A fast adaptive reweighted residual-feedback iterative algorithm for fractional-order total variation regularized multiplicative noise removal of partly-textured images

Jun Zhang \(^{a,b,}\), Zhihui Wei \(^{b}\), Liang Xiao \(^{b}\)

\(^{a}\) School of Science, Nanjing University of Science & Technology, Nanjing 210094, China
\(^{b}\) School of Computer Science & Technology, Nanjing University of Science & Technology, Nanjing 210094, China

**Abstract**

In this paper, we introduce a simple reweighted residual-feedback iterative (RRFI) algorithm which provides a general framework to solve the fractional-order total variation regularized models with different fidelity terms. We provide a sufficient condition for the convergence of this algorithm. As an application, we use this algorithm to solve the TV and fractional-order TV regularized models with two special fidelity terms for multiplicative noise removal of partly-textured images. To improve the performance, we define gradually varying fuzzy membership degrees to mark the possibilities of a pixel belonging to edges, textured regions and flat regions. Using the fuzzy membership degrees, we add local behavior to the choice of the parameters and the updating of the weighting matrix, and then propose an adaptive RRFI algorithm for multiplicative noise removal. Numerical results show that the RRFI algorithm has low computational cost and fast convergence speed. The adaptive RRFI algorithm performs well for preserving details and eliminating the staircase effect while removing noise, and therefore can improve the result visually efficiently.

1. Introduction

Multiplicative noise removal is central to the study of coherent imaging systems and appears in many applications, e.g. in synthetic aperture radar (SAR) where the noise is assumed to follow Gamma distribution, and in positron emission tomography where the Poisson noise appears. Given a noisy image \(f : \Omega \rightarrow \mathbb{R}\), with \(\Omega \subset \mathbb{R}^2\) an open and bounded domain, the problem is to extract the noise free image \(u\) from \(f\). In most practical applications, two important factors should be considered seriously: one is the preservation of the details such as edges and textures of the image; the other is the computational efficiency.

For the edge-preserving, the total variation (TV) regularized energy minimization \([1]\) is one of the most well-known techniques. It performs well for preserving edges while removing noise and has been used for multiplicative noise removal \([2–5]\) in recent years. However, TV regularized term favors piecewise constant solution, and therefore it goes against texture preservation and often causes staircase effect. To improve the performance of texture preservation, the non-local methods have been used widely in recent years \([6–10]\). However, these time-consuming non-local methods cannot eliminate the staircase effect and even cause extra blocky artifacts. For the staircase effect elimination, high-order derivative based models \([11–13]\) perform well, but they often cause blurring of the edges. As a compromise between the first-order TV
regularized models and the high-order derivative based models, some fractional-order derivative based models have been introduced in [14,15] for additive noise removal and subsequently used for image restoration [16] and super-resolution [17]. They can ease the conflict between staircase elimination and edge preservation by choosing the order of derivative properly. Moreover, the fractional-order derivative operator has a “non-local” behavior because the fractional-order derivative at a point depends upon the characteristics of the entire function and not just the values in the vicinity of the point [18], which is beneficial to improve the performance of texture preservation. The numerical results in literatures [14–17] have demonstrated that the fractional-order derivative performs well for eliminating staircase effect and preserving textures. Motivated by these works, we propose to consider fractional-order total variation regularized models with different fidelity terms for multiplicative noise removal, which can be written uniformly as follows:

$$\min_{u \in BV_{\Omega}(\alpha)} \{ J_\alpha(u) + \lambda H(f, u), \ 1 \leq \alpha \leq 2 \}. \quad (1)$$

where $BV_{\Omega}(\alpha) = \{ u : J_\alpha(u) < \infty \}$ is the set of functions of fractional-order bounded variation with $J_\alpha(u) = \int_{\Omega} \sqrt{ (D^\alpha_x u)^2 + (D^\alpha_y u)^2 } dx dy$ the fractional-order ($\alpha = \mathbf{th}$ order) total variation of $u$ [15]. Here, $D^\alpha_x u$ and $D^\alpha_y u$ are the $\alpha = \mathbf{th}$ order Grünwald-Letnikov fractional-order partial derivatives [18] with respect to the variable $x$ and $y$ respectively, which are defined by

$$D^\alpha_x u(x,y) = \lim_{\Delta x \to 0^+} \sum_{k \geq 0} (-1)^k C^\alpha_k (x-k\Delta x, y),$$

$$D^\alpha_y u(x,y) = \lim_{\Delta y \to 0^+} \sum_{k \geq 0} (-1)^k C^\alpha_k (x, y-k\Delta y), \quad (2)$$

where $C^\alpha_k = \Gamma (\alpha+1) / [k! \Gamma (\alpha-k+1)]$ is the generalized binomial coefficient with $\Gamma (\alpha)$ the Gamma function. Especially when $\alpha = 1$, we have $C^1_0 = 1$, $C^1_1 = -1$ and $C^1_k = 0$ for any $k \geq 2$, and therefore $J_1(u)$ is the total variation of $u$ as usual, and therefore the fractional-order total variation can be cast as an extension of the traditional total variation.

The term $H(f, u)$ in Eq. (1) is a fidelity term related to the prior information of the noise and $\lambda > 0$ is the regularization parameter. For instance, when we consider the multiplicative Gamma noise removal, the AA model [3] with $H(f, u) = \int_{\Omega} \log u + f / u \, dx dy$ and the I-divergence model [5] with $H(f, u) = \int_{\Omega} u - f \log u \, dx dy$ are two most commonly used TV regularized models. However, the computation of these two models suffers from serious difficulties caused by the non-linearity and non-differentiability of the fidelity terms. To simplify the computation, Huang et al. [20] proposed a splitting-and-penalty based algorithm so-called the HNM algorithm to solve the AA model, and Steidl et al. [5] proposed the Douglas–Rachford splitting (DRS) algorithm to solve the I-divergence model. However, each iteration loop of these two algorithms consists of solving two nonlinear equations. For the HNM algorithm, one equation can be solved by using Chambolle’s algorithm [19], but the other one should be solved by using Newton-like iterative method which leads to slow convergence speed and high computational cost. For the DRS algorithm, though one equation has a close-form solution, the other one should be linearized firstly and solved by using iterative method to solve the linear equation. Moreover, once we replace the TV regularized model, it is difficult to extend these two algorithms to solve the improved fractional-order total variation regularized models due to the complexity of the fractional-order derivative.

To solve these fractional-order TV regularized models with complex fidelity terms, we introduce a reweighted residual-feedback iterative (RRFI) algorithm in this paper. This new algorithm can be cast as an extension of the attractive iterative regularization method proposed by Osher et al. in [21] which has led to the popularity of the Bregman iteration in signal and image processing. As an application, we utilize the RRFI algorithm to solve model (1) with the fidelity terms used in the AA model and the I-divergence model. Moreover, we propose a strategy to update the parameters adaptively, and then develop a fast adaptive RRFI algorithm for multiplicative noise removal.

