OPTIMIZATION OF COMPOUND REGULARIZATION PARAMETERS BASED ON STEIN'S UNBIASED RISK ESTIMATE

Feng Xue[a], Hanjie Pan[b], Runhui Wu[a], Xin Liu[a] and Jiaqi Liu[a]

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

[b]Audiovisual Communications Laboratory, School of Computer and Communication Sciences, Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland

ABSTRACT

Recently, the type of compound regularizers has become a popular choice for signal reconstruction. The estimation quality is generally sensitive to the values of multiple regularization parameters. In this work, based on BDF algorithm, we develop a data-driven optimization scheme based on minimization of Stein's unbiased risk estimate (SURE)—statistically equivalent to mean squared error (MSE). We propose a recursive evaluation of SURE to monitor the MSE during BDF iteration; the optimal values of the multiple parameters are then identified by the minimum SURE. Monte-Carlo simulation is applied to compute SURE for large-scale data. We exemplify the proposed method with image deconvolution. Numerical experiments show that the proposed method leads to highly accurate estimates of regularization parameters and nearly optimal restoration performance.

Index Terms— Stein's unbiased risk estimate (SURE), compound regularizers, regularization parameter, BDF algorithm, signal deconvolution

1. INTRODUCTION

Consider the standard linear inverse problem: find a good estimate of $\mathbf{x}_0 \in \mathbb{R}^N$ from the following observation model [1, 2]:

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

where $\mathbf{y} \in \mathbb{R}^M$ is the observed noisy data, $\mathbf{A} \in \mathbb{R}^{M\times N}$ is an observation matrix, $\epsilon \in \mathbb{R}^M$ is an additive Gaussian white noise with known variance $\sigma^2 > 0$.

Regularization has been a standard technique for solving the inverse problem. Recently, people considered the regularizer as a linear combination of "simple" regularizers, i.e., the objective function is [3–5]:

$$\min_{\mathbf{x}} \underbrace{\frac{1}{2}\left\|\mathbf{A}\mathbf{x}-\mathbf{y}\right\|_2^2 + \lambda_1 \cdot \mathcal{J}_1(\mathbf{D}_1\mathbf{x}) + \lambda_2 \cdot \mathcal{J}_2(\mathbf{D}_2\mathbf{x})}_{\mathcal{L}(\mathbf{x})} \tag{2}$$

where both $\mathcal{J}_1$ and $\mathcal{J}_2$ are simple regularizers, λ_1 and λ_2 are their respective regularization parameters.

This type of hybrid regularizers stems mainly from the following observation: it may be desired to encourage the solution to exhibit characteristics that are not easily enforced/described by a single regularizer. In this paper, we choose BDF algorithm to solve (2) [6], since it provides a basic scheme for tackling the multiple regularizers, and that is easy to extend for other types of regularizer. The 'BDF' stands for the last names 'Bioucas-Dias' and 'Figueiredo' of both authors of [6].

For a pleasant reconstruction quality, it is essential to select the proper values of multiple regularization parameters, to keep a good balance between data fidelity and compound regularizers. The choice of λ_1 and λ_2 is generally a difficult problem. There are two well-known general approaches capable of selecting the parameters in non-linear inverse problems: maximum likelihood and cross validation [7]. However, both methods suffer from a problem of computational complexity.

In this paper, we quantify the reconstruction performance by the mean squared error (MSE) [1, 8]:

$$\mathrm{MSE} = \frac{1}{N}\mathbb{E}\left\{\left\|\widehat{\mathbf{x}}-\mathbf{x}_0\right\|_2^2\right\} \tag{3}$$

and attempt to select the values of λ_1 and λ_2, such that the corresponding solution $\widehat{\mathbf{x}}$ achieves minimum MSE. Notice that MSE (3) 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 (if $\mathbf{A}$ is full-rank matrix) [9, 10]:

$$\mathrm{SURE} = \frac{1}{N}\left(\left\|\widehat{\mathbf{x}}\right\|_2^2 - 2\mathbf{y}^{\mathrm{T}}\mathbf{A}(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\widehat{\mathbf{x}} + 2\sigma^2\mathrm{Tr}\left(\mathbf{A}(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{J}_{\mathbf{y}}(\widehat{\mathbf{x}})\right) + \left\|\mathbf{x}_0\right\|_2^2\right) \tag{4}$$

since it depends on the observed data $\mathbf{y}$ only[1]. Here, $\mathbf{J}_{\mathbf{y}}(\widehat{\mathbf{x}}) \in \mathbb{R}^{N\times N}$ is a Jacobian matrix defined as [11]:

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

Recently, SURE has become a popular criterion for optimization, in the context of non-linear denoising and deconvolution [1, 8], and ℓ_1-based sparse reconstruction [11–13].

