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

Feng Xuea, Hanjie Panb, Runhui Wua, Xin Liua and Jiaqi Liua

aNational Key Laboratory of Science and Technology on Test Physics and Numerical Mathematics, Beijing, 100076, China
bAudiovisual 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 $x_0 \in \mathbb{R}^N$ from the following observation model [1, 2]:

$$y = Ax_0 + \epsilon$$

where $y \in \mathbb{R}^M$ is the observed noisy data, $A \in \mathbb{R}^{M \times N}$ 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]:

$$\min_x \frac{1}{2} \|Ax - y\|_2^2 + \lambda_1 \cdot J_1(x) + \lambda_2 \cdot J_2(x)$$

where both $J_1$ and $J_2$ are simple regularizers, $\lambda_1$ and $\lambda_2$ 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 $\lambda_1$ and $\lambda_2$ 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]:

$$\text{MSE} = \frac{1}{N} \mathbb{E} \left[ \|\hat{x} - x_0\|_2^2 \right]$$

and attempt to select the values of $\lambda_1$ and $\lambda_2$, such that the corresponding solution $\hat{x}$ achieves minimum MSE. Notice that MSE (3) is inaccessible due to the unknown $x_0$. In practice, Stein’s unbiased risk estimate (SURE) has been proposed as a statistical substitute for MSE (if $A$ is full-rank matrix) [9, 10]:

$$\text{SURE} = \frac{1}{N} \left( \|y\|_2^2 - 2y^T A (A^T A)^{-1} \hat{x} + \lambda^2 \text{Tr}(A (A^T A)^{-1} J_2(\hat{x})) + \|x_0\|_2^2 \right)$$

since it depends on the observed data $y$ only. Here, $J_2(\hat{x}) \in \mathbb{R}^{N \times N}$ is a Jacobian matrix defined as [11]:

$$J_2(\hat{x})_{i,j} = \frac{\partial x_i}{\partial y_j}$$

Recently, SURE has become a popular criterion for optimization, in the context of non-linear denoising and deconvolution [1, 8], and $\ell_1$-based sparse reconstruction [11–13].

1The last term of (4) $\|x_0\|_2^2$—is constant irrelevant to the optimization of $\hat{x}$. 
2. AN APPLICATION OF BDF ALGORITHM TO TV+$\ell_1$ MINIMIZATION

2.1. Basic scheme of BDF algorithm

The problem (2) is equivalent to the following:

$$\min_{x} \frac{1}{2} \|Ax - y\|^2 + \lambda_1 \cdot f_1(z_1) + \lambda_2 \cdot f_2(z_2)$$

subject to $\|z_1 - D_1 x\|^2 = 0; \|z_2 - D_2 x\|^2 = 0$

which, by Lagrangian, is equivalent to:

$$\min_{x} \frac{1}{2} \|Ax - y\|^2 + \lambda_1 f_1(z_1) + \lambda_2 f_2(z_2) + \mu_1 \|z_1 - D_1 x\|^2 + \mu_2 \|z_2 - D_2 x\|^2$$

where $\mu_1$ and $\mu_2$ are the augmented Lagrangian penalty parameters.

To minimize this functional w.r.t. $x$, $z_1$, and $z_2$, BDF algorithm is to alternatively minimize w.r.t. these variables:

$$x^{(i+1)} = \arg\min_{x} \frac{1}{2} \|Ax - y\|^2 + \sum_{i=1}^{2} \mu_i \|D_i x - z_i^{(i)}\|^2$$

$$z_1^{(i+1)} = \arg\min_{z_1} \frac{1}{2} \|z_1 - D_1 x^{(i+1)}\|^2 + \frac{1}{\mu_1} f_1(z_1)$$

$$z_2^{(i+1)} = \arg\min_{z_2} \frac{1}{2} \|z_2 - D_2 x^{(i+1)}\|^2 + \frac{1}{\mu_2} f_2(z_2)$$

