Second order decomposition model for image processing : numerical experimentation

Maïtine Bergounioux

Maïtine Bergounioux

Second order decomposition model for image processing : numerical experimentation

Abstract: This paper is a companion paper of [8] where a second order image decomposition model to perform denoising and texture extraction has been studied. Here we perform some numerical experimentation to make the behavior of the model as clear as possible. For highly textured images the model gives a two-scale texture decomposition.

Keywords: Second order total variation, image decomposition, variational method, inf-convolution, texture extraction

Classification: 65D18, 68U10, 65K10

1 Introduction

The Rudin-Osher-Fatemi [23] is the most famous denoising variational model. It involves a total variation regularization term that preserves discontinuities. The image to recover/denoise is split in two parts namely the noise and a smooth part (here a function of bounded variation [1, 2, 18]). A lot of people have investigated such decomposition models based on variational formulation, considering that an image can be decomposed into many components, each component describing a particular property of the image ([3, 5, 20, 21, 22, 25] and references therein for example).

In [10] we have presented a second order model where the first order classical total variation term and a second order total variation one are involved. The suitable functional framework is the space of functions with bounded hessian introduced in [17] (see also [9, 10, 7]). To achieve this goal K. Bredies and al. have recently introduced a second order generalized total variation definition [12, 13, 11] which is a nice compromise/mixture between the first and second order derivatives. In [8], we have investigated an inf-convolution type model for texture extraction, mixing first and second order regularization terms. This is

Maïtine Bergounioux: Université d'Orléans , UFR Sciences, Math., Labo. MAPMO, UMR 7349, Route de Chartres, BP 6759, 45067 Orléans cedex 2, France, maitine.bergounioux@univ-orleans.fr

not more efficient than the Total Generalized Variation approach of [12, 13, 11] for denoising. However, it provides a decomposition of the image at different scales what the other model does not a priori. We focused on this decomposition that provides a multiscale description of textured images and gave a rough mathematical analysis of this model in [8].

More precisely, we assumed that an image (in $L^2(\Omega)$) can be split in three components: a smooth (continuous) part v, a *cartoon* (piecewise constant) part u and an oscillating part w that should involve noise and/or fine textures. The oscillating part of the image is included in the remainder term $w = u_d - u - v$, while v is the smooth part (in $BH(\Omega)$) and u belongs to $BV(\Omega)$: we hope u to be piecewise constant so that its jump set gives the image contours. For highly textured images, the model provides a two-scale texture decomposition: u can be viewed as a *macro-texture* (large scale) whose oscillations are not too large and w is the *micro-texture* (much more oscillating) that contains the noise. The same model has been considered in [6] from a different point of view and the numerical realization has been performed with Bregman-type algorithms.

In this paper, we perform numerical experimentations to check the behavior of the method that we do not completely understand. Preliminary results et conjectures have been set in [8]. The aim of these numerical tests is to validate some theoretical results (as uniqueness in particular cases) and reinforce conjectures. From that point of view, the numerical method to compute the solutions is far to be optimal. In particular, the methods we used to compute the different projections are not the most efficient with respect to the Chambolle- Pock algorithm [14] or Bregman-type algorithms.

2 Presentation of the model

We now assume that u_d belongs to $L^2(\Omega)$ and that the image we want to recover can be decomposed as $u_d = w + u + v$ where u, v and w are functions that characterize different parts of u_d. Components belong to different functional spaces: v is the (smooth) second order part and belongs to $BH(\Omega)$, u is a $BV(\Omega)$ component and $w \in L^2(\Omega)$ is the remainder term. We consider the following cost functional defined on $BV(\Omega) \times BH(\Omega)$:

$$\mathcal{F}_{\lambda,\mu}(u,v) = \frac{1}{2}\|u_d - u - v\|^2_{L^2(\Omega)} + \lambda TV(u) + \mu TV^2(v), \tag{1}$$

where $\lambda, \mu > 0$. We are looking for a solution to the optimization problem

$$\inf\{\ \mathcal{F}_{\lambda,\mu}(u,v) \mid (u,v) \in BV(\Omega) \times BH_0(\Omega)\ \} \tag{$\mathcal{P}_{\lambda,\mu}$}$$

We refer to [8] and the references therein for the different spaces definition.

We expect v to be the smooth part of the image, u to be a $BV(\Omega)\backslash BH(\Omega)$ function which derivative is a measure supported by the contours and $w := u_d - u - v \in L^2$ is the noise and/or small textures (we shall detail this point later). We have proved in [8] that problem $(\mathcal{P}_{\lambda,\mu})$ has at least an optimal solution (u^*, v^*) in $BV(\Omega) \times BH_0(\Omega)$. Moreover the dual problem to $(\mathcal{P}_{\lambda,\mu})$ writes

$$\inf_{w \in \lambda\mathcal{K}_1 \cap \mu\mathcal{K}_2} \frac{1}{2}\|u_d - w\|_2^2. \tag{2}$$

where where $\mathcal{K}_1 = \overline{\mathbf{K}_1}$ is the L^2-closure of

$$\mathbf{K}_1 := \left\{\xi = \operatorname{div} \varphi \mid \varphi \in \mathcal{C}_c^1(\Omega),\ \|\varphi\|_\infty \leq 1\right\}. \tag{3}$$

and and $\mathcal{K}_2 \supset \overline{\mathbf{K}_2}$ with $\overline{\mathbf{K}_2}$ is the L^2-closure of

$$\mathbf{K}_2 := \left\{\xi = \operatorname{div}^2 \psi \mid \psi \in \mathcal{C}_c^2(\Omega, \mathbb{R}^{d\times d}),\ \|\psi\|_\infty \leq 1 \right\}. \tag{4}$$

The unique solution w^* is the L^2-projection of u_d on the closed convex set $\lambda\mathcal{K}_1 \cap \mu\mathcal{K}_2$:

$$w^* = \Pi_{\lambda\mathcal{K}_1 \cap \mu\mathcal{K}_2}(u_d)\ .$$

Next we have a relation between the solutions to $(\mathcal{P}_{\lambda,\mu})$ and the (unique) solution of the dual problem.

Theorem 2.1. *1. Let w^* be the (unique) solution to the dual problem $(\mathcal{P}_{\lambda,\mu})^*$:*

$$w^* = \Pi_{\lambda\mathcal{K}_1 \cap \mu\mathcal{K}_2}(u_d)\ .$$

Then there exists $(\bar{u}, \bar{v}) \in BV(\Omega) \times BH_0(\Omega)$ an optimal solution to $(\mathcal{P}_{\lambda,\mu})$ such that

$$w^* = u_d - \bar{u} - \bar{v} \text{ and } w^* \in \partial\Phi_\mu^2(\bar{v}) \cap \partial\Phi_\lambda^1(\bar{u})\ .$$

2. Conversely, if $(\bar{u}, \bar{v}) \in BV(\Omega) \times BH_0(\Omega)$ is any solution to $(\mathcal{P}_{\lambda,\mu})$ then

$$\bar{w} = u_d - \bar{u} - \bar{v} = \Pi_{\lambda\mathcal{K}_1 \cap \mu\mathcal{K}_2}(u_d)\ . \tag{5}$$

Here

$$\Phi_\lambda^1(u) = \begin{cases} \lambda TV(u) & \text{if } u \in BV(\Omega) \\ +\infty & \text{else.} \end{cases} \quad \text{and} \quad \Phi_\mu^2(v) = \begin{cases} \mu TV^2(v) & \text{if } v \in BH_0(\Omega) \\ +\infty & \text{else.} \end{cases}$$

This theorem is useful from many points of view. First, we have a necessary and sufficient condition for a pair (u, v) to be a solution. As we explain in the sequel, this theorem is the key theorem for a fixed point method to compute a solution.

Moreover, solutions (u, v) are not unique but we get uniqueness for the remainder part $w = u_d - u - v$. In addition, we have (partial) results in [8]. We expect numerical results to be consistent and give hints for uniqueness open cases. Let us recall these results thereafter.

Theorem 2.2 ([8] Section 4.2)**.** *1. Assume (u_1, v_1) and (u_2, v_2) are two optimal solutions of $(\mathcal{P}_{\lambda,\mu})$. Then, there exists $\varphi \in BV(\Omega) \cap \mathrm{BH}_0(\Omega)$ such that $u_2 = u_1 - \varphi$ and $v_2 = v_1 + \varphi$.*
2. Let (λ, μ) be nonnegative real numbers such that $\lambda \geq \|u_d\|_G$ and $\mu >> \lambda$, where $\|\cdot\|$ is the Meyer-norm [19]. Then the $\mathbf{C}(\Omega)$ functions are the only solutions to $(\mathcal{P}_{\lambda,\mu})$,where

$$\mathbf{C}(\Omega) := \{(u, v) \in BV(\Omega) \times \mathrm{BH}_0(\Omega) : \exists c \in \mathbb{R}\ u = c \text{ and } v = -c \text{ a.e on } \Omega\ \}.$$

3 Numerical aspects

3.1 Discretized problem and algorithm

We assume that the image is rectangular with size $N \times M$. We note $X := \mathbb{R}^{N\times M} \simeq \mathbb{R}^{NM}$ endowed with the usual (normalized) inner product and the associated Euclidean norm

$$\langle u, v\rangle_X := \frac{1}{NM} \sum_{1\leq i\leq N} \sum_{1\leq j\leq M} u_{i,j} v_{i,j}, \quad \|u\|_X := \sqrt{\frac{1}{NM} \sum_{1\leq i\leq N} \sum_{1\leq j\leq M} u_{i,j}^2}\ . \tag{6}$$

We set $Y = X \times X$. It is classical to define the discrete total variation with finite difference schemes as following (see for example [4]): the discrete gradient of the numerical image $u \in X$ is $\nabla u \in Y$ and may be computed by the following forward scheme for instance:

$$(\nabla u)_{i,j} = \left((\nabla u)^1_{i,j}\,, (\nabla u)^2_{i,j}\right), \tag{7}$$

where

$$(\nabla u)^1_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } 1 < i < N \\ 0 & \text{if } i = 1, N, \end{cases}$$

and

$$(\nabla u)^2_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } 1 < j < M \\ 0 & \text{if } j = 1, M. \end{cases}$$

