NUMERICAL MATHEMATICS: Theory, Methods and Applications
Numer. Math. Theor. Meth. Appl., Vol. **xx**, No. x, pp. 1-23 (200x)

An Adaptive Strategy for the Restoration of Textured Images using Fractional Order Regularization

R. H. Chan[1], A. Lanza[2], S. Morigi[3], and F. Sgallari[2*]

[1] *Department of Mathematics, The Chinese University of Hong Kong, Shatin, Hong Kong, P. R. China*
rchan@math.cuhk.edu.hk
[2] *CIRAM, University of Bologna*
Via Saragozza, 8, Bologna, Italy
fiorella.sgallari@unibo.it, alanza@arces.unibo.it
[3] *Department of Mathematics, University of Bologna*
P.zza Porta San Donato, 5, Bologna, Italy
serena.morigi@unibo.it

Abstract. Total variation regularization has good performance in noise removal and edge preservation but lacks in texture restoration. Here we present a texture-preserving strategy to restore images contaminated by blur and noise. According to a texture detection strategy, we apply spatially adaptive fractional order diffusion. A fast algorithm based on the half-quadratic technique is used to minimize the resulting objective function. Numerical results show the effectiveness of our strategy.

Key words: ill-posed problem, deblurring, fractional order derivatives, regularizing iterative method

1. Introduction

Noise reduction and deblurring are usually used in a pre-processing stage in image restoration to improve image quality. In this paper, we focus on texture preserving restoration of images corrupted by additive noise and spatially-invariant Gaussian blur. The most common image degradation model, where the observed data $f \in \mathbb{R}^{n^2}$ is related to the underlying $n \times n$ image rearranged into a vector $u \in \mathbb{R}^{n^2}$, is

$$f = Bu + e, \tag{1.1}$$

where $e \in \mathbb{R}^{n^2}$ accounts for the random perturbations due to noise, and B is a $n^2 \times n^2$ matrix representing the linear blur operator.

*Corresponding author.

It is well known that restoring the image u is a very ill-conditioned problem and a regularization method should be used. A popular approach determines an approximation of u as the solution of the minimization problem

$$\min_u \{\frac{1}{p}\|Bu - f\|_p^p + \frac{\lambda}{q}\|A(u)\|_q^q\}, \tag{1.2}$$

where A is a regularization operator, and λ is a positive regularization parameter that controls the trade-off between the data fitting term and the regularization term [13, 25, 27]. For $p = 2$ and $q = 2$, we get the classical Tikhonov regularization [11, 13]. This approach enforces smoothness of the solution and suppresses noise by penalizing high-frequency components, thus also image edges can be smoothed out in the process.

Numerous regularization approaches and advances numerical methods have been proposed in the literature to better preserve edges, including alternating minimization algorithms [1], multilevel approaches [16], non-local means filters [4].

A very popular choice in the literature for regularization is based on the total variation (TV) norm. Total variation minimization was originally introduced for noise reduction [25] [7], and has also been used for image deblurring [14] and super-resolution image reconstruction [17]. The TV regularization (ℓ_2-TV) is obtained from (1.2) by setting $p = 1$, $q = 2$, and $A(u)$ the gradient magnitude of u. If we let $\nabla u_i := (G_{x,i}u, G_{y,i}u)^T$, with $G_{x,i}$, $G_{y,i}$ representing the ith rows of the x and y-directional finite difference operators G_x, G_y, respectively, then the regularization term is defined by the TV-norm

$$\|u\|_{TV} = \|A(u)\|_1 := \sum_{i=1}^{n^2} \sqrt{(G_{x,i}u)^2 + (G_{y,i}u)^2}.$$

The distinctive feature of TV regularization is that image edges can be preserved, but the restoration can present staircase effects.

A variant of the ℓ_2-TV regularization is the ℓ_1-TV regularization which is obtained from (1.2) by replacing the ℓ_2 norm in the data-fitting term by ℓ_1 norm:

$$\min_u \{\ \|Bu - f\|_1 \ + \ \lambda \|u\|_{TV}\ \}, \tag{1.3}$$

see [5], [28], [29], [26], [27] for discussions on this model. This model has a number of advantages, including superior performance with non-Gaussian noise such as impulse noise, see [20]. However, it is well known that the ℓ_1-TV regularization has problems in preserving textures, see [28], [6].

In image denoising, recent works have dealt with this drawback mainly by two different strategies. In [10] the ℓ_2-TV model is adopted using a spatially variant regularizing parameter λ, selected according to local variance measures. In [2], a fractional order anisotropic diffusion model is introduced, which leads to a "natural interpolation" between the Perona-Malik equations [22], and fourth-order anisotropic diffusion equations [15]. An adaptive fractional-order multi-scale model is proposed in [31, 32], where the model is applied to noise removal and the texture is detected by a variant of the strategy in [10].

The main goal of this work is to apply an adaptive fractional order regularization term for the restoration of textured images corrupted by additive noise and blur. To achieve this aim, we use a 2-phase approach. First we apply a suitable texture detection method on the observed image to obtain a texture map. Then a fractional order regularization is applied to the parts of the image which are characterized to be texture regions by the map, and the classical TV regularization (ℓ_1-TV) is applied elsewhere. In particular, we propose to replace the TV regularization term $\| u \|_{TV}$ in (1.3) with a spatially adaptive fractional order TV regularization term, thus integrating the following four ingredients:

- use of the fractional order α of derivatives to better preserve textures,
- spatial adaptivity of α in order to allow flexibility in choosing the correct regularizing operator,
- spatial adaptivity of λ in order to locally control the extent of restoration over image regions according to their content,
- an effective texture detection methodology based on the noise auto-correlation energy which makes no assumption about the noise level of the image.

The paper is organized as follows. We describe the proposed adaptive fractional model for image restoration in Section 2. An iterative solution of the proposed model obtained by the half-quadratic strategy and its numerical aspects are discussed in Section 3. In Section 4 we briefly illustrate the computational aspects related to the fractional order derivatives and in Section 5 we describe the texture detection method we used. Numerical examples and comments are provided in Section 6 and the paper is concluded in Section 7.

2. The proposed adaptive model

We propose to modify the functional in (1.2) to the following adaptive fractional variational model

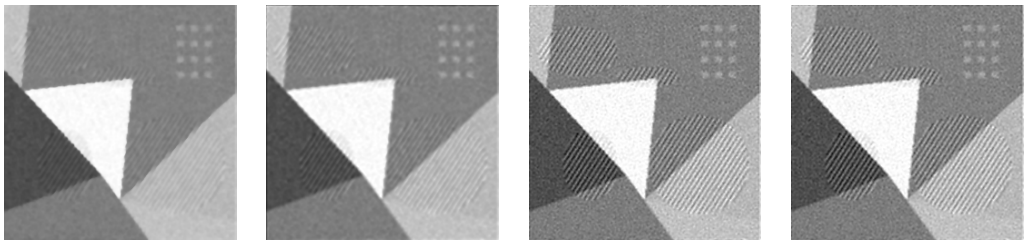
$$\min_u \{ \| Bu - f \|_1 + \| \Lambda \, A_\alpha(u) \|_1 \}, \tag{2.1}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{n^2})$ is an $n^2 \times n^2$ diagonal matrix with λ_i representing the regularization parameter for the ith pixel, $A_\alpha(u_i) = \| \nabla^{\alpha_i} u_i \|_2$, where α_i represents the fractional order of differentiation for the ith pixel, and

$$\nabla^{\alpha_i} u_i := (G^{\alpha_i}_{x,i} u, G^{\alpha_i}_{y,i} u)^T, \tag{2.2}$$

is the fractional-order discrete gradient operator, with components representing the x and y-directional fractional finite difference operators.

Let us motivate our model by analyzing the high-pass filtering character of the fractional order derivative operator. In Figure 1 we show the restored images of the blurred and noisy `test` image in Figure 6(b) by applying model (2.1) with different α



(a) $\alpha = 1.0$ (b) $\alpha = 1.5$ (c) $\alpha = 1.8$ (d) $\alpha = 2.0$

Figure 1: Restoration of `test` image shown in Figure 6(b) using model (2.1) with different values of α while $\lambda = 1.0$ is fixed for all tests.

while keeping $\lambda = 1.0$. With $\alpha = 1$ we get the ℓ_1-TV model (1.3) which preserves edges but fails to preserve fine scale features such as textures in the images, see Figure 1(a). The high-pass capability becomes stronger with larger α, so more and more texture regions are better preserved when α increases to 1.5, 1.8 and then 2.0, see Figures 1(b)–(d) respectively. However, for $\alpha = 2.0$ the high pass capability of fractional order derivatives is too strong so that most components of the image remain and only little high frequencies noise is removed. Thus a best filtering is obtained by adaptively applying the fractional order regularization in texture regions and TV regularization in other non-texture perturbed regions.