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

2.2. Illustration with wavelet-$\ell_1$ and TV regularizers

To exemplify the iterative algorithm, we consider signal deconvolution problem, with both wavelet-$\ell_1$ and TV regularizers, i.e., $f_1(D_1 x) = \|D_1 x\|_1$ and $f_2(x) = TV(x)$, where $D_1$ denotes wavelet decomposition. Thus, the problem becomes:

$$\min_{x} \frac{1}{2} \|Ax - y\|^2 + \lambda_1 \|z_1\|_1 + \lambda_2 TV(z_2) + \mu_1 \|D_1 x - z_1\|^2 + \mu_2 \|x - z_2\|^2$$

which yields the following iteration:

$$x^{(i)} = B^{-1} (A^T y + \mu_1 D_1^T z_1^{(i-1)} + \mu_2 z_2^{(i-1)})$$

$$z_1^{(i)} = T_{\lambda_1/\mu_1} (D_1 x^{(i)})$$

$$z_2^{(i)} = \arg\min_{z_2} \frac{1}{2} \|z_2 - x^{(i)}\|^2 + \frac{1}{\mu_2} TV(z_2)$$

where $B = A^T A + \mu_1 D_1^T D_1 + \mu_2 I$. $T_{\tau} (\cdot)$ denotes the pointwise soft-thresholding with threshold $T$ [11]. $z_2^{(i)}$ can be efficiently solved by Chambolle’s algorithm [16].

For 2-D case, we consider the TV definition as $TV(x) = \sum_{n=1}^{N} \sqrt{(D_1^{(1)} x_n)^2 + (D_2^{(2)} x_n)^2} + \alpha$, where $D_1^{(1)}$ and $D_2^{(2)}$ denote the first-order differences along horizontal and vertical directions, $\alpha$ 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 $z_2^{(i)}$ of (5) can be expressed in matrix language as (iterate by $j$):

$$u^{(i,j+1)} = \nabla (u^{(i,j)} - \frac{T\mu_2}{\mu_2} D_2 (x^{(i,j)} + \frac{\lambda_2}{\mu_2} D_1^T u^{(i,j)}))$$

where $\tau$ is a step-size, $D_2$ and diagonal matrix $\nabla (i,j)$ are

$$D_2 = \begin{bmatrix} D_1^{(1)} \\ D_2^{(2)} \end{bmatrix} \in \mathbb{R}^{2N \times N}; \quad \nabla (i,j) = \begin{bmatrix} V^{(i,j)} & 0 \\ 0 & V^{(i,j)} \end{bmatrix} \in \mathbb{R}^{2N \times 2N}$$

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

$$V^{(i,j)} = \left(1 + \frac{T\mu_2}{\mu_2} \sqrt{(D_1^{(1)} x^{(i,j)})^2 + (D_2^{(2)} z^{(i,j)}_2)^2} + \alpha \right)^{-1}$$

Finally, $z_2^{(i)}$ is obtained by the convergence of Chambolle’s iteration (6): $z_2^{(i)} = z_2^{(i,\infty)}$ 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:

$$SURE = \frac{1}{N} \left( \|x^{(i)}\|^2 - 2y^T A (A^T A)^{-1} x^{(i)} + 2\sigma^2 Tr(A (A^T A)^{-1} J_y(x^{(i)})) \right)$$

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

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

$$J_y(x^{(i)}) = B^{-1} (A^T + \mu_1 D_1^T J_y(x^{(i-1)}) + \mu_2 J_y(x^{(i-1)}))$$

$$J_y(z_2^{(i)}) = P^{(i)} D_1 J_y(x^{(i)})$$

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

$$P^{(i)} = \begin{cases} 1, & \text{if } |D_1 x^{(i)}| \geq \lambda_1/\mu_1 \\ 0, & \text{otherwise} \end{cases}$$

The recursion of $J_y(z_2^{(i)})$ has to be obtained by Chambolle’s algorithm. We consider 2-D case only.

