Recursive SURE for image recovery via total variation minimization

Feng Xue, Jiaqi Liu & Xia Ai

Signal, Image and Video Processing

ISSN 1863-1703
Volume 13
Number 4

SIViP (2019) 13:795-803
DOI 10.1007/s11760-019-01415-6

Springer

Signal, Image and Video Processing (2019) 13:795–803
https://doi.org/10.1007/s11760-019-01415-6

ORIGINAL PAPER

Recursive SURE for image recovery via total variation minimization

Feng Xue[1] · Jiaqi Liu[1] · Xia Ai[1]

Received: 28 October 2017 / Revised: 31 October 2018 / Accepted: 3 January 2019 / Published online: 12 January 2019

Abstract

Recently, total variation regularization has become a standard technique, and even a basic tool for image denoising and deconvolution. Generally, the recovery quality strongly depends on the regularization parameter. In this work, we develop a recursive evaluation of Stein's unbiased risk estimate (SURE) for the parameter selection, based on specific reconstruction algorithms. It enables us to monitor the evolution of mean squared error (MSE) during the iterations. In particular, to deal with large-scale data, we propose a Monte Carlo simulation for the practical computation of SURE, which is free of any explicit matrix operation. Experimental results show that the proposed recursive SURE could lead to highly accurate estimate of regularization parameter and nearly optimal restoration performance in terms of MSE.

Keywords Total variation · Denoising · Deconvolution · Stein's unbiased risk estimate (SURE) · Jacobian recursion

1 Introduction

Problem statement—Consider the standard image recovery problem: find a good estimate of original image $\mathbf{x}_0 \in \mathbb{R}^N$ from the following degradation model [15,20,22]:

$$\mathbf{y} = \mathbf{H}\mathbf{x}_0 + \epsilon \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^N$ is the observed image, $\mathbf{H} \in \mathbb{R}^{N\times N}$ denotes the observation matrix, which represents either identity for denoising or convolution for deconvolution, and $\epsilon \in \mathbb{R}^N$ is an additive white Gaussian noise with known variance $\sigma^2 > 0$.

Since the seminal work of ROF [12], total variation (TV) regularization has become a standard technique [17,19]:

$$\widehat{\mathbf{x}}_\lambda = \arg\min_{\mathbf{x}} \underbrace{\frac{1}{2}\|\mathbf{H}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda \cdot \mathrm{TV}(\mathbf{x})}_{\mathcal{L}(\mathbf{x})} \tag{2}$$

Here, λ is a regularization parameter, which is essential for the recovery quality of $\widehat{\mathbf{x}}_\lambda$. The isotropic TV term is defined as [9]:

$$\mathrm{TV}(\mathbf{x}) = \sum_{n=1}^{N} \sqrt{\left|(\mathbf{D}_1\mathbf{x})_n\right|^2 + \left|(\mathbf{D}_2\mathbf{x})_n\right|^2 + \alpha} \tag{3}$$

where $\mathbf{D}_1$ and $\mathbf{D}_2$ denote the horizontal and vertical first-order differences, respectively. The parameter $\alpha = 0$ corresponds to the standard TV definition. We use the smooth approximation with small $\alpha > 0$, since it simplifies numerical computations due to the differentiability [9].

TV is particularly effective for recovering those signals with piecewise constant region while preserving edges [12]. Recently, people extended the basic TV-norm to more general form of $\varphi(\|\mathbf{D}\mathbf{x}\|_2)$ that models *a priori* of the first-order gradients of an image [13]. Here, φ is a potential, possibly non-convex, function. It is reduced to the standard TV-norm when $\varphi(t) = t$. In this work, we focus on the TV minimization and attempt to find a proper value of λ for a good restoration quality. This work may help to gain some insights into more complicated function of φ.

Related works—There have been a number of criteria for this selection of λ, for example:

– Generalized cross-validation [5]: It is often used for linear estimates, not applicable for the nonlinear reconstruction considered here.
– L-curve method [7]: This procedure is not fully automated and often requires hand tuning or selection.

This work was supported by the National Natural Science Foundation of China (No. 61401013).

✉ Feng Xue
fxue@hotmail.com

[1] National Key Laboratory of Science and Technology on Test Physics and Numerical Mathematics, Beijing 100076, China

– Discrepancy principle [8]: This criterion is easy to compute and however may cause a loss of restoration quality.[1]

In this paper, we quantify the restoration performance by the mean squared error (MSE) [1,22]:

$$\text{MSE} = \frac{1}{N}\mathbb{E}\left\{\|\widehat{\mathbf{x}}_\lambda - \mathbf{x}_0\|_2^2\right\} \tag{4}$$

and attempt to select a value of λ, such that the corresponding solution $\widehat{\mathbf{x}}_\lambda$ achieves minimum MSE.

Notice that the MSE is inaccessible due to the unknown $\mathbf{x}_0$. In practice, Stein's unbiased risk estimate (SURE) has been proposed as a statistical substitute for MSE [1,16]:

$$\begin{aligned}\text{SURE} = \frac{1}{N}\Big(\|\widehat{\mathbf{x}}_\lambda\|_2^2 - 2\mathbf{y}^{\mathrm{T}}\mathbf{H}^{-\mathrm{T}}\widehat{\mathbf{x}}_\lambda + 2\sigma^2\mathrm{Tr}\big(\mathbf{H}^{-\mathrm{T}}\mathbf{J}_{\mathbf{y}}(\widehat{\mathbf{x}}_\lambda)\big)\Big)\\ +\frac{1}{N}\|\mathbf{x}_0\|_2^2\end{aligned} \tag{5}$$

since it depends on the observed data $\mathbf{y}$ only.[2] Tr in (5) denotes the matrix trace. Here, $\mathbf{J}_{\mathbf{y}}(\widehat{\mathbf{x}}_\lambda) \in \mathbb{R}^{N\times N}$ is a Jacobian matrix defined as [21,23]:

$$\left[\mathbf{J}_{\mathbf{y}}(\widehat{\mathbf{x}}_\lambda)\right]_{m,n} = \frac{\partial(\widehat{\mathbf{x}}_\lambda)_m}{\partial y_n}$$

The statistical unbiasedness of SURE w.r.t. true MSE has been proved in [22]. Recently, SURE has become a popular criterion for parameter selection, in the context of nonlinear denoising/deconvolution [1,22], and ℓ_1-based sparse recovery [18,23]. However, to our best knowledge, there are very few researches on the application of SURE to TV-based reconstruction, which is the purpose of this paper.

Our contributions—Our main contributions are twofold. First, we develop a recursive evaluation of SURE during the reconstruction iterations, which finally provides a reliable estimate of the MSE for the TV-based recovery. Second, the Monte Carlo (MC) simulation is used to facilitate the SURE computation for large-scale data, without explicit matrix operation.

Additional remarks—Throughout this paper, we use boldface lowercase letters, e.g. $\mathbf{x} \in \mathbb{R}^N$, to denote N-dimensional real vectors, where N is typically the number of pixels in an image. The matrices are denoted by boldface uppercase letters, e.g. $\mathbf{A} \in \mathbb{R}^{M\times N}$. $\mathbf{A}^{\mathrm{T}} \in \mathbb{R}^{N\times M}$ denotes the transpose of matrix $\mathbf{A}$. The superscripts (i) and (j) denote the iterative indices of outer or inner loops. The notation diag($\mathbf{v}$) transforms the vector $\mathbf{v}$ to the diagonal matrix $\mathbf{V}$ with $\mathbf{V}_{n,n} = v_n$.

[1] See Sect. 4 for the complete comparisons between discrepancy principle and the proposed SURE.

[2] Note that the last constant term—$\|\mathbf{x}_0\|_2^2/N$—is irrelevant to the optimization of $\widehat{\mathbf{x}}_\lambda$.

2 Recursive evaluation of SURE for TV denoising

Now, we consider image denoising problem, i.e. $\mathbf{H} = \mathbf{I}$ in (2). To perform the SURE-based selection of λ, we need to compute the solution $\widehat{\mathbf{x}}_\lambda$ and its SURE.

