UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MATEMÁTICA

ALGUNOS RESULTADOS TEÓRICOS Y NUMÉRICOS EN PROCESAMIENTO DE IMÁGENES DIGITALES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO

MATÍAS MAXIMILIANO GODOY CAMPBELL

PROFESOR GUÍA:
JAIME ORTEGA PALMA

MIEMBROS DE LA COMISIÓN:
TAKESHI ASAHI KODAMA
MANUEL DEL PINO MANRESA

Este trabajo ha sido financiado por FONDECYT
a través del proyecto 1111012

SANTIAGO DE CHILE
ABRIL 2013

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL MATEMÁTICO
POR: MATIAS GODOY CAMPBELL
FECHA: 11/04/2013
PROF. GUÍA: JAIME ORTEGA

ALGUNOS RESULTADOS TEÓRICOS Y NUMÉRICOS EN PROCESAMIENTO DE IMÁGENES DIGITALES

El objetivo principal del presente trabajo es el estudio, tanto teórico como numérico, de métodos de procesamiento de imágenes orientados al área de las señales e imágenes con texturas.

Se comienza con una revisión de conceptos básicos tanto en procesamiento de imágenes, como en herramientas matemáticas de interés, se exploran trabajos previos que motivan buena parte de esta memoria. En particular los trabajos de filtros no locales, formulados por Buades en 2005 y los funcionales no locales del tipo Mumford-Shah, formulados por Jung et al. en 2011, además de varios trabajos clásicos del área del cálculo de variaciones ligados al procesamiento de imágenes. Más adelante, se explora en detalle el problema de segmentación, estableciendo una definición concreta y ejemplos de aplicación, presentando luego el funcional de Mumford-Shah. Se analiza la limitación de este funcional desde el punto de vista numérico para realizar segmentación y por esto se introduce el funcional de Ambrosio-Tortorelli, donde destacan sus resultados más importantes, en particular la Γ-convergencia al funcional de Mumford-Shah relajado.

Como desarrollo central, se trabajó el problema de segmentación no local, considerando el funcional planteando por Jung et al. en 2011. Se prueba que en el caso unidimensional el funcional no es adecuado en el sentido del Γ-límite obtenido, el cual no penaliza el conjunto de discontinuidades de la señal u. A partir de esto, se propone un funcional modificado que, bajo ciertas suposiciones sobre la función de peso no local $w(x,y)$, permite obtener un término que es equivalente como semi-norma a la semi-norma de Slobodeckij, lo que implica que el funcional quede definido en el espacio $H^s \times H^1$, donde H^s es el espacio de Sobolev fraccionario con $s \in (1/2, 1)$. Se prueba rigurosamente la Γ-convergencia a un funcional que se puede interpretar como el funcional de Mumford-Shah relajado con gradiente no local. Este resultado es relevante porque en este caso el Γ-límite sí penaliza el conjunto de discontinuidades de la señal u, que es el comportamiento deseado para estos funcionales. A continuación, se exponen las llamadas funciones de Gabor generalizadas, para ser utilizadas en la aproximación de una señal, utilizándose como ejemplo las *splines exponenciales* (complejas), que corresponden a funciones trigonométricas con soporte compacto, permitiendo aproximar una señal en diferentes niveles de resolución.

Finalmente, se presenta la implementación numérica de los modelos considerados, partiendo por filtros no locales, modelos de segmentación local y no local, y concluyendo con la aproximación por splines exponenciales. Se exponen simulaciones numéricas que permiten comparar diversos métodos además de explorar las ventajas y limitaciones de cada método en particular, concluyendo que existe evidencia de que estos métodos efectivamente permiten mejorar el análisis de señales e imágenes que contienen texturas.

A mis padres con cariño.

AGRADECIMIENTOS

Agradezco en primer lugar a mis padres, Soledad y Víctor, es a ambos a quienes debo las grandes oportunidades que he tenido en la vida, espero algún día llegar tan lejos como ustedes. También agradezco a mis hermanos, en especial por soportarme durante todos estos años, a mi sobrina Antonia, por su siempre necesaria alegría infantil, y a mi familia como conjunto, por todas las buenas y malas que hemos pasado.

Agradezco a Javiera, mi gran compañera, por todo su amor durante estos 4 años juntos, tu amor, paciencia y comprensión han sido fundamentales, sin ellos todo hubiese sido más dificil. Gracias por ser parte de mi sueño. Agradezco también a tu familia por acogerme tal como si fuese uno de ustedes.

Agradezco también a mi profesor guía, Jaime Ortega, por su fundamental ayuda en la realización de este trabajo y con quien más que una relación de trabajo he podido establecer una gran amistad. Estoy seguro que este no será nuestro último trabajo en conjunto. Del mismo modo agradezco a mi profesor co-guia, Takeshi Asahi, ha sido un honor y un gusto aprender buena parte de lo que se en imágenes con tu ayuda. Agradezco también al profesor Manuel del Pino por aceptar amablemente el ser parte de la comisión de evaluación de este trabajo. Además agradezco a Rodrigo Lecaros por sus continuos aportes y observaciones los cuales fueron tremendamente útiles para el problema en estudio.

A mis grandes amigos durante estos años: Orly, Emilio y Nico. Por la infinidad de buenos momentos juntos y por el permanente apoyo que me han dado en los malos. Gracias por su amistad a los tres. También agradezco a mis buenos amigos durante estos años en la universidad, con quienes he tenido grandes momentos en innumerables horas (en especial de distracción) juntos: Conti, Obando, Gomena, Fili, Seba D., Felipe S., Víctor C., Mauro, Tonho, Andrés Z., Pancho, Benja P., Vale, Tish, Víctor V., Nacho, Niko, Alf y Ricardo. Del mismo modo, gracias a mis amigos de la vida, quienes han estado y siguen estando a pesar de los años: Rixie, Fafo, Cabezón, Jose Luis, Panzo, Paloma y Pancha.

Un especial agradecimiento a los profesores que me han permitido ejercer la docencia por darme la oportunidad de hacer algo que me apasiona, gracias a Maya Stein, Leonardo Sánchez, Pablo Navarrete, José Zamora, Jaime Ortega y Carlos Conca.

Agradezco también a todos los funcionarios del DIM, quienes me han 'salvado la vida' en variadas ocasiones y han hecho de mi paso por el departamento algo mucho más sencillo, en especial a Gladys Cavallone, Eterin Jaña, Silvia Mariano, Oscar Mori y Luis Mella.

Al equipo del laboratorio MOTIV en el Centro de Modelamiento Matemático, muchas gracias. Fue un gusto trabajar con ustedes durante cerca de un año y ser parte de un equipo como el suyo.

Finalmente agradezco a FONDECYT por financiar este trabajo a través del proyecto 1111012.

Contents

Chapter 1

Introduction

The present work consists, mainly, in the development of theoretical and numerical results in the study of a new functional proposed in the field of image processing. Basically the considered functional is an approximation, in some sense that we will specify later, of the well-known Mumford-Shah functional used widely in image processing, the objective of this functional is to perform a task called **segmentation**, which basically consists in the recognition of edges of the different objects in a (digital) image. In this case, we want to prove some essential results which leads to correctness of the numerical tasks which are performed in the practice. We also perform numerical experiments in order to verify the known numerical results and also to study how this methods perform in the practice.

This chapter is intended to introduce the basic terminology that will be used in this work, in a mathematical and descriptive way.

1.1 Digital Images

A natural question when one work in mathematics with images is: *What is a digital image?*

A digital image, or discrete image also, is obtained (from the continuous world) by sampling and quantization. Certainly this depends on the acquisition devices, for example on CCDs (Charge-Coupled Device) for digital cameras. The basic idea to acquire is to superimpose a regular grid on an analogue image, and assign a number to each square on the grid, that represents some feature in order to characterize the actual image, an example for this number could be the average brightness in each square. The squares are called pixels, and they are the smallest elements in an image, its assigned value is usually the grey-level or brightness. Notice that this quantities in the practice are always bounded and moreover, usually takes integer values between 0 and 255.

To describe a pixel, depending on the way we want to represent an image, we may need different channels or bands, in fact, if we want an image on grey-scale we just need one channel, but if we want a color image we will need three channels: red, green and blue.

The final characteristic of an image is its size or resolution. This is the number of rows and columns in the image, this is simply the size of the regular grid we superimpose on the analogue image. Notice that this characteristic depends on the acquisition device and is not relative to the real image itself.

So, one can describe a digital image as the discretization of an analog image. In a mathematical setting could be described as follows:

Definition 1.1.1 (Image)**.** A digital image can be defined as a function $u \in L^\infty(\Omega)$ (in the case of grey-scale digital image) or $u \in (L^\infty(\Omega))^3$ (in the case of color images), where Ω is a bounded set, specifically, in our context: $\Omega = [0, N-1] \times [0, M-1]$ where N, M are the numbers of rows and columns in the image, u takes values only on the points $(n, m) \in \Omega$ where n, m are integers between 0 and N and 0 and M respectively.

Notice that this definition could be completely defined by a matrix, but we prefer to describe the image in a continuous setting, because in the theoretical problems we will work with continuous models.

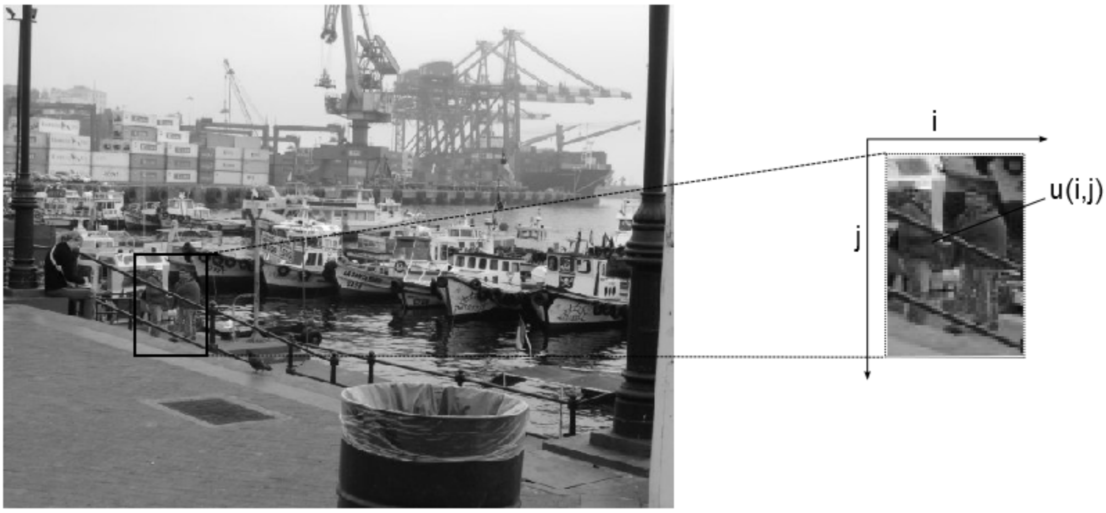


Figure 1.1: Example of a digital image: Notice that a digital image is nothing but a two-dimensional array of pixels where each pixel/point has an assigned brightness value.

It is natural to notice that digital images are just an approximation of reality, we have a size and representation measures that depends on the acquisition devices, so, for example, if the size of the digital image is low, the approximation of the real image in the moment that the picture is taken, will be low in comparison from a digital image with a bigger size, which will be able to retain more specific details, moreover, we just have a finite number of possible values for each pixel, and finally we have to consider that the digitalization process carries random noise, so, the digitalization process is limited by its own restrictions. Reality have infinite choices, digitalization just a finite ones.

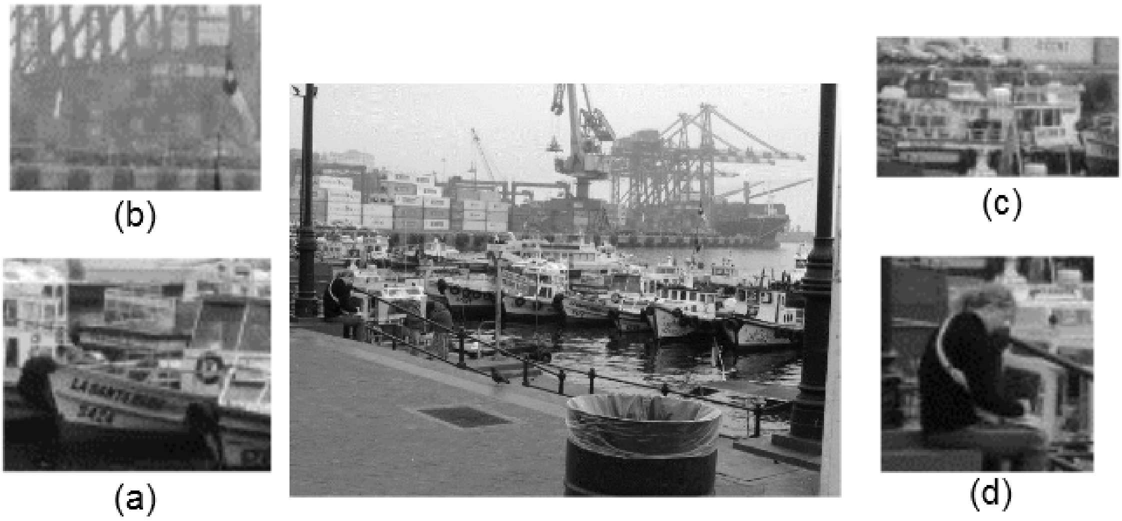


Figure 1.2: Example Digital Image Defects: **(a)** low resolution which can distort objects - **(b)** low contrast which can 'hide' objects - **(c)** small elements which can be loss - **(d)** strong transitions between objects

Due to these restrictions, several processes or tasks have been developed in order to improve the quality of a given digital image, these tasks are usually formulated in a mathematical continuous setting, rather than the discrete one represented in a digital image. The use of continuous setting is focused in to be able to apply all the mathematical results available from optimization, partial differential equations and functional analysis, areas where image processing tasks could be formulated. Notice that obviously in the practice we will work with discrete evaluations of the involved image functions.

1.2 Image Representation

Our first definition of an image is the formalization of what we seen in the practice. In order to be able to handle this images in the usual image tasks, we need to represent them in a more specific ways, in order to have an appropiated mathematical definition for the tasks involved. Here we present several representations for digital images:

- **Random fields**: In this case, an observed image u_0 is modeled as the sampling of a random field. Images are modeled by some Gibbs/Markovian random fields [30]. The statistical properties of fields are often established through a filtering technique and learning theory. This modeling is the ideal approach for describing natural images with several patterns such as trees and mountains.
- **Wavelet Representation**: An image is often acquired from the responses of a collection of microsensors, either digital or biological. It has been realized, supported with experiments, such that local responses can be well approximated by a functions called wavelets, this functions can be described as a 'brief oscillation' (like the one can see on a seismograph or heart monitor). This representation is highly useful for multiscale structures [40], and is used in widely known compression protocols such as JPEG2000. The theory that have been started by this representation is highly

active and pushed forward by the generation of several new models coming from the idea involved of wavelet representations.

- **Regularity Spaces**: In order to be able to use the rich structure and results from Sobolev Spaces, is conventional to consider an image u in the Sobolev space $H^1(\Omega)$, or more generally, $u \in W^{1,p}(\Omega)$ for some $p \in [1, \infty]$. This model works well for homogeneous regions, but it is insufficient as a global image model, since it 'smears' an important visual cue: the edges. In order to find them, there exists some models that are defined precisely for images in this representation: the 'object-edge' model of Mumford and Shah [41] and the Bounded Variation image model of Rudin, Osher and Fatemi [42].

In this work we will consider the Regularity Spaces representation of images when we deal with the variational formulation of the segmentation problem, and also we will consider a 'Wavelet-like' representation of images in order to 'decompose' it in several structures that are of our interest.

1.3 Classical Issues on Image Processing

In this section we will present the classical image processing issues, in order to improve the quality of a given digital image, we will start defining in an abstract idea the Image processing concept which follows the ideas given in [23].

An **Image Processor** could be understand as an abstract input-output system:

$$Q_0 \rightarrow \boxed{\text{Image Processor } T} \rightarrow Q$$

where the input data Q_0 could be an observed or measured single image (or a series of images, in the case of Movie purposes), and the output $Q = (q_1, q_2, \ldots)$ contains the targeted image features.

