




Applying smoothing technique and semi-proximal ADMM for image deblurring

Caiying Wu¹ · Xiaojuan Chen¹ · Qiyu Jin¹ · Jein-Shan Chen² 

Received: 24 March 2022 / Revised: 30 June 2022 / Accepted: 7 September 2022

© The Author(s) under exclusive licence to Istituto di Informatica e Telematica (IIT) 2022

Abstract

We present a new approach which combines smoothing technique and semi-proximal alternating direction method of multipliers for image deblurring. More specifically, in light of a nondifferentiable model, which is indeed of the hybrid model of total variation and Tikhonov regularization models, we consider a smoothing approximation to conquer the disadvantage of nonsmoothness. We employ four smoothing functions to approximate the hybrid model and build up a new model accordingly. It is then solved by semi-proximal alternating direction method of multipliers. The algorithm is shown globally convergent. Numerical experiments and comparisons affirm that our method is an efficient approach for image deblurring.

Keywords TV regularization · Image restoration · SP-ADMM · Smoothing function

Mathematics Subject Classification 49K10 · 90C90

1 Introduction

In this paper, the target problem is image deblurring, which has wide applications in image processing tasks, such as image segmentation, edge detection and pattern recognition. Tremendous amount of articles related to this topic can be found

✉ Jein-Shan Chen
jschen@math.ntnu.edu.tw

Caiying Wu
wucaiyingyun@163.com

Xiaojuan Chen
chenxiaojuan05@163.com

Qiyu Jin
qyjin2015@aliyun.com

¹ School of Mathematical Sciences, Inner Mongolia University, Hohhot 010021, China

² Department of Mathematics, National Taiwan Normal University, Taipei 11677, Taiwan

in the literature, and hence we do not repeat its importance and realistic applications here. Therefore, we look into its mathematical model directly and convey our new idea for tackling this problem. The image deblurring model is described as follows:

$$f = A \otimes u + \varepsilon, \quad (1)$$

where $u \in \mathbb{R}^{m \times n}$ is the unknown clear image, $f \in \mathbb{R}^{m \times n}$ is the observed degraded image, $\varepsilon \in \mathbb{R}^{m \times n}$ is the Gaussian white noise, A is the blurring operator, and $A \otimes u$ is the convolution of A with u meaning

$$[A \otimes u](i, j) = \sum_{(s, t) \in \Omega} A[(i, j) - (s, t)] \times u(s, t), \quad \Omega = \{1, 2, \dots, m\} \times \{1, 2, \dots, n\}.$$

For each 2D image u , $u(i, j)$ denotes the image value at pixel $(i, j) \in \Omega$. Restoring the unknown image u from the degraded image f is an ill-posed problem. In order to deal with the image deblurring problem (1), many researchers consider the associated regularized optimization problem [20]:

$$\min_u \frac{\mu}{2} \|A \otimes u - f\|_2^2 + \|\nabla u\|_1, \quad (2)$$

where ∇ denotes the gradient operator, $\nabla u = [\nabla_x u, \nabla_y u] \in \mathbb{R}^{2m \times n}$ is the gradient in the discrete setting with $\nabla u(i, j) = (\nabla_x u(i, j), \nabla_y u(i, j))$, and $\nabla_x u, \nabla_y u$ denote the horizontal and vertical first order differences with Dirichlet boundary condition respectively, which are defined as follows:

$$\begin{aligned} \nabla_x u(i, j) &= \begin{cases} u(i+1, j) - u(i, j) & \text{if } i < m, \\ 0 & \text{if } i = m, \end{cases} \\ \nabla_y u(i, j) &= \begin{cases} u(i, j+1) - u(i, j) & \text{if } j < m, \\ 0 & \text{if } j = m. \end{cases} \end{aligned}$$

Here $\|\nabla u\|_1$ means the l_1 -norm of ∇u , that is,

$$\|\nabla u\|_1 = \sum_{(i, j) \in \Omega} |\nabla u(i, j)|, \quad \text{where } |\nabla u(i, j)| = \sqrt{(\nabla_x u(i, j))^2 + (\nabla_y u(i, j))^2}. \quad (3)$$

The above regularized minimization model (2) is called the total variation (TV) model for image recovery. This model is powerful for preserving sharp edges, and plenty of studies and variants based on this model have been investigated, please see [2–4, 8, 14, 16, 19] and references therein.

However, image deblurring with TV model often leads to staircase effects. In order to avoid this phenomenon, one popular approach is combining the TV regularization term with some more effective models. In [9, 21], through mixing TV and Tikhonov regularization terms, the authors presented a hybrid model, and their

experiments demonstrated that their model can effectively reduce noise. In [10, 15, 18], some other hybrid regularization models were investigated with combining the TV model and LLT model (originally proposed by Lysaker, Lundervold and Tai in [15]). In [1, 12], the authors employed the TV model with harmonic model for the image recovery. Among all these hybrid models, we focus on the model of combining TV and Tikhonov regularization, which is described by

$$\min_u \frac{\mu}{2} \|A \otimes u - f\|_2^2 + \|\nabla u\|_1 + \frac{1}{2} \|\nabla u\|_2^2, \quad (4)$$

where $\mu \in \mathbb{R}_+$ is a parameter.

Although all the aforementioned hybrid models based on total variation and Tikhonov regularization models are widely used in the field of image recovery, there is one drawback that the objective function is nondifferentiable. There are many ways to conquer the disadvantage of nonsmoothness, one of them is using the smoothing approximation. In this paper, we employ four smoothing functions, which were studied in [17, 25, 26], to approximate the TV part in model (4). Accordingly, we reformulate the image deblurring problem as a smoothing convex optimization problem, and then apply semi-proximal alternating direction method of multipliers (SP-ADMM) to solve the smoothing model. In view of the smoothing feature, we can achieve the analytical solutions of subproblems in SP-ADMM without using soft threshold operator. To the contrast, in the literature of image processing, solving the model with TV part usually requires soft threshold operator. Our proposed algorithm is shown globally convergent. Moreover, we compare the proposed algorithm with different kinds of denoising and deblurring algorithms, including TV [20], DCA [13], TRL2 [24], SOCF [11], and BM3D [6] respectively. Numerical simulations demonstrate that our model effectively eliminates the noise and blur at the same time, and it has higher peak signal to noise ratio (PSNR) and structural similarity index (SSIM) compared to other methods. In summary, our new approach provides a promising way in this field, which is a good contribution.

2 The smoothing hybrid regularization model

Recently, there are six smoothing functions studied in [17] to approximate the absolute value function. In particular, they were employed for signal reconstruction in [25, 26], which show promising performance and effectiveness in real applications. Inspired by this, we adopt those smoothing functions to approximate the TV part in model (4). Since our algorithm needs to exploit and calculate

their analytic solutions, we could only use four of them due to this reason. First, we present them out as below:

$$\begin{aligned}\varphi_1(\mu_1, t) &= \begin{cases} t & \text{if } t \geq \frac{\mu_1}{2}, \\ \frac{t^2}{\mu_1} + \frac{\mu_1}{4} & \text{if } -\frac{\mu_1}{2} < t < \frac{\mu_1}{2}, \\ -t & \text{if } t \leq -\frac{\mu_1}{2}, \end{cases} \\ \varphi_2(\mu_1, t) &= \sqrt{4\mu_1^2 + t^2}, \\ \varphi_3(\mu_1, t) &= \begin{cases} \frac{t^2}{2\mu_1} & \text{if } |t| \leq \mu_1, \\ |t| - \frac{\mu_1}{2} & \text{if } |t| > \mu_1, \end{cases} \\ \varphi_4(\mu_1, t) &= \begin{cases} t & \text{if } t > \mu_1, \\ -\frac{t^4}{8\mu_1^3} + \frac{3t^2}{4\mu_1} + \frac{3\mu_1}{8} & \text{if } -\mu_1 \leq t \leq \mu_1, \\ -t & \text{if } t < -\mu_1, \end{cases}\end{aligned}$$

where $\mu_1 > 0$ is a smoothing parameter. With these smoothing functions, the hybrid model (4) can be replaced by

$$\min_u \frac{\mu}{2} \|A \otimes u - f\|_2^2 + \sum_{(i,j) \in \Omega} \varphi_l(\mu_1, |\nabla u(i,j)|) + \frac{1}{2} \|\nabla u\|_2^2, \quad (5)$$

where $l = 1, 2, 3, 4$. More specifically, each φ_l is a smoothing approximation of $|\nabla u(i,j)|$, i.e.,

$$\varphi_l(\mu_1, |\nabla u(i,j)|) \approx |\nabla u(i,j)|.$$

Figure 1 depicts the graphs of φ_l for $l = 1, 2, 3, 4$. With this, we apply a version of semi-proximal alternating direction method of multipliers (SP-ADMM), which will be presented in the next section, to solve the smoothing model (5).