2.1 Basic scheme of Chambolle's algorithm [2]

Many algorithms can be used to find the TV solution $\widehat{\mathbf{x}}_\lambda$, e.g. [3,4,6,11,14]. Here, we apply a dual-based iterative projection algorithm—Chambolle's algorithm—to solve (2), since it is one of the most popular TV minimization solvers and has been extensively used in the recent decade. The original form of the Chambolle's iteration was described in [2]. Now, we rewrite the algorithm in matrix language:

$$\mathbf{u}^{(i+1)} = \overline{\mathbf{V}}^{(i)}\Bigg(\overbrace{\mathbf{u}^{(i)} - \frac{\tau}{\lambda}\mathbf{D}\underbrace{(\mathbf{y} + \lambda\mathbf{D}^{\mathrm{T}}\mathbf{u}^{(i)})}_{\mathbf{x}^{(i)}}}^{\mathbf{w}^{(i)}}\Bigg) \tag{6}$$

where τ is a step size and the gradient operator $\mathbf{D}$ is $\mathbf{D} = [\mathbf{D}_1^{\mathrm{T}}, \mathbf{D}_2^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{2N\times N}$ [$\mathbf{D}_1$ and $\mathbf{D}_2$ are the same as in (3)]. The vector $\mathbf{u}^{(i)} \in \mathbb{R}^{2N}$ lives in $\mathbf{D}$-domain of an image, which can be transformed back to the image domain by $\mathbf{D}^{\mathrm{T}}\mathbf{u}$. The diagonal matrix $\overline{\mathbf{V}}^{(i)}$ is:

$$\overline{\mathbf{V}}^{(i)} = \begin{bmatrix}\mathbf{V}^{(i)} & \mathbf{0}\\ \mathbf{0} & \mathbf{V}^{(i)}\end{bmatrix} \in \mathbb{R}^{2N\times 2N}$$

where the diagonal block $\mathbf{V}^{(i)} \in \mathbb{R}^{N\times N}$ is given by:

$$\mathbf{V}_{n,n}^{(i)} = \left(1 + \frac{\tau}{\lambda}\sqrt{\big((\mathbf{D}_1\mathbf{x}^{(i)})_n\big)^2 + \big((\mathbf{D}_2\mathbf{x}^{(i)})_n\big)^2 + \alpha}\right)^{-1}$$

2.2 Recursive evaluation of SURE

The next question is how to compute the SURE of the TV-denoised image $\widehat{\mathbf{x}}_\lambda$? We propose to compute the SURE of $\mathbf{x}^{(i)}$ during each iteration until final convergence. The similar treatment has been used in [18,21,23] for ℓ_1-based sparse deconvolution.

The SURE for the ith update is (noting $\mathbf{H} = \mathbf{I}$ for denoising problem):

$$\text{SURE} = \frac{1}{N}\|\mathbf{x}^{(i)} - \mathbf{y}\|_2^2 + \frac{2\sigma^2}{N}\mathrm{Tr}\left(\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\right) - \sigma^2 \tag{7}$$

The Jacobian matrix $\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})$ can be evaluated by the basic calculus as follows.

First, we split (6) into two parts—vertical and horizontal differences:

$$\underbrace{\begin{bmatrix} \mathbf{u}_1^{(i+1)} \\ \mathbf{u}_2^{(i+1)} \end{bmatrix}}_{\mathbf{u}^{(i+1)}} = \underbrace{\begin{bmatrix} \mathbf{V}^{(i)} & \mathbf{0} \\ \mathbf{0} & \mathbf{V}^{(i)} \end{bmatrix}}_{\overline{\mathbf{V}}^{(i)}} \underbrace{\begin{bmatrix} \mathbf{w}_1^{(i)} \\ \mathbf{w}_2^{(i)} \end{bmatrix}}_{\mathbf{w}^{(i)}}$$

From (6), the Jacobian matrix of $\mathbf{u}_1^{(i+1)}$ is:

$$\left[\mathbf{J_y}(\mathbf{u}_1^{(i+1)})\right]_{m,n} = \left(\mathbf{w}_1^{(i)}\right)_m \frac{\partial \mathbf{V}_{m,m}^{(i)}}{\partial y_n} + \mathbf{V}_{m,m}^{(i)} \frac{\partial (\mathbf{w}_1^{(i)})_m}{\partial y_n}$$

Let $\mathbf{a} = \mathbf{D}_1\mathbf{x}^{(i)}$ and $\mathbf{b} = \mathbf{D}_2\mathbf{x}^{(i)}$, we have:

$$\begin{aligned} \frac{\partial \mathbf{V}_{m,m}^{(i)}}{\partial y_n} &= \frac{\partial \mathbf{V}_{m,m}^{(i)}}{\partial a_m} \cdot \frac{\partial a_m}{\partial y_n} + \frac{\partial \mathbf{V}_{m,m}^{(i)}}{\partial b_m} \cdot \frac{\partial b_m}{\partial y_n} \\ &= -\frac{\tau}{\lambda} \cdot \underbrace{\frac{a_m (\mathbf{V}_{m,m}^{(i)})^2}{\sqrt{a_m^2 + b_m^2 + \alpha}}}_{(\mathbf{C}_1^{(i)})_{m,m}} \cdot \frac{\partial a_m}{\partial y_n} \\ &\quad -\frac{\tau}{\lambda} \cdot \underbrace{\frac{b_m (\mathbf{V}_{m,m}^{(i)})^2}{\sqrt{a_m^2 + b_m^2 + \alpha}}}_{(\mathbf{C}_2^{(i)})_{m,m}} \cdot \frac{\partial b_m}{\partial y_n} \\ &= -\frac{\tau}{\lambda} \left[\mathbf{C}_1^{(i)} \mathbf{D}_1 \mathbf{J_y}(\mathbf{x}^{(i)}) - \mathbf{C}_2^{(i)} \mathbf{D}_2 \mathbf{J_y}(\mathbf{x}^{(i)})\right]_{m,n} \end{aligned}$$

Thus, we obtain:

$$\begin{aligned} \mathbf{J_y}\left(\mathbf{u}_1^{(i+1)}\right) &= -\frac{\tau}{\lambda} \mathbf{W}_1^{(i)} \left(\mathbf{C}_1^{(i)} \mathbf{D}_1 + \mathbf{C}_2^{(i)} \mathbf{D}_2\right) \mathbf{J_y}(\mathbf{x}^{(i)}) \\ &\quad + \mathbf{V}^{(i)} \mathbf{J_y}(\mathbf{w}_1^{(i)}) \end{aligned} \tag{8}$$

where diagonal matrix $\mathbf{W}_1^{(i)} = \operatorname{diag}(\mathbf{w}_1^{(i)})$.

