Convergence Analysis of Tight Framelet Approach for Missing Data Recovery

Jian-Feng Cai[*], Raymond H. Chan[†], Lixin Shen[‡], and Zuowei Shen[§]

Abstract

How to recover missing data from an incomplete samples is a fundamental problem in mathematics and it has wide range of applications in image analysis and processing. Although many existing methods, e.g. various data smoothing methods and PDE approaches, are available in the literature, there is always a need to find new methods leading to the best solution according to various cost functionals. In this paper, we propose an iterative algorithm based on tight framelets for image recovery from incomplete observed data. The algorithm is motivated from our framelet algorithm used in high-resolution image reconstruction and it exploits the redundance in tight framelet systems. We prove the convergence of the algorithm and also give its convergence factor. Furthermore, we derive the minimization properties of the algorithm and explore the roles of the redundancy of tight framelet systems. As an illustration of the effectiveness of the algorithm, we give an application of it in impulse noise removal.

1 Introduction

The recovery of missing data from incomplete data is an essential part of any image processing procedures whether the final image is to be utilized for visual interpretation or for automatic analysis. The problem can be formulated as follows. Let f be the original image defined on an image domain Ω and $(f+\epsilon)_{|\Lambda}$ be the observable data on $\Lambda \subset \Omega$. Here ϵ is the noise. The task of the missing data recovery is to find or approximate f on $\Omega \backslash \Lambda$ from the available data $(f+\epsilon)_{|\Lambda}$ such that the restored image retains as many important image features of $(f+\epsilon)_{|\Lambda}$ as possible. The problem arises, for examples, in image inpainting and impulse noise removal. A number of approaches for solving the problem has been proposed in recent years, see, for examples, [2, 4, 10, 11, 19, 20, 25].

In our previous work [4, 10], we proposed iterative algorithms based on tight frames for recovering images from missing data. In each iteration of these algorithms, we modify the tight frame coefficients of each iterate via some thresholding operators. This has a built-in regularization effect in the sense that the tight frame coefficients minimize a functional which penalizes a weighted ℓ_1 norm of the tight frame coefficients as well as the distance between these coefficients and the range of the underlying tight frame system. The effect allows the tight frame coefficients from Λ

[*]Temasek Laboratories and Department of Mathematics, National University of Singapore, 2 Science Drive 2, Singapore 117543. Email:tslcaij@nus.edu.sg

[†]Department of Mathematics, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, P. R. China. Email:rchan@math.cuhk.edu.hk. The research was supported in part by HKRGC Grant 400505 and CUHK DAG 2060257.

[‡]Department of Mathematics, Syracuse University, Syracuse, NY 13244. Email: lshen03@syr.edu. The research was supported by the US National Science Foundation under grant DMS-0712827.

[§]Department of Mathematics, National University of Singapore, 2 Science Drive 2, Singapore 117543. Email: matzuows@nus.edu.sg. The research was supported in part by Grant R-146-000-060-112 at the National University of Singapore.

to affect the tight frame coefficients in $\Omega \setminus \Lambda$ in a smooth way. By interpreting the algorithm as a proximal forward-backward splitting [13] iteration for a minimization problem, we proved in [4] the convergence of the algorithm and found the minimization properties that the limit of the algorithm satisfies. However, there is no result on the rate of the convergence of the algorithm.

To obtain an algorithm with a convergence factor, in this paper we modify the algorithms in [4, 10] by fixing the scaling (i.e., low-frequency) coefficients of each iteration to be those of the initial guess. We prove that the modified algorithm converges provided that a low-pass filtering matrix associated with the low-pass filter of the tight frame system is nonsingular. We then show that the convergence factor of the new algorithm depends on the smallest singular value of this low-pass filtering matrix. We also prove that the non-singularity of the matrix can always be satisfied by adjusting the size of the images. Furthermore, we present the minimization properties of the limit of the new algorithm, and explore the roles of the redundancy of tight frame systems. We then show the efficiency of our algorithm by applying it to impulse noise removal.

The outline of this paper is as follows: In Section 2, we briefly introduce the tight frame theory. In Section 3, we review the framelet-based missing data recovery algorithms in [4, 10], and propose our new modified algorithm. In Section 4, we prove the convergence of the new algorithm and give the minimization properties of the solution (i.e., the limit) of the algorithm. In Section 5, we introduce the impulse noise models and adapt our new algorithm to impulse noise removal. The numerical tests are presented in Section 6 and conclusions are given in Section 7.

2 Tight Frames

In this section, we review some basics of tight frames that are sufficient to present our algorithm in this paper. More detailed theory can be found, for example, in [14, 16, 27].

Let $\mathcal{A}$ be a K-by-N ($K \geq N$) matrix whose rows are vectors in $\mathbb{R}^N$. The system, also denoted by $\mathcal{A}$, consisting of all the rows of $\mathcal{A}$, is a *tight frame* for $\mathbb{R}^N$ if for an arbitrary $f \in \mathbb{R}^N$,

$$\|f\|^2 = \sum_{g \in \mathcal{A}} |\langle f, g \rangle|^2.$$

The associated *analysis operator* is defined again as $\mathcal{A} : \mathbb{R}^N \to \mathbb{R}^K$ with

$$\mathcal{A}f = \{\langle f, g \rangle\}_{g \in \mathcal{A}}, \qquad \forall f \in \mathbb{R}^N.$$

Its adjoint operator, the *synthesis operator*, is $\mathcal{A}^* : \mathbb{R}^K \to \mathbb{R}^N$ with

$$\mathcal{A}^* c = \sum_{g \in \mathcal{A}} c_g g, \qquad c = \{c_g\}_{g \in \mathcal{A}}.$$

Hence, $\mathcal{A}$ is a tight frame if and only if $\mathcal{A}^*\mathcal{A} = \mathcal{I}$. When $\ker \mathcal{A}^* \neq \{0\}$, the system $\mathcal{A}$ is redundant and $\mathcal{A}\mathcal{A}^* \neq \mathcal{I}$. If a tight frame satisfies $\ker \mathcal{A}^* = \{0\}$, then it becomes an orthogonal basis.

The tight frame $\mathcal{A}$ used in our algorithm are generated from the spline wavelet frame filters by using the unitary extension principle given in [27]. Let us mention how they are generated. Let

$$\widehat{h}_0(\omega) = \cos^{2m}(\omega/2). \tag{2.1}$$

Note that the trigonometric polynomial $\widehat{h}_0$ is the refinement symbol of the B-spline

$$\widehat{\phi}(\omega) = \frac{\sin^{2m}(\omega/2)}{(\omega/2)^{2m}}$$

of order $2m$. The wavelets corresponding to ϕ are given by

$$\widehat{\psi}_n(\omega) = \widehat{h}_n(\omega/2)\widehat{\phi}(\omega/2)$$

with wavelet masks

$$\widehat{h}_n(\omega) = \sqrt{\binom{2m}{n}} \sin^n(\omega/4)\cos^{2m-n}(\omega/4), \tag{2.2}$$

for $1 \le n \le 2m$. The low-pass filter (scaling mask) $\widehat{h}_0$ and the high-pass filters (wavelet masks) $\widehat{h}_n$, $1 \le n \le 2m$ satisfy the condition of the unitary extension principle (see [27])

$$\sum_{k=0}^{2m} |\widehat{h}_k(\omega)|^2 = 1 \quad \text{and} \quad \sum_{k=0}^{2m} \overline{\widehat{h}_k(\omega)}\widehat{h}_k(\omega+\pi) = 0 \tag{2.3}$$

for $\omega \in [-\pi, \pi]$. It is shown in [27] that the system $\mathcal{X} = \{2^{k/2}\psi_n(2^k \cdot -j) : k,\, j \in \mathbb{Z}; n = 1, \ldots, 2m\}$ is a tight framelet system. The choices of $m = 1$ and $m = 2$ give the so-called *piecewise linear* and *piecewise cubic* tight framelet systems, respectively. Specifically, the filters associated with the piecewise cubic tight framelet systems are

$$\begin{aligned} h_0 &= \frac{1}{16}[1,4,6,4,1]; \quad h_1 = \frac{1}{8}[1,2,0,-2,-1]; \\ h_2 &= \frac{\sqrt{6}}{16}[-1,0,2,0,-1]; \quad h_3 = \frac{1}{8}[-1,2,0,-2,1]; \quad h_4 = \frac{1}{16}[1,-4,6,-4,1]. \end{aligned}$$

where h_0 is the low-pass filter while h_k, $1 \le k \le 2m$, are the high-pass filters. The filters h_0, h_2, h_4 are symmetric while h_1, h_3 are antisymmetric.

