a 2D vector valued function related to vector field $\nabla u - \boldsymbol{p}$;
- $\boldsymbol{v} = \begin{pmatrix} v_{11} & v_3 \\ v_3 & v_{22} \end{pmatrix}$ is a 4D vector valued function related to symmetrised derivative $\varepsilon(\boldsymbol{p})$;
- $\boldsymbol{b} = (b_1\ b_2)$ denotes the Bregman iterative parameter for enforcing $\boldsymbol{w} = \nabla u - \boldsymbol{p}$;
- $\boldsymbol{d} = \begin{pmatrix} d_{11} & d_3 \\ d_3 & d_{22} \end{pmatrix}$ denotes the Bregman iterative parameter for enforcing $\boldsymbol{v} = \varepsilon(\boldsymbol{p})$.

Applying alternating optimisation method, we first fix variables $(\boldsymbol{p},\boldsymbol{w},\boldsymbol{v};\boldsymbol{b},\boldsymbol{d})$ to obtain the following Euler equation with respect to u

$$u - \theta_1 div\left(\nabla u\right) = f - \theta_1 div\left(\boldsymbol{w}+\boldsymbol{p}-\boldsymbol{b}\right). \tag{3.39}$$

The solution of this linear PDE is given as

$$u_{i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{i,j}\right)}{\xi_1}\right)\right), \tag{3.40}$$

where G is right-hand side of (3.39) and ξ_1 is defined as (3.22).

Fixing variables $(u,\boldsymbol{w},\boldsymbol{v};\boldsymbol{b},\boldsymbol{d})$ next, the Euler equation with respect to $\boldsymbol{p} = (p_1\ p_2)$ can be derived. For each p_1 and p_2, we have following two discrete formulations

$$\left(\theta_1 - \theta_2\partial_x^+\partial_x^- - \frac{\theta_2}{2}\partial_y^+\partial_y^-\right)p_{1i,j} - \frac{\theta_2}{2}\partial_y^+\partial_x^- p_{2i,j} = h_{1i,j} \tag{3.41}$$

$$\left(\theta_1 - \frac{\theta_2}{2}\partial_x^+\partial_x^- - \theta_2\partial_y^+\partial_y^-\right)p_{2i,j} - \frac{\theta_2}{2}\partial_x^+\partial_y^- p_{1i,j} = h_{2i,j}, \tag{3.42}$$

where the differential operators in (3.41) and (3.42) can be found in section 2 and

$$\begin{aligned} h_1 &= \theta_1\left(\partial_x^+ u + b_1 - w_1\right) - \theta_2\partial_x^+\left(v_{11}-d_{11}\right) - \theta_2\partial_y^+\left(v_3-d_3\right) \\ h_2 &= \theta_1\left(\partial_y^+ u + b_2 - w_2\right) - \theta_2\partial_y^+\left(v_{22}-d_{22}\right) - \theta_2\partial_x^+\left(v_3-d_3\right) \end{aligned}.$$

By applying the discrete Fourier transform to both sides of (3.41) and (3.42), we have the following system of linear equations

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}\begin{pmatrix} \mathcal{F}(p_{1i,j}) \\ \mathcal{F}(p_{2i,j}) \end{pmatrix} = \begin{pmatrix} \mathcal{F}(h_{1i,j}) \\ \mathcal{F}(h_{2i,j}) \end{pmatrix},$$

where the coefficients are

$$\begin{aligned} a_{11} &= \theta_1 - \theta_2\left(2\cos\frac{2\pi s}{N} - 2\right) - \frac{\theta_2}{2}\left(2\cos\frac{2\pi r}{M} - 2\right) \\ a_{12} &= -\frac{\theta_2}{2}\left(-1+\cos\frac{2\pi r}{M} + \sqrt{-1}\sin\frac{2\pi r}{M}\right)\left(1-\cos\frac{2\pi s}{N} + \sqrt{-1}\sin\frac{2\pi s}{N}\right) \\ a_{21} &= -\frac{\theta_2}{2}\left(-1+\cos\frac{2\pi s}{N} + \sqrt{-1}\sin\frac{2\pi s}{N}\right)\left(1-\cos\frac{2\pi r}{M} + \sqrt{-1}\sin\frac{2\pi r}{M}\right) \\ a_{22} &= \theta_1 - \frac{\theta_2}{2}\left(2\cos\frac{2\pi s}{N} - 2\right) - \theta_2\left(2\cos\frac{2\pi r}{M} - 2\right) \end{aligned}.$$

The coefficient matrix $\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ is a $M \times N$ numbers of 2×2, whose determinant is

$$D = \left(\theta_1 - 2\theta_2 \left(\cos\frac{2\pi s}{N} + \cos\frac{2\pi r}{M} - 2\right)\right)\left(\theta_1 - \theta_2 \left(\cos\frac{2\pi s}{N} + \cos\frac{2\pi r}{M} - 2\right)\right),$$

which is always positive for all discrete frequencies if $(\theta_1, \theta_2) > 0$. After the systems of linear equations are solved for each frequency r and s over the discrete frequency domain, we use the discrete inverse Fourier transform to obtain the analytical forms of p_1 and p_2

$$p_{1i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{a_{22}\mathcal{F}\left(h_{1i,j}\right) - a_{12}\mathcal{F}\left(h_{2i,j}\right)}{D}\right)\right) \tag{3.43}$$

$$p_{2i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{a_{11}\mathcal{F}\left(h_{2i,j}\right) - a_{21}\mathcal{F}\left(h_{1i,j}\right)}{D}\right)\right). \tag{3.44}$$

Fixing variables $(u, \boldsymbol{p}, \boldsymbol{v}; \boldsymbol{b}, \boldsymbol{d})$ for 2D vector $\boldsymbol{w}$, we have

$$\alpha\frac{\boldsymbol{w}}{|\,\boldsymbol{w}|} + \theta_1\left(\boldsymbol{w} - \nabla u + \boldsymbol{p} - \boldsymbol{b}\right) = 0,$$

which can be solved by following discrete 2D analytical generalised soft thresholding equation with convention that $0/0 = 0$

$$\boldsymbol{w}_{i,j} = \max\left(\left|\nabla u_{i,j} - \boldsymbol{p}_{i,j} + \boldsymbol{b}_{i,j}\right| - \frac{\alpha}{\theta_1}, 0\right)\frac{\nabla u_{i,j} - \boldsymbol{p}_{i,j} + \boldsymbol{b}_{i,j}}{\left|\nabla u_{i,j} - \boldsymbol{p}_{i,j} + \boldsymbol{b}_{i,j}\right|}. \tag{3.45}$$

After $\boldsymbol{w}$ is solved, the Euler equation with respect to $\boldsymbol{v}$ is given as

$$\beta\frac{\boldsymbol{v}}{|\,\boldsymbol{v}|} + \theta_2\left(\boldsymbol{v} - \varepsilon\left(\boldsymbol{p}\right) - \boldsymbol{d}\right) = 0$$

whose solution is following 4D analytical generalised soft thresholding equation with convention that $0/0 = 0$

$$\boldsymbol{v}_{i,j} = \max\left(\left|\varepsilon\left(\boldsymbol{p}_{i,j}\right) + \boldsymbol{d}_{i,j}\right| - \frac{\beta}{\theta_2}, 0\right)\frac{\varepsilon\left(\boldsymbol{p}_{i,j}\right) + \boldsymbol{d}_{i,j}}{\left|\varepsilon\left(\boldsymbol{p}_{i,j}\right) + \boldsymbol{d}_{i,j}\right|} \tag{3.46}$$

Finally, the overall **Algorithm** 3.8 is presented as

Algorithm 3.8 : Split Bregman algorithm for TGV model (1.9)

```
function TGV_Denoising(f)
Initialization: Set $(p, w, v; b, d) = 0$, $(\alpha, \beta, \theta_1, \theta_2) > 0$
repeat
    Compute $u$ according to (3.40)
    Compute $p$ according to (3.43) and (3.44)
    Compute $w$ according to (3.45)
    Compute $v$ according to (3.46)
    Update Bregman iterative parameter $b \leftarrow b + \nabla u - p - w$
    Update Bregman iterative parameter $d \leftarrow d + \varepsilon(p) - v$
until convergence of $u$
return $u$
end function
```

3.9. *Split Bregman for total curvature (TC) model (1.11)*

In order to solve TC model (1.11), we first consider the following splitting

$$E(u) = \frac{1}{2}\int_\Omega (u - f)^2 dx + \alpha\int_\Omega |q| dx \tag{3.47}$$

$$\text{s.t. } q = div\left(\boldsymbol{n}\right), \boldsymbol{p} = \nabla u, \boldsymbol{n} = \boldsymbol{p}/|\boldsymbol{p}|.$$

We note that constraint $\boldsymbol{n} = \boldsymbol{p}/|\boldsymbol{p}|$ is equivalent to following two constraints according to the well-known Hölder inequality

$$|\boldsymbol{n}| \leq 1,\ |\boldsymbol{p}| = \boldsymbol{n}\cdot\boldsymbol{p}. \tag{3.48}$$

A new auxiliary vector $\boldsymbol{m}$ is then introduced to replace variable $\boldsymbol{n}$ in (3.48), which can be deemed as a relaxation of $\boldsymbol{n}$ and FFT can be thus applied. After that, we have following five constraints

$$q = div\left(\boldsymbol{n}\right), \ \ \boldsymbol{p} = \nabla u, \ \ |\boldsymbol{m}| \leq 1, \ \ |\boldsymbol{p}| = \boldsymbol{m} \cdot \boldsymbol{p}, \ \ \boldsymbol{m} = \boldsymbol{n}. \tag{3.49}$$