[1]The last term of (4)—$\|\mathbf{x}_0\|_2^2$—is constant irrelevant to the optimization of $\widehat{\mathbf{x}}$.

However, there are very few literature on the application of SURE to the compound regularizers.

This paper is to optimize the regularization parameters λ_1 and λ_2 for (2), based on minimization of SURE (4). Our main contribution is to develop a recursive evaluation of SURE for BDF algorithm, which finally provides a reliable estimate of the MSE for the non-linear reconstruction. The optimal λ_1 and λ_2 can then be identified by exhaustive search for minimum SURE. In addition, the Monte-Carlo method is applied for practical computation of large-scale data.

2. AN APPLICATION OF BDF ALGORITHM TO TV+ℓ_1 MINIMIZATION

2.1. Basic scheme of BDF algorithm

The problem (2) is equivalent to the following:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \tfrac{1}{2}\left\|\mathbf{A}\mathbf{x}-\mathbf{y}\right\|_2^2+\lambda_1\cdot\mathcal{J}_1(\mathbf{z}_1)+\lambda_2\cdot\mathcal{J}_2(\mathbf{z}_2) \\ \text{subject to} \quad & \left\|\mathbf{z}_1-\mathbf{D}_1\mathbf{x}\right\|_2^2=0;\quad \left\|\mathbf{z}_2-\mathbf{D}_2\mathbf{x}\right\|_2^2=0 \end{aligned}$$

which, by Lagrangian, is equivalent to:

$$\min_{\mathbf{x}}\ \frac{1}{2}\left\|\mathbf{A}\mathbf{x}-\mathbf{y}\right\|_2^2+\lambda_1\mathcal{J}_1(\mathbf{z}_1)+\lambda_2\mathcal{J}_2(\mathbf{z}_2)+\frac{\mu_1}{2}\left\|\mathbf{z}_1-\mathbf{D}_1\mathbf{x}\right\|_2^2+\frac{\mu_2}{2}\left\|\mathbf{z}_2-\mathbf{D}_2\mathbf{x}\right\|_2^2$$

where μ_1 and μ_2 are the augmented Lagrangian penalty parameters.

To minimize this functional w.r.t. $\mathbf{x}$, $\mathbf{z}_1$ and $\mathbf{z}_2$, BDF algorithm is to alternatively minimize w.r.t. these variables:

$$\begin{cases} \mathbf{x}^{(i+1)}=\arg\min_{\mathbf{x}}\ \left\|\mathbf{A}\mathbf{x}-\mathbf{y}\right\|_2^2+\sum_{t=1}^{2}\mu_t\left\|\mathbf{D}_t\mathbf{x}-\mathbf{z}_t^{(i)}\right\|_2^2 \\ \mathbf{z}_1^{(i+1)}=\arg\min_{\mathbf{z}_1}\ \tfrac{1}{2}\left\|\mathbf{z}_1-\mathbf{D}_1\mathbf{x}^{(i+1)}\right\|_2^2+\frac{\lambda_1}{\mu_1}\cdot\mathcal{J}_1(\mathbf{z}_1) \\ \mathbf{z}_2^{(i+1)}=\arg\min_{\mathbf{z}_2}\ \tfrac{1}{2}\left\|\mathbf{z}_2-\mathbf{D}_2\mathbf{x}^{(i+1)}\right\|_2^2+\frac{\lambda_2}{\mu_2}\cdot\mathcal{J}_2(\mathbf{z}_2) \end{cases}$$

which can be efficiently expressed and computed by Moreau's proximal operator for a number of typical regularizers of interest [14, 15].