Note that the constraint $\dfrac{\partial u}{\partial n} = 0$ is involved in the discretization process of the gradient. Therefore, in a discrete setting, the sets $\mathbf{K_2}$ and $\mathcal{K}_2$ coincide. The

(discrete) total variation corresponding to $\Phi_1(u)$ is given by

$$J_1(u) = \frac{1}{NM} \sum_{1\le i\le N} \sum_{1\le j\le M} \left\|(\nabla u)_{i,j}\right\|_{\mathbb{R}^2}, \tag{8}$$

where $\left\|(\nabla u)_{i,j}\right\|_{\mathbb{R}^2} = \left\|\left(\nabla u^1_{i,j}, \nabla u^2_{i,j}\right)\right\|_{\mathbb{R}^2} = \sqrt{\left(\nabla u^1_{i,j}\right)^2 + \left(\nabla u^2_{i,j}\right)^2}$.
The discrete divergence operator -div is the adjoint operator of the gradient operator:

$$\forall (p,u) \in Y \times X, \quad \langle -\text{div } p, u\rangle_X = \langle p, \nabla u\rangle_Y .$$

To define a discrete version of the second order total variation Φ_2 we have to introduce the discrete Hessian operator. For any $v \in X$, the Hessian matrix of v, denoted Hv is identified to a X^4 vector:

$$(Hv)_{i,j} = \left((Hv)^{11}_{i,j}, (Hv)^{12}_{i,j}, (Hv)^{21}_{i,j}, (Hv)^{22}_{i,j}\right).$$

We refer to [10, 9] for the detailed expressions of these quantities. The discrete second order total variation corresponding to $\Phi_2(v)$ writes

$$J_2(v) = \frac{1}{NM} \sum_{1\le i\le N} \sum_{1\le j\le M} \|(Hv)_{i,j}\|_{\mathbb{R}^4}, \tag{9}$$

with

$$\|(Hv)_{i,j}\|_{\mathbb{R}^4} = \sqrt{(Hv^{11}_{i,j})^2 + (Hv^{12}_{i,j})^2 + (Hv^{21}_{i,j})^2 + (Hv^{22}_{i,j})^2}\ .$$

The discretized problem stands

$$\inf_{(u,v)\in X\times X} F_{\lambda,\mu} := \frac{1}{2}\|u_d - u - v\|^2_X + \lambda J_1(u) + \mu J_2(v). \tag{10}$$

Problem (10) has obviously a solution $\tilde{u}$ and $\tilde{v}$ that satisfies the following necessary and sufficient optimality conditions

$$\tilde{u} = u_d - \tilde{v} - \Pi_{\lambda K_1}\left(u_d - \tilde{v}\right)\ , \tag{11a}$$

$$\tilde{v} = u_d - \tilde{u} - \Pi_{\mu K_2}\left(u_d - \tilde{u}\right), \tag{11b}$$

where K_1 and K_2 are the following convex closed subsets :

$$K_1 = \{\text{div } p \mid p \in X^2,\ \|p_{i,j}\|_{\mathbb{R}^2} \le 1\ \forall i = 1,\dots,N, j = 1,\dots,M\}, \tag{12a}$$

$$K_2 = \{H^* p \mid p \in X^4,\ \|p_{i,j}\|_{\mathbb{R}^4} \le 1,\ \forall i = 1,\dots,N, j = 1,\dots,M\}, \tag{12b}$$

and Π_{K_i} denotes the orthogonal projection on K_i. These projections are computed with a Nesterov-type scheme as in [24] where the maximal number of (inner) iterations has been set to 30 and the accuracy to 10^{-7}. We refer to [9]

for more details. As already mentioned our main concern is to check the relevance of the model and highlight some results and/or conjectures about uniqueness or the structure of solutions. Thus, our numerical method is far to be optimal. The numerical implementation could be widely improved using methods as the ones described in [14, 15, 16] for example. This leads to the following fixed-point algorithm:

Algorithm

- **Initialization step.** Choose $u_0,\ v_0$, set $0 < \alpha < 1/2$ and $n = 1$.
- **Iteration.** Define the sequences $((u_n, v_n))_n$ as

$$\begin{cases} u_{n+1} = u_n + \alpha\left(u_d - u_n - v_n - \Pi_{\lambda K_1}\left(u_d - v_n\right)\right) \\ v_{n+1} = v_n + \alpha\left(u_d - u_n - v_n - \Pi_{\mu K_2}\left(u_d - u_n\right)\right). \end{cases}$$

- **Stopping test.** If

$$\max(\|u_{n+1} - u_n\|_{L^2}, \|v_{n+1} - v_n\|_{L^2}) \leq \varepsilon \tag{13}$$

  where $\varepsilon > 0$ is a prescribed tolerance, or if the iterations number is larger than a prescribed maximum number itmax, then STOP.

We have proved in [10] that for any $\alpha \in (0, 1/2)$, the sequence generated by the algorithm converges to a stationary point, solution of (11) that we generically denote (u^*, v^*) in the sequel. The tolerance was set to $\varepsilon = 10^{-2}$ so that the stopping criterion is *de facto* the maximum number of iterations itmax. In the sequel, we have set itmax=10 000 for the 1D case and itmax = 400 for the 2D case.

We do not report on CPU time since all tests have been done with MATLAB© and the code is not optimized. A parallelled C^{++} is version is written that reduces the computational time significantly. Moreover, as explained before, the projections can be computed more efficiently with recent algorithms to spare computation time and improve accuracy; nevertheless, we are interested in the behavior of the model and the optimization of projections computation is not our main concern here.

3.2 Examples

We use 1D and 2D examples.

For the first (1D) example we set $s = s_0 + s_1 + s_2$ on [0,1] with s_0 a white

gaussian noise with standard deviation $\sigma = 0.02$ and

$$s_1 = \begin{cases} 0.4 & \text{on } [\frac{3}{10}, \frac{6}{10}] \\ 0 & \text{elsewhere} \end{cases}, \qquad s_2(x) = \begin{cases} 0.8\,x + 0.2 & \text{on } [0, \frac{1}{2}] \\ -1.2\,(x-1) & \text{elsewhere.} \end{cases}$$

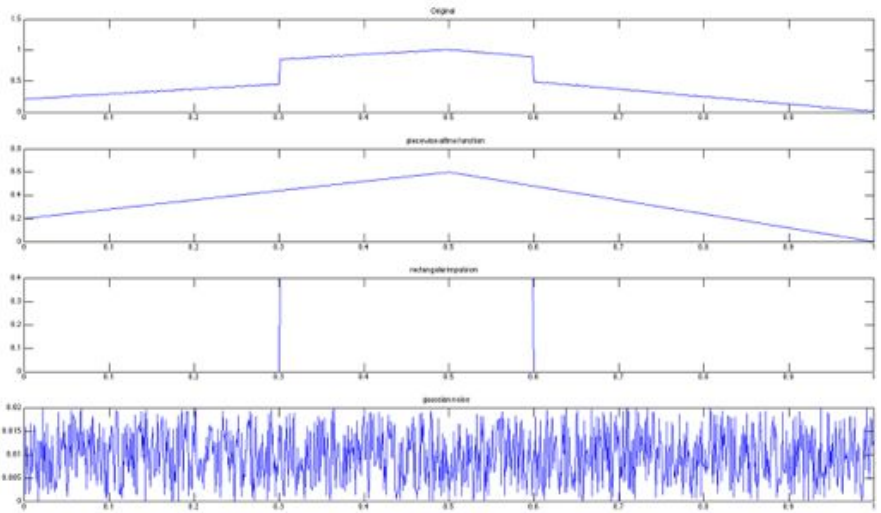
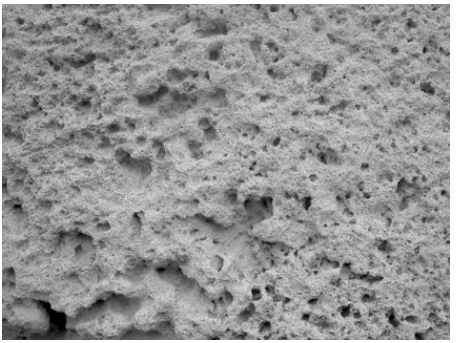


Fig. 1. 1D example - 1000 points. First line : signal s, second line : BH-part s_2, third line : BV-part s_1 and last line ; L^2 noise s_0.

The second example is a 2D picture of a *butterfly* and the third one an highly textured image (old *wall*). We used geometrical images as well but we do not report on them.[1] We present some results and comments in the next subsections.

1 Complete results (text files, movies, other examples) and MATLAB© code, are available at http://maitinebergounioux.net/PagePro/Movies.html.

(a) Test 2D - Butterfly



(b) Test 2D - Wall

Fig. 2. 2D examples

3.3 Initialization process

We have tested many initialization choices for algorithm. Indeed, we have not proved uniqueness (though we conjecture it). So the computed solution is only a stationary point. As we may have many, we may think that the initialization process has a significant influence on the generated sequence.
More precisely, we used

- $u_0 = 0,\ v_0 = u_d$, that we call initialization (a) in the sequel,
- $u_0 = u_d,\ v_0 = 0$ that we call initialization (a') in the sequel,
- $u_0 = 0,\ v_0 = 0$: initialization (b),
- randomized initializations around u_d mean value.

Initialization (a) (resp. (a')) provides a stationary pair (u^*, v^*) such that u^* (resp. v^*) has null mean value.