For $|m| \leq 1$ in image domain, we have Ω, $|\boldsymbol{p}| \geq \boldsymbol{m} \cdot \boldsymbol{p}$. By introducing auxiliary variables and penalty parameters, the split Bregman method can be employed to transform (3.47) with constraints in (3.49) into following constrained energy functional

$$\begin{aligned} E\left(u, q, \boldsymbol{p}, \boldsymbol{n}, \boldsymbol{m}; b_1, \boldsymbol{b_2}, b_3, \boldsymbol{b_4}\right) = &\frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega |q| \\ &+ \int_\Omega (\theta_1 + b_1)(|\boldsymbol{p}| - \boldsymbol{p}\cdot\boldsymbol{m}) + \frac{\theta_2}{2}\int_\Omega (\boldsymbol{p} - \nabla u - \boldsymbol{b_2})^2 \\ &+ \frac{\theta_3}{2}\int_\Omega (q - div(\boldsymbol{n}) - b_3)^2 + \frac{\theta_4}{2}\int_\Omega (\boldsymbol{n} - \boldsymbol{m} - \boldsymbol{b_4})^2 \\ &\text{s.t.} \ \ |\boldsymbol{m}| \leq 1. \end{aligned} \tag{3.50}$$

Note that for the constraint $|\boldsymbol{p}| = \boldsymbol{m} \cdot \boldsymbol{p}$, we use augmented Lagrangian method with L1-norm for the penalisation in functional (3.50) as it is true that $|\boldsymbol{p}| - \boldsymbol{m} \cdot \boldsymbol{p} > 0$ when $|\boldsymbol{m}| \leq 1$. The meaning of the primal and dual variables in (3.50) is listed as in the following:

- u denotes the denoised image we need to find;
- q is a scalar valued function related to divergence of vector function $\boldsymbol{n}$;
- $\boldsymbol{p} = (p_1 \ p_2)$ is a vector valued function related to gradient of the function u;
- $\boldsymbol{n} = (n_1 \ n_2)$ is a vector valued function related to the unit vectors of the level curves of u;
- $\boldsymbol{m} = (m_1 \ m_2)$ is a vector valued function used to relax variable $\boldsymbol{n}$ and thus FFT can be used;
- b_1 denotes a scalar Lagrangian multiplier for constraint $|\boldsymbol{p}| = \boldsymbol{m} \cdot \boldsymbol{p}$;
- $\boldsymbol{b_2} = (b_{21} \ b_{22})$ denotes a vector Bregman iterative parameter for constraint $\boldsymbol{p} = \nabla u$;
- b_3 denotes a scalar Bregman iterative parameter for constraint $q = div(\boldsymbol{n})$;
- $\boldsymbol{b_4} = (b_{41} \ b_{42})$ denotes a vector Bregman iterative parameter for constraint $\boldsymbol{m} = \boldsymbol{n}$.

Using alternating optimisation technique and first fixing variables $(q, \boldsymbol{p}, \boldsymbol{n}, \boldsymbol{m}; b_1, \boldsymbol{b_2}, b_3, \boldsymbol{b_4})$, we obtain the Euler equation with respect to u as follows:

$$u - \theta_2 div\left(\nabla u\right) = f - \theta_2 div\left(\boldsymbol{p} - \boldsymbol{b_2}\right). \tag{3.51}$$

The solution of this second order linear PDE is given as

$$u_{i,j} = \Re\left(\mathcal{F}^{-1}\left(\frac{\mathcal{F}\left(G_{i,j}\right)}{\xi_1}\right)\right), \tag{3.52}$$

where G is right-hand side of (3.51) and ξ_1 is defined as (3.22).

Variables $(u, \boldsymbol{p}, \boldsymbol{n}, \boldsymbol{m}; b_1, \boldsymbol{b_2}, b_3, \boldsymbol{b_4})$ are fixed next, and we obtain the following equation

$$\alpha\frac{q}{|q|} + \theta_3\left(q - div\left(\boldsymbol{n}\right) - b_3\right) = 0\,,$$

Its solution reads as

$$q_{i,j} = \max\left(\left|div(\boldsymbol{n})_{i,j} + b_{3i,j}\right| - \frac{\alpha}{\theta_3}, 0\right) sign\left(div\left(\boldsymbol{n}_{i,j}\right) + b_{3i,j}\right) \tag{3.53}$$

Fixing variables $(u, q, \boldsymbol{n}, \boldsymbol{m}; b_1, \boldsymbol{b_2}, b_3, \boldsymbol{b_4})$, we obtain the Euler equation with respect to $\boldsymbol{p}$

$$(\theta_1 + b_1)\frac{\boldsymbol{p}}{|\boldsymbol{p}|} + \theta_2\left(\boldsymbol{p} - \nabla u - \boldsymbol{b_2}\right) - (\theta_1 + b_1)\boldsymbol{m} = 0,$$

which can be solved by the analytical soft thresholding equation as

$$\boldsymbol{p}_{i,j} = \max\left(\left|\boldsymbol{S}_{i,j}\right| - \frac{\left(\theta_1 + b_{1i,j}\right)}{\theta_2}, 0\right)\frac{\boldsymbol{S}_{i,j}}{\left|\boldsymbol{S}_{i,j}\right|}, \tag{3.54}$$

where $\boldsymbol{S}_{i,j} = \nabla u_{i,j} + \boldsymbol{b_2}_{i,j} + \frac{(\theta_1 + b_{1i,j})\boldsymbol{m}_{i,j}}{\theta_2}$. If the $\left|\boldsymbol{S}_{i,j}\right|$ is 0, we have $0 \cdot (0/0) = 0$.

Fixing $(u, q, \boldsymbol{p}, \boldsymbol{m}; b_1, \boldsymbol{b_2}, b_3, \boldsymbol{b_4})$, the following two Euler equations for vector variable $\boldsymbol{n} = (n_1 \ n_2)$ are derived, respectively

$$\left(\theta_4 - \theta_3\partial_x^+\partial_x^-\right)n_1 - \theta_3\partial_x^+\partial_y^- n_2 = h_{1i,j} \tag{3.55}$$

$$\left(\theta_4-\theta_3\partial_y^+\partial_y^-\right)n_2-\theta_3\partial_y^+\partial_x^-n_1=h_{2i,j}, \tag{3.56}$$

where

$$h_1=\theta_4\left(m_1+b_{41}\right)-\theta_3\partial_x^+\left(q-b_3\right)$$

$$h_2=\theta_4\left(m_2+b_{42}\right)-\theta_3\partial_y^+\left(q-b_3\right)$$

By applying the discrete Fourier transform to both sides of (3.55) and (3.56), we have the following linear system

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}\begin{pmatrix} \mathcal{F}(n_{1i,j}) \\ \mathcal{F}(n_{2i,j}) \end{pmatrix}=\begin{pmatrix} \mathcal{F}(h_{1i,j}) \\ \mathcal{F}(h_{2i,j}) \end{pmatrix},$$

where the coefficients are

$$\begin{aligned} a_{11}&=\theta_4-2\theta_3\left(\cos\frac{2\pi s}{N}-1\right)\\ a_{12}&=-\theta_3\left(1-\cos\frac{2\pi r}{M}+\sqrt{-1}\sin\frac{2\pi r}{M}\right)\left(-1+\cos\frac{2\pi s}{N}+\sqrt{-1}\sin\frac{2\pi s}{N}\right)\\ a_{21}&=-\theta_3\left(1-\cos\frac{2\pi s}{N}+\sqrt{-1}\sin\frac{2\pi s}{N}\right)\left(-1+\cos\frac{2\pi r}{M}+\sqrt{-1}\sin\frac{2\pi r}{M}\right)^{\cdot}\\ a_{22}&=\theta_4-2\theta_3\left(\cos\frac{2\pi r}{M}-1\right)\end{aligned}$$

The coefficient matrix $\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ is also a $M\times N$ numbers of 2×2 system, whose determinant is

$$D=\theta_4^2-2\theta_3\theta_4\left(\cos\frac{2\pi s}{N}+\cos\frac{2\pi r}{M}-2\right),$$

which is always positive for all discrete frequencies if $(\theta_3,\theta_4)>0$. After the systems of linear equations are solved for each frequency r and s over the discrete frequency domain, we use the discrete inverse Fourier transform to obtain the analytical forms of n_1 and n_2

$$n_{1i,j}=\Re\left(\mathcal{F}^{-1}\left(\frac{a_{22}\mathcal{F}\left(h_{1i,j}\right)-a_{12}\mathcal{F}\left(h_{2i,j}\right)}{D}\right)\right) \tag{3.57}$$

$$n_{2i,j}=\Re\left(\mathcal{F}^{-1}\left(\frac{a_{11}\mathcal{F}\left(h_{2i,j}\right)-a_{21}\mathcal{F}\left(h_{1i,j}\right)}{D}\right)\right). \tag{3.58}$$

Fixing $(u,q,\boldsymbol{p},\boldsymbol{n};b_1,\boldsymbol{b_2},b_3,\boldsymbol{b_4})$, variable $\boldsymbol{m}$ can be derived from the following closed form:

$$\tilde{\boldsymbol{m}}_{i,j}=\frac{\left(\theta_1+b_1\right)\boldsymbol{p}_{i,j}}{\theta_4}+\boldsymbol{n}_{i,j}-\boldsymbol{b_4}_{i,j}.$$

In order to satisfy constraint $|\boldsymbol{m}|\le 1$, the following projection formulation is then imposed on the $\tilde{\boldsymbol{m}}_{\boldsymbol{i},\boldsymbol{j}}$

$$\boldsymbol{m}_{i,j}=\frac{\tilde{\boldsymbol{m}}_{i,j}}{\max\left(\left|\tilde{\boldsymbol{m}}_{i,j}\right|,\,1\right)} \tag{3.59}$$

Finally, updating the auxiliary variables $(b_1,\boldsymbol{b_2},b_3,\boldsymbol{b_4})$, which are listed from step 09 to 12 in the **Algorithm** 3.9. The split Bregman algorithm for the TC denoising problem (1.11) is as follows:

4. Numerical experiments

In this section, the performance of different methods is compared quantitatively and qualitatively. We apply peak signal-to-noise ratio, signal-to-noise ratio, root mean square error and structure similarity index map to perform quantitative evaluation for different methods. The models and their energy functionals are summarised in Table I in order to easily assess the difference between them. All experiments are performed using Matlab 2014b on a Windows 7 platform with an Intel Xeon CPU E5-1620 at 3.70GHz and 32GB memory.