Essentially, an image processor is the link between Computer Vision, which we understand as the field that tries to reconstruct the 3-D world from observed 2-D images and Computer Graphics, which tries to design 2-D scene images to simulate the 3-D world.

As a basical example we can consider the image processor T as the human visual system, in this case the input Q_0 represents the image sequence projected onto the retina, and the output Q contains all the major features that are relevant to our daily life, passing by the most basical features as relative orders, shapes and grouping rules to the most specifical features that help to classify or identify patterns and objects.

We will present the most important image processing tasks as a particular image processors, detailing the input and output associated:

- **Denoising and Deblurring**: Image blur and noise are the most commom problem in photography, which came from the nature of the acquisition devices and the discretization of the image to become a digital one. Image deblurring is the process of recovering a sharp image from an input image corrupted by blurring and noise, where the blurring is due to convolution with a known or unknown kernel. The task associated with deblurring and denoising can be stated with:

$$Q_0 : u_0 = K * u + n$$

where u is the 'original' image, K is a blurring kernel, and n is the noise model, and

$$Q : \text{clean and sharp } u$$

- **Inpainting**: Image inpainting, also known as image interpolation, is the process of reconstructing lost or corrupted parts of an image. This is an important inverse problem with many applications, for example: removal of scratches in old photographs or filling-in missing blocks in unreliably transmitted images. The task associated with Inpainting can be stated with:

  $$Q_0 : u_0|_{\Omega\setminus D}$$

  where Ω is the original domain of u and D is the 'missing' part, and

  $$Q : \text{entire image } u|_{\Omega}$$

- **Super-Resolution**: This task corresponds to the reconstruction of a high resolution image from a filtered and down-sampled (i.e. reduced resolution) image. This task have many applications in video. The task associated with Super-Resolution can be stated with:

  $$Q_0 : u_0$$

  $$Q : \text{multiscale images } (u_{\lambda_1}, u_{\lambda_2}, \ldots)$$

- **Segmentation**: Images are the proper 2-D projections of the 3-D world which contains several objects. In order to (approximately) reconstruct the 3-D world the first step is to identify the regions in which each object belongs. The problem of segmentation is to identify this regions. This problem have a lot of applications in a wide variety of fields, such as computer vision, medical image processing and military image processing. The task associated with Segmentation can be stated with:

  $$Q_0 : u_0$$

  $$Q : \text{the boundary of objects, the objects, and their associated regions: } \Gamma, (u_k, \Omega_k)$$

  here Γ represents the set of boundaries of the objects, Ω_k represents a subset of the image domain Ω and u_k represents the values of u in Ω_k, i.e. the image of the object in the region Ω_k.

As we mentioned in the first section, in this work we will focus on Image Segmentation, studying a classical method to perform it, and a new method which promises to perform better than the classical ones in an important class of images.

1.4 Texture in Images

One of the motivations of this work is to improve the classical denoising and segmentation models in order to get better results in images with texture, which we can define informally as a repetitive structure in an image, this kind of images is strongly usual in natural environments and usual methods tend to have trouble to deal with them.

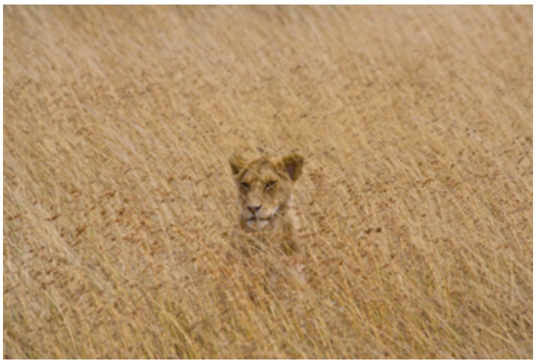


Figure 1.3: The change between different textures

Try to characterize the texture concept takes relevance when we talk about segmentation, because our first impression is that the human eye recognizes the change between different textures, which can be related by the segmentation procedure itself, see Figure 1.3.

Define texture is a difficult task, because we don't have a precise and unique mathematical definition or a clear concept. The basic idea is that the texture can be seen as a repetition of basic texture elements called texels or textons made of pixels whose placement obey some rule.

Let us give some recompilations ideas about textures:

1. The textured region can contain texture elements of various sizes, each of which can itself be textured.

2. The order consists in the nonrandom arrangement of elementary parts.

3. The parts are roughly uniform entities having approximately the same dimension everywhere within the textured region.

4. A region in an image has a constant texture if a set of local statistics or other properties of the picture function are constant, slowly varying, or approximately periodic.

Notice that this characterizations of texture have some similarities with the idea of noise in an image, and naturally some differences, this become a problem for the common methods of denoising and segmentation for images with texture, specially with ones from natural environments. The classical methods tend to fail in this tasks because the associated methods usually identify texture as noise, which leads to a loss of information of the texture on the outputs of the image processors.

1.5 Objectives of this Work

We conclude this chapter listing the main objectives of this work:

- Review the actual works on denoising and non-local segmentation proposed models for textured images.
- Study the non-local segmentation proposed model in order to obtain theorical results about the functionals involved.
- Build implementations for denoising and non-local segmentation proposed models to perform a numerical testings for this methods in order to understand them and discover their advantages from the classical models.
- Propose and implement a new (approximated) representation of images in 1-Dimension, in order to be able to decompose them in resolution levels and identify ‘hidden’ textures.
- Finally, concluding from all the previous objectives, describe suitable methods to process 1-D and 2-D images in order to understand better the textures in them.

Chapter 2

Preliminaries: Mathematical Tools for Image Processing

In this chapter, we will review some theoretical results and definitions, that will be used in this work, and also we review some interesting mathematical models.

We start reviewing some theoretical tools that we will use in this work. Firstly we review the classical tool from calculus of variations, the called *Direct Method of Caluclus of Variations*. This is the classical method to prove the existence of solutions on problems that involve the minimization of a functional, defined in a suitable space. Secondly we present and review some properties of a variational convergence: the called Γ-convergence, we have to use this tool, because when we want to approximate some functional with a numerical *bad-behavior* by a family of a *well-behaved* ones. We cannot expect to have strong convergence of the functionals, this requirement is almost impossible in general, and also is much more than we need, the idea of Γ-convergence is to ensure convergence of minima, which is the only thing we will need in our context. Finally we also include some tools of interest like Euler-Lagrange equations and some useful lemmas.

After the mathematical introduction, we will review some interesting works : we start with an article published in 2005 by Antoni Buades [19], the importance of this article is the proposal of a new denoising filter which is designed to work with textures. The main feature of this filter is the ability to denoise correctly images with texture, a task in which the common filters fail due to their local nature. This feature is based on the nature of the filter itself, Buades propose a filter based on replace the value of a fixed pixel by averages of pixels in the whole image, this non local behavior retains textures and do not consider them as noise unlike the classical filters which consider texture as noise and therefore delete them in denoising process. This model may seem away from the purpose of this work, but this filter is the basis of all the work on non-local image tasks, due to the simplicity of the involved formulas and the ability of them to capture texture on images without consider them as noise.

Finally we review an article published in 2011 by M. Jung, X. Bresson, T. Chan and L. Vese [34], in this article the authors propose a new functional designed to have a better performance in images with texture, they propose an Ambrosio-Tortorelli-like functional, changing the local behavior of the functional replacing the regularization term in the image by a non-local regularization term, this one is based on the Buades filter due to their great performance on images with texture. This model seems to work great numerically in images with texture in some common image processing tasks, but nothing as been proved about the functional itself. This is the starting point for our work, because we already know that the functional seems to work great numerically (something that we will check in this work), but theoretical results has not been proved yet.

2.1 The Direct Method of Calculus of Variations

The typical problem of the calculus of variations is to minimize an integral of the form:

$$F(u) := \int_\Omega f(x, u(x), \nabla u(x))dx,$$

where Ω is an open subset of $\mathbb{R}^N$, usually Ω is also bounded, the minimization is among functions $u : \Omega \to \mathbb{R}$, belonging to some suitable function space, and usually satisfying a boundary condition, for example a Dirichlet boundary one: $u(y) = g(y) \quad y \in \partial\Omega$.

Thus, the problem is

$$\min_{u \in \mathcal{C}} F(u)$$

The classic way to prove that this problem have solution is the procedure called *the direct method of the calculus of variations*, the idea is very simple:

We have to take a minimizing sequence $(u_n)_n \subset \mathcal{C}$, i.e.:

$$\lim_{n\to\infty} F(u_n) = \inf_{u\in\mathcal{C}} F(u)$$

and show that some subsequence of (u_n) converges to a minimizer $u \in \mathcal{C}$.

In order to make this strategy successful, we need several conditions to be met:

1. Some compactness condition has to hold so that a minimizing sequence contains a convergent subsequence. Notice that this requires a careful selection of a suitable topology on $\mathcal{C}$ (Notice this also implies that we may need to change the suitable space $\mathcal{C}$ when we want to prove the existence of minimizers for the functional without knowing in principle the 'good' space).

2. The limit u of such subsequence should be contained in $\mathcal{C}$, i.e. a closedness condition on $\mathcal{C}$.

3. Some lower semicontinuity condition of the form

   $$F(u) \leq \liminf_{n\to\infty} F(u_n) \quad \text{if } u_n \text{ converges to } u$$

   has to hold, in order to ensure that the limit of a minimizing sequence is indeed a minimizer for F

Notice that the conditions 1 and 2 suggest that we need to consider a space $\mathcal{C}$ which is not too restrictive, but, in the other hand, to hold the condition 3 we may need to consider a topology in $\mathcal{C}$ which is more restrictive. So, we have to make a balance in the choice of the topology in $\mathcal{C}$ (and even in the choice of the space itself).

We will follow this method in order to prove the existence of solution for some functionals of our interest, moreover, we will face the difficulty of chosing a suitable space $\mathcal{C}$ due to the problems that we will face to establish the conditions 1, 2 and 3.

There are several conditions for which we can ensure the existence of solutions for the typical problem of the calculus of variations using the direct method, we don't expose them here because this results will not apply in our context but the interested reader could find them in [2] pp. 11-16, [33] Chapter 4 and [25] Chapter 3.

2.2 Γ Convergence

In this section we will review the definition and main properties of the Γ-convergence that we will use in this work.

Let (X, d) be a metric space, and let $f_j : X \to \overline{\mathbb{R}}$ a family of functions.

We need first to recall the definition of a lower semicontinuous function:

Definition 2.2.1. We say that a function $f : X \to \overline{\mathbb{R}}$ is (sequentially) lower semicontinuous (or lsc) in $x \in X$, if and only if $\forall\ (x_j) \to x$ we have:

$$f(x) \leq \liminf_j f(x_j)$$

or equivalently,

$$f(x) = \min\{\liminf_j f(x_j) :\ (x_j) \to x\},$$

we will say that f is lsc in X if it is $\forall x \in X$.

Given this definition, we can provide the definition of the Γ-convergence:

Definition 2.2.2. We say that the sequence (f_j) Γ-converges to a function $f_\infty : X \to \overline{\mathbb{R}}$ if $\forall x \in X$ we have:

1. (lim inf inequality): $\forall (x_j) \to x :\quad f_\infty(x) \leq \liminf_j f_j(x_j)$
2. (lim sup inequality): $\exists (x_j) \to x :\quad f_\infty(x) \geq \limsup_j f_j(x_j)$

if this conditions hold, we say that f_∞ is the Γ-limit of (f_j) and we denote it as

$$f_\infty = \Gamma - \lim f_j.$$

Notice that if (x_j) satisfies the second condition of the definition we have, for this sequence:

$$f_\infty(x) \leq \liminf_j f_j(x_j) \leq \limsup_j f_j(x_j) \leq f_\infty(x) \Rightarrow f_\infty(x) = \lim_j f_j(x_j)$$

So, we can replace the second condition of the definition by:

2′. (existence of a recovery sequence) $\exists (x_j) \to x : \quad f_\infty(x) = \lim_j f_j(x_j)$

Notice that the Γ-limit, if it exists, is unique and lower semi-continuous. Moreover, every sequence f_j admits a subsequence which Γ-converges.

In the case where the family of functions depends on a continuous parameter, we have to make a precise definition of the Γ-convergence, obviously this definition will rely on the Γ-convergence for a discrete sequence of functions, and for this case we have the following definition:

Definition 2.2.3. We say that $f_\varepsilon : X \to \overline{\mathbb{R}}$ Γ-converges when $\varepsilon \to 0$ to f_0 if:

$$\forall (\varepsilon_j) \to 0 \ \text{ we have: } \Gamma - \lim_j f_{\varepsilon_j} = f_0$$

Let us enounce now the principal properties of the Γ-convergence:

Proposition 2.2.4. *Assume that $f_j : X \to \overline{\mathbb{R}}$ Γ-converges to f. Then, the following statements hold:*

1. *If $g : X \to \overline{\mathbb{R}}$ is a continuous function, then $f_j + g$ Γ-converges to $f + g$.*
2. *Let $t_j \to 0$. Then, every cluster point of the sequence of sets*

$$\{x \in X : \ f_j(x) \leq \inf_X f_j + t_j\}$$

minimizes f.

3. *Assume that the functions f_j are lower semicontinuous and for every $t \in [0, \infty)$ there exists a compact set $K_t \subset X$ with:*

$$\{x \in X : f_j(x) \leq t\} \subset K_t \quad \forall j \in \mathbb{N}.$$

Then, the functions f_j have minimizers in X, and any sequence x_j of minimizers of f_j admits subsequence converging to some minimizer of f

There are many characterizations of Γ-convergence and many other properties of interest about it, we just need the ones presented here in order to prove the Γ-convergence for some family of functionals of our interest, so, for more information of this variational convergence we suggest the refer the reader to check the following references: [14], [13], [9].

2.3 Some useful results

In this section we review some results of interest, that will be useful in our work, first of all we review the well known Euler-Lagrange equations and then we define some other result of interest.

2.3.1 Euler-Lagrange Equations

Using the Direct Method of the calculus of variation we can prove the existence of solution for the problem

$$\min_{u\in\mathcal{C}} F(u) = \min_{u\in\mathcal{C}} \int_{\Omega} f(x, u(x), \nabla u(x))dx \quad (P),$$

the obvious question after the existence of the solution will be how to characterize it. This is important, because this is the fundamental step in order to develop computational methods to compute numerical simulations in the case of functionals which represents some real problems.

A classic result in the field of the calculus of variations give us the called *Euler-Lagrange equations*, and could be considered as the necessary first order conditions for functional minimization problems, the result can be stated as follows:

If we denote $f : \Omega \times \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ such that: $(x, u, v) \mapsto f(x, u, v)$ then:

Theorem 2.3.1. *Suppose that $f \in \mathcal{C}^1(\Omega \times \mathbb{R}^m \times \mathbb{R}^m; \mathbb{R})$. If $\overline{u} \in \mathcal{C}^1(\Omega, \mathbb{R}^m)$ is a solution of (P), then $\partial_v f(x, \overline{u}(x), \nabla\overline{u}(x))$ is differentiable and we have:*

$$\frac{d}{dt}\frac{\partial f}{\partial v}(x, \overline{u}(x), \nabla\overline{u}(x)) = \frac{\partial f}{\partial u}(x, \overline{u}(x), \nabla\overline{u}(x)), \quad x \in \Omega \tag{2.1}$$

this second order equations on $\overline{u}$ are known as the Euler-Lagrange equations for the problem (P).

This result is also valid with less regular minimizer $\overline{u}$, for example, the equations are also valid if f is superlinear (i.e. $f(x, y) \geq a+b||y||^p$ with $p > 1$), in this case the minimizer $\overline{u}$ is just Lipschitz, but the equations still holds.

2.3.2 The Fundamental Lemma of Calculus of Variations

The following result is a little lemma used in the proof of the theorem of the Euler-Lagrange equations.

Lemma 2.3.2. *Let f be a function of class $\mathcal{C}^1$ on the interval (a, b) such that:*

$$\int_a^b f(x)h(x)dx = 0$$

for all functions $h(x)$ of class $\mathcal{C}^1$ on the interval (a, b) such that $h(a) = h(b) = 0$. Then $f(x) = 0$ in $[a, b]$.