Proposition 3.1. *Assume $u_0 = 0$ and $v_0 = u_d$. Then any solution (u^*, v^*) given by the algorithm satisfies $\int_\Omega u^* = 0$. Similarly, if $u_0 = u_d$ and $v_0 = 0$, the pair (u^*, v^*) given by the algorithm satisfies $\int_\Omega v^* = 0$.*

Proof. Though we consider a discrete setting we use a continuous setting notation (using for example a piecewise affine approximation). We first note that

$$w \in K_1 \cup K_2 \Longrightarrow \int_\Omega w = 0 \ .$$

We prove the first assertion. Assume that $u_0 = 0$ and $v_0 = u_d$. It is easy to see by induction that

$$\forall n \in \mathbb{N} \qquad \int_\Omega u_n = 0 \text{ and } \int_\Omega (v_n - u_d) = 0 \ . \tag{14}$$

using

$$\begin{cases} u_{n+1} = u_n + \alpha \left(u_d - u_n - v_n - \Pi_{\lambda K_1} \left(u_d - v_n\right)\right) \\ v_{n+1} = v_n + \alpha \left(u_d - u_n - v_n - \Pi_{\mu K_2} \left(u_d - u_n\right)\right) . \end{cases}$$

Passing to the limit we get

$$\int_\Omega u^* = 0 \text{ and } \int_\Omega (v^* - u_d) = 0 \ .$$

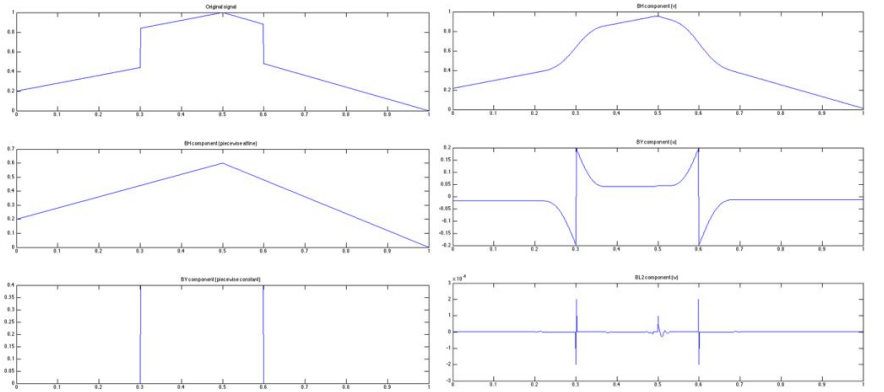
The second assertion is proved similarly. □

Proposition 3.1 yields that the BV- part u^* (or the BH- part v^*) belongs to the discrete Meyer space G (see [5]) if we perform the appropriate initialization step. This means it is an oscillating function. More precisely, choosing $u_0 = u_d,\ v_0 = 0$ gives a BH- part that belongs to G. This is not what we want, since the BH- part should not be oscillating. Therefore, we shall never use such an initialization.
Initializations (a) and (a') seem to give different results from initialization (b). We shall see in the sequel that the difference is *small* if the iteration number is large enough. Therefore, we think that the initial guess has no influence on the result, but only on the convergence speed.

Initialization	$F_{\lambda,\mu}(u^*, v^*)$	$\|w^*\|_{L^2}$	$TV(u^*)$	$TV^2(v^*)$
$\lambda = 10^{-3},\ \mu = 10^{-2}$				
$u_0 = 0,\ v_0 = u_d$	6.401836e-06	1.424060e-03	5.134659e-03	2.532040e-05
$u_0 = u_d,\ v_0 = 0$	6.242638e-06	1.434416e-03	5.076678e-03	1.371843e-05
$u_0 = 0,\ v_0 = 0$	6.235598e-06	1.437834e-03	5.022263e-03	1.796523e-05
Random	6.226676e-06	1.441939e-03	5.007722e-03	1.793599e-05
$\lambda = 10^{-2},\ \mu = 10^{-2}$				
$u_0 = 0,\ v_0 = u_d$	2.230749e-05	4.897618e-03	8.834750e-04	1.479412e-04
$u_0 = u_d,\ v_0 = 0$	2.201165e-05	4.972975e-03	8.022872e-04	1.623540e-04
$u_0 = 0,\ v_0 = 0$	2.200473e-05	4.977910e-03	7.968618e-04	1.646320e-04
Random	2.200287e-05	4.978969e-03	7.959566e-04	1.648234e-04
$\lambda = 1,\ \mu = 10^{-1}$				
$u_0 = 0,\ v_0 = u_d$	1.144536e-04	1.027027e-02	8.406575e-06	5.330783e-04
$u_0 = u_d,\ v_0 = 0$	1.144536e-04	1.027027e-02	8.406575e-06	5.330783e-04
$u_0 = 0,\ v_0 = 0$	1.144536e-04	1.027027e-02	8.406575e-06	5.330783e-04
Random	1.144536e-04	1.027027e-02	8.406575e-06	5.330783e-04
$\lambda = 10^{-1},\ \mu = 1$				
$u_0 = 0,\ v_0 = u_d$	1.716468e-04	6.577422e-03	6.381764e-04	8.619797e-05
$u_0 = u_d,\ v_0 = 0$	1.716467e-04	6.577421e-03	6.381750e-04	8.619794e-05
$u_0 = 0,\ v_0 = 0$	1.716468e-04	6.577422e-03	6.381758e-04	: 8.619796e-05
Random	1.716468e-04	6.577422e-03	6.381759e-04	8.619796e-05

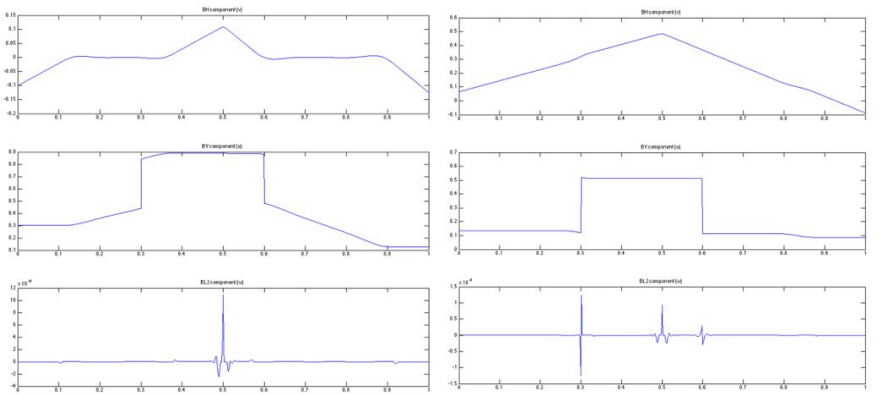
Table 1. Comparison of different initializations - 1D case with noise - itmax=50 000 -The stationary pair is denoted (u^*, v^*) and $w^* = u_d - u^* - v^*$. - The number of inner iterations (in the projections) has been set to 100, to achieve the best accuracy as possible.

We can see on Figure 3 (1D) the oscillating effect of initialization $u_0 = 0$, $v_0 = u_d$:



(a) Original signal (without noise) - $F_{\lambda,\mu}(u^*, v^*) = 8.19e-07$

(b) Initialization $u_0 = 0,\ v_0 = u_d$ (a) - $F_{\lambda,\mu}(u^*, v^*) = 1.65e-06$



(c) Initialization $u_0 = u_d,\ v_0 = 0$ (a') - $F_{\lambda,\mu}(u^*, v^*) = 1.43e-06$

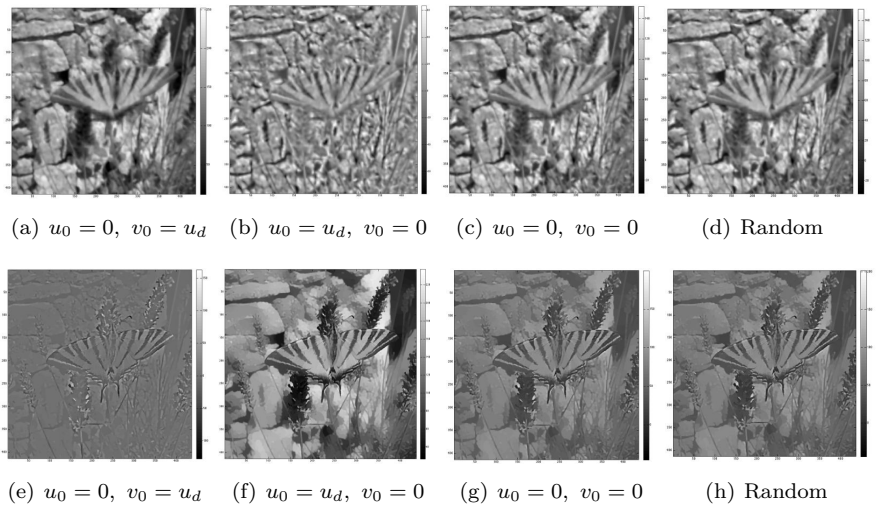
(d) Initialization $u_0 = 0,\ v_0 = 0$ (b) - $F_{\lambda,\mu}(u^*, v^*) = 8.95-07$

Fig. 3. Example 1D without noise ($s_1 + s_2$). The number of inner iterations (to compute projections) is 100 and the number of outer iterations is 50 000. We present the case where $\lambda = 10^{-3},\ \mu = 10^{-2}$ with different initializations. Note that both u^* and w^* have null mean value for initialization (a) ($u_0 = 0,\ v_0 = u_d$). We recover the original decomposition with initialization (b) ($u_0 = 0,\ v_0 = 0$). Initialization (a') ($u_0 = u_d,\ v_0 = 0$) gives a different solution : we shall comment in the sequel.

Figure 4 and Table 2 give the computed pairs with initializations (a) (a') (b) and a randomized initialization around the mean value of u_d for a 2D example.