Algorithm 3.9 : Split Bregman algorithm for TC model (1.11)

```
function TC_{Denoising}(f)
Initialization: Set $(q, \boldsymbol{p}, \boldsymbol{n}, \boldsymbol{m}; b_1, \boldsymbol{b_2}, b_3, \boldsymbol{b_4}) = 0$, $(\alpha, \theta_1, \theta_2, \theta_3, \theta_4) > 0$
repeat
    Compute $u$ according to (3.52)
    Compute $q$ according to (3.53)
    Compute $\boldsymbol{p}$ according to (3.54)
    Compute $\boldsymbol{n}$ according to (3.57) and (3.58)
    Compute $\boldsymbol{m}$ according to (3.59)
    Update Lagrangian multiplier $b_1 \leftarrow b_1 + \theta_1(|\boldsymbol{p}| - \boldsymbol{m} \cdot \boldsymbol{p})$
    Update Bregman iterative parameter $\boldsymbol{b_2} \leftarrow \boldsymbol{b_2} + \nabla u - \boldsymbol{p}$
    Update Bregman iterative parameter $b_3 \leftarrow b_3 + div(\boldsymbol{n}) - q$
    Update Bregman iterative parameter $\boldsymbol{b_4} \leftarrow \boldsymbol{b_4} + \boldsymbol{m} - \boldsymbol{n}$
until convergence of $u$
return $u$
end function;
```

Table I. Models for comparison

No.	Model	Energy functional
1	TV	$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \lvert\nabla u\rvert$
2	TL	$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \lvert\Delta u\rvert$
3	BH	$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \lvert\nabla^2 u\rvert$
4	CEP2-L^2	$E(u_1, u_2) = \frac{1}{2}\int_\Omega (f-u_1-u_2)^2 + \alpha \int_\Omega \lvert\nabla u_1\rvert + \beta \int_\Omega \lvert\Delta u_2\rvert$
5	TVL	$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \lvert\nabla u\rvert + \beta \int_\Omega \lvert\Delta u\rvert$
6	INFCON	$E(u_1, u_2) = \frac{1}{2}\int_\Omega (f-u_1-u_2)^2 + \alpha \int_\Omega \lvert\nabla u_1\rvert + \beta \int_\Omega \lvert\nabla^2 u_2\rvert$
7	TVBH	$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \lvert\nabla u\rvert + \beta \int_\Omega \lvert\nabla^2 u\rvert$
8	TGV '	$E\,(u,p) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \lvert\nabla u - p\rvert + \beta \int_\Omega \lvert\varepsilon(p)\rvert$
9	TC	$E(u) = \frac{1}{2}\int_\Omega (u-f)^2 + \alpha \int_\Omega \left\lvert div\left(\frac{\nabla u}{\lvert\nabla u\rvert}\right)\right\rvert$

4.1. Comparison 1

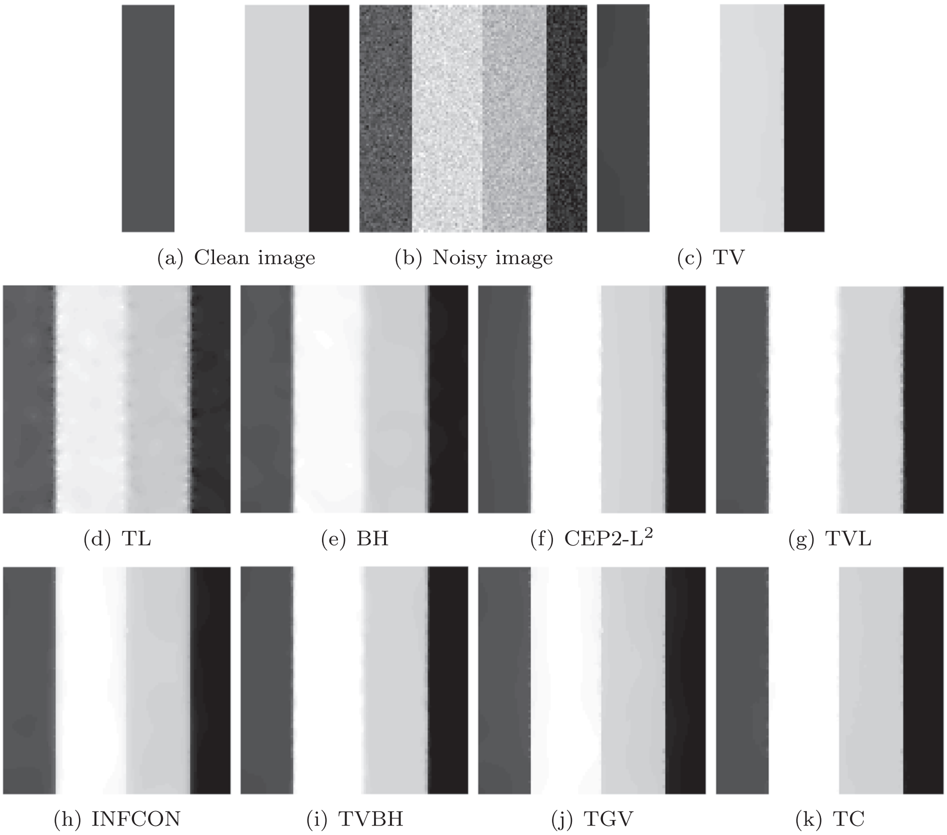
In Figure 2, we test the edge preservation ability of different models on a piecewise constant image (a) through visual inspection. From (c) to (k), it is clear that the first-order TV and second-order TC models preserve the edges of the strips perfectly. Their denoised results look almost same as the original image (a). TGV also gives very good edge-preservation result. However, it produces additional lines on the denoised stripes because it approximates image (a) with piecewise affine function. Similar visual effect can be observed on results by INFCON and CEP2-L^2 models. Image (e) illustrates directly applying BH regulariser can blur the edges of object, whilst image (b) shows TL model using Laplace regulariser is the worst method for edge preservation. Its resulting denoised image takes on some undesirable spikes. By incorporating first-order TV regulariser into pure second-order BH and TL models, TVL and TVBH models can improve the denoised quality as can be seen from images (g) and (i) qualitatively and Table II quantitatively. However, the object edges are still smeared by these two models.

In Figure 3, we present the associated residual images $f - u$ for all the models, where f is noisy image Figure 2(b) and u denotes each corresponding denoised one in Figure 2. Image (j) by TC model constains almost all noise, whilst the other eight images contains more or less structural information of Figure 2(a), which means some contrast information has been lost in the denoised images handled by the eight models. Figure 3 illustrates TC model is the best in preserving image contrast.

In Figure 4, the middle slices of the examples in Figure 2 are presented. We can see that the denoised slice curve (red) in (j) almost overlaps with the original noise free slice curve (blue), which demonstrates the preservation of image contrast as well as corners of TC model. Because of the contrast loss, the red curve generated by TV model (b) sightly deviates from original one along vertical directions. Compared with TV and TC models, the other seven high-order models impose too much smoothness on the curve edges, leading to less pleasant fitting results.

4.2. Comparison 2

In Figure 5, we test the smoothness preservation ability of different models on a piecewise smooth image (a). To do so, the original image (a) is corrupted with Gaussian noise of variance 0.005 and 0.015, respectively. The noise has been produced with MATLAB's built in function *imnoise*.



(a) Clean image (b) Noisy image (c) TV

(d) TL (e) BH (f) CEP2-L^2 (g) TVL

(h) INFCON (i) TVBH (j) TGV (k) TC

Figure 2. Edge preservation ability test. All the methods in Table I are performed on noisy image (b), and their corresponding results are demonstrated from (c) to (k), respectively.

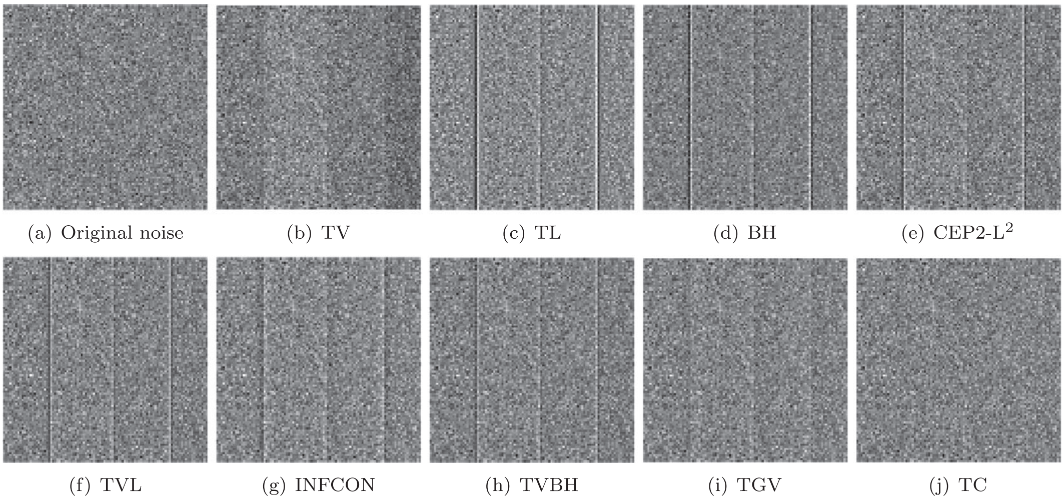
Table II. Comparison of PSNR, SNR, RMSE and SSIM using different methods on images of 2 different noise variances.