Similarly, $\mathbf{J_y}(\mathbf{u}_2^{(i+1)})$ is given by:

$$\begin{aligned} \mathbf{J_y}\left(\mathbf{u}_2^{(i+1)}\right) &= -\frac{\tau}{\lambda} \mathbf{W}_2^{(i)} \left(\mathbf{C}_1^{(i)} \mathbf{D}_1 + \mathbf{C}_2^{(i)} \mathbf{D}_2\right) \mathbf{J_y}(\mathbf{x}^{(i)}) \\ &\quad + \mathbf{V}^{(i)} \mathbf{J_y}\left(\mathbf{w}_2^{(i)}\right) \end{aligned} \tag{9}$$

By the basic property of Jacobian matrix [23], we further have $\mathbf{J_y}(\mathbf{w}_s^{(i)}) = \mathbf{J_y}(\mathbf{u}_s^{(i)}) - \frac{\tau}{\lambda} \mathbf{D}_s \mathbf{J_y}(\mathbf{x}^{(i)})$ for $s = 1, 2$. Substituting $\mathbf{J_y}(\mathbf{w}^{(i)})$ into $\mathbf{J_y}(\mathbf{u}^{(i)})$ yields (after rearrangements):

$$\begin{aligned} \mathbf{J_y}(\mathbf{u}^{(i+1)}) &= \begin{bmatrix} \mathbf{J_y}\left(\mathbf{u}_1^{(i+1)}\right) \\ \mathbf{J_y}\left(\mathbf{u}_2^{(i+1)}\right) \end{bmatrix} = \begin{bmatrix} \mathbf{V}^{(i)} & \mathbf{0} \\ \mathbf{0} & \mathbf{V}^{(i)} \end{bmatrix} \mathbf{J_y}(\mathbf{u}^{(i)}) - \frac{\tau}{\lambda} \cdot \\ &\quad \begin{bmatrix} \mathbf{W}_1^{(i)} \mathbf{C}_1^{(i)} + \mathbf{V}^{(i)} & \mathbf{W}_1^{(i)} \mathbf{C}_2^{(i)} \\ \mathbf{W}_2^{(i)} \mathbf{C}_1^{(i)} & \mathbf{W}_2^{(i)} \mathbf{C}_2^{(i)} + \mathbf{V}^{(i)} \end{bmatrix} \mathbf{D} \mathbf{J_y}(\mathbf{x}^{(i)}) \end{aligned} \tag{10}$$

Noting that $\mathbf{x}^{(i)} = \mathbf{y} + \lambda \mathbf{D}^{\mathrm{T}} \mathbf{u}^{(i)}$ in (6), we have:

$$\mathbf{J_y}(\mathbf{x}^{(i)}) = \mathbf{I} + \lambda \mathbf{D}_1^{\mathrm{T}} \mathbf{J_y}\left(\mathbf{u}_1^{(i)}\right) + \lambda \mathbf{D}_2^{\mathrm{T}} \mathbf{J_y}\left(\mathbf{u}_2^{(i)}\right) \tag{11}$$

Thus, the Jacobian matrix $\mathbf{J_y}(\mathbf{x}^{(i)})$ can be evaluated in this recursive manner, until the convergence of Chambolle's iteration, summarized in Algorithm 1.

Algorithm 1: SURE evaluation for Chambolle's denoising algorithm

Input: $\mathbf{y}, \lambda, \alpha, \tau$, initial $\mathbf{u}^{(0)}$
Output: reconstructed $\widehat{\mathbf{x}}_\lambda$ and $\mathrm{SURE}(\widehat{\mathbf{x}}_\lambda)$
for $i = 1, 2, \ldots$ *(Chambolle's iteration)* **do**
1 compute $\mathbf{x}^{(i)}$ by (6);
2 update $\mathbf{J_y}(\mathbf{x}^{(i)})$ by (10) and (11);
3 compute SURE of ith iterate by (7);
end

This algorithm enables us to solve the TV denoising problem with a prescribed value of λ and simultaneously evaluate the SURE during the Chambolle's iterations.

2.3 Monte Carlo for practical computation

From (10)–(11), we can see that the Jacobian recursions require the explicit matrix computations. However, for a typical image of size 256×256, the related matrices, e.g. $\mathbf{W}_s^{(i)}$, $\mathbf{C}_s^{(i)}$, $\mathbf{V}^{(i)}$ and $\mathbf{D}_s$ $(s = 1, 2)$, are of size $256^2 \times 256^2$. Due to the limited computational resources (e.g. RAM), it is impractical to explicitly store and compute such the huge matrices. Thus, the Jacobian recursions cannot be computed in the explicit matrix form of (10)–(11).

However, Monte Carlo simulation provides an alternative way to compute the trace by the following fact [1]:

$$\operatorname{Tr}\left(\mathbf{J_y}(\mathbf{x}^{(i)})\right) = \mathbb{E}\left\{\mathbf{n}_0^{\mathrm{T}} \underbrace{\mathbf{J_y}(\mathbf{x}^{(i)}) \mathbf{n}_0}_{\mathbf{n}_{\mathbf{x}}^{(i)}}\right\} \tag{12}$$

with input white Gaussian noise $\mathbf{n}_0 \sim \mathcal{N}(0, \mathbf{I}_N)$, provided that $\mathbf{n}_{\mathbf{x}}^{(i)}$ can be computed without explicit form of $\mathbf{J_y}(\mathbf{x}^{(i)})$.

Multiplying the input $\mathbf{n}_0$ on both sides of (10)–(11), we obtain:

$$\begin{cases} \overbrace{\mathbf{J_y}\left(\mathbf{u}_1^{(i+1)}\right) \mathbf{n}_0}^{\mathbf{n}_{\mathbf{u}_1}^{(i+1)}} = \mathbf{V}^{(i)} \overbrace{\mathbf{J_y}\left(\mathbf{u}_1^{(i)}\right) \mathbf{n}_0}^{\mathbf{n}_{\mathbf{u}_1}^{(i)}} - \frac{\tau}{\lambda} \mathbf{P}_1^{(i)} \overbrace{\mathbf{J_y}(\mathbf{x}^{(i)}) \mathbf{n}_0}^{\mathbf{n}_{\mathbf{x}}^{(i)}} \\ \underbrace{\mathbf{J_y}\left(\mathbf{u}_2^{(i+1)}\right) \mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_2}^{(i+1)}} = \mathbf{V}^{(i)} \underbrace{\mathbf{J_y}\left(\mathbf{u}_2^{(i)}\right) \mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_2}^{(i)}} - \frac{\tau}{\lambda} \mathbf{P}_2^{(i)} \underbrace{\mathbf{J_y}(\mathbf{x}^{(i)}) \mathbf{n}_0}_{\mathbf{n}_{\mathbf{x}}^{(i)}} \end{cases} \tag{13}$$

and

$$\underbrace{\mathbf{J_y}(\mathbf{x}^{(i)})\mathbf{n}_0}_{\mathbf{n}_\mathbf{x}^{(i)}} = \mathbf{n}_0 + \lambda \mathbf{D}_1^\mathrm{T} \underbrace{\mathbf{J_y}\left(\mathbf{u}_1^{(i)}\right)\mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_1}^{(i)}} + \lambda \mathbf{D}_2^\mathrm{T} \underbrace{\mathbf{J_y}\left(\mathbf{u}_2^{(i)}\right)\mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_2}^{(i)}} \tag{14}$$

where $\mathbf{P}_1^{(i)} = (\mathbf{W}_1^{(i)}\mathbf{C}_1^{(i)} + \mathbf{V}^{(i)})\mathbf{D}_1 + \mathbf{W}_1^{(i)}\mathbf{C}_2^{(i)}\mathbf{D}_2$ and $\mathbf{P}_2^{(i)} = \mathbf{W}_2^{(i)}\mathbf{C}_1^{(i)}\mathbf{D}_1 + (\mathbf{W}_2^{(i)}\mathbf{C}_2^{(i)} + \mathbf{V}^{(i)})\mathbf{D}_2$.

By MC simulation, the Jacobian recursions of (10)–(11) boil down to a simple noise evolution (13)–(14). We further have three remarks to facilitate the SURE-MC computations.

- $\mathbf{D}_1\mathbf{n}_0, \mathbf{D}_2\mathbf{n}_0, \mathbf{D}_1^\mathrm{T}\mathbf{n}_0$ and $\mathbf{D}_2^\mathrm{T}\mathbf{n}_0$ can be computed by simple first-order differences.
- The diagonal matrix–vector multiplications $\mathbf{W}_s^{(i)}\mathbf{n}_0$, $\mathbf{C}_s^{(i)}\mathbf{n}_0$ ($s = 1, 2$) and $\mathbf{V}^{(i)}\mathbf{n}_0$ are essentially simple componentwise products: there is no need to explicitly write out the full diagonal matrices.
- For simplicity, we initialize $\mathbf{u}^{(0)} = \mathbf{0}$, and hence $\mathbf{x}^{(0)} = \mathbf{y}$. Thus, $\mathbf{J_y}(\mathbf{u}^{(0)}) = \mathbf{0}$ and $\mathbf{J_y}(\mathbf{x}^{(0)}) = \mathbf{I}$, which yields $\mathbf{n}_\mathbf{u}^{(0)} = \mathbf{0}$ and $\mathbf{n}_\mathbf{x}^{(0)} = \mathbf{n}_0$.