The rest of this paper is organized as follows: In Section 2, we introduce the RRFI algorithm and analyze its convergence. In Section 3, we propose a strategy to update the weighting matrix adaptively and obtain the adaptive RRFI algorithm for multiplicative noise removal. In Section 4, numerical examples and comments are given to show the efficiency of the new algorithm. Finally, the paper is concluded in Section 5.

2. Reweighted residual-feedback iterative algorithm

2.1. Reweighted residual-feedback iteration (RRFI)

In [21], Osher et al. have proposed an attractive iterative regularization method to solve the TV regularized model for additive noise removal. Each iteration loop of this algorithm consists of two steps: one is to generate a new modified noisy image by feeding a residual image obtained in the last iteration loop back to the original noisy image; the other is to remove the noise from the modified noisy image again. The convergence speed of this algorithm is very fast. However, the denoised image sequence converges to the original noisy image. To remedy this, a simple idea is that when we generate the modified noisy image, we just feed useful components such as edges and textures back to the noisy image. Inspired by this idea, we consider the following reweighted residual-feedback iteration scheme:

$$\begin{align*}
& g^{(n)} = f + w^{(n)} (f - u^{(n)}), \\
& u^{(n+1)} = \arg \min_{u \in BV_{\Omega}(\alpha)} \{ J_\alpha(u) + \frac{\lambda}{2} \| g^{(n)} - u \|_2^2 \}, \ 1 \leq \alpha \leq 2
\end{align*} \quad (3)$$

where $w^{(n)}$ is the weighting matrix and $\gamma > 0$ is the regularization parameter of the minimization sub-problem. Here, $u^{(0)} = f$ and $w^{(0)} = 1$ with 1 the matrix of ones.

In the RRFI scheme (3), once we obtain a denoised image $u^{(n)}$, we can obtain additive residual image $v^{(n)} = f - u^{(n)}$. The residual image $v^{(n)}$ can be used as the weight matrix in the next iteration. This iteration loop will continue until the residual image $v^{(n)}$ is very small, which is the stopping criteria of the RRFI algorithm.
Then we feed the weighted residual image \( w^{(n)}(f - u^{(n)}) \) back to the original noisy image \( f \) and generate a new modified noisy image \( g^{(n)} \) for the next iteration loop. This process is similar to iterative regularization method proposed in [21]. In fact, when \( a = 1, \gamma = \lambda \) and \( w^{(n)} = (g^{(n)} - u^{(n)})/(f - u^{(n)}) \), the RRFI scheme (3) is exactly the same as the method proposed in [21]. However, the RRFI scheme is obviously more general and more flexible than the method proposed in [21]. Here, we can choose variant weighting matrix according to different fidelity terms, and can also adjust it space-varyingly in different regions of the image to improve the performance of noise removal and detail preservation. Moreover, we can choose proper parameter \( a \) space-varyingly to improve the performance of staircase effect elimination and detail preservation. From this perspective, the RRFI scheme (3) can also be cast as an extension of the iterative regularization method proposed in [21].

The minimization sub-problem in the RRFI scheme (3) can be solved easily by using the extended Chambolle’s algorithm proposed in [15]. Here, we restate this algorithm briefly. To simplify, we assume that the images are two-dimensional matrices in \( \mathbb{R}^{N \times N} \). For any \( u \in \mathbb{R}^{N \times N} \), we denote \( u_{ij} = 0 \) for any \( i, j < 1 \) or \( i, j > N \). Then minimization sub-problem in (3) can be solved by using the following steps [15]:

(i) For fixed \( g^{(n)} \), given \( p^{(n)} = (0, 0) \in \mathbb{R}^{N \times N} \times \mathbb{R}^{N \times N} \) and the parameter \( r > 0 \), we compute

\[
p_{ij}^{(k+1)} = (p_{ij}^{(k)} - \gamma H_{ij}^{(k)})/(1 + \gamma r H_{ij}^{(k)})^{-1}, \quad i, j = 1, 2, 3, \ldots, N,
\]

where \( H_{ij}^{(k)} = \nabla u_{ij} (\nabla^{2} u_{ij})_{ij} = (\sum_{k=0}^{a}(1-k)C_{k}^{a}u_{i-kj} + \sum_{k=0}^{a}(-1)^{k}C_{k}^{a}u_{ij-k})
\]

\[
\text{(div) p}_{ij} = (-1)^{r} \sum_{k=0}^{a}(1-k)C_{k}^{a}(p_{i+1,j+k} + p_{i-1,j-k})
\]

where \( K \geq 2 \) is an integer. For fixed \( a \in (1,2) \), the coefficient \( C_{k}^{a} \) is vanishingly close to zero very quickly with the increasing of \( k \), and therefore we just choose \( K = 20 \) in this paper.

(ii) Once the given termination condition is satisfied, we terminate the iteration (4) and compute the approximate solution directly as follows:

\[
u^{(n+1)} = g^{(n)} - \gamma^{-1} (-1)^{r} \text{div} p^{(n+1)} = -w^{(n)}u^{(n)} + (w^{(n)} + 1)f - \gamma^{-1} (-1)^{r} \text{div} p^{(n+1)}
\]

To simplify, if we denote \( h^{(n)} = (w^{(n)} + 1)f - \gamma^{-1} (-1)^{r} \text{div} p^{(n+1)} \), then we have

\[
u^{(n+1)} = -w^{(n)}u^{(n)} + h^{(n)} = (-1)^{n+1}w^{(n)}u^{(n)} - \cdots w^{(0)}u^{(0)} + \sum_{k=0}^{n} (-1)^{n-k}w^{(k)}u^{(k)} + \sum_{i=j=1,2,\ldots,N, N-1}^{N} w^{(n)}u^{(n)} + h^{(n)}
\]

and therefore

\[
lm \sum_{n=-\infty}^{\infty} \sum_{k=0}^{n} (-1)^{n-k}w^{(k+1)}u^{(k)} + \cdots w^{(0)}u^{(0)} + \sum_{i=j=1,2,\ldots,N, N-1}^{N} w^{(n)}u^{(n)} + h^{(n)}
\]

It is easy to check that the sequence \( \{p_{ij}^{(n)}\} \) generated by the iterative scheme (4) satisfies \( \{p_{ij}^{(n)}\} \leq 1 \) for any \( k \geq 0 \) and

The core of the RRFI algorithm is the updating of the weighting matrix. We can update it accordingly to different rules for different fidelity terms. However, no matter what kind of rule we use, we must ensure the convergence of the algorithm firstly.

2.2. Convergence analysis

In this paper, we provide a sufficient condition for the convergence of the RRFI algorithm as follows:

**Theorem.** An efficient condition of the RRFI algorithm is that the parameter \( r \) in the iterative scheme (4) and the weighting matrix \( w^{(n)} = [w^{(n)}]_{ij} \) satisfy

\[
r \leq (2K \sum_{k=0}^{a}(C_{k}^{a})^{-1} \text{div} p^{(n+1)} \leq C, \quad i, j = 1, 2, \ldots, N, \quad n = 0, 1, 2, \ldots
\]

where \( 0 < C < 1 \) is a constant.