Initialization	$F_{\lambda,\mu}(u^*,v^*)$	$\|w^*\|_{L^2}$	$TV(u^*)$	$TV^2(v^*)$	Error	# it.
$\lambda=1,\ \mu=10$						
$u_0=0,\ v_0=u_d$	23.68	1.04	12.70	1.04	2.35	400
$u_0=u_d,\ v_0=0$	18.29	1	13.43	0.43	0.73	400
$u_0=0,\ v_0=0$	20.36	1.03	12.87	0.69	0.89	400
Random	20.39	1.03	12.88	0.69	0.87	400
$\lambda=2,\ \mu=0.1$						
$u_0=0,\ v_0=u_d$	1.5414	2.24 e-01	3.64 e-04	15.15	8.48 e-03	22
$u_0=u_d,\ v_0=0$	8.0239	2.76 e-01	3.31	13.52	4.35	400
$u_0=0,\ v_0=0$	3.3335	2.45 e-01	0.92	14.65	3.12	400
Random	3.5384	3.18 e-01	1.02	14.62	3.30	400
$\lambda=5,\ \mu=7$						
$u_0=0,\ v_0=u_d$	61.7005	4.22	5.71	3.45	1.67	400
$u_0=u_d,\ v_0=0$	62.7803	4.02	7.29	2.60	3.25	400
$u_0=0,\ v_0=0$	61.6248	4.15	6.34	3.04	1.50	400
Random	61.6331	4.15	6.35	3.04	1.55	400
$\lambda=7,\ \mu=7$						
$u_0=0,\ v_0=u_d$	69.6775	5.23	2.29	5.69	9.76 e-01	400
$u_0=u_d,\ v_0=0$	72.6262	4.96	4.09	4.51	4.74	400
$u_0=0,\ v_0=0$	70.2957	5.13	2.97	5.19	2.45	400
Random	70.3114	5.12	2.98	5.18	2.52	400
$\lambda=7,\ \mu=9$						
$u_0=0,\ v_0=u_d$	79.7064	5.42	4.10	4.03	1.33	400
$u_0=u_d,\ v_0=0$	80.1229	5.18	5.40	3.33	4.09	400
$u_0=0,\ v_0=0$	79.8224	5.33	4.58	3.72	1.86	400
Random	79.8297	5.33	4.58	3.72	1.89	400
$\lambda=10,\ \mu=15$						
$u_0=0,\ v_0=u_d$	116.2130	7.04	3.59	3.69	1.39	400
$u_0=u_d,\ v_0=0$	116.9598	6.79	4.33	3.36	4.67	400
$u_0=0,\ v_0=0$	116.0822	6.95	3.83	3.56	2.02	400
Random	116.0918	6.95	3.84	3.56	2.10	400

Table 2. Comparison of different initializations (Butterfly)- The number of outer iterations is limited to itmax=400 -The stationary pair is denoted (u^*,v^*) and $w^*=u_d-u^*-v^*$. The error reported in column 5 is defined as in (13) (and should be less that 10^{-2}).



(a) $u_0 = 0,\ v_0 = u_d$ (b) $u_0 = u_d,\ v_0 = 0$ (c) $u_0 = 0,\ v_0 = 0$ (d) Random

(e) $u_0 = 0,\ v_0 = u_d$ (f) $u_0 = u_d,\ v_0 = 0$ (g) $u_0 = 0,\ v_0 = 0$ (h) Random

Fig. 4. BH-part v (first line) and BV-part u (second line) given by initializations (a), (a'), (b) and random for $\lambda = 5,\ \mu = 7$ - *Butterfly* example with 400 iterations.

The blue (grey) lines of Table 2 show what we call the *optimal* solution, that is the computed pair whose cost functional value is the lowest. We observed that

- the randomized initialization gives the same result as initialization (b),
- the component $w^* = u_d - u^* - v^*$ is always the same, which is consistent with the theoretical result of uniqueness,
- the values of the cost functional may be quite close and the computed pairs quite different: see for example $\lambda = 5, \mu = 7$ (and figure 4),
- initialization (b) gives a pair (u_b, v_b) such that neither u_b nor v_b has null mean value.

In the sequel, (u_a, v_a) denotes the pair given by the algorithm with initialization (a) and (u_b, v_b) the one given by the algorithm with initialization (b). Moreover, we set the signed relative error as

$$\delta\mathcal{F}_{\lambda,\mu} = \frac{F_{\lambda,\mu}(u_a, v_a) - F_{\lambda,\mu}(u_b, v_b)}{\min(F_{\lambda,\mu}(u_a, v_a), F_{\lambda,\mu}(u_b, v_b))}. \tag{15}$$

Figures 5 and 6 show the behavior of $\delta\mathcal{F}_{\lambda,\mu}$ with respect to λ and μ.

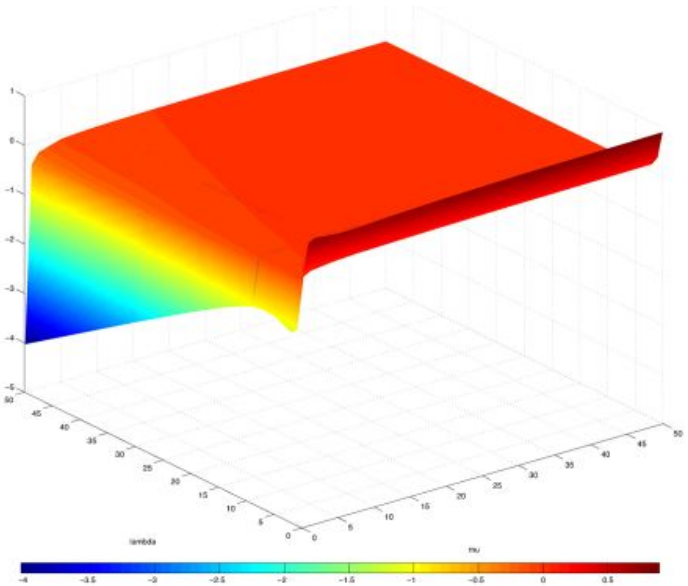


Fig. 5. Behavior of $\delta\mathcal{F}_{\lambda,\mu}$ for 400 iterations (*Butterfly* example) with respect to λ and μ. If λ and μ are large enough ($\lambda > 0.1$ and $\mu > 0.1$ for example), both optimal values are very close.

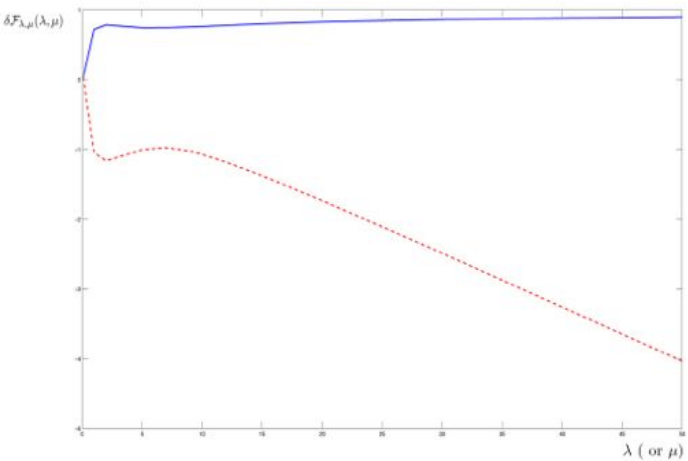


Fig. 6. Behavior of $\delta\mathcal{F}_{\lambda,\mu}$ for 400 iterations (*Butterfly* example). Slices $\lambda = 0.1$ (red dotted line) and $\mu = 0.1$ (blue solid line) .

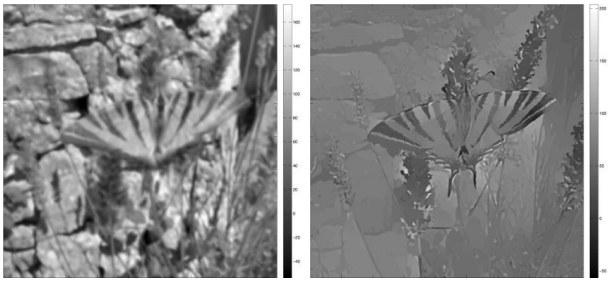
# it.	$F_{\lambda,\mu}(u_a, v_a)$	$F_{\lambda,\mu}(u_b, v_b)$	- $log_{10}\left(\lvert\delta\mathcal{F}_{\lambda,\mu}\rvert\right)$
50	82.38439	81.69328	2.072
100	80.9713	80.9679	4.379
200	80.18443	80.35509	2.672
400	79.83481	79.94497	2.860
600	79.73564	79.80224	3.078
800	79.68948	79.73411	3.252
1000	79.66213	79.69571	3.375
1200	79.64396	79.67121	3.466
1500	79.62567	79.64659	3.580
5000	79.5738	79.5718	4.618

Table 3. Cost functional and relative error (log scale) for pairs given by initializations (a) and (b) for $\lambda = 7,\ \mu = 9$, as the number of iterations increases. We observe convergence : the cost functional is exponentially decreasing.

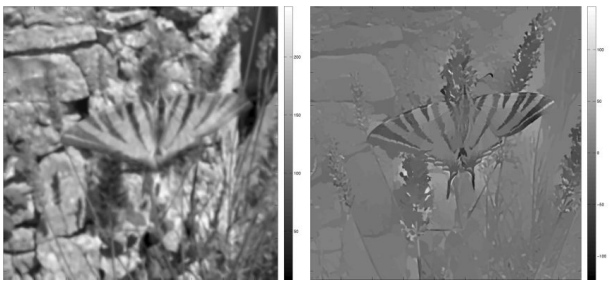
# it.	$TV(u_a)$	$TV(u_b)$	$TV(\varphi)$	$TV^2(v_a)$	$TV^2(v_b)$	$TV^2(\varphi)$	Error (a)	Error (b)
50	2.45	5.62	3.57	5.40	3.20	4.05	7.41	7.14
100	3.29	5.31	2.53	4.69	3.33	2.82	4.88	5.03
200	3.85	4.93	1.66	4.23	3.52	1.87	2.87	3.26
400	4.12	4.60	1.03	4.01	3.70	1.18	1.34	1.87
600	4.19	4.47	0.771	3.95	3.77	0.896	1.02	1.32
800	4.22	4.40	0.628	3.93	3.81	0.736	0.855	1.03
1000	4.23	4.37	0.536	3.92	3.83	0.632	0.735	0.845
1200	4.24	4.35	0.470	3.91	3.84	0.556	0.642	0.723
1500	4.24	4.33	0.396	3.90	3.86	0.472	0.535	0.595
5000	4.28	4.26	0.148	3.88	3.89	0.180	0.207	0.208

Table 4. First and second order total variations TV and TV^2 of pairs given by initializations (a) and (b) for $\lambda = 7,\ \mu = 9$, as the number of iterations increases. Here $\varphi = u_b - u_a = v_b - v_a$ and the error is given by the stopping criterion (13) of Algorithm.