2.2. Exemplification with wavelet-ℓ_1 and TV regularizers

To exemplify the iterative algorithm, we consider signal deconvolution problem, with both wavelet-ℓ_1 and TV regularizers, i.e., $\mathcal{J}_1(\mathbf{D}_1\mathbf{x})=\|\mathbf{D}_1\mathbf{x}\|_1$ and $\mathcal{J}_2(\mathbf{x})=\mathrm{TV}(\mathbf{x})$, where $\mathbf{D}_1$ denotes wavelet decomposition. Thus, the problem becomes:

$$\min_{\mathbf{x}}\ \frac{1}{2}\left\|\mathbf{A}\mathbf{x}-\mathbf{y}\right\|_2^2+\lambda_1\left\|\mathbf{z}_1\right\|_1+\lambda_2\mathrm{TV}(\mathbf{z}_2)+\frac{\mu_1}{2}\left\|\mathbf{D}_1\mathbf{x}-\mathbf{z}_1\right\|_2^2+\frac{\mu_2}{2}\left\|\mathbf{x}-\mathbf{z}_2\right\|_2^2$$

which yields the following iteration:

$$\begin{cases} \mathbf{x}^{(i)}=\mathbf{B}^{-1}\left(\mathbf{A}^{\mathrm{T}}\mathbf{y}+\mu_1\mathbf{D}_1^{\mathrm{T}}\mathbf{z}_1^{(i-1)}+\mu_2\mathbf{z}_2^{(i-1)}\right) \\ \mathbf{z}_1^{(i)}=\mathcal{T}_{\lambda_1/\mu_1}\left(\mathbf{D}_1\mathbf{x}^{(i)}\right) \\ \mathbf{z}_2^{(i)}=\arg\min_{\mathbf{z}_2}\ \tfrac{1}{2}\left\|\mathbf{z}_2-\mathbf{x}^{(i)}\right\|_2^2+\frac{\lambda_2}{\mu_2}\cdot\mathrm{TV}(\mathbf{z}_2) \end{cases} \tag{5}$$

where $\mathbf{B}=\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mu_1\mathbf{D}_1^{\mathrm{T}}\mathbf{D}_1+\mu_2\mathbf{I}$, $\mathcal{T}_T(\cdot)$ denotes the pointwise soft-thresholding with threshold T [11]. $\mathbf{z}_2^{(i)}$ can be efficiently solved by Chambolle's algorithm [16].

For 2-D case, we consider the TV definition as $\mathrm{TV}(\mathbf{x})=\sum_{n=1}^{N}\sqrt{|(\mathbf{D}_2^{(1)}\mathbf{x})_n|^2+|(\mathbf{D}_2^{(2)}\mathbf{x})_n|^2+\alpha}$, where $\mathbf{D}_2^{(1)}$ and $\mathbf{D}_2^{(2)}$ denote the first-order differences along horizontal and vertical directions, α is a very small number (e.g. 10^{-12}) [17]. Such an approximation simplifies numerical computations due to the differentiability of TV, and may help to avoid the staircasing effect in some cases [17].

Chambolle's algorithm for solving $\mathbf{z}_2^{(i)}$ of (5) can be expressed in matrix language as (iterate by j):

$$\mathbf{u}^{(i,j+1)}=\overline{\mathbf{V}}^{(i,j)}\Big(\mathbf{u}^{(i,j)}-\frac{\tau\mu_2}{\lambda_2}\mathbf{D}_2\underbrace{\Big(\mathbf{x}^{(i)}+\frac{\lambda_2}{\mu_2}\mathbf{D}_2^{\mathrm{T}}\mathbf{u}^{(i,j)}\Big)}_{\mathbf{z}_2^{(i,j)}}\Big) \tag{6}$$

where τ is a step-size, $\mathbf{D}_2$ and diagonal matrix $\overline{\mathbf{V}}^{(i,j)}$ are

$$\mathbf{D}_2=\begin{bmatrix}\mathbf{D}_2^{(1)}\\ \mathbf{D}_2^{(2)}\end{bmatrix}\in\mathbb{R}^{2N\times N};\quad \overline{\mathbf{V}}^{(i,j)}=\begin{bmatrix}\mathbf{V}^{(i,j)} & \mathbf{0}\\ \mathbf{0} & \mathbf{V}^{(i,j)}\end{bmatrix}\in\mathbb{R}^{2N\times 2N}$$

with diagonal $\mathbf{V}^{(i,j)}\in\mathbb{R}^{N\times N}$ given by:

$$\mathbf{V}_{n,n}^{(i,j)}=\left(1+\frac{\tau\mu_2}{\lambda_2}\sqrt{\left((\mathbf{D}_2^{(1)}\mathbf{z}_2^{(i,j)})_n\right)^2+\left((\mathbf{D}_2^{(2)}\mathbf{z}_2^{(i,j)})_n\right)^2+\alpha}\right)^{-1}$$

Finally, $\mathbf{z}_2^{(i)}$ is obtained by the convergence of Chambolle's iteration (6): $\mathbf{z}_2^{(i)}=\mathbf{z}_2^{(i,\infty)}$ at $j=\infty$ when converged.