3 Algorithm and convergence analysis

This section is devoted to the detailed description and implementation of our algorithmic idea and its global convergence. In order to apply semi-proximal alternating direction method of multipliers (SP-ADMM), we further recast the smoothing model (5) as an equivalent form,

$$\begin{aligned}\min_u \quad & \frac{\mu}{2} \|A \otimes u - f\|_2^2 + \sum_{(i,j) \in \Omega} \varphi_l(\mu_1, |k(i,j)|) + \frac{1}{2} \|k\|_2^2 \\ \text{s.t.} \quad & \nabla u - k = 0.\end{aligned} \quad (6)$$

The augmented Lagrangian function of problem (6) is expressed as

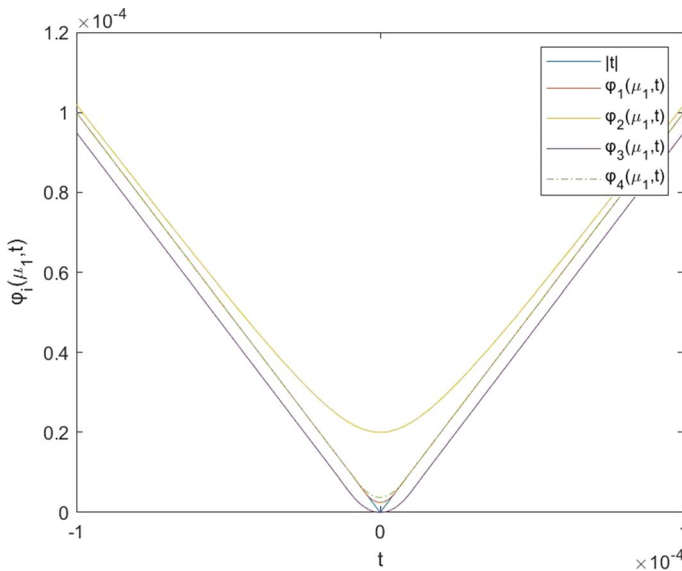


Fig. 1 The figure of approximate effects when $\mu_1 = 1.0e - 5$

$$\begin{aligned}
 H(u, k, \lambda) = & \frac{\mu}{2} \|A \otimes u - f\|_2^2 + \sum_{(i,j) \in \Omega} \varphi_l(\mu_1, |k(i,j)|) + \frac{1}{2} \|k\|_2^2 \\
 & + \langle \lambda, \nabla u - k \rangle + \frac{\beta}{2} \|\nabla u - k\|_2^2,
 \end{aligned} \quad (7)$$

where $\lambda = (\lambda_1, \lambda_2) \in \mathbb{R}^{2m \times n}$ is the Lagrange multiplier parameter matrix with $\lambda(i, j) = (\lambda_1(i, j), \lambda_2(i, j))$, and $\beta > 0$ is a parameter. Accordingly, the SP-ADMM iterative scheme for solving (7) goes as below:

$$\begin{aligned}
 u^n &= \arg \min_u H(u, k^{n-1}, \lambda^{n-1}) + \frac{1}{2} \|u - u^{n-1}\|_{S_1}^2, \\
 k^n &= \arg \min_k H(u^n, k, \lambda^{n-1}) + \frac{1}{2} \|k - k^{n-1}\|_{S_2}^2, \\
 \lambda^n &= \lambda^{n-1} + \eta \beta (\nabla u^n - k^n),
 \end{aligned}$$

where $\eta \in \left(0, \frac{1+\sqrt{5}}{2}\right)$. The matrix norm is defined by $\|x\|_S := \sqrt{\langle x, Sx \rangle}$ for any $u \in \mathbb{R}^{m \times n}$ and $S \in \mathbb{R}^{m \times m}$. In our implementations, we set $S_1 = \rho_1 I$, $S_2 = \rho_2 I$, where $\rho_1 > 0$, $\rho_2 > 0$ and I is the unit matrix.

Algorithm 1

Step 0. Input a blurred noisy image f , a blurring operator A , standard deviation σ_ϵ and Gaussian kernel. Set parameters μ , μ_1 , β , η , S_1 , S_2 , tol, MaxIter.

Step 1. Using the BM3D technique as in [5] to preprocess blurred image f .

Step 2. Initialize $u^0 = 0, k^0 = 0, \lambda^0 = 0$.

Step 3. For $n = 1$: MaxIter, iterate.

$$\begin{cases} u^n = \arg \min_u H(u, k^{n-1}, \lambda^{n-1}) + \frac{1}{2} \|u - u^{n-1}\|_{S_1}^2, & (8) \\ k^n = \arg \min_k H(u^n, k, \lambda^{n-1}) + \frac{1}{2} \|k - k^{n-1}\|_{S_2}^2, & (9) \\ \lambda^n = \lambda^{n-1} + \eta \beta (\nabla u^n - k^n). & (10) \end{cases}$$

Step 4. If $\left\{ \frac{\|u^n - u^{n-1}\|_2}{\|f\|_2}, \frac{\|k^n - \nabla u^n\|_2}{\|\nabla f\|_2} \right\} \leq \text{tol}$, stop.

Step 5. Output u^n .

To proceed, we elaborate more about how to solve the subproblems (8) and (9) in Algorithm 1. From subproblem (8), we know

$$\mu A^T (A \otimes u - f) + \nabla^T \lambda - \beta \nabla^T k^{n-1} + \beta \Delta u + \rho_1 (u - u^{n-1}) = 0,$$

where $\Delta u = \nabla^T \nabla u$. In light of the convolution theorem of Fourier transforms [22], the solution of subproblem (8) can be written as

$$u = \mathcal{F}^{-1} \left\{ \frac{\mu \mathcal{F}(A^T) \odot \mathcal{F}(f) + \mathcal{F}(\nabla^T (\beta k^{n-1} - \lambda^{n-1})) + \mathcal{F}(\rho_1 u^{n-1})}{\mu \mathcal{F}(A^T) \otimes \mathcal{F}(A) + \mathcal{F}(\beta \Delta + \rho_1 I)} \right\}, \quad (11)$$

where \odot denotes componentwise multiplication. As for solving the subproblem (9), we notice that the subproblem (9) is equivalent to

$$\begin{aligned} k^n = \arg \min_k \sum_{(i,j) \in \Omega} \varphi_l(\mu_1, |k(i,j)|) + \frac{1}{2} \|k\|_2^2 + \langle \lambda^{n-1}, \nabla u^n - k \rangle \\ + \frac{\beta}{2} \|\nabla u^n - k\|_2^2 + \frac{1}{2} \|k - k^{n-1}\|_{S_2}^2 \end{aligned} \quad (12)$$

for each $l = 1, 2, 3, 4$. It is easy to verify that (12) is indeed the same as

$$\begin{aligned} k^n(i,j) = \arg \min_{k(i,j)} \varphi_l(\mu_1, |k(i,j)|) + \frac{1}{2} |k(i,j)|^2 + \langle \lambda^{n-1}(i,j), \nabla u^n(i,j) - k(i,j) \rangle \\ + \frac{\beta}{2} |\nabla u^n(i,j) - k(i,j)|^2 + \frac{\rho_2}{2} |k(i,j) - k^{n-1}(i,j)|^2. \end{aligned} \quad (13)$$

Since each φ_l for $l = 1, 2, 3, 4$ is differentiable, the optimality condition of (13) is characterized by

$$\nabla \varphi_l(\mu_1, |k^n(i, j)|) - \lambda^{n-1}(i, j) - \beta \nabla u^n(i, j) - \rho_2 k^{n-1}(i, j) + (1 + \beta + \rho_2) k^n(i, j) = 0. \quad (14)$$

(i) For $l = 1$, when $|k^n(i, j)| \geq \frac{\mu_1}{2}$, we have

$$k^n(i, j) = \frac{|k^n(i, j)|}{1 + (1 + \beta + \rho_2)|k^n(i, j)|} (\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j)).$$

It is nonlinear, by using the information of the previous step in the algorithm, we approximate it as

$$k^n(i, j) = \frac{|k^{n-1}(i, j)|}{1 + (1 + \beta + \rho_2)|k^{n-1}(i, j)|} (\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j)). \quad (15)$$

Likewise, when $|k^n(i, j)| < \frac{\mu_1}{2}$, we obtain

$$k^n(i, j) = \frac{\mu_1}{2 + (1 + \beta + \rho_2)\mu_1} (\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j)). \quad (16)$$

(ii) For $l = 2$, in light of (14), we have

$$\left(\frac{1}{\sqrt{|k^n(i, j)|^2 + 4\mu_1^2}} + 1 + \beta + \rho_2 \right) k^n(i, j) - \lambda^{n-1}(i, j) - \beta \nabla u^n(i, j) - \rho_2 k^{n-1}(i, j) = 0. \quad (17)$$

Like (15), the problem (17) is approximately solved as

$$k^n(i, j) = \frac{P}{1 + (1 + \beta + \rho_2)P} (\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j)). \quad (18)$$

where $P = \sqrt{|k^{n-1}(i, j)|^2 + 4\mu_1^2}$.

(iii) For $l = 3$, when $|k^n(i, j)| > \mu_1$, applying the same approximation technique yields

$$k^n(i, j) = \frac{|k^{n-1}(i, j)|}{1 + (1 + \beta + \rho_2)|k^{n-1}(i, j)|} (\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j)). \quad (19)$$

In addition, when $|k^n(i, j)| \leq \mu_1$, it leads to

$$k^n(i, j) = \frac{\mu_1}{1 + (1 + \beta + \rho_2)\mu_1} (\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j)). \quad (20)$$

(iv) For $l = 4$, when $|k^n(i, j)| > \mu_1$, the approximate solution is

$$k^n(i, j) = \frac{|k^{n-1}(i, j)|}{1 + (1 + \beta + \rho_2)|k^{n-1}(i, j)|} (\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j)), \quad (21)$$

while, when $|k^n(i, j)| \leq \mu_1$, the approximate solution is

$$k^n(i, j) = \frac{2\mu_1^3(\rho_2 k^{n-1}(i, j) + \beta \nabla u^n(i, j) + \lambda^{n-1}(i, j))}{(3 + (1 + \beta + \rho_2)2\mu_1)\mu_1^2 - (k^{n-1}(i, j))^2}. \quad (22)$$

From all the above discussions, we achieve explicit expressions for k^n with explicit $k^n(i, j)$, which is the solution to the subproblem (9).

With the four smoothing functions and the above results, we are ready to present the global convergence of our algorithm. The proof follows the similar idea and analysis techniques used in [7, 11], albeit a bit tedious.