	Figure 6: noise variance 0.005.				Figure 8: noise variance 0.015.			
	PSNR	SNR	RMSE	SSIM	PSNR	SNR	RMSE	SSIM
Noisy image	23.6978	19.1788	7.9693	0.3278	19.1072	14.5882	9.0784	0.1993
TV	32.6621	26.1032	3.6633	0.9015	27.3922	23.0732	6.2329	0.8689
TL	27.8122	23.2933	3.9571	0.8812	24.3702	19.8512	6.4307	0.8269
BH	30.0447	25.5257	3.7131	0.8985	25.9380	21.4190	6.3468	0.8517
CEP2-L^2	32.2822	27.7633	3.6491	0.9043	28.1452	23.6263	6.0445	0.8712
TVL	30.6670	26.1480	3.4764	0.9049	27.7093	23.1903	6.1722	0.8733
INFCOV	32.9872	28.4682	3.3737	0.9088	28.4539	23.9349	5.5835	0.8742
TVBH	32.3548	27.8358	3.2725	0.9118	28.2275	23.8085	5.8195	0.8778
TGV	**34.4020**	**29.8830**	**3.2120**	**0.9243**	**29.2505**	**24.7316**	**5.4590**	**0.9006**
TC	34.2254	29.7064	3.2412	0.9039	28.0062	23.7986	5.7840	0.8798

In Figure 6, the denoised results on Figure 5(b) by different methods are presented. One can easily find that image (a) by TV model are composed of jagged appearance which is staircase effect. In addition, the cleaned results shown from (d)-(g) illustrate these high order models that involve TV regulariser also lead to staircase. Even though TC model cannot completely remove the side effect, which can be observed from image (i). The pure second order TL and BH do not produce unfavourable staircase artifact. However, their results are not very visually satisfactory. For TL model, there exists some spikes at the edges of the geometrical shapes, whilst BH slightly blurs the object edges. The visual effects show TGV totally eliminates staircase effect without smearing edges in the image. Thus it gives best smoothness preservation result (i.e. highest PSNR, SNR, SSIM and lowest RMSE shown in Table II).

In Figure 7, we show the middle slices of the examples in Figure 6. It is obvious to see TL, BH and TGV models produce smoother contour than TV, CEP2-L^2, TVL, INFCON, TVBH and TC models.

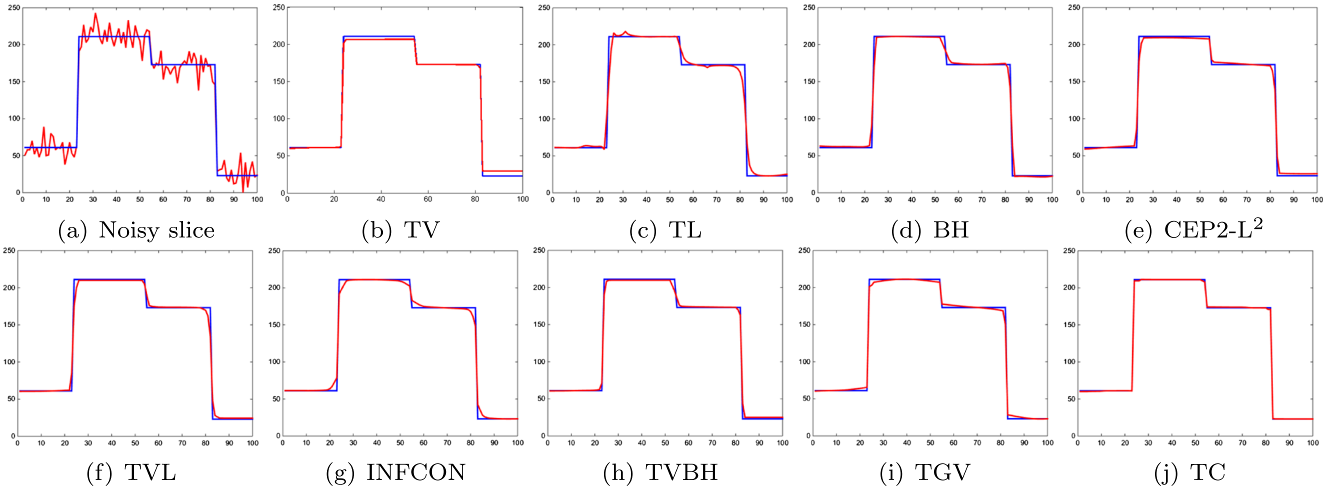
In Figure 8, the denoised results on Figure 6(c) are demonstrated. As can be seen, as noise increases, the differences between each methods are much clearer. TC and BH models give more blurred results while the staircase effect produced by TV, CEP2-L^2, TVL, INFCON,



(a) Original noise (b) TV (c) TL (d) BH (e) CEP2-L^2

(f) TVL (g) INFCON (h) TVBH (i) TGV (j) TC

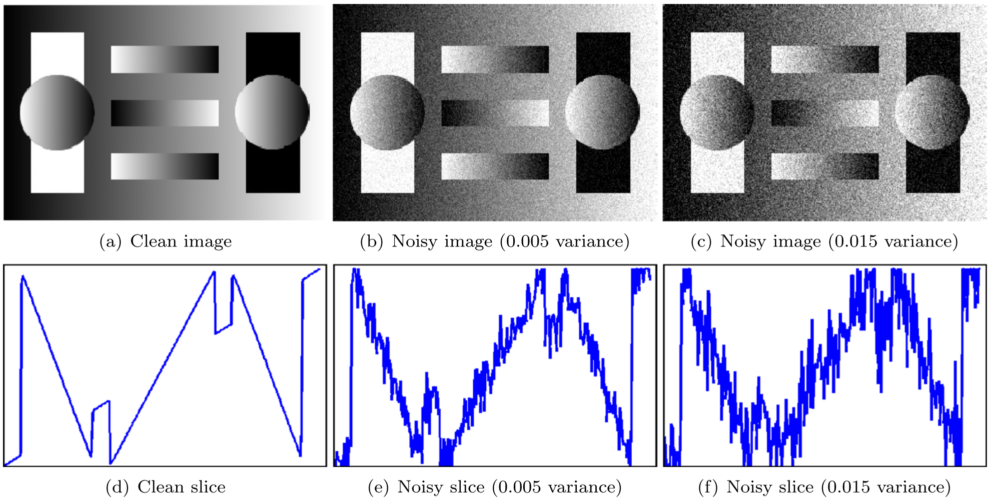
Figure 3. Residual images of the denoised examples in Figure 2 by different models.



(a) Noisy slice (b) TV (c) TL (d) BH (e) CEP2-L^2

(f) TVL (g) INFCON (h) TVBH (i) TGV (j) TC

Figure 4. Plots of middle slices of the denoised images in Figure 2 by different models.



(a) Clean image (b) Noisy image (0.005 variance) (c) Noisy image (0.015 variance)

(d) Clean slice (e) Noisy slice (0.005 variance) (f) Noisy slice (0.015 variance)

Figure 5. Smoothness preservation ability test. The images in second row are the middle slices of these in the first row.

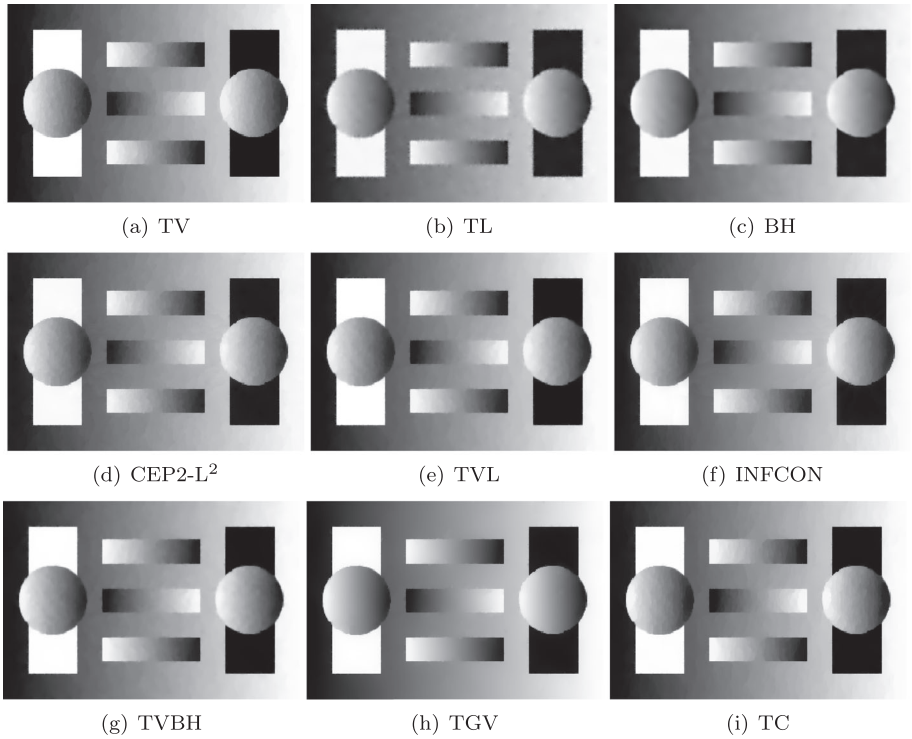


Figure 6. Performance comparison of different methods on Figure 5(b).