3. RECURSIVE EVALUATION OF SURE FOR BDF ALGORITHM

3.1. Recursive evaluation of SURE

From (4), the SURE for the i-th iterate is[2]:

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

The computation of SURE requires to compute $\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})$, which can be evaluated in a recursive manner, as shown later.

From (5), by the basic property of Jacobian, we have:

$$\begin{cases} \mathbf{J}_{\mathbf{y}}\left(\mathbf{x}^{(i)}\right)=\mathbf{B}^{-1}\left[\mathbf{A}^{\mathrm{T}}+\mu_1\mathbf{D}_1^{\mathrm{T}}\mathbf{J}_{\mathbf{y}}\left(\mathbf{z}_1^{(i-1)}\right)+\mu_2\mathbf{J}_{\mathbf{y}}\left(\mathbf{z}_2^{(i-1)}\right)\right] \\ \mathbf{J}_{\mathbf{y}}\left(\mathbf{z}_1^{(i)}\right)=\mathbf{P}^{(i)}\mathbf{D}_1\mathbf{J}_{\mathbf{y}}\left(\mathbf{x}^{(i)}\right) \end{cases} \tag{8}$$

where $\mathbf{P}^{(i)}$ is a diagonal matrix with diagonal element:

$$\left[\mathbf{P}^{(i)}\right]_{n,n}=\begin{cases}1, & \text{if } \left|(\mathbf{D}_1\mathbf{x}^{(i)})_n\right|\geq\lambda_1/\mu_1\\ 0, & \text{otherwise}\end{cases}$$

The recursion of $\mathbf{J}_{\mathbf{y}}\left(\mathbf{z}_2^{(i)}\right)$ has to be obtained by Chambolle's algorithm. We consider 2-D case only.

[2] In the remainder of this paper, the last constant term—$\|\mathbf{x}_0\|_2^2$—is ignored for brevity.

3.2. Recursion of $\mathbf{J}_{\mathbf{y}}\left(\mathbf{z}_2^{(i)}\right)$ for Chambolle's algorithm

For 2-D case, we express $\mathbf{u}^{(i,j+1)}$ of (6) as $\mathbf{u}^{(i,j+1)} = \begin{bmatrix}\mathbf{u}_1^{(i,j+1)}\\ \mathbf{u}_2^{(i,j+1)}\end{bmatrix}$, where the first part $\mathbf{u}_1^{(i,j+1)}$ is:

$$\mathbf{u}_1^{(i,j+1)} = \mathbf{V}^{(i,j)}\underbrace{\left(\mathbf{u}_1^{(i,j)} - \frac{\tau\mu_2}{\lambda_2}\mathbf{D}_2^{(1)}\mathbf{z}_2^{(i,j)}\right)}_{\mathbf{w}_1^{(i,j)}}$$

After derivations, we obtain[3]:

$$\mathbf{J}_{\mathbf{y}}\left(\mathbf{u}^{(i,j+1)}\right) = \begin{bmatrix}\mathbf{V}^{(i,j)} & \mathbf{0}\\ \mathbf{0} & \mathbf{V}^{(i,j)}\end{bmatrix}\mathbf{J}_{\mathbf{y}}\left(\mathbf{u}^{(i,j)}\right) - \frac{\tau\mu_2}{\lambda_2}\cdot \begin{bmatrix}\mathbf{W}_1^{(i,j)}\mathbf{C}_1^{(i,j)} + \mathbf{V}^{(i,j)} & \mathbf{W}_1^{(i,j)}\mathbf{C}_2^{(i,j)}\\ \mathbf{W}_2^{(i,j)}\mathbf{C}_1^{(i,j)} & \mathbf{W}_2^{(i,j)}\mathbf{C}_2^{(i,j)} + \mathbf{V}^{(i,j)}\end{bmatrix}\mathbf{D}_2\mathbf{J}_{\mathbf{y}}\left(\mathbf{z}_2^{(i,j)}\right) \quad (9)$$