This result was generalized by du Bois-Reymond in the following version

Lemma 2.3.3. *Suppose f is in $L^1_{loc}(\Omega)$, where $\Omega \subset \mathbb{R}^N$ is an open set. If*

$$\int_\Omega f(x)h(x)dx = 0$$

for all $h \in C_0^\infty(\Omega)$, then $f(x) = 0$ a.e. in $x \in \Omega$.

These lemmas allow us to rewrite Euler-Lagrange equations for complicated functionals in a simpler PDE formulation, we refer to [2] pp. 19-22 and [25] pp. 47-60 for details.

2.4 The Non Local Denoising Filter

In [20] the author develop a new denoising method which is able to handle textured signal/images, this denoising filter is essentially a nonlocal one in the sense that it replaces a pixel value with a mean of other similar (in a sense which we will explore later) pixels but with no spatial restriction. In this section we will review this important work to explore its main results.

2.4.1 Introduction: Neighborhood Filters and NL-means

We will say that a filter (for images or videos) is a neighborhood/NBH filter, if this reduces the noise by averaging similar pixels. Note that, we can use another statistical estimates like the median.
General CCD noise models are signal dependent, fortunately, two pixels, which received the same energy from the outdoor scene, undergo the same kind of perturbations and therefore have the same noise model.

We will accept the following general assumption, which is the basic idea where this models relies:

Assumption: *At each energy level the noise model is additive and white, then denoising can be achieved by first finding out the pixels which received the same original energy, and then, averaging their observed grey levels.*

Since the original values of the image are lost, the filters proceed by picking for each pixel i, the set of pixels $\mathcal{N}_i$, spatially close to i with similar grey value. The NBH filters proceed by replacing the grey level of i, which will be denoted $u(i)$, by the average

$$NF(u(i)) = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} u(j).$$

Under the assumption that the pixels of $\mathcal{N}_i$ have the same energy as i, we have that, $NF(u(i))$ is a denoised version of $u(i)$.

Most popular NBH filters are: $\sigma-$filter (see [38]), SUSAN (see [44]) and the bilateral filter (see [45]), where the neighborhoods are Gaussian in space and grey level.

In contrast, the Non-local means, proposed by Buades et al. in 2005 (see [19]), is based in the following idea:

He extends the concept of neighborhood to a wide class, in a method which is called *non-local means (NL-means)*. These algorithms defines the neighborhood $\mathcal{N}_i$ of i under the following condition:

$j \in \mathcal{N}_i$ iff the grey level of a whole window around j is close to the grey level of the window around i. In simple words, we are relaxing the spatial constraint of the classical neighborhood filters.

2.4.2 Comparison Principles

A systematic comparison of the huge variety of denoising methods is requested. The authors consider that visual comparison of artificially noisy images with their denoised version is subjective, which is a usual technique, moreover, this comparison method depends strongly on the choice of the image, and does not permit to address the main issues: *the loss of image structure in noise and the creation of artifacts.*

The authors propose three comparison principles aiming at more objective benchmarks.

1. The first principle states that noise and only noise should be removed from an image. It has to be perceptually tested directly on an image, with no artificial noise added, then the idea is to compare the difference between the image and its denoised version for each method. We will call this difference *method noise*. With this comparison method, it is much easier to evaluate whether a *method noise* contains some structure removed from the image or not.

2. The second principle, which we will call *noise to noise*, relies on the idea that a denoising algorithm transforms a white noise into a white noise. This may be seen as a paradoxical requirement, but it is a good way to characterize artifact-free algorithms. Also, we have a powerful mathematical tools for testing: Mathematical analysis and Fourier spectrum testing.

3. The third principle, which we will call *statistical optimality*, it is restricted to neighborhood filters, and is based to the question if a given NBH filter is able or not to retrieve faithfully the neighborhood $\mathcal{N}_i$ of any pixel i.

2.4.3 Noise Model

In this section we study barely a classic model for noise, this is the model for the CCD devices and the main result of this model is an hypothesis that will made the NBH filters useful for denoising.

In CCD devices, we have three kinds of noise:

1. *Shot Noise:* This noise is proportional to the square root of the number of incoming photons in the captors during the exposure time, namely:

   $$n_0 = \sqrt{\Phi \frac{t}{h\nu} \cdot A \cdot \eta} = C\sqrt{\Phi},$$

   where Φ is the light power, $h\nu$ the photon energy, t the exposure time, A the pixel area, and η the quantum efficiency. Joining all constants in C we have the last formula, where Φ can be understood as the true image.

2. *Dark or Obscurity Noise:* We will denote this noise as n_1, and it is due to spurious photons produced by the captor itself. We can assume the dark noise to be white, additive and with zero mean.

3. *Read out Noise:* We will denote this noise as n_2, and it is another electronic additive and signal independent noise. Can be assumed to have zero mean.

Also, we have to consider another correction, a *gamma correction*, which is a nonlinear increasing contrast change, we will denote it as a function f applied to the noisy image. It is applied as an internal adjustment in rendering of images through photography, television and computer imaging. Usually we take: $f(x) = x^\alpha$ with $\alpha \in (0, 1)$

Summarizing we have:

$$u(i) = f(\Phi(i) + c\sqrt{\Phi(i)} + n_1(i) + n_2(i)).$$

When $\Phi(i)$ is large the shot noise $\sqrt{\Phi(i)}$ dominates n_1 and the signal $\Phi(i)$ dominates n_2, thus we have that

$$u(i) \sim f(\Phi(i)) + f'(\Phi(i))(C\sqrt{\Phi(i)} + n_1(i) + n_2(i)) =: f(\Phi(i)) + n(i).$$

If $\Phi(i)$ is small with respect to $n_1(i) + n_2(i)$:

$$n(i) \sim u(i) \sim f(n_1(i) + n_2(i)),$$

in the particular, and interesting, case of $\alpha = 1/2$ we have:

$$n(i) \sim \begin{cases} n_0(i) & \text{bright parts of the image} \\ \\ \sqrt{n_1(i) + n_2(i)} & \text{dark parts of the image} \end{cases}$$

In all these cases the noise is signal dependent but independent at different pixels.

In the following we aim at recovering $f(\Phi(i))$, i.e. the true image up to the unknown gamma correction. The approximations we made for $u(i)$ and the white noise and independence assumptions on n_0, n_1, n_2 legitimate the following important hypothesis:

Hyphotesis: In a digital image, the noise model at each pixel i only depends on the original pixel value $\Phi(i)$ and it is additive. Let $\mathcal{N}_i$ be the set of pixels with the same original value as i. Then $n(j), j \in \mathcal{N}_i$ is independent and identically distributed. (i.i.d.).

Remark 2.4.1. This hypothesis leads us to gives a *proof* of the correctness of NBH (and NL-means) algorithms:

Given a pixel i, let $j \in \mathcal{N}_i$ all the pixels that follow the same model of i, i.e.,

$$\forall j \in \mathcal{N}_i : \; u(j) = v(i) + n(j),$$

where $v(i)$ is a deterministic function, $n(j)$ are i.i.d. noise.

Then, considering the denoising filter:

$$NF(u(i)) := \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} u(j),$$

thanks to the hypothesis and variance formula for independent variables, we have that,

$$NF(u(i)) = v(i) + \tilde{n}(i),$$

where,

$$Var(\tilde{n}(i)) = \frac{1}{|\mathcal{N}_i|} Var(n(i)) \leq Var(n(i)),$$

i.e., these filters reduce the variance of the residual noise.

2.4.4 General Neighborhood Filters

2.4.4.1 Local NBH Filters

We will present these filters in order or complexity. The first one, and then the most primitive, is based in replacing the color of a pixel with an average of the nearby pixels colors, i.e., $\mathcal{N}_i$ is just a spatial neighborhood. The filtered value for the pixel x is given by

$$\mathcal{M}_\rho(u(x)) = \frac{1}{\pi\rho^2} \int_{\mathbb{R}^2} e^{-\frac{|x-y|^2}{\rho^2}} u(y)dy,$$

where ρ is a control parameter, roughly, the size of the neighborhood. The problem of this filter relies on the case when a spatially closer pixels of the pixel i do not have similar colors as i. When the color is replaced by an average of very distinct colors, it produces blurring in the border of the transition of colors.

This suggests the need of a model which includes a weight to discard closer, but *too much* different pixels, for the averaging, this is the idea for the Sigma filter (see [38], [46]): *Average neighboring pixels which have a similar color value,* thus the filtered value is given by

$$NF_{h,\rho}(u(x)) = \frac{1}{C(x)} \int_{B_\rho(x)} e^{-\frac{|u(x)-u(y)|^2}{h^2}} u(y)dy.$$

Only pixels inside $B_\rho(x)$ are averaged, h controls the color similarity, that is, roughly speaking, the tolerance for the color similarity, and $C(x)$ a is normalization factor.

Later, to avoid the dependence on a *Ball Neighborhoods,* we have the filters SUSAN [44] and bilateral [45], where the ball neighborhoods are replaced by exponential penalization on space, i.e., we have *bilateral* Gaussian depending on both space and grey level, that is,

$$SNF_{h,\rho}(u(x)) = \frac{1}{C(x)} \int_{\mathbb{R}^2} e^{-\frac{|x-y|^2}{\rho^2}} \cdot e^{-\frac{|u(x)-u(y)|^2}{h^2}} u(y)dy.$$

Another way to avoid the blurring of the spatial filtering $\mathcal{M}_\rho$ is considering a statistical correction due to Lee in 1980 (see [37]):

$$L\mathcal{M}_\rho(u(x)) = \mathcal{M}_\rho(u(x)) + \frac{\sigma_x^2}{\sigma_x^2 + \sigma^2}(u(x) - \mathcal{M}_\rho(u(x))),$$

where

$$\sigma_x^2 = \max\left(0, \frac{1}{\pi\rho^2} \int_{\mathbb{R}^2} e^{-}\frac{|x-y|^2}{\rho^2}(u(y) - \mathcal{M}_\rho(u(x)))^2 dy - \sigma^2\right).$$

The idea of this correction is based on the following observation: When the Gaussian mean is performed on an edge, the variance of the performed mean can become larger than the variance of the noise, this phenomena is not desired, and the correction tries to avoid this.

Bilateral filters anyway have a better performance than Lee's correction. A small comparison of this neighborhood filters can be seen on the original works, and gives us non fully acceptable results: Gaussian filtering don't maintain sharp edges, anisotropic filter removes small details and fine structures, Lee's statistical filter leave some areas untouched, then noisy, sigma and bilateral creates irregularities on the edges. This comparison make the needs to consider a new model.

2.4.4.2 Non Local Averaging

As we said before, the main idea of this model is based on the simply observation that, *the most similar pixels to a given pixel have no reason to be close of it*, for example in periodic patterns, then the idea is *to construct a filter which consider pixels with neighborhoods with similar average values as the neighborhood of the original pixel*, then we do not have spatial constraint.

Then, the proposed formula is:

$$NL(u(x)) = \frac{1}{C(x)} \int_{\Omega} \exp\left(-\frac{g_\rho * |u(x+\cdot) - u(y+\cdot)|^2(0)}{h^2}\right) u(y)dy \tag{2.2}$$

where g_ρ is a Gaussian kernel with standard deviation ρ, $C(x)$ is the normalizing factor, h acts as a filtering parameter and

$$g_\rho * |u(x+\cdot) - u(y+\cdot)|^2(0) = \int_{\mathbb{R}^2} g_\rho(t)|u(x+t) - u(y+t)|^2 dt.$$

This last formula reveals the most important characteristic of this filter, NL replace the value of $u(x)$ by a weighted mean of $u(y)$. The weight is relevant only if a Gaussian window around y is similar to the same window around x. This is the concept of self-similarity.

NL-means works great with text images, but is limited when an image have structured noise, like JPEG compression, in that case NL loose details. More specific information can be seen on the original paper.

2.4.5 Principles for Denoising Algorithms Evaluation

We will enounce the formal assertions for this principles that we mentioned before in the introduction.

2.4.5.1 Method Noise

As we said before, the idea of this principle is to evaluate if an algorithm just removes noise, or it also removes some structure of the image.

Definition 2.4.2. Let u be am image, not necessarily noisy, and D_h a denoising operator depending on h. The method noise of u is the image difference:

$$n(D_h, u) = u - D_h(u)$$

and the formal principle will be:

Principle 1: For every denoising algorithm, the method noise must be zero if the image contains no noise, and should be in general an image of independent zero-mean random variables.

Examples of the evaluation of algorithms under this principle can be seen on the original paper, anyway, roughly speaking, the NL-means have the best results for this principle.

2.4.5.2 Noise to Noise Principle

As we said before, the idea of this principle is: *accepting that no algorithm can remove all the noise from an image, at least we want to transform noise in noise with less variance.* This is a way to check if an algorithm reduces the noise, and also do not create artifacts on images.

Principle 2: A denoising algorithm must transform a white noisy image into a white noisy image, with lower variance.

As we said before, this principle have a good way to be checked: Studying the Fourier transform of denoised image, because we know that the Fourier Transform of a white Gaussian noise is a white Gaussian noise, so, visualizing the Fourier Transform of the denoised image, we will see if it remains as a white Gaussian noise, or it have changed in wrong way, creating artifacts.

Several algorithms have been checked with this principle, that can be seen on the original paper, bilateral filters and NL-means report the best results.

2.4.6 Statistical Optimality

We will understand statistical optimality as *the ability of a generalized neighborhood filter to find the right set of pixels $\mathcal{N}_i$, performing the average yielding the new estimate for* $u(i)$, obviously, this principle will be useful just for neighborhood, or averaging in general methods.

Principle 3: A generalized neighborhood filter is optimal, if it finds for each pixel i, all and only the pixels j, having the same model as i.

Obviously is impossible to check if the pixels in $\mathcal{N}_i$ satisfy $\Phi(j) = \Phi(i)$, then in general this condition is relaxed to check if the pixels j are likely to have the same value as i. Examples are given in the original paper, anyway, this principle is more useful for movie denoising.

2.4.7 Numerical Examples

In what follows we will present some numerical examples performed in [19]

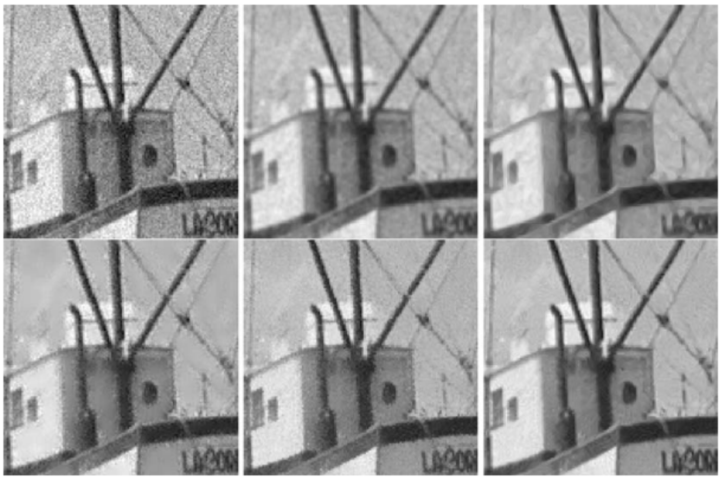


Figure 2.1: Comparison of neighborhood filters. From top to bottom and left to right: noisy image (with gaussian noise with $\sigma = 15$), Gaussian filtering, anisotropic filtering, Lee's statistical filter, sigma or bilateral filter and the NL-means algorithm. All methods except the Gaussian filtering maintain sharp edges. However, the anisotropic filtering removes small details and fine structures. These features are nearly untouched by Lee's statistical filter and therefore completely noisy. The comparison of noisy grey level values by the sigma or bilateral filter is not so robust and irregularities are created on the edges. The NL-means better cleans the edges without losing too many fine structures and details.