Theorem 3.1 *For any $\mu_1 > 0$, consider the Algorithm 1. Let $\{(u^n, k^n, \lambda^n)\}$ be the sequence generated by Algorithm 1. Then, we have*

$$\lim_{n \rightarrow \infty} \nabla H(u^n, k^n, \lambda^n) = 0.$$

Proof For convenience, we denote

$$F(u) := \frac{\mu}{2} \|A \otimes u - f\|_2^2 \quad \text{and} \quad G(k) := \sum_{(i,j) \in \Omega} \varphi_l(\mu_1, |k(i, j)|) + \frac{1}{2} \|k\|_2^2 \quad \text{for } l = 1, 2, 3, 4.$$

We see that (\bar{u}, \bar{k}) is an optimal solution of problem (6) if and only if there exists Lagrange multiplier $\bar{\lambda}$ such that

$$\begin{cases} 0 = \nabla F(\bar{u}) + \nabla^\top \bar{\lambda}, \\ 0 = \nabla G(\bar{k}) - \bar{\lambda}, \\ 0 = \nabla \bar{u} - \bar{k}, \end{cases}$$

where $\nabla F = \mu A^T(A \otimes u - f)$. From the KKT conditions, we know that if (u, k, λ) is the KKT point of problem (6), then

$$\begin{cases} 0 = \nabla F(u) + \nabla^\top \lambda, \\ 0 = \nabla G(k) - \lambda, \\ 0 = \nabla u - k. \end{cases} \quad (23)$$

From the definition of φ_l for $l = 1, 2, 3, 4$, we know that the function φ_l is convex. Since $\|k\|_2^2$ is strongly convex, the function G is strongly convex. It is also clear to see that the function F is strongly convex. Thus, the gradients ∇F and ∇G are

strongly monotone. These indicate that there exist positive constants σ_1 and σ_2 such that for all $u, \hat{u} \in \mathbb{R}^{m \times n}$, $\omega = \nabla F(u)$ and $\hat{\omega} = \nabla F(\hat{u})$, there holds

$$\langle \omega - \hat{\omega}, u - \hat{u} \rangle \geq \sigma_1 \|u - \hat{u}\|_2^2 = \|u - \hat{u}\|_{\Sigma_F}^2, \quad \Sigma_F = \sigma_1 I; \quad (24)$$

as well as for all $k, \hat{k} \in \mathbb{R}^{2m \times n}$, $x = \nabla G(k)$ and $\hat{x} = \nabla G(\hat{k})$, there holds

$$\langle x - \hat{x}, k - \hat{k} \rangle \geq \sigma_2 \|k - \hat{k}\|_2^2 = \|k - \hat{k}\|_{\Sigma_G}^2, \quad \Sigma_G = \sigma_2 I. \quad (25)$$

On the other hand, from the Algorithm 1, we have

$$\begin{cases} 0 = \nabla F(u^{n+1}) + \nabla^\top (\beta(\nabla u^{n+1} - k^n) + \lambda^n) + S_1(u^{n+1} - u^n), \\ 0 = \nabla G(k^{n+1}) - (\beta(\nabla u^{n+1} - k^{n+1}) + \lambda^n) + S_2(k^{n+1} - k^n), \\ 0 = (\nabla u^{n+1} - k^{n+1}) - (\beta\eta)^{-1}(\lambda^{n+1} - \lambda^n). \end{cases} \quad (26)$$

For notational simplicity, let $\epsilon(u, k) := \nabla u - k$ and denote $u_e^n := u^n - \bar{u}$. Similar notational ideas are applied to k_e^n, λ_e^n as well. Then, we have

$$\begin{cases} \omega^{n+1} = \lambda^{n+1} + (1 - \eta)\beta\epsilon(u^{n+1}, k^{n+1}) + \beta(k^{n+1} - k^n), \\ x^{n+1} = \lambda^{n+1} + (1 - \eta)\beta\epsilon(u^{n+1}, k^{n+1}), \end{cases}$$

which imply

$$\begin{aligned} \epsilon(u_e^{n+1}, k_e^{n+1}) &= \nabla u^{n+1} - k^{n+1} = \epsilon(u^{n+1}, k^{n+1}) \\ &= \beta\eta^{-1}(\lambda_e^{n+1} - \lambda^n) \\ &= (\beta\eta)^{-1}(\lambda^{n+1} - \lambda^n). \end{aligned}$$

Using (23)–(26) all together yields

$$\begin{aligned} \|u_e^{n+1}\|_{\Sigma_F}^2 &= \|u^{n+1} - \bar{u}\|_{\Sigma_F}^2 \leq \langle \omega^{n+1} - \bar{\omega}, u^{n+1} - \bar{u} \rangle \\ &= \langle \nabla^\top (\bar{\lambda} - \lambda^n) + \beta(\nabla^\top k^n - \Delta u^{n+1}) - S_1(u^{n+1} - u^n), u_e^{n+1} \rangle, \end{aligned} \quad (27)$$

$$\begin{aligned} \|k_e^{n+1}\|_{\Sigma_G}^2 &= \|k^{n+1} - \bar{k}\|_{\Sigma_G}^2 \leq \langle x^{n+1} - \bar{x}, k^{n+1} - \bar{k} \rangle \\ &= \langle (\lambda^n - \bar{\lambda}) + \beta(\nabla u^n - k^{n+1}) - S_2(k^{n+1} - k^n), k_e^{n+1} \rangle. \end{aligned} \quad (28)$$

By calculation, it follows that

$$\begin{aligned} -\nabla^\top w^{n+1} &= -\nabla^\top [\lambda^{n+1} + (1 - \eta)\beta\epsilon(u^{n+1}, k^{n+1}) + \beta(k^{n+1} - k^n)] \\ &= -\nabla^\top \lambda^n - \beta\nabla^\top \nabla u^{n+1} + \beta\nabla^\top k^n, \\ x^{n+1} &= \lambda^{n+1} + (1 - \eta)\beta\epsilon(u^{n+1}, k^{n+1}) \\ &= \lambda^n + \beta(\nabla u^{n+1} - k^{n+1}). \end{aligned}$$

Hence, inequality (27) is converted to (29) and inequality (28) becomes (30), that is,

$$\|u_e^{n+1}\|_{\Sigma_F}^2 \leq \left\langle -\nabla^\top w^{n+1} - S_1(u^{n+1} - u^n) + \nabla^\top \bar{\lambda}, u_e^{n+1} \right\rangle, \quad (29)$$

$$\|k_e^{n+1}\|_{\Sigma_G}^2 \leq \left\langle x^{n+1} - S_2(k^{n+1} - k^n) - \bar{\lambda}, k_e^{n+1} \right\rangle. \quad (30)$$

Then, according to (29) and (30), we achieve

$$\begin{aligned} \|u_e^{n+1}\|_{\Sigma_F}^2 + \|k_e^{n+1}\|_{\Sigma_G}^2 &\leq (\beta\eta)^{-1} \langle \lambda_e^{n+1}, \lambda_e^n - \lambda_e^{n+1} \rangle - \beta \langle k^{n+1} - k^n, \epsilon(u^{n+1}, k^{n+1}) \rangle \\ &\quad - \beta \langle k^{n+1} - k^n, k_e^{n+1} \rangle - \beta(1 - \eta) \|\epsilon(u^{n+1}, k^{n+1})\|_2^2 \\ &\quad - \langle S_1(u^{n+1} - u^n), u_e^{n+1} \rangle - \langle S_2(k^{n+1} - k^n), k_e^{n+1} \rangle. \end{aligned} \quad (31)$$

Next, we estimate the term $\beta \langle k^{n+1} - k^n, \epsilon(u^{n+1}, k^{n+1}) \rangle$. First, it follows from Eq. (26) that

$$x^{n+1} - S_2(k^{n+1} - k^n) = \nabla G(k^{n+1}), \quad x^n - S_2(k^n - k^{n-1}) = \nabla G(k^n).$$

In addition, by the strong monotonicity of $\nabla G(\cdot)$, we have

$$\begin{aligned} &\langle k^{n+1} - k^n, x^{n+1} - x^n \rangle \\ &= \langle k^{n+1} - k^n, x^{n+1} - S_2(k^{n+1} - k^n) - (x^n - S_2(k^n - k^{n-1})) \rangle \\ &\quad + \langle k^{n+1} - k^n, S_2(k^{n+1} - k^n) \rangle - \langle k^{n+1} - k^n, S_2(k^n - k^{n-1}) \rangle \\ &\geq \|k^{n+1} - k^n\|_{S_2}^2 + \|k^{n+1} - k^n\|_{S_2}^2 - \langle k^{n+1} - k^n, S_2(k^n - k^{n-1}) \rangle \\ &\geq \|k^{n+1} - k^n\|_{S_2}^2 - \langle k^{n+1} - k^n, S_2(k^n - k^{n-1}) \rangle. \end{aligned}$$

Moreover, by denoting $\alpha_{n+1} = -(1 - \eta)\beta \langle k^{n+1} - k^n, \epsilon(u^n, k^n) \rangle$, it further implies

$$\begin{aligned} &-\beta \langle k^{n+1} - k^n, \epsilon(u^{n+1}, k^{n+1}) \rangle \\ &= -(1 - \eta)\beta \langle k^{n+1} - k^n, \epsilon(u^{n+1}, k^{n+1}) \rangle \\ &\quad - \langle k^{n+1} - k^n, x^{n+1} - x^n - (1 - \eta)\beta \epsilon(u^{n+1}, k^{n+1}) + (1 - \eta)\beta \epsilon(u^n, k^n) \rangle \\ &= \alpha_{n+1} + \langle k^{n+1} - k^n, x^n - x^{n+1} \rangle \\ &\leq \alpha_{n+1} - \|k^{n+1} - k^n\|_{S_2}^2 + \langle k^{n+1} - k^n, S_2(k^n - k^{n-1}) \rangle \\ &\leq \alpha_{n+1} - \|k^{n+1} - k^n\|_{S_2}^2 + \frac{1}{2} \|k^{n+1} - k^n\|_{S_2}^2 + \frac{1}{2} \|k^n - k^{n-1}\|_{S_2}^2 \\ &= \alpha_{n+1} - \frac{1}{2} \|k^{n+1} - k^n\|_{S_2}^2 + \frac{1}{2} \|k^n - k^{n-1}\|_{S_2}^2. \end{aligned} \quad (32)$$