Thus, all the computations of (13)–(14) can be efficiently performed by element-wise operations (e.g. scalar difference, multiplication, etc). We are able to evaluate the SURE without any explicit matrix computation, summarized as follows.

Algorithm 2: Monte Carlo counterpart of Algorithm 1

for $i = 1, 2, \ldots$ *(Chambolle's iteration)* **do**
  1 compute $\mathbf{x}^{(i)}$ by (6);
  2 compute $\mathbf{n}_{\mathbf{u}_1}^{(i)}$, $\mathbf{n}_{\mathbf{u}_2}^{(i)}$ and $\mathbf{n}_\mathbf{x}^{(i)}$ by (13) and (14);
  3 compute the trace of $\mathbf{J_y}(\mathbf{x}^{(i)})$ by (12);
  4 compute SURE of ith iterate by (7);
end

To find the optimal value of λ, we repeatedly implement Algorithm 2 for various tentative values of λ, and then, the minimum SURE indicates the optimal λ. This *global search* has been frequently used in [18,23].

3 Recursive evaluation of SURE for TV deconvolution

Following the similar procedure with Sect. 2, we now consider the TV deconvolution problem, where the SURE also requires to compute the solution $\widehat{\mathbf{x}}_\lambda$ and its SURE.

3.1 Basic scheme of ADMM

To find $\widehat{\mathbf{x}}_\lambda$, we choose a typical alternating direction method of multipliers (ADMM) for TV deconvolution [i.e. $\mathbf{H}$ being convolution in (1)] [17,19]. By the variable splitting [17,19], (2) is equivalent to the following problem:

$$\min_{\mathbf{x}} \ \frac{1}{2}\|\mathbf{Hx}-\mathbf{y}\|_2^2 + \lambda\cdot\mathrm{TV}(\mathbf{z}), \quad \text{s.t. } \mathbf{z}=\mathbf{x}$$

which, by Lagrangian, becomes:

$$\min_{\mathbf{x},\mathbf{z}} \ \frac{1}{2}\|\mathbf{Hx}-\mathbf{y}\|_2^2 + \lambda\cdot\mathrm{TV}(\mathbf{z}) + \frac{\mu}{2}\|\mathbf{z}-\mathbf{x}\|_2^2$$

where μ is an augmented Lagrangian penalty parameter. The ADMM alternatively minimizes this functional w.r.t. both variables $\mathbf{x}$ and $\mathbf{z}$ (iterate on i):

$$\begin{cases} \mathbf{x}^{(i)} &= \arg\min_\mathbf{x} \frac{1}{2}\|\mathbf{Hx}-\mathbf{y}\|_2^2 + \frac{\mu}{2}\|\mathbf{x}-\mathbf{z}^{(i)}\|_2^2 \\ \mathbf{z}^{(i+1)} &= \arg\min_\mathbf{z} \frac{\mu}{2}\|\mathbf{z}-\mathbf{x}^{(i)}\|_2^2 + \lambda\cdot\mathrm{TV}(\mathbf{z}) \end{cases}$$

i.e.

$$\begin{cases} \mathbf{x}^{(i)} &= \left(\mathbf{H}^\mathrm{T}\mathbf{H}+\mu\mathbf{I}\right)^{-1}\left(\mathbf{H}^\mathrm{T}\mathbf{y}+\mu\mathbf{z}^{(i)}\right) \\ \mathbf{z}^{(i+1)} &= \arg\min_\mathbf{z} \frac{1}{2}\|\mathbf{z}-\mathbf{x}^{(i)}\|_2^2 + \frac{\lambda}{\mu}\cdot\mathrm{TV}(\mathbf{z}) \end{cases} \tag{15}$$

We found that the update of $\mathbf{z}^{(i)}$ is essentially a TV denoising problem: estimate a 'denoised' version of a 'noisy' image $\mathbf{x}^{(i)}$. It can be efficiently obtained by Chambolle's algorithm [2] (with the index j, for the fixed i)[3]:

$$\mathbf{u}^{(i,j+1)} = \overline{\mathbf{V}}^{(i,j)}\left(\overbrace{\mathbf{u}^{(i,j)} - \frac{\tau\mu}{\lambda}\mathbf{D}\underbrace{\left(\mathbf{x}^{(i)}+\frac{\lambda}{\mu}\mathbf{D}^\mathrm{T}\mathbf{u}^{(i,j)}\right)}_{\mathbf{z}^{(i,j)}}}^{\mathbf{w}^{(i,j)}}\right) \tag{16}$$

where diagonal block $\mathbf{V}^{(i,j)}$ is given by:

$$\mathbf{V}_{n,n}^{(i,j)} = \left(1+\frac{\tau\mu}{\lambda}\sqrt{\left((\mathbf{D}_1\mathbf{z}^{(i,j)})_n\right)^2+\left((\mathbf{D}_2\mathbf{z}^{(i,j)})_n\right)^2+\alpha}\right)^{-1}$$

When the Chambolle's iteration (inner iterate on j) reaches the final convergence, $\mathbf{z}^{(i+1)}$ is updated as $\mathbf{z}^{(i+1)} = \mathbf{z}^{(i,\infty)}$.

3.2 Jacobian recursion of ADMM

For the deconvolution problem, the SURE needs to compute $\mathbf{H}^{-1}$ in (5). However, it is observed that for the ill-conditioned matrix $\mathbf{H}$, the simple inversion $\mathbf{H}^{-1}$ may cause numerical

[3] We can see that the Chambolle's iteration is readily incorporated into ADMM, see Sect. 2.1 for details.

instability of SURE [22]. Hence, we use the regularized inverse $\mathbf{H}_\beta^{-1}$ to replace $\mathbf{H}^{-1}$:

$$\mathbf{H}_\beta^{-1} = (\mathbf{H}^{\mathrm{T}}\mathbf{H} + \beta\mathbf{I})^{-1}\mathbf{H}^{\mathrm{T}}$$

with a parameter β. The regularized SURE becomes:

$$\begin{aligned}\mathrm{SURE} = \frac{1}{N}\Big(&\|\mathbf{x}^{(i)}\|_2^2 - 2\mathbf{y}^{\mathrm{T}}\mathbf{H}_\beta^{-\mathrm{T}}\mathbf{x}^{(i)} + 2\sigma^2\mathrm{Tr}\big(\mathbf{H}_\beta^{-\mathrm{T}}\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\big)\Big)\\ &+\frac{1}{N}\|\mathbf{x}_0\|_2^2 \end{aligned} \tag{17}$$

Refer to [18,22] for the similar treatment.