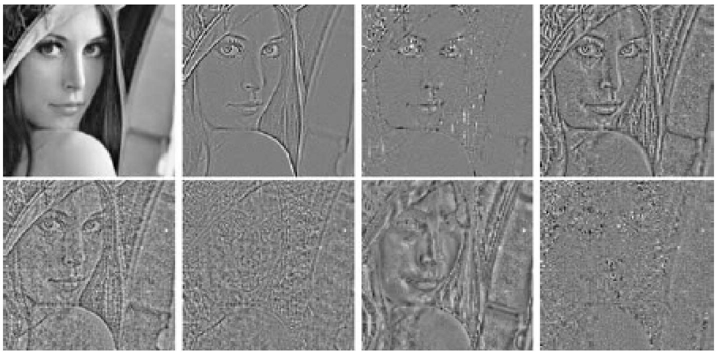


Figure 2.2: Method noise experiment on Lena (gray levels only). From top to bottom and left to right: original image, Gaussian mean, mean curvature motion, total variation minimization, translation invariant soft and hard thresholding, bilateral filter and NL-means

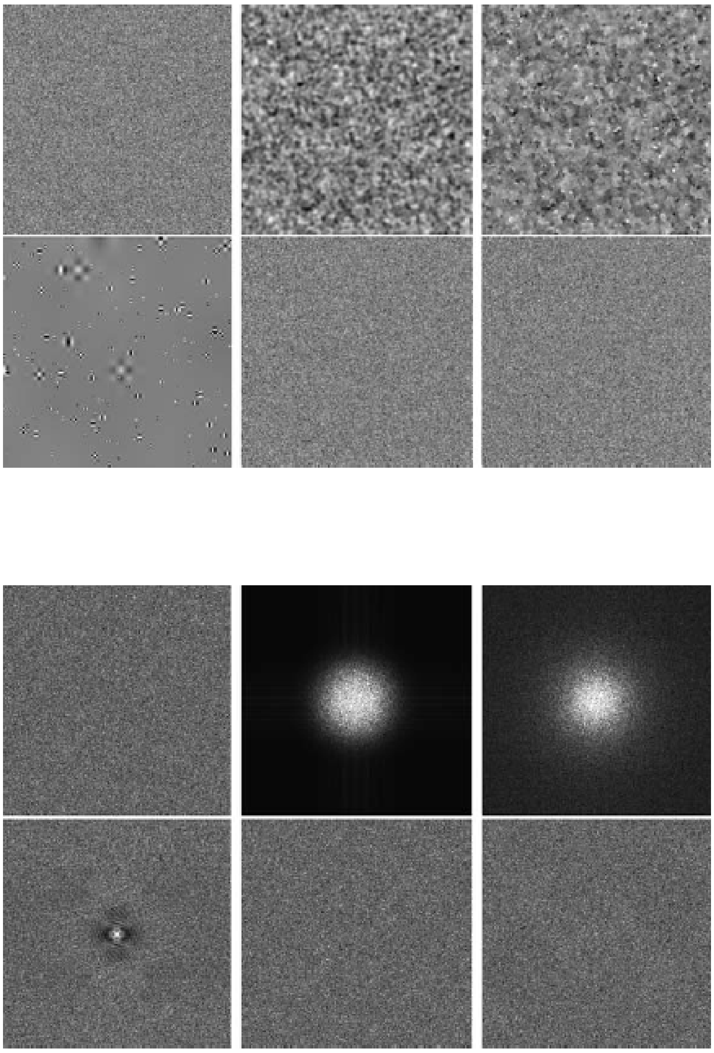


Figure 2.3: Noise to noise principle: Upper images: Application of the denoising algorithms to a noise sample. From left to right and top to bottom: noise sample, filtered noise by the Gaussian filtering, total variation minimization, hard wavelet thresholding, bilateral filter and the NL-means algorithm. The parameters of each algorithm have been tuned in order to have a filtered noise of standard deviation 2.5. For the neighborhood or bilateral filter the research zone has been fixed to 21 × 21 and for NL-means we have used the whole image. Therefore, only the h parameter has been tuned in order to obtain the desired standard deviation Lower Images: Noise to Noise principle: Fourier transforms of the filtered noises displayed in the upper images. The Fourier transform of a Gaussian white noise is a Gaussian white noise

2.5 Non Local Mumford-Shah Regularizers for Color Image Restoration

In [34] the authors develop several functionals based on approximations of the original Mumford-Shah functional with nonlocal characteristics incorporated, this nonlocal characteristics are based on the work of Buades et al. [19] and Gilboa et al. [31]. In this section we will review this important work to explore its main results.

It is important to incorporate the nonlocal characteristics in new models, because this performs better, than local methods, in image denoising and restoration when the image have textures, for example, local methods usually consider textures as noise, and then, in denoising tasks this algorithms just remove the textures.

The authors presents non local extensions for the widely known approximations for the Mumford-Shah functional, this approximations are due to Ambrosio-Tortorelli and Shah, with the primary objective of better restoration of fine structures and textures. The functionals proposed in this paper are the starting point of our work, because the proposed functionals are just used in [34] in a numerical way, moreover, the authors use this functionals for different image tasks than the one of our interest.

2.5.1 Introduction - Background

First of all, we need to recall some basic results and concepts about image regularization methods. We will extend this concepts in later chapters, but we will to introduce them slightly now, to define the model in what we will work.

2.5.1.1 Local Regularizers

The basic Mumford-Shah regularizating functional is used commonly in segmentation and restoration algorithms, it is given by the following formulation:

Given $u : \Omega \to \mathbb{R}$ and K its edge set, the MSH^1 regularizer is:

$$J^{MSH^1}(u, K) = \beta \int_{\Omega \setminus K} |\nabla u|^2 dx + \alpha \int_K d\mathcal{H}^1,$$

where $|\nabla u| = \sqrt{u_{x_1}^2 + u_{x_2}^2}$, $x = (x_1, x_2)$, $\mathcal{H}^1$ is the 1-D Hausdorff measure and $\Omega \subset \mathbb{R}^2$ is the open image domain.

Notice that the first term enforces u to be smooth everywhere, except on the edge set K, and the second term enforce to minimize the total length of edges.

In general is very hard to minimize this functional, due to its non convexity. A way to solve this problem is to consider another functionals (with better structure) which approximate this one in some sense that asserts that the minimum points of this new functionals approximates the minimum points of the original one.

Ambrosio and Tortorelli approximated the Mumford-Shah functional by considering a sequence of more regular functionals, denoted by J_ε, which converges to J^{MSH^1} in the sense of Γ-convergence. The idea of this functional is to approximate the edge set K by a smooth function v, the approximation is given by:

$$J_\varepsilon^{MSH^1}(u,v) = \beta \int_\Omega v^2 |\nabla u|^2 dx + \frac{\alpha}{2} \int_\Omega \left(\varepsilon |\nabla v|^2 + \frac{(v-1)^2}{\varepsilon} \right) dx,$$

where $0 \leq v(x) \leq 1$ represents the edges: $v(x) \sim 0$ if $x \in K = S_u$ (discontinuity set of u), $v(x) \sim 1$ otherwise; ε is a small positive constant, α, β positive weights.

If we add a fidelity term to this functional we have that a minimizer $u = u_\varepsilon$ of $J_\varepsilon^{MSH^1}$ approaches a minimizer of J^{MSH^1} as $\varepsilon \to 0$.

Another approach is given by Shah, using the total variation regularization proposed in image restoriation mainly by Rudin, Osher and Fatemi, this is very useful due to its benefits of preserving edges and convexity.

The total variation regularization is defined in the following way: given a locally integrable function u define:

$$J^{TV}(u) = \sup \left\{ \int_\Omega u \nabla \cdot \phi dx : \phi \in C_c^1(\Omega, \mathbb{R}), ||\phi||_{L^\infty(\Omega)} \leq 1 \right\},$$

which coincides with $\int_\Omega |\nabla u| dx$ when $u \in W^{1,1}(\Omega)$.

Based on this approach, Shah proposed a modified version of Ambrosio-Tortorelli approximation, by replacing the term $|\nabla u|^2$ by $|\nabla u|$ in the first term, then, the Shah approximation for the Mumford-Shah functional is given by:

$$J_\varepsilon^{MSTV}(u,v) = \beta \int_\Omega v^2 |\nabla u| dx + \alpha \int_\Omega \left(\varepsilon |\nabla v|^2 + \frac{(v-1)^2}{4\varepsilon} \right) dx.$$

This functional Γ-converges to the J^{MSTV} functional given by:

$$J^{MSTV} = \beta \int_{\Omega \backslash K} |\nabla u| dx + \alpha \int_K \frac{|u^+ - u^-|}{1 + |u^+ - u^-|} d\mathcal{H}^1 + |D_c u|(\Omega)$$

where u^+, u^- denotes the values of u *at each side of* K and $D_c u$ is the Cantor part of Du. This last functional is very similar with the total variation of $u \in BV(\Omega)$, that can be written for $K = J_u$ as:

$$J^{TV} = \beta \int_{\Omega \backslash K} |\nabla u| dx + \alpha \int_K |u^+ - u^-| d\mathcal{H}^1 + |D_c u|(\Omega).$$

The only difference is that the MSTV regularizer does not penalize the jump part, as much as, the TV regularizer does.

These functionals are considered only for monochromatic images, but is naturally extended to color images by Blomgren and Chan, which propose a color TV regularization by coupling the channels, i.e., considering:

$$J^{TV} = \int_\Omega ||\nabla u|| dx = \int_\Omega \sqrt{|\nabla u^R|^2 + |\nabla u^G|^2 + |\nabla u^B|^2} dx.$$

Bar et al. in [11] extend this idea for the Mumford-Shah approximations for color images, by replacing $|\nabla u|$ by $||\nabla u||$ in J_ε^{MSH} and J_ε^{MSTV}. Notice that the scalar-valued edge map v is common for the three channels and provides the necessary coupling between colors.

2.5.1.2 Nonlocal Methods

As we seen before in review of the paper of Buades et al. the importance of nonlocal methods is based on their well adaptation to texture denoising in contrast to standard local methods. Recall that the basic idea is, to extend the concept of neighborhood filters which replace the value of a pixel with an average of its spatial neighbors, the nonlocal filters extend this concept to the one of patch-similarity, i.e., we will replace the value of a pixel for an averaging of pixels which have similar patch values, and then, the spatial restriction is relaxed. The classical filter for this task is the NL-means filter due to Buades et al. [19]:

$$NL(f(x)) = \frac{1}{C(x)} \int_\Omega \exp\left(-\frac{d_a(f(x), f(y))}{h^2}\right) f(y) dy$$

$$d_a(f(x), f(y)) = \int_{\mathbb{R}^2} g_a(t) ||f(x+t) - f(y+t)||^2 dt$$

where d_a is the patch distance, f is the image to be filtered and g_a is a Gaussian kernel with standard deviation a, which determines the patch size.

2.5.1.3 Nonlocal Regularizers

The idea of this regularizers is to see the nonlocal filtering as a quadratic regularization based upon a nonlocal graph (a graph with weights). The most important contributions on this field are given by Gilboa and Osher.

We will need some operators from this theory, the so-called *non local differential operators over graphs*, proposed for image processing by Gilboa and Osher in [31].

Let $u : \Omega \to \mathbb{R}$ and $w : \Omega \times \Omega \to \mathbb{R}$ be a non negative and symmetric weighted functions. We define the non local gradient vector $\nabla_w u : \Omega \times \Omega \to \mathbb{R}$ as:

$$(\nabla_w u)(x, y) = (u(y) - u(x))\sqrt{w(x, y)},$$

and the norm of the nonlocal gradient of u is defined by:

$$|\nabla_w u|(x) := \sqrt{\int_\Omega (u(y) - u(x))^2 w(x, y) dy}$$

We also define the non local divergence of the vector $\vec{v} : \Omega \times \Omega \to \mathbb{R}$ by:

$$(div_w \vec{v})(x) := \int_\Omega (v(x,y) - v(y,x))\sqrt{w(x,y)}dy$$

Inspired in this operators, Gilboa and Osher in [31] proposed the following general form for nonlocal regularizing functionals:

$$J(u) = \int_\Omega \phi(|\nabla_w u|^2)dx,$$

where $s \mapsto \phi(s)$ is positive, increasing and convex in $\sqrt{s}$, and $\phi(0) = 0$.
If $\phi(s) = \sqrt{s}$, they propose the NL/TV (NonLocal Total Variation) regularizer:

$$J^{NL/TV}(u) = \int_\Omega |\nabla_w u|dx = \int_\Omega \sqrt{\int_\Omega (u(y) - u(x))^2 w(x,y)dydx},$$

which coincides, in the 2D local case, to $J^{TV}(u) = \int_\Omega |\nabla u|dx$

2.5.2 Proposed Nonlocal Mumford-Shah Regularizers

Based on the above approximations, the authors propose nonlocal versions of the approximating functionals of Ambrosio-Tortorelli and Shah to the Mumford-Shah functional. It is important to recall that they also incorporate the vector case, i.e. color images, in their formulation, in the way as we seen above.

Then, the general model proposed by the authors is:

$$\begin{aligned} J^{NL/MS}(u,v) &:= \underbrace{\beta \int_\Omega v^2 \phi(||\nabla_w u||^2)dx}_{F_{reg}^{NL/MS}} + \underbrace{\frac{\alpha}{2}\int_\Omega \left(\varepsilon|\nabla v|^2 + \frac{(v-1)^2}{\varepsilon}\right)dx}_{F_{AT}} \\ &= F_{reg}^{NL/MS}(u,v) + F_{AT}(v), \end{aligned} \tag{2.3}$$

where $u : \Omega \to \mathbb{R}^3$, $v : \Omega \to [0,1]$, and $\phi(s) = s$ or $\phi(s) = \sqrt{s}$, the first choice correspond to NL/MSH^1 and the second to MS/TV, i.e.,

$$\begin{aligned} J^{NL/MSH^1}(u,v) &:= \underbrace{\beta \int_\Omega v^2 ||\nabla_w u||^2 dx}_{F_{regAT}^{NL/MS}} + \frac{\alpha}{2}\int_\Omega \left(\varepsilon|\nabla v|^2 + \frac{(v-1)^2}{\varepsilon}\right)dx \\ &= F_{regAT}^{NL/MS}(u,v) + F_{AT}(v) \end{aligned} \tag{2.4}$$

$$\begin{aligned} J^{NL/MSTV}(u,v) &:= \underbrace{\beta \int_\Omega v^2 ||\nabla_w u||dx}_{F_{regS}^{NL/MS}} + \frac{\alpha}{2}\int_\Omega \left(\varepsilon|\nabla v|^2 + \frac{(v-1)^2}{\varepsilon}\right)dx \\ &= F_{regS}^{NL/MS}(u,v) + F_{AT}(v), \end{aligned} \tag{2.5}$$

and recall that:

$$||\nabla_w u||(x) = \sqrt{\sum_{i=R,G,B} |\nabla_w u^i|^2(x)} = \sqrt{\sum_{i=R,G,B} \int_\Omega (u^i(x) - u^i(y))^2 w(x,y)dy}.$$

As we said before, adding a fidelity term to this functionals, we will be able to perform a specific restoration task, we will discuss this in the next section.

As an additional remark the authors recall that in the practice the weight function that will be used is the classic NL-means weight (given an image q):

$$\begin{aligned} w(x,y) &= \exp\left(-\frac{d_a(q(x),q(y))}{h^2}\right) \\ d_a(q(x),q(y)) &= \int_{\mathbb{R}^2} g_a(t)||q(x+t) - q(y+t)||^2 dt \end{aligned} \tag{2.6}$$

and we use search windows $S(x) = \{y \ / \ |x-y| \le r\}$.

The functional $J^{NL/MS}(u,v)$ is the most important functional of this work, we will try to study it in the context of segmentation (i.e. with a L^2 similarity term) in a theoretical way. This is motivated because in numerical testings this functional seems to approximate in a good way the original Mumford-Shah segmentation functional, but nothing has been proved before about the theoretical properties of this functional, this will be the central objective of this work and also, we will perform independent numerical testings for this functional and we will try to improve its performance for some tasks of our interest.

Chapter 3

The Segmentation Problem

3.1 Introduction to the Problem of Segmentation

Segmentation of an image can be understood basically as the process of partitioning a digital image into multiple sets of pixels. The goal of this process is to simplify and/or change the representation of an image into something, that is more meaningful and easier to analyze. More precisely, in this process we could assign a label to each pixel in an image, such that pixels with the same label share a common visual characteristic, for example: color, intensity, texture.

Segmentation is the key process in order to differentiate objects in a digital image, practical applications of image segmentation are:

1. Medical Imaging: Location of tumors and other pathologies, Measure tissue volumes, Computer-guided surgery.

2. Satellite Imaging: Location and classification of several objects: roads, forests, etc.

3. Human recognition: Face, iris and fingerprint recognition.

4. Military Imaging and Surveillance: Camouflage recognition.

and, in general, any application which needs the identification of objects in a digital image.