Since $\lambda^{n+1} = \lambda^n + (\beta\eta)\epsilon(u^{n+1}, k^{n+1})$, applying (31) and (32) indicate

$$\begin{aligned}
& 2\|u_e^{n+1}\|_{\Sigma_F}^2 + 2\|k_e^{n+1}\|_{\Sigma_G}^2 \\
& \leq 2(\beta\eta)^{-1}\langle\lambda_e^{n+1}, \lambda_e^n - \lambda_e^{n+1}\rangle - 2\beta\langle k^{n+1} - k^n, \epsilon(u^{n+1}, k^{n+1})\rangle \\
& \quad - 2\beta\langle k^{n+1} - k^n, k_e^{n+1}\rangle - 2\beta(1-\eta)\|\epsilon(u^{n+1}, k^{n+1})\|_2^2 \\
& \quad - 2\langle S_1(u^{n+1} - u^n), u_e^{n+1}\rangle - 2\langle S_2(k^{n+1} - k^n), k_e^{n+1}\rangle \\
& \leq (\beta\eta)^{-1}(\|\lambda_e^n\|^2 - \|\lambda_e^{n+1}\|^2) - (2-\eta)\beta\|\epsilon(u^{n+1}, k^{n+1})\|^2 \\
& \quad + 2\alpha_{n+1} - \|k^{n+1} - k^n\|_{S_2}^2 + \|k^n - k^{n-1}\|_{S_2}^2 \\
& \quad - \beta\|k^{n+1} - k^n\|^2 - \beta\|k_e^{n+1}\|^2 + \beta\|k_e^n\|^2 \\
& \quad - \|u^{n+1} - u^n\|_{S_1}^2 - \|u_e^{n+1}\|_{S_1}^2 + \|u_e^n\|_{S_1}^2 \\
& \quad - \|k^{n+1} - k^n\|_{S_2}^2 - \|k_e^{n+1}\|_{S_2}^2 + \|k_e^n\|_{S_2}^2.
\end{aligned} \tag{33}$$

To proceed, we further define

$$\begin{cases} \delta_{n+1} = \min\{\eta, 1 + \eta - \eta^2\}\beta\|k^{n+1} - k^n\|^2 + \|k^{n+1} - k^n\|_{S_2}^2, \\ t_{n+1} = \delta_{n+1} + \|u^{n+1} - u^n\|_{S_1}^2 + 2\|u^{n+1} - \bar{u}\|_{\Sigma_F}^2 + 2\|k^{n+1} - \bar{k}\|_{\Sigma_G}^2, \\ \psi_{n+1} = \theta(u^{n+1}, k^{n+1}, \lambda^{n+1}) + \|k^{n+1} - k^n\|_{S_2}^2, \\ \theta(u, k, \lambda) = (\beta\eta)^{-1}\|\lambda - \bar{\lambda}\|^2 + \|u - \bar{u}\|_{S_1}^2 + \|k - \bar{k}\|_{S_2}^2 + \beta\|k - \bar{k}\|^2. \end{cases} \tag{34}$$

and discuss two cases as below.

Case (1): $\eta \in (0, 1]$. It is obvious that

$$2\langle k^{n+1} - k^n, \epsilon(u^n, k^n) \rangle \leq \|k^{n+1} - k^n\|^2 + \|\epsilon(u^n, k^n)\|^2.$$

From the definition of α_{n+1} and (33), we see

$$\begin{aligned}
& \psi_{n+1} + (1-\eta)\beta\|\epsilon(u^{n+1}, k^{n+1})\|^2 - [\psi_n + (1-\eta)\beta\|\epsilon(u^n, k^n)\|^2] \\
& \quad + t_{n+1} + \beta\|\epsilon(u^{n+1}, k^{n+1})\|^2 \leq 0.
\end{aligned} \tag{35}$$

Case (2): $\eta \in (1, \frac{1+\sqrt{5}}{2})$. Similarly, we obtain

$$\begin{aligned}
& \psi_{n+1} + (1-\eta^{-1})\beta\|\epsilon(u^{n+1}, k^{n+1})\|^2 - [\psi_n + (1-\eta^{-1})\beta\|\epsilon(u^n, k^n)\|^2] \\
& \quad + t_{n+1} + \eta^{-1}(1+\eta-\eta^2)\beta\|\epsilon(u^{n+1}, k^{n+1})\|^2 \leq 0.
\end{aligned} \tag{36}$$

In other words, by introducing a new notation g_{n+1} , it says

$$g_{n+1} = \begin{cases} \psi_{n+1} + (1-\eta)\beta\|\epsilon(u^{n+1}, k^{n+1})\|^2, & \eta \in (0, 1], \\ \psi_{n+1} + (1-\eta^{-1})\beta\|\epsilon(u^{n+1}, k^{n+1})\|^2, & \eta \in (1, \frac{1+\sqrt{5}}{2}). \end{cases}$$

Thus, from (35) and (36), we conclude that the sequence $\{g_n\}$ is bounded and monotonically decreasing, which guarantees a limit. Since $\psi_n > 0$, the sequence

$\{\psi_n\}$ is bounded. Let $\gamma = 1$ or $\gamma = \eta^{-1}(1 + \eta - \eta^2)$. Again, applying (35) and (36) implies

$$0 \leq t_{n+1} + \gamma\beta \|\epsilon(u^{n+1}, k^{n+1})\|^2 \leq g_n - g_{n+1}.$$

This says that there must have

$$t_{n+1} \rightarrow 0, \quad n \rightarrow \infty, \quad (37)$$

$$\|\epsilon(u^{n+1}, k^{n+1})\| \rightarrow 0, \quad n \rightarrow \infty. \quad (38)$$

Considering the relationship $\lambda^{n+1} - \lambda^n = (\eta\beta)\epsilon(u^{n+1}, k^{n+1})$, we have $\|\lambda^{n+1} - \lambda^n\| \rightarrow 0$. In view of (37) and the boundedness of $\{\psi_n\}$, one can conclude these sequences $\{\|\lambda^{n+1}\|\}$, $\{\|u_e^{n+1}\|_{S_1}^2\}$, $\{\|u_e^{n+1}\|_{\Sigma_F}^2\}$, $\{\|k_e^{n+1}\|_{\Sigma_G}^2\}$, $\{\|k_e^{n+1}\|_{S_2}^2\}$, $\{\|k_e^{n+1}\|^2\}$ are all bounded. Then, by using the inequality

$$\|\nabla u_e^{n+1}\| \leq \|\nabla u_e^{n+1} - k_e^{n+1}\| + \|k_e^{n+1}\| = \|\epsilon(u^{n+1}, k^{n+1})\| + \|k_e^{n+1}\|,$$

we deduce that $\{\|\nabla u_e^{n+1}\|\}$ is also bounded, hence $\{\|u_e^{n+1}\|_{\nabla^\top \nabla}\}$ is bounded. Since

$$\|u_e^{n+1}\| = \|u_e^{n+1}\|_{S_1 + \Sigma_F + \nabla^\top \nabla + I} - \|u_e^{n+1}\|_{S_1 + \Sigma_F + \nabla^\top \nabla},$$

and the positive definiteness of $S_1 + \Sigma_F + \nabla^\top \nabla + I$, the sequence $\{\|u_e^{n+1}\|\}$ is guaranteed bounded, and hence the sequence $\{\|u^{n+1}\|\}$ is bounded. In summary, the sequence $\{(u^n, k^n, \lambda^n)\}$ is bounded, which gives the existence of a convergent subsequence to a cluster point, denoted by $\lim_{i \rightarrow \infty} (u^{n^i}, k^{n^i}, \lambda^{n^i}) = (u^*, k^*, \lambda^*)$.

Consequently, it follows from (34) and (37) that

$$\begin{aligned} \lim_{n \rightarrow \infty} \|k^{n+1} - k^n\| &= 0, \\ \lim_{n \rightarrow \infty} \|k^{n+1} - k^n\|_{S_2} &= 0, \\ \lim_{n \rightarrow \infty} \|u^{n+1} - u^n\|_{S_1} &= 0. \end{aligned} \quad (39)$$

Therefore, from $\|\nabla u^{n+1} - k^n\| \leq \|\nabla u^{n+1} - k^{n+1}\| + \|k^{n+1} - k^n\|$, we have $\lim_{n \rightarrow \infty} \|\nabla u^{n+1} - k^n\| = 0$. Taking limits on both sides of Eq. (26) along the subsequence $\{(u^{n^i}, k^{n^i}, \lambda^{n^i})\}$ and using the closedness of subdifferential, there hold

$$\begin{cases} -\nabla^\top \lambda^* = \nabla F(u^*), \\ \lambda^* = \nabla G(k^*), \\ 0 = \nabla u^* - k^*. \end{cases}$$

In other words, (u^*, k^*) is the optimal solution of (6) and λ^* is the corresponding Lagrange multiplier.

Now, it remains to verify that $\lim_{i \rightarrow \infty} (u^{n^i}, k^{n^i}, \lambda^{n^i}) = (u^*, k^*, \lambda^*)$. Since (u^*, k^*, λ^*) satisfies (26), we could replace $(\bar{u}, \bar{k}, \bar{\lambda})$ with (u^*, k^*, λ^*) in the above analysis. From (35)–(38), we find $g_{n^i} \rightarrow 0$ as $i \rightarrow \infty$. In particular, (35) and (36) indicate the

sequence $\{g_n\}$ has limits, then $g_n \rightarrow 0$ as $n \rightarrow \infty$. Accordingly, $\psi_n \rightarrow 0$ as $n \rightarrow \infty$. Furthermore, we know from the definition of ψ_n that when $n \rightarrow \infty$,

$$\begin{aligned}\|\lambda^n - \lambda^*\| &\rightarrow 0 \implies \lambda^n \rightarrow \lambda^* \\ \|k^n - k^*\| &\rightarrow 0 \implies k^n \rightarrow k^* \\ \|u_e^{n+1}\|_{S_1}^2 &\rightarrow 0\end{aligned}\quad (40)$$

Taking into account of (37) and (38), we conclude that

$$\|u_e^{n+1}\|_{\Sigma_F}^2 \rightarrow 0, \quad n \rightarrow \infty. \quad (41)$$

From (41), it also guarantees that $\lim_{n \rightarrow \infty} \|u_e^n\| = 0$, namely, $\lim_{n \rightarrow \infty} u^n = u^*$. To sum up, when $\eta \in (0, \frac{1+\sqrt{5}}{2})$, we show that

$$\lim_{n \rightarrow \infty} (u^n, k^n, \lambda^n) = (u^*, k^*, \lambda^*).$$

Then, the proof is complete. \square

4 Numerical experiments

In this section, we report the results of our simulations demonstrating the applicability, efficiency and merits of our algorithm. All the experiments are performed under windows 10 and MATLAB R2018a running on a desktop (Inter(CR) Core(TM) i5-8250 CPU @ 1.60 GHZ).