By the similar derivations with Sect. 2.2, we obtain the Jacobian recursions for ADMM as:

$$\begin{cases}\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)}) &= (\mathbf{H}^{\mathrm{T}}\mathbf{H} + \mu\mathbf{I})^{-1}\big(\mathbf{H}^{\mathrm{T}} + \mu\mathbf{J}_{\mathbf{y}}(\mathbf{z}^{(i)})\big)\\ \mathbf{J}_{\mathbf{y}}(\mathbf{u}_1^{(i,j+1)}) &= \mathbf{V}^{(i,j)}\mathbf{J}_{\mathbf{y}}(\mathbf{u}_1^{(i,j)}) - \frac{\tau}{\lambda}\mathbf{P}_1^{(i,j)}\mathbf{J}_{\mathbf{y}}(\mathbf{z}^{(i,j)})\\ \mathbf{J}_{\mathbf{y}}(\mathbf{u}_2^{(i,j+1)}) &= \mathbf{V}^{(i,j)}\mathbf{J}_{\mathbf{y}}(\mathbf{u}_2^{(i,j)}) - \frac{\tau}{\lambda}\mathbf{P}_2^{(i,j)}\mathbf{J}_{\mathbf{y}}(\mathbf{z}^{(i,j)})\\ \mathbf{J}_{\mathbf{y}}(\mathbf{z}^{(i,j)}) &= \mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)}) + \frac{\lambda}{\mu}\mathbf{D}^{\mathrm{T}}\mathbf{J}_{\mathbf{y}}(\mathbf{u}^{(i,j)})\end{cases} \tag{18}$$

where $\mathbf{P}_1^{(i,j)} = (\mathbf{W}_1^{(i,j)}\mathbf{C}_1^{(i,j)} + \mathbf{V}^{(i,j)})\mathbf{D}_1 + \mathbf{W}_1^{(i,j)}\mathbf{C}_2^{(i,j)}\mathbf{D}_2$ and $\mathbf{P}_2^{(i,j)} = \mathbf{W}_2^{(i,j)}\mathbf{C}_1^{(i,j)}\mathbf{D}_1 + (\mathbf{W}_2^{(i,j)}\mathbf{C}_2^{(i,j)} + \mathbf{V}^{(i,j)})\mathbf{D}_2$. Here, $\mathbf{W}_1^{(i,j)}$, $\mathbf{W}_2^{(i,j)}$, $\mathbf{C}_1^{(i,j)}$ and $\mathbf{C}_2^{(i,j)}$ are defined similarly with Sect. 2.2.

3.3 Monte Carlo for SURE evaluation

Similar to Sect. 2.3, we adopt Monte Carlo to evaluate the trace term of SURE as:

$$\mathrm{Tr}\big(\mathbf{H}_\beta^{-\mathrm{T}}\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\big) = \mathbb{E}\left\{\mathbf{n}_0^{\mathrm{T}}\mathbf{H}_\beta^{-\mathrm{T}}\underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{x}}^{(i)}}\right\} \tag{19}$$

with input white Gaussian noise $\mathbf{n}_0 \sim \mathcal{N}(0, \mathbf{I}_N)$. Then, multiplying $\mathbf{n}_0$ on both sides of (18), we obtain the noise evolution during ADMM:

$$\begin{cases}\mathbf{n}_{\mathbf{x}}^{(i)} &= \mathbf{B}^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{n}_0 + \mu\mathbf{B}^{-1}\mathbf{n}_{\mathbf{z}}^{(i)}\\ \mathbf{n}_{\mathbf{u}_1}^{(i,j+1)} &= \mathbf{V}^{(i,j)}\mathbf{n}_{\mathbf{u}_1}^{(i,j)} - \frac{\tau}{\lambda}\mathbf{P}_1^{(i,j)}\mathbf{n}_{\mathbf{z}}^{(i,j)}\\ \mathbf{n}_{\mathbf{u}_2}^{(i,j+1)} &= \mathbf{V}^{(i,j)}\mathbf{n}_{\mathbf{u}_2}^{(i,j)} - \frac{\tau}{\lambda}\mathbf{P}_2^{(i,j)}\mathbf{n}_{\mathbf{z}}^{(i,j)}\\ \mathbf{n}_{\mathbf{z}}^{(i,j)} &= \mathbf{n}_{\mathbf{x}}^{(i,j)} + \frac{\lambda}{\mu}\mathbf{D}_1^{\mathrm{T}}\mathbf{n}_{\mathbf{u}_1}^{(i,j)} + \frac{\lambda}{\mu}\mathbf{D}_2^{\mathrm{T}}\mathbf{n}_{\mathbf{u}_2}^{(i,j)}\end{cases} \tag{20}$$

where $\mathbf{B} = \mathbf{H}^{\mathrm{T}}\mathbf{H} + \mu\mathbf{I}$, $\mathbf{n}_{\mathbf{x}}^{(i)} = \mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\mathbf{n}_0$ and other noises are similarly defined as $\mathbf{n}_{\mathbf{x}}^{(i)}$.

The flowchart of the algorithm is shown in Fig. 1. Besides from the three remarks mentioned in Sect. 2.3, we notice that $\mathbf{B}^{-1}$, $\mathbf{H}^{\mathrm{T}}$ and $\mathbf{H}_\beta^{-1}$ can be computed by Fourier transform. Thus, all computations of (20) can be performed without any explicit matrix computations.

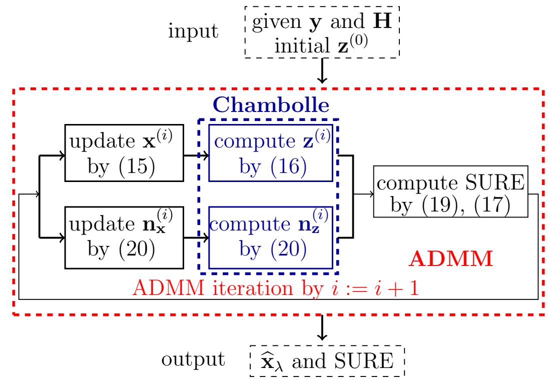

Fig. 1 SURE-MC evaluation for ADMM (Chambolle's algorithm is for obtaining $\mathbf{z}^{(i)}$)

4 Experimental results and discussion

4.1 Experimental setting

The test dataset contains four 8-bit images of size 256×256 or 512×512 displayed in Fig. 2, covering a wide range of natural images.

For both denoising and deconvolution, we always terminate the iterative algorithms, when the relative error of the objective value $\mathcal{L}(\mathbf{x}^{(i)})$ in (2) is below 10^{-5}.

The restoration performance is measured by the peak signal-to-noise ratio (PSNR), defined as (in dB) [20,22]:

$$\mathrm{PSNR} = 10 \times \log_{10}\left(\frac{255^2}{\|\hat{\mathbf{x}} - \mathbf{x}_0\|_2^2/N}\right)$$

We choose $\alpha = 10^{-12}$ in the TV definition of (3) and set the parameter $\tau = 1/4$ in (6), as suggested in [2].

4.2 Image denoising

4.2.1 SURE evaluation for Chambolle's algorithm

First, we need to verify the accuracy of SURE w.r.t. MSE for the Chambolle's iteration. Figure 3 shows the convergence of Chambolle and evolution of SURE, under the noise levels of $\sigma^2 = 1$, 10 and 100, respectively. We can see that: (1)

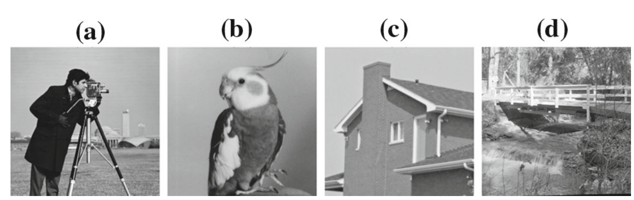

Fig. 2 Original test images: **a** *Cameraman* 256×256; **b** *Coco* 256×256; **c** *House* 256×256; **d** *Bridge* 512×512

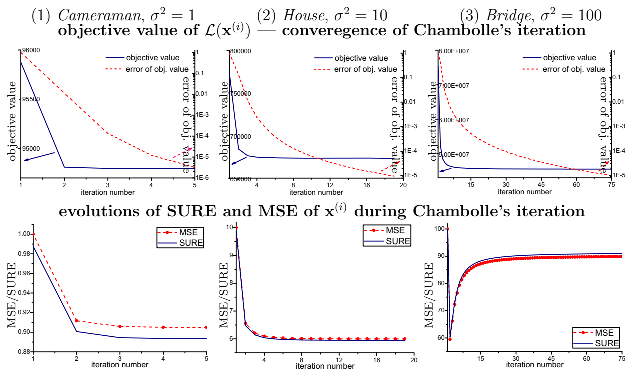

Fig. 3 The convergence of Chambolle and evolution of SURE for fixed λ: (1) $\lambda = 0.1$; (2) $\lambda = 1$; (3) $\lambda = 10$