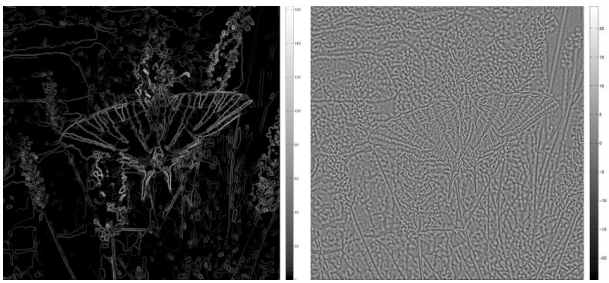
Though $F_{\lambda,\mu}(u_a, v_a) \simeq F_{\lambda,\mu}(u_b, v_b)$ the pairs (u_a, v_a) and (u_b, v_b) may be very different. More precisely, we have $u_b = u_a - \varphi$ and $v_b = v_a + \varphi$. Though the computed function φ_k at iteration k is not a constant function (see Figure 7), we infer that φ_k converges to a constant function as the iteration number increases. Indeed, we have numerically observed (see Table 4) that both $TV(\varphi)$ and $TV^2(\varphi)$ decreases to 0 as the iteration number increases. Nevertheless, we can perform only a limited number of iterations. So the computed solutions differ from a (small) piecewise constant function (see Figure 7). In addition, it is numerically confirmed that $w_a = u_d - u_a - v_a = w_b$ (what was theoretically proved).



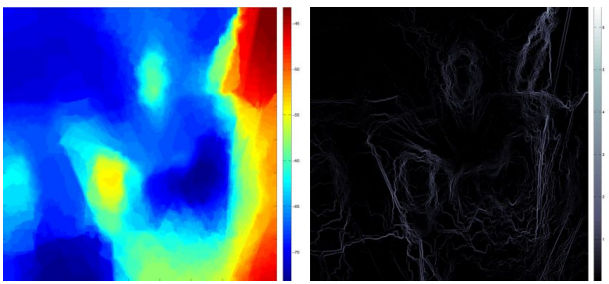
(a) v_a - init : $u_0 = 0,\ v_0 = u_d$ (b) u_a - init : $u_0 = 0,\ v_0 = u_d$



(c) v_b - init : $u_0 = 0,\ v_0 = 0$ (d) u_b - init : $u_0 = 0,\ v_0 = 0$



(e) Norm of u_b gradient (f) $w_b = w_a$



(g) $\varphi = u_b - u_a = v_a - v_b$ (h) Norm of φ gradient

Fig. 7. Difference between the solutions given by initializations (a) and (b) for $\lambda = 7, \mu = 9$ and 5000 iterations . $\|\varphi\|_2 = 0.1518,\ TV(\varphi) = 0.1484,\ TV^2(\varphi) = 0.1803$. The function φ is close to be piecewise constant as we see it on the gradient norm. This is consistent with Theorem 2.2

3.4 Convergence

We chose $\alpha = 0.25$ in the fixed point algorithm and we always observed convergence. We set the maximal number of iterations quite large but we noticed that the solution is satisfactory with less iterations (400 for 2D case and 1000 for 1D case).

# it.	$F_{\lambda,\mu}(u_a, v_a)$	$F_{\lambda,\mu}(u_b, v_b)$	$\lvert\delta\mathcal{F}_{\lambda,\mu}\rvert$
$\lambda = 1,\ \mu = 10$			
50	39.132	25.513	5 e-01
100	31.727	23.113	3.7 e-01
200	26.907	21.440	2.5 e-01
400	23.711	20.377	1.6 e-01
600	22.410	19.978	1.2 e-01
800	21.688	19.774	9.6 e-02
$\lambda = 10,\ \mu = 15$			
50	119.448	117.102	2 e-02
100	117.578	116.601	8.3 e-03
200	116.612	116.257	3 e-03
400	116.215	116.083	1.1 e-03
600	116.106	116.031	6.5 e-04
800	116.052	116.006	4 e-04
$\lambda = 10,\ \mu = 2$			
50	25.90989	39.586	5.2 e-01
100	25.91003	33.501	2.9 e-01
200	25.91008	29.512	1.4 e-01
400	25.91009	27.558	6.3 e-02
600	25.91009	26.986	4.1 e-02
800	25.91009	26.699	3 e-02

Table 5. Sensitivity with respect to number of iterations : cost functional value. On can refer to Table 3 as well.

Figures 8-10 illustrate the generic behavior of the cost-functional $F_{\lambda,\mu}$.[2]

2 One can look at http://maitinebergounioux.net/PagePro/Movies.html to see the convergence process.

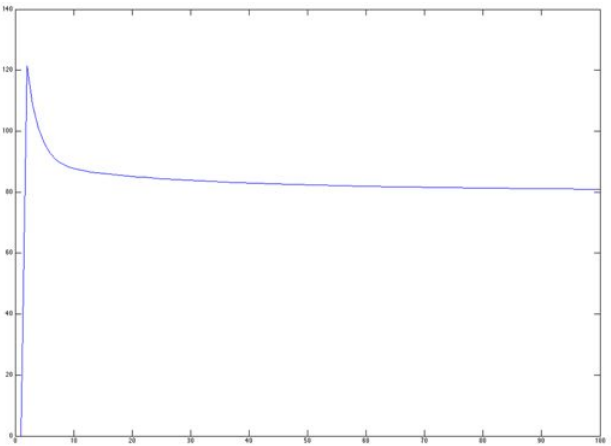


Fig. 8. Behavior of the cost functional for $\lambda = 7,\ \mu = 9$, 100 iterations and initialization (a) $(u_0 = 0, v_0 = u_d)$

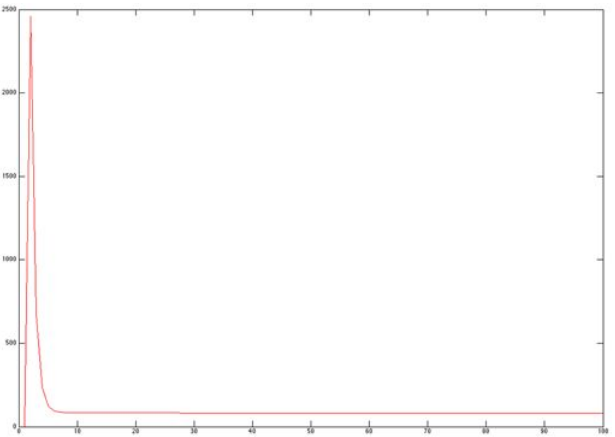


Fig. 9. Behavior of the cost functional for $\lambda = 7,\ \mu = 9$, 100 iterations and initialization (b) $(u_0 = 0, v_0 = 0)$

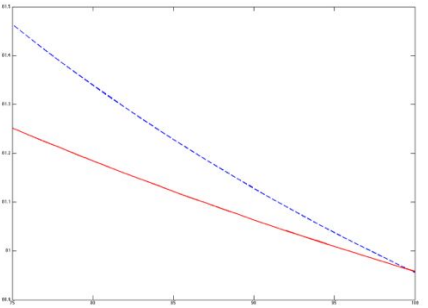


Fig. 10. Behavior of the cost functional for $\lambda = 7,\ \mu = 9$, 100 iterations- Dotted (blue) line is initialization (a) and solid (red) line is initialization (b) . Zoom around 75-100 iterations

# it.	$TV(u_a)$	$TV(u_b)$	$TV(\varphi)$	$TV^2(v_a)$	$TV^2(v_b)$	$TV^2(\varphi)$	Error (a)	Error (b)
$\lambda = 1,\ \mu = 10$								
50	10.73	12.12	3.98	2.74	1.27	2.19	14.58	6.51
100	11.81	12.51	3.13	1.92	1	1.50	8.30	3.33
200	12.39	12.74	2.47	1.39	0.81	1.09	4.39	1.72
400	12.70	12.87	1.96	1.04	0.69	0.82	2.37	0. 90
600	12.80	12.92	1.70	0.9	0.65	0.70	1.53	0.58
800	12.85	12.94	1.53	0.83	0.63	0.63	1.09	0.42
$\lambda = 10,\ \mu = 15$								
50	2.59	4.83	2.74	4.42	3.09	3.14	9.04	9.33
100	3.19	4.44	1.85	3.98	3.27	2.14	5.29	6.17
200	3.48	4.07	1.18	3.77	3.45	1.39	2.49	3.77
400	3.59	3.83	0.73	3.69	3.56	0.87	1.39	2.03
600	3.61	3.76	0.55	3.67	3.60	0.67	1.08	1.40
800	3.62	3.72	0.45	3.67	3.62	0.54	0.87	1.07
$\lambda = 10,\ \mu = 2$								
50	8.93 e-03	1.61	1.61	11.21	9.88	1.80	1.1 e-02	18.18
100	8.95 e-03	0.89	0.89	11.21	10.56	9.6 e-01	4.7 e-03	12.92
200	8.95 e-03	0.41	0.41	11.21	11.01	3.3 e-01	1.9 e-03	8.92
400	8.95 e-03	0.18	0.18	11.21	11.18	6.0 e-02	9.3 e-04	4.48
600	8.95 e-03	0.11	0.11	11.21	11.20	1.5 e-02	6.2 e-04	2.54
800	8.95 e-03	0.08	0.08	11.21	11.21	6.4 e-03	4.2 e-04	1.77

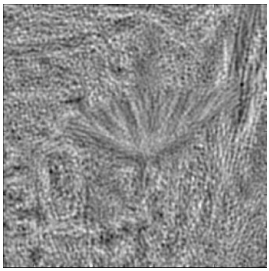
Table 6. Sensitivity with respect to number of iterations. Here $\varphi = u_b - u_a = v_b - v_a$ and the error is given by the stopping criterion (13) of Algorithm. On can refer to Table 4 as well.