Our method consists of two phases. At the first phase, we apply the texture detector proposed in Section 5 to the observed image f to obtain a texture detection map. The texture map contains a texture measure (a real positive value) for each pixel and it is thresholded to obtain the texture and non-texture classes. The texture class is further partitioned into C subclasses according to the texture measure. Following this classification a value α_i is associated to each pixel i, with $\alpha_i = 1$ if the ith pixel belongs to the non-texture class, and $\alpha_i \in \{\hat{\alpha}_1, \ldots, \hat{\alpha}_C\}$ if the ith pixel belongs to one of the C texture subclasses. The regularization parameters λ_i in the diagonal matrix Λ in (2.1) are then chosen according to α_i's. In particular, each texture class has an associated regularization value, while for the non-texture class we set $\lambda = 1.0$. In the second phase, we apply the classical TV regularization (ℓ_1-TV) to the non-texture regions while applying a fractional order TV regularization (ℓ_1-TV$^\alpha$) in the texture class. The corresponding minimization problem is solved by the half-quadratic method in [6].

3. The numerical algorithm

The fidelity and regularization terms in (2.1) are not differentiable, therefore in the following we use a smoothed version of them. To this end, let us define $\|v\|_{1,\epsilon} := \sum_i |v_i|_\epsilon$, with $|v_i|_\epsilon := \sqrt{v_i^2 + \epsilon}$ for any $v_i \in \mathbb{R}$ and $\epsilon > 0$, and let β and γ be two small

regularization parameters. Hence, we want to minimize the functional

$$\begin{aligned}\Phi(u) &= \|Bu - f\|_{1,\gamma} + \|\,\Lambda\, A_\alpha(u)\|_{1,\beta} \\ &= \textstyle\sum_{i=1}^{n^2} |B_i u - f_i|_\gamma + \lambda_i\, |\nabla^{\alpha_i}(u_i)|_\beta \\ &= \textstyle\sum_{i=1}^{n^2} \sqrt{(B_i u - f_i)^2 + \gamma} + \lambda_i \sqrt{(G^{\alpha_i}_{x,i}\, u)^2 + (G^{\alpha_i}_{y,i}\, u)^2 + \beta}\end{aligned} \tag{3.1}$$

where B_i is the ith row of B, f_i is the intensity of the ith pixel of the observed image.

The minimization of the functional (3.1) is obtained in a way similar to what is done for the half quadratic ℓ_1-TV [6, 9, 19]. Half-quadratic regularization is based on the following expression for the modulus of a real, nonzero number x:

$$|x| = \min_{v>0} \left\{ v\, x^2 \,+\, \frac{1}{4\, v} \right\}, \tag{3.2}$$

whose minimum is at $v = \frac{1}{2\,|x|}$, and the function inside the curly bracket in (3.2) is quadratic in x but not in v; hence the name *half-quadratic*. By using (3.2), the minimum of the function in (3.1) can be determined by minimizing the $\mathcal{L}$ operator defined as follows:

$$\begin{aligned}\min_u \Phi(u) = \;& \min_u \textstyle\sum_{i=1}^{n^2} \left[|B_i u - f_i|_\gamma + \lambda_i\, |\nabla^{\alpha_i} u_i|_\beta \right] \\ = \;& \min_u \textstyle\sum_{i=1}^{n^2} \Big[\min_{w_i>0} \left(w_i |B_i u - f_i|^2_\gamma + \frac{1}{4w_i} \right) \\ & \qquad + \lambda_i \min_{v_i>0} \left(v_i\, |\nabla^{\alpha_i} u_i|^2_\beta + \frac{1}{4v_i} \right) \Big] \\ = \;& \min_{u,v>0,w>0} \textstyle\sum_{i=1}^{n^2} \left[w_i |B_i u - f_i|^2_\gamma + \frac{1}{4w_i} + \lambda_i (v_i\, |\nabla^{\alpha_i} u_i|^2_\beta + \frac{1}{4v_i}) \right] \\ := \;& \min_{u,v>0,w>0} \mathcal{L}(u, v, w).\end{aligned} \tag{3.3}$$

In order to solve (3.3), we apply the alternating minimization procedure, namely, for $k = 0, 1, \ldots$, we solve successively

$$\begin{aligned} v^{(k+1)} = \;& \operatorname{argmin}_{v>0} \mathcal{L}(u^{(k)}, v, w^{(k)}) \\ w^{(k+1)} = \;& \operatorname{argmin}_{w>0} \mathcal{L}(u^{(k)}, v^{(k+1)}, w) \\ u^{(k+1)} = \;& \operatorname{argmin}_{u>0} \mathcal{L}(u, v^{(k+1)}, w^{(k+1)}). \end{aligned} \tag{3.4}$$

The first two minimizations in (3.4) have explicit solutions for each iteration:

$$v_i^{(k+1)} = \frac{1}{2} |\nabla^{\alpha_i} u_i^{(k)}|^{-1}_\beta, \qquad w_i^{(k+1)} = \frac{1}{2} |B_i u^{(k)} - f_i|^{-1}_\gamma. \tag{3.5}$$

Since $\mathcal{L}(u, v^{(k+1)}, w^{(k+1)})$ is continuous differentiable in u, the solution $u^{(k+1)}$ of the third minimization in (3.4) is obtained by imposing

$$0 = \nabla_u \mathcal{L}(u, v^{(k+1)}, w^{(k+1)}) = (G^\alpha)^T \widehat{\Lambda}\, \widehat{D}_\beta(u^{(k)})\, G^\alpha u + B^T D_\gamma(u^{(k)})(Bu - f), \tag{3.6}$$

where $G^\alpha := (G^\alpha_x; G^\alpha_y) \in \mathbb{R}^{2n^2 \times n^2}$ is the discretization matrix of the adaptive fractional gradient operator, $\widehat{\Lambda} := \operatorname{diag}(\Lambda, \Lambda) \in \mathbb{R}^{2n^2 \times 2n^2}$ is a diagonal matrix of the adaptive regularization parameters, $\widehat{D}_\beta(u^{(k)}) := \operatorname{diag}(D_\beta(u^{(k)}), D_\beta(u^{(k)})) \in \mathbb{R}^{2n^2 \times 2n^2}$ where

$D_\beta(u^{(k)}) \in \mathbb{R}^{n^2\times n^2}$ and $D_\gamma(u^{(k)}) \in \mathbb{R}^{n^2\times n^2}$ are diagonal matrices with their ith diagonal entries being

$$(D_\beta(u^{(k)}))_i = 2v_i^{(k+1)} = |\nabla^{\alpha_i} u_i^{(k)}|_\beta^{-1}, \tag{3.7}$$

$$(D_\gamma(u^{(k)}))_i = 2w_i^{(k+1)} = |B_i u^{(k)} - f_i|_\gamma^{-1}, \tag{3.8}$$

respectively.

Algorithm 3.1 outlines the computations for the adaptive fractional method for solving (3.3).

Algorithm 3.1. ***Adaptive-Fractional (AF) Algorithm***

Input: *degraded image f, number of texture classes C;*

Output: *approximate solution $u^{(k)}$ of (3.3);*

1. $\{\lambda_i,\ \alpha_i,\ i = 1,\ldots,n^2\} = TD(f, C)$ *compute the texture-adaptive parameters on the degraded image f;*
2. *Initialize the iterative process by setting $u^{(0)} = f$;*
3. *For $k = 1, 2, \ldots$ until convergent, solve*

$$\left[(G^{\boldsymbol{\alpha}})^T \widehat{\Lambda}\, \widehat{D}_\beta(u^{(k)})\, G^{\boldsymbol{\alpha}} + B^T D_\gamma(u^{(k)}) B\right] u^{(k+1)} = B^T D_\gamma(u^{(k)}) f \tag{3.9}$$

endfor

The linear system of equation (3.9) is solved by the conjugate gradient method where we terminate the iterations as soon as the norm of the residual is less than or equal to 10^{-4}. The use of an iterative solver allows us to avoid storing the large dimension matrices B and $G^{\boldsymbol{\alpha}}$; the only requirement is matrix-vector products. In particular, the product which involves matrix B makes use of FFT convolution, while the product by $G^{\boldsymbol{\alpha}}$ is done according to (4.5), described in Section 4. This makes it possible to solve large-scale problems on fairly small computers. The texture detection procedure, step 1 in Algorithm 3.1, will be outlined in Section 5.

The model (2.1) allows the use of the proof techniques in [6] to prove the convergence of sequence $\{u^{(k)}\}$ to the minimum of $\Phi(u)$, as illustrated by the following result.

THEOREM (Convergence). *For the sequence $u^{(k)}$ generated by the half-quadratic AF Algorithm, if*

$$\ker((G^\alpha)^T G^\alpha) \cap \ker(B^T B) = \{\mathbf{0}\} \tag{3.10}$$

we have

- $\{\Phi(u^{(k)})\}$ is monotonic decreasing and convergent;

- $\lim_{k\to\infty} \|u^{(k)} - u^{(k+1)}\|_2 = 0$;
- $\{\Phi(u^{(k)})\}$ converges to the unique minimizer u^* *of* $\Phi(u)$ from any initial guess $u^{(0)}$;

We remark that, in our case, for $\alpha \in [1,2]$, $\ker((G^\alpha)^T G^\alpha)$ is spanned at most by the two vectors: $\mathbf{1}_{n^2}$, an n^2 vector of ones, and $(1,2,\ldots,n^2)$, while the blurring matrix B is a low-pass filter. Thus (3.10) always holds.

4. Fractional-order derivatives of variable α order

In this section, in order to allow for a complete overview of the discrete numerical procedure for computing the matrix-vector product $\left((G^{\boldsymbol{\alpha}})^T \widehat{\Lambda}\, \widehat{D}_\beta\, G^{\boldsymbol{\alpha}}\right) u^{(k+1)}$ in (3.9), we will explain how the fractional order derivatives have been discretized and made spatially-adaptive by using variable fractional order.