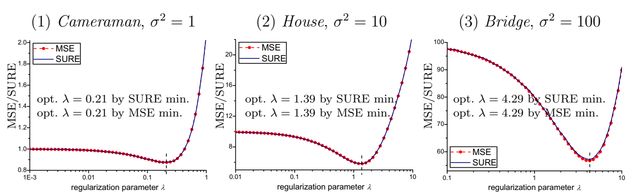

Fig. 4 The global optimization of λ for TV denoising by Chambolle's algorithm

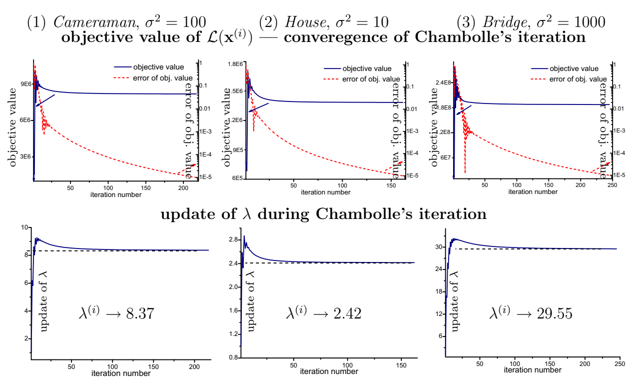

Fig. 5 The convergence of Chambolle with the parameter update by *discrepancy principle*

the objective value keeps decreasing until convergence; (2) SURE is always close to MSE during the iterations.

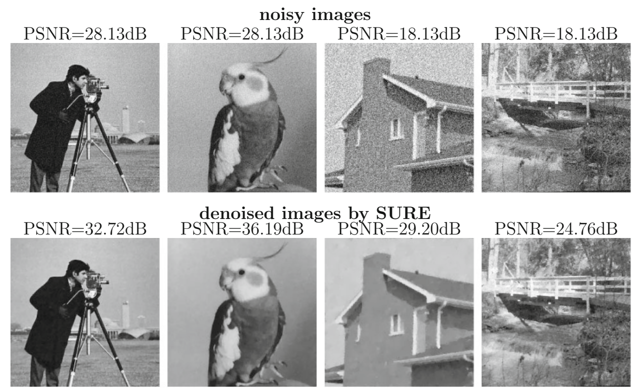

Fig. 6 Examples of visual comparisons: (1) *Cameraman*, $\sigma^2 = 100$; (2) *Coco*, $\sigma^2 = 100$; (3) *House*, $\sigma^2 = 1000$; (4) *Bridge*, $\sigma^2 = 1000$

We repeatedly implement the algorithm for various values of λ and obtain Fig. 4, where the optimal λ is easy to recognize.

4.2.2 Comparisons with discrepancy principle

Discrepancy principle (DP) believes that a good value of λ should satisfy the discrepancy condition $\|\mathbf{y} - \widehat{\mathbf{x}}_\lambda\|_2^2 = N\sigma^2$, according to the observation model (1) [8]. The original Chambolle's algorithm applied DP to update the parameter λ during iterations as [2]:

$$\lambda^{(i+1)} = \sqrt{\frac{N\sigma^2}{\|\mathbf{y} - \mathbf{x}^{(i)}\|_2^2}}\lambda^{(i)}$$

which finally satisfies the discrepancy condition when converged. Figure 5 shows a few examples of the parameter update by DP.

Table 1 shows the complete comparisons between the proposed SURE-based method and *discrepancy principle*. Here, 'DP' denotes *discrepancy principle*. The format of this table is $\frac{\text{est. } \lambda}{\text{PSNR}}$, where the upper value is the selected value of λ by DP/SURE/MSE and the lower one is its resultant denoising PSNR (in dB) using the corresponding λ. The symbol '–'

Table 1 The complete comparisons of selected value of λ and denoising PSNR between DP and SURE

σ^2	1	10	100	1000	1	10	100	1000
Image	*Cameraman*				*House*			
DP [2]	$\frac{0.64}{46.97}$	$\frac{2.32}{39.11}$	$\frac{8.37}{32.05}$	$\frac{31.21}{26.23}$	$\frac{0.67}{47.01}$	$\frac{2.42}{39.54}$	$\frac{9.97}{33.89}$	–
SURE	$\frac{0.21}{48.72}$	$\frac{1.15}{40.22}$	$\frac{5.70}{32.74}$	$\frac{23.36}{26.61}$	$\frac{0.19}{48.74}$	$\frac{1.26}{40.50}$	$\frac{6.87}{34.35}$	$\frac{29.76}{29.20}$
MSE	$\frac{0.21}{48.72}$	$\frac{1.26}{40.23}$	$\frac{5.70}{32.74}$	$\frac{23.36}{26.61}$	$\frac{0.21}{48.75}$	$\frac{1.39}{40.50}$	$\frac{6.87}{34.35}$	$\frac{29.76}{29.20}$
Image	*Coco*				*Bridge*			
DP [2]	$\frac{0.73}{47.71}$	$\frac{3.25}{41.25}$	$\frac{13.78}{35.15}$	–	$\frac{0.64}{45.83}$	$\frac{2.19}{36.95}$	$\frac{7.83}{29.52}$	$\frac{29.55}{24.27}$
SURE	$\frac{0.28}{49.13}$	$\frac{2.02}{42.02}$	$\frac{7.54}{36.19}$	$\frac{29.76}{30.62}$	$\frac{0.11}{48.25}$	$\frac{0.72}{38.76}$	$\frac{4.29}{30.60}$	$\frac{20.34}{24.76}$
MSE	$\frac{0.28}{49.13}$	$\frac{1.84}{42.04}$	$\frac{8.29}{36.21}$	$\frac{29.76}{30.62}$	$\frac{0.10}{48.25}$	$\frac{0.72}{38.76}$	$\frac{4.29}{30.60}$	$\frac{20.23}{24.76}$

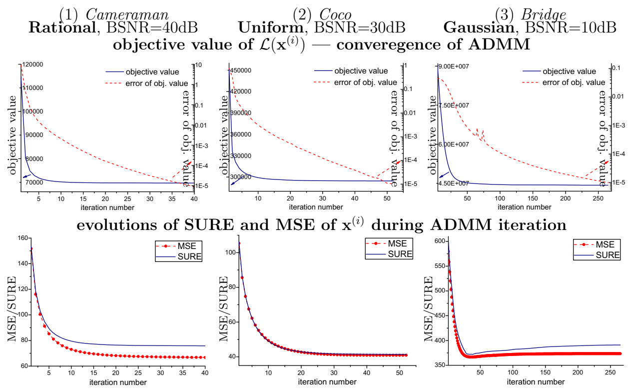

Fig. 7 The convergence of ADMM and evolution of SURE for fixed λ: (1) $\lambda = 0.1$, (2) $\lambda = 1$, (3) $\lambda = 10$

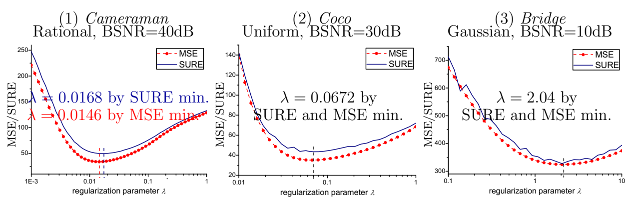

Fig. 8 The global optimization of λ for TV deconvolution by ADMM

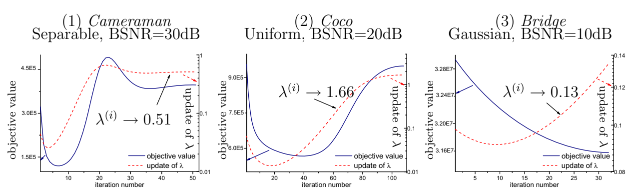

Fig. 9 The parameter update for TV deconvolution by DP [10]

indicates that the method fails to find an optimal λ for this case. The MSE is not accessible in practice and thus shown in italics. It is the comparison benchmark, indicating the best PSNR performance we can achieve. We can see that compared to the MSE minimization, the PSNR by DP is worse than optimal PSNR by 1 dB in average, whereas the SURE minimization yields negligible PSNR loss (within 0.02 dB). Figure 6 shows a number of visual examples.