From the given spline tight framelet filters h_k, $0 \le k \le 2m$, we can design a tight frame system $\mathcal{A}$ associated with these filters. Let h be a filter with length $2m+1$, i.e.,

$$h = [h(-m), h(-m+1), \ldots, h(-1), h(0), h(1), \ldots, h(m-1), h(m)]. \tag{2.4}$$

If Neumann boundary condition is used, then the matrix representation of h will be an N-by-N matrix H given by

$$[H]_{i,j} = \begin{cases} h(i-j) + h(i+j-1), & \text{if } i+j \le (m+1), \\ h(i-j) + h(-1-(2n-i-j)) & \text{if } i+j \ge 2N-m+1, \\ h(i-j) & \text{otherwise.} \end{cases} \tag{2.5}$$

When the filter h is symmetric, the resulting matrix H will have a Hankel-plus-Toeplitz structure and its spectra can be computed easily, see [24]. We note that Neumann boundary conditions usually produce restored images having less artifacts near the boundary, see [6, 24] for instances.

Next we define the matrices $\mathcal{L}_k$ and $\mathcal{H}_k$:

$$\mathcal{L}_k = H_0^{(k)} H_0^{(k-1)} \cdots H_0^{(1)} \quad \text{and} \quad \mathcal{H}_k = \begin{bmatrix} H_1^{(k)}\mathcal{L}_{k-1} \\ \vdots \\ H_{2m}^{(k)}\mathcal{L}_{k-1} \end{bmatrix}, \quad k > 0, \tag{2.6}$$

where $H_k^{(\ell)}$ are the matrix representations of the filters formed from h_k by inserting $2^{\ell-1}-1$ zeros between every two adjacent components of h_k. With $\mathcal{L}_k$ and $\mathcal{H}_k$ in hand, for any given integer

$L > 0$, a matrix $\mathcal{A}$ induced from the spline tight framelets is given as follows

$$\mathcal{A} := \begin{bmatrix} \mathcal{L}_L \\ \hline \mathcal{H}_L \\ \mathcal{H}_{L-1} \\ \vdots \\ \mathcal{H}_1 \end{bmatrix} \equiv \left[\begin{array}{c} \mathcal{L}_L \\ \hline \\ \\ \mathcal{A}_L \\ \\ \end{array}\right]. \tag{2.7}$$

Applying (2.3), one obtains

$$\mathcal{L}_k^*\mathcal{L}_k + \mathcal{H}_k^*\mathcal{H}_k = \mathcal{L}_{k-1}^*\mathcal{L}_{k-1}, \qquad 0 < k \le L,$$

which leads to

$$\mathcal{A}^*\mathcal{A} = I, \quad \text{and} \quad \mathcal{L}_L^*\mathcal{L}_L + \mathcal{A}_L^*\mathcal{A}_L = I.$$

Hence, $\mathcal{A}$ is a tight frame in $\mathbb{R}^N$, and $\|\mathcal{A}v\| = \|v\|$ for all $v \in \mathbb{R}^N$.

So far we have only considered tight framelet systems in 1-D. Since images are 2-D objects, when we handle images, we use tensor product tight framelet system generated by the corresponding univariate tight framelet system. Let $H_k^{(\ell)}$, $0 \le k \le 2m$ and $1 \le \ell < L$, be matrices used in (2.6). Define

$$G_{(2m+1)i+j}^{(\ell)} := H_i^{(\ell)} \otimes H_j^{(\ell)} \tag{2.8}$$

where $0 \le i, j \le 2m$. Clearly

$$\sum_{k=0}^{(2m+1)^2-1} (G_k^{(\ell)})^* G_k^{(\ell)} = I.$$

We then define

$$\mathcal{L}_k := G_0^{(k)} G_0^{(k-1)} \cdots G_0^{(1)} \quad \text{and} \quad \mathcal{H}_k := \begin{bmatrix} G_1^{(k)}\mathcal{L}_{k-1} \\ \vdots \\ G_{(2m+1)^2-1}^{(k)}\mathcal{L}_{k-1} \end{bmatrix}. \tag{2.9}$$

With these notations, we can similarly form the matrices $\mathcal{A}$ and $\mathcal{A}_L$ using (2.7).

3 Algorithms for Recovering Missing Data

In the first part of this section, we give a brief review of the framelet-based algorithm proposed in [4, 10] for the recovery of missing data. The algorithm is motivated by the redundant nature of the tight frame system and it has been applied to high-resolution image reconstruction [9]. The idea was first mentioned in [6] for frames generated by bi-orthogonal wavelet systems. In the second part of the section, we propose a new algorithm by modifying the algorithm proposed in [4, 10].

Recall that we are given $g = f + \epsilon$ on Λ and like to recover f in $\Omega \setminus \Lambda$. Define the projection operator

$$P_\Lambda g(i) = \begin{cases} g(i), & i \in \Lambda, \\ 0, & i \in \Omega \backslash \Lambda, \end{cases} \tag{3.1}$$

where $g(i)$ is the grey-level of the pixel i. Here we are raster scanning images column by column to obtain the vector representations of the images. Starting with the identities $f = P_\Lambda f + (I - P_\Lambda)f$ and $f = \mathcal{A}^*\mathcal{A}f$, we have

$$f = P_\Lambda f + (I - P_\Lambda)\mathcal{A}^*\mathcal{A}f. \tag{3.2}$$

Given a non-negative vector u and a vector w, we define the thresholding operator D_u as follows

$$\mathcal{D}_u w := \left[t_{u(0)}(w(0)), t_{u(1)}(w(1)), \ldots\right]^t, \tag{3.3}$$

where $t_\lambda(x) := \text{sgn}(x)\max(|x| - \lambda, 0)$ is the soft thresholding function [15]. The framelet-based algorithm in [4, 10] is as follows:

$$f^{k+1} = P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\mathcal{D}_u\mathcal{A}f^k, \qquad k = 0, 1, \cdots, \tag{3.4}$$

where $f^0 := g$. We remark that a similar algorithm was also proposed in [19, 20] when $\mathcal{A}$ is orthonormal or a union of wavelet systems.

The thresholding operator D_u plays two roles. On the one hand, it removes noise. On the other hand, it gives a small perturbation of the frame coefficients $\mathcal{A}f^k$, which makes it possible to permeate information contained in the given data to those places where the data is missing. As we know, in the identity $f^k = \mathcal{A}^*\mathcal{A}f^k$, $\mathcal{A}f^k$ is not only the vector of frame coefficients of f^k under the system $\mathcal{A}$, but it also has the smallest ℓ_2 norm among those vectors which can perfectly represent f^k (see e.g. [14]). In this sense, at each iteration, the algorithm automatically has a built-in regularization effect which minimizes the ℓ_2 norm of the frame coefficients of f^k among all the choices. We make the claim more precise below.

Let $\alpha^k := \mathcal{D}_u\mathcal{A}f^k$. In [4], we proved that (3.4) can be viewed as a proximal forward-backward splitting (PFBS) iteration on α^k. Moreover, using the convergence theory for PFBS in [13], we have shown that (3.4) converges, and the sequence α^k converges to a minimizer of

$$\min_\alpha \left\{\frac{1}{2}\|P_\Lambda(f^0 - \mathcal{A}^*\alpha)\|^2 + \frac{1}{2}\|(I - \mathcal{A}\mathcal{A}^*)\alpha\|^2 + \|\text{diag}(u)\alpha\|_1\right\}. \tag{3.5}$$

Note that

- the term $\|P_\Lambda(f^0 - \mathcal{A}^*\alpha)\|^2$ measures the closeness of the recovered image to the given data,
- the term $\|(I - \mathcal{A}\mathcal{A}^*)\alpha\|^2$ is the distance of α to the range of $\mathcal{A}$, and
- the term $\|\text{diag}(u)\alpha\|_1$ is related to the sparsity of the solution α.

Thus we see that α^k is in fact minimizing a regularized functional.

As shown in [4, 10], the algorithm in (3.4) performs well in our numerical experiments. However, the convergence rate of the algorithm is not available and cannot be derived using the convergence theory for PFBS in [13]. The goal of this paper is to modify this algorithm to a new algorithm that has a convergence rate.

For this, we first rewrite (3.4). We partition the non-negative vector u in (3.4) as

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{3.6}$$

where u_1 is a zero vector with size the same as the number of rows of $\mathcal{L}_L$, and the size of u_2 equals the number of rows of $\mathcal{A}_L$. If we define the thresholding operator $\mathcal{T}$ as

$$\mathcal{T}\mathcal{A}_L v := \mathcal{D}_{u_2}(\mathcal{A}_L v), \tag{3.7}$$

(i.e., we do not threshold the low-frequency coefficients), then (3.4) becomes

$$f^{k+1} = P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\begin{bmatrix} \mathcal{L}_L f^k \\ \mathcal{T}(\mathcal{A}_L f^k) \end{bmatrix}, \qquad k = 0, 1, \cdots. \tag{3.8}$$

In [4], we proved that the sequence $\alpha^k = \begin{bmatrix} \mathcal{L}_L f^k \\ \mathcal{T}(\mathcal{A}_L f^k) \end{bmatrix}$ converges to a minimizer for (3.5).