Fractional-order derivatives have a long history and can be seen as a generalization of the integer-order derivatives. Many definitions of fractional-order derivative exist, and all are consistent in some respect with the integer-order one. One of the most popular is the frequency domain definition [8, 21].

Since images are functions defined on a bounded domain, the computation of the fractional-order partial derivatives by the discrete Fourier transforms requires that the function must be symmetric and even, so that a prolongation of the image has to be implemented. This yields a high computational cost. Moreover, here we are interested in variable-order fractional differentiation, i.e. in the computation of spatially-adaptive fractional-order derivatives. In this case the fractional order derivatives in the frequency domain should be properly reconsidered.

Hence, following [30], in this paper we use the Grunwald-Letnikov fractional-order derivative [23]. Let an image be an $n \times n$ matrix with i and j denoting the column and the row pixel coordinates. The discrete, fractional-order gradient at a pixel (i,j) is defined as $(\nabla^{\alpha_{i,j}} u)_{i,j} = \left((\Delta_x^{\alpha_{i,j}} u)_{i,j} \,,\, (\Delta_y^{\alpha_{i,j}} u)_{i,j} \right)$ where

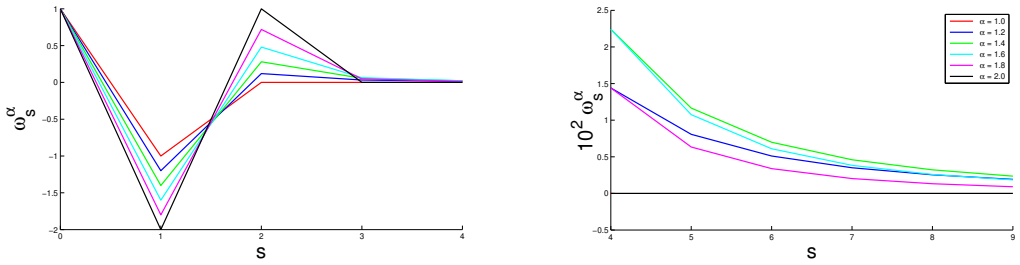
$$\begin{aligned} (\Delta_x^{\alpha_{i,j}}\, u)_{i,j} &= \sum_{s=0}^{K-1} \omega_s^{\alpha_{i,j}}\, u_{i-s,j} \\ (\Delta_y^{\alpha_{i,j}}\, u)_{i,j} &= \sum_{s=0}^{K-1} \omega_s^{\alpha_{i,j}}\, u_{i,j-s} \end{aligned} \qquad \alpha_{i,j} \in \mathbb{R}^+, \tag{4.1}$$

with $K > 0$ being the number of pixels used for the approximation, and ω_s^α, for a generic $\alpha = \alpha_{i,j}$, being the real coefficients defined as

$$\omega_s^\alpha \;=\; (-1)^s \binom{\alpha}{s} = \; (-1)^s \frac{\Gamma(\alpha+1)}{\Gamma(s+1)\Gamma(\alpha-s+1)} \;, \quad \alpha \in \mathbb{R}^+,\, s \in \mathbb{N}. \tag{4.2}$$

The generalized binomial coefficients $\binom{\alpha}{s}$, defined in (4.2) in terms of the Gamma function, can be computed by the following recurrence relationships

$$\binom{\alpha}{0} = 1; \quad \binom{\alpha}{s} = \binom{\alpha}{s-1} \cdot \left(1 - \frac{\alpha+1}{s}\right), \quad \alpha \in \mathbb{R}^+, \; s = 1, 2, \ldots. \tag{4.3}$$



(a) (b)

Figure 2: Values of the coefficients ω_s^α used in (4.1) for the discrete approximation of the fractional-order partial derivatives.

In Figure 2 we show the values of the coefficients ω_s^α used in (4.1) and defined in (4.2)–(4.3) for values of α between 1 and 2 and increasing s values. Only the first ten coefficients $\omega_0^\alpha, \ldots, \omega_9^\alpha$ are reported since they vanish to zero very fast and only $\alpha \in [1, 2]$ values are considered since this is the range we use in AF algorithm. In Figure 2(a) the first fourth s values are shown, while in Figure 2(b), the remaining coefficients are shown using a different plot scale. We remark that, independently on α, the coefficients vanish to zero very fast when s increases, so that a small number of nodes K in (4.1) is sufficient at each pixel for an accurate approximation of the fractional-order gradient.

For $\alpha = 1$ and $\alpha = 2$ the coefficients exactly vanish for $s > 1$ and $s > 2$, respectively, and in fact they reduce to the classical finite difference discretizations of the first and second order derivatives respectively. Finally, we point out that the coefficients sum up to zero independently on $\alpha \in [1, 2]$.

In light of what said, we can make some considerations about the $2n^2 \times n^2$ matrix $G^\alpha = (G_x^\alpha, G_y^\alpha)^T$ in (3.9) when implementing the fractional-order gradient operator. In the simplest case of a non-adaptive α, clearly the computations in (4.1) are nothing more than discrete convolutions of the image with two shift-invariant kernels discretizing the fractional-order partial derivatives. Hence, assuming Dirichlet homogeneous boundary conditions, the two $n^2 \times n^2$ matrices G_x^α and G_y^α are block Toeplitz with Toeplitz blocks, i.e.

$$G_x^\alpha = I_n \otimes U^\alpha, \qquad G_y^\alpha = U^\alpha \otimes I_n, \tag{4.4}$$

where $\otimes$ denotes the Kronecker product, I_n is the nth-order identity matrix and U^α is the $n \times n$ Toeplitz lower triangular banded matrix whose first column is $(\omega_0^\alpha, \omega_1^\alpha, \ldots, \omega_{K-1}^\alpha, 0, \ldots, 0)^T$ with K denoting the number of nodes used for the computation of

the fractional derivatives. Different structured matrices can be obtained for different boundary conditions, see [24].

In the case of variable α, the two matrices G_x^{α} and G_y^{α} retain the same sparsity structure but are no longer Toeplitz: each row will contain different coefficients $\omega_s^{\alpha_{i,j}}$ depending on the fractional-order of differentiation selected for the corresponding pixel. Finally, to compute the matrix-vector product $\left((G^{\boldsymbol{\alpha}})^T \widehat{\Lambda}\, \widehat{D}_\beta\, G^{\boldsymbol{\alpha}}\right) u^{(k+1)}$ in (3.9), let $m = nj + i$ be the lexicographical coordinate of a pixel (i, j), and let $(A)_m$ denote the mth row of the matrix A, then the m-th element of the resulting vector is

$$\begin{aligned} & \left((G^{\boldsymbol{\alpha}})^T \widehat{\Lambda}\, \widehat{D}_\beta\, G^{\boldsymbol{\alpha}}\right)_m u^{(k+1)} \\ = \;& \lambda_{i,j} \left[\sum_{s=0}^{K-1} \omega_s^{\alpha_{i,j}} \frac{(\Delta_x^{\alpha_{i,j}}\, u^{(k+1)})_{i+s,j}}{|\nabla^{\alpha_{i,j}} u^{(k)}_{i+s,j}|_\beta} + \sum_{s=0}^{K-1} \omega_s^{\alpha_{i,j}} \frac{(\Delta_y^{\alpha_{i,j}}\, u^{(k+1)})_{i,j+s}}{|\nabla^{\alpha_{i,j}} u^{(k)}_{i,j+s}|_\beta} \right], \end{aligned} \tag{4.5}$$

with the fractional finite difference operators $\Delta_x^{\alpha_{i,j}}$ and $\Delta_y^{\alpha_{i,j}}$ defined in (4.1).

5. Texture detection method

An automatic texture detection procedure was proposed in [10] where the noise is assumed to be Gaussian with known variance, and the authors use the ℓ_2-TV denoising method to detect texture regions in noisy images. Here, we present a new strategy for computing a measure of texture at each pixel of a degraded image based only on the assumption that the additive noise e in (1.1) is the realization of a white random process. We make no assumption about the noise distribution and the noise variance. We only assume that the noise is not correlated to the image and to itself.

Our idea is to use the auto-correlation function to detect non-whiteness in data.