**Proof.** In the RRFI algorithm, the iterative scheme (4) is the inner iteration and has been proved to be convergent when \( r \leq (2K \cdot \sum_{k=0}^{a}(C_{k}^{a})^{-1} \) is satisfied [15]. Therefore, we just need to prove that the denoised image sequence \( \{u^{(n)}\} \) is convergent when \( \{w_{ij}^{(n)}\} \leq C < 1 \).

According to the Eqs. (3) and (5), we have

\[
u^{(n+1)} = g^{(n)} - \gamma^{-1} (-1)^{r} \text{div} p^{(n+1)} = -w^{(n)}u^{(n)} + (w^{(n)} + 1)f - \gamma^{-1} (-1)^{r} \text{div} p^{(n+1)}
\]

where \( n \geq 0 \) and \( \gamma > 0 \)
any long as the condition (7) is satisfied, which brings great
the series \( \lim_{n \to \infty} \sum_{k=0}^{n} |w^{(k+1)}_{ij}| \cdots |w^{(m)}_{ij}| \) satisfies
\[
\lim_{n \to \infty} \sum_{k=0}^{n} |w^{(k+1)}_{ij}| \cdots |w^{(m)}_{ij}| \leq \sum_{k=0}^{\infty} BC^k = \frac{B}{1-C}.
\]

According to the comparison test for positive term series,
\[
\sum_{n=0}^{\infty} \left| \frac{w^{(k+1)}_{ij}}{\lambda} \cdots \frac{w^{(m)}_{ij}}{\lambda} \right| \leq \sum_{k=0}^{\infty} BC^k = \frac{B}{1-C},
\]
the sequence \( \{u^{(m)}\} \) generated by the proposed RRFI algorithm is convergent.

It is worth noting that the sequence \( \{u^{(m)}\} \) generated by the RRFI algorithm is convergent point-wisely, which implies that no matter the parameters are constants or point-wisely varying, the sequence \( \{u^{(m)}\} \) is convergent as long as the condition (7) is satisfied, which brings great convenience and flexibility in practice for choosing the parameters.

3. Adaptive RRFI algorithm for multiplicative noise removal

The RRFI algorithm provides a general framework to solve the TV and fractional-order TV regularized models with different fidelity terms. In this paper, as an application, we propose to use the RRFI algorithm to solve the TV and fractional-order TV regularized models with certain fidelity terms for multiplicative noise removal.

3.1. The fractional-order TV regularized models for multiplicative noise removal

In this paper, we focus on the multiplicative Gamma noise removal. Here, we assume the mean of the noise is 1.

We consider two fractional-order total variation regularized models for multiplicative noise removal as follows:

(i) The fractional-order AA model:
\[
\min_{u \in BV, u \geq 0} \int_{\Omega} [J_f(u) + \lambda u] dx dy, \quad 1 \leq \alpha \leq 2.
\]

(ii) The fractional-order I-divergence model:
\[
\min_{u \in BV, u \geq 0} \int_{\Omega} [J_f(u) + \lambda (u - f) \log u] dx dy, \quad 1 \leq \alpha \leq 2.
\]

where \( J_f(u) \) is the sub-gradient of the operator \( f(\cdot) \) at \( u \) and \( q \in [1, 2] \) is a constant. When \( q = 2 \), the Eq. (10) is the Euler–Lagrange equation of the model (8); when \( q = 1 \), the Eq. (10) is the Euler–Lagrange equation of the model (9).

Using the operator splitting technique, we can solve the Eq. (10) according to the following alternating iterative scheme:

\[
\begin{aligned}
\lambda^{(m)} &= \lambda^{(m-1)}(u^{(m)})^{-q} - 1, \\
\mu^{(m)} &= \min_{u > 0} \left\{ J_f(u) + \frac{\lambda}{q} \|g^{(m)} - u\|_2^2 \right\}.
\end{aligned}
\]

Obviously, this iterative scheme is the RRFI scheme (3) with \( \lambda^{(m)} = \lambda^{(m-1)}(u^{(m)})^{-q} - 1 \). It should be noted that if the parameters \( \lambda \) and \( \gamma \) are constants, the weighting matrix \( \lambda^{(m)} \) may not meet the condition (7). Fortunately, the RRFI algorithm is convergent point-wisely, and therefore we can adjust these parameters point-wisely to ensure that \( \lambda^{(m)} \) can meet the condition (7). To simplify, we set \( \gamma \) as a constant and adjust the parameter \( \lambda \) point-wisely. Furthermore, for the convenience of computation in practice, instead of adjusting the parameter \( \lambda \), we update the weighting matrix \( \lambda^{(m)} \), we prefer to adjust the weighting matrix directly after updating it with a constant \( \lambda \).

As we have mentioned, to improve the performance of noise removal and detail preservation, we should feed more textures and edges but less noise back in the RRFI algorithm. It implies that the weights at the edges and in the textured regions should be larger than that in the flat regions. In addition, the numerical results in [15] showed that we should choose \( \alpha = 1 \) at edges to prevent the edge-blurring, and choose \( \alpha \) larger slightly than 1 in the flat regions to eliminate the staircase effect and even larger in the textured regions to improve the performance of texture preservation. Toward these goals, we need to classify the pixels of the image into edges, texture regions and flat regions firstly.

In [22], Gilboa et al. presented a method to detect the textured regions automatically, which was subsequently used in [15,22–24]. They computed a preliminary denoised image and the corresponding residual image firstly, and then classified the pixels into textured regions and flat regions (without considering the edges) by performing hard-thresholding on the local variance measures of the residual image. However, the accuracy of the detection is sensitive to the estimation of the variances (including the variance of the noise and the local variances of the residual image) and the choice of the threshold. In addition, the hard-thresholding often leads to fragment effect which is responsible for the unnatural visual result. To remedy these defects, we present a strategy to estimate the noise variance and local variance adaptively firstly, and then define gradually varying fuzzy membership degrees to mark the possibilities of a pixel belonging to edges, textured regions and flat regions.

3.2. Variances estimation and fuzzy membership degrees

3.2.1. Preliminary noise removal and variances estimation

Firstly, we use the RRFI scheme (11) with \( \alpha = 1 \) to compute a preliminary denoised image and the corresponding residual
image. Here, we want to remove both noise and textures as much as possible, and therefore we can obtain a clean denoised cartoon image used to detect the edges and a residual image which is conducive to distinguish the textured regions and non-textured regions by using the local variance measures. In this case, the weight \( w_{ij} \) can be adjusted to be just larger slightly than \(-1\) for any \( i,j \). According to the Eq. (10), if \( \lambda \) as a constant, then we can obtain

\[
\lambda = \frac{1}{\sigma^2} \sum_{k,l=1}^{N} u_{k,l}^2 \partial J(u_{k,l})(f_{k,l} - u_{k,l}),
\]

(12)

where \( \sigma^2 \) is the variance of the noise which is often unknown in practice. Using the method proposed in [24], we can compute a preliminary estimation of the noise variance as follows:

\[
\hat{\sigma}^2 = \left( \frac{\text{Mid}_{HH}}{0.6745} \right)^2,
\]

(13)

where Mid_{HH} is the median absolute deviation of the wavelet coefficients in the HH subband at the finest resolution scale of the multiplicative residual image \( f/\exp(W_{of}(\log f)) \). Here, \( W_{of}(\cdot) \) denotes the traditional wavelet soft-thresholding operator.