Our tests consist of 16 images, including 11 gray images and 5 color images, as shown in Fig. 2. When running Algorithm 1, we use three types of blurring functions, Gaussian blur (GB), Motion blur (MB) and Average blur (AB), with different levels of Gaussian noise. The notation $\text{GB}(9, 5)/\sigma_\varepsilon = 5$ denotes the Gaussian kernel with the free parameter $\sigma_B = 5$, size 9×9 , and the Gaussian noise with standard deviation $\sigma_\varepsilon = 5$. Other symbols of blur and noise have similar meanings.

We compare our algorithm with other famous algorithms in the literature. They are the classic TV [20], DCA with $L_1 - 0.5L_2$ [13], TRL2 [24], SOCF [11], and BM3D [6]. We adopt the peak signal to noise ratio (PSNR) and the structural similarity index (SSIM) to evaluate the quality of image restoration. The PSNR is defined by

$$\text{PSNR}(\hat{u}) = 10 \log_{10} \frac{255^2}{\text{MSE}(\hat{u})}, \quad \text{MSE}(\hat{u}) = \frac{1}{m \times n} \sum_{(i,j) \in \Omega} (u(i,j) - \hat{u}(i,j))^2,$$

where u is the original image and \hat{u} is the restored image, and SSIM is defined in [23].

The parameters of our algorithm are summarized as follows:

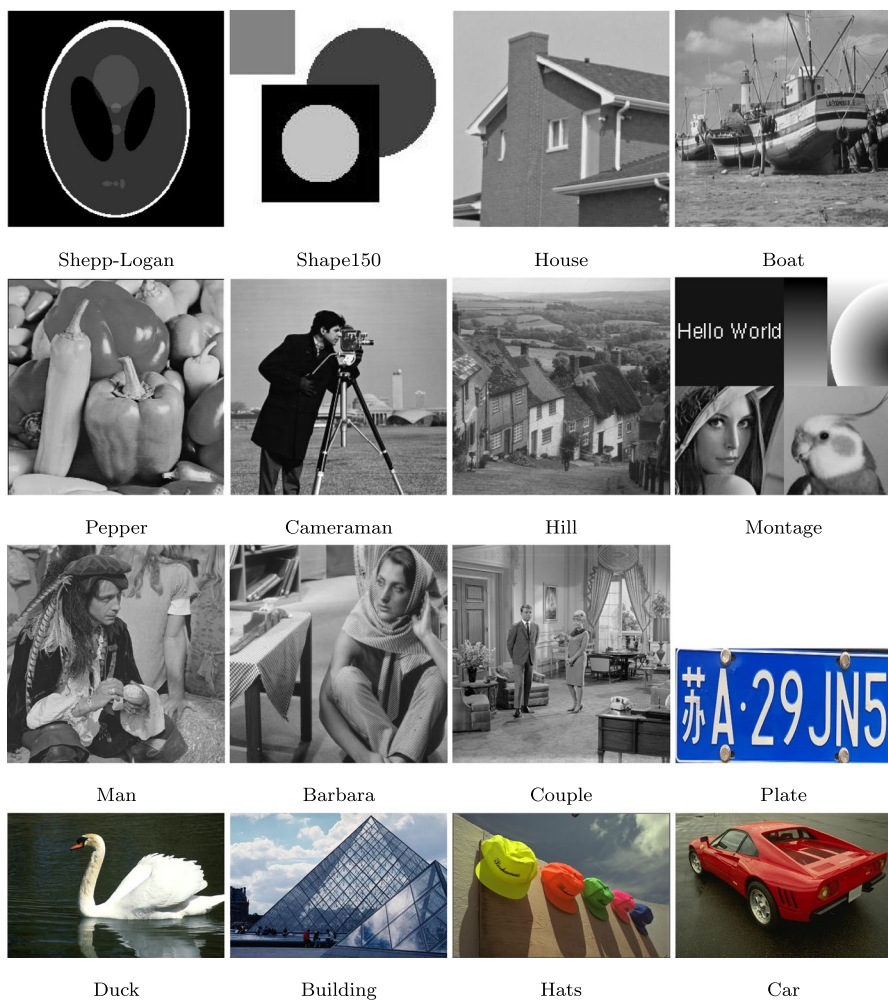


Fig. 2 Set of 16 test images

$$\mu_1 = 1e - 10, \quad \mu \in (2000, 7000), \quad \beta \in (0.2, 0.7), \quad \eta = 1.1.$$

In order to express clearly the influence of parameter μ on our algorithm, we plot the PSNR values and SSIM values versus the values of μ in Fig. 3.

In particular, we stop Algorithm 1 when

$$\sqrt{\frac{1}{m \times n} \sum_{(i,j) \in \Omega} (u(i,j) - \hat{u}(i,j))^2} < 5e - 5$$

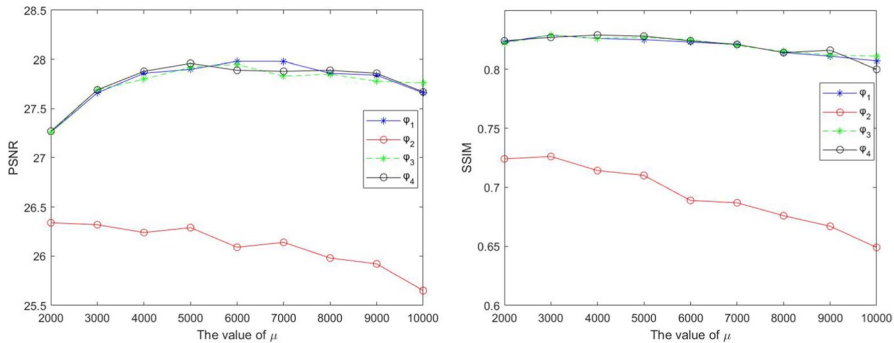


Fig. 3 **a, b** Are graphs of the PSNR and SSIM variation curve with different parameter values of μ for gray image "Pepper" with $AB(9, 9)/\sigma_\epsilon = 3$

or the iteration number exceeds 500. The parameter settings for the other compared algorithms are given below.

- For TV [20], we set

$$\mu^{\text{TV}} \in \{120, 150, 160, 170, 180, 190, 200\} \quad \text{and} \quad \lambda^{\text{TV}} \in \{10, 11, 12, 13, 14, 15, 16\}.$$

- For DCA [13], we set

$$\lambda^{\text{DCA}} = 1 \quad \text{and} \quad \mu^{\text{DCA}} \in \{160, 170, 180, 190, 200, 220\}.$$

- For TRL2 [24], we set

$$\begin{aligned} \tau^{\text{TRL2}} &\in \{0.04, 0.05, 0.06\}, \quad \alpha^{\text{TRL2}} \in \{60, 80, 100, 120, 150, 180\} \\ \text{and} \quad \beta^{\text{TRL2}} &\in \{80, 100, 120, 150\}. \end{aligned}$$

- For SOCF [11], we set

$$\begin{aligned} \lambda^{\text{SOCF}} &\in \{5, 7, 10, 11, 13, 15, 16, 18\}, \quad \mu^{\text{SOCF}} \in \{1e-8, 1e-10\}, \\ \mu_1^{\text{SOCF}} &= 0.2, \quad \mu_2^{\text{SOCF}} = 1e-5 \quad \text{and} \quad \eta^{\text{SOCF}} = 1.5. \end{aligned}$$

- For BM3D [6], we set

$$\text{RI} \in \{4e-4, 4e-3, 1e-3\} \quad \text{and} \quad \text{RWI} \in \{5e-3, 3e-2\}.$$

Experiment 1 We compare the performance of our Algorithm 1 with other famous algorithms under different blur kernels and different Gaussian noise levels. The summary of numerical results are presented in Tables 1, 2 and 3, respectively. From these, we see that Algorithm 1 has higher PSNR and SSIM when φ_l for $l = 1, 2, 3, 4$ is used. The performance of the proposed algorithm with the function φ_2 is relatively poor.

Table 1 The value of PSNR/SSIM for different algorithms with $GB(9, 5)/\sigma_\epsilon = 5$