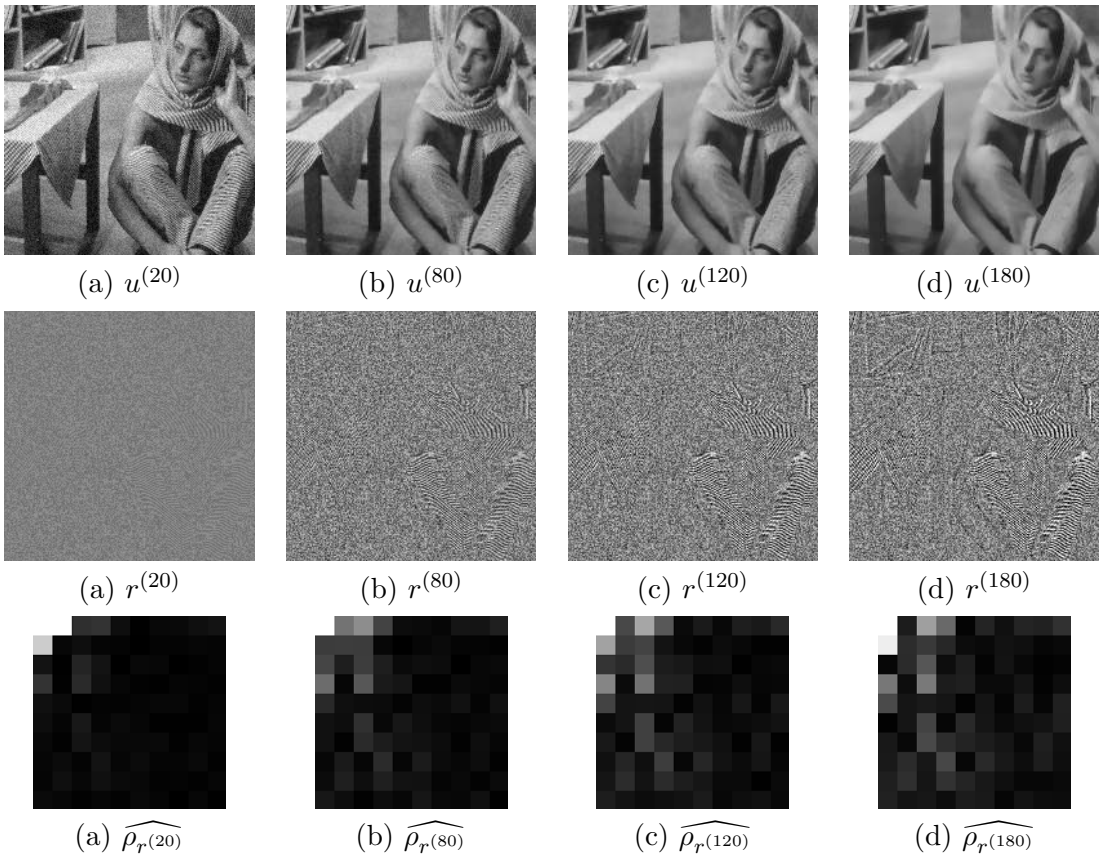
Inspired by [10], starting from the observed degraded image, i.e. $u^{(0)} = f$, we apply a simple TV-flow with Neumann homogeneous boundary conditions

$$u^{(k+1)} \;=\; u^{(k)} + \tau\, \nabla \cdot \left(\frac{\nabla u^{(k)}}{|\nabla u^{(k)}|} \right) \tag{5.1}$$

which approaches a piecewise constant image, so-called "cartoon model", that we denote by $u^{(k)}$. The oscillatory parts removed by the TV flow are the noise e and the "non-cartoon" part, that is the texture parts u_{nc}, which represent small-scale geometric details in the image, see [28] for a detailed analysis of the ℓ_1-TV model for decomposing a real image into the sum of cartoon and texture. Under this decomposition, the residual can be represented as

$$r^{(k)} := f - u^{(k)} = u_{nc} + e.$$

The sequence $u^{(k)}$ converges asymptotically to the constant image with value the mean of the input image f (that we denote by $\overline{f}$), so that the corresponding $r^{(k)}$ approaches



(a) $u^{(20)}$ (b) $u^{(80)}$ (c) $u^{(120)}$ (d) $u^{(180)}$

(a) $r^{(20)}$ (b) $r^{(80)}$ (c) $r^{(120)}$ (d) $r^{(180)}$

(a) $\widehat{\rho_{r^{(20)}}}$ (b) $\widehat{\rho_{r^{(80)}}}$ (c) $\widehat{\rho_{r^{(120)}}}$ (d) $\widehat{\rho_{r^{(180)}}}$

Figure 3: Some steps of the TV-flow applied to the `Barbara` image corrupted by a white Gaussian noise with $\sigma = 20$ (first row); corresponding residual images (second row); corresponding sample auto-correlation, restricted to the smaller lags $[l, m] : l, m = 0, 1, \ldots, 9$ (third row).

to $f - \overline{f}$, We want to stop the TV flow at a characteristic scale $\widetilde{k}$ which allows us to well detect textured parts u_{nc} in the image. Unlike the heuristic criterium adopted in [10], we propose to consider the auto-correlation of the residue $r^{(k)}$ and to choose $\widetilde{k}$ accordingly.

To describe the details of the approach, we briefly introduce some required statistical concepts. Let $\mathcal{E} = \{ E_{i,j} : \ i, j = 1, 2, \ldots n \}$ be an $n \times n$ discrete random field with $E_{i,j}$ denoting the scalar random variable modeling noise at pixel (i, j). The auto-correlation of $\mathcal{E}$ is a function $\rho_{\mathcal{E}}$ mapping pairs of pixel locations (i_1, j_1), (i_2, j_2) into a scalar value that must lie in the range $[-1, 1]$, which represents the Pearson's correlation coefficient between the two corresponding random variables E_{i_1,j_1}, E_{i_2,j_2}, i.e.

$$\rho_{\mathcal{E}}[i_1, j_1, i_2, j_2] \quad = \quad \frac{\mathbf{E}\Big[\big(E_{i_1,j_1} - \mu_{i_1,j_1}\big)\big(E_{i_2,j_2} - \mu_{i_2,j_2}\big)\Big]}{\sigma_{i_1,j_1}\, \sigma_{i_2,j_2}} \tag{5.2}$$

where $\mathbf{E}$ is the expected value operator, $\mu_{i,j}$ and $\sigma_{i,j}$ are the mean and standard devi-

ation of the random variable $E_{i,j}$.

Since we assume that noise is white, i.e. wide-sense stationary, zero-mean, uncorrelated, the auto-correlation of $\mathcal{E}$ depends only on the *lag* between the two pixel locations $[l,m] = (i_2 - i_1, j_2 - j_1)$, and (5.2) can be rewritten as follows

$$\rho_{\mathcal{E}}[l,m] \;=\; \frac{1}{\sigma^2}\mathbf{E}\Big[E_{i,j}E_{i+l,j+m}\Big] = \begin{cases} 1 & \text{if } (l,m) = (0,0), \\ 0 & \text{otherwise,} \end{cases} \qquad l,m = 0,\ldots,n-1, \tag{5.3}$$

independently on i, j. That is, a white noise is characterized by zero values of the auto-correlation function at all non-vanishing lags.

Moreover, assuming that the noise process is also ergodic, provided that the observed realization e of the noise random field $\mathcal{E}$ is "sufficiently long", implies that $\rho_{\mathcal{E}}$ in (5.3) is well estimated by the *sample auto-correlation* function of e defined as

$$\widehat{\rho}_e[l,m] = \frac{1}{n^2\,\widehat{\sigma}^2}\sum_{i,\,j=1}^{n} e_{i,j}e_{i+l,j+m} \tag{5.4}$$

where $\widehat{\sigma}^2$ is the sample variance of the observed noise realization e. We remark that, for a generic observed realization x, the sample auto-correlation $\widehat{\rho_x}[l,m] \in [-1,1]$, with 1 indicating perfect correlation, and -1 indicating perfect anti-correlation.

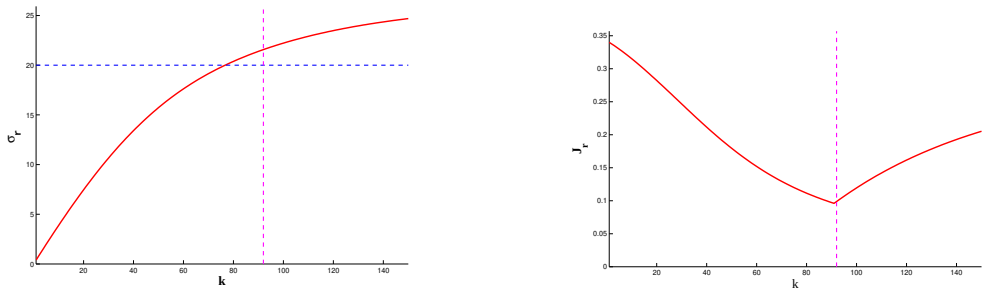
In order to find a characteristic scale $\widetilde{k}$ to detect textures, we propose to minimize the following residual auto-correlation energy

$$J_{r^{(k)}} \;:=\; \max_{[l,m]\neq[0,0]} \Big|\, \widehat{\rho_{r^{(k)}}}[l,m] \,\Big| \tag{5.5}$$

that, according to (5.4), for a cartoon image corrupted by white noise should be zero. For a cartoon image without textures, the energy $J_{r^{(k)}}$ monotonically decreases and vanishes. In the presence of textures, initially, the TV-flow makes the residual image be essentially given by noise, so that the auto-correlation energy $J_{r^{(k)}}$ decreases. As soon as the texture part u_{nc} initiates to contaminate the residual, the energy $J_{r^{(k)}}$ starts increasing since textures are typically correlated.

Our proposal is based on the idea to find the characteristic scale $\widetilde{k}$ which makes the auto-correlation energy of the residual image $J_{r^{(k)}}$ minimal.