where $\mathbf{C}_1^{(i,j)}$ and $\mathbf{C}_2^{(i,j)}$ are diagonal:

$$\left[\mathbf{C}_1^{(i,j)}\right]_{n,n} = \frac{a_m\cdot\left(\mathbf{V}_{m,m}^{(i)}\right)^2}{\sqrt{a_m^2+b_m^2+\alpha}};\quad \left[\mathbf{C}_2^{(i,j)}\right]_{n,n} = \frac{b_m\cdot\left(\mathbf{V}_{m,m}^{(i)}\right)^2}{\sqrt{a_m^2+b_m^2+\alpha}}$$

with $\mathbf{a} = \mathbf{D}_2^{(1)}\mathbf{z}_2^{(i,j)}$ and $\mathbf{b} = \mathbf{D}_2^{(2)}\mathbf{z}_2^{(i,j)}$. $\mathbf{W}_1^{(i,j)}$ and $\mathbf{W}_2^{(i,j)}$ are diagonal: $[\mathbf{W}_1^{(i,j)}]_{n,n} = [\mathbf{w}_1^{i,j}]_n$ and $[\mathbf{W}_2^{(i,j)}]_{n,n} = [\mathbf{w}_2^{i,j}]_n$.

Note that $\mathbf{z}_2^{(i,j)} = \mathbf{x}^{(i)} + \frac{\lambda_2}{\mu_2}\mathbf{D}^{\mathrm{T}}\mathbf{u}^{(i,j)}$, we have:

$$\mathbf{J}_{\mathbf{y}}\left(\mathbf{z}_2^{(i,j)}\right) = \mathbf{J}_{\mathbf{y}}\left(\mathbf{x}^{(i)}\right) + \frac{\lambda_2}{\mu_2}\mathbf{D}^{\mathrm{T}}\mathbf{J}_{\mathbf{y}}\left(\mathbf{u}^{(i,j)}\right) \quad (10)$$

3.3. Summary of BDF algorithm with SURE evaluation

Finally, we summarize the proposed algorithm as **Algorithm 1**, which enables us to solve (2) with a prescribed values of λ_1 and λ_2, and simultaneously evaluate the SURE during the BDF iterations.

Algorithm 1: SURE evaluation for BDF algorithm

for $i = 1,2,\ldots$ *(BDF iteration)* **do**
- **1** update $\mathbf{x}^{(i)}$, $\mathbf{z}_1^{(i)}$ and $\mathbf{z}_2^{(i)}$ by (5) and (6);
- **2** update $\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})$, $\mathbf{J}_{\mathbf{y}}(\mathbf{z}_1^{(i)})$ and $\mathbf{J}_{\mathbf{y}}(\mathbf{z}_2^{(i)})$ by (8), (9) and (10);
- **3** compute SURE of i-th iterate by (7);

end

3.4. Monte-Carlo for practical computation

For 2-D case, due to the limited computational resources (e.g. RAM), it is impractical to store and compute the huge matrices $\mathbf{A}$, $\mathbf{D}_1$ and Jacobians. Monte-Carlo (MC) simulation provides an alternative way to compute the trace by the following fact [8]:

$$\mathrm{Tr}\left(\mathbf{A}(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\right) = \mathbb{E}\left\{\mathbf{n}_0^{\mathrm{T}}\mathbf{A}(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\mathbf{n}_0\right\} \quad (11)$$

[3]The derivation of $\mathbf{J}_{\mathbf{y}}\left(\mathbf{u}^{(i,j)}\right)$ based on vector calculus is very complicated, we omit it here to save the page space.