The main change in our new algorithm is that we fix the low-frequency coefficients of each iteration in (3.8) to be those of the initial guess. This leads to the following new algorithm.

Algorithm 1 (Missing Data Recovering Algorithm)
Let $f_{|\Lambda}$ be given and f^0 be the initial guess satisfying $P_\Lambda f^0 = f_{|\Lambda}$. Define

$$f^{k+1} = P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}(\mathcal{A}_L f^k) \end{bmatrix}, \qquad k = 0, 1, \cdots . \tag{3.9}$$

The initial seed f^0 is normally chosen by spline interpolation, e.g., cubic spline interpolation or as the given data for the simplicity. Since spline interpolation provides a good approximation for smooth functions, the low frequency component $\mathcal{L}_L f^0$ of f^0 approximates well to the low frequency content of the original image. Iterative algorithm (3.9) is essentially the same as (3.8). The only difference is that the term $\mathcal{L}_L f^k$ in (3.8) is replaced by $\mathcal{L}_L f^0$ in (3.9). When L is sufficiently large, $\mathcal{L}_L$ is a long low-pass filter. Hence, both the difference between $\mathcal{L}_L f^0$ and $\mathcal{L}_L f^k$ contains mainly low frequency content of the original image. In the next section, we will see that by fixing $\mathcal{L}_L f^0$, we can prove the convergence of the new algorithm and obtain its convergence rate. Our convergence proof here is different from that given in [4] for (3.8). In fact, we will prove that the framelet coefficient $\beta = \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix}$, where v is the limit of f^k of (3.9), is a minimizer of

$$\min_{\alpha_1 = \mathcal{L}_L f^0} \left\{ \frac{1}{2}\|P_\Lambda(f^0 - \mathcal{A}^*\alpha)\|^2 + \frac{1}{2}\|(I - \mathcal{A}\mathcal{A}^*)\alpha\|^2 + \|\mathrm{diag}(u)\alpha\|_1 \right\}, \tag{3.10}$$

where α_1 denotes the partition of α according to $\mathcal{L}_L$. Clearly (3.10) is a constrained version of (3.5), and hence (3.9) is actually solving a constrained minimization problem. A different initial seed f^0 leads to a different constraint in the above minimization problem, hence, gives a different solution.

4 Analysis of Algorithm 1

This section consists of three parts. In the first part, we prove that Algorithm 1 converges, and then we obtain its convergence factor. In the second part, we derive the minimization properties satisfied by the limit of Algorithm 1. In the last part, we explain the role of the redundance of the tight frame system in the algorithm.

4.1 Convergence

To prove the convergence of Algorithm 1, we need the contracting property of the thresholding operator defined in (3.7). This contracting property as stated in the following lemma is crucial in the proof of the convergence and in deriving the optimal properties of the algorithm.

Lemma 4.1 *For any two vectors v_1 and v_2, we have*

$$\|\mathcal{T}\mathcal{A}_L(v_1) - \mathcal{T}\mathcal{A}_L(v_2)\| \le \|v_1 - v_2\|.$$

In particular, $\mathcal{T}\mathcal{A}_L$ is a continuous map.

Proof: Due to $|t_\lambda(x) - t_\lambda(y)| \le |x - y|$ for arbitrary real numbers x and y, and $\|\mathcal{A}_L v\| \le \|v\|$ for any vector v, we have, for any two vectors v_1 and v_2,

$$\|\mathcal{T}\mathcal{A}_L v_1 - \mathcal{T}\mathcal{A}_L v_2\| \le \|\mathcal{A}_L v_1 - \mathcal{A}_L v_2\| \le \|v_1 - v_2\|.$$

□

Theorem 4.2 *Algorithm 1 converges with a convergence factor $\rho = 1 - \mu^2$ whenever the smallest singular value μ of $\mathcal{L}_L$ defined in (2.9) satisfies $\mu > 0$. In particular, the limit v satisfies*

$$v = P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix}. \tag{4.1}$$

Furthermore, the limit is unique.

Proof: The idea of the proof is to show that the sequence f^k is a Cauchy sequence by using Lemma 4.1. Let m, n be arbitrary two integers, then,

$$\begin{aligned} \|f^{n+m} - f^n\| &\le \|\mathcal{A}_L^*(\mathcal{T}\mathcal{A}_L f^{n+m-1} - \mathcal{T}\mathcal{A}_L f^{n-1})\| \le \|\mathcal{A}_L^*\|\|\mathcal{T}\mathcal{A}_L f^{n+m-1} - \mathcal{T}\mathcal{A}_L f^{n-1}\| \\ &\le \|\mathcal{A}_L^*\|\|\mathcal{A}_L f^{n+m-1} - \mathcal{A}_L f^{n-1}\| \le \|\mathcal{A}_L^*\|\|\mathcal{A}_L\|\|f^{n+m-1} - f^{n-1}\|. \end{aligned}$$

Since $\mu > 0$, we conclude that the largest eigenvalue of the matrix $\sum_{\ell=1}^{L} \mathcal{H}_\ell^* \mathcal{H}_\ell$ is strictly smaller than 1. Furthermore,

$$\|\mathcal{A}_L^*\| = \| \begin{bmatrix} \mathcal{H}_1^* & \mathcal{H}_2^* & \dots & \mathcal{H}_L^* \end{bmatrix} \| = \sqrt{1 - \mu^2}.$$

Hence, we have

$$\begin{aligned} \|f^{n+m} - f^n\| &\le (1 - \mu^2)\|f^{n+m-1} - f^{n-1}\| \\ &\le (1 - \mu^2)^n \|f^m - f^0\| \\ &\le (1 - \mu^2)^n (\|f^m\| + \|f^0\|). \end{aligned}$$

Next we show that f^m is uniformly bounded. In fact, for all integer $m > 0$, by (3.9),

$$\begin{aligned} \|f^m\| &\le 2\|f^0\| + (1 - \mu^2)\|f^{m-1}\| \\ &\le 2[1 + (1 - \mu^2) + (1 - \mu^2)^2 + \cdots + (1 - \mu^2)^m]\|f^0\| \le \frac{2}{\mu^2}\|f^0\|. \end{aligned}$$

Thus $\|f^{n+m} - f^n\| \to 0$ as $n, m \to \infty$. Hence the sequence f^k converges. Finally, (4.1) follows from the continuity of $\mathcal{T}\mathcal{A}_L$ by Lemma 4.1.

Next, we prove the uniqueness of our limit for any arbitrary initial guess. Suppose that v_1 and v_2 are two limits. We have

$$\begin{aligned} \|v_1 - v_2\| &\le \|\mathcal{A}_L^*(\mathcal{T}\mathcal{A}_L v_1 - \mathcal{T}\mathcal{A}_L v_2)\| \le \|\mathcal{A}_L^*\|\|\mathcal{T}\mathcal{A}_L v_1 - \mathcal{T}\mathcal{A}_L v_2\| \\ &\le \|\mathcal{A}_L^*\|\|\mathcal{A}_L v_1 - \mathcal{A}_L v_2\| \le \|\mathcal{A}_L^*\|\|\mathcal{A}_L\|\|v_1 - v_2\|. \end{aligned}$$

Since $\|\mathcal{A}_L^*\| = \sqrt{1 - \mu^2} < 1$, we know that $\|v_1 - v_2\| = 0$, i.e., $v_1 = v_2$. The convergence factor ρ can be obtained by

$$\begin{aligned} \|f^{n+1} - v\| &\le \|\mathcal{A}_L^*(\mathcal{T}\mathcal{A}_L f^n - \mathcal{T}\mathcal{A}_L v)\| \le \|\mathcal{A}_L^*\|\|\mathcal{T}\mathcal{A}_L f^n - \mathcal{T}\mathcal{A}_L v\| \\ &\le \|\mathcal{A}_L^*\|\|\mathcal{A}_L f^n - \mathcal{A}_L v\| \le \|\mathcal{A}_L^*\|\|\mathcal{A}_L\|\|f^n - v\|. \end{aligned}$$