To validate this idea, in Figure 3 we report some steps of the TV-flow applied to the `Barbara` image corrupted by additive white Gaussian noise with $\sigma = 20$. In particular, in the first, second and third rows we show, respectively, the images filtered by the TV-flow, $u^{(k)}$, the residual images, $r^{(k)}$, and a zoom of the auto-correlation function of the residual images, $\widehat{\rho_{r^{(k)}}}$, restricted to the smaller lags $[l,m] : l,m = 0,1,\ldots,9$. Moreover, in Figure 4 we report the values of the standard deviation and auto-correlation energy of the residual image versus iterations. Figure 4(b) shows clearly how the auto-correlation energy exhibits a very well-defined first local minimum and looking at Figure 4(a) one can notice that corresponding to this minimum the standard deviation of the residual image is slightly greater than the standard deviation of the noise (horizontal blue line).

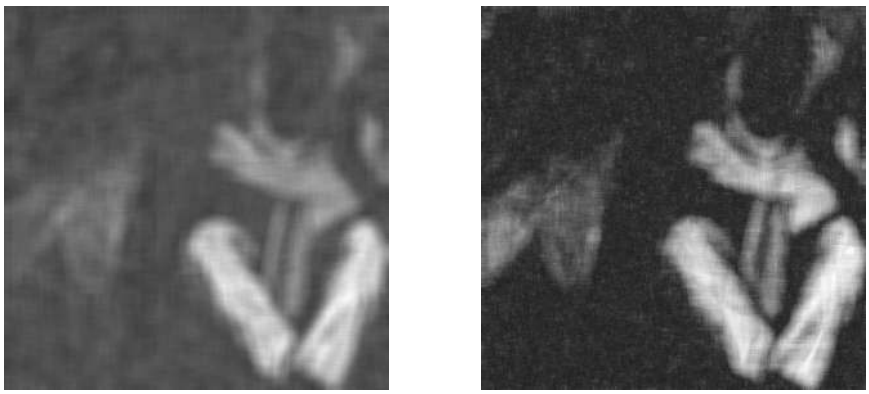


(a) $\sigma_{r(k)}$ (b) $J_{r(k)}$

Figure 4: Standard deviation (a) and auto-correlation energy (b) of the residue.

Once the characteristic scale $\widetilde{k}$ has been selected, we assign as a measure of texture at each pixel the auto-correlation energy of the residual image computed by (5.5) locally in a surrounding neighborhood of the pixel. We name this procedure *Compute-TextureMeasure()* in Algorithm 5.1.

In Figures 5(a) and 5(b) we illustrate the texture maps obtained by applying the method in [10] and our proposal (with $\widetilde{k} = 92$), respectively, on the same corrupted `Barbara` image used in producing the figures in Figure 3. In the implementation of both [10] and our algorithm, we used a 21×21 local neighborhood centered at each pixel for computing variance values and auto-correlation energy, respectively. We remark that our approach does not require any information about the noise distribution and variance, moreover the obtained texture map is of better quality than the one computed by [10]. This can be due to the fact that in [10] the texture map is defined using only the local variance information.



(a) (b)

Figure 5: Texture maps compute by [10] (a) and by our proposed strategy (b). The maps have been linearly scaled so that the maximum value corresponds to the gray value 255.

We finally summarize the proposed texture detection procedure in the following Algorithm.

Algorithm 5.1. ***Texture Detection (TD) Algorithm***

Input: *degraded image f, number of texture classes C;*

Output: *texture-adaptive parameters λ_i, α_i, $i = 1, \ldots, n^2$;*

1. *Initialize the iterative process by setting $u^{(0)} = f$;*
2. *Repeat*
3. *perform one step of the TV flow $u^{(k+1)} = TV(u^{(k)})$ by (5.1)*
4. *compute the residue image $r^{(k+1)} = f - u^{(k+1)}$*
5. *compute the residue auto-correlation $\rho_{r^{(k+1)}}$ by (5.4)*
6. *compute the residue auto-correlation energy $J_{r^{(k+1)}}$ by (5.5)*
7. *until $J_{r^{(k+1)}} > J_{r^{(k)}}$*
8. *$\tilde{k} := k$ characteristic scale found at the first local minimum;*
9. *$T = ComputeTextureMeasure\,(u^{(\tilde{k})})$ with T taking values in [0, 1];*
10. *partition T into C classes $T_1, T_2, \ldots, T_C$;*
11. *assign (λ_i, α_i) according to T_i for $i = 1, \ldots, C$.*

To conclude, we give some details about the computational complexity of Algorithm (5.1). The most demanding steps are step 5 and step 8. As the residue auto-correlation $\rho_{r^{(k)}}$ is computed by (5.4) using a two-dimensional FFT, step 5 exhibits a complexity of $O(\widetilde{k} n^2 \log n^2)$ for a characteristic scale $\widetilde{k}$. Step 8 computes $\rho_{r^{(\tilde{k})}}$ for each pixels locally, i.e. in a surrounding neighborhood of size w (we used $w = 21$), using FFT; hence its complexity is $O(n^2 w^2 \log w^2)$.

6. Computed examples

This section illustrates the performance of the AF algorithm which solves (3.9) when applied to both synthetic and real images that have been contaminated by blur and noise. We compare the results of the AF algorithm with that of the ℓ_1-TV method, which solves (1.3), and ℓ_2-TV method, the well-known Rudin-Osher-Fatemi model [25]. The ℓ_1-TV algorithm solves (1.3) with $\lambda = 0.1$ and it is implemented as the AF algorithm where we set $\alpha_i = 1$, and $\lambda_i = 0.1$, for all i in the difference matrix G^α and in the diagonal matrix Λ, respectively. The ℓ_2-TV algorithm is implemented as the ℓ_1-TV algorithm by setting the matrix D_γ in (3.9) to the identity matrix. In AF algorithm

we fix the regularization parameters $\beta = 10^{-3}$ and $\gamma = 10^{-6}$ in (3.1), and the number of nodes $K = 8$ in (4.1), for all tests. Neumann homogeneous boundary conditions for the difference matrix G^{α} have been considered in all tests, which corresponds to a reflection of the original image at the boundary [18].

We compare the accuracy of the methods by the Signal-to-Noise Ratio (SNR) defined by

$$\mathrm{SNR}(u, \hat{u}) := 10 \log_{10} \frac{\|u - E(\hat{u})\|}{\|u - \hat{u}\|} \,\mathrm{dB},$$

where u is an available approximation of the desired blur- and noise-free image $\hat{u}$, and $E(\hat{u})$ represents the mean gray-level value of the uncorrupted image. This measure provides a quantitative measure of the quality of u. A large SNR-value indicates that u is an accurate approximation of $\hat{u}$; however, the SNR-values are not always in agreement with visual perception.

The matrix B in (1.1) represents a Gaussian blurring operator and is generated by the Matlab function `blur.m` in Regularization Tools [12]. This function has two parameters `band` and `sigma`. The former specifies the half-bandwidth of the Toeplitz blocks and the latter the variance of the Gaussian point spread function. The larger `sigma`, the more blurring. Enlarging `band` increases the storage requirement, the arithmetic work required for the evaluation of matrix-vector products with B, and to some extent the blurring.

The entries of f are contaminated either by additive Gaussian noise or by salt-and-pepper noise. The noise is added to the blurred image to obtain the observed image f. In the salt-and-pepper noise white and black pixels randomly occur, while unaffected pixels always remain unchanged. The salt-and-pepper noise is usually quantified by the percentage of pixels which are corrupted. In the case of Gaussian noise, let $\widetilde{f} \in \mathbb{R}^{n^2}$ be the associated vector with the unknown noise-free entries, i.e., $f = \widetilde{f} + e$ where the vector e represents the noise. We define the noise-level

$$\nu = \frac{\|e\|}{\|\widetilde{f}\|} \tag{6.1}$$

in all the examples.

In these examples we partitioned the texture-map only into four classes, three texture classes and one non-texture class, with associated fractional order and regularization parameter values $\alpha_1 = 1.9$, $\lambda_1 = 0.05$, $\alpha_2 = 1.8$, $\lambda_2 = 0.05$, $\alpha_3 = 1.7$, $\lambda_3 = 0.05$ and $\alpha_4 = 1.0$, $\lambda_4 = 1.0$, respectively. For this particular case, the diagonal entries of the matrix Λ may assume one of the four different values λ_1, λ_2, λ_3 and λ_4.

All computations are carried out in MATLAB with about 16 significant decimal digits.

As concerning the computational complexity of the AF algorithm, sketched in (3.1), we have a preliminary texture detection step (see step 1. in (3.1)), whose complexity has been discussed in Section 5. Then, the core of the algorithm consists of an outer iteration loop (step 3. in (3.1)), and an inner iteration loop required by the iterative

ν	SNR_i	SNR_{AF}	$\text{SNR}_{\ell 1-\text{TV}}$	$\text{SNR}_{\ell 2-\text{TV}}$
0.01	15.98	20.46	20.00	18.22
0.05	13.29	15.86	15.06	15.48
0.10	9.44	13.06	11.56	12.44

Table 1: Example 6.1. Results for restorations of `test` images that have been corrupted by spatially-invariant Gaussian blur, defined by `band` $= 3$ and `sigma` $= 1.5$, and by noise corresponding to noise-level ν.