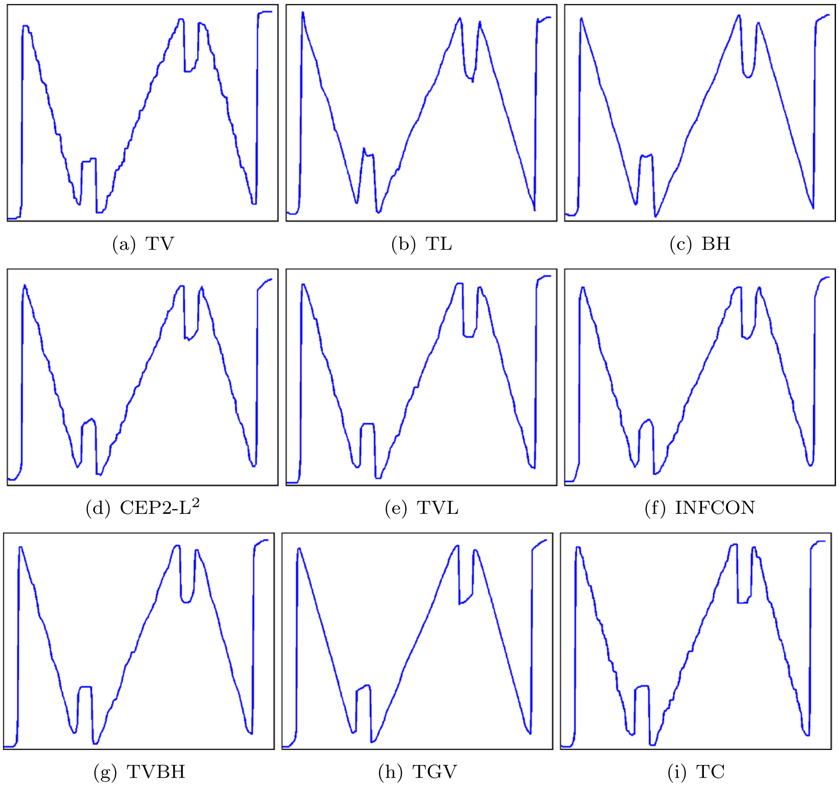


Figure 7. Plots of middle slices of the denoised images in Figure 6 by different models.

TVBH and TC models become stronger. The TGV again presents a visually more pleasant smooth result. The quantitative metrics in Table II also validate the better smoothness preservation capability of TGV model over the other 8 variational models. Moreover, in Figure 9, the middle slices of examples in Figure 8 can provide further more detailed comparison between different methods.

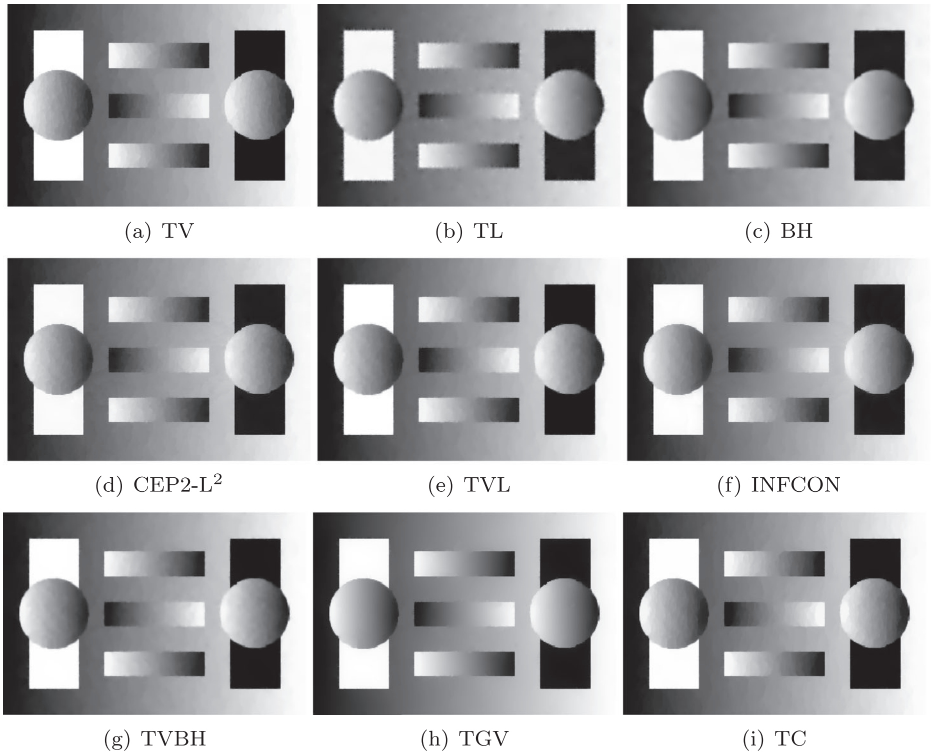


Figure 8. Performance comparison of different methods on Figure 5(c).

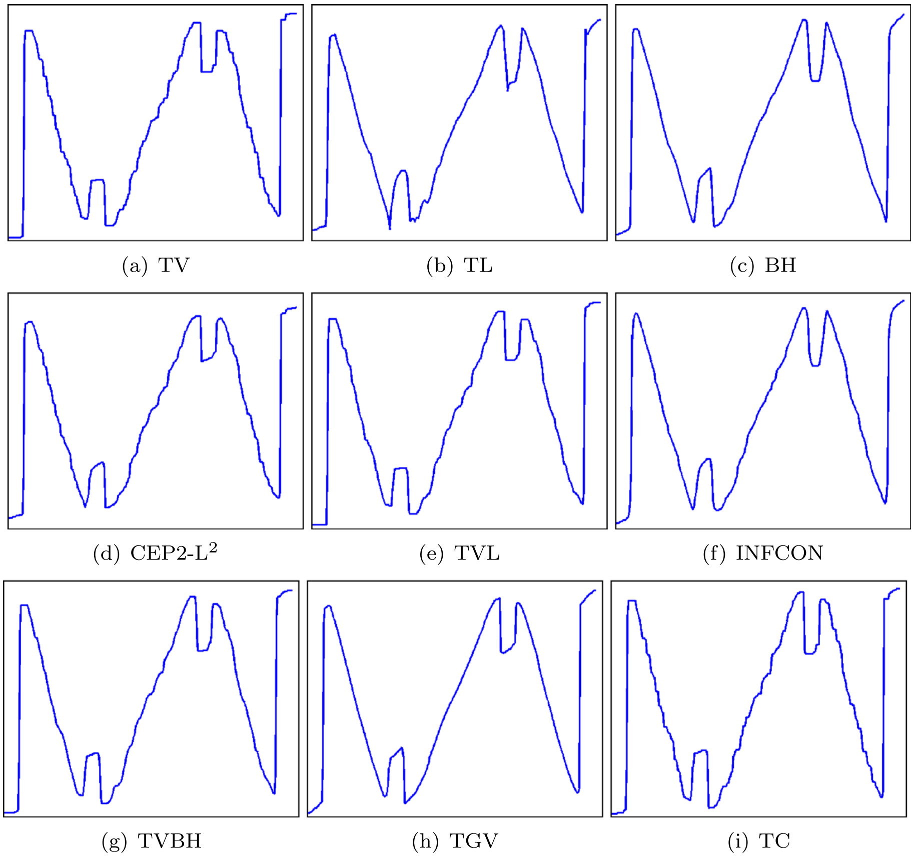
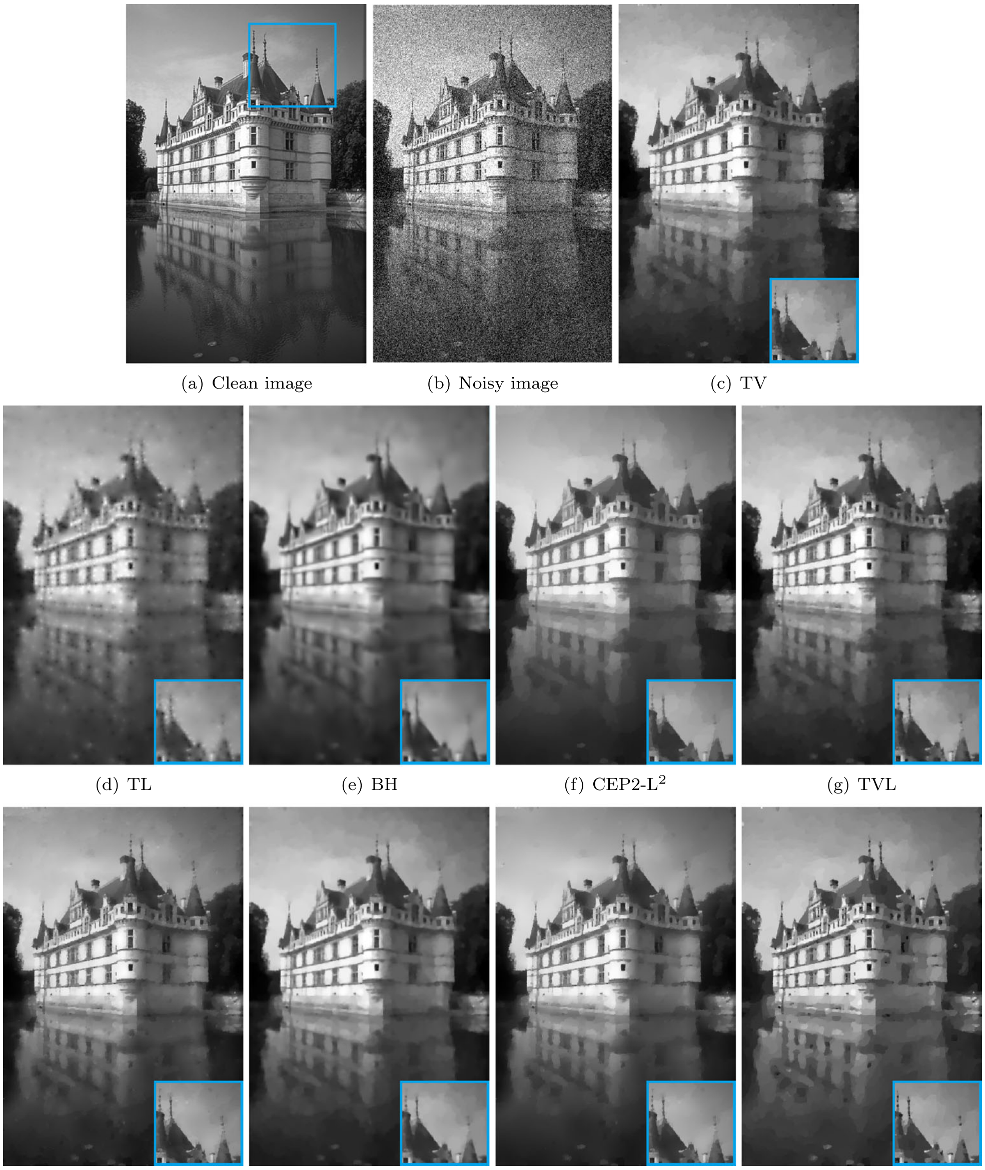


Figure 9. Plots of middle slices of the denoised images in Figure 8 by different models.