with $\mathbf{n}_0 \sim \mathcal{N}(0,\mathbf{I}_N)$. From (8), we have:

$$\begin{cases}\overbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\mathbf{n}_0}^{\mathbf{n}_{\mathbf{x}}^{(i)}} = \overbrace{\mathbf{B}^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{n}_0}^{\mathbf{n}_1} + \mu_1\mathbf{B}^{-1}\mathbf{D}_1^{\mathrm{T}}\overbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{z}_1^{(i-1)})\mathbf{n}_0}^{\mathbf{n}_{\mathbf{z}_1}^{(i-1)}} + \mu_2\mathbf{B}^{-1}\overbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{z}_2^{(i-1)})\mathbf{n}_0}^{\mathbf{n}_{\mathbf{z}_2}^{(i-1)}}\\ \underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{z}_1^{(i)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{z}_1}^{(i)}} = \mathbf{P}^{(i)}\mathbf{D}_1\underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{x}}^{(i)}}\end{cases} \quad (12)$$

The $\mathbf{n}_{\mathbf{z}_2}^{(i)}$ can be obtained by Chambolle's algorithm from $\mathbf{n}_{\mathbf{x}}^{(i)}$:

$$\begin{cases}\overbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{u}_1^{(i,j+1)})\mathbf{n}_0}^{\mathbf{n}_{\mathbf{u}_1}^{(i,j+1)}} = \mathbf{V}^{(i,j)}\overbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{u}_1^{(i,j)})\mathbf{n}_0}^{\mathbf{n}_{\mathbf{u}_1}^{(i,j)}} - \frac{\tau\mu_2}{\lambda_2}\mathbf{Q}_1^{(i,j)}\overbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{z}_2^{(i,j)})\mathbf{n}_0}^{\mathbf{n}_{\mathbf{z}_2}^{(i,j)}}\\ \underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{u}_2^{(i,j+1)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_2}^{(i,j+1)}} = \mathbf{V}^{(i,j)}\underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{u}_2^{(i,j)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_2}^{(i,j)}} - \frac{\tau\mu_2}{\lambda_2}\mathbf{Q}_2^{(i,j)}\underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{z}_2^{(i,j)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{z}_2}^{(i,j)}}\end{cases} \quad (13)$$

and

$$\underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{z}_2^{(i,j)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{z}_2}^{(i,j)}} = \underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{x}^{(i)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{x}}^{(i)}} + \frac{\lambda_2}{\mu_2}(\mathbf{D}_2^{(1)})^{\mathrm{T}}\underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{u}_1^{(i,j)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_1}^{(i,j)}} + \frac{\lambda_2}{\mu_2}(\mathbf{D}_2^{(2)})^{\mathrm{T}}\underbrace{\mathbf{J}_{\mathbf{y}}(\mathbf{u}_2^{(i,j)})\mathbf{n}_0}_{\mathbf{n}_{\mathbf{u}_2}^{(i,j)}} \quad (14)$$

where $\mathbf{Q}_1^{(i,j)} = (\mathbf{W}_1^{(i,j)}\mathbf{C}_1^{(i,j)} + \mathbf{V}^{(i,j)})\mathbf{D}_2^{(1)} + \mathbf{W}_1^{(i,j)}\mathbf{C}_2^{(i,j)}\mathbf{D}_2^{(2)}$ and $\mathbf{Q}_2^{(i,j)} = \mathbf{W}_2^{(i,j)}\mathbf{C}_1^{(i,j)}\mathbf{D}_2^{(1)} + (\mathbf{W}_2^{(i,j)}\mathbf{C}_2^{(i,j)} + \mathbf{V}^{(i,j)})\mathbf{D}_2^{(2)}$.

Thus, instead of using (8), (9) and (10), we can successfully compute the trace by (11)–(14), without the explicit storage of huge matrices, summarized as follows. The flowchart is shown in Fig.1.

Algorithm 2: MC for SURE of BDF algorithm (2-D)

for $i = 1,2,\ldots$ *(BDF iteration)* **do**
- **1** update $\mathbf{x}^{(i)}$, $\mathbf{z}_1^{(i)}$ and $\mathbf{z}_2^{(i)}$ by (5) and (6);
- **2** update $\mathbf{n}_{\mathbf{x}}^{(i)}$, $\mathbf{n}_{\mathbf{z}_1}^{(i)}$ and $\mathbf{n}_{\mathbf{z}_2}^{(i)}$ by (12), (13) and (14);
- **3** compute the trace by (11);
- **4** compute SURE of i-th iterate by (7);

end

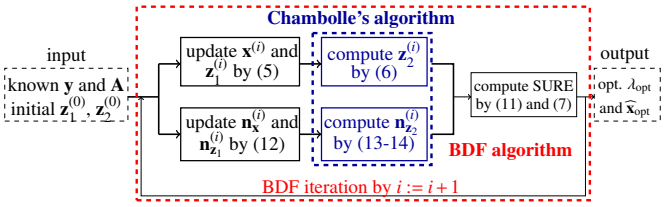

Fig. 1. SURE-MC evaluation for BDF algorithm (Chambolle's algorithm is for $\mathbf{z}_2^{(i)}$).