Therefore, $\rho = \|\mathcal{A}_L^*\|\|\mathcal{A}_L\| = 1 - \mu^2$. □

Next we show that the smallest singular value μ of $\mathcal{L}_L$ is indeed greater than zero if we choose N properly. To do this, we review a remarkable property of the matrix H in (2.5) if the sequence h is symmetric, i.e., $h(j) = h(-j)$, $j = 0, 1, \ldots, m$. It was stated in [24] that the matrix H in (2.5) can always be diagonalized by the discrete cosine transform (DCT) and the corresponding eigenvectors of H are simply the DCT of the first column of the matrix H if the sequence h is symmetric. More precisely, let C be the N-by-N discrete cosine transform matrix. Its entries are

$$[C]_{p,q} = \sqrt{\frac{2-\delta_{p1}}{N}} \cos\left(\frac{(p-1)(2q-1)\pi}{2N}\right), \quad 1 \le p, q \le N, \tag{4.2}$$

where δ_{pq} is the Kronecker delta. We note that C is orthogonal, i.e., $C^tC = CC^t = I$. It is proven in [24] that $H = C^t\Gamma C$, where Γ is the diagonal matrix containing the eigenvalues of H. By multiplying $e_1 = (1, 0, \ldots, 0)^t$ to both sides of $CH = \Gamma C$, we see that the eigenvalues γ_p of H are given by

$$\gamma_p = \frac{[C(He_1)]_{p+1}}{[Ce_1]_{p+1}}, \quad 0 \le p < N. \tag{4.3}$$

Using (4.2), (4.3) reduces to:

$$\gamma_p = \frac{\sum_{q=1}^{N} [He_1]_q \cos[(2q-1)\theta_p]}{\cos\theta_p}, \quad 0 \le p < N, \tag{4.4}$$

where $\theta_p = p\pi/(2N)$, i.e. $0 \le \theta_p < \pi/2$.

For the symmetric sequence h given in (2.4), the first column of H is given by:

$$[h(0) + h(1), h(1) + h(2), \ldots, h(m-1) + h(m), h(m), 0, \ldots, 0]^t.$$

Hence,

$$\begin{aligned} \sum_{q=1}^{N} [He_1]_q \cos[(2q-1)\theta] &= \sum_{q=1}^{m+1} h(q-1)\cos[(2q-1)\theta] + \sum_{q=1}^{m} h(q)\cos[(2q-1)\theta] \\ &= \cos\theta \sum_{q=-m}^{m} h(q) e^{-i2q\theta}. \end{aligned}$$

Therefore,

$$\gamma_p = \sum_{q=-m}^{m} h(q) e^{-i2q\theta_p}, \quad 0 \le p < N.$$

Immediately, we have the following result:

Proposition 4.3 *Let $H_0^{(\ell)}$ be the matrix representation of the filter formed from the low-pass filter h_0 of the $2m$-th spline by inserting $2^{\ell-1} - 1$ zeros between every two adjacent components of h_0. Then the matrix $H_0^{(\ell)}$ has eigenvalues $\gamma_p^{(\ell)} = \cos^{2m}(2^{\ell-1}\theta_p)$, where $\theta_p = \frac{p\pi}{2N}$ for $0 \le p < N$. In particular, for $\mathcal{L}_L$ defined in (2.9), the smallest eigenvalue μ^2 of $\mathcal{L}_L^*\mathcal{L}_L$ is positive if $2^{L-1}p$ is not divisible by N for all $1 \le p < N$.*

Proof: Since h_0 is the low-pass filter of the $2m$-th spline whose symbol $\widehat{h}_0$ is given in (2.1), we have

$$\sum_{q=-m}^{m} h_0(q)e^{-iq\omega} = \cos^{2m}\left(\frac{\omega}{2}\right).$$

It follows that the symmetric matrix $H_0^{(\ell)}$ has eigenvalues $\gamma_{\ell,p} = \cos^{2m}(2^{\ell-1}\theta_p)$, $0 \le p < N$. Since $H_0^{(\ell)}$, $1 \le \ell \le L$, can be diagonalized the DCT, so does the matrix $\mathcal{L}_L$ in (2.6). Furthermore, the matrix $\mathcal{L}_L$ has eigenvalues

$$\gamma_p := \gamma_{L,p}\gamma_{L-1,p}\cdots\gamma_{1,p} = \cos^{2m}(2^{L-1}\theta_p)\cos^{2m}(2^{L-2}\theta_p)\cdots\cos^{2m}(2^0\theta_p) = \left(\frac{\sin(2^L\theta_p)}{2^L\sin\theta_p}\right)^{2m},$$

where $0 \le p < N$. Clearly, $\gamma_0 = 1$. Note that $\sin\theta_p \neq 0$ for all $1 \le p \le N-1$. Hence, for a given integer p between 1 and $N-1$, γ_p being zero happens only when $\sin(2^L\theta_p) = 0$. If so, $2^L\theta_p$ is a multiple of π, that is, $\frac{2^L p}{2N}$ must be an integer since $\theta_p = \frac{p\pi}{2N}$. In other words, $2^{L-1}p$ is divisible by N. Thus, $\sin(2^L\theta_p) \neq 0$ if $2^{L-1}p$ are not divisible by N for all $1 \le p \le N-1$. Therefore γ_p, $0 \le p < N$, are all positive. For $\mathcal{L}_L$ defined in (2.9), because of its tensor product definition, all its eigenvalues are of the form $\gamma_p\gamma_q$ with $0 \le p, q < N$. Hence they are all positive too under the same assumption. In particular, the smallest eigenvalue of $\mathcal{L}_L^*\mathcal{L}_L$ is positive too. □

Thus we see that by choosing the size of the image N appropriately, the assumption of Theorem 4.2 can be satisfied and Algorithm 1 converges. For example, we will use $N = 255$ and $N = 511$ in our experiments. Finally, we note that one can always enlarge the image size N by using, for example, symmetric extension such that N satisfies the assumption in Proposition 4.3.

4.2 Minimization

Next we study the minimization properties of the limit of Algorithm 1. Define a new sequence α^k in the framelet domain by

$$\alpha^k := \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L f^k \end{bmatrix}, \qquad k = 1, 2, \cdots. \tag{4.5}$$

In terms of α^k, Algorithm 1 can be rewritten as

$$\alpha^{k+1} = \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\alpha^k\big) \end{bmatrix}, \qquad k = 0, 1, \cdots. \tag{4.6}$$

By Lemma 4.1, the mapping from f^k to α^k is continuous. It, together with Theorem 4.2 which states that f^k converges, implies that α^k converges. Denote the limit

$$\beta := \lim_{k\to\infty} \alpha^k. \tag{4.7}$$

Due to Lemma 4.1, the mapping from α^k to α^{k+1} is also continuous. Therefore, the limit β satisfies

$$\beta = \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big) \end{bmatrix}. \tag{4.8}$$

Moreover, the limit is unique.

For any framelet coefficient vector $\alpha \in \mathbb{R}^K$, we partition it as follows

$$\alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix},$$

where α_1 is the framelet coefficients corresponding to $\mathcal{L}_L$, and α_2 is those corresponding to $\mathcal{A}_L$. We will show in the following that β is a minimizer of the minimization problem

$$\min_{\alpha_1=\mathcal{L}_L f^0} F(\alpha), \tag{4.9}$$

where

$$F(\alpha) = \frac{1}{2}\|P_\Lambda(f^0 - \mathcal{A}^*\alpha)\|^2 + \frac{1}{2}\|(I - \mathcal{A}\mathcal{A}^*)\alpha\|^2 + \|\mathrm{diag}(u)\alpha\|_1 \tag{4.10}$$

with u being defined in (3.6). Comparing (4.9) with (3.5), the latter is an unconstrained version of the former.

In order to prove that β is a minimizer for (4.9), we need some notations in convex analysis. We only consider the case of finite dimensional Euclidean space $\mathbb{R}^K$ having the Euclidean inner product. For any proper, lower semi-continuous and convex function $\varphi(x)$, where $x \in \mathbb{R}^K$, its *subdifferential* is defined by

$$\partial\varphi(x) := \left\{w \in \mathbb{R}^K \mid \forall y \in \mathbb{R}^K, \langle y - x, w\rangle + \varphi(x) \le \varphi(y)\right\}.$$

When $\varphi(x)$ is differentiable, the subdifferential is the ordinary gradient. According to the Fermat's rule,

$$\varphi(x) = \inf_{y\in\mathbb{R}^K} \varphi(y) \quad \Longleftrightarrow \quad 0 \in \partial\varphi(x). \tag{4.11}$$

The proximity operator of φ is defined by

$$\mathrm{prox}_\varphi(x) := \arg\min_{y\in\mathbb{R}^K} \left\{\frac{1}{2}\|x - y\|^2 + \varphi(y)\right\}.$$

For any convex set $\Gamma \subset \mathbb{R}^K$, the indicator function is defined by

$$\iota_\Gamma(x) := \begin{cases} 0, & \text{if } x \in \Gamma, \\ +\infty, & \text{otherwise.} \end{cases} \tag{4.12}$$

It is shown (see (2.8) in [13]) that

$$\partial\iota_\Gamma(x) = \begin{cases} \{w \in \mathbb{R}^K \mid \forall y \in \Gamma, \langle y - x, w\rangle \le 0\}, & \text{if } x \in \Gamma, \\ \emptyset, & \text{otherwise.} \end{cases} \tag{4.13}$$

We need the following lemmas.