\footnote{In the remainder of this paper, the last constant term—$|x_0|^2$—is ignored for brevity.}
3.2. Recursion of $J_y(z_i^{(0)})$ for Chambolle’s algorithm

For 2-D case, we express $u^{(i,j+1)}$ of (6) as $u^{(i,j+1)} = \begin{bmatrix} u_1^{(i,j+1)} \\ u_2^{(i,j+1)} \end{bmatrix}$ where the first part $u_1^{(i,j+1)}$ is:

$$u_1^{(i,j+1)} = \mathbf{v}^{(i,j)} \left( \frac{\frac{\partial^2 J}{\partial u_1^2}}{\frac{\partial^2 J}{\partial u_1^2}} \right) = _{w_i^{(i,j)}}$$

After derivations, we obtain:

$$J_y(u^{(i,j+1)}) = \begin{bmatrix} v_1^{(i,j)} & 0 \\ 0 & v_2^{(i,j)} \end{bmatrix} J_y(u^{(i,j)}) - \frac{\tau \mu_1}{\lambda_2}$$

where $C_1^{(i,j)}$ and $C_2^{(i,j)}$ are diagonal:

$$C_1^{(i,j)} = \frac{a_m (v_1^{(i,j)})^2}{\sqrt{a_m^2 + b_m^2 + \alpha}}; \quad C_2^{(i,j)} = \frac{b_m (v_2^{(i,j)})^2}{\sqrt{b_m^2 + \alpha}}$$

with $a = D_1^{(i,j)}$, $b = D_2^{(i,j)}$, $D_1$ and $D_2$ are diagonal: $[W_1^{(i,j)}]_{m,n} = [w_1^{(i,j)}]_{m,n}$ $[W_2^{(i,j)}]_{m,n} = [w_2^{(i,j)}]_{m,n}$.

Note that $\mathbf{z}_2^{(i,j)} = \mathbf{x}_1^{(i,j)}$ and $
abla J_y(u^{(i,j)})$, we have:

$$J_y(z_2^{(i,j)}) = J_y(x_1^{(i,j)}) + \frac{\lambda_2}{\mu_2} \mathbf{D} \nabla J_y(u^{(i,j)})$$

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 $\lambda_1$ and $\lambda_2$, and simultaneously evaluate the SURE during the BDF iterations.