ν	SNR_i	SNR_{AF}	$\text{SNR}_{\ell 1-\text{TV}}$	$\text{SNR}_{\ell 2-\text{TV}}$
0.01	11.41	14.00	13.53	12.14
0.05	10.71	11.94	10.89	11.64
0.10	9.05	10.59	9.73	10.26

Table 2: Example 6.2. Results for restorations of `barbara` images that have been corrupted by spatially-invariant Gaussian blur, defined by `band` $= 3$ and `sigma` $= 1.5$, and by additive Gaussian noise corresponding to noise-level ν.

solver CG for the linear system in (3.9). From our experience very good results are obtained by at most 10 outer iterations, while the number of inner iterations depends on the required tolerance. In our experiments we used 10^{-4} as stopping tolerance for the CG algorithm, that involves an average of 18 inner iterations. Each iteration in the inner loop requires the evaluation of one matrix-vector product with the matrix which is composed by the sum of two matrices, one related to the regularizer and the other to the blur operator. The implementation details of the former are given in Section 4, while the latter makes use of FFT convolutions.

Example 6.1. We consider the restoration of a blur- and noise-contaminated `test` image represented by 255×255 pixels. The desired blur- and noise-free image is depicted in Figure 6(a). The image is contaminated by a spatially-invariant Gaussian blur, determined by the parameters `band` $= 3$ and `sigma` $= 1.5$, and by 10% noise. The resulting image is displayed in Figure 6(b).

In Table 1 the second column with header SNR_i reports SNR values for images `test` that have been corrupted by the spatially-invariant Gaussian blur, characterized by `band` $= 3$ and `sigma` $= 1.5$, and by noise of different noise-levels ν defined in (6.1). The value of ν are shown in the first column. Column three, labeled SNR_{AF}, shows the SNR values for restorations by AF algorithm, and column four and fifth, labeled $\text{SNR}_{\ell_1-\text{TV}}$, and $\text{SNR}_{\ell_2-\text{TV}}$ show the SNR values for restorations by TV method, implemented by (1.3) with ℓ_1 and ℓ_2 data fidelity terms, respectively.

The adaptive regularization process applies different levels of denoising in different

n	SNR_i	SNR_{AF}	$\text{SNR}_{\ell 1-\text{TV}}$	$\text{SNR}_{\ell 2-\text{TV}}$
5%	3.89dB	14.12	13.39	7.56
10%	1.42dB	13.17	12.93	6.61

Table 3: Example 6.2. Results for restorations of `barbara` images that have been corrupted by spatially-invariant Gaussian blur, defined by `band` $= 3$ and `sigma` $= 1.5$, and by salt-and-pepper noise corresponding to noise-level n.

image regions. This improves the result visually, in terms of texture preservation and smoothing of the homogeneous regions, and in terms of signal-to noise-ratio. Figure 6(d) shows the restoration by AF algorithm and Figure 6(f) depicts the texture-map used for the restoration. The texture measures are obtained on the corrupted image in Figure 6(b) and the four texture classes are shown using gray-level values. In particular, in black regions the AF algorithm is applied with $\alpha_4 = 1.0$, $\alpha_2 = 1.7$, and $\alpha_3 = 1.8$ in the gray regions, and $\alpha_1 = 1.9$ in white regions. The restoration by ℓ_1-TV is illustrated in Figure 6(c) using $\lambda = 0.1$ and it preserves textures but it does not remove the noise. However, when we apply ℓ_2-TV restoration with $\lambda = 1.0$ it does the opposite, that is, it preserves very well the edges but it destroys the textures. The restoration by ℓ_2-TV, illustrated in Figure 6(e), shows less sharpness and smoothing of the texture regions.

Example 6.2. We demonstrate how well our adaptive method performs on natural images by processing a noisy version of the `barbara` image. The noise- and blur-free 510×510 image `barbara` used in this example is shown in Figure 7(a). The corresponding image corrupted by the Gaussian blur, defined by `band` $= 3$ and `sigma` $=$ 1.5, and by 10% of Gaussian noise, is shown in Figure 7(b).

Table 2 reports the SNR values for the contaminated images and for the restorations by the three methods. The table is analogous to Table 1. Figures 7(c) shows the best restoration by the ℓ_1-TV method using $\lambda = 1.0$. Figure 7(d) shows the restoration by AF algorithm using the texture-map depicts in Figure 7(f) which is computed from the observed image in Figure 7(b). Some texture parts of the image are lost in the texture-map, i.e. parts of the pants, and thus these regions cannot benefit from the fractional order restoration. A well-defined texture-map is obtained, and illustrated in Figure 7(e), when the texture-map is computed on the blur- and noise- free image in Figure 7(a). In case severe blur or noise levels corrupt the image, the texture-map can present both erroneously detected texture parts and missing textures. In these cases the performance of AF algorithm reduces to that of a fractional order image restoration without spatial adaptivity.

Table 3 provides a comparison of the three methods when applied to a small part of the `barbara` image corrupted by a salt-and-pepper noise of level n (given in the first column). The ℓ_2-TV algorithm performs well when applied to Gaussian noise removal, as shown in Table 2, but it fails when applied to salt-and-pepper noise removal, as shown in the fifth column in Table 3. Both ℓ_2-TV and ℓ_1-TV implementation used the regularization parameter $\lambda = 0.1$ which produced the best results. The best quality in SNR is obtained with the AF algorithm and it is shown in the third column of Table 3. The image in Fig. 8(a) has been degraded by %5 salt-and-pepper noise and by Gaussian blur, defined by the parameters `band` $= 3$ and `sigma` $= 1.5$, the restored image determined by AF algorithm is illustrated in Fig. 8(b). In Section 5 we showed the texture map obtained by applying the proposed texture detection strategy to an image corrupted by additive Gaussian noise. In Fig. 8(c), the texture map computed on the corrupted image in Fig. 8(a) is illustrated, together with the four textured classes computed from it (Fig. 8(d)).

Example 6.3. As a final example, we show the performance of AF algorithm

when applied to a photographic image, `skyscraper`, of size 256×256, which presents large textured parts. The original unperturbed image is shown in Figure 9(a), and the degraded version of it by the Gaussian blur, defined by `band` $= 3$ and `sigma` $= 1.5$, and by 10% of Gaussian noise, is shown in Figure 9(b). The perturbed image has a qualitative measure $SNR = 8.17$. The restored image determined by AF algorithm is illustrated in Fig. 9(c) and provides a SNR value of 9.79, the ℓ_1-TV restored image is reported in in Fig. 9(d), with SNR equal to 8.44. The λ regularization parameter used in ℓ_1-TV algorithm is 0.5 and we kept the same value for the λ_i corresponding to pixels belonging to the non-textured class when the AF algorithm is applied. The best ℓ_2-TV restoration has been obtained for $\lambda = 0.05$ which gives a restored image with $SNR = 8.65$. The corresponding restored image is visually similar to Fig. 9(d), therefore is not reported. In Fig. 9(f), which illustrates the four classes used, pixels belonging to the non-textured class are black colored. Fig. 9(e) illustrates the texture map computed on the corrupted image shown in Fig. 9(b), which well highlights the textured zones. Consequently, the details present in these textured parts are well reconstructed by AF algorithm as shown in Fig. 9(c).

7. Conclusion remarks

This paper describes a new adaptive method for image deblurring and denoising. The regularization operator is constructed by using fractional order derivatives, where the choice of the fractional order for each pixel in the image is driven by the texture map of the image. Moreover, the regularization parameters are also chosen adaptively according to the texture map. This makes the proposed algorithm an efficient tool to preserve texture well in the texture regions while removing noise and staircase effects in the image. We have developed a simple iterative algorithm to solve the model which is based on the half-quadratic strategy. Numerical results showed that the proposed algorithm yields better signal-to-noise ratio and visual effects than using non-adaptive fractional order image restorations based on TV regularization and ℓ_1 or ℓ_2 data fitting terms.

Increasing the number of texture classes would maintain the same computational cost of the AF algorithm while improving the accuracy in terms of SNR value. We will investigate this important aspect in future experiments.

References

[1] J.O. Abad, S. Morigi, L. Reichel, and F. Sgallari, Alternating Krylov subspace image restoration methods, JCAM, 236(8) (2012), pp.2049-2062.

[2] J. Bai and X.C. Feng, Fractional-order anisotropic diffusion for image denoising, IEEE Trans Image Process. , 16(10) (2007), pp. 2492–502.

[3] D. Bertaccini, R.H. Chan, S. Morigi, and F. Sgallari, An adaptive norm algorithm for image restoration, A.M. Bruckstein et al. (Eds.): SSVM 2011, LNCS 6667, pp. 194–205, Springer-Verlag Berlin Heidelberg, 2011.