Image	Degraded	TV [20]	DCA [13]	TRL2 [24]	SOCF [11]	BM3D [6]	φ_1	φ_2	φ_3	φ_4
Shepp-Logan	18.96/0.581	22.75/0.927	23.34/0.895	23.85/0.662	24.52/0.792	24.56/0.846	26.39/0.927	23.57/0.672	26.50/0.937	26.10/ 0.940
Shape150	18.46/0.543	25.08/0.854	25.45/0.866	24.29/0.753	26.19/0.829	26.44/0.873	27.97/0.899	23.76/0.611	28.28/ 0.912	28.16/0.910
House	24.11/0.558	27.93/0.784	28.09/0.797	28.24/0.726	29.09/0.759	29.90/0.783	30.20/0.795	28.75/0.712	30.32/ 0.816	30.28/0.815
Boat	23.34/0.488	25.06/0.632	25.37/0.652	25.97/0.658	26.66/0.682	26.76/0.700	26.92/0.702	26.28/0.649	26.82/0.699	26.88/ 0.706
Pepper	21.43/0.575	23.44/0.735	23.49/0.730	24.69/0.737	26.25/0.759	26.42/0.763	25.98/0.739	24.90/0.664	25.85/0.765	25.90/ 0.776
Cameraman	20.87/0.500	23.08/0.716	23.44/0.731	24.16/0.682	24.35/0.710	24.74/0.737	24.83/0.752	24.05/0.660	24.88/0.758	24.87/ 0.760
Hill	24.85/0.497	26.87/0.631	27.10/0.650	27.34/0.657	27.67/0.670	28.09/0.688	28.01/0.691	27.27/0.634	27.88/0.686	28.00/ 0.692
Montage	20.19/0.584	22.36/0.812	23.14/0.817	24.35/0.760	24.91/0.796	25.24/0.828	25.48/0.807	24.02/0.695	25.30/0.841	25.52/ 0.849
Man	24.37/0.524	25.79/0.645	26.32/0.680	26.78/0.681	27.02/0.688	27.37/ 0.718	27.37/0.713	26.68/0.651	27.24/0.710	27.42/ 0.718
Barbara	22.36/0.499	23.08/0.608	23.11/ 0.618	23.31/0.603	23.65/0.626	23.98/ 0.665	23.76/0.635	23.67/0.602	23.79/0.649	23.73/0.651
Couple	23.22/0.461	24.90/0.606	25.39/0.643	25.84/0.658	26.56/0.690	26.78/ 0.711	26.62/0.692	26.25/0.658	26.61/0.703	26.77/0.709
Plate	17.39/0.778	21.65/0.911	22.44/0.923	23.14/0.929	24.35/0.941	24.57/0.942	24.95/0.945	23.04/0.922	25.14/ 0.948	25.10/ 0.948
Duck	24.17/0.659	27.29/0.795	27.41/0.810	27.91/0.782	28.40/0.797	28.60/0.830	29.17/0.828	27.77/0.767	29.17/0.830	29.24/ 0.834
Building	19.54/0.637	20.34/0.701	20.57/0.728	20.89/0.734	21.17/0.744	21.33/0.638	21.35/0.753	21.13/0.724	21.33/0.758	21.30/ 0.761
Hats	27.04/0.859	29.29/0.940	29.29/0.939	28.90/0.900	29.66/0.918	30.10/0.944	30.49/0.944	29.64/0.920	30.43/0.945	30.41/ 0.946
Redcar	23.92/0.756	25.65/0.835	25.80/0.839	26.09/0.813	26.31/0.824	26.74/0.847	26.93/0.848	26.27/0.820	26.83/0.849	26.92/ 0.851
Average	22.13/0.593	24.66/0.764	24.99/0.770	25.35/0.734	26.05/0.764	26.35/0.782	26.65/0.792	25.44/0.710	26.65/0.800	26.66/ 0.804

Table 2 The value of PSNR/SSIM for different algorithms with MB(20, 60)/ $\sigma_e = 5$

Image	Degraded	TV [20]	DCA [13]	TRL2 [24]	SOCF [11]	BM3D [6]	φ_1	φ_2	φ_3	φ_4
Shepp-Logan	17.47/0.548	21.35/0.911	24.35/0.910	24.95/0.499	25.59/0.825	25.56/0.806	27.46/0.895	23.89/0.533	28.44/0.913	28.26/0.896
Shape150	16.32/0.905	26.74/0.910	30.95/0.948	23.85/0.624	25.50/0.825	26.62/0.865	27.22/0.859	22.94/0.621	27.80/0.874	27.79/0.869
House	21.46/0.505	25.34/0.736	27.25/0.783	25.80/0.602	28.63/0.752	29.24/0.786	29.51/0.790	27.22/0.684	29.78/0.807	29.66/0.800
Boat	22.00/0.457	24.03/0.598	24.93/0.636	24.83/0.601	26.70/0.689	26.63/0.698	26.72/0.694	26.06/0.652	26.82/0.700	26.86/0.701
Pepper	20.05/0.532	22.57/0.708	23.19/0.723	23.36/0.652	24.92/0.756	25.39/0.732	24.66/0.736	23.92/0.666	24.81/0.754	24.83/0.756
Cameraman	19.79/0.478	22.41/0.689	23.37/0.722	23.95/0.607	24.96/0.722	25.25/0.745	25.37/0.743	24.26/0.648	25.44/0.757	25.40/0.754
Hill	23.21/0.450	25.49/0.578	26.29/0.617	25.57/0.584	27.24/0.658	27.42/0.670	27.39/0.668	26.72/0.629	27.51/0.674	27.54/0.675
Montage	19.31/0.548	22.10/0.788	23.23/0.814	24.61/0.656	25.70/0.802	26.02/0.818	26.45/0.803	24.19/0.651	26.80/0.825	26.95/0.831
Man	23.05/0.501	25.41/0.649	26.00/0.678	25.28/0.616	27.30/0.715	27.32/0.728	27.22/0.717	26.25/0.666	27.42/0.727	27.50/0.728
Barbara	21.76/0.495	23.06/0.616	23.48/0.643	23.04/0.586	24.78/0.691	25.53/0.732	24.88/0.697	24.79/0.663	24.85/0.701	25.01/0.707
Couple	21.83/0.421	23.57/0.553	24.51/0.604	24.34/0.591	26.17/0.673	26.16/0.690	26.34/0.688	25.73/0.649	26.39/0.690	26.51/0.693
Plate	15.93/0.731	20.01/0.880	21.53/0.911	22.33/0.920	23.43/0.934	24.11/0.941	24.48/0.943	21.78/0.909	24.57/0.944	24.57/0.945
Duck	22.48/0.622	26.54/0.773	26.81/0.783	26.53/0.681	27.77/0.756	28.03/0.808	28.38/0.789	27.15/0.738	28.39/0.792	28.51/0.796
Building	18.94/0.628	20.38/0.715	20.97/0.749	21.57/0.750	22.41/0.788	22.86/0.816	22.86/0.802	22.46/0.772	22.80/0.805	22.81/0.806
Hats	25.44/0.839	28.29/0.931	28.78/0.935	27.43/0.836	28.81/0.898	29.22/0.937	29.63/0.931	28.80/0.906	29.71/0.935	29.70/0.937
Redcar	22.04/0.737	24.36/0.821	25.14/0.833	25.04/0.760	26.10/0.816	26.49/0.847	26.77/0.840	25.91/0.808	26.86/0.846	26.89/0.845
Average	20.70/0.587	23.86/0.741	25.05/0.768	24.53/0.661	26.00/0.769	26.37/0.789	26.58/0.787	25.15/0.700	26.78/0.796	26.77/0.796

Table 3 The value of PSNR/SSIM for different algorithms with $AB(9, 9)/\sigma_\epsilon = 3$

Image	Degraded	TV [20]	DCA [13]	TRL2 [24]	SOCF [11]	BM3D [6]	φ_1	φ_2	φ_3	φ_4
Shepp-Logan	18.66/0.706	21.74/0.916	22.40/0.914	24.88/0.733	24.90/0.937	25.71/0.884	27.39/0.945	24.06/0.671	27.96/0.951	27.79/0.956
Shape150	18.16/0.613	27.92/0.940	30.09/ 0.960	26.05/0.812	27.74/0.942	28.01/0.928	29.41/0.933	24.79/0.662	29.27/0.926	29.33/0.933
House	23.97/0.622	26.78/0.763	25.14/0.640	29.72/0.774	30.73/0.830	31.47/0.836	31.62/0.840	29.93/0.763	31.67/0.840	31.57/0.839
Boat	23.24/0.521	24.50/0.608	23.23/0.718	27.12/0.710	27.48/0.731	28.06/0.751	28.15/0.755	27.50/0.711	28.15/0.753	28.18/0.753
Pepper	21.25/0.613	22.94/0.709	23.08/0.728	26.52/0.786	27.35/0.829	27.75/0.806	28.11/0.831	26.36/0.732	27.93/0.819	26.43/0.737
Cameraman	20.71/0.561	22.58/0.698	23.13/0.727	25.04/0.733	24.88/0.771	25.81/0.781	25.96/0.796	24.95/0.713	26.05/0.797	26.08/0.796
Hill	24.85/0.527	26.45/0.610	26.90/0.640	28.34/0.710	28.62/0.717	29.03/0.731	29.12/0.741	28.44/0.700	28.91/0.733	28.91/0.733
Montage	20.01/0.648	22.05/0.801	23.15/0.818	26.16/0.819	25.87/0.882	27.18/0.869	27.19/0.886	25.42/0.752	27.41/0.879	27.42/0.877
Man	24.35/0.564	25.90/0.649	26.26/0.671	27.74/0.728	27.89/0.744	28.23/0.755	28.43/0.762	27.71/0.710	28.29/0.754	28.21/0.749
Barbara	22.39/0.542	23.12/0.610	23.11/0.616	23.75/0.642	24.00/0.676	24.44/0.695	24.20/0.686	24.16/0.651	24.17/0.681	24.17/0.681
Couple	23.12/0.484	24.31/0.574	25.14/0.627	26.92/0.716	27.34/0.735	28.15/0.770	28.09/0.768	27.45/0.728	28.03/0.767	28.05/0.766
Plate	17.08/0.766	20.93/0.897	22.24/0.920	25.23/0.951	25.70/0.956	26.19/0.958	26.86/0.963	24.39/0.941	26.95/0.964	26.91/0.963
Duck	24.12/0.708	27.12/0.791	27.42/0.809	29.25/0.831	29.80/0.860	29.91/0.863	30.65/0.868	29.02/0.816	30.33/0.859	30.51/0.863
Building	19.48/0.651	20.17/0.702	20.40/0.720	21.26/0.761	21.31/0.771	22.06/0.794	21.76/0.788	21.65/0.761	21.92/0.791	21.74/0.778
Hats	27.38/0.896	29.20/0.939	29.32/0.941	29.70/0.918	30.99/0.954	31.24/0.954	31.48/0.956	30.62/0.937	31.55/0.956	31.46/0.956
Redcar	23.93/0.785	25.46/0.833	25.60/0.838	26.72/0.837	26.99/0.857	27.41/0.861	27.59/ 0.863	26.91/0.841	27.61/0.863	27.60/0.863
Average	22.04/0.638	24.45/0.753	24.79/0.768	26.53/0.779	26.97/0.824	27.54/0.826	27.88/ 0.836	26.46/0.756	27.89/0.833	27.77/0.828

Table 4 The average value of PSNR/SSIM for different algorithms under different Gaussian noise levels