In order to develop models for image segmentation, it is important to get a clear and precise objective of what can lead us to *differentiate one region or object* from another.

Let's review a simple example for this purpose, which is presented in [10] Chapter 4:

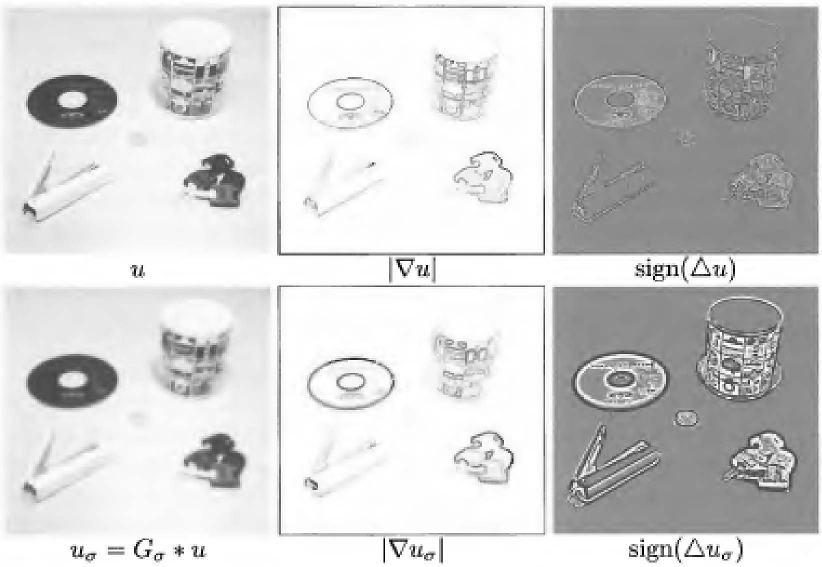

u
$|\nabla u|$
$\text{sign}(\triangle u)$
$u_\sigma = G_\sigma * u$
$|\nabla u_\sigma|$
$\text{sign}(\triangle u_\sigma)$

Figure 3.1: Edges and Derivatives

We can see in Figure 3.1 that an effective way to differentiate regions in an image is detecting the edges or contours of each physical object. Edge detection has been studied widely in computer vision. The classical approaches are based on local differential properties of an edge, for instance, as we see on the Figure 3.1, on the first and/or second derivatives of the image. The consideration of a regularization, applying a convolution mask, of the input image (second row of images) is based on the fact that this process enhance edges.

This idea can be resumed as follows: *Starting from an image u_0, we look for a pair (u, K), such that, u is a nearly piecewise constant approximation of u_0 and K which corresponds to the set of edges.* This idea was proposed by Mumford and Shah in 1989, and will be our choice to work with.

It is important to notice that there are other ways to perform segmentation, the decision of which one we use depends of the objective of the process: differentiate physical objects or reveal object structure for example. In the first case, one can consider a big curve K which encloses the group of objects to be differentiated and make it evolve, with some defined criteria, until it reaches the boundary of each object. This idea was proposed by Kass, Witkin and Terzopoulos in [35] and it is called *active contours*, but we won't work with that.

In our case, the edge detection will be our principal objective, so, we will start exploring the best way to perform segmentation based on edge detection: Considering the *Mumford and Shah Functional.*

3.2 Mumford-Shah Functional

Let $\Omega \subset \mathbb{R}^N$ an open and bounded set, with $N = 2, 3$ and u_0 is the input image, which we can assume, without loss of generality, such that, $0 \leq u_0(x) \leq 1,\ a.e.\ x \in \Omega$.

We will search for a pair (u, K), where $K \subset \Omega$ is the set of discontinuities, which will represent the edges of the objects, such that, minimizes some functional that we will define before.

The idea of this functionals is to define a minimization problem in which the following conditions prevail:

1. u is closer to u_0.
2. u is constant in regions away from the set K.
3. The length of the set K is the minimum possible.

Considering these conditions, Mumford and Shah proposed the following functional, known as the *Mumford and Shah Functional*:

$$F(u, K) = \int_{\Omega \setminus K} (u - u_0)^2 dx + \alpha \int_{\Omega \setminus K} |\nabla u|^2 dx + \beta \int_K d\sigma \tag{3.1}$$

where α and β are nonnegative constants, and $\displaystyle\int_K d\sigma$ is the length of K. This functional is defined in [**?**].

Then, the segmentation problem can be solved, in this model, by solving the following variational problem:

To find suitable (u, K) *such that minimizes* $F(u, K)$.

Notice that, if we want to apply the direct method of the calculus of variations, we need to define a correct spaces for the function u and for the edge set K. It is clear that we cannot impose that K to be in the space of piecewise $\mathcal{C}^{1,1}$ curves, since one cannot hope to obtain compactness properties, and hence any existence theorem with this restriction. Regularity issues about K should be studied a posteriori. To overcome this difficulty we have to look for solutions in a wider class of sets of finite length. We will define the length of K as its $(N-1)$-dimensional Haussdorf measure $\mathcal{H}^{N-1}(K)$, this is the classical way to extend the notion of length to nonsmooth sets.

Considering this relaxation, we have to rewrite the Mumford and Shah functional as:

$$F(u, K) = \int_{\Omega \setminus K} (u - u_0)^2 dx + \alpha \int_{\Omega \setminus K} |\nabla u|^2 dx + \beta \mathcal{H}^{N-1}(K). \tag{3.2}$$

It is interesting to see that this functional is minimal, in the sense that, if we remove one of the terms involved, then $\inf F(u, K) = 0$ and we could get trivial solutions. For example, if we remove the first term, then $u = constant$ and $K = \emptyset$ is a solution, and if we remove the second term, then $u = u_0$ and $K = \emptyset$ is a solution. Notice also that, in principle, we have $u \in H^1(\Omega)$.

In the next sections we will discuss how the minimization problem have to be defined, in order to, at least, expect non trivial solutions and then we will review some results about the existence of solutions.

3.3 Spaces of Work: $BV(\Omega)$, $SBV(\Omega)$ and $GSBV(\Omega)$

In order to have good properties on the functionals involved in this chapter, we will have to consider this functionals defined on some function spaces which are not the classical spaces like the Sobolev ones.

First of all we need to define some preliminary things: Given $\Omega \subset \mathbb{R}^N$ and $u : \Omega \to [-\infty, +\infty]$ a measurable function, we define the approximate upper limit of u at $x \in \Omega$ as:

$$u_+(x) := \inf\left\{t \in [-\infty, +\infty] : \lim_{\rho \downarrow 0} \frac{|\{y : \; u(y) > t\} \cap B_\rho(x)|}{\rho^N} = 0\right\}$$

where $B_\rho(x)$ is the ball of radius ρ centered at x and $|E|$ denotes the Lebesgue measure of the set E. The approximate lower limit $u_-(x)$ is defined in the same way:

$$u_-(x) := -(-u)_+(x)$$

The set:

$$S_u = \{x \in \Omega : \; u_-(x) < u_+(x)\}$$

is the set of essential discontinuities of u, it is a (Lebesgue)-negligible Borel set. If $x \notin S_u$, we say that u is approximately continuous at x and we write:

$$\tilde{u}(x) = u_-(x) = u_+(x) = aplim_{y \to x} u(y)$$

The first space that we will define is the space of bounded variation functions: $BV(\Omega)$, this space is well known from measure theory and will be the starting point to define the most interesing spaces to work with:

Definition 3.3.1. Let $u \in L^1(\Omega)$; we say that u is a function of bounded variation in Ω if the distributional derivative of u is representable by a finite Radon measure in Ω, i.e. if:

$$\int_\Omega u \frac{\partial \phi}{\partial x_i} dx = -\int_\Omega \phi dD_i u \quad \forall \phi \in \mathcal{C}_c^\infty(\Omega), \quad i = 1, \ldots, N \tag{3.3}$$

for some $\mathbb{R}^N$-valued measure $Du = (D_1 u, \ldots, D_N u)$ in Ω. The vector space of all functions of bounded variation in Ω is denoted by $BV(\Omega)$

Notice that this definition is equivalent to the following one

Definition 3.3.2. Let $u \in L^1(\Omega)$; Then u is a function of bounded variation in Ω if the distributional derivative of u is a vector valued measure with finite total variation in Ω. Where total variation is defined by:

$$V(f,\Omega) := \sup \left\{ \int_\Omega u div\varphi dx : \varphi \in [C_c^1(\Omega)]^N, \ ||\varphi||_\infty \leq 1 \right\}$$

An important theorem about this spaces is the following one:

Theorem 3.3.3. *If $u \in BV(\Omega)$, the set S_u is countably $(\mathcal{H}^{N-1}, N-1)$-rectifiable, i.e.:*

$$S_u = \bigcup_{i \geq 1} K_i \cup \mathcal{N}$$

where $\mathcal{H}^{N-1}(\mathcal{N}) = 0$ and each K_i is a compact subset of a $\mathcal{C}^1$-hypersurface Γ_i.

A consequence of this, is that there exists a Borel function $\nu_u : S_u \to \mathbb{S}^{N-1}$ such that $\mathcal{H}^{N-1}$-a.e. in S_u the vector $\nu_u(x)$ is normal to S_u at x in the sense that it is normal to Γ_i if $x \in K_i$. For every $u, v \in BV(\Omega)$, we must therefore have $\nu_u = \pm\nu_v$, $\mathcal{H}^{N-1}$-a.e. in $S_u \cap S_v$.

Notice that, for every $u \in BV(\Omega)$, the measure Du can be decomposed as follows:

$$Du = \nabla u(x)dx + (u_+ - u_-)\nu_u \mathcal{H}^{N-1} \llcorner S_u + Cu \tag{3.4}$$

where ∇u is the approximate gradient of u, defined a.e. in Ω by:

$$aplim_{y \to x} \frac{u(y) - u(x) - \langle \nabla u(x), y - x \rangle}{|y - x|} = 0$$

and $\mathcal{H}^{N-1} \llcorner S_u$ is the restriction of the $N-1$ dimensional Hausdorff measure to the set S_u, and Cu is the Cantor part of the measure Du, which is singular with respect to the Lebesgue measure and such that $|Cu|(E) = 0$ for any E with $\mathcal{H}^{N-1}(E) < \infty$.

This decomposition lead us to define the space of Special functions of Bounded Variation, denoted by $SBV(\Omega)$:

Definition 3.3.4. We say that a function $u \in BV(\Omega)$ is a special function of bounded variation if $Cu = 0$ where Cu is the same as in 3.4. This condition means that the singular part of the distributional derivative Du is concentrated on the jump set S_u. This vector space is denoted by $SBV(\Omega)$

This space is not trivial, in the sense that we can find a function u such that $u \in BV(\Omega) \setminus SBV(\Omega)$, the classical example is to take u as the well known Cantor-Vitali function.

Finally we define the define the space of Generalized SBV functions:

Definition 3.3.5. We say that a measurable function $u : \Omega \to [-\infty, +\infty]$ is a generalized SBV function if for any $k > 0$ the function $u^k = (-k \wedge u) \vee k \in SBV(\Omega)$, where $X \wedge Y = \min(X, Y)$ and $X \vee Y = \max(X, Y)$. We denote this vector space by $GSBV(\Omega)$.

This last two spaces were defined by Ambrosio in [4] in order to obtain compactness theorems to work in the minimization of functionals in free-discontinuity problems.

An extensive treatment of this spaces can be found in [7] in chapters 3 and 4.

3.3.1 Existence of Minima

If we want to apply the direct method of the calculus of variations, it is necessary to find a suitable topology, such that, we can ensure the lower semicontinuity of F and compactness of the minimizing sequence. Notice that we have two unknowns in the functional u and K, which have a very different nature, u is a function defined on a N dimensional space, and K is an $(N-1)$ dimensional set. So, we will have a big difficulty in our purpose, basically based on the term $\mathcal{H}^{N-1}(K)$. Because we have the following known result about the Hausdorff's measure:

Theorem 3.3.6. *Let E be a Borel set of $\mathbb{R}^N$, with topological boundary ∂E, then, the map*

$$E \to \mathcal{H}^{N-1}(\partial E)$$

is not lower semicontinuous with respect to any compact topology.

Due to this result, it is necessary to find another formulation for $F(u, K)$. A new, and suitable, formulation involves the use of the space $BV(\Omega)$ that we defined before. The idea is that we can identify the set of edges K with the jump set of u, denoted by S_u, this allows us to eliminate the unknown K. Then, we have to consider the functional:

$$G(u) := \int_\Omega (u-u_0)^2 dx + \alpha \int_\Omega |\nabla u|^2 dx + \beta \mathcal{H}^{N-1}(S_u) \tag{3.5}$$

Since we have removed the problem of the set K, now we have a functional depending just on the function u, by passing to the space $BV(\Omega)$. Unfortunately if we want to minimize G in $BV(\Omega)$ we will have the following bad behaviour: *There exists functions on $BV(\Omega)$, for example the Cantor-Vitali function v^*, such that, they are non constant, continuous and with approximate derivative equal to zero a.e.*, thus we have:

$$G(v^*) = \int_\Omega (v^* - u_0)^2 dx \geq \inf_{u \in BV(\Omega)} G(u) \geq 0,$$

but it can be proved that this kind of functions are dense in $L^2(\Omega)$ so, in that case we have:

$$\inf_{u \in BV(\Omega)} G(u) = 0,$$

this implies in particular, that the infimum cannot be achieved in $BV(\Omega)$ in general. Details can be seen on [5].

In order to avoid this bad behavior, we have to work in the space $SBV(\Omega)$, the space of Special Functions of Bounded Variation, that we defined before. Recall that the only difference of this space in comparison with $BV(\Omega)$ is that, in $SBV(\Omega)$ we force that these functions to have support outside *Cantor-like sets.* So, the suitable space for the functional 3.5 will be $SBV(\Omega)$, and using the following compactness theorem on this space we can conclude, using the direct method of calculus of variations, the existence of minimizers for $G(u)$:

Theorem 3.3.7. *Let $u_n \in SBV(\Omega)$ be a sequence of functions such that there exists a constant C with $|u_n(x)| \leq C < \infty$ a.e. $x \in \Omega$ and*

$$\int_\Omega |\nabla u_n|^2 dx + \mathcal{H}^{N-1}(S_{u_n}) \leq C.$$

Then there exists a subsequence u_{n_k} converging a.e. x to a function $u \in SBV(\Omega)$. Moreover ∇u_{n_k} converges weakly in $L^2(\Omega)^N$ to ∇u and

$$\liminf \mathcal{H}^{N-1}(S_{u_{n_k}}) \geq \mathcal{H}^{N-1}(S_u).$$

Finally, we want to relate the minima of $G(u)$ and minima of $F(u,K)$, which is the original functional. Ambrosio in [3] have proved the following theorem:

Theorem 3.3.8. *Let $K \subset \Omega$ be a closed set, such that, $\mathcal{H}^{N-1}(K) < \infty$ and let $u \in H^1(\Omega \setminus K) \cap L^\infty(\Omega)$. Then $u \in SBV(\Omega)$ and $S_u \subset K \cup R$ with $\mathcal{H}^{N-1}(R) = 0$.*

This theorem leads us to relate the minima of the following problems:

$$\inf_{u,K} \left\{ F(u,K),\ u \in H^1(\Omega \setminus K) \cap L^\infty(\Omega), K \subset \Omega, K \text{ closed}, \mathcal{H}^{N-1}(K) < \infty \right\} \quad (P_1)$$

and

$$\inf_u \{G(u),\ u \in SBV(\Omega) \cap L^\infty(\Omega)\} \quad (P_2)$$

in the following way:

Notice that, from Theorem 3.2.3 we have:

$$\inf P_2 \leq \inf P_1,$$

but from Theorem 3.2.2, applied to a minimizing sequence of (P_2), such that,

$$|u_n|_{L^\infty} \leq |u_0|_{L^\infty},$$

which is always satisfied by a truncation argument, we have a solution u for $(P2)$. For such minimizer De Giorgi et al proved in [27] Lemma 5.2 that:

$$\mathcal{H}^{N-1}((\Omega \cap \overline{S}_u) \setminus S_u)) = 0$$

So, setting $K = \Omega \cap \overline{S}_u$ we get a solution of (P_1) and finally we can conclude that:

$$\min(P_1) = \min(P_2)$$

3.3.2 Approximation Schemes

We have established the existence of minimizers for the Mumford-Shah functional $F(u,K)$, in order to compute minimizers we would like to have optimality conditions associated with this functional, this is possible and, in fact, there exists results of this conditions, but due to the numerical complexity of this conditions (specially in order to reproduce conditions for the set K), we will prefer to avoid this results. Moreover, our approach in order to compute numerically minimizers for this functional will be based in compute the minimizers for a suitable family of functionals F_ε, which Γ-converges to the Mumford-Shah functional $F(u, K)$.