**Algorithm 1: SURE evaluation for BDF algorithm**

```plaintext
for i = 1, 2, ... (BDF iteration) do
    1. update $\mathbf{x}_1^{(i)}$, $\mathbf{z}_1^{(i)}$ and $\mathbf{z}_2^{(i)}$ by (5) and (6);
    2. update $J_y(x_1^{(i)})$, $J_y(z_1^{(i)})$ and $J_y(z_2^{(i)})$ 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 $A$, $D_1$ and Jacobians. Monte-Carlo (MC) simulation provides an alternative way to compute the trace by the following fact [8]:

$$\text{Tr} \left( \mathbf{A} (\mathbf{A}^T \mathbf{A})^{-1} J_y(x^{(i)}) \right) = \mathbb{E} \left[ n_i^T \mathbf{A} (\mathbf{A}^T \mathbf{A})^{-1} J_y(x^{(i)}) n_i \right]$$

The derivation of $J_y(u^{(i,j)})$ based on vector calculus is very complicated, we omit it here to save the page space.

with $n_0 \sim N(0, I_N)$. From (8), we have:

$$\begin{cases}
J_y(x_1^{(i)}) | n_0 = \mathbf{A}^T \mathbf{A}^{-1} J_y(x_1^{(i)}) | n_0 + \frac{\mu_1}{\lambda_2} D_1 J_y(x_1^{(i)}) | n_0 + \mu_2 B^{-1} D_2 J_y(x_1^{(i)}) | n_0 \\
J_y(x_2^{(i)}) | n_0 = \mathbf{A}^T \mathbf{A}^{-1} J_y(x_2^{(i)}) | n_0 + \frac{\mu_1}{\lambda_2} D_1 J_y(x_2^{(i)}) | n_0 + \mu_2 B^{-1} D_2 J_y(x_2^{(i)}) | n_0
\end{cases}$$

The $\mathbf{n}_1^{(i)}$ can be obtained by Chambolle’s algorithm from $\mathbf{n}_x^{(i)}$.

$$\begin{cases}
J_y(x_1^{(i)}) | n_0 = \mathbf{A}^T \mathbf{A}^{-1} J_y(x_1^{(i)}) | n_0 + \frac{\mu_1}{\lambda_2} Q_1^{(i)} | n_0 + \mu_2 B^{-1} D_2 J_y(x_1^{(i)}) | n_0 \\
J_y(x_2^{(i)}) | n_0 = \mathbf{A}^T \mathbf{A}^{-1} J_y(x_2^{(i)}) | n_0 + \frac{\mu_1}{\lambda_2} Q_2^{(i)} | n_0 + \mu_2 B^{-1} D_2 J_y(x_2^{(i)}) | n_0
\end{cases}$$

and

$$\begin{cases}
J_y(x_1^{(i)} | n_0 = J_y(x_1^{(i)}) | n_0 + \frac{\lambda_1}{\mu_2} (D_1^{-1})^T J_y(x_1^{(i)}) | n_0 + \frac{\lambda_1}{\mu_2} (D_2^{-1})^T J_y(x_2^{(i)}) | n_0 \\
J_y(x_2^{(i)} | n_0 = J_y(x_2^{(i)}) | n_0 + \frac{\lambda_1}{\mu_2} (D_1^{-1})^T J_y(x_1^{(i)}) | n_0 + \frac{\lambda_1}{\mu_2} (D_2^{-1})^T J_y(x_2^{(i)}) | n_0
\end{cases}$$

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 $i = 1, 2, ..., \text{(BDF iteration)}$ do
    1. update $\mathbf{x}_1^{(i)}$, $\mathbf{z}_1^{(i)}$ and $\mathbf{z}_2^{(i)}$ by (5) and (6);
    2. update $\mathbf{n}_1^{(i)}$, $\mathbf{z}_1^{(i)}$ and $\mathbf{z}_2^{(i)}$ 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 $\mathbf{z}_2^{(i)}$).
4. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, we exemplify the proposed algorithm with image deconvolution, by considering a test image Cameraman, 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 A, D, and Jacobians.

First, we solve (2) with fixed values of \(\lambda_1\) and \(\lambda_2\). 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 \(\lambda_1\) and \(\lambda_2\).](image)

We repeatedly implement Alg. 2 to perform global optimization of \(\lambda_1\) or \(\lambda_2\), with fixed another, and show the results in Fig.3. Fig.4 shows the global optimization of \(\lambda_1\) and \(\lambda_2\), within the interval of \([10^{-3}, 10^0]\). The optimal values of \(\lambda_1\) and \(\lambda_2\) obtained by minimizing SURE are very close to the \textit{oracle} results of minimum MSE.

![Fig. 3. The global optimization of \(\lambda_1\) or \(\lambda_2\), when fixing another.](image)

\(^4\text{Blur signal-to-noise ratio (BSNR) is defined as:}
10\log_{10}\left(\frac{||Ax_0 - \text{mean}(Ax_0)||^2}{M^2}\right)\text{ in dB.}\)

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 \textit{oracle} optimal performance.

![Fig. 4. The global optimization of \(\lambda_1\) and \(\lambda_2\).](image)

![Fig. 5. A visual example of Cameraman.](image)

5. CONCLUSIONS

In this paper, we presented a SURE-based automatic method of tuning multiple regularization parameters for \((TV+f_1)\) 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