(a) BV part - $\lambda = 1,\ \mu = 10$

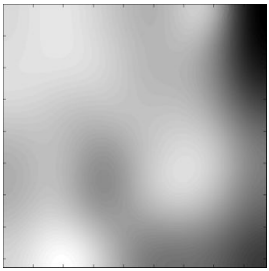
(b) φ - $\lambda = 1,\ \mu = 10$

(c) BV part - $\lambda = 10,\ \mu = 15$

(d) φ - $\lambda = 10,\ \mu = 15$

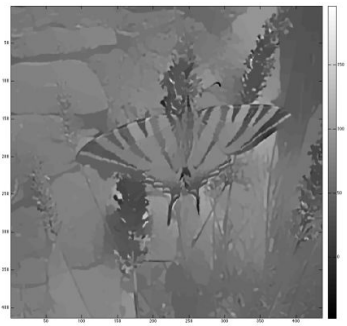


(e) BV part -$\lambda = 10,\ \mu = 2$. In this case $u\ \simeq 0$

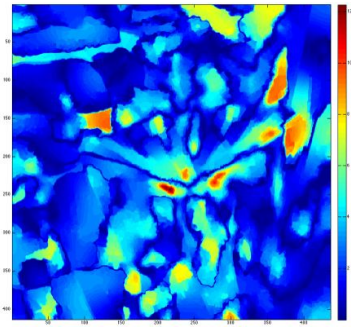


(f) φ -$\lambda = 10,\ \mu = 2$

Fig. 11. BV component u_a and φ corresponding to Table 6 - 800 iterations. Functionn φ turns to be constant. The choice $\lambda = 10,\ \mu = 15$ gives a satisfactory cartoon part.

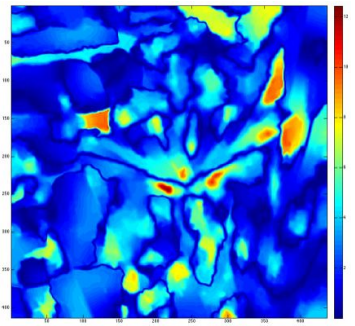


(a) BV part u - 800 iterations

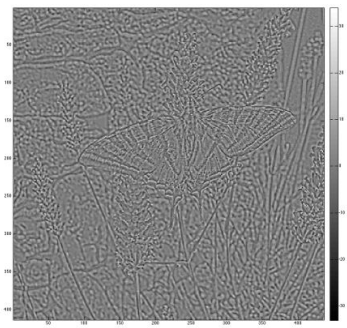


(b) Absolute difference of BV parts between it 400 and it 800

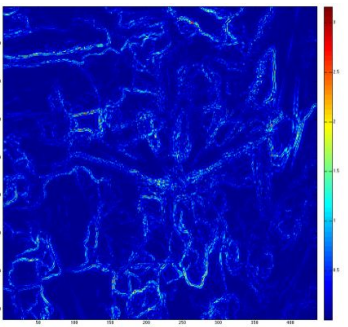
(c) BH part v - 800 iterations



(d) Absolute difference of BH parts between it 400 and it 800



(e) L^2- part w - 800 iterations



(f) Absolute difference of L^2- parts between it 400 and it 800

Fig. 12. Test 2D - Initialization (b) for $\lambda = 10,\ \mu = 15$- Difference between the computed pairs at iteration 400 and iteration 800.

3.5 Sensitivity with respect to sampling and quantification

We first investigate the sensitivity with respect to sampling. Table 7, figures 13 and 14 show that the model is robust with respect to sampling. Here, we have discretized the analogical signal of example 1D with 10^3, 10^4 and 10^5 points respectively.

λ	μ	$\mathcal{F}_{\lambda,\mu}(u_b, v_b)$	$\|w_b\|_{L^2}$	$TV(u_b)$	$TV^2(v_b)$
		10^3 points	10^3 points	10^3 points	10^3 points
		10^4 points	10^4 points	10^4 points	10^4 points
		10^5 points	10^5 points	10^5 points	10^5 points
1e -03	1 e-02	6.47 e-06	1.43 e-03	5.01 e-03	4.28 e-05
		5.78 e-06	1.47e-03	4.37 e-03	3.19 e-05
		5.73 e-06	1.5 e-03	4.34 e-03	3.09 e-05
1e-03	1	4.86 e-05	1.43 e-03	5.01 e-03	4.25 e-05
		3.73 e-05	1.47 e-03	4.37 e-03	3.18 e-05
		3.63 e-05	1.47 e-03	4.34 e-03	3.09 e-05
1e -02	1e-01	3.27 e-05	5.07 e-03	8.80 e-04	1.10 e-04
		2.62 e-05	5.16 e-03	1.81 e-04	1.10 e-04
		2.52 e-05	5.24 e-03	7.26 e-05	1.08 e-04
1e -02	1	1.32 e-04	5.07 e-03	8.80 e-04	1.10 e-04
		1.26 e-04	5.16 e-03	1.81 e-04	1.10 e-04
		1.22 e-04	5.24 e-03	7.26 e-05	1.08 e-04
1e-01	1e-01	1.01 e-04	6.90 e-03	5.43 e-04	2.32 e-04
		3.71 e-05	5.53 e-03	8.74 e-05	1.30 e-04
		2.73 e-05	5.38 e-03	1.20 e-05	1.16 e-04

Table 7. Test 1D (with noise) - sensitivity with respect to sampling - Initialization (b) ($u_0 = v_0 = 0$) and 10 000 iterations

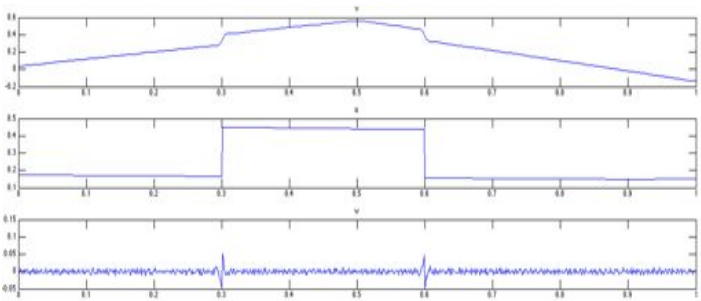


Fig. 13. Test 1D (with noise) - Pair given by initialization (b) for $\lambda = 0.1,\ \mu = 1$, 10 000 iterations and 10^3 points sampling.

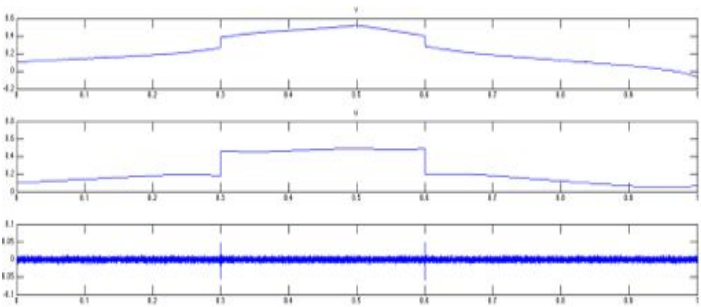


Fig. 14. Test 1D (with noise) - Pair given by initialization (b) for $\lambda = 0.1,\ \mu = 1$, 10 000 iterations and 10^4 points sampling.

We now investigate the sensitivity of the model with respect to quantification. Let u_d a data (with values in [0, 255] for example). Let (λ, μ) be chosen parameters and $(u_{\lambda,\mu}, v_{\lambda,\mu})$ the corresponding computed pair (with the appropriate initialization). Let $\alpha > 0$ and consider the new data αu_d. This is the case, for example, if we get 16 bits images and convert them to 8 bits : in this case $\alpha = (2^8 - 1)/(2^{16} - 1)$. We may want to normalize the data as well: in this case $\alpha = 1/\max(u_d)$. The question is to know what new parameters $(\tilde{\lambda}, \tilde{\mu})$ must be chosen to get $u_{\tilde{\lambda},\tilde{\mu}} = \alpha u_{\lambda,\mu}$ and $v_{\tilde{\lambda},\tilde{\mu}} = \alpha v_{\lambda,\mu}$. For any $(u_{\lambda,\mu}, v_{\lambda,\mu})$ solution to $(\mathcal{P}_{\lambda,\mu})$, we get

$$
\begin{aligned}
F_{\lambda,\mu}(u_{\lambda,\mu}, v_{\lambda,\mu}) &= \frac{1}{2}\|u_d - u_{\lambda,\mu} - v_{\lambda,\mu}\|^2 + \lambda TV(u_{\lambda,\mu}) + \mu TV^2(v_{\lambda,\mu}) \\
&= \frac{1}{2\alpha^2}\|\alpha u_d - \alpha u_{\lambda,\mu} - \alpha v_{\lambda,\mu}\|^2 + \frac{\lambda}{\alpha} TV(\alpha u_{\lambda,\mu}) + \frac{\mu}{\alpha} TV^2(\alpha v_{\lambda,\mu}) \\
&= \frac{1}{\alpha^2}\left(\frac{1}{2}\|\alpha u_d - u_{\tilde\lambda,\tilde\mu} - v_{\tilde\lambda,\tilde\mu}\|^2 + \alpha\lambda TV(u_{\tilde\lambda,\tilde\mu}) + \alpha\mu TV^2(v_{\tilde\lambda,\tilde\mu})\right) \\
&= \frac{1}{\alpha^2} F_{\tilde\lambda,\tilde\mu}(u_{\tilde\lambda,\tilde\mu}, v_{\tilde\lambda,\tilde\mu}) \text{ with } \tilde\lambda = \alpha\lambda \text{ and } \tilde\mu = \alpha\mu.
\end{aligned}
$$

α	1/255	100
$\lambda = 7,\ \mu = 9$ Initialization (a)		
$\|u_{\alpha\lambda,\alpha\mu} - \alpha u_{\lambda,\mu}\|_\infty/\alpha$	3.0291e-01	3.0291e-01
$\|v_{\alpha\lambda,\alpha\mu} - \alpha v_{\lambda,\mu}\|_\infty/\alpha$	3.1291e-01	3.1291e-01
$\lambda = 7,\ \mu = 9$ Initialization (b)		
$\|u_{\alpha\lambda,\alpha\mu} - \alpha u_{\lambda,\mu}\|_\infty/\alpha$	1.8006e-01	1.8006e-01
$\|v_{\alpha\lambda,\alpha\mu} - \alpha v_{\lambda,\mu}\|_\infty/\alpha$	1.7924e-01	1.7924e-01
$\lambda = 10,\ \mu = 2$ Initialization (a)		
$\|u_{\alpha\lambda,\alpha\mu} - \alpha u_{\lambda,\mu}\|_\infty/\alpha$	8.0280e-15	8.4421e-15
$\|v_{\alpha\lambda,\alpha\mu} - \alpha v_{\lambda,\mu}\|_\infty/\alpha$	1.1324e-13	1.4552e-13
$\lambda = 10,\ \mu = 2$ Initialization (b)		
$\|u_{\alpha\lambda,\alpha\mu} - \alpha u_{\lambda,\mu}\|_\infty/\alpha$	3.9216e-03	4.5475e-14
$\|v_{\alpha\lambda,\alpha\mu} - \alpha v_{\lambda,\mu}\|_\infty/\alpha$	1.1324e-13	1.0914e-13

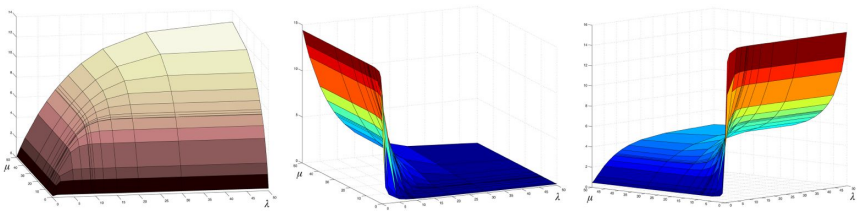
Table 8. Sensitivity with respect to quantification- Initialization (b) - itmax = 400. The model is robust with respect to quantifications as expected.