Once we obtain a denoised image \( \hat{u}^{(n)} \), according to the iterative scheme (11) and the Eqs. (12) and (13), we compute

\[
\hat{\lambda}^{(n)}(i,j) = (N\hat{\sigma})^{-\frac{2}{3}} (u_{k,l}^{(n)})^{-\frac{2}{3}} \sum_{k,l=1}^{N} \left[ (u_{k,l}^{(n)})^{-\frac{2}{3}} - \hat{u}_{k,l}^{(n)} \right] f_{k,l} - \hat{u}_{k,l}^{(n)} \right],
\]

(14)

and update the weighting matrix \( w^{(n)} \) by mapping \( \hat{\lambda}^{(n)}(i,j) \) on to \([-0.99, -0.95]\) as follows

\[
w^{(n)}(i,j) = \begin{cases} \frac{\hat{\lambda}^{(n)}(i,j) - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}} 0.05 + \frac{\hat{\lambda}^{(n)}(i,j) - \lambda_{\max}}{\lambda_{\min} - \lambda_{\max}} 0.01 & \text{if } \lambda_{\min} < \hat{\lambda}_{i,j} \leq \lambda_{\max}, \\ 1 & \text{otherwise}. \end{cases}
\]

(15)

where \( \lambda_{\min} = \min \{\hat{\lambda}^{(n)}(i,j) \} \) and \( \lambda_{\max} = \max \{\hat{\lambda}^{(n)}(i,j) \} \). When we compute \( \hat{\lambda}^{(n)}(i,j) \), for the convenience of computation in practice, the term \( (u_{k,l}^{(n)})^{-\frac{2}{3}} \) in the Eq. (12) is replaced by \( r(g_{k,l}^{(n-1)} - u_{k,l}^{(n)}) \) according to Euler–Lagrange equation of the minimization sub-problem of the iterative scheme (11).

In this iterative processing, once we obtain a denoised image \( \hat{u}^{(n)} \), we estimate the variance of the noise as

\[
\hat{\sigma}^2 = \left( \text{Mid}_{HH}/0.6745 \right)^2,
\]

where Mid_{HH} is the median absolute deviation of the wavelet coefficients in the HH subband at the finest resolution scale of the corresponding multiplicative residual image \( \psi^{(n)} = f/\hat{u}^{(n)} \). If \( \hat{\sigma}^2 \geq \hat{\sigma}^2 \) or \( \hat{\sigma}^2_{\text{HH}} \neq \hat{\sigma}^2_{\text{HH}} \), we terminate the RRIR iteration and obtain

\[
\begin{align*}
\hat{u}_{ij}^{(n+1)} &= \hat{u}_{ij}^{(n)}, \\
\hat{\sigma}^2 &= \min \{\hat{\sigma}^2_{\text{HH}}, \hat{\sigma}^2_{\text{HH}} \}, \\
\hat{P}_{ij}^{(n+1)} &= \sum_{p,q=1}^{M^{-2}} \psi^{(n)}(p,q) - M^{-2} \sum_{p,q=1}^{M^{-2}} \psi^{(n)}(p,q)^2, \quad i,j = 1,2,\ldots,N.
\end{align*}
\]

(16)

where \( \sigma^2 \) is a updated estimation of the noise variance, \( u_{ij}^{(n)} \) is the preliminary denoised image mainly contains the flat regions and edges, and \( P_{ij}^{(n)} \) is the local variance at pixel \((i,j)\) of the residual image \( \psi^{(n)} = f/\hat{u}^{(n)} \). Here, \( W_{ij} = [i-(M-1)/2;i+(M-1)/2]/[j-(M-1)/2;j+(M-1)/2] \) is a window of size \( M \times M \) (\( M \) is odd) centered at the pixel \((i,j)\).

### 3.2.2. Fuzzy membership degrees

In this paper, we define three fuzzy membership degrees for each pixel \((i,j)\) to mark its possibilities of belonging to edges, textured regions and flat regions, respectively. The fuzzy membership degrees are varying gradually, which is beneficial to eliminate the fragment effect caused by the hard-thresholding used in [15,22–24].

#### (i) The fuzzy membership degree matrix of edges

The preliminary denoised image \( u_{ij}^{(n)} \) mainly contains the edges and flat regions. We can detect the edges by using traditional edge detectors such as Canny edge detector. However, due to the staircase effect caused by the TV regularization, the detected edges often contain pseudo edges. Before we use canny edge detector, we use Gaussian filter to eliminate these pseudo edges. Namely, the binary indicative function of the edges is defined by

\[
\chi_E = \text{Canny} \left( u_{ij}^{(n)} \right),
\]

(17)

where \( \text{Canny} \left( \cdot \right) \) is the canny edge detector, \( G_1 \) is a 2-dimensional Gaussian window and the star \( * \) denotes the convolution.

However, the edges detected are often not exactly correct in practice. The pixels near the detected edges may well belong to the real edges. Therefore, we define the fuzzy membership degree to mark the possibility of each pixel of the image belongs to edges. In this paper, the fuzzy membership degree matrix of edges denoted \( \text{FMD}_E \) is defined by

\[
\text{FMD}_E(i,j) = \text{Eblur}(i,j)/\max_{i,j\in\text{Eblur}} \text{Eblur}(i,j),
\]

(18)

where \( \text{Eblur} \) is a 2-dimensional Gaussian window. The size of \( G_2 \) is small \((5 \times 5 \text{ in this paper})\), because we just deem that the points near the detected edges are likely to belong to real edges.

#### (ii) The fuzzy membership degree matrix of textured regions

Affected by the textures, local variances in the textured regions are much larger than that in the flat regions. Therefore, hard-thresholding on the local variances is one of the simplest methods to detect the textured regions, and it is legitimate to choose the threshold in \([\hat{p}_{\lambda_{\min}}^{\text{max}}, \hat{p}_{\lambda_{\max}}^{\text{max}}]\) with \( \hat{p}_{\lambda_{\min}}^{\text{max}} = \text{mean}(\hat{p}_{\hat{\lambda}_{\min}}^{\text{max}}, i,j = 1,2,\ldots,N) \) and \( \hat{p}_{\lambda_{\max}}^{\text{max}} = \text{max}(\hat{p}_{\hat{\lambda}_{\max}}^{\text{max}}, i,j = 1,2,\ldots,N) \), as in [15,24]. However, this hard-thresholding based method is sensitive to the choice of the threshold. In this paper, we propose to search the threshold and obtain a preliminary indicative function of the textured regions adaptively.
by using the following iterative steps:

**Step 1:** Estimate the binary indicative function of the textured regions as
\[ x_{i,j}^{T} = \begin{cases} 1, & P_{pre}^{ij} \geq P_{max}^{pre} + 0.01(k-1)(P_{max}^{pre} - P_{mean}^{pre}), \\ 0, & \text{otherwise}. \end{cases} \] (19)

**Step 2:** Compute the mean of the local variances in the flat regions as follows
\[ \sigma_{k}^{2} = \text{mean}(x_{i,j}^{T} | x_{i,j}^{T} = 0, FMD_{T}(i,j) = 0, i, j = 1, 2, \ldots, N) \] (20)

**Step 3:** If \( \sigma_{k}^{2} \neq \sigma_{pre}^{2} \) or \( k > 101 \), then we terminate the iteration and obtain a preliminary indicative function of the textured region as \( x_{i,j}^{T} = x_{i,j}^{T_{k_{0}}} \), where \( k_{0} \) is the stopping index; otherwise, we denote \( k := k + 1 \) and go to Step 1. Here, \( \sigma_{pre}^{2} \) is the variance of noise in the residual image \( \psi_{pre} = f/\psi_{pre} \) estimated by using the method proposed in [25].