To find the optimal values of λ_1 and λ_2, an intuitive idea is to repeatedly implement **Alg. 1** for various tentative values of λ_1 and λ_2, then, the minimum SURE indicates the optimal values (see Fig.4 for example). This *global search* has been frequently used in [11–13].

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, we exemplify the proposed algorithm with image deconvolution, by considering a test image *Camerman*, blurred by a Gaussian kernel. The noise level corresponds to blur-SNR[4] of 30dB. For image processing, we have to use MC to compute SURE as **Alg. 2** and Fig.1, due to the large sizes of $\mathbf{A}$, $\mathbf{D}_1$ and Jacobians.

First, we solve (2) with fixed values of λ_1 and λ_2. Fig.2 shows the BDF convergence and the evolution of SURE. We can see that the SURE is always a reliable substitute for MSE during the iterations.

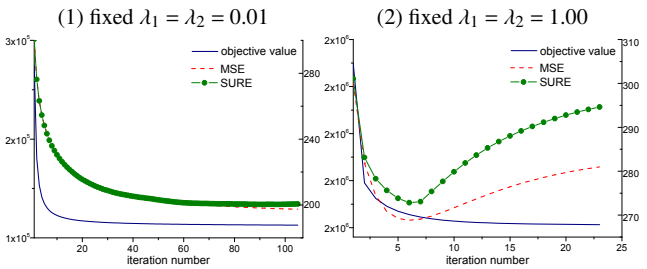

Fig. 2. The convergence of BDF algorithm with fixed values of λ_1 and λ_2.

We repeatedly implement **Alg. 2** to perform global optimization of λ_1 or λ_2, with fixed another, and show the results in Fig.3. Fig.4 shows the global optimization of λ_1 and λ_2, within the interval of $[10^{-3}, 10^{0}]$. The optimal values of λ_1 and λ_2 obtained by minimizing SURE are very close to the *oracle* results of minimum MSE.

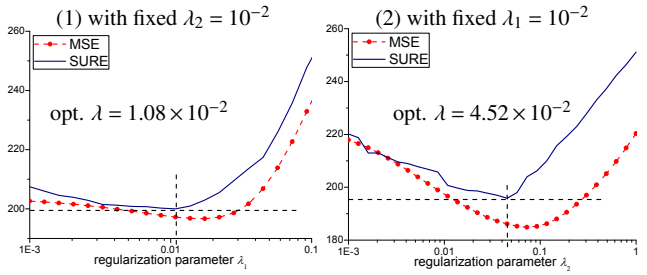

Fig. 3. The global optimization of λ_1 or λ_2, when fixing another.

[4]Blur signal-to-noise ratio (BSNR) is defined as: $10\log_{10}\left(\frac{\|\mathbf{A}\mathbf{x}_0 - \text{mean}(\mathbf{A}\mathbf{x}_0)\|_2^2}{M\sigma^2}\right)$ in dB.

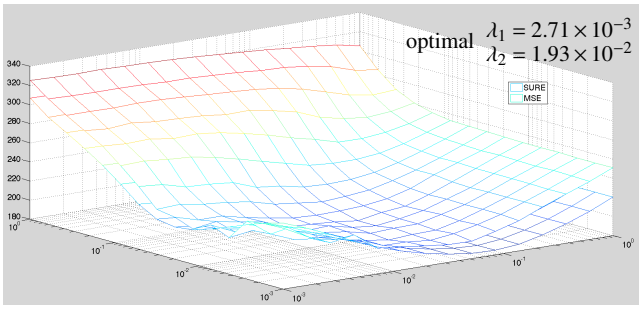

Fig. 4. The global optimization of λ_1 and λ_2.

Fig.5 shows a visual comparison between SURE and MSE minimization. We can see that the SURE minimization yields the PSNR loss within 0.2dB, compared to the *oracle* optimal performance.