The main advantage of consider this approach is that we can work with sequence of functionals F_ε which have better properties (for example, we can define this sequence of functionals on Sobolev spaces, and in this case we can develop computational algorithms based on solving Euler-Lagrange equations which are suitable in this case) to work with. The convergence of F_ε to F as $\varepsilon \to 0$ will be understood in the Γ-convergence sense.

There are several approaches to approximate the functional $F(u, K)$, for example:

1. Approximation by elliptic functionals:
   In this approach the set S_u (or K) is replaced by an auxiliary variable v, which is a function, that approximates the characteristic function $\mathbf{1}_{S_u^c}$, so, $v(x) \sim 0$ if $x \in S_u$ and $v(x) \sim 1$ if $x \notin S_u$. There are several functionals for this purpose, the most important is the functional defined by Ambrosio and Tortorelli in [6]:

   $$F_\varepsilon(u, v) = \int_\Omega (u - u_0)^2 dx + \int_\Omega v^2 |\nabla u|^2 dx + \int_\Omega \left(\varepsilon |\nabla v|^2 + \frac{1}{4\varepsilon}(v-1)^2 \right) dx \quad (3.6)$$

   we will discuss about how this functional Γ-converge to $G(u)$ in the next section.

2. Approximation by finite-difference schemes:
   This could be seen as the most natural way to approximate the Mumford-Shah functional, in the sense that this could be seen as a numerical discretization of the functional. The idea is to consider $u(x)$ as a discrete image defined on a mesh with step h and F^h as the discrete version of the Mumford-Shah functional. The most successful works with this kind of approximation are made by Chambolle, which defines this kind of approximation on [21] and then proved the Γ-convergence of a proposed functional of this kind in [22].
   An example family of functionals of this type are the following: In the $1D$ case, let:

   $$g_k^h \text{ a reasonable discretization of the input image } u_0, \quad u^h = (u_k^h)_{kh \in \Omega}$$

   Then: $F^h(u^h) = h \sum_k W_h \left(\frac{u_{k+1}^h - u_k^h}{h} \right) + h \sum_k (u_k^h - g_k^h)^2$. Where $W_h(t) = \min(t^2, 1/h)$, and he proved that F^h Γ-converges to

   $$F(u) = \int_\Omega (u - u_0)^2 dx + \int_{\Omega \setminus S_u} (u')^2 dx + |S_u| \quad u \in GSBV(\Omega)$$

We will focus in the study of the Γ-convergence for the case of approximation functionals, in specific in the Ambrosio-Tortorelli functional of approximation, because this functional is very similar to the functional that we will study in the later chapters, moreover, the techniques to prove the Γ-convergence of our objective functional will be similar.

3.3.3 Γ Convergence in the case of Ambrosio-Tortorelli functional

As we detailed before, the key point of the Γ convergence is the fact that this kind of convergence is able to ensure the convergence of minima, in the sense that if F_ε Γ converges to F, then if we have a sequence of minimizers of the family F_ε then the limit of that sequence is a minimizer of F.

In this section we will briefly describe the Γ convergence to the Mumford-Shah functional of the Ambrosio-Tortorelli functional, in order to study this convergence we will need to convince in first place that the Ambrosio-Tortorelli functional (or a modification, as we see later) actually have minimizers, otherwise the most important property of the Γ convergence will be satisfied trivially, and this process will have no value.

In the following discussion we will consider the Ambrosio-Tortorelli functional defined in 3.6

3.3.3.1 Existence of Minima for F_ε

Given $\epsilon > 0$ we have that $F_\varepsilon(u, v)$ is well defined in $\mathcal{C} = \{(u, v) \in H^1(\Omega)^2 \mid 0 \leq v \leq 1\}$, moreover, is easy to see that the functional is weakly lower semicontinuous on $\mathcal{C}$. In order to apply the direct method of the calculus of variations it suffices to bound (on $\mathcal{C}$) the minimizing sequence $(u_\varepsilon^n, v_\varepsilon^n)$ independently of n. Is easy to bound in $L^2(\Omega)$ the sequences $u_\varepsilon^n, v_\varepsilon^n$ and ∇v_ε^n, but we don't have a direct bound for ∇u_ε^n (because we have no control on the term $\int_\Omega (v_\varepsilon^n)^2 |\nabla u_\varepsilon^n|^2 dx$).

The natural question arises: How to fix this problem?

The solution is quite simple: we just have to modify slightly the functional $F_\varepsilon(u, v)$, let us consider the perturbed functional:

$$\tilde{F}_\varepsilon(u, v) = F_\varepsilon(u, v) + o(\varepsilon) \int_\Omega |\nabla u|^2 dx \tag{3.7}$$

With this small modification, the functional $\tilde{F}_\varepsilon$ is coercive on $\mathcal{C}$, and in this case we can apply the direct method of the calculus of variations. We resume this result on the following theorem:

Theorem 3.3.9. *Given $u_0 \in L^\infty(\Omega)$, the problem*

$$\inf_{\mathcal{C}} \tilde{F}_\varepsilon(u, v)$$

admits a solution $(u_\varepsilon, v_\varepsilon)$ which satisfy: $|u_\varepsilon|_{L^\infty} \leq |u_0|_{L^\infty}$.

Proof. This is just an application of the direct method of the calculus of variations 2.1. □

Notice that in this way we are able to ensure something that the functional F_ε was unable to (and, in fact, can be proved that the original functional will not be able): The 'sequence of minimizers' $(u_\varepsilon, v_\varepsilon)_\varepsilon$ for the functionals $\tilde{F}_\varepsilon$ converge, as $\varepsilon \to 0$ to a limit $(u,v) \in H^1(\Omega)^2$, this behaviour is needed, because the result for the minimizers of $F(u,K)$ asserts that $u \in H^1(\Omega \setminus K)$, then, in order to study Γ convergence, we need at least that the limit of the sequence of minimas for each $\tilde{F}_\varepsilon$ lives on the space of the minimas for the original functional.

Then we can conclude that this modification solves two problems: Ensures the existence of minima and forces the limit of the sequences of minima for the approximating functional lives on the appropiate space.

3.3.3.2 The Γ Convergence Theorem

In this section we finally enounce the most important theorem for the classical Ambrosio-Tortorelli functional: The Γ convergence of $\tilde{F}_\varepsilon$ to the Mumford-Shah functional.

Theorem 3.3.10. *Let $\Omega \subset \mathbb{R}^N$ be an open set, and let $\tilde{F}_\varepsilon : L^1(\Omega) \times L^1(\Omega) \to [0,\infty]$ defined by:*

$$\tilde{F}_\varepsilon(u,v) = \begin{cases} \int_\Omega (u-u_0)^2 dx + \int_\Omega v^2 \|\nabla u\|^2 dx + \int_\Omega \left(\varepsilon |\nabla u|^2 + \frac{1}{4\varepsilon}(1-v)^2\right) dx \\ \textit{if } (u,v) \in H^1(\Omega)^2, \ 0 \leq v \leq 1 \\ +\infty \quad \textit{otherwise} \end{cases}$$

and Let $F : L^1(\Omega)^2 \to [0,\infty]$ be defined by

$$G(u,v) = \begin{cases} \int_\Omega (u-u_0)^2 dx + \int_\Omega \|\nabla u\|^2 dx + \mathcal{H}^{N-1}(S_u) & \textit{if } u \in GSBV(\Omega) \textit{ and } v = 1 \textit{ a.e.} \\ +\infty & \textit{otherwise} \end{cases}$$

Then $\tilde{F}_\varepsilon(u,v)$ Γ converges to $G(u,v)$ in the $L^1(\Omega) \times L^1(\Omega)$ topology. Moreover, $\tilde{F}_\varepsilon$ admits a minimizer $(u_\varepsilon, v_\varepsilon)$ such that, up to subsequences, u_ε converges in $L^1(\Omega)$ to a minimizer of G, $u \in SBV(\Omega)$, and

$$\inf \tilde{F}_\varepsilon \to \inf G(u,v)$$

There are several proofs for this result, the only difference is the topology and generality used on each cases, detailed references are: [6] Theorem 1.1, [14] Theorem 8.1 and [13] Theorem 3.15.

Chapter 4

The Non Local Segmentation Problem

4.1 The Non-Local Ambrosio-Tortorelli Functional

As we mentioned before in 2.5, in order to perform several image processing tasks including the advantages of Buades-like denoising filters, we can consider the functional 2.4 with a suitable similarity term, which, in order to perform segmentation, is:

$$J_{sim}(u) := \beta \int_{\Omega} (u - u_0)^2 dx$$

where u_0 is the input image for the segmentation.

Therefore, the functional on study will be, given a bounded interval $\Omega \subset \mathbb{R}$:

$$\begin{aligned} F_{\varepsilon}^{NLAT}(u,v) &:= \alpha \int_{\Omega} v^2 |\nabla_w u|^2 dx + \frac{\nu}{2} \int_{\Omega} \varepsilon |\nabla v|^2 + \frac{(1-v)^2}{\varepsilon} dx + J_{sim}(u) \\ &= \alpha \int_{\Omega} v^2 |\nabla_w u|^2 dx + \frac{\nu}{2} \int_{\Omega} \varepsilon |\nabla v|^2 + \frac{(1-v)^2}{\varepsilon} dx + \beta \int_{\Omega} (u - u_0)^2 dx \\ &= G_{\varepsilon}(u,v) + J_{sim}(u) \end{aligned} \tag{4.1}$$

recalling that the non-local gradient norm is defined by:

$$|\nabla_w u|^2(x) = \int_{\Omega} (u(y) - u(x))^2 w(x,y) dy$$

For a suitable weight function $w(x,y)$ (typically the Buades non-local denoising filter weight function given by 2.6), we will study how the non-local gradient function should be (re)defined in order to obtain interesting theorical properties. Notice that in general we will work with the term $G_{\varepsilon}(u,v)$ due to the continuity of $J_{sim}(u)$ in L^2 (and therefore L^1)

The minimization of this functional will reveal the non-local segmentation of the image u_0, in this chapter we will study this functional, we will explain why the classical formulation for this functional is not appropiate for Γ-convergence purposes, we will re-define the formulation in order to obtain interesting results in Γ-convergence sense and we finally prove this important results.

We will first reviewing the classical assumptions that we consider on the weight function $w(x,y)$ and study briefly the behaviour of the non-local gradient, next we will define the spaces of work in this chapter, after that we will review some results on 'Perimeter-like' functionals that will be useful for our main proof and then we will study why we have to redefine the non-local functional. Finally we will prove the Γ-convergence result for the modified functional.

4.2 Some assumptions on $w(x,y)$ and behaviour of $|\nabla_w u|^2(x)$

In this section we will study briefly the behaviour of the term $|\nabla_w u|^2(x)$, in order to understand 'how it works', this will let also to 'begin to see' that the classical formulation of the functional is not enough for Γ-convergence purposes and is the starting point to believe that the term should be redefined. We will also enounce some assumptions that we will use in the rest of this chapter, some of them will let us to model some terms in a more representative way, other will let us to simplify the original model.

4.2.1 Assumptions on $w(x,y)$

We first have to notice that, usually, the weight function $w(x,y)$ will be chosen such that: Given an image/signal u_0, that will be segmented, the function $w(x,y)$ is constructed by 2.6, after the construction of w (which depends on u_0) the minimization process never deal again with a 'reconstruction' or 'update' of $w(x,y)$, thus, the first assumption for theorical purposes will be:

The function $w(x,y)$ is fixed in the minimization process

We also consider, due to classical form of $w(x,y)$ given by 2.6, that, if Ω is a bounded set, then:

$$0 < m := \inf_{(x,y)\in\Omega} w(x,y) < M := \sup_{(x,y)\in\Omega} w(x,y) < \infty \tag{4.2}$$

We finally describe a few classical assumptions on $w(x,y)$ which are valid not only for the one given by 2.6, we will assume that:

$$w(x,y) = w(y,x)\ \forall x,y\in\Omega, \quad w(x,x) = 1\ \forall x\in\Omega, \quad \int_\Omega w(x,y)dy = 1\ \forall x\in\Omega \tag{4.3}$$

4.2.2 A small review of $|\nabla_w u|^2(x)$

In this section we will explore slightly the behaviour of the term $|\nabla_w u|^2$ under the classical definition, and also we will explore the behaviour under a redefinition of the term, that will be justified later.

If we consider the classical definition of $|\nabla_w u|$, we have:

$$|\nabla_w u|^2(x) = \int_\Omega (u(y) - u(x))^2 w(x,y)dy$$

then we have that, if $u \in L^2(\Omega)$ (then in $L^1(\Omega)$ due to the boundness of Ω):

$$|\nabla_w u|^2(x) = \int_\Omega (u^2(y) - 2u(x)u(y) + u^2(x))w(x,y)dy \leq M(\|u\|_{L^2} + u^2(x)|\Omega| - 2u(x)\|u\|_{L^1})$$

Then

$$|\nabla_w u|^2 < \infty \ \ a.e.$$

this is due by the boundness of $w(x,y)$ (recall, from measure theory, that a function in $L^1(\Omega)$ with Ω bounded is finite a.e.). This behaviour is 'undesired' in the sense that the finiteness of this term depend only on the finiteness of u, therefore, this 'gradient-like' term does not replicate the undefinition of the classical derivatives when we deal with finite discontinuities (which are our point of study, in the practice).

Moreover, if we define the space:

$$J_w(\Omega) = \{u \in L^2(\Omega) : \ |\nabla_w u|^2 \in L^2(\Omega)\} \tag{4.4}$$

We have that $J_w(\Omega) = L^2(\Omega)$, in fact, let $u \in L^2(\Omega)$:

$$\begin{aligned}
\int_\Omega |\nabla_w u|^2 dx &= \int_\Omega \int_\Omega (u^2(y) - 2u(x)u(y) + u^2(x))w(x,y)dydx \\
&\leq M \int_\Omega \int_\Omega u^2(y) - 2u(x)u(y) + u^2(x)dydx \\
&= M \int_\Omega \|u\|_{L^2}^2 + |\Omega|u^2(x)dx - 2\int_\Omega u(x) \int_\Omega u(y)dydx \\
&\leq M \left(2|\Omega|\|u\|_{L^2}^2 + 2\int_\Omega |u|(x) \int_\Omega |u|(y)dydx \right) \\
&= M \left(2|\Omega|\|u\|_{L^2}^2 + 2\|u\|_{L^1}^2 \right) < \infty
\end{aligned}$$

This is also an 'undesired' behaviour, because the space becomes 'too large', and as we will see later, the functional defined in this way will have a 'bad Γ-limit' in a sense which we will explain later.

Now consider the following definition for the non-local gradient norm:

$$|\nabla_w u|^2(x) = \int_\Omega \frac{|u(y) - u(x)|^2}{|y - x|^\alpha} w(x,y)dy \tag{4.5}$$

where α is a positive fixed number that we will choose later, and $w(x,y)$ satisfy the same assumptions given in 4.2.1, and usually is taken as the non-local denoising filter weight 2.6. Notice that, considering this definition, we 'replicate' a sort of derivative norm plus 'non-local terms', let us see how it behaves with a simple example:

Let $\Omega = (-1,1)$, and $u(x) = H(x) = \mathbb{1}_{[0,\infty)}(x)$, the Heaviside (right-continuous) step function, let us compute the non-local gradient norm defined in this way, for this function in the point $x = 0$ which is a point of discontinuity of u:

$$\begin{aligned}
|\nabla_w u|^2(0) &= \int_{-1}^{1} \frac{(u(y)-1)^2}{|y|^\alpha} w(x,y)dy \\
&= \int_{-1}^{0} \frac{1}{|y|^\alpha} w(x,y)dy \\
&\geq m\int_{-1}^{0} \frac{1}{|y|^\alpha} dy = m\int_{0}^{1} \frac{1}{y^\alpha} dy = \infty \text{ if } \alpha > 1
\end{aligned}$$

therefore, at least if we choose $\alpha > 1$ (and we will choose it for the rest of this section), this gradient-like term is able to recover a classical behaviour of the normal derivative: it becomes undefined when the function is discontinuous. Notice that this behaviour is present (using a similar argument) for any function with isolated finite discontinuities.