In general, textured regions are region-continuous. However, the regions indicated by \( x_{i,j}^{T} \) often have fragments or even isolated points which cannot be classified into textured regions. Therefore, we modify the binary indicative function of the textured regions as
\[ x_{i,j} = \begin{cases} 1, & G_{1}(x_{i,j}^{T} | x_{i,j}^{T} \geq \lambda_{0}), \\ 0, & \text{otherwise}. \end{cases} \] (21)

where \( G_{1} \) is a 2-dimensional Gaussian window.

Similar to the case of edges, points near the detected textured regions may well belong to the real textured regions too. Furthermore, the detected textured regions often contain the edges and we should consider the differences between these two different components. Therefore, we define the fuzzy membership degree matrix of textured regions denoted \( FMD_{T} = [FMD_{T}(i,j)]_{i,j} \) as follows
\[ \left\{ \begin{array}{l} T_{\text{blur}} = (1 - FMD_{T}) \cdot (G_{2}(x_{i,j}^{T})), \\ FMD_{T}(i,j) = T_{\text{blur}} (i,j) / \max_{i,j=1,2,\ldots,N} (T_{\text{blur}} (i,j)), \end{array} \right. \] (22)

where \( G_{2} \) is the 2-dimensional Gaussian window used in the Eq. (18).

### (iii) The fuzzy membership degree matrix of flat regions

In this paper, we just consider three kinds of regions and the sum of the possibilities of each pixel belonging to them must be 1. Therefore, according to the Eqs. (18) and (22), we can define the fuzzy membership degree matrix of the flat regions as follows
\[ FMD_{F}(i,j) = 1 - FMD_{T}(i,j) - FMD_{T}(i,j). \] (23)

The fuzzy membership degrees defined in this section transit gradually at the border areas between the edges, the textured regions and the flat regions, and therefore we can eliminate the fragment effect caused by the hard-thresholding.

### 3.3. Computation of the adaptive parameters

To improve the performance in noise removal, edge preservation, texture preservation and staircase effect elimination simultaneously, we want to add local adaptability behaviors to the weighting matrix \( w_{n}^{(m)} \) and parameter \( \alpha \). In a nutshell, we want to choose different parameters in different regions of the image.

As we have mentioned, to improve the performance of detail preservation and noise removal, we want to feed more textures and edges but less noise back in the RRFI algorithm. It implies that we should choose large weights at the edges and in the textured regions, but small weights in the flat regions. In this paper, once we obtain a new denoised image \( u_{n}^{(m)} \), we can compute \( \lambda \) according to the Eq. (14) firstly and then update the \( w_{n}^{(m)} \) to meet the convergence condition (7) by mapping \( \lambda_{n}^{(m)}(i,j) \) on to \( [C_{1} - 1, C_{4} - 1] \in (-1, 1) \) as follows:

\[
\begin{align*}
    w_{n}^{(m)}(i,j) &= FMD_{T}(i,j) \left[ \frac{\lambda_{n}^{(m)}(i,j) - \lambda_{F}^{min}}{\lambda_{F}^{max} - \lambda_{F}^{min}} C_{1} + \frac{\lambda_{n}^{(m)}(i,j) - \lambda_{F}^{min}}{\lambda_{F}^{max} - \lambda_{F}^{min}} C_{2} \right] \\
    &+ FMD_{T}(i,j) \left[ \frac{\lambda_{n}^{(m)}(i,j) - \lambda_{E}^{max}}{\lambda_{E}^{max} - \lambda_{E}^{min}} C_{2} + \frac{\lambda_{n}^{(m)}(i,j) - \lambda_{E}^{min}}{\lambda_{E}^{max} - \lambda_{E}^{min}} C_{3} \right] \\
    &+ FMD_{T}(i,j) \left[ \frac{\lambda_{n}^{(m)}(i,j) - \lambda_{T}^{max}}{\lambda_{T}^{max} - \lambda_{T}^{min}} C_{3} + \frac{\lambda_{n}^{(m)}(i,j) - \lambda_{T}^{min}}{\lambda_{T}^{max} - \lambda_{T}^{min}} C_{4} \right] - 1 \\
\end{align*}
\]

where
\[
\begin{align*}
    \lambda_{F}^{min} &= \min_{1 \leq i,j \leq N} \{ \lambda_{n}^{(m)}(i,j) | FMD_{T}(i,j) \neq 0 \}, \\
    \lambda_{F}^{max} &= \max_{1 \leq i,j \leq N} \{ \lambda_{n}^{(m)}(i,j) | FMD_{T}(i,j) \neq 0 \}, \\
    \lambda_{E}^{min} &= \min_{1 \leq i,j \leq N} \{ \lambda_{n}^{(m)}(i,j) | FMD_{T}(i,j) \neq 0 \}, \\
    \lambda_{E}^{max} &= \max_{1 \leq i,j \leq N} \{ \lambda_{n}^{(m)}(i,j) | FMD_{T}(i,j) \neq 0 \}, \\
    \lambda_{T}^{min} &= \min_{1 \leq i,j \leq N} \{ \lambda_{n}^{(m)}(i,j) | FMD_{F}(i,j) \neq 0 \}, \\
    \lambda_{T}^{max} &= \max_{1 \leq i,j \leq N} \{ \lambda_{n}^{(m)}(i,j) | FMD_{F}(i,j) \neq 0 \}.
\end{align*}
\]

Here, the parameters \( C_{1}, C_{2}, C_{3} \) and \( C_{4} \) are given by
\[
\begin{align*}
    C_{1} &= \max \{ 0.7, P_{min}^{pre} / P_{max}^{pre} | 0 \}, \\
    C_{2} &= C_{0}, \\
    C_{3} &= \min \{ 1.9, (P_{mean}^{pre} / P_{max}^{pre})^{1/2} C_{0} \}, \\
    C_{4} &= \min \{ 1.9, (P_{max}^{pre} / P_{mean}^{pre})^{2} C_{0} \}.
\end{align*}
\]

with
\[
\begin{align*}
    C_{0} &= \max \{ 0.05, 1 - e^{-0.5} \}, \\
    P_{min}^{pre} &= \min_{1 \leq i,j \leq N} \{ P_{pre}(i,j) \}, \\
    P_{max}^{pre} &= \max_{1 \leq i,j \leq N} \{ P_{pre}(i,j) \}.
\end{align*}
\]
Table 1
Comparison of the SSIM, PSNR and CPU time obtained by solving the AA model and the fractional-order AA model.