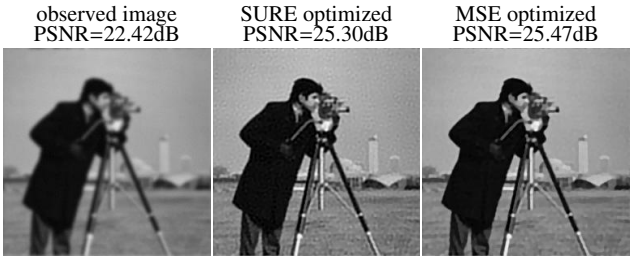

Fig. 5. A visual example of *Cameraman*.

5. CONCLUSIONS

In this paper, we presented a SURE-based automatic method of tuning multiple regularization parameters for (TV+ℓ_1) compound regularizers, based on BDF algorithm [6]. Future work will deal with extension of this technique to handle other hybrid regularizers and the development of fast optimization algorithm instead of the time-consuming global search (shown as Fig.4).

6. ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China under grant number 61401013.

7. REFERENCES

[1] F. Xue, F. Luisier, and T. Blu, "Multi-Wiener SURE-LET deconvolution," *IEEE Transactions on Image Processing*, vol. 22, no. 5, pp. 1954–1968, 2013.

[2] F. Xue and T. Blu, "A novel SURE-based criterion for parametric PSF estimation," *IEEE Transactions on Image Processing*, vol. 24, no. 2, pp. 595–607, 2015.

[3] M. Lustig, D. Donoho, and J. Pauly, "Sparse MRI: the application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine*, vol. 58, pp. 1182–1195, 2007.

[4] J. Starck, M. Elad, and D. Donoho, "Image decomposition via the combination of sparse representations and a variational approach," *IEEE Trans. Image. Proc.*, vol. 14, no. 10, pp. 1570–1582, 2005.

[5] You-Wei Wen, Michael K. Ng, and Wai-Ki Ching, "Iterative slgorithms based on the decouple of deblurring and denoising for image restoration," *SIAM Journal on Scientific Computing*, vol. 30, no. 5, pp. 2655–2674, 2008.

[6] J.M. Bioucas-Dias and M.A.T. Figueiredo, "An iterative algorithm for linear inverse problems with compound regularizers," in *IEEE Int. Conf. on Image Proc.*, 2008, pp. 685–688.

[7] F. Bauer and M.A. Lukas, "Comparing parameter choice methods for regularization of ill-posed problem," *Mathematics and Computers in Simulation*, vol. 81, no. 9, pp. 1795–1841, 2011.

[8] T. Blu and F. Luisier, "The SURE-LET approach to image denoising," *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2778–2786, 2007.

[9] Charles M Stein, "Estimation of the mean of a multivariate normal distribution," *The Annals of Statistics*, pp. 1135–1151, 1981.

[10] Y.C. Eldar, "Generalized SURE for exponential families: Applications to regularization," *IEEE Transactions on Signal Processing*, vol. 57, no. 2, pp. 471–481, 2009.

[11] F. Xue, Anatoly G. Yagola, J.Q. Liu, and G. Meng, "Recursive SURE for iterative reweighted least square algorithms," *Inverse Problems in Science and Engineering*, vol. 24, no. 4, pp. 625–646, 2016.

[12] R. Giryes, M. Elad, and Y.C. Eldar, "The projected GSURE for automatic parameter tuning in iterative shrinkage methods," *Applied and Computational Harmonic Analysis*, vol. 30, no. 3, pp. 407–422, 2010.

[13] C. Vonesch, S. Ramani, and M. Unser, "Recursive risk estimation for non-linear image deconvolution with a wavelet-domain sparsity constraint," in *IEEE Int. Conf. on Image Proc.*, 2008, pp. 665–668.

[14] P.L. Combettes and V.R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling and Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.

[15] N. Parikh and S. S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 123–231, 2013.

[16] A. Chambolle, "An algorithm for total variation minimization and applications," *Journal of Mathematical Imaging and Vision*, vol. 20, pp. 89–97, 2004.

[17] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W.T. Yin, "An iterative regularization method for total variation-based image restoration," *SIAM J. Multiscale Model. Simul.*, vol. 4, no. 2, pp. 460–489, 2005.