[4] A. Buades, B. Coll and J.-M. Morel, A non-local algorithm for image denoising, IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR 2005), San Diego, CA, USA, 2005.
[5] T. Chan and S. Esedoglu, Aspects of total variation regularized L1 function approximation, SIAM J. Appl. Math., 65 (2005), pp. 18171837.
[6] R. H. Chan and H. X. Liang, A fast and efficient half-quadratic algorithm for TV-L1 Image restoration, CHKU research report 370, 2010, submitted; available at `ftp://ftp.math.cuhk.edu.hk/report/2010-03.ps.Z`.
[7] T. F. Chan, S. Osher and J. Shen, The digital TV filter and nonlinear denoising, IEEE Trans. on Image Processing, 10(2), (2001) pp. 231–241.
[8] H. Farid. Discrete-Time Fractional Differentiation from Integer Derivatives. TR2004-528, Department of Computer Science, Dartmouth College, 2004.
[9] D. Geman and C. Yang, Nonlinear image recovery with half-quadratic regularization and FFTs, IEEE Trans. Image Proc., 4, (1995), pp. 932–946.
[10] G. Gilboa, Y. Y. Zeevi, and N. Sochen, Texture preserving variational denoising using adaptive fidelity term, Proc. Variational and Level-Set Methods 2003, Nice, France, pp. 137-144, 2003.
[11] P.C. Hansen, Rank-deficient and Discrete ill-posed problems, SIAM , 1998.
[12] P. C. Hansen, Regularization tools: A Matlab package for analysis and solution of discrete ill-posed problems, Numer. Algorithms, 6 (1994), pp. 1–35.
[13] P. C. Hansen, J. G. Nagy, and D. P. O'Leary, Deblurring Images: Matrices, Spectra, and Filtering, SIAM, Philadelphia, 2006.
[14] Y. Li and F. Santosa, A computational algorithm for minimizing total variation in image restoration, IEEE Trans. on Image Processing, 5(6), (1996), pp. 987–995.
[15] M. Lysaker, A. Lundervold, and X.C. Tai, Noise Removal Using Fourth-Order Partial Differential Equation With Applications to Medical Magnetic Resonance Images in Space and Time, IEEE Trans. on Image Process. , 12(12), (2003), pp.1579–1590.
[16] S. Morigi, L. Reichel, F. Sgallari, and A. Shyshkov, Cascadic multiresolution methods for image deblurring, SIAM J. Imaging Sci., 1 (2008), pp. 51–74.
[17] M.K. Ng,H. Shen,E.Y. Lam,L. Zhang, A total variation regularization based super-resolution reconstruction algorithm for digital video, EURASIP Journal on Advances in Signal Processing, vol. 2007, Article ID 74585 (2007).
[18] M. K. Ng, R. H. Chan, and W.-C. Tang, A fast algorithm for deblurring models with Neumann boundary conditions, SIAM J. Sci. Comput., 21 (1999), pp. 851–866.
[19] M. Nikolova and R. Chan, The equivalence of half-quadratic minimization and the gradient linearization iteration, IEEE Trans. Image Proc.,vol. 16 (2007), pp. 1623–1627.
[20] M. Nikolova, A Variational Approach to Remove Outliers and Impulse Noise, In Journal of Mathematical Imaging and Vision, 20(1-2), (2004), pp. 99-120.
[21] M. Ortigueira, A coherent approach to non integer order derivatives, Signal Processing, 86, (2006), pp. 2505-2515.
[22] P. Perona and J. Malik, Scale-space and edge detection using anisotropic diffusion, IEEE Trans. Pattern Analysis and Machine Intelligence, 12, (1990), pp. 629–639.
[23] I. Podlubny, Fractional Differential Equations, Academic Press, New York, 1999.
[24] I. Podlubny, A. Chechkin, T. Skovranek, Y. Chen, B.M.V. Jara, Matrix approach to discrete fractional calculus II: Partial fractional differential equations, Journal of Computational Physics, 228(8), (2009), pp. 3137–3153.
[25] L. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms, Physica D, 60 (1992), pp. 259–268.

[26] C. Wu, J. Zhang, and X. Tai, Augmented Lagrangian method for total variation restoration with non-quadratic fidelity, UCLA CAM Report 09-82, 2009.

[27] J. Yang, Y. Zhang, and W. Yin, An efficient TVL1 algorithm for deblurring multichannel images corrupted by impulsive noise, SIAM J. Sci. Comput., 31, (2009), pp. 28422865, 2009.

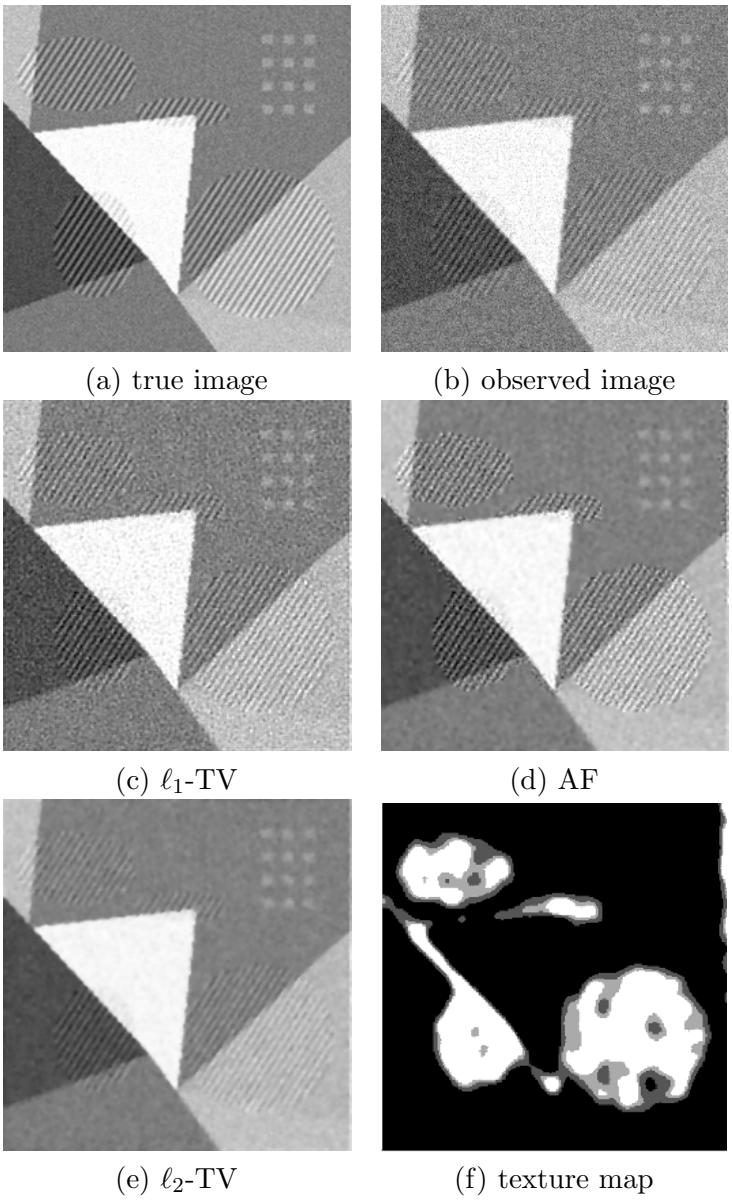
[28] W. Yin, D. Goldfarb, and S. Osher, Image cartoon-texture decomposition and feature selection using the total variation regularized L1 functional, in Variational, Geometric, and Level Set Methods in Computer Vision, Lecture Notes in Computer Science, 3752, pp. 7384, Springer, 2005.

[29] W. Yin, D. Goldfarb, and S. Osher, The total variation regularized L1 model for multiscale decomposition, Multiscale Model. Simul., 6, (2006), pp. 190211.

[30] J. Zhang and Z. Wei, Fractional Variational model and algorithm for image denoising, Proceed. of the Fourth International conference on Natural Computation, IEEE, 2008.

[31] J. Zhang, Z. Wei, and L. Xiao, Adaptive Fractional-order Multi-scale Method for Image Denoising, Journal of Mathematical Imaging and Vision, Springer Netherlands 0924-9907 - Computer Science (2011), pp. 1–11.

[32] J. Zhang and Z. Wei, A class of fractional-order multi-scale variational models and alternating projection algorithm for image denoising Applied Mathematical Modelling, 35(5), (2011), pp. 2516–2528.