(a) Clean image (b) Noisy image (c) TV

(d) TL (e) BH (f) CEP2-L^2 (g) TVL

(h) INFCON (i) TVBH (j) TGV (k) TC

Figure 10. Performance comparison of different methods on real noisy image Figure 10(b) (size 480 × 320).

4.3. Comparison 3

In the previous comparison, the real clean image Figure 10(a) is used to further evaluate different methods. In order to make the denoising evaluation more objective and convincing, the clean image is corrupted by additive Gaussian noise with zero-mean and different variance. Six variance values from 0.005 to 0.03 at 0.005 intervals are used.

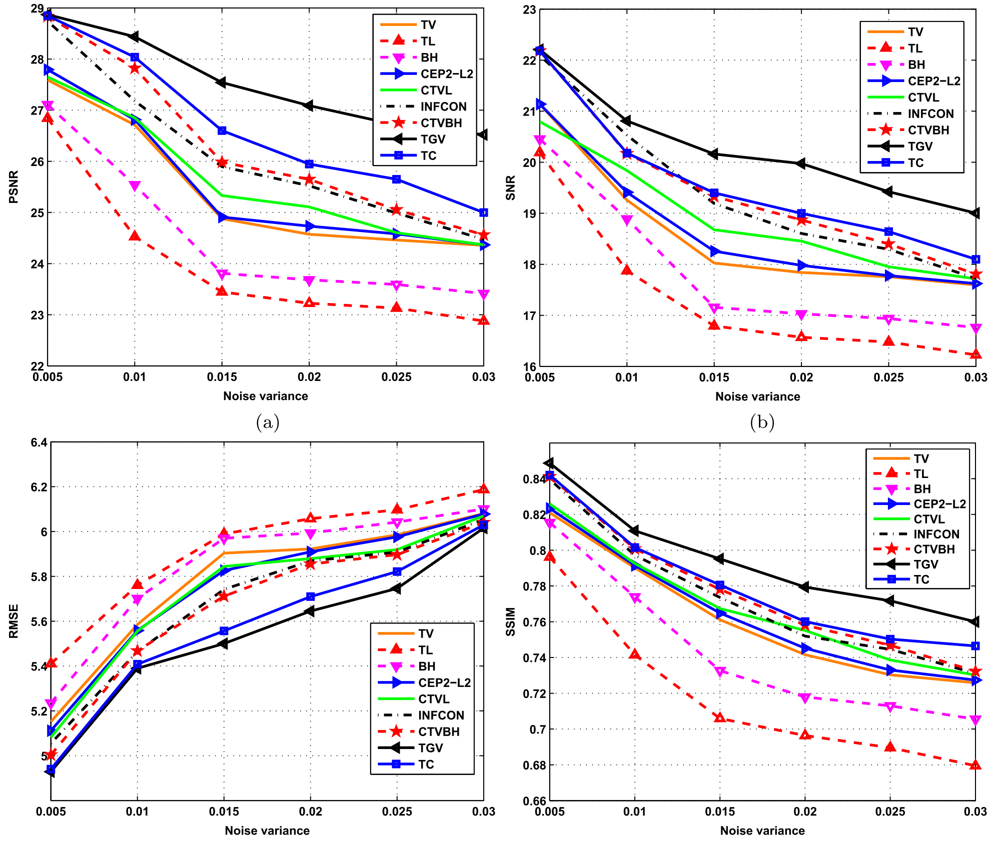
In Figure 10, we show the cleaned results of (b) which contains Gaussian noise with 0.015 variance. For each denoised image, the corresponding close-up region is displayed on the bottom right corner for detailed comparison. The visual inspection illustrates TV, CEP2-L^2, TVL, INFCON,TVBH and TC models yields staircase effect, whilst TL and BH models smear the object edges in the resulting images. TGV model again gives a visually more pleasant result, total eliminating staircase while well preserving sharp edges. Hence, TGV used as a penalty functional prefers images which appear more natural and leads to a more faithful denoising for real images.

Table III contains the four evaluation metrics for quantitative comparison of different methods on the clean image Figure 10(a) with 6 different noise variance. In order to easily assess these metrics, Figure 11 provide, for each of the 6 experiments, plots of PSNR, SNR, RMSE and SSIM against increasing levels of noise. According to Figure 11, these methods can be quantitatively ranked in a decreasing order as TGV > TC > TVBH > INFCON > TVL > CEP2-L^2 > TV > BH > TL.

Table IV demonstrates computational efficiency of different methods. For all the methods, we set stopping criteria as $|E^k - E^k - 1|/E^k \leq \epsilon$, where E is the value of energy functional of each model, and ϵ is a small tolerance used to stop iteration. $\epsilon = 10^{-5}$ in all cases. To clearly compare the data in Table IV, we draw the bar charts in Figure 12, where each bar corresponds to a average value of the

Table III. Comparison of PSNR, SNR, RMSE and SSIM using different methods on image Figure 10(a) with 6 different noise variances.

	PSNR test						SNR test					
Noise variance	0.005	0.01	0.015	0.02	0.025	0.03	0.005	0.01	0.015	0.02	0.025	0.03
Noisy image	23.1360	20.2415	18.5938	17.4469	16.5558	15.8704	16.4828	13.5883	11.9405	10.7936	9.9026	9.2171
TV	27.5911	26.7111	24.8781	24.5730	24.4611	24.3582	20.1908	19.2579	18.0248	17.8397	17.7578	17.5949
TL	26.8441	24.5273	23.4462	23.2245	23.1309	22.8787	20.1908	17.8740	16.7929	16.5712	16.4776	16.2254
BH	27.1086	25.5418	23.8114	23.6834	23.5913	23.4151	20.4554	18.8885	17.1582	17.0302	16.9380	16.7619
CEP2-L^2	27.7928	26.8130	24.9105	24.7330	24.5808	24.3667	21.1396	19.4097	18.2572	17.9797	17.7776	17.6234
TVL	27.6490	26.8506	25.3305	25.1088	24.6049	24.3680	20.7958	19.8373	18.6773	18.4555	17.9517	17.7147
INFCON	28.7433	27.1825	25.8957	25.5212	24.9891	24.4739	22.0900	20.5293	19.1924	18.6034	18.2958	17.7206
TVBH	28.8336	27.8218	25.9831	25.6466	25.0526	24.5616	22.1803	20.1685	19.3298	18.8680	18.3994	17.8083
TGV	**28.8697**	**28.4359**	**27.5416**	**27.0896**	**26.7111**	**26.5212**	**22.2064**	**20.8092**	**20.1619**	**19.9746**	**19.4216**	**19.0020**
TC	28.8479	28.0366	26.6016	25.9456	25.6491	24.9981	22.1824	20.1791	19.4009	18.9976	18.6411	18.0981
	RMSE test						SSIM test					
Noise variance	0.005	0.01	0.015	0.02	0.025	0.03	0.005	0.01	0.015	0.02	0.025	0.03
Noisy image	8.4469	9.2784	9.6300	9.8620	10.0313	10.1224	0.4166	0.3138	0.2617	0.2296	0.2067	0.1890
TV	5.1521	5.5854	5.9036	5.9226	5.9854	6.0803	0.8210	0.7901	0.7611	0.7416	0.7303	0.7259
TL	5.4106	5.7604	5.9893	6.0573	6.0969	6.1873	0.7962	0.7415	0.7059	0.6964	0.6896	0.6795
BH	5.2352	5.7011	5.9699	5.9932	6.0424	6.1005	0.8155	0.7739	0.7327	0.7179	0.7130	0.7055
CEP2-L^2	5.1115	5.5586	5.8260	5.9095	5.9762	6.0782	0.8234	0.7915	0.7649	0.7452	0.7330	0.7274
TVL	5.0821	5.5556	5.8444	5.9194	5.9214	6.0699	0.8259	0.7926	0.7674	0.7551	0.7386	0.7302
INFCON	5.0549	5.4648	5.7420	5.8689	5.9082	6.0534	0.8396	0.7972	0.7736	0.7520	0.7446	0.7311
TVBH	5.0036	5.4678	5.7115	5.8550	5.8968	6.0401	0.8412	0.8008	0.7781	0.7581	0.7469	0.7323
TGV	**4.9287**	**5.3898**	**5.5003**	**5.6452**	**5.7459**	**6.0159**	**0.8487**	**0.8109**	**0.7952**	**0.7794**	**0.7717**	**0.7600**
TC	4.9403	5.4081	5.5569	5.7095	5.8216	6.0299	0.8420	0.8015	0.7806	0.7601	0.7503	0.7465



(a)

(b)

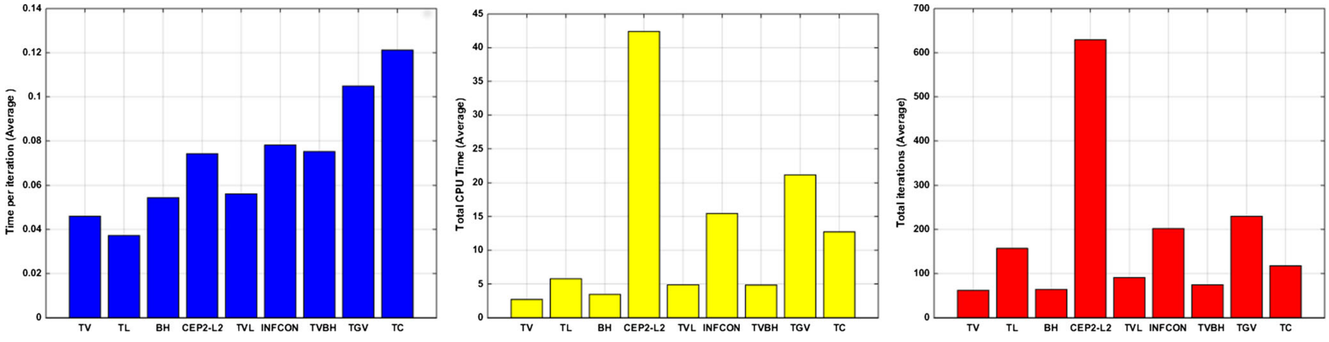
(c)

(d)

Figure 11. Performance plots of different methods applied to real data with different noise variance. (a) PSNR index; (b) SNR index; (c) RMSE index; (d) SSIM index.