3.6 Sensitivity with respect to parameters

As mentioned before the computed stationary pair depends on the initialization guess via the convergence speed. We consider three cases and we illustrate them on test 2D (*Butterfly*).

– If $\mu << \lambda$, then initialization (a) : $u_0 = 0$ and $v_0 = u_d$ is the best choice to make the algorithm converge quickly. So we use this initialization to get the *solution* (u^*, v^*). In this case, the BV part is close to 0. However, we

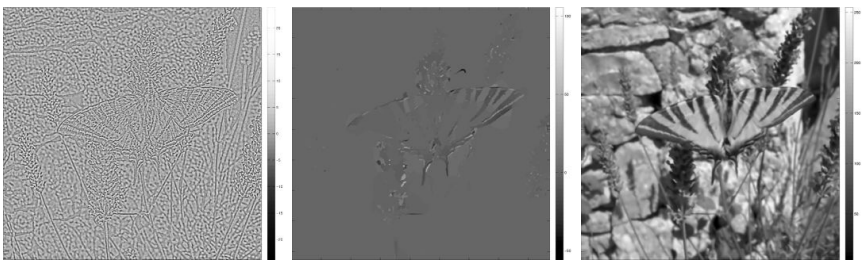
note that if we fix μ then $TV(u^*(\lambda,\mu))$ decreases to 0 and $TV^2(v^*(\lambda,\mu))$ increases to become constant (see Figure 15) when $\lambda \to +\infty$. This means that if λ is large then u^* is constant. As we know that u^* has a null mean value, then $u^* = 0$.



(a) $\|w^*\|_{L^2}$ (b) $TV(u^*)$ (c) $TV^2(v^*)$

Fig. 15. Generic L^2- norm, TV and TV^2 behavior (μ fixed) 400 iterations - Example 2D (Butterfly).

On can see an example on Figure 11 for $\lambda = 10,\ \mu = 2$ and Figure 16.



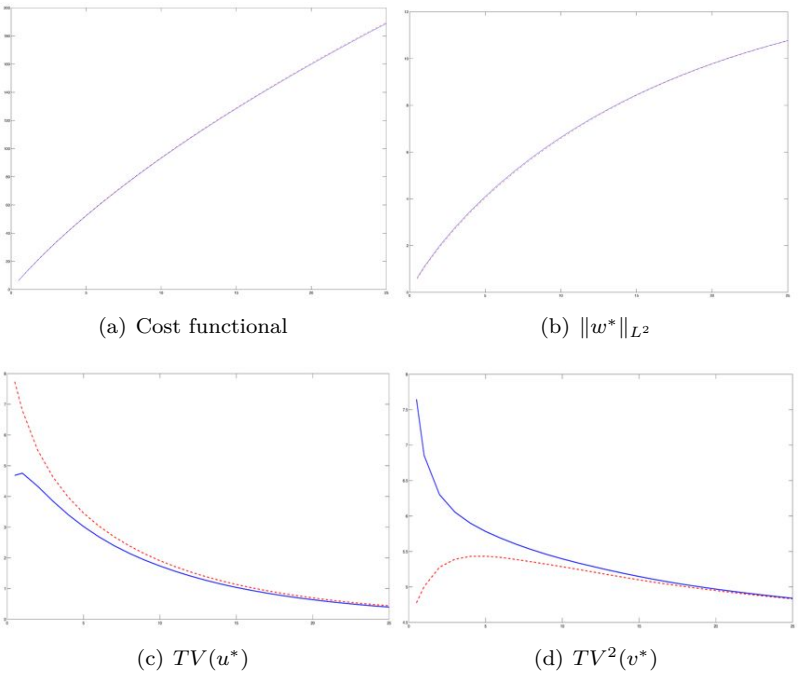
(a) L^2 part (b) BV part (c) BH part

Fig. 16. $\lambda = 7,\ \mu = 5$ - initialization $u_0 = 0$ and $v_0 = u_d$, 400 iterations

– If $\mu \simeq \lambda$, both initializations seem equivalent. For the *Butterfly* test, init (a) remains slightly faster (in this case the minimum value of cost functional is achieved first) while it is the converse for the *Wall* test and small values of λ. Figures 17 and 18 show the behavior of the cost functional, L^2- norm, TV and TV^2 for both initializations and $\lambda = \mu \in [0.5, 1, 2, 3 \cdots 25]$. We report the behavior of cost functional, L^2- norm, TV and TV^2 in Table 9

$\lambda = \mu$	$\mathcal{F}_{\lambda,\lambda}$	$\|w_a\|_2$	$TV(u_a)$	$TV2(v_a)$	Error
0.5	6.3577	1.459 e-03	4.685	7.646	4.13 e-01
1	12.2448	2.655 e-03	4.756	6.853	5.19 e-01
5	52.4043	9.654 e-03	3.020	5.782	5.83 e-01
10	93.2718	1.562 e-02	1.733	5.395	5.59 e-01
13	114.9198	1.835 e-02	1.268	5.237	5.11 e-01
17	141.3794	2.128 e-02	8.506 e-01	5.066	4.78 e-01
21	165.9126	2.357 e-02	5.768 e-01	4.940	4.36 e-01
25	188.8569	2.537 e-02	3.941 e-01	4.840	3.86 e-01

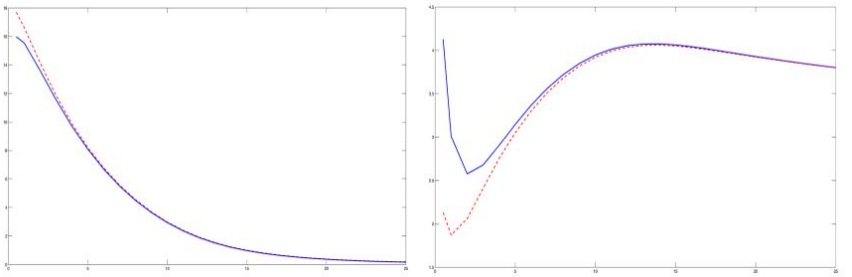
Table 9. Cost functional, L^2- norm, TV and TV^2 for $\lambda = \mu = 0.5, 1, 2, 3 \cdots 25$- init (a) - 800 iterations - *Butterfly*



(a) Cost functional (b) $\|w^*\|_{L^2}$

(c) $TV(u^*)$ (d) $TV^2(v^*)$

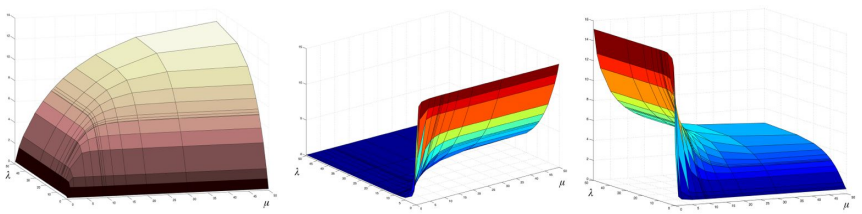
Fig. 17. Cost functional, L^2- norm, TV and TV^2 for $\lambda = \mu = 0.5, 1, 2, 3 \cdots 25$ - Dotted (blue) line is initialization (a) and solid (red) line is initialization (b) - 800 iterations - Error is given by (13). In this case the cartoon part tends to a constant function as $\lambda = \mu$ increases (the total variation tends to 0) while the second order total variation of the smooth part converges to a limit (not equal to 0, a priori) (*Butterfly* test)



(a) $TV(u^*)$ (b) $TV^2(v^*)$

Fig. 18. TV and TV^2 for $\lambda = \mu = 0.5, 1, 2, 3 \cdots 25$ - Dotted (blue) line is initialization (a) and solid (red) line is initialization (b) - 800 iterations - *Wall* test - The behavior of TV and TV^2 is similar to the previous example.

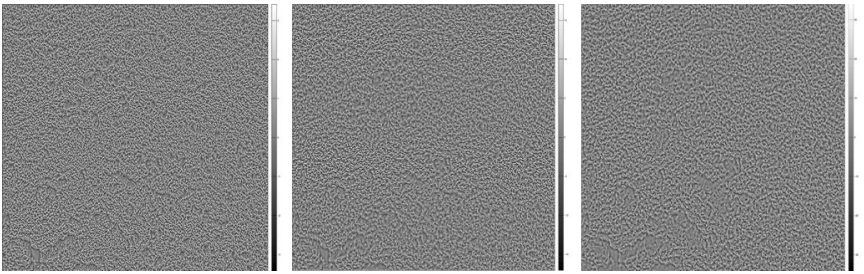
– If $\lambda << \mu$, then we choose initialization (b) : $u_0 = 0$ and $v_0 = 0$ to get the *solution*. The behavior is similar to the case $\mu < \lambda$: if we fix λ, then $TV(u^*(\lambda, \mu))$ increases to a constant value and $TV^2(v^*(\lambda, \mu))$ converges to 0 as $\mu \to +\infty$ (see figure 19). This means that if μ is large enough then solution is always the same : v^* is an affine function.