Blur	Method	$\sigma_\varepsilon = 3$	$\sigma_\varepsilon = 5$	$\sigma_\varepsilon = 7$
GB(9, 5)	Degraded	22.36/0.653	22.13/0.593	21.84/0.538
	TV [20]	24.55/0.759	24.66/0.764	24.75/0.761
	DCA [13]	24.83/0.770	24.99/0.770	24.69/0.756
	TRL2 [24]	26.32/0.774	25.35/0.734	24.95/0.739
	SOCF [11]	27.07/0.804	26.05/0.764	25.67/0.773
	BM3D [6]	27.26/0.819	26.35/0.782	25.74/0.771
	φ_1	27.62/0.830	26.65/0.792	26.10/0.789
	φ_2	26.21/0.739	25.44/0.710	24.96/0.712
	φ_3	27.60/0.830	26.65/0.800	26.06/0.785
	φ_4	27.59/0.825	26.66/0.804	26.10/0.784

Table 5 The average value of PSNR/SSIM for different algorithms under different Gaussian noise levels

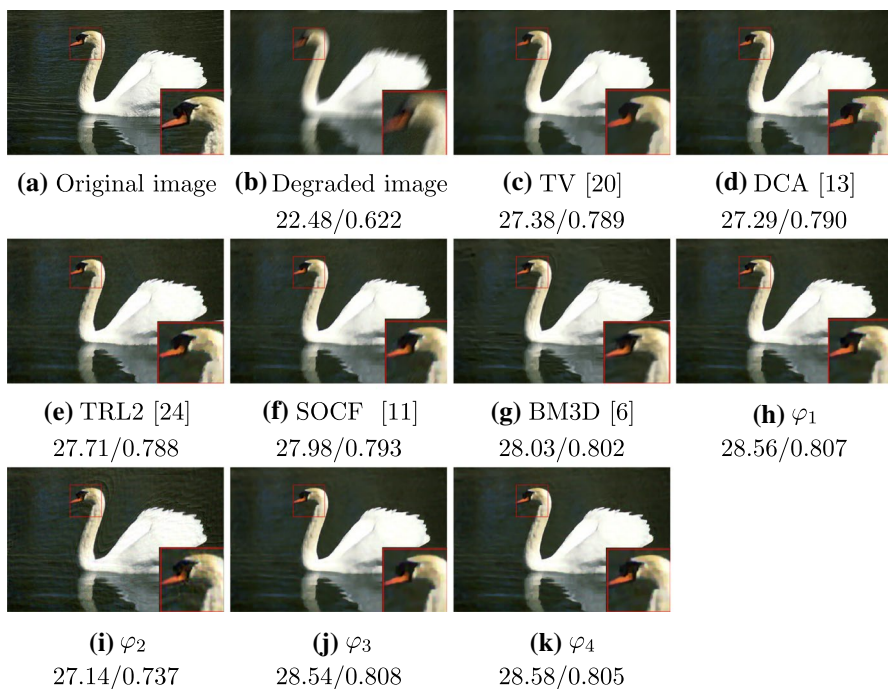
Blur	Method	$\sigma_\varepsilon = 3$	$\sigma_\varepsilon = 5$	$\sigma_\varepsilon = 7$
MB(20, 60)	Degraded	19.90/0.627	20.70/0.587	20.47/0.500
	TV [20]	26.12/0.807	23.86/0.741	24.81/0.767
	DCA [13]	26.48/0.812	25.05/0.768	25.20/0.759
	TRL2 [24]	26.56/0.759	24.53/0.661	24.43/0.706
	SOCF [11]	27.14/0.824	26.00/0.769	25.20/0.738
	BM3D [6]	27.69/0.821	26.37/0.789	25.52/0.770
	φ_1	27.85/0.833	26.58/0.787	25.92/0.778
	φ_2	26.14/0.724	25.15/0.700	24.50/0.678
	φ_3	28.02/0.835	26.78/0.796	26.00/0.782
	φ_4	28.64/0.836	26.77/0.796	26.01/0.782

Table 6 The average value of PSNR/SSIM for different algorithms under different Gaussian noise levels

Blur	Method	$\sigma_\varepsilon = 3$	$\sigma_\varepsilon = 5$	$\sigma_\varepsilon = 7$
AB(9, 9)	Degraded	22.04/0.638	21.84/0.579	21.56/0.516
	TV [20]	24.45/0.753	24.42/0.752	25.24/0.772
	DCA [13]	24.79/0.768	24.92/0.768	25.09/0.772
	TRL2 [24]	26.53/0.779	25.22/0.696	25.04/0.740
	SOCF [11]	26.97/0.824	26.47/0.800	25.85/0.781
	BM3D [6]	27.54/0.826	26.58/0.798	25.96/0.779
	φ_1	27.88/0.836	26.90/0.800	26.27/0.793
	φ_2	26.46/0.756	25.61/0.712	25.08/0.699
	φ_3	27.89/0.833	26.93/0.809	26.28/0.786
	φ_4	27.77/0.828	26.92/0.804	26.29/0.793

Table 7 The average time for all compared methods

Image	Blur	Method	Time	Blur	Time	Blur	Time
512 × 512 gray 768 × 512 RGB	GB(9, 5)	TV [20]	1.97	MB(20, 60)	2.04	AB(9, 9)	1.84
			2.06		1.86		2.12
		DCA [13]	284.02		263.39		152.97
			408.62		321.50		262.73
		TRL2 [24]	21.06		21.90		20.38
			23.30		29.35		32.20
		SOCF [11]	20.70		28.91		22.03
			174.25		266.01		193.18
		BM3D [6]	4.34		4.39		4.42
			8.46		7.80		7.47
		φ_1	32.42		38.72		53.96
			178.76		162.20		126.54
		φ_2	25.54		22.60		34.71
			165.72		103.79		160.37
		φ_3	24.31		51.96		51.31
			54.36		203.06		181.94
		φ_4	39.46		43.85		41.03
			181.02		175.02		179.04

**Fig. 4** Deblurring result of MB(20, 60) and $\sigma_\epsilon = 5$ for gray image “Duck” with zoomed areas and PSNR values and SSIM values

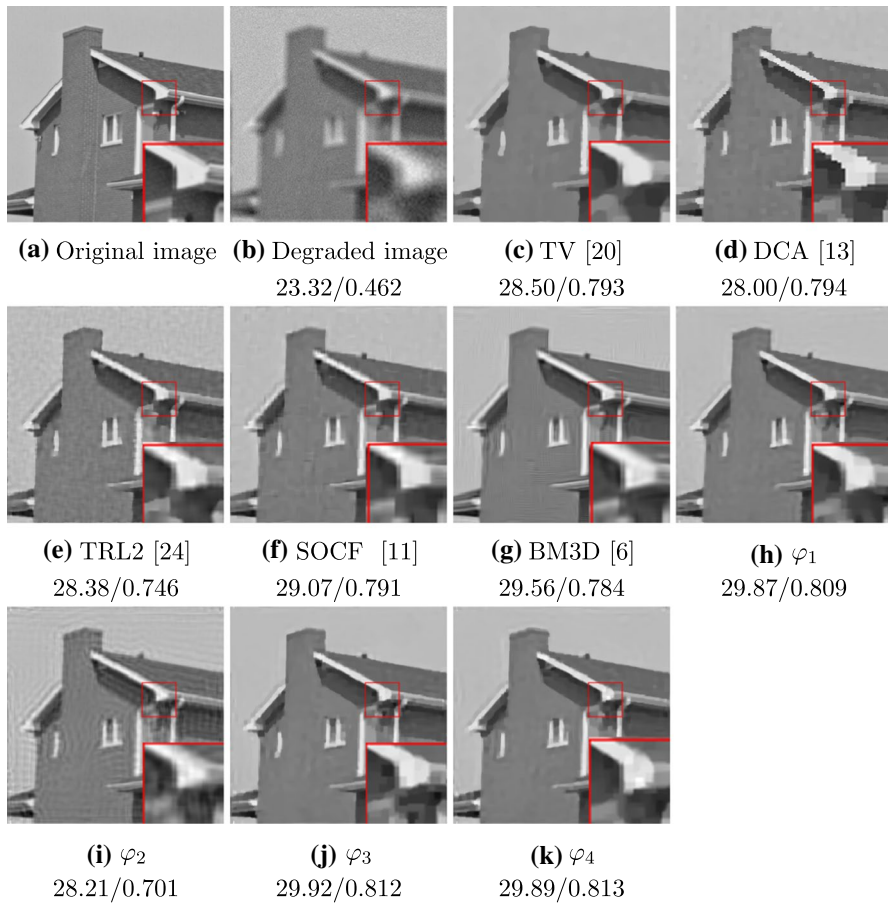


Fig. 5 Deblurring result of AB(20, 60) and $\sigma_\epsilon = 7$ for gray image “House” with zoomed areas and PSNR values and SSIM values

Experiment 2 In this experiment, on the premise of the same blur kernel, we provide the average value of PSNR and SSIM of different methods with different Gaussian noise levels for all test images. The results are shown in Tables 4, 5 and 6. From these tables, we find that the average value of PSNR and SSIM reduces as the noise level increases. We observe the same pattern that Algorithm 1 provides higher average values by using function φ_l for $l = 1, 2, 3, 4$ and has lower average values by using function φ_2 .

Experiment 3 In this experiment, we try to do comparison based on CPU time. In our test, we take 5 gray images with size 512×512 and one color image with size 768×512 . Then, for each algorithm, we get two data, one is the average time of these 5 grey images, and the other is the time of the given color image, see Table 7.