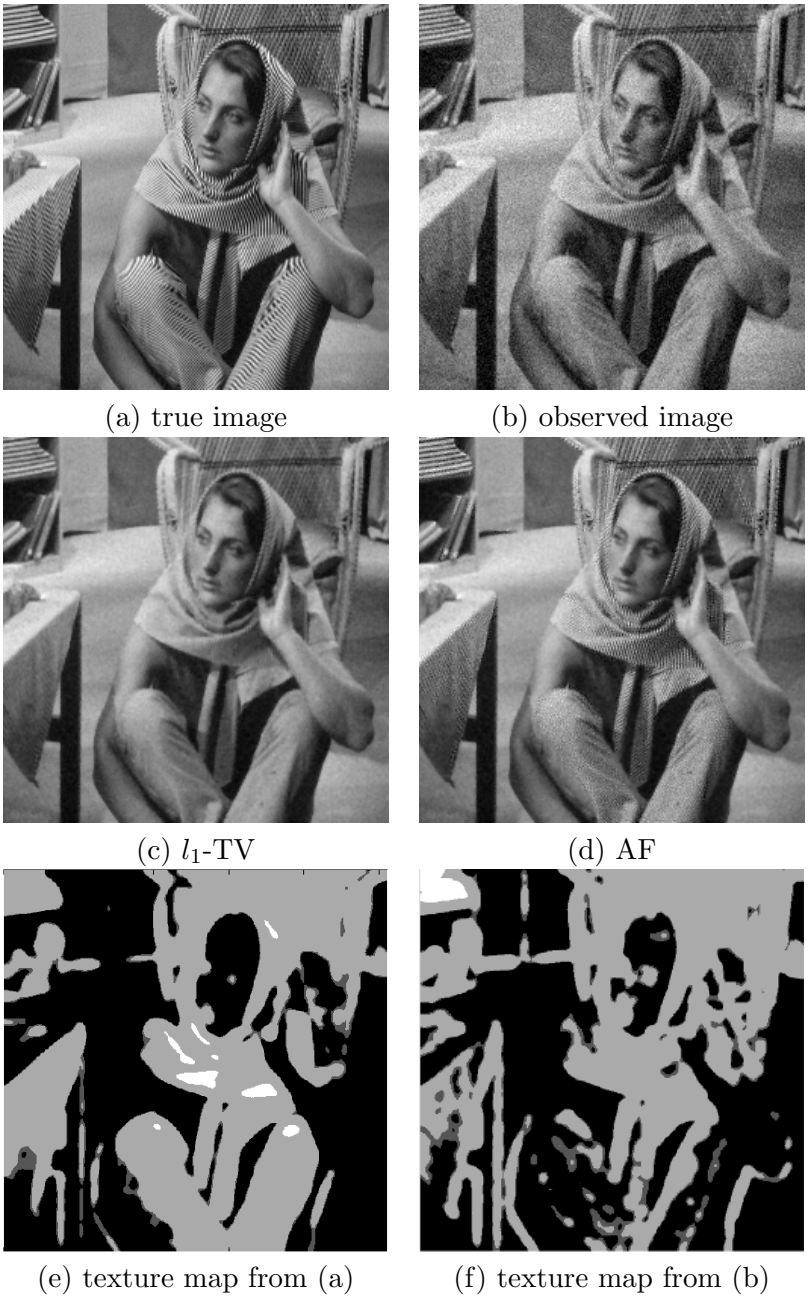


(a) true image (b) observed image

(c) ℓ_1-TV (d) AF

(e) ℓ_2-TV (f) texture map

Figure 6: Example 6.1. `test` images: (a) blur- and noise-free image; (b) the corrupted image produced by Gaussian blur, defined by the parameters `band` $= 3$ and `sigma` $= 1.5$, and by 10% of Gaussian noise; (c) restoration with ℓ_1-TV with $\lambda_1 = 0.1$; (d) restored image determined by AF algorithm; (c) restoration with ℓ_2-TV with $\lambda_1 = 0.1$; (f) texture classes computed on (b).

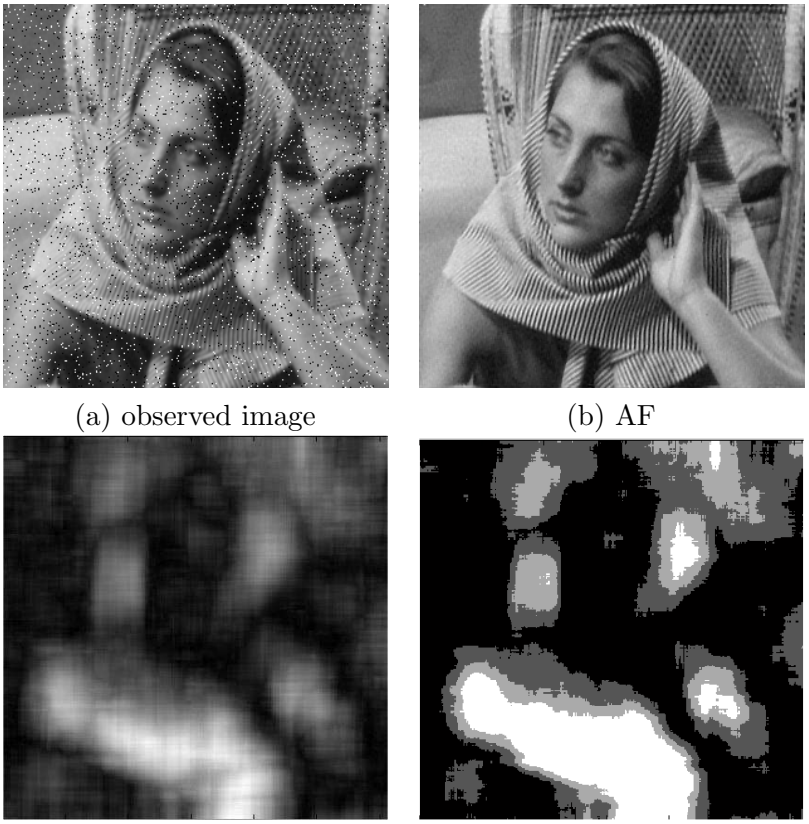


(a) true image (b) observed image

(c) l_1-TV (d) AF

(e) texture map from (a) (f) texture map from (b)

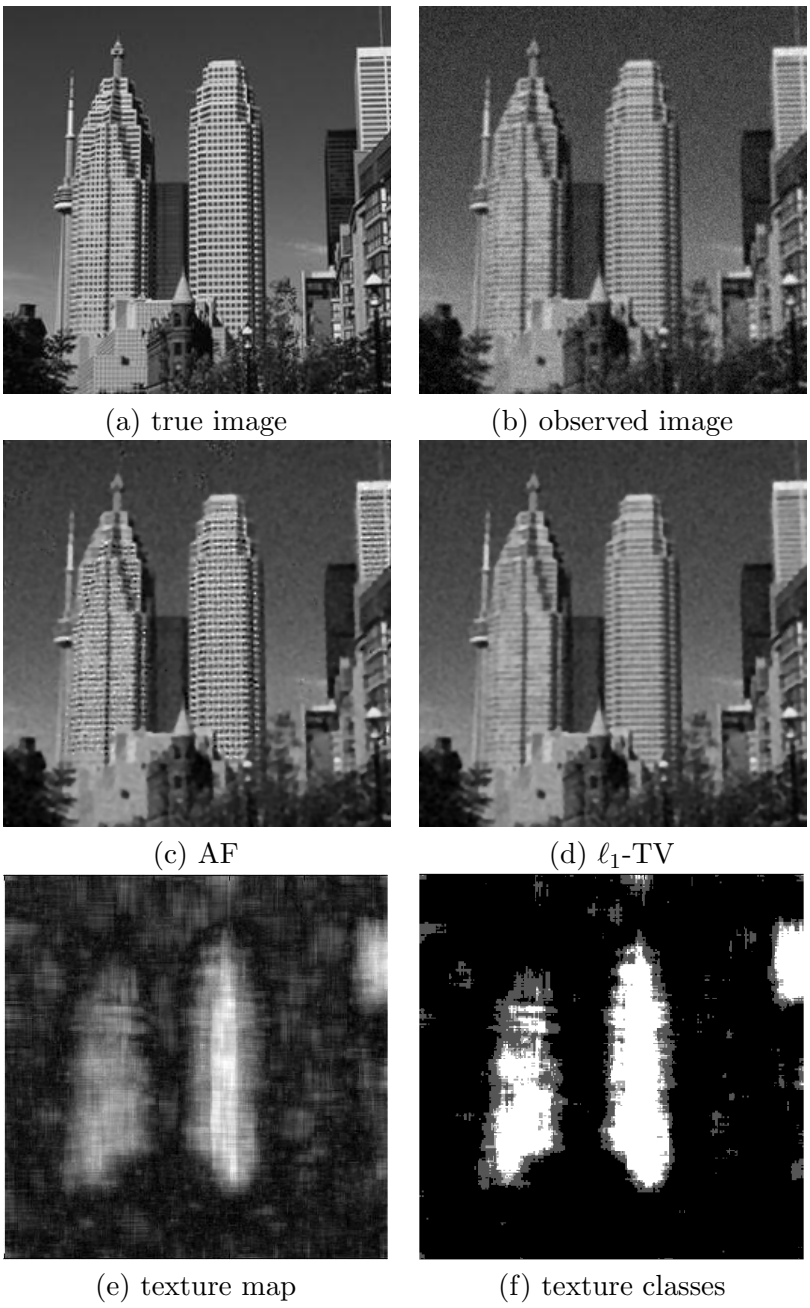
Figure 7: Example 6.2. `barbara` images: (a) blur- and noise-free image; (b) the corrupted image produced by Gaussian blur, defined by the parameters `band` $= 3$ and `sigma` $= 1.5$, and by Gaussian additive 10% noise; (c) restoration with $\ell 1$- TV with $\lambda_1 = 0.1$; (d) restored image determined by AF algorithm; (e) texture classes computed on (a); (f) texture classes computed on (b).



(a) observed image (b) AF

(c) texture map (d) texture classes

Figure 8: Example 6.2. `barbara` images: (a) the corrupted image produced by Gaussian blur, defined by the parameters `band` $= 3$ and `sigma` $= 1.5$, and by 5% salt $\&$ pepper additive noise; (b) restored image determined by AF algorithm; (c) texture map computed on (a); (d) texture classes from (c).



(a) true image (b) observed image

(c) AF (d) ℓ_1-TV

(e) texture map (f) texture classes

Figure 9: Example 6.3. `skyscraper` images: (a) original unperturbed image (b) corrupted image produced by Gaussian blur, defined by the parameters `band` $= 3$ and `sigma` $= 1.5$, and by 10% Gaussian additive noise; (c) restored image determined by AF algorithm; (d) restoration with ℓ_1- TV with $\lambda_1 = 0.5$; (e) texture map computed on (b); (f) texture classes computed on (e).