Also, notice that if we have: $|\nabla_w u|^2(x) = 0$ then:

$$\frac{(u(y)-u(x))^2}{|y-x|^\alpha} = 0 \ a.e.$$

and we get that $u(y) = u(x)$ a.e. in Ω, therefore, if the non-local gradient is 0 in some point, then the function is constant a.e. in Ω.

So, the question that arises now is what topology we endow to this space, and if we can recognize this space to someone 'known' one, to answer this questions first let us define:

$$|u|_J = \||\nabla_w u|\|_{L^2} = \left(\int_\Omega |\nabla_w u|^2(x)dx\right)^{1/2} = \left(\int_\Omega\int_\Omega \frac{(u(y)-u(x))^2}{|y-x|^\alpha} w(x,y)dydx\right)^{1/2} \tag{4.6}$$

we have the following result about this operator:

Proposition 4.2.1. *Let $|\cdot|_J$ be the operator defined by 4.6, with $w : \Omega\times\Omega \to \mathbb{R}$ a non-negative and symmetric function (i.e. $w(x,y) = w(y,x)$). then $|\cdot|_J$ is a seminorm in $J_w(\Omega)$.*

Proof. To see this first notice that $|\lambda u|_J = |\lambda||u|_J$ is trivial, then the only thing we have to check is the triangle inequality, i.e. we have to prove that:

$$\forall u,v \in J_w(\Omega): \quad |u+v|_J \leq |u|_J + |v|_J \tag{4.7}$$

in order to prove this let us recall, to simplify the notation $\tilde{w}(x,y) = \dfrac{w(x,y)}{|y-x|^\alpha}$, notice that the proof is unreleavant due to the independence of this function from the functions taken in the seminorm. Now let us see that:

$$\begin{aligned}
|u+v|_J &= \||\nabla_w(u+v)|\|_{L^2} = \left(\int_\Omega |\nabla_w(u+v)|^2(x)dx\right)^{1/2} \\
&= \left(\int_\Omega\int_\Omega ((u+v)(y)-(u+v)(x))^2\tilde{w}(x,y)dydx\right)^{1/2} \\
&= \left(\int_\Omega\int_\Omega [(u(y)-u(x))^2+(v(y)-v(x))^2+2(u(y)-u(x))\cdot(v(y)-v(x))]\right. \\
&\quad \left.\tilde{w}(x,y)dydx\right)^{1/2}
\end{aligned}$$

Let us focus on the last term in the inner integral:

$$\begin{aligned}
\int_\Omega (u(y)-u(x))\cdot(v(y)-v(x))\tilde{w}(x,y)dy &= \int_\Omega (u(y)-u(x))\sqrt{\tilde{w}(x,y)}\cdot(v(y)-v(x))\sqrt{\tilde{w}(x,y)}dy \\
&\leq \left(\int_\Omega [(u(y)-u(x))\sqrt{\tilde{w}(x,y)}]^2dy\right)^{1/2}\cdot \\
&\quad \left(\int_\Omega [(v(y)-v(x))\sqrt{\tilde{w}(x,y)}]^2dy\right)^{1/2} \\
&= \left(\int_\Omega (u(y)-u(x))^2\tilde{w}(x,y)dy\right)^{1/2}\cdot \\
&\quad \left(\int_\Omega (v(y)-v(x))^2\tilde{w}(x,y)dy\right)^{1/2} \\
&= |\nabla_w u|\cdot|\nabla_w v|
\end{aligned}$$

The inequality is due to Cauchy-Schwartz inequality.

Now, we can notice that:

$$\begin{aligned}
|u+v|_J^2 &= \||\nabla_w u|\|_{L^2}^2 + \||\nabla_w v|\|_{L^2}^2 + 2\int_\Omega (u(y)-u(x))\cdot(v(y)-v(x))\tilde{w}(x,y)dy \\
&\leq \||\nabla_w u|\|_{L^2}^2 + \||\nabla_w v|\|_{L^2}^2 + 2\||\nabla_w u|\cdot|\nabla_w v|\|_{L^1} \\
&\leq \||\nabla_w u|\|_{L^2}^2 + \||\nabla_w v|\|_{L^2}^2 + 2\||\nabla_w u|\|_{L^2}\cdot\||\nabla_w v|\|_{L^2} = (\||\nabla_w u|\|_{L^2} + \||\nabla_w v|\|_{L^2})^2
\end{aligned}$$

Where the first inequality comes from the previous computation and the second by Hölder inequality.

Finally, taking root we have:

$$|u+v|_J \leq \||\nabla_w u|\|_{L^2} + \||\nabla_w v|\|_{L^2} = |u|_J + |v|_J$$

From which we conclude that $|u|_J$ is a seminorm. □

Thank to the previous result, we can endow $J_w(\Omega)$ the topology generated by the norm:

$$\|u\|_J = \|u\|_{L^2} + |u|_J = \|u\|_{L^2} + \||\nabla_w u|\|_{L^2}$$

Is easy to prove (using the completeness of L^2 and the dominated convergence theorem) that $(J_w(\Omega), \|\cdot\|_J)$ is a Banach space.

Now notice that:

$$m\int_{\Omega\times\Omega} \frac{(u(y)-u(x))^2}{\|y-x\|^\alpha} dydx \leq |u|_J^2 \leq \int_{\Omega\times\Omega} \frac{(u(y)-u(x))^2}{\|y-x\|^\alpha} dydx$$

Then, the norm $\|\cdot\|_J$ is equivalent to the norm of the Fractional Sobolev space $W^{s,p}(\Omega)$ which is given by:

$$\|u\|_{W^{s,p}}^p = \|u\|_{L^p} + \int_{\Omega\times\Omega} \frac{|u(x)-u(y)|^p}{|x-y|^{n+sp}} dydx$$

when the dimension is $n=1$, $p=2$ and $s \in (0,1)$ we get that $\alpha = 1+2s > 1$. Then, in the unidimensional case we have that

$$J_w(\Omega) = W^{s,2}(\Omega) = H^s(\Omega)$$

and our parameter $\alpha > 1$ becomes to $\alpha = 1+2s$, $s \in (0,1)$.

This spaces have a rich structure and known properties, we will try to explore them in order to restrict s (and α) to the maximal interval in which this spaces let us obtain results of interest.

From this little discussion we can see that modifying the non-local gradient norm term we get a 'closer' extension of the classical derivative norm, we will discuss the consequences of how define the term later.

4.3 The Spaces of Work

Here we will describe formally the space $W^{s,p}(\Omega)$ in which the non-local functional is well defined.

Definition 4.3.1. Let Ω be a general, possibly non smooth, open set in $\mathbb{R}^n$. Fix $s \in (0,1)$. For any $p \in [1,\infty)$, we define the fractional Sobolev space $W^{s,p}(\Omega)$ as:

$$W^{s,p}(\Omega) = \left\{ u \in L^p(\Omega) : \frac{|u(x)-u(y)|}{|x-y|^{\frac{n}{p}+s}} \in L^p(\Omega\times\Omega) \right\} \tag{4.8}$$

Notice that this is a space between $L^p(\Omega)$ and $W^{1,p}(\Omega)$, this space is endowed with the norm:

$$\|u\|_{W^{s,p}} = \left(\|u\|_{L^p} + \int_{\Omega\times\Omega} \frac{|u(x)-u(y)|^p}{|x-y|^{n+sp}} dydx \right)^{\frac{1}{p}} \tag{4.9}$$

where the term

$$[u]_{W^{s,p}(\Omega)} := \left(\int_\Omega \int_\Omega \frac{|u(x) - u(y)|^p}{|x-y|^{n+sp}} dxdy \right)^{\frac{1}{p}} \tag{4.10}$$

is the Gagliardo semi-norm of u.

Notice that this space is Banach, also if we take $p = 2$, we use the notation:

$$W^{s,2}(\Omega) = H^s(\Omega)$$

this space is a Hilbert space with the natural inner product.

We also need to define the space of piecewise fractional Sobolev functions in 1 dimension:

Definition 4.3.2. We will say that $u \in PW^{s,p}(a,b)$ if there exists $a = t_0 < t_1 < \ldots < t_N < t_{N+1} = b$ such that $u \in W^{s,p}(t_{i-1}, t_i)$, $\forall i \in \{1, \ldots, N\}$. In this definition S_u is interpreted as the minimal of such sets of points, and $u \in L^2(a,b)$ is defined piecewise on $(a,b) \setminus S_u$.

The choice of $s \in (0,1)$ is not decorative, it can be proved (see [17] Proposition 2) that if $s \geq 1$ and u is a measurable function such that:

$$\int_\Omega \int_\Omega \frac{|u(x) - u(y)|^p}{|x-y|^{n+sp}} dxdy < \infty$$

then u is constant a.e. in Ω, therefore if $s \geq 1$ the space $W^{s,p}(\Omega)$ defined in this way becomes trivial. To generalize Sobolev fractional space for $s > 1$ another definition is needed, this topic can be seen on detail in [29] pp. 9-10.

This spaces, like the integer exponent Sobolev spaces, have interesting and useful properties, which we enounce now starting with the result that $W^{s',p}$ is continuously embedded in $W^{s,p}$ when $s \leq s'$

Proposition 4.3.3. *Let $p \in [1, \infty)$ and $0 < s \leq s' < 1$. Let Ω be an open set in $\mathbb{R}^n$ and $u : \Omega \to \mathbb{R}$ be a measurable function. Then*

$$\|u\|_{W^{s,p}(\Omega)} \leq C \|u\|_{W^{s',p}(\Omega)}$$

for some suitable positive constant $C = C(n, s, p) \geq 1$. In particular,

$$W^{s',p}(\Omega) \subset W^{s,p}(\Omega)$$

Moreover, if Ω is a set of class $C^{0,1}$ with bounded boundary we get that:

$$\|u\|_{W^{s,p}(\Omega)} \leq C \|u\|_{W^{1,p}(\Omega)}$$

In particular,

$$W^{1,p}(\Omega) \subset W^{s,p}(\Omega)$$

The proof of this result can be found in [29] Proposition 2.1.

Now we enounce one of the most useful results for this spaces, this result in particular is needed in order to prove the Γ-convergence and will lead to make a choice about the exponent $s \in (0, 1)$

Theorem 4.3.4. *Let $\Omega \subset \mathbb{R}^n$ be an extension domain for $W^{s,p}$ (i.e. a domain in which we can extend a function u to whole $\mathbb{R}^n$ preserving the regularity, for example a Lipschitz domain) with no external cusps and let $p \in [1, \infty)$, $s \in (0, 1)$ such that $sp > n$. Then, there exists $C = C(n, s, p, \Omega)$ such that*

$$\|f\|_{C^{0,\beta}(\Omega)} \leq C\|f\|_{W^{s,p}(\Omega)}$$

for any $f \in L^p(\Omega)$, with $\beta := s - n/p$

The proof of this result can be found in [29] Theorem 8.2.

The 1 dimensional version of this result is enounced and proved in [43] Corollary 26, as:

Theorem 4.3.5. *Given I an interval of $\mathbb{R}$, assume $s > 1/p$, $(s \in (0, 1),\ p \in (1, \infty])$. Then:*

$$W^{s,p}(I) \subset C^{0,s-1/p}(I)$$

and

$$\|f\|_{C^{0,s-1/p}(I)} \leq \frac{36}{s(s-1/p)}\|f\|_{W^{s,p}(I)} \quad \forall f \in W^{s,p}(I)$$

if I is unbounded 36 may be replaced by 6.

Notice that, in order to use this theorem, we need to consider $s > 1/2$, then, we will consider for the rest of this work:

$$s \in (1/2, 1),\ \ \alpha \in (2, 3)$$

Therefore, in what follows we will consider the non-local gradient functional defined by:

$$|\nabla_w u|^2(x) = \int_\Omega \frac{|u(y) - u(x)|^2}{|y - x|^\alpha} w(x, y)dy = \int_\Omega \frac{|u(y) - u(x)|^2}{|y - x|^{1+2s}} w(x, y)dy \tag{4.11}$$

with $\alpha \in (2, 3)$ or equivalently $s \in (1/2, 1)$.

4.4 Preliminary Results on 'Perimeter Like' Functionals

In order to prove a Γ-convergence result for the functional defined before, we will need two results that arises from the study of the Γ-convergence for ‘Phase-transition Problems’, this is, from functionals which satisfies the following form: If $v \in H^1(a,b)$:

$$L_\varepsilon(v) = \int_\Omega \left(\frac{W(v)}{\varepsilon} + \varepsilon |v'|^2 \right) dt$$

and $L_\varepsilon(v) = +\infty$ if $v \notin H^1(a,b)$.

Where $W : \mathbb{R} \to [0,\infty)$ is a $\mathcal{C}^1$ function such that the set $Z = \{W = 0\}$ is a finite set of points, and:

$$\limsup_{|s|\to\infty} W(s) > 0$$

The idea of this kind of functionals is to approximate the so called ‘Phase-transition energies’ which are of the form:

$$F(v) = \sum_{t\in S(v)} \theta(v^-(t), v^+(t)) \quad v \in PC(a,b),\ v(t) \in Z\ a.e.$$

where: $PC(a,b)$ is the set of piecewise constant functions in (a,b), $v^+(t)$ is the limit from the right of v in t (resp. $v^-(t)$ is the limit from the left of v in t), Z is some fixed set, baptized as the set of phases of v.

In this case, the function $\theta : Z \times Z \to [0,\infty]$ is set to:

$$\theta(w,z) = 2 \left| \int_w^z \sqrt{W(r)} dr \right|$$

The key results which will help us to prove the Γ-convergence of the non-local functional in 1-dimension are:

Lemma 4.4.1. *If $(\varepsilon_j)_j$ is a sequence of positive numbers converging to 0 and such that $\sup_j L_{\varepsilon_j}(v_j) < +\infty$ then there exists a subsequence of $(v_j)_j$ converging in $L^1(a,b)$ to some function $v \in PC(a,b)$ which satisfies $v \in Z$ a.e., moreover, we get that for every $\eta > 0$ there exists a finite set $S = S_\eta$ such that the oscillation of v_j is definitively less than η on each fixed compact subset of $(a,b) \setminus S$.*

And the essential theorem which enounce the Γ-convergence result for the functionals L_ε

Theorem 4.4.2. *Let W and L_ε be defined as above. Then there exists the Γ-limit $\Gamma - \lim_{\varepsilon\to 0^+} L_\varepsilon$ with respect to $L^1(a,b)$ convergence, and it equals the functional F defined on $L^1(a,b)$ by:*

$$F(v) = \begin{cases} \sum_{t\in S(u)} \theta(u^+(t), u^-(t)) & \text{if } u \in PC(a,b) \text{ and } u \in Z \text{ a.e.} \\ +\infty & \text{otherwise} \end{cases} \tag{4.12}$$

where

$$\theta(w,z) = 2 \left| \int_w^z \sqrt{W(r)} dr \right|$$

Notice that if $W(v) = (1-v)^2$ then $Z = \{W = 0\} = \{1\}$, in that case the theorem asserts that

$$F(v) = c_W \cdot |S(v)|$$

where $c_W = 2\int_0^1 \sqrt{W(s)}ds = 2\int_0^1 |1-s|ds = 1.$

The proof of this results can be found in [14] in Lemma 6.2 (and Remark 6.3) and Theorem 6.4.

4.5 Reformulation of the Problem in the Continuous Setting

As we seen before in 2.5 the functional defined by Jung et al. comes from a discrete setting based on non-local operators defined by Gilboa and Osher in [31]. Is very important to see that the definition in this way should be passed to continuous setting with caution, in order to define a functional which have sense in the study of Γ-convergence.