<table>
<thead>
<tr>
<th>Noisy image (noisy variance)</th>
<th>Algorithm</th>
<th>SSIM</th>
<th>PSNR (db)</th>
<th>CPU time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barbara ($\sigma^2 = 0.01$)</td>
<td>HNW algorithm</td>
<td>0.8661</td>
<td>29.6621</td>
<td>48.9531</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm ($\alpha = 1$)</td>
<td>0.8686</td>
<td>29.8533</td>
<td>22.2769</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8665</td>
<td>29.7966</td>
<td>33.1658</td>
</tr>
<tr>
<td>Barbara ($\sigma^2 = 0.03$)</td>
<td>HNW algorithm</td>
<td>0.7734</td>
<td>26.6273</td>
<td>95.2542</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm ($\alpha = 1$)</td>
<td>0.8247</td>
<td>27.2042</td>
<td>16.3021</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8269</td>
<td>27.2145</td>
<td>30.0926</td>
</tr>
<tr>
<td>Barbara ($\sigma^2 = 0.1$)</td>
<td>HNW algorithm</td>
<td>0.6719</td>
<td>24.2543</td>
<td>194.1916</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm ($\alpha = 1$)</td>
<td>0.7055</td>
<td>24.4241</td>
<td>28.3610</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.7072</td>
<td>24.5652</td>
<td>39.3123</td>
</tr>
<tr>
<td>Lena ($\sigma^2 = 0.01$)</td>
<td>HNW algorithm</td>
<td>0.8805</td>
<td>33.6021</td>
<td>50.6223</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm ($\alpha = 1$)</td>
<td>0.8912</td>
<td>33.7029</td>
<td>17.9713</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8949</td>
<td>33.9005</td>
<td>34.1798</td>
</tr>
<tr>
<td>Lena ($\sigma^2 = 0.03$)</td>
<td>HNW algorithm</td>
<td>0.8464</td>
<td>31.0830</td>
<td>99.2166</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm ($\alpha = 1$)</td>
<td>0.8444</td>
<td>31.1224</td>
<td>17.5033</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8512</td>
<td>31.3746</td>
<td>33.0254</td>
</tr>
<tr>
<td>Lena ($\sigma^2 = 0.1$)</td>
<td>HNW algorithm</td>
<td>0.7916</td>
<td>28.0234</td>
<td>182.6148</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm ($\alpha = 1$)</td>
<td>0.7813</td>
<td>27.9883</td>
<td>27.5810</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.7957</td>
<td>28.1702</td>
<td>39.2967</td>
</tr>
</tbody>
</table>

Fig. 1. The noise free images and the noisy images polluted by the multiplicative Gamma noisy with $\sigma^2 = 0.03$. (a) Original Barbara (512 × 512). (b) Original Lena (512 × 512). (c) Noisy Barbara. (d) Noisy Lena.
and
\[ P_{\text{mean}}^{\text{pre}} = \text{mean} \left\{ P_{\text{pre}}^{\text{FMDT}(i,j) \neq 0} \right\} \]
\[ P_{\text{mean}}^{\text{post}} = \text{mean} \left\{ P_{\text{pre}}^{\text{FMDT}(i,j) \neq 0} \right\}. \]

Here, \( \sigma^2 \) is the variance of the noise estimated according to the Eq. (16). The parameter \( C_0 \) will decrease with the increasing of the noise level to ensure the performance of the noise removal. On the other hand, to ensure enough underlying information can be fed back, we set lower

Fig. 2. The curves of PSNR, SSIM and Relative Errors of the denoised images obtained by solving the AA model and the fractional-order AA model. The curves in the first column are obtained by removing noise from the noisy image Fig. 1(c). The curves in the second column are obtained by removing noise from the noisy image Fig. 1(d).
bound to $C_0$. If the texture features are not obvious in the residual image $v_{\text{pre}} = f/u_{\text{pre}}$ (as is the case when the noise level is high or the texture features are too weak), then $p_{\text{pre min}} \approx f_{\text{max}}$ and therefore $C_4 \approx C_3 \approx C_2 \approx C_0$. In this case, noise removal is dominated. If the texture features are obvious in the residual image, then $p_{\text{pre min}} < p_{\text{pre mean}} < p_{\text{pre max}}$ and therefore $C_3$ and $C_4$ are large. As expected, it implies that the weights at the edges and in the textured regions are large, and therefore we can feed more textures and edges back, which is beneficial to the preservation of the edges and the textures.

The numerical experiments in [15] showed that we should choose $\alpha = 1$ to preserve the edges, and choose $\alpha$ larger slightly than 1 in the flat regions to eliminate the staircase effect and even larger in the textured regions to improve the performance of texture preservation. According to results, we propose to choose the parameter $\alpha$ at the point $(i,j)$ as follows

$$
al(i,j) = FMD_T(i,j) + FMD_T(i,j) \left[ 1.0\frac{p_{\text{pre min}} - p_{\text{pre mean}}}{p_{\text{pre max}} - p_{\text{pre mean}}} + 1.5\frac{p_{\text{pre mean}} - p_{\text{pre min}}}{p_{\text{pre max}} - p_{\text{pre mean}}} \right] + FMD_T(i,j)1.5,$$

(26)

where $p_{\text{pre min}} = \min_{1 \leq i,j \leq} \{p_{\text{pre ij}} | FMD_T(i,j) \neq 0\}$ and $p_{\text{pre max}} = \max_{1 \leq i,j \leq} \{p_{\text{pre ij}} | FMD_T(i,j) \neq 0\}$.

Fig. 3. Comparison of the denoised images, flat regions and textured regions of the Lena image obtained by using the HNW algorithm and RRFI algorithm with $\alpha = 1$ for the AA model, and the adaptive RRFI algorithm for the fractional-order AA model. (a) The HNW algorithm for the AA model. (b) The RRFI algorithm with $\alpha = 1$ for the AA model. (c) The adaptive RRFI algorithm for the fractional-order AA model.
In addition, to meet the convergence condition (7), we choose the parameter \( \tau \) in the iteration scheme (4) as
\[
\tau = \min_{i, j, \ldots, N} \left\{ \frac{2K\sum_{k=1}^{K} \left( C_{n(i,j)}^k \right)^2}{C_{n(i,j)}^k} \right\}^{-1}.
\]

4. Numerical experiments

In this section, we illustrate the performance of the RRFI algorithm for multiplicative Gamma noise removal of partly-textured images. The test noise free images are shown in Fig. 1. There are high presences of textures combined with non-textured parts in these images. We use different models and algorithms to remove Gamma distributed multiplicative noise with different variances (the noisy images with noise variance \( \sigma^2 = 0.03 \) are shown in Fig. 1). The models and algorithms compared in this section include:

(i) The AA model solved by using the HNW algorithm [20] and the RRFI algorithm with \( \alpha = 1 \) and \( q = 2 \). (\( w^{(n)} \) is updated adaptively according to the Eq. (24)).
(ii) The I-divergence model solved by using the DR S algorithm [5] and the RRFI algorithm with \( \alpha = 1 \) and \( q = 1 \). (\( w^{(n)} \) is updated adaptively according to the Eq. (24)).
(iii) The fractional-order AA model and the fractional-order I-divergence model solved by using the adaptive RRFI algorithm. In this case, \( w^{(n)} \) and \( \alpha \) are computed adaptively according to the Eqs. (24) and (26), respectively. We set \( q = 2 \) for the fractional-order AA model and \( q = 1 \) for the fractional-order I-divergence model.