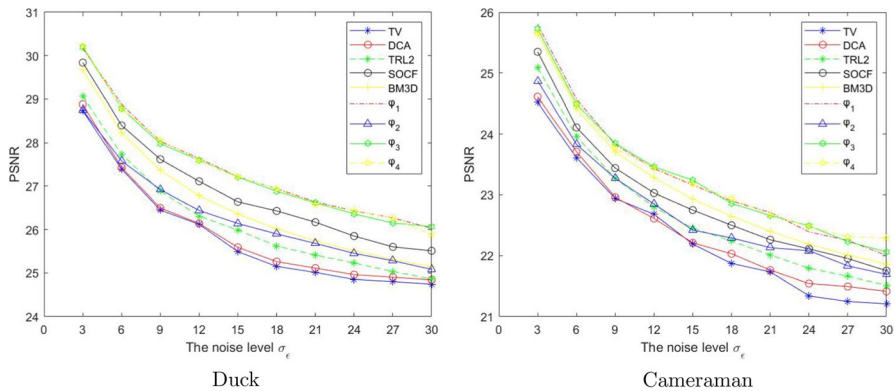


Fig. 6 The convergence behavior: the noise level vs. PSNR

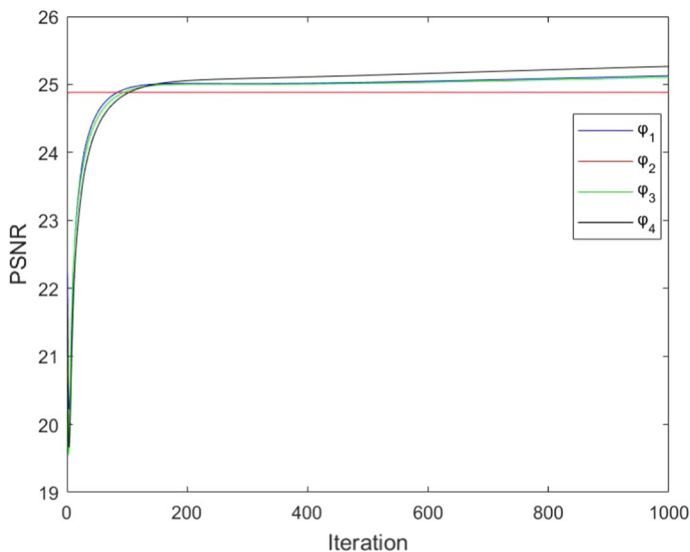


Fig. 7 The convergence behavior: the iteration vs. PSNR

From Table 7, we observe that our algorithm takes more CPU time, this is caused by the fact that Algorithm 1 needs to evaluate the function values φ_l for $l = 1, 2, 3, 4$ at each step. Acceleration will be considered in the future work.

Experiment 4 In this test, we present detailed information of restored images with zoomed area for better illustration. Figures 4 and 5 show that our method yields the better image quality in terms of deblurring and removing noise.

Experiment 5 In this test, we observe and discuss the overall convergence behavior of our algorithm. First, we do experiments about the noise level versus PSNR. We compare our algorithm with five other algorithms: TV, DCA, TRL2, SOCF and BM3D. We select the gray image Cameraman and the color image Duck under GB(9, 5), see Fig. 6. On the other hand, we do experiments about the iteration versus PSNR for our four functions φ_l for $l = 1, 2, 3, 4$, and we choose the gray image Cameraman under AB(9, 9)/ $\sigma_\epsilon = 3$, see Fig. 7.

According to the simulations and numerical results in Tables 1, 2, 3, 4, 5, 6 and Table 7, we summarize our numerical findings:

1. The SP-ADMM with the functions φ_1 , φ_3 and φ_4 provide higher PSNR and SSIM. On the whole, Algorithm 1 with the smoothing function φ_4 provides the best results, and the functions φ_1 and φ_3 have similar effects, the performance of our algorithm with the function φ_2 is relatively poor. Taking into account both the PSNR and SSIM, which are two important indicators for evaluating image quality, we believe that our algorithm provides better performance among all algorithms considered as it can solve more images at higher PSNR and SSIM. In this respect, our algorithm has a major advantage.
2. Even though Algorithm 1 takes more CPU time of convergence, our method still has high image reconstruction effect when the noise increases.
3. While BM3D [6] is often the best solver among other algorithms considered in this paper, we note that our algorithm has highest average value of PSNR/SSIM for different algorithms under different Gaussian noise levels. In this sense, our algorithm has good generalization ability.
4. The visual results of image deblurring show that the proposed method yields the better image quality in terms of deblurring and removing noises.
5. Analyzing all the numerical results, we can be sure that smoothing method can deal with image reconstruction problem well. Usually, In the field of image reconstruction, solving the TV part requires soft threshold operator. Thus, this paper provides a new technical support for image reconstruction from the mathematical point of view.

5 Conclusion

In this paper, the image deblurring problem is investigated. We build up a new smoothing model (5), which employs smoothing approximations $\varphi_l(\mu_1, t)$ for $l = 1, 2, 3, 4$. We propose an unified smoothing approach to solve the image deblurring problem. On the whole, the smoothing strategy along with SP-ADMM for image deblurring is the main contribution of this paper, both theoretically and numerically. In particular, from various experiments and comparisons, we suggest that all φ_l for $l = 1, 3, 4$ except for φ_2 are good choices to work with Algorithm 1. In addition, it is observed that our algorithm gives relatively better behavior whether in

terms of the visual quality or the values of PSNR and SSIM. We also zoom in key parts of the restored image for better illustration. In brief, we provide a new effective way to deal with the image deblurring problem.

Acknowledgements C. Wu and X. Chen: The research is supported by the Natural Science Foundation of Inner Mongolia Autonomous Region (2018MS01016). Q. Jin: The research is supported by the National Natural Science Foundation of China (12061052) and the Natural Science Fund of Inner Mongolia Autonomous Region (2020MS01002). J.-S. Chen: The research is supported by Ministry of Science and Technology, Taiwan.

References

1. Ali, G., Mohammad, H.: A balanced combination of Tikhonov and total variation regularizations for reconstruction of piecewise-smooth signals. *Signal Process.* **93**(7), 1945–1960 (2013)
2. Bai, Y., Jia, H., Jiang, M., Liu, X., Xie, X., Gao, W.: Single-image blind deblurring using multi-scale latent structure prior. *IEEE Trans. Circuits Syst. Video Technol.* **30**(7), 2033–2045 (2020)
3. Bertocchi, C., Chouzenoux, E., Corbineau, M., Pesquet, J., Prato, M.: Deep unfolding of a proximal interior point method for image restoration. *Inverse Probab.* **36**(3), 034005 (27pp) (2019)
4. Buccini, A., Donatelli, M., Ramlau, R.: A semiblind regularization algorithm for inverse problems with application to image deblurring. *SIAM J. Sci. Comput.* **40**(1), 452–483 (2018)
5. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3d transform-domain collaborative filtering. *IEEE Trans. Image Process.* **16**(8), 2080–2095 (2007)
6. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image restoration by sparse 3d transform-domain collaborative filtering. In: *Proceedings of SPIE—The International Society for Optical Engineering* (2008)
7. Fazel, M., Png, T.K., Sun, D., Tseng, P.: Hankel matrix rank minimization with applications to system identification and realization. *SIAM J. Matrix Anal. Appl.* **34**(3), 946–977 (2013)
8. Gazzola, S., Kilmer, M., Nagy, J., Semerci, O., Miller, E.: An inner-outer iterative method for edge preservation in image restoration and reconstruction. *Inverse Probab.* **36**(12), 124004 (33pp) (2020)
9. Gholami, A., Hosseini, S.: A balanced combination of Tikhonov and total variation regularizations for reconstruction of piecewise-smooth signals. *Signal Process.* **93**(7), 1945–1960 (2013)
10. Liu, K., Tan, J., Ai, L.: Hybrid regularization-based adaptive anisotropic diffusion for image denoising. *SpringerPlus* **5**(1), 24 (2016)
11. Liu, J., Lou, Y., Ni, G., Zeng, T.: An image sharpening operator combined with framelet for image deblurring. *Inverse Probab.* **36**(4), 045015(29pp) (2020)
12. Liu, K., Tan, J., Su, B.: An adaptive image denoising model based on Tikhonov and tv regularizations. *Adv. Multimed.* **1–10**, 2014 (2014)
13. Lou, Y., Zeng, T., Osher, S., Xin, J.: A weighted difference of anisotropic and isotropic total variation model for image processing. *SIAM J. Imaging Sci.* **8**(3), 1798–1823 (2015)
14. Lu, J., Qiao, K., Li, X., Lu, Z., Zou, Y.: l_0 -minimization methods for image restoration problems based on wavelet frames. *Inverse Probab.* **35**(6), 064001(25pp) (2019)
15. Lysaker, M., Lundervold, A., Tai, X.: Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time. *IEEE Trans. Image Process.* **12**(12), 1579–1590 (2003)
16. Mei, J., Dong, Y., Huang, T.: Simultaneous image fusion and denoising by using fractional-order gradient information. *J. Comput. Appl. Math.* **351**, 212–227 (2018)
17. Nguyen, C., Saheya, B., Chang, Y., Chen, J.-S.: Unified smoothing functions for absolute value equation associated with second-order cone. *Appl. Numer. Math.* **135**, 206–227 (2019)
18. Oh, S., Woo, H., Yun, S., Kang, M.: Non-convex hybrid total variation for image denoising. *J. Vis. Commun. Image Represent.* **24**(3), 332–344 (2013)
19. Rioux, G., Choksi, R., Hoheisel, T., Pierre, M., Scarvelis, C.: The maximum entropy on the mean method for image deblurring. *Inverse Probab.* **37**(1), 015011(29pp) (2021)
20. Rudin, L., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. *Phys. D Nonlinear Phenom.* **60**, 259–268 (1992)

21. Song, X., Xu, Y., Dong, F.: A hybrid regularization method combining Tikhonov with total variation for electrical resistance tomography. *Flow Meas. Instrum.* **46**, 268–275 (2015)
22. Wang, Y., Yang, J., Yin, W., Zhang, Y.: A new alternating minimization algorithm for total variation image reconstruction. *SIAM J. Imaging Sci.* **1**(3), 248–272 (2008)
23. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* **13**(4), 600–612 (2004)
24. Wu, C., Liu, Z., Wen, S.: A general truncated regularization framework for contrast-preserving variational signal and image restoration: Motivation and implementation. *Sci. China Math.* **61**(9), 1711–1732 (2018)
25. Wu, C., Wang, J., Alcantara, J., Chen, J.-S.: Smoothing strategy along with conjugate gradient algorithm for signal reconstruction. *J. Sci. Comput.* **87**(1), 21 (2021)
26. Wu, C., Zhan, J., Lu, Y., Chen, J.-S.: Signal reconstruction by conjugate gradient algorithm based on smoothing l_1 norm. *Calcolo* **56**(4), 42 (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.