The first important thing to recall, is the definition of the non local derivative: Let $\Omega \subset \mathbb{R}^N$, $x \in \Omega$, $u(x)$ a real function $u : \Omega \to \mathbb{R}$, then the non local derivative in direction y is given by:

$$\partial_y u(x) := \frac{u(y) - u(x)}{d(x,y)}, \quad y, x \in \Omega$$

where $0 < d(x,y) \leq \infty$ is a positive measure defined between points x and y (notice that they assume $x \neq y$, probably because in graphs where it is defined is useless to define this quantity in $x = y$). They define, in order to keep standard notations related to graphs, the weights as:

$$w(x,y) = d^{-2}(x,y)$$

then $0 \leq w(x,y) < \infty$, so we have:

$$\partial_y u(x) := (u(y) - u(x))\sqrt{w(x,y)}, \quad y, x \in \Omega$$

The finiteness of $w(x,y)$ deduced here leads to take this function as the non local weight function defined in 2.4. This choice, for theorical purposes is not adequate in the following sense:

We will prove that, if $w(x,y) \in L^\infty(\Omega \times \Omega)$ then the functional:

$$F_\varepsilon^{NLAT}(u,v) = \alpha \int_\Omega v^2 |\nabla_w u|^2 dx + \beta \int_\Omega (u-u_0)^2 dx + \frac{\nu}{2}\int_\Omega \varepsilon|\nabla v|^2 + \frac{(1-v)^2}{\varepsilon} dx$$

with

$$|\nabla_w u|^2(x) = \int_\Omega (u(y) - u(x))^2 w(x,y)dy$$

Γ-converges (in a topology that we will specify later) to:

$$F(u,v) = \alpha \int_\Omega |\nabla_w u|^2 dx + \beta \int_\Omega (u - u_0)^2 dx$$

i.e. In the Γ-convergence sense we don't measure the set of discontinuities of u, therefore we didn't perform segmentation properly, due to the possibility of 'Fractal' sets with same energy as non-fractal ones.

A brief explanation of this behaviour is based on noticing that the non-local gradient norm, defined in the original way, is always a.e. finite, moreover, as we seen before, it is finite even in the points of discontinuities of u, then, the set function v never needs to vanish, then, v could be equal to 1 in Ω (and not only a.e.), which implies that no segmentation is performed.

Using the properties of Γ-convergence and results about perimeter-like functionals, one can prove the following theorem about the Γ-convergence for this case, we only deal with the 1-dimensional case:

Theorem 4.5.1. *Let $\Omega \subset \mathbb{R}$ be an open interval, and let $F_\varepsilon : L^1(\Omega) \times L^1(\Omega) \to [0, \infty]$ defined by:*

$$F_\varepsilon(u,v) = \begin{cases} \int_\Omega (u-u_0)^2 dx + \int_\Omega v^2 |\nabla_w u|^2 dx + \int_\Omega \left(\varepsilon |\nabla v|^2 + \frac{1}{\varepsilon}(1-v)^2\right) dx \\ \textit{if } (u,v) \in L^2(\Omega) \times H^1(\Omega),\ 0 \le v \le 1 \\ +\infty \quad \textit{otherwise} \end{cases}$$

with $w(x,y) \in L^\infty(\Omega \times \Omega)$, and let $F : L^1(\Omega)^2 \to [0,\infty]$ be defined by

$$F(u,v) = \begin{cases} \int_\Omega (u-u_0)^2 dx + \int_\Omega |\nabla_w u|^2 dx & \textit{if } u \in L^2(\Omega) \textit{ and } v = 1 \textit{ a.e.} \\ +\infty & \textit{otherwise} \end{cases}$$

Then $F_\varepsilon(u,v)$ Γ converges to $F(u,v)$ in the $L^1(\Omega) \times L^1(\Omega)$ topology.

We left the proof pending until the end of this chapter, due to the similarity of the proof from the main theorem of this chapter.

This theorem asserts that, without redefining the 'non-local gradient norm' term the model becomes useless in the sense of Γ-convergence, briefly this term regularizes 'too much', in the sense that the convergence is to a unrelevant functional, because v don't need to 'jump' in order to control the term $|\nabla_w u|$ in the discontinuities of u, therefore, the Γ-limit don't considerate the jump set size, and therefore, no segmentation is performed.

Then, we need to introduce a term which include all the good behaviour of the non-local denoising function but also penalizes discontinuities with unbounded terms, as we seen before this leads to propose the following $|\nabla_w u|^2$ form:

$$|\nabla_w u|^2(x) = \int_\Omega \frac{(u(y) - u(x))^2}{|y - x|^\alpha} w(x,y)dy, \quad w(x,y) \in L^\infty(\Omega \times \Omega),\ \alpha \in (2,3)$$

Usually $w(x,y)$ will be the standard non-local weight function introduced by Buades et al. given by 2.6 (notice that, this implies the assumptions proposed in 4.2.1 for $w(x,y)$) and the norm involved is the usual on each dimension. Notice that considering this term the non-local part of the functional becomes to:

$$\begin{aligned} F^{NL/MS}_{regAT}(u,v) &= \int_\Omega v^2 |\nabla_w u|^2 dx \\ &= \int_\Omega v^2(x) \int_\Omega \frac{(u(y)-u(x))^2}{|y-x|^{1+2s}} w(x,y) dy dx \end{aligned}$$

This functional is defined in $H^s(\Omega) \times H^1(\Omega)$. For technical purposes we will assume that $s \in (1/2, 1)$.

4.6 Γ-Convergence in 1-Dimensional Case

With this tools we can finally enounce the key theorem of this chapter, but first of all, let us define the notation for this section, given $\Omega = (a,b)$ a bounded interval and $s \in (1/2,1)$ fixed, let:

$$G_\varepsilon(u,v) = \begin{cases} \alpha \displaystyle\int_a^b v^2 |\nabla_w u|^2 dt + \frac{\beta}{2} \int_a^b \left(\varepsilon |v'|^2 + \frac{1}{\varepsilon}(1-v)^2 \right) dt & \text{if } (u,v) \in H^s(a,b) \times H^1(a,b) \\ +\infty & \text{otherwise} \end{cases} \tag{4.13}$$

$$G(u,v) = \begin{cases} \alpha \displaystyle\int_a^b |\nabla_w u|^2 dt + \beta |S(u)| & \text{if } u \in PH^s(a,b) \text{ and } v = 1 \text{ a.e.} \\ +\infty & \text{otherwise} \end{cases} \tag{4.14}$$

Theorem 4.6.1. *The functionals $G_\varepsilon : (L^1(\Omega))^2 \to [0,+\infty]$ defined by 4.13 Γ-converge as $\varepsilon \to 0^+$ to the functional $G : L^1(\Omega)^2 \to [0,+\infty]$, defined by 4.14.*

Proof. Given $I \subset (a,b)$ define the functionals depending on set I:

$$G_\varepsilon(u,v;I) = \begin{cases} \alpha \displaystyle\int_I v^2 |\nabla_w u|^2 dt + \frac{\beta}{2} \int_I \left(\varepsilon |v'|^2 + \frac{1}{\varepsilon}(1-v)^2 \right) dt & \text{if } (u,v) \in H^s(I) \times H^1(I) \\ +\infty & \text{otherwise} \end{cases} \tag{4.15}$$

$$G(u,v;I) = \begin{cases} \alpha \displaystyle\int_I |\nabla_w u|^2 dt + \beta |S(u) \cap I| & \text{if } u \in H^s(I) \text{ and } v = 1 \text{ a.e.} \\ +\infty & \text{otherwise} \end{cases} \tag{4.16}$$

Let us see the lim inf inequality, the idea involved (as in the classical Ambrosio-Tortorelli functional) is to study the inequality near 'regular points' (where the non local gradient is well defined) of u and 'discontinuity points' of u (where the non local gradient explodes and therefore v is forced to become 0) separately, for simplicity the proof is given for $\alpha = \beta = 1$:

Let $\varepsilon_j \to 0^+$, $u_j \to u$ and $v_j \to v$ in $L^1(a,b)$. Up to a subsequence (by a classical theorem of measure theory) we can suppose $u_j \to u$ and $v_j \to v$ a.e. and:

$$\lim_j G_{\varepsilon_j}(u_j, v_j) < +\infty \tag{4.17}$$

Because in the other case the Γ-convergence is trivial.

Notice that $v = 1$ a.e., in fact $v_j \to 1$ in $L^2(a,b)$, if not: $\int_\Omega (1-v_j)^2 dt \not\to 0$ and in that case: $G_{\varepsilon_j}(u_j, v_j) \to +\infty$, and the convergence is trivial again.

Now, we can see in the notation of Lemma 4.4.1 that we have

$$\sup_j L_{\varepsilon_j}(v_j) < \infty$$

then, the Lemma 4.4.1 applies with $Z = \{1\}$ and $W(s) = (1-s)^2$ for the sequence $(v_j)_j$, so, we have that there exists a finite set S such that $\forall I \subset\subset (a,b) \setminus S$ we have:

$$\sqrt{\frac{1}{2}} < v_j < 2 - \sqrt{\frac{1}{2}} \text{ on } I$$

In virtue of this we get:

$$\frac{m}{2} \sup_j [u_j]^2_{H^s(I)} \leq \frac{1}{2} \sup_j \int_I |\nabla_w u_j|^2 dt \leq \sup_j \int_I v_j^2 |\nabla_w u_j|^2 dt \leq \sup_j \int_\Omega v_j^2 |\nabla_w u_j|^2 dt < \infty$$

Therefore, recalling that $H^s(I)$ is a Hilbert space, we have that there exists $u \in H^s(I)$ such that $u_j \rightharpoonup u$ in $H^s(I)$. Using that $v_j \to 1$ in $L^2(I)$ we have:

$$\begin{aligned} [u]^2_{H^s(I)} &\leq \liminf_j \int_I \int_I \frac{|u_j(x) - u_j(y)|^2}{|x-y|^{1+2s}} v_j^2(x) dx dy \\ &\leq \frac{1}{m} \liminf_j \int_I v_j^2(x) \int_I \frac{|u_j(x) - u_j(y)|^2}{|x-y|^{1+2s}} w(x,y) dy dx \\ &= \frac{1}{m} \liminf_j \int_I v_j^2(x) |\nabla_w u_j|^2(x) dx \\ &\leq \frac{1}{m} \liminf_j \int_\Omega v_j^2(x) |\nabla_w u_j|^2(x) dx < \infty \end{aligned}$$

Noticing that this estimate is independent of I we get that $u \in PH^s(a,b)$ moreover, thanks to theorem 4.3.5 we have that u is continuous in I, therefore $S_u \subset S$ (then $S \neq \emptyset$). Notice also that (due to the equivalence of norms):

$$\int_I |\nabla_w u|^2 dt \leq \liminf_j \int_I v_j^2 |\nabla_w u_j|^2 dt \leq \liminf_j G_{\varepsilon_j}(u_j, v_j; I) \tag{4.18}$$

Concluding the lim inf inequality for regular points of u.

In the case of the discontinuity points the inequality follows as the same as for the Ambrosio-Tortorelli case:

Let $t \in S_u$, then there exists t_j^1, t_j^2, s_j such that $t_j^1 < s_j < t_j^2$ and:

$$\lim_j t_j^1 = \lim_j t_j^2 = \lim_j s_j = t, \ \lim_j v_j(t_j^1) = \lim_j v_j(t_j^2) = 1 \text{ and } \lim_j v_j(s_j) = 0$$

This is justified by the a.e. convergence of v_j to 1 and to the fact that $t \in S_u$: If not, let us consider $I \subset\subset (a,b)$ such that $t \in I$, and define $L := \liminf_j \inf_{t\in I} v_j^2(t)$. If $L > 0$ (i.e. we can't construct the sequences) we have:

$$m[u_j]^2_{H^s(I)} \leq \int_I |\nabla_w u_j|^2 dt \leq \frac{1}{L}\int_I v_j^2 |\nabla_w u_j|^2 \leq \frac{1}{L}\int_I |\nabla_w u_j|^2 \leq \frac{C}{L}$$

But this implies, as before, that $u \in H^s(I)$ and then again by theorem 4.3.5 we have that u is continuous in I, but that implies that $t \notin S_u$, which is a contradiction.

Now, assuming the existence of the sequences, using the inequality $a^2 + b^2 \geq 2ab$ and the change of variables formula we have:

$$\begin{aligned}
\liminf_j \frac{1}{2}\int_{t_j^1}^{s_j} \left(\frac{1}{\varepsilon}(1-v_j)^2 + \varepsilon|v_j'|^2\right) dt &\geq \liminf_j \int_{t_j^1}^{s_j} (1-v_j)|v_j'| dt \\
&\geq \liminf_j \left|\int_{t_j^1}^{s_j} (1-v_j)v_j' dt\right| \\
&= \liminf_j \left|\int_{v_j(t_j^1)}^{v_j(s_j)} (1-s) ds\right| \\
&= \left|\int_1^0 (1-s)ds\right| = \frac{1}{2}
\end{aligned}$$

and

$$\begin{aligned}
\liminf_j \frac{1}{2}\int_{s_j}^{t_j^2} \left(\frac{1}{\varepsilon}(1-v_j)^2 + \varepsilon|v_j'|^2\right) dt &\geq \liminf_j \int_{s_j}^{t_j^2} (1-v_j)|v_j'| dt \\
&\geq \liminf_j \left|\int_{s_j}^{t_j^2} (1-v_j)v_j' dt\right| \\
&= \liminf_j \left|\int_{v_j(s_j)}^{v_j(t_j^2)} (1-s) ds\right| \\
&= \left|\int_0^1 (1-s)ds\right| = \frac{1}{2}
\end{aligned}$$

Using this argument on each $t \in S_u$ and noticing that 4.18 is valid $\forall I \subset\subset (a,b)$ we conclude the $\liminf$ inequality, to get this define for all $\nu > 0$ the sets:

$$I_\nu := (a,b) \setminus (S_u + [-\nu,\nu]) \text{ and } \tilde{I}_\nu = (S_u + (-\nu,\nu)) \cap (a,b)$$

Then, we get by 4.18:

$$\int_{I_\nu} |\nabla_w u|^2 \leq \liminf_j G_{\varepsilon_j}(u_j, v_j; I_\nu)$$

and, from the previous inequality for discontinuity points applied to each one of them:

$$\left(\frac{1}{2} + \frac{1}{2}\right) \cdot |S_u| \leq \liminf_j G_{\varepsilon_j}(u_j, v_j; \tilde{I}_\nu)$$

then:

$$\begin{aligned} \int_{I_\nu} |\nabla_w u|^2 + |S_u| &\leq \liminf_j G_{\varepsilon_j}(u_j, v_j; I_\nu) + \liminf_j G_{\varepsilon_j}(u_j, v_j; \tilde{I}_\nu) \\ &\leq \liminf_j G_{\varepsilon_j}(u_j, v_j) \end{aligned}$$

Letting $\nu \to 0$ we conclude the $\liminf$ inequality.

For the $\limsup$ inequality, we need to notice first that without loss of generality we can suppose $(a, b) = (-1, 1)$, $u \in PH^s(-1, 1)$ such that $S_u = \{0\}$ (the finiteness of S_u is needed, if not the inequality is trivial), then 'the recovery sequence' can be constructed as follows:

The idea for u_ε is just to build it as a regularization of u around the point of discontinuity, in order to do it is sufficient to build $u_\varepsilon \in H^1(-1, 1)$ such that $u_\varepsilon(t) = u(t)$ if $|t| > \varepsilon^2$ and if $|t| \leq \varepsilon^2$ we consider the regularized function (we regularize as usual by convolving with a regularization Kernel).

The idea for v_ε is to build it such that it is 1 'away from the discontinuity', 0 in the discontinuity, regularized in order to get a well defined functional, and defined carefully in order to have a controlled 'length-term', the construction follows as: Let $T > 0$, and fix $\nu > 0$, consider $v \in H^1(0, T)$ such that:

$$\int_0^T ((1 - v)^2 + |v'|^2) dt \leq 1 + \nu$$

and $v(0) = 0$, $v(T) = 1$; now set:

$$v_\varepsilon(t) = \begin{cases} 0 & \text{if } |t| \leq \varepsilon^2 \\ v\left(\dfrac{|t| - \varepsilon^2}{\varepsilon}\right) & \text{if } \varepsilon^2 < |t| < \varepsilon^2 + \varepsilon T \\ 1 & \text{if } |t| \geq \varepsilon^2 + \