As the objective evaluation indexes, we use the peak signal to noise ratio (PSNR) and the structural similarity index (SSIM) [26] to evaluate the performance of noise removal and structure preservation, respectively. CPU times used by these algorithms when the iterations are terminated are compared to evaluate the computational efficiency of the algorithms. For all of these algorithms, we use the same termination condition as follows
\[
\frac{||w^{(n)} - w^{(n-1)}||_2}{||w^{(n-1)}||_2} \leq 10^{-4}
\]

To preserve edges and textures as much as possible, we adjust the parameters in the HNW algorithm and the DR S algorithm to achieve the highest SSIM when the iteration is terminated. Moreover, the inner iteration loops in all algorithms in this paper are 10 times. All algorithms are performed under 64 bit Windows 7 and MATLAB R2011b running on a laptop with Intel Core i7-2620M @ 2.70 GHz CPU and 8 GB of memory.

In the first experiment, we deal with the AA model and the fractional-order AA model. For the noisy image showed in Fig. 1, the objective evaluation indexes obtained are showed in Table 1. We illustrate the curves of PSNR, SSIM and the relative error defined by (27) of the denoised images in Fig. 2.

The results in Table 1 and Fig. 2 demonstrate that the RRFI algorithm can improve PSNR and SSIM much better than HNM algorithm, especially for the Barbara image which contains plenty of strong textures. Moreover, convergence speed of the RRFI algorithm is much faster than that of HNW algorithm. We notice that the curves of relative error obtained by the RRFI algorithm and the HNW algorithm have significant difference. In fact, the mechanisms of these two algorithms are different. The HNW algorithm is to remove the noise gradually from the noisy image; the RRFI algorithm is to feed the details of the image back gradually to the cartoon image obtained at the first iteration loop, which is similar to the algorithm proposed in [21]. The result implies that the latter has faster convergence speed.

In Fig. 3, we illustrate the denoised images, flat regions and textured regions of the denoised images obtained by solving the AA model and the fractional-order AA model (the noisy images are showed in Fig. 2(d)). The results in

<table>
<thead>
<tr>
<th>Noisy image</th>
<th>Algorithm</th>
<th>SSIM</th>
<th>PSNR (db)</th>
<th>CPU time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barbara ( \sigma^2 = 0.01 )</td>
<td>DRS algorithm</td>
<td>0.8683</td>
<td>29.6353</td>
<td>32.3546</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm (( \alpha = 1 ))</td>
<td>0.8701</td>
<td>29.9143</td>
<td>21.6373</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8690</td>
<td>29.7582</td>
<td>34.8194</td>
</tr>
<tr>
<td>Barbara ( \sigma^2 = 0.03 )</td>
<td>DRS algorithm</td>
<td>0.7745</td>
<td>26.6889</td>
<td>33.6650</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm (( \alpha = 1 ))</td>
<td>0.8259</td>
<td>27.1588</td>
<td>16.3177</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8293</td>
<td>27.3735</td>
<td>29.8274</td>
</tr>
<tr>
<td>Barbara ( \sigma^2 = 0.1 )</td>
<td>DRS algorithm</td>
<td>0.6724</td>
<td>24.2675</td>
<td>42.1047</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm (( \alpha = 1 ))</td>
<td>0.7106</td>
<td>24.5777</td>
<td>27.8388</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.7204</td>
<td>24.3635</td>
<td>41.9487</td>
</tr>
<tr>
<td>Lena ( \sigma^2 = 0.01 )</td>
<td>DRS algorithm</td>
<td>0.8821</td>
<td>33.1097</td>
<td>40.3887</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm (( \alpha = 1 ))</td>
<td>0.8908</td>
<td>33.6615</td>
<td>18.8605</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8947</td>
<td>33.8864</td>
<td>34.3514</td>
</tr>
<tr>
<td>Lena ( \sigma^2 = 0.03 )</td>
<td>DRS algorithm</td>
<td>0.8493</td>
<td>31.2138</td>
<td>42.4011</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm (( \alpha = 1 ))</td>
<td>0.8517</td>
<td>31.7125</td>
<td>16.3133</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.8586</td>
<td>31.4626</td>
<td>31.1066</td>
</tr>
<tr>
<td>Lena ( \sigma^2 = 0.1 )</td>
<td>DRS algorithm</td>
<td>0.8005</td>
<td>28.2046</td>
<td>47.8143</td>
</tr>
<tr>
<td></td>
<td>RRFI algorithm (( \alpha = 1 ))</td>
<td>0.7900</td>
<td>28.1058</td>
<td>28.4858</td>
</tr>
<tr>
<td></td>
<td>Adaptive RRFI algorithm</td>
<td>0.7967</td>
<td>28.2855</td>
<td>42.5571</td>
</tr>
</tbody>
</table>
Fig. 3 show that by adding the local behavior to the weighting matrix $w(n)$, the RRFI algorithm can preserve textures much better than the HNW algorithm. It is the reason that the RRFI algorithm can improve PSNR and SSIM much better than the HNW algorithm. Moreover, by choosing the parameter $\alpha$ adaptively, we can eliminate the staircase effect efficiently, and therefore the fractional-order AA model can improve the SSIM and PSNR better than the AA model, as showed in Table 1.

In the second experiment, we consider the l-divergence model and the fractional-order l-divergence model. The objective evaluation indexes are compared in Table 2. The results demonstrate that the RRFI algorithm is better than the DRS algorithm. In each iteration loop of the DRS
algorithms, we need to solve two non-linear equations as well, but the computational cost is much lower than that of HNW algorithm. When $\alpha = 1$, the RRFI algorithm can improve the SSIM and the PSNR better with less CPU times than the DRS algorithm in most cases. When we use the adaptive RRFI algorithm to solve the fractional-order I-divergence model, the CPU times used are close to that used by the DRS model. However, the adaptive RRFI algorithm can improve the SSIM and the PSNR much better than the DRS algorithm. Moreover, the parameters are computed adaptively in the adaptive RRFI algorithm, which brings great convenience in practice. Therefore, the adaptive RRFI algorithm is better than the DRS algorithm on the overall performance.