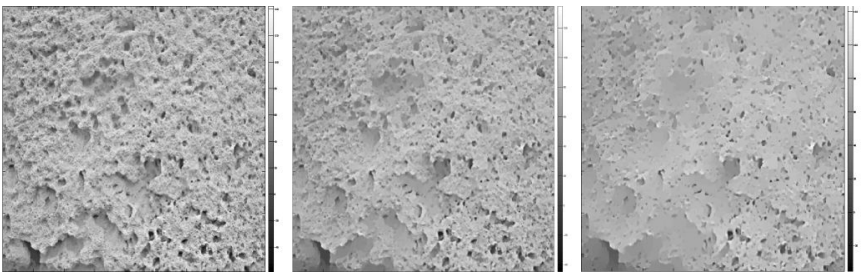
(a) $\|w^*\|_{L^2}$ (b) $TV(u^*)$ (c) $TV^2(v^*)$

Fig. 19. Generic L^2- norm, TV and TV^2 behavior - 400 iterations - Example 2D (Butterfly). Surfaces giving L^2- norm, TV and TV^2 respectively with respect to λ and μ.

Examples of solutions are given in Figures 4, 11 and 12. We give another example below on a textured image:



(a) L^2- part $\lambda = 1,\ \mu = 5$ (b) L^2- part $\lambda = 5,\ \mu = 10$ (c) L^2- part $\lambda = 10,\ \mu = 20$



(d) BV part $\lambda = 1,\ \mu = 5$ (e) BV part $\lambda = 5,\ \mu = 10$ (f) BV part $\lambda = 10,\ \mu = 20$

Fig. 20. BV and L^2 components with $\lambda < \mu$ - 800 iterations - Wall example - The value of λ ($\mu > \lambda$) gives a texture scaling information.

λ	μ	$\mathcal{F}_{\lambda,\mu}(u^*, v^*)$	$\|w^*\|_{L^2}$	$TV(u^*)$	$TV^2(v^*)$
1	5	21.6334	4.270 e-03	18.453	3.868 e-01
5	10	87.2937	1.759 e-02	10.402	1.413
10	20	152.5461	2.854 e-02	4.922	2.382

Table 10. Wall- example, initialization $u_0 = 0,\ v_0 = 0$ - 800 iterations - The larger λ is, the more the texture information is involved in the L^2 part, w^*.

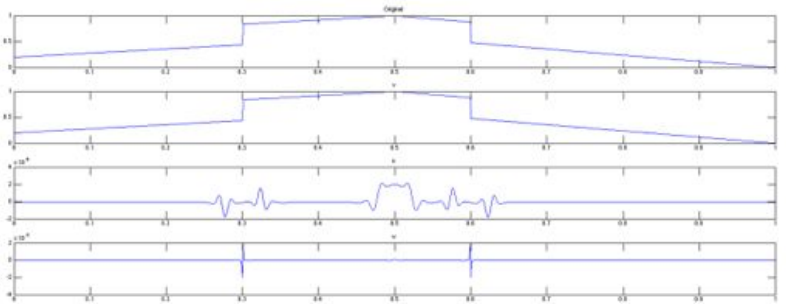
4 Conclusion

The model is well adapted to texture extraction. In the case, where the data is noiseless and/or is not too much textured, the decomposition given par $\lambda \precsim \mu$

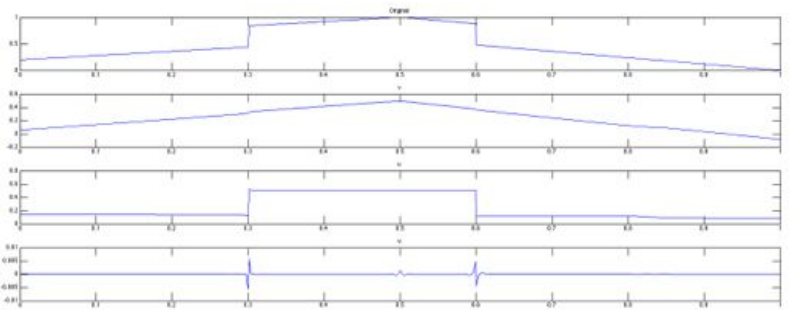
and initialization $u_0 = v_0 = 0$, gives a cartoon part which is piecewise constant as expected. This means that $u = \sum_i u_i \mathbf{1}_{\Gamma_i}$ where $\bigcup_i \Gamma_i$ is the contour set. In this case, the remainder L^2 term is the texture and/or noise. The decomposition is robust with respect to quantification, sampling and is always the same for any $\mu >> \lambda$, once λ has been chosen.

In the case where the image is highly textured the model provides a two-scale decomposition. The TV part represents the macro-texture and the L^2 part the micro-texture and/or noise. The scaling is tuned via the ratio $\rho = \dfrac{\lambda}{\mu}$. The notion of *highly textured* may be quantified par the G-norm. In our 2D examples, the *butterfly* G norm was $\simeq 7.71$ and the *wall* one was $\simeq 4.92$.

Figure 21 shows the behavior of the different components with respect to λ and μ. We have chosen the 1D noiseless case, to see the multi-scale effect on components u and w when $\mu < \lambda$.



(a) $\lambda = 5\ 10^{-3},\ \mu = 10^{-3}$- (initialization (a))



(b) $\lambda = 5\ 10^{-3},\ \mu = 10^{-2}$- (initialization (b))

Fig. 21. Test 1D without noise (1000 points)

Moreover, the initialization process has no influence on the solution (up to a constant function) but rather on the algorithm speed. The choice has to be made with respect to the parameters: roughly speaking, if $\lambda < \mu$ we choose $u_0 = 0,\ v_0 = 0$ and if $\lambda \geq \mu$ we choose $u_0 = 0,\ v_0 = u_d$. Finally, we have observed (numerically) that the L^2-component w is unique.

Next issue is to speed up the algorithm (using more performant algorithms) and set an automatic parameter tuning with respect to data properties (G norm, Signal to Noise Ratio , and so on.) From the theoretical point of view, we infer that problem $(\mathcal{P}_{\lambda,\mu})$ has a unique solution (up to constant functions) but the question is still open.

References

[1] L. Ambrosio, N. Fusco, and D. Pallara. *Functions of bounded variation and free discontinuity problems*. Oxford Mathematical Monographs. The Clarendon Press Oxford University Press, New York, 2000.

[2] H. Attouch, G. Buttazzo, and G. Michaille. *Variational analysis in Sobolev and BV spaces*, volume 6 of *MPS/SIAM Series on Optimization*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2006. Applications to PDEs and optimization.

[3] G. Aubert and J.-F. Aujol. Modeling very oscillating signals. Application to image processing. *Appl. Math. Optim.*, 51(2):163–182, 2005.

[4] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing, Partial Differential Equations and the Calculus of Variations*, volume 147 of *Applied Mathematical Sciences*. Springer Verlag, 2006.

[5] J.-F. Aujol, G. Aubert, L. Blanc-Féraud, and A. Chambolle. Image decomposition into a bounded variation component and an oscillating component. *J. Math. Imaging Vision*, 22(1):71–88, 2005.

[6] M. Benning, C. Brune, M. Burger, and J. Müller. Higher-order tv methods enhancement via bregman iteration. *Journal of Scientific Computing*, 54(2-3):269–310, 2012.

[7] M. Bergounioux. On Poincaré-Wirtinger inequalities in BV - spaces. *Control & Cybernetics*, 4(40):921–929, 2011.

[8] M. Bergounioux. Mathematical analysis of a inf-convolution model for image processing. *Journal of Optimization Theory and Applications*, pages 1–21, 2015.

[9] M. Bergounioux and L. Piffet. A second-order model for image denoising. *Set-Valued Var. Anal.*, 18(3-4):277–306, 2010.

[10] M. Bergounioux and L. Piffet. A full second order variational model for multiscale texture analysis. *Computational Optimization and Applications*, 54:215–237, 2013.

[11] K. Bredies and M Holler. Regularization of linear inverse problems with total generalized variation. *Journal of Inverse and Ill-posed Problems*, 68, 2014.

[12] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM J. Imaging Sci.*, 3(3):492–526, 2010.

[13] K. Bredies, K. Kunisch, and T. Valkonen. Properties of l^1-tgv^2: the one-dimensional case. *J. Math. Anal. Appl.*, 398(1):438–454, 2013.

[14] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[15] P. L. Combettes and B. C. Vũ. Variable metric forward-backward splitting with applications to monotone inclusions in duality. *Optimization*, 63(9):1289–1318, 2014.

[16] L. Condat. A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms. *J. Optim. Theory Appl.*, 158(2):460–479, 2013.

[17] F. Demengel. Fonctions à hessien borné. *Annales de l'institut Fourier*, 34(2):155–190, 1984.

[18] L.C. Evans and R. Gariepy. *Measure theory and fine properties of functions*. CRC Press, 1992.

[19] Y. Meyer. *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations*, volume 22 of *University Lecture Series*. AMS, 2001.

[20] S. Osher, A. Sole, and Vese L. Image decomposition and restoration using total variation minimization and the H^1 norm. *SIAM Journal on Multiscale Modeling and Simulation*, 1-3(349-370), 2003.

[21] S. Osher and L. Vese. Modeling textures with total variation minimization and oscillating patterns in image processing. *Journal of Scientific Computing*, 19(1-3):553–572, 2003.

[22] S. Osher and L. Vese. Image denoising and decomposition with total variation minimization and oscillatory functions. special issue on mathematics and image analysis. *J. Math. Imaging Vision,*, 20(1-2):7–18, 2004.

[23] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[24] P. Weiss, G. Aubert, and L. Blanc-Féraud. Efficient schemes for total variation minimization under constraints in image processing. *SIAM Journal on Scientific Computing*, 31(3):2047–2080, 2009.

[25] W. Yin, D. Goldfarb, and S. Osher. A comparison of three total variation based texture extraction models. *J. Vis. Commun. Image*, 18:240–252, 2007.