4.3 Image deconvolution

4.3.1 Experimental setting

For deconvolution problem, we consider the following benchmark convolution kernels commonly used in [10,22]:

- *Rational filter* $h(i, j) = C \cdot (1 + i^2 + j^2)^{-1}$ for $i, j = -7, \ldots, 7$;
- *Separable filter* 5×5 filter with weights $[1, 4, 6, 4, 1]/16$ along both horizontal and vertical directions;
- 9×9 uniform blur;
- *Gaussian kernel* $h(i, j) = C \cdot \exp\big(-\frac{i^2+j^2}{2s^2}\big)$ with $s = 2.0$.

where C is a normalization factor, s.t. $\sum_{i,j} h(i, j) = 1.0$. The blurred images are subsequently contaminated by i.i.d Gaussian noise with various variance σ^2, corresponding to blur signal-to-noise ratio (BSNR) being 40, 30, 20 and 10 dB, respectively, where the BSNR is defined as (in dB) [20]:

Table 2 The selected λ and corresponding PSNR by DP and SURE (rational and separable filtering)

Blur kernel	Rational filtering				Separable filtering			
BSNR (in dB)	40	30	20	10	40	30	20	10
Cameraman								
DP [10]	$\frac{0.05}{31.35}$	$\frac{0.20}{27.84}$	$\frac{0.54}{24.64}$	$\frac{0.18}{19.81}$	$\frac{0.12}{30.55}$	$\frac{0.51}{28.81}$	$\frac{4.90}{26.85}$	$\frac{2.15}{23.69}$
SURE	$\frac{0.01}{32.75}$	$\frac{0.05}{28.26}$	$\frac{0.28}{24.99}$	$\frac{3.59}{22.58}$	$\frac{0.01}{31.44}$	$\frac{0.10}{29.44}$	$\frac{0.60}{27.10}$	$\frac{3.60}{24.31}$
MSE	$\frac{0.02}{32.81}$	$\frac{0.06}{28.31}$	$\frac{0.36}{25.05}$	$\frac{4.64}{22.59}$	$\frac{0.02}{31.49}$	$\frac{0.10}{29.44}$	$\frac{0.77}{27.13}$	$\frac{3.60}{24.31}$
Coco								
DP [10]	$\frac{0.21}{38.59}$	$\frac{0.48}{35.76}$	$\frac{1.64}{31.62}$	$\frac{4.83}{28.80}$	$\frac{0.96}{40.37}$	$\frac{1.42}{38.03}$	$\frac{3.70}{33.36}$	$\frac{6.82}{31.78}$
SURE	$\frac{0.04}{39.92}$	$\frac{0.08}{36.13}$	$\frac{0.77}{32.50}$	$\frac{5.99}{28.73}$	$\frac{0.02}{41.30}$	$\frac{0.10}{38.58}$	$\frac{0.61}{34.98}$	$\frac{7.74}{32.06}$
MSE	$\frac{0.03}{39.95}$	$\frac{0.11}{36.35}$	$\frac{0.60}{32.55}$	$\frac{3.59}{28.89}$	$\frac{0.04}{41.35}$	$\frac{0.12}{38.70}$	$\frac{0.72}{35.23}$	$\frac{7.74}{32.06}$
House								
DP [10]	$\frac{0.08}{35.83}$	$\frac{0.29}{33.39}$	$\frac{1.03}{29.81}$	$\frac{2.77}{27.03}$	$\frac{0.36}{36.33}$	$\frac{0.75}{34.47}$	$\frac{2.57}{32.43}$	$\frac{4.74}{29.01}$
SURE	$\frac{0.02}{37.05}$	$\frac{0.11}{33.79}$	$\frac{0.60}{30.64}$	$\frac{3.59}{27.00}$	$\frac{0.01}{37.55}$	$\frac{0.07}{35.42}$	$\frac{1.29}{32.91}$	$\frac{12.92}{29.80}$
MSE	$\frac{0.02}{37.14}$	$\frac{0.08}{33.81}$	$\frac{0.60}{30.64}$	$\frac{2.78}{27.03}$	$\frac{0.01}{37.56}$	$\frac{0.10}{35.55}$	$\frac{1.29}{32.91}$	$\frac{11.57}{30.00}$
Bridge								
DP [10]	$\frac{0.03}{30.00}$	$\frac{0.15}{26.97}$	$\frac{0.34}{24.75}$	$\frac{0.18}{20.61}$	$\frac{0.06}{29.42}$	$\frac{0.34}{28.00}$	$\frac{1.09}{26.38}$	$\frac{9.24}{24.33}$
SURE	$\frac{0.01}{31.05}$	$\frac{0.05}{27.57}$	$\frac{0.17}{24.90}$	$\frac{1.67}{22.54}$	$\frac{0.007}{30.16}$	$\frac{0.06}{28.58}$	$\frac{0.36}{26.71}$	$\frac{3.59}{24.49}$
MSE	$\frac{0.008}{31.06}$	$\frac{0.05}{27.57}$	$\frac{0.22}{24.97}$	$\frac{2.78}{22.72}$	$\frac{0.007}{30.16}$	$\frac{0.06}{28.58}$	$\frac{0.46}{26.74}$	$\frac{3.59}{24.49}$

Table 3 The selected λ and corresponding PSNR by DP and SURE (uniform and Gaussian blurs)

Blur kernel	9 × 9 uniform				Gaussian kernel			
BSNR (in dB)	40	30	20	10	40	30	20	10
Cameraman								
DP [10]	0.03 / 28.04	0.002 / 24.44	0.007 / 22.46	0.01 / 19.22	0.05 / 26.06	0.007 / 24.67	0.02 / 23.32	0.12 / 19.89
SURE	0.01 / 28.42	0.08 / 25.44	0.11 / 22.98	7.74 / 21.29	0.01 / 26.08	0.11 / 25.02	0.28 / 23.86	2.78 / 22.57
MSE	*0.01 / 28.43*	*0.08 / 25.44*	*0.17 / 23.02*	*2.78 / 21.43*	*0.01 / 26.09*	*0.08 / 25.06*	*0.46 / 23.87*	*4.64 / 22.67*
Coco								
DP [10]	0.10 / 35.47	0.34 / 32.94	1.66 / 28.35	0.05 / 23.62	0.12 / 35.26	0.47 / 33.01	0.07 / 29.96	0.14 / 23.94
SURE	0.02 / 36.54	0.08 / 33.54	0.60 / 30.02	2.78 / 27.46	0.01 / 35.36	0.06 / 33.60	0.60 / 31.67	5.99 / 29.29
MSE	*0.01 / 36.61*	*0.10 / 33.59*	*0.60 / 30.02*	*2.15 / 27.46*	*0.03 / 35.63*	*0.10 / 33.74*	*0.77 / 31.68*	*4.64 / 29.34*
House								
DP [10]	0.04 / 34.66	0.006 / 29.52	0.02 / 26.74	0.02 / 22.75	0.08 / 32.44	0.34 / 30.54	0.03 / 28.23	0.12 / 23.45
SURE	0.008 / 35.04	0.08 / 32.05	0.77 / 28.41	2.15 / 25.41	0.01 / 32.59	0.08 / 31.27	0.77 / 29.62	2.15 / 27.30
MSE	*0.01 / 35.10*	*0.08 / 32.05*	*0.46 / 28.52*	*1.67 / 25.45*	*0.03 / 32.82*	*0.10 / 31.34*	*0.77 / 29.62*	*2.15 / 27.42*
Bridge								
DP [10]	0.02 / 26.80	0.32 / 25.07	0.007 / 22.99	0.01 / 20.01	0.03 / 25.66	0.008 / 24.85	0.02 / 23.70	0.13 / 19.36
SURE	0.006 / 27.25	0.03 / 25.25	0.17 / 23.40	1.29 / 21.76	0.002 / 25.99	0.02 / 25.09	0.17 / 24.03	1.67 / 22.70
MSE	*0.005 / 27.26*	*0.03 / 25.25*	*0.13 / 23.42*	*1.67 / 21.80*	*0.002 / 25.99*	*0.01 / 25.10*	*0.17 / 24.03*	*2.78 / 22.81*