Lemma 4.4 *For any $y \in \mathbb{R}^K$, the thresholding operator $\mathcal{T}$ satisfies*

$$y_2 - \mathcal{T}(y_2) \in \partial\|\mathrm{diag}(u_2)\mathcal{T}(y_2)\|_1.$$

Proof: It follows, for example, from [13] that

$$t_\lambda(a) = \arg\min_{b\in\mathbb{R}} \left\{\frac{1}{2}(a - b)^2 + \lambda|b|\right\}.$$

Therefore, by the definition of $\mathcal{T}(y_2)$ and (3.3), we obtain

$$\mathcal{T}(y_2) = \arg\min_{x\in\mathbb{R}^{K_2}} \left\{\frac{1}{2}\|y_2 - x\|^2 + \|\mathrm{diag}(u_2)x\|_1\right\},$$

where K_2 is the number of rows in $\mathcal{A}_L$. By (4.11), the above equality is equivalent to

$$0 \quad \in \quad \partial\left\{\frac{1}{2}\|y_2 - x\|^2 + \|\mathrm{diag}(u)x\|_1\right\}\Big|_{x=\mathcal{T}(y_2)} = \mathcal{T}(y_2) - y_2 + \partial\|\mathrm{diag}(u_2)\mathcal{T}(y_2)\|_1.$$

□

Lemma 4.5 *Let* $\Gamma = \{\alpha \in \mathbb{R}^K \mid \alpha_1 = \mathcal{L}_L f^0\}$. *Then,*

$$\partial\iota_\Gamma(x) = \begin{cases} \{w \in \mathbb{R}^K \mid w_2 = 0\}, & \text{if } x \in \Gamma, \\ \emptyset, & \text{otherwise.} \end{cases}$$

Proof: By (4.13), it is obvious that $\partial\iota_\Gamma(x) = \emptyset$ when $x \notin \Gamma$. We only consider the case when $x \in \Gamma$. Let $\Delta = \{w \in \mathbb{R}^K \mid w_2 = 0\}$. First we prove that $\Delta \subseteq \partial\iota_\Gamma(x)$. For any $y \in \Gamma$, since $x \in \Gamma$, we have $(y - x)_1 = 0$. Therefore, for any $z \in \Delta$, $\langle y - x, z\rangle = 0$. Hence, $z \in \partial\iota_\Gamma(x)$.

Next we show that $\partial\iota_\Gamma \subseteq \Delta$ by showing $\Delta^c \subseteq (\partial\iota_\Gamma)^c$, where the superscript c denotes the set complement. For any $z \notin \Delta$, there exists an index i_0 corresponding to $\mathcal{A}_L$ such that $z(i_0) \neq 0$. For $x \in \Gamma$ we can find a $y \in \Gamma$ such that $y(i_0) - x(i_0) = z(i_0)$ and $y(i) - x(i) = 0$ for $i \neq i_0$. Therefore, $\langle y - x, z\rangle = z(i_0)^2 > 0$. Hence, $z \notin \partial\iota_\Gamma$. □

Now we can prove the following result.

Theorem 4.6 *The limit* β *of* α^k *defined in* (4.7) *is a minimizer for* (4.9).

Proof: Let $\Gamma = \{\alpha \in \mathbb{R}^K \mid \alpha_1 = \mathcal{L}_L f^0\}$. By (4.12) and (4.10), the minimization (4.9) is equivalent to the unconstrained minimization $\min_\alpha G(\alpha)$, where

$$G(\alpha) = \frac{1}{2}\|P_\Lambda(f^0 - \mathcal{A}^*\alpha)\|^2 + \frac{1}{2}\|(I - \mathcal{A}\mathcal{A}^*)\alpha\|^2 + \|\mathrm{diag}(u)\alpha_2\|_1 + \iota_\Gamma(\alpha).$$

To prove the theorem, by (4.11) we only need to show that $0 \in \partial G(\beta)$. We note that

$$\partial G(\beta) = \mathcal{A}P_\Lambda(\mathcal{A}^*\beta - f^0) + (I - \mathcal{A}\mathcal{A}^*)\beta + \begin{bmatrix} 0 \\ \partial\|\mathrm{diag}(u_2)\beta_2\|_1 \end{bmatrix} + \partial\iota_\Gamma(\beta). \tag{4.14}$$

By (4.7),

$$\beta_1 = \mathcal{L}_L f^0 \quad \text{and} \quad \beta_2 = \mathcal{T}\mathcal{A}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big). \tag{4.15}$$

It implies that $\beta \in \Gamma$, and hence by Lemma 4.5

$$\partial\iota_\Gamma(\beta) = \begin{bmatrix} w_1 \\ 0 \end{bmatrix}, \tag{4.16}$$

where w_1 is an arbitrary vector of length equal to the number of rows in $\mathcal{L}_L$. On the other hand, the expression of β_2 in (4.15) and Lemma 4.4 imply that

$$\mathcal{A}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big) - \beta_2 \in \partial\|\mathrm{diag}(u_2)\beta_2\|_1. \tag{4.17}$$

Substituting (4.16) and (4.17) into (4.14), we obtain

$$\begin{bmatrix} w_1 \\ \mathcal{A}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big) - \beta_2 \end{bmatrix} + \mathcal{A}P_\Lambda(\mathcal{A}^*\beta - f^0) + (I - \mathcal{A}\mathcal{A}^*)\beta \in \partial G(\beta). \tag{4.18}$$

Since w_1 can be arbitrary, we let

$$w_1 = \mathcal{L}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big) - \beta_1.$$

Then the left hand side of (4.18) is

$$\begin{aligned} & \begin{bmatrix} \mathcal{L}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big) - \beta_1 \\ \mathcal{A}_L\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big) - \beta_2 \end{bmatrix} + \mathcal{A}P_\Lambda(\mathcal{A}^*\beta - f^0) + (I - \mathcal{A}\mathcal{A}^*)\beta \\ = \;& \mathcal{A}\big(P_\Lambda f^0 + (I - P_\Lambda)\mathcal{A}^*\beta\big) - \beta + \mathcal{A}P_\Lambda(\mathcal{A}^*\beta - f^0) + (I - \mathcal{A}\mathcal{A}^*)\beta = 0, \end{aligned}$$

i.e., we have proved $0 \in \partial G(\beta)$. □

4.3 A Role of Redundance

Finally, we discuss what roles the redundance of the tight frames play in reducing the jumps around the boundary of Λ of the algorithm. Notice that the limiting vector v satisfies

$$P_\Lambda v = P_\Lambda f^0 = f_{|_\Lambda}$$

and

$$(I - P_\Lambda)\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L v \\ \mathcal{A}_L v \end{bmatrix} = (I - P_\Lambda)\mathcal{A}^*\mathcal{A}v = (I - P_\Lambda)v = (I - P_\Lambda)\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix}. \tag{4.19}$$

Since v is obtained by replacing $P_\Lambda\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix}$ by $P_\Lambda\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L v \\ \mathcal{A}_L v \end{bmatrix} = P_\Lambda f^0 = f_{|_\Lambda}$ as shown in (4.1), to reduce the jumps around the the boundary of Λ, we require that the norm of

$$P_\Lambda\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L v \\ \mathcal{A}_L v \end{bmatrix} - P_\Lambda\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix} = P_\Lambda f^0 - P_\Lambda\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix}$$

to be small. The larger difference in the norm will lead to bigger jumps around the boundary of Λ. This, together with (4.19), means that the norm of

$$\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L v \\ \mathcal{A}_L v \end{bmatrix} - \mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix} = v - \mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix}$$

should be small. However, this is determined by

$$e = \begin{bmatrix} \mathcal{L}_L v \\ \mathcal{A}_L v \end{bmatrix} - \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix} = \mathcal{A}v - \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix}.$$

What does the redundancy bring us here? Apparently, $e \neq 0$ and

$$\mathcal{A}^* \begin{bmatrix} \mathcal{L}_L v \\ \mathcal{A}_L v \end{bmatrix} - \mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix} = v - \mathcal{A}^* \begin{bmatrix} \mathcal{L}_L f^0 \\ \mathcal{T}\mathcal{A}_L v \end{bmatrix} = \mathcal{A}^* e.$$

Since $\mathcal{A}^*$ projects the sequence down to the orthogonal complement of $\ker \mathcal{A}^*$ which is the range of $\mathcal{A}$, the component of e in the kernel of $\mathcal{A}^*$ does not contribute. The redundant system reduces the errors as long as the component of e in the kernel of $\mathcal{A}^*$ is not zero. Since larger the kernel of $\mathcal{A}^*$ is, the more redundant the frame is; and the higher the redundancy is, the more the errors reduces in general. The smaller norm of $\mathcal{A}^* e$ implies smaller jumps around the boundary of Λ. In contrast, if $\mathcal{A}$ is not a redundant system but an orthonormal system, then $\ker \mathcal{A}^*$ is $\{0\}$. In this case, $\|\mathcal{A}^* e\| = \|e\|$.