Similar to the Fig. 2, we illustrate the curves of PSNR, SSIM and the relative error of the denoised images of the noisy images in Fig. 4. The result implies that the convergence speed of the RRFI algorithm is close to that of the DRS algorithm. However, as showed in Table 2, when $\alpha = 1$, the CPU times of RRFI algorithm are lower than that of the DRS algorithm. It implies that the RRFI algorithm with $\alpha = 1$ has lower computational cost than the DRS algorithm. Moreover, we notice that the curves of relative error obtained by the RRFI algorithm and the DRS algorithm are similar. In fact, in the DRS algorithm

![Fig. 5. Comparison of the denoised images, flat regions and textured regions of the Barbara image obtained by using the DRS algorithm and RRFI algorithm with $\alpha = 1$ for the I-divergence model, and the adaptive RRFI algorithm for the fractional-order I-divergence model. (a) The DRS algorithm for the I-divergence model. (b) The RRFI algorithm with $\alpha = 1$ for the I-divergence model. (c) The adaptive RRFI algorithm for the I-divergence model.]
proposed in [5], the Bregman iteration is used, which also contains a process of residual feedback.

In Fig. 5, we illustrate the denoised images, flat regions and textured regions obtained by solving the I-divergence model and the fractional-order I-divergence model (the noisy images are showed in Fig. 1(c)). The results demonstrate that the RRFI algorithm can also preserve textures much better than the DRS algorithm, but the staircase effect is still serious in the denoised image obtained by RRFI algorithm with $\alpha = 1$. When we choose the parameter $\alpha$ adaptively, we can eliminate

![Fig. 6. Noise removal for SAR image. (a) Noisy image (256 × 256). (b) The regions for computing ENL. (c) HNW (AA). (d) RRFI with $\alpha = 1$ (AA). (e) Adaptive RRFI (AA). (f) DRS (I-divergence). (g) RRFI with $\alpha = 1$ (I-divergence). (h) Adaptive RRFI (I-divergence).]

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>ENL (A)</th>
<th>ENL (B)</th>
<th>ENL (C)</th>
<th>EPI</th>
<th>CPU time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HNW [20]</td>
<td>102.6070</td>
<td>205.4367</td>
<td>154.0833</td>
<td>0.7448</td>
<td>13.7281</td>
</tr>
<tr>
<td>DRS [5]</td>
<td>102.0949</td>
<td>183.6730</td>
<td>150.6751</td>
<td>0.7651</td>
<td>4.4148</td>
</tr>
<tr>
<td>RRFI($\alpha = 1$,AA)</td>
<td>116.7995</td>
<td>145.2224</td>
<td>326.8407</td>
<td>0.8903</td>
<td>3.1668</td>
</tr>
<tr>
<td>RRFI($\alpha = 1$,I-divergence)</td>
<td>107.7375</td>
<td>133.1310</td>
<td>289.4285</td>
<td>0.8963</td>
<td>3.1356</td>
</tr>
<tr>
<td>Adaptive RRFI (AA)</td>
<td>108.5516</td>
<td>138.3621</td>
<td>283.0077</td>
<td>0.8852</td>
<td>4.4772</td>
</tr>
<tr>
<td>Adaptive RRFI(I-divergence)</td>
<td>101.0890</td>
<td>127.9850</td>
<td>252.9348</td>
<td>0.8908</td>
<td>5.8344</td>
</tr>
</tbody>
</table>
staircase effect efficiently and avoid the blurring of edges caused by the high-order derivative. It is the reason that the adaptive RRFI algorithm can improve the PSNR and SSIM much better than the RRFI algorithm with $\alpha = 1$.

In the last experiment, we remove the speckle noise from a real SAR image showed in Fig. 6(a), where the noise is assumed to follow Gamma distribution. Because the original image is unknown, we cannot compute the PSNR and SSIM. In this case, we use the equivalent number of looks (ENL) and the edge-preserving index (EPI) [27] as the objective evaluation indexes. Larger ENL implies better performance of noise removal in the flat regions, and larger EPI implies better performance of edge-preserving. We compare the ENLs computed in the regions A, B, and C, which are showed in Fig. 6(b).

We illustrate the denoised images in Fig. 6(c)–(h) and evaluation indexes in Table 3. The results demonstrated that the RRFI algorithm can obtain high ENL and high EPI simultaneously with low computational cost. The RRFI algorithm can preserve edges and textures much better than the HNW algorithm and the DRS algorithm. It should be noted that in this experiment, the adaptive RRFI algorithm performs worse slightly than the RRFI algorithm with $\alpha = 1$. It is because the flat regions of the underlying image are piecewise constant, and therefore $\alpha = 1$ will be better in the flat regions. However, the adaptive RRFI algorithm is still much better than the HNW algorithm and the DRS algorithm.

5. Conclusions

In this paper, we propose a simple adaptive reweighted residual-feedback iterative algorithm for fractional-order total variation regularized multiplicative noise removal of partly-textured images. The main contributions of this paper are as follows:

(i) We proposed a reweighted residual-feedback iterative algorithm to solve the fractional-order TV regularized models with different fidelity terms. It is a general framework to solve the TV regularized and fractional-order TV regularized models with different fidelity terms. We provided a sufficient condition for the convergence of this algorithm.

In many problems in image processing such as multiplicative noise removal, image deblurring and image reconstruction etc., the fidelity terms are variant and complex which lead to serious computational difficulties. In general, we need to propose special algorithms to solve these models with certain fidelity terms, such as the HNM algorithm for the AA model and the DRS algorithm for the $l$-divergence model. But in the RRFI algorithm, we just need to estimate the weighting matrix according to the fidelity terms and solve a simple fractional-order TV regularized model quickly by using the extended Chambolle’s algorithms [15]. Therefore, it can reduce the computational difficulties significantly.

(ii) We defined fuzzy membership degrees to mark the possibilities of a pixel belonging to edges, textured regions and flat regions. The fuzzy membership degrees transit gradually at the border areas between the edges, the textured regions and the flat regions, and therefore we can eliminate the fragment effect caused by the hard-thresholding used in [15,22–24].

(iii) We utilize the RRFI algorithm for multiplicative Gamma noise removal. To improve the performance, we propose a strategy to update the weighting matrix and compute the order of derivative adaptively. The RRFI algorithm can be cast as an extension of the iterative regularization method proposed in [21]. The adaptive RRFI algorithm differs from iterative regularization method mainly in that it just adds a weighted residual but not the whole residual back to the noisy image, and therefore the RRFI algorithm can preserve structures well but does not add much noise back in the flat regions. The numerical results show that the RRFI algorithm has low computational cost and fast convergence speed. The adaptive RRFI algorithm can improve the result visually efficiently.

The performance of our algorithm depends on the detection of the edges and the textures in the image. In this paper, we just use some rough detection method such as the Canny edge detector and the local variance. For example, when estimate the textured region, we do not consider different texture features of pixels, such as directions, scales and so on. In our future work, we will improve the structure detector to detect the structures adaptively in different directions and different scales, and then make the algorithm perform better.

Acknowledgments

This work is supported by the National Natural Science Fund of China (Nos. 61101198 and 61071146), the Natural Science Fund of Jiangsu Province of China (No. BK2012800), the China Postdoctoral Science Fund (No. 2012MS11281) and the Jiangsu Planned Projects for Postdoctoral Research Fund (No. 1102064C).

Reference