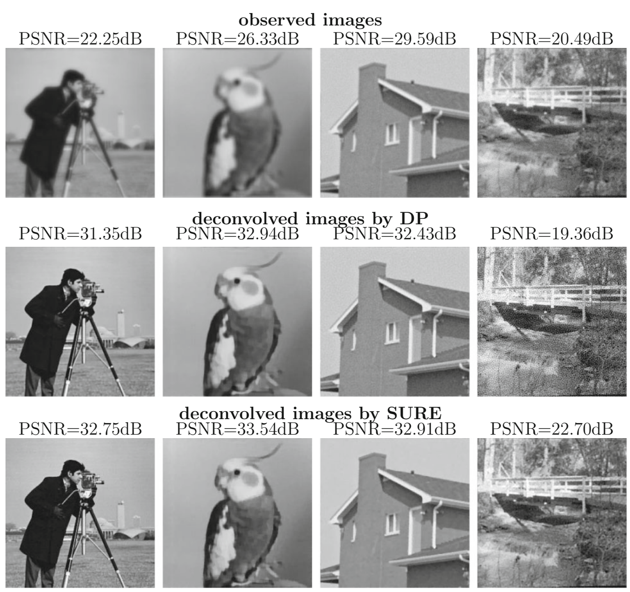

Fig. 10 Examples of visual comparisons: (1) *Cameraman*, rational, BSNR = 40 dB; (2) *Coco*, uniform, BSNR = 30 dB; (3) *House*, separable, BSNR = 20 dB; (4) *Bridge*, Gaussian, BSNR = 10 dB

$$\text{BSNR} = 10 \times \log_{10}\left(\frac{\|\mathbf{Hx}_0 - \text{mean}(\mathbf{Hx}_0)\|_2^2}{N\sigma^2}\right)$$

The deconvolution performance is also measured by PSNR. In addition, we always choose the parameters $\mu = 0.1\sigma^2$ and $\beta = 10^{-5}$.

4.3.2 SURE evaluation for ADMM

First, we implement the procedure shown in Fig. 1, i.e. apply ADMM to solve (2) with fixed λ, and evaluate the SURE. Figure 7 shows the convergence of ADMM and the evolution of SURE. Figure 8 shows the global optimization of λ.

4.3.3 Comparisons with discrepancy principle

Similar to [2], the DP has also been adopted to the deconvolution problem, for example, the i-LET method updated the parameter λ by the following rule:

$$\lambda^{(i+1)} = \frac{N\sigma^2}{\|\mathbf{y} - \mathbf{Hx}^{(i)}\|_2^2}\lambda^{(i)}$$

in ℓ_1-based sparse deconvolution [10]. We use this equation to update λ in TV deconvolution and obtain Fig. 9.

Tables 2 and 3 show the comparisons between the proposed SURE and DP. The format is the same as Table 1. We can see that compared to the best PSNR results obtained by MSE minimization, the PSNR by DP is worse than optimal PSNR by 1–3 dB in average, whereas the SURE minimization yields negligible PSNR loss (at most 0.20 dB). Figure 10 shows a number of visual examples, where we can see the better visual quality by SURE than by DP.[4]

[4] It is better to recognize the visual difference by zoom-in on larger screen.

5 Conclusions

In this paper, we presented a SURE-based method for automatically tuning regularization parameter for TV-based recovery. In particular, we proposed a recursive evaluation and Monte Carlo simulation for the practical computation. Numerical results showed the superior performance of SURE to other criteria for parameter selection, e.g. discrepancy principle.

This proposed method, in principle, can be extended to more complicated (possibly non-convex) regularizers [13,18,23]. Future work will also deal with the SURE-based multiple parameter selection and faster optimization of SURE, to accelerate the global search used here.

References

1. Blu, T., Luisier, F.: The SURE-LET approach to image denoising. IEEE Trans. Image Process. **16**(11), 2778–2786 (2007)
2. Chambolle, A.: An algorithm for total variation minimization and applications. J. Math. Imaging Vis. **20**, 89–97 (2004)
3. Chen, D.: Inext alternating direction method based on Newton descent algorithm with application to Poisson image deblurring. Signal Image Video Process. **11**, 89–96 (2017)
4. Dell'Acqua, P.: ν acceleration of statistical iterative methods for image restoration. Signal Image Video Process. **10**, 927–934 (2016)
5. Golub, G., Heath, M., Wahba, G.: Generalized cross-validation as a method for choosing a good ridge parameter. Technometrics **21**(2), 215–223 (1979)
6. Guo, C., Li, Q., Wei, C., Tan, J., Liu, S., Liu, Z.: Axial multi-image phase retrieval under tilt illumination. Sci. Rep. **7**, 7562 (2017)
7. Hansen, P.C.: Analysis of discrete ill-posed problems by means of the L-curve. SIAM Rev. **34**(4), 561–580 (1992)
8. Morozov, V.: Methods for Solving Incorrectly Posed Problems. Springer, New York (1984)
9. Osher, S., Burger, M., Goldfarb, D., Xu, J., Yin, W.: An iterative regularization method for total variation-based image restoration. SIAM J. Multiscale Model. Simul. **4**(2), 460–489 (2005)
10. Pan, H., Blu, T.: An iterative linear expansion of thresholds for ℓ_1-based image restoration. IEEE Trans. Image Process. **22**(9), 3715–3728 (2013)
11. Ramirez, C., Argaez, M.: An ℓ_1 minimization algorithm for non-smooth regularization in image processing. Signal Image Video Process. **9**, 373–386 (2015)
12. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D **60**, 259–268 (1992)
13. Selesnick, I., Parekh, A., Bayram, I.: Convex 1-D total variation denoising with non-convex regularization. IEEE Signal Process. Lett. **22**(2), 141–144 (2015)
14. Shen, C., Bao, X., Tan, J., Liu, S., Liu, Z.: Two noise-robust axial scanning multi-image phase retrieval algorithms based on Pauta criterion and smoothness constraint. Opt. Express **25**(14), 16235–16249 (2017)
15. Siadat, M., Aghazadeh, N., Öktem, O.: Reordering for improving global Arnoldi–Tikhonov method in image restoration problems. Signal Image Video Process. **12**, 497–504 (2018)
16. Stein, C.M.: Estimation of the mean of a multivariate normal distribution. In: Stein, C.M. (ed.) The Annals of Statistics, vol. 9, No. 6, pp. 1135–1151 (1981)
17. Tao, M., Yang, J.: Alternating Direction Algorithms for Total Variation Deconvolution in Image Reconstruction. Optimization Online, TR0918, Department of Mathmatics, Nanjing University (2009)
18. Vonesch, C., Ramani, S., Unser, M.: Recursive risk estimation for non-linear image deconvolution with a wavelet-domain sparsity constraint. In: IEEE International conference on Image processing, pp. 665–668 (2008)
19. 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)
20. Xue, F., Blu, T.: A novel SURE-based criterion for parametric PSF estimation. IEEE Trans. Image Process. **24**(2), 595–607 (2015)
21. Xue, F., Du, R., Liu, J.: A recursive predictive risk estimate for proximal algorithms. In: Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing, Shanghai, China, March 20–25, pp. 4498–4502 (2016)
22. Xue, F., Luisier, F., Blu, T.: Multi-Wiener SURE-LET deconvolution. IEEE Trans. Image Process. **22**(5), 1954–1968 (2013)
23. Xue, F., Yagola, A.G., Liu, J., Meng, G.: Recursive SURE for iterative reweighted least square algorithms. Inverse Probl. Sci. Eng. **24**(4), 625–646 (2016)

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