5 Framelets-based Impulse Noise Removal

In this section, we apply Algorithm 1 to the problem of impulse noise removal. First, we briefly review impulse noise models. Then we give our impulse noise removal algorithms.

Let f be the original image defined on the set Ω and g the noisy image corrupted by impulse noise. Impulse noise can be modeled as

$$g(i,j) = \begin{cases} s(i,j) & \text{with probability } r, \\ f(i,j), & \text{with probability } 1-r, \end{cases}$$

where $s(i,j) = 0$ or 255 for the *salt-pepper noise*, and $s(i,j)$ is a uniformly-distributed random number in $[0, 255]$ for the *random-valued impulse noise*, see [18]. The number r is called the *noise level*. There are two popular types of filtering methods for impulse noise removal. One is the median filter and its variants (see [1, 12, 17, 21, 22, 26, 28, 29, 30]); another one is the variational approach (see [7, 25]).

By the very nature of impulse noise, many existing techniques for impulse noise removal have two stages. The first stage is to detect the pixels that are corrupted by impulse noises and label them, and the second stage is to replace those labeled by some schemes while leaving the unlabeled pixels unchanged, see [12, 18, 22]. Two methods, namely, adaptive median filter (AMF) [21] and adaptive center-weighted median filter (ACWMF) [12], will be used in this paper for detecting and labeling the salt-pepper impulse noise and the random-valued impulse noise, respectively. They provide a good tradeoff between computational complexity and robust noise detection even for high noise levels. For completeness, the algorithms of AMF and ACWMF are reviewed in the appendix. For more details, the interested readers should consult [21, 22].

Recently, a powerful two-phase scheme for impulse noise removal was developed in [7, 8]. In the first phase, a noise detector, such as AMF or ACWMF method, are applied to g to determine a set $\mathcal{N}$ which contains the indices of the noise candidates. In the second phase, variational methods constrained on $\mathcal{N}$ are performed to restore pixels on $\mathcal{N}$. We note that once $\mathcal{N}$ is obtained in the first stage, the second stage is like doing an inpainting on the set $\Lambda = \Omega \setminus \mathcal{N}$. Hence we can apply Algorithm 1 to solve this problem.

To use Algorithm 1, we have to define the non-negative vector u_2 in (3.6), whose entries are the thresholding parameters. They are defined as follows. All entries of u_2 corresponding to the matrix $G_i^{(k)}\mathcal{L}_{k-1}$ in (2.9) are the same and equal to a factor of $2^{1-k}T$, where T is a parameter independent of k. Specifically, for the cubic tight frame ($m = 2$), we define

$$[\kappa_0, \kappa_1, \kappa_2, \kappa_3, \kappa_4] = [1, \frac{3}{4}, \frac{\sqrt{6}}{4}, \frac{3}{4}, 1].$$

Every entry of u_2 corresponding to the matrix $G_{5i+j}^{(k)}\mathcal{L}_{k-1}$ is $\kappa_i\kappa_j 2^{1-k}T$ for all possible i, j, and k.

The strategy of selecting the parameter T is as follows. It is well known that the large framelet coefficients keep the edges and details of the underlying image. The amplitudes of the framelet coefficients in the neighborhood of $\mathcal{N}$ are relatively large, partially because it contains the information of the missing data and it is also due to the artifacts created by the missing data. If the parameter T is too small, most of these coefficients will remain unchanged and the artifacts will not be removed. If the parameter T is too large, many small features of the image will disappear. To resolve this, we apply Algorithm 1 several times with decreasing T's. We propose the following two algorithms for the two different noise models.

Algorithm 2 (Salt-Pepper Noise Removal) *1. Apply AMF to the given noisy image g to detect the pixel set $\mathcal{N}$ and obtain a denoised image $f^{(0)}$. Use $f^{(0)}$ as the initial guess and denote $\Lambda = \Omega \setminus \mathcal{N}$. Set $f_\Lambda = g_\Lambda$.*

2. For $j = 0, 1, \dots J$:

Perform Algorithm 1 with initial image $f^{(j)}$ and parameter $T = 2^{J-j}$. The output of Algorithm 1 is denoted by $f^{(j+1)}$.

In the implementation, we choose $J = 5$.

For random-valued impulse noise, because they are more difficult to detect, we use the idea developed in [8]. Namely, in each iteration, we pass the iterate through a median-type filter to detect a new noise set $\mathcal{N}$. Then we have the following modified algorithm.

Algorithm 3 (Random-Valued Impulse Noise Removal)

1. For $k = 0$ and $f^{(0)} = g$, where g is the given noisy image.

2. For $k = 1, 2, \dots, k_{\max}$:

(a) Apply ACWMF to image $f^{(k-1)}$ to detect a noisy pixel set $\mathcal{N}_{k-1}$ and use ACWMF to obtain a denoised image. Denote this denoised image still by $f^{(k-1)}$. Set $\Lambda_{k-1} = \cap_{j=0}^{k-1} \mathcal{N}_j^c$.

(b) Apply Item 2 of Algorithm 2 with $J = 4$, the initial image $f^{(k-1)}$, and the noise pixels set $\cup_{j=0}^{k-1} \mathcal{N}_j$. The output is denoted as $f^{(k)}$.

In the implementation, we choose $k_{\max} = 4$.

6 Numerical Experiments

In this section, we test the efficiency of Algorithms 2 and 3 on three images: "Bridge"of size 511×511, "Cameraman", and "Goldhill" of size 255×255, see Figure 1. The frame system $\mathcal{A}$ we used is the piecewise cubic tight framelet system with level $L = 6$. Algorithm 1 is stopped when the relative error is less than 10^{-4} or the number of iterations exceeds 30. We compare our algorithms with the filtering methods and variational methods. The quality of the restored images are measured quantitatively by the peak signal-to-noise ratio (PSNR) defined as follows:

$$\text{PSNR} \quad = \quad 10 \log_{10} \frac{255^2}{\frac{1}{MN} \sum_{s,t} (\widetilde{f}(s,t) - f(s,t))^2}$$

where $\widetilde{f}(s,t)$ and $f(s,t)$ denote the pixel values of the restored image and the original image respectively. Each PSNR value reported in Tables 1–2 is averaged over five runs.

6.1 Results for Salt-Pepper Noise

In our simulation, the three test images were corrupted with *"pepper"* (pixel value 0) and *"salt"* (pixel value 255) noise with noise level varying from 50% to 90%. For AMF, the maximum window size $w_{\max}$ is set to 39, see Appendix. We compare our results with the two-phase variational method in [7], where its solutions are computed by the conjugate gradient method in [3].

Figure 1: Original images: "Bridge", "Cameraman", and "Goldhill".

The PSNR of the restored images are tabulated in Table 1. Our Algorithm 2 provides significant improvement over AMF for all test images at all noise levels. There are also moderate improvements over the variational method [7] for all tests except one. In addition to the gains in PSNR, Algorithm 2 also provides noticeable better visual edge integrity and noise reduction as compared with AMF and the variational approach, especially when the noise level is high. This point is illustrated in Figures 2 and 3 for images corrupted with 70% and 90% salt-pepper noise.

Table 1: PSNRs values after applying AMF, two-phase variational method [7], and Algorithm 2 to images corrupted by salt-pepper impulse noise of various noise levels.