Table IV. Comparison of time per iteration, total CPU time and total iterations of different models on image Figure 10(a) with 6 different noise variances.

Noise variance		TV	TL	BH	CEP2-L^2	TVL	INFCON	TVBH	TGV	TC
	Time per iteration (s)	0.0471	0.0390	0.0551	0.0739	0.0581	0.0752	0.0741	0.1003	0.1278
0.005	Total CPU time (s)	2.6864	3.6391	3.4713	27.5693	5.1863	17.8236	3.6246	13.5646	7.6698
	Total iterations	62	100	63	415	96	235	57	147	70
	Time per iteration (s)	0.0463	0.0340	0.0628	0.0781	0.0515	0.0837	0.0747	0.0956	0.1070
0.01	Total CPU time (s)	2.7551	4.8098	3.4373	49.5393	4.8496	18.9164	3.9162	20.2070	22.8778
	Total iterations	61	121	57	653	90	239	61	218	215
	Time per iteration (s)	0.0457	0.0386	0.0499	0.0659	0.0566	0.0738	0.0735	0.1034	0.1175
0.015	Total CPU time (s)	2.6152	5.5070	2.8969	42.4323	4.7385	15.1736	4.3204	18.7648	12.6513
	Total iterations	61	156	55	654	91	202	67	204	117
	Time per iteration (s)	0.0447	0.0368	0.0536	0.0702	0.0576	0.0765	0.0734	0.1113	0.1241
0.02	Total CPU time (s)	2.9139	6.4935	3.4635	45.9972	4.5916	13.3800	5.6789	19.5086	8.3592
	Total iterations	62	177	65	702	87	176	86	212	77
	Time per iteration (s)	0.0463	0.0379	0.0528	0.0815	0.0577	0.0816	0.0760	0.1083	0.1303
0.025	Total CPU time (s)	2.5942	7.3905	3.4833	45.6954	4.8650	12.8345	5.5162	30.8093	14.2515
	Total iterations	60	203	69	697	90	170	83	335	130
	Time per iteration (s)	0.0456	0.0368	0.0520	0.0754	0.0552	0.0779	0.0793	0.1102	0.1205
0.03	Total CPU time (s)	2.6202	6.6830	3.9454	43.2448	4.9202	14.5317	5.9294	24.2398	10.4639
	Total iterations	62	185	74	655	91	189	92	262	96



(a) (b) (c)

Figure 12. Bar expression of the experimental data in Table IV (i.e. average denoising results of 6 noise variance cases). (a) Time per iteration of different methods; (b) Total CPU time of different methods; (c) Total iterations of different methods.

denoising results produced by its corresponding method on different noise level. Figure 12 illustrates CEP2-L^2 is the slowest method, while TV is the fastest method. Furthermore, TL, INFCON, TGV and TC models are less efficient than BH, TVL and TVBH models.

4.4. Comparison 4

In addition to the proposed split Bregman algorithm, there exist quite a few of approaches, such as the augmented Lagrangian algorithm [10, 15, 22, 23, 35, 36, 43], the primal-dual algorithm [19, 44–47] and etc., to address the difficulties of minimising the high order energy functionals. For example, the authors in [15, 22, 23, 35, 43] used the augmented Lagrangian algorithm to solve the TV, BH, TVL and TC models, while the primal dual has been developed for the TV and TGV models in [44, 46, 47]. Note that both the augmented Lagrangian and split Bregman algorithms employ the alternating direction method of multipliers (ADMM) to optimise their objective functionals and the convergence of ADMM is always guaranteed. In [43], the authors has proven that the two algorithms are in fact equivalent. Thus, in the experiment, we only compare the proposed split Bregman algorithm with the popular primal dual method in terms of both the quality of the result and computational speed. To do so, we choose two representative models from the Table I, i.e. the first order TV model and the second order TGV model, and then apply the split Bregman and primal dual algorithms for each of the two models.

Figure 13 shows the denoised results of Figure 10(b) using the two algorithms. (a) and (b) are the TV denoised results, where one can observe that neither the split bregman nor primal dual can eliminate the staircase artifact. However, the artifact can be totally suppressed by the TGV model, as shown in (c) and (d). Note that the quality of the denoised results by the same model looks

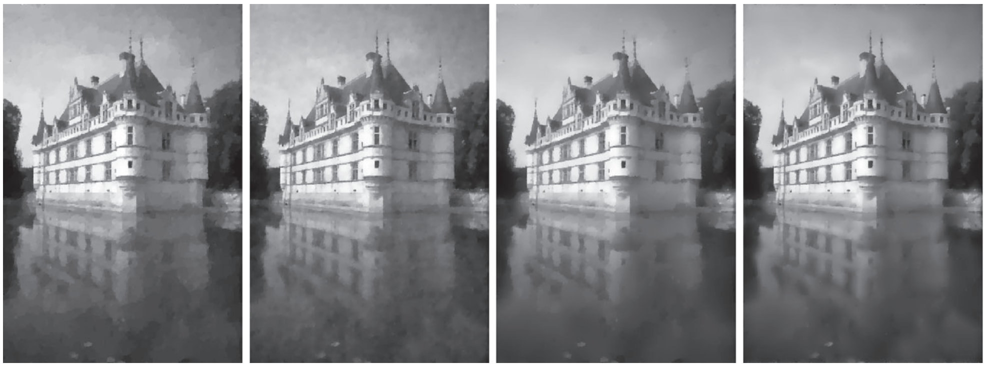


Figure 13. Performance comparison of the split Bregman and primal dual algorithms on Figure 10(b). (a) and (b) are the TV denoised results using the split Bregman and primal dual, respectively; (c) and (d) are the TGV denoised results using the split Bregman and primal dual, respectively.

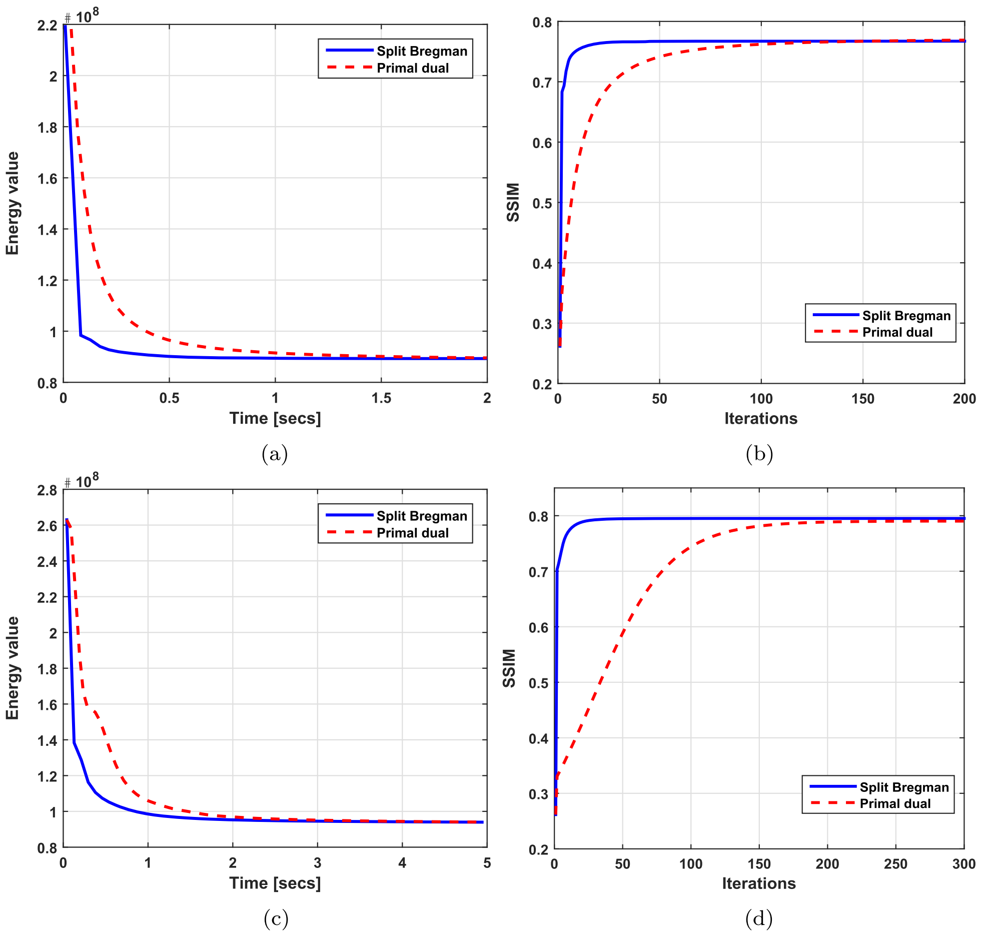


Figure 14. Comparison of the convergence rate of the split Bregman and primal dual algorithms for the examples in Figure 13. (a) and (b) compare the convergence speed of the two algorithms for the TV model; (c) and (d) compare the convergence speed of the two algorithms for the TGV model.

almost identical. Therefore, it can be concluded that the different numerical algorithms do not significantly influence the final image restoration performance. In contrast, the obvious differences in image quality can be easily found from different models.