Image	Noise Rate	AMF	2-Phase Variational Method	Algorithm 2
Bridge	50%	24.34	27.41	27.22
	70%	21.70	24.87	24.98
	90%	18.25	21.57	21.85
Cameraman	50%	24.07	27.37	28.31
	70%	21.26	24.70	25.40
	90%	17.64	20.98	21.58
Goldhill	50%	26.02	29.42	29.47
	70%	23.47	26.93	27.10
	90%	20.06	23.53	23.82

6.2 Results for Random-Valued Impulse Noise

Here we test images corrupted with random-valued impulse noise of noise level from 30% to 50%. The parameter s in ACWMF is varying for different images and noise levels, see Step 3 of the algorithm ACWMF in Appendix. In our experiments, it is 0.5, 0.3, 0.2 for "Bridge" image of noise levels 30%, 40%, and 50% respectively, 0.5, 0.4, 0.3 for "Cameraman" image, and 0.2, 0.2, 0.1 for "Goldhill" image. Also since we are applying ACWMF for $k_{\max}$'s times in Algorithm 3, we systematically change the parameters $\delta_n^{(k)}$, $n = 0, 1, 2, 3$, in ACWMF for each iteration $k = 1, \ldots, k_{\max}$, see [23]. More precisely, we use

$$[\delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}, \delta_3^{(k)}] = [40, 25, 10, 5] + 20(3-k)[1, 1, 1, 1]$$

for $k \leq 3$ and

$$[\delta_0^{(k)}, \delta_1^{(k)}, \delta_2^{(k)}, \delta_3^{(k)}] = [40, 25, 10, 5]$$

for $k > 3$. Table 2 gives the PSNR of the restored images. We can see from it that our Algorithm 3 performs much better than ACWMF. It is also better than the two-phase variational method [8]. Moreover, noise patches produced by ACWMF are significantly reduced by our algorithm as shown in Figures 4 and 5.

Table 2: PSNR values after applying ACWMF, two-phase variational method in [8] and Algorithm 3 to images corrupted by random-valued impulse noise of various noise levels.

Image	Noise Rate	ACWMF	2-phase Variational Method	Algorithm 3
Bridge	30%	25.16	26.06	26.07
	40%	23.36	24.81	24.87
	50%	21.47	23.40	23.60
Cameraman	30%	24.36	24.88	24.95
	40%	22.57	23.66	23.87
	50%	20.61	22.35	22.65
Goldhill	30%	26.88	27.65	27.70
	40%	25.10	26.54	26.71
	50%	23.21	25.29	25.58

6.3 Results for Image Inpainting

To further demonstrate the efficiency of the proposed algorithm, we end the paper by presenting one more example of image inpainting. Image inpainting refers to the filling-in of missing data in digital images based on the information available in the observed region. Applications of this technique include film restoration, text or scratch removal, and digital zooming.

Let $\mathcal{N}$ be the set where original image data is missing. In the left images in the first and third rows of Figure 6, the set $\mathcal{N}$ is the location of the text imposed in the "Cameraman" and "Goldhill" images. We remove those text by using Algorithm 2 with $J = 6$. We present inpainted images for three different initial guess $f^{(0)}$. For the "Cameraman" image, the first chosen $f^{(0)}$ is the image to be inpainted itself which is the left image in the first row of Figure 6. The second chosen $f^{(0)}$ shown in the middle of the first row of Figure 6 is obtained by the cubic spline interpolation from the available information. The third chosen $f^{(0)}$ shown in the right of the first row of Figure 6 is the one whose pixel values in $\mathcal{N}$ are randomly between 0 and 255. The corresponding inpainted images (from left to right) are presented in the second row of Figure 6 with values of PSNR 30.64dB, 30.52dB, and 30.50dB, respectively. Similarly, for the three different initial guesses $f^{(0)}$ (from left to right) shown in the third row of Figure 6, the corresponding inpainted "Goldhill" images (from left to right) are presented in the last row with values of PSNR 32.62dB, 32.67dB, and 32.61dB, respectively. From above results, we can see that the proposed algorithm produces comparable results for different initial guesses in terms of the PSNR values and the visual quality. We note that the initial guesses generated by the cubic spline in the middle of the first row and the third row in Figure 6 have PSNR values 29.43dB and 31.24dB, respectively.

7 Conclusion

In this paper, we have modified the framelet-based algorithm in [4] to obtain a new algorithm which we can prove convergence and can derive its convergence rate. In addition, we showed that

the solution of the new algorithm is minimizing a constrained problem, where the unconstrained one was minimized by the framelet-based algorithm in [4]. Application of the algorithm in impulse noise removal shows that it is effective when compared to median-type filters [21, 22] and variational approaches in [3, 7, 8].

Appendix

We first introduce the adaptive median filter (AMF) method given in [21]. It is used to detect salt-pepper noise in our tests. Let $\mathcal{S}_{ij}^w$ be a window of size $w \times w$ centered at pixel location (i, j), i.e.,

$$\mathcal{S}_{ij}^w = \{(k, \ell) : |k - \ell| \leq w \quad \text{and} \quad |j - \ell| \leq w\}$$

and let $w_{\max}$-by-$w_{\max}$ be the maximum window size. The algorithm tries to identify the noise candidates $g(i, j)$ and then replace each $g(i, j)$ by the median of the pixels in $\mathcal{S}_{ij}^w$. AMF consists of the following five steps:

Algorithm 4 (AMF)

1. *Initialize* $w = 3$ *and* $\mathcal{N} = \emptyset$.
2. *Compute* $s_{ij}^{\min,w}$, $s_{ij}^{\text{med},w}$, *and* $s_{ij}^{\max,w}$, *which are the minimum, median, and maximum of the pixel values in* S_{ij}^w, *respectively.*
3. *If* $s_{ij}^{\min,w} < s_{ij}^{\text{med},w} < s_{ij}^{\max,w}$, *then go to Step 5. Otherwise, set* $w = w + 2$.
4. *If* $w \leq w_{\max}$, *go to Step 2. Otherwise, we replace* $g(i, j)$ *by* $s_{ij}^{\text{med},w_{\max}}$ *and add the location* (i, j) *into the noise set* $\mathcal{N}$.
5. *If* $s_{ij}^{\min,w} < g(i, j) < s_{ij}^{\max,w}$, *then* $g(i, j)$ *is not a noise candidate. Otherwise we replace* $g(i, j)$ *by* $s_{ij}^{\text{med},w_{\max}}$ *and add the location* (i, j) *to the noise set* $\mathcal{N}$.

The only required parameter in AMF is the maximum window size $w_{\max}$. The AMF starts with an initial window size $w = 3$ with a step of 2 to $w_{\max}$ and detects the impulse noise in an adaptive way. In this respect, maximum window size $w_{\max}$ should increase with the noise level. In our experiments, we set $w_{\max} = 39$.

Next we briefly describe the adaptive center-weight median filter (ACWMF) method in [22]. It is used to detect random-valued impulse noise in our tests. Let

$$Y_{i,j}^{w,r} = \text{median}\{g(s, t), r \diamond g(i, j) | (s, t) \in \mathcal{S}_{ij}^w\}$$

where the operator $\diamond$ denotes the repetition operation. In other words, $r \diamond g(i, j)$ means r copies of $g(i, j)$ in total. Clearly, $Y_{ij}^{w,1}$ is the output of the standard median filter. ACWMF has the following four steps:

Algorithm 5 (ACWMF)

1. *Calculate* $Y_{ij}^{3,1}$, $Y_{ij}^{3,3}$, $Y_{ij}^{3,5}$, *and* $Y_{ij}^{3,7}$.
2. *Compute the differences* $d_k = |Y_{ij}^{3,2k+1} - g(i, j)|$, *where* $k = 0, 1, 2, 3$. *It was shown in [12] that* $d_k \leq d_{k+1}$ *for* $k \geq 1$.

3. *Compute the four threshold* S_k, $k = 0, 1, 2, 3$, *as follows:*

$$S_k = s \cdot \text{MAD} + \delta_k$$

where

$$\text{MAD} = \text{median}\{|g(s,t) - Y_{ij}^{3,1}| : (s,t) \in \mathcal{S}_{ij}^3\}$$

with $\delta_0 = 40$, $\delta_1 = 25$, $\delta_2 = 10$, $\delta_3 = 5$ *and* $0 \le s \le 0.6$.

4. *If* $d_k > S_k$ *for some* k, *then replace* $g(i,j)$ *by* Y_{ij}^1 *and add* (i,j) *to the noise candidate set* $\mathcal{N}$. *Otherwise leave* $g(i,j)$ *unchanged.*

Figure 2: From top to bottom: images corrupted with 70% salt-pepper noise, restored by AMF, restored by 2-phase variational method [7], and restored by Algorithm 2.

References

[1] E. Abreu, M. Lightstone, S. Mitra, and K. Arakawa. A new efficient approach for the removal of impulse noise from highly corrupted images. *IEEE Trans. Image Process.*, 5(3):1012–1025, 1996.

[2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proceedings of SIGGRAPH*, pages 417–424, New Orleans, LA, 2000.

[3] J.-F. Cai, R. Chan, and C. Di Fiore. Minimization of a detail-preserving regularization functional for impulse noise removal. *Journal of Mathematical Imaging and Vision*, 29(1):79–91, 2007.

[4] J.-F. Cai, R. Chan, and Z. Shen. A framelet-based image inpainting algorithm. *Applied and Computational Harmonic Analysis*, 24(2):131–149, 2008.

[5] A. Chai and Z. Shen. Deconvolution: A wavelet frame approach. *Numerische Mathematik*, 106:529–587, 2007.

[6] R. Chan, T. Chan, L. Shen, and Z. Shen. Wavelet algorithms for high-resolution image reconstruction. *SIAM Journal on Scientific Computing*, 24(4):1408–1432, 2003.

[7] R. Chan, C.-W. Ho, and M. Nikolova. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *IEEE Transactions on Image Procssing*, 14:1479–1485, 2005.

[8] R. Chan, C. Hu, and M. Nikolova. An iterative procedure for removing random-valued impulse noise. *IEEE Signal Procssing Letters*, 11:921–924, 2004.

[9] R. Chan, S. D. Riemenschneider, L. Shen, and Z. Shen. Tight frame: The efficient way for high-resolution image reconstruction. *Applied and Computational Harmonic Analysis*, 17(1):91–115, 2004.

[10] R. Chan, L. Shen, and Z. Shen. A framelet-based approach for image inpainting. Technical Report 2005-4, The Chinese University of Hong Kong, Feb. 2005.

[11] T. Chan and J. Shen. Nontexture inpainting by curvature driven diffusion (CDD). *J. Visul Comm. Image Rep.*, 12:436–449, 2001.

[12] T. Chen and H. Wu. Space variant median filters for the restoration of impulse noise corrupted images. *IEEE Trans. Circuits and Systems II*, 48:784–789, 2001.

[13] P. Combettes and V. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 4:1168–1200, 2005.

[14] I. Daubechies. *Ten Lectures on Wavelets*, volume 61 of *CBMS Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1992.

[15] D. Donoho and I. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81:425–455, 1994.

[16] R. Duffin and A. Schaeffer. A class of nonharmonic Fourier series. *Transactions on American Mathematics Society*, 72:341–366, 1952.

[17] A. Flaig, G. Arce, and K. Barner. Affine order statistics filters: "medianization" of linear FIR filters. *IEEE Trans. Signal Processing*, 46:2101–2112, 1998.

[18] R. Gonzalez and R. Woods. *Digital Image Processing.* Addison-Wesley, Boston, MA, 1993.

[19] O. G. Guleryuz. Nonlinear approximation based image recovery using adaptive sparse reconstruction and iterated denoising: Part I - theory. *IEEE Transaction on Image Processing*, 2006.

[20] O. G. Guleryuz. Nonlinear approximation based image recovery using adaptive sparse reconstruction and iterated denoising: Part II - adaptive algorithms. *IEEE Transaction on Image Processing*, 2006.

[21] H. Hwang and R. Haddad. Adaptive median filters: new algorithms and results. *IEEE Trans. Image Processing*, 4:499–502, 1995.

[22] S. Ko and Y. Lee. Adaptive center weighted median filter. *IEEE Trans. Circuits and Systems II*, 38:984–993, 1998.

[23] S. Ko and Y. Lee. Adaptive center weighted median filter. *IEEE Transactions on Circuits and Systems II*, 38:984–993, 1998.

[24] M. Ng, R. Chan, and W. Tang. A fast algorithm for deblurring models with Neumann boundary conditions. *SIAM Journal on Scientific Computing*, 21:851–866, 2000.

[25] M. Nikolova. A variational approach to remove outliers and impulse noise. *J. Math. Imaging Vision*, 20:99–120, 2004.

[26] G. Pok, J.-C. Liu, and A. S. Nair. Selective removal of impulse noise based on homogeneity level information. *IEEE Trans. Image Processing*, 12:85–92, 2003.

[27] A. Ron and Z. Shen. Affine system in $L_2(R^d)$: the analysis of the analysis operator. *Journal of Functional Analysis*, 148:408–447, 1997.

[28] T. Sun and Y. Neuvo. etail-preserving based filters in image processing. *Pattern-Recognition Letters*, 15:341–347, 1994.

[29] Windyga. Fast impulsive noise removal. *IEEE Trans. Image Processing*, 10:173–179, 2001.

[30] L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo. Weighted median filters: a tutorial. *IEEE Trans. Circuit Theory*, 41:157–192, 1996.

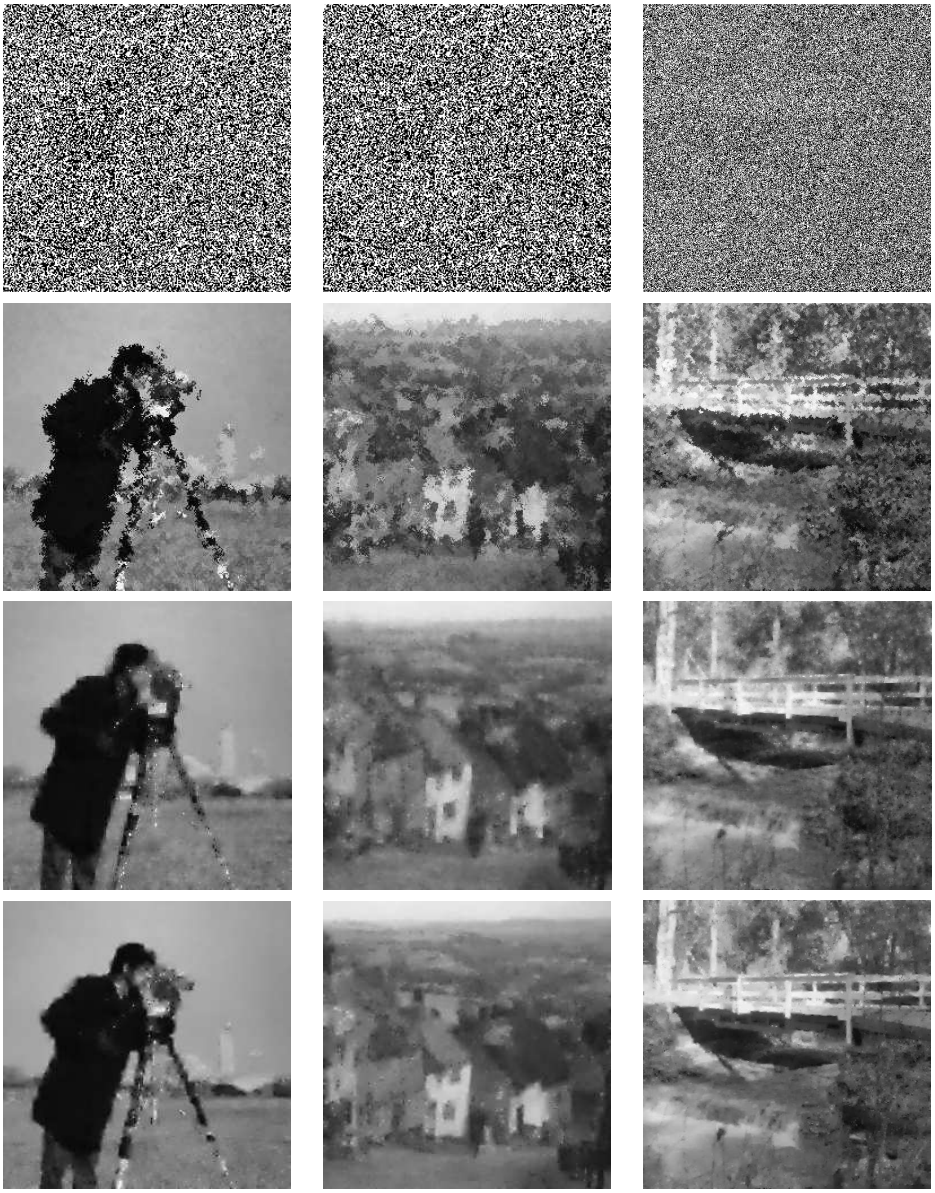

Figure 3: From top to bottom: images corrupted with 90% salt-pepper noise, restored by AMF, restored by 2-phase variational method [7], and restored by Algorithm 2.

Figure 4: From top to bottom: images corrupted with 30% random-valued impulse noise, restored by ACWMF, restored by 2-phase variational method [8], and restored by Algorithm 3.

Figure 5: From top to bottom: images corrupted with 50% random-valued impulse noise, restored by ACWMF, restored by 2-phase variational method [8], and restored by Algorithm 3.

Figure 6: The "Cameraman" image with text is shown in the left of the first row. Algorithm 2 with $\mathcal{N}$ being the location of the text is used to remove the text. Images (from left to right) in the first row are three different initial guesses for Algorithm 2. The corresponding inpainted images (from left to right) are presented in the second row with values of PSNR 30.64dB, 30.52dB, and 30.50dB, respectively. Similarly, for the three different initial guesses (from left to right) shown in the third row, the corresponding inpainted "Goldhill" images (from left to right) are presented in the last row with values of PSNR 32.62dB, 32.67dB, and 32.61dB, respectively.