In Figure 14, we test the convergence rate of the split Bregman and primal dual algorithms. In (a) and (c), the energy values against CPU runtimes are plotted to compare the energy decreasing rate of the two algorithms. The energy values in (a) and (c) are calculated from the TV (1.1) and TGV (1.9) functionals, and the horizontal axis in (a) and (c) represent the total CPU time (number of iterations times CPU time per iteration). As can be seen from (a) and (c), the numerical energy by the split Bregman use much less time than that by the primal dual algorithm in reaching to a steady state. Moreover, in (b) and (d), the SSIM values are plotted to examine the quality of the denoised images against the number of iterations. Compared with the proposed split Bregman, the primal dual can obtain nearly same SSIM value but with much more iterations. This experiment validates the split Bregman is faster than the primal dual, and the final results by the two algorithms are quantitatively identical.

5. Conclusion

In this paper, several high-order variational models are presented, and their application for image denoising is demonstrated. We present the detailed descretisation process based on discrete finite different scheme and numerical implementation of fast Fourier transform (FFT)-based split Bregman algorithm for solving the equations. We draw following conclusions about the advantages and disadvantages of these models by extensive comparative experiments in last section.

Among all the models, the first-order TV and second-order TC models are the best in preserving edges of object in the piecewise constant image. However, TV causes image contrast loss while TC not only maintains contrast but also object corners. TGV model yields best denoising results for the piecewise smooth image. It removes the staircase effect associated with TV regulariser. However, TGV is more computational expensive than other models. In addition, we notice that those high order models that involve TV regulariser (i.e. CEP2-L^2, TVL, INFCON and TVBH) produce the staircase artifact. Directly applying Laplacian regulariser (i.e. TL model) or bounded Hessian regulariser (i.e. BH model) cannot achieve desirable denoising result. However, combining them with TV regulariser (i.e. CEP2-L^2, TVL, INFCON, and TVBH) can improve the quality of resulting image. Moreover, high-order models that use bounded Hessian regulariser (i.e. BH, INFCON and TVBH) outperform their counterparts that use Laplacian regulariser (i.e. TL, CEP2-L^2 and TVL), and the former are more efficient than the latter.

Lastly, the proposed split Bregman algorithm is more efficient than the primal dual algorithm, and applying these two algorithms to different image restoration models do not significantly influence the quality of the result.

Acknowledgements

This project is supported by National Natural Science Foundation of China (No.61305045, No.61303079 and No.61170106), Qingdao Science and Technology Development Project (No. 13-1-4-190-jch).

References

1. Chan TF, Shen JH. *Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods*. SIAM: Philadelphia, USA, 2005.
2. Aubert G, Kornprobst P. *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*. Springer Science and Business Media: New York, USA, 2006,147.
3. Paragios N, Chen YM, Faugeras O. *Handbook of Mathematical Models in Computer Vision*. Springer Science and Business Media: New York, USA, 2006.
4. Scherzer O. *Handbook of Mathematical Methods in Imaging*. Springer Science and Business Media: New York, USA, 2011,1.
5. Rudin L, Osher S, Fatemi E. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena* 1992; **60**(1):259–268.
6. You YL, Kaveh M. Fourth-order partial differential equations for noise removal. *IEEE Transactions on Image Processing* 2000; **9**(10):1723–1730.
7. Lysaker M, Lundervold A, Tai XC. Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time. *IEEE Transactions on Image Processing* 2003; **12**(12):1579–1590.
8. Scherzer O. Denoising with higher order derivatives of bounded variation and an application to parameter estimation. *Computing* 1998; **60**(1):1–27.
9. Hinterberger W, Scherzer O. Variational methods on the space of functions of bounded hessian for convexification and denoising. *Computing* 2006; **76**(1):109–133.
10. Lai RJ, Tai XC, Chan TF. A ridge and corner preserving model for surface restoration. *SIAM Journal on Scientific Computing* 2013; **35**(2):A675–A695.
11. Bergounioux M, Piffet L. A second-order model for image denoising. *Set-Valued and Variational Analysis* 2010; **18**(3–4):277–306.
12. Chan TF, Esedoglu S, Park F. A fourth order dual method for staircase reduction in texture extraction and image restoration problems. *2010 17th IEEE International Conference on Image Processing (ICIP)*, Hongkong, China, 2010, 4137–4140.
13. Zheng SX, Pan ZK, Jiang CX, Wang GD. A new fast algorithm for image denoising, 2013.
14. Wang GD, Xu J, Dong Q, Pan ZK. Active contour model coupling with higher order diffusion for medical image segmentation. *International Journal of Biomedical Imaging* 2014; **2014**:1–8.
15. Chan RH, Liang HX, Wei SH, Nikolova M, Tai XC. High-order total variation regularization approach for axially symmetric object tomography from a single radiograph. *Inverse Problems and Imaging* 2015; **9**(1):55–77.
16. Papafitsoros K, Schönlieb CB, Sengul B. Combined first and second order total variation inpainting using split Bregman. *Image Processing On Line* 2013; **2013**:112–136.
17. Papafitsoros K, Schönlieb CB. A combined first and second order variational approach for image reconstruction. *Journal of Mathematical Imaging and Vision* 2014; **48**(2):308–338.
18. Chambolle A, Lions PL. Image recovery via total variation minimization and related problems. *Numerische Mathematik* 1997; **76**(2):167–188.
19. Bredies K, Kunisch K, Pock T. Total generalized variation. *SIAM Journal on Imaging Sciences* 2010; **3**(3):492–526.
20. Goldluecke B, Cremers D. Introducing total curvature for image processing. *2011 IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, 2011, 1267–1274.
21. Zhu W, Chan TF. Image denoising using mean curvature of image surface. *SIAM Journal on Imaging Sciences* 2012; **5**(1):1–32.
22. Zhu W, Tai XC, Chan TF. Augmented lagrangian method for a mean curvature based image denoising model. *Inverse Problems and Imaging* 2013; **7**(4):1409–1432.
23. Tai XC. *Fast Numerical Schemes Related to Curvature Minimization: a Brief and Elementary Review*. Courbure Discrète: Théorie et Applications: Marseille, France, 2013,17.
24. Nitzberg M, Mumford D. The 2.1-d sketch. *Proceedings, Third International Conference on Computer Vision*, Vol. 1990, Osaka, Japan, 1990, 138–144.

25. Masnou S, Morel JM. Level lines based disocclusion. *Proceedings. 1998 International Conference on Image Processing, 1998. ICIP 98*, Chicago, USA, 1998, 259–263.
26. Chan TF, Shen JH. Nontexture inpainting by curvature-driven diffusions. *Journal of Visual Communication and Image Representation* 2001; **12**(4): 436–449.
27. Esedoglu S, Shen JH. Digital inpainting based on the mumford–shah–euler image model. *European Journal of Applied Mathematics* 2002; **13**(04): 353–370.
28. Zhu W, Chan TF, Esedoglu S. Segmentation with depth: a level set approach. *SIAM Journal on Scientific Computing* 2006; **28**(5):1957–1973.
29. Kang SH, Zhu W, Jianhong J. Illusory shapes via corner fusion. *SIAM Journal on Imaging Sciences* 2014; **7**(4):1907–1936.
30. Goldstein T, Osher S. The split bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences* 2009; **2**(2):323–343.
31. Duan JM, Pan ZK, Zhang BC, Liu WQ, Tai XC. Fast algorithm for color texture image inpainting using the non-local ctv model. *Journal of Global Optimization* 2015; **62**(4):853–876.
32. Duan JM, Pan ZK, Yin XF, Wei WB, Wang GD. Some fast projection methods based on chan-vese model for image segmentation. *EURASIP Journal on Image and Video Processing* 2014; **2014**(1):1–16.
33. Duan JM, Qiu ZW, Lu WQ, Wang GD, Pan ZK, Bai L. An edge-weighted second order variational model for image decomposition. *Digital Signal Processing* 2016; **49**:162–181.
34. Duan JM, Lu WQ, Tench C, Gottlob I, Proudlock F, Samani NN, Bai L. Denoising optical coherence tomography using second order total generalized variation decomposition. *Biomedical Signal Processing and Control* 2016; **24**:120–127.
35. Wu CL, Tai XC. Augmented lagrangian method, dual methods, and split bregman iteration for rof, vectorial tv, and high order models. *SIAM Journal on Imaging Sciences* 2010; **3**(3):300–339.
36. Tai XC, Hahn J, Chung GJ. A fast algorithm for euler's elastica model using augmented lagrangian method. *SIAM Journal on Imaging Sciences* 2011; **4**(1):313–344.
37. Wang YL, Yang JF, Yin WT, Zhang Y. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences* 2008; **1**(3):248–272.
38. Duan JM, Tench C, Gottlob I, Proudlock F, Bai L. New variational image decomposition model for simultaneously denoising and segmenting optical coherence tomography images. *Physics in Medicine and Biology* 2015; **60**(22):8901–8922.
39. Duan JM, Ding YC, Pan ZK, Yang J, Bai L. Second order mumford-shah model for image denoising. *2015 IEEE International Conference on Image Processing (ICIP)*, Quebec, Canada, 2015, 547–551.
40. Getreuer P. Rudin-Osher-Fatemi total variation denoising using split Bregman. *Image Processing On Line* 2012; **2**:74–95.
41. Ng MK, Chan RH, Tang WC. A fast algorithm for deblurring models with Neumann boundary conditions. *SIAM Journal on Scientific Computing* 1999; **21**(3):851–866.
42. Getreuer P. Total variation deconvolution using split Bregman. *Image Processing On Line* 2010; **2**(1):158–174.
43. Tai XC, Wu CL. *Augmented Lagrangian Method, Dual Methods and Split Bregman Iteration for Rof Model*. Scale Space and Variational Methods in Computer Vision Springer: Voss, Norway, 2009, 502–513.
44. Chambolle A. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision* 2004; **20**(1–2):89–97.
45. Pock T, Chambolle A. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. *2011 IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, 2011, 1762–1769.
46. Chambolle A, Pock T. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision* 2011; **40**(1):120–145.
47. Knoll F, Bredies K, Pock T, Stollberger R. Second order total generalized variation (tgv) for mri. *Magnetic Resonance in Medicine* 2011; **65**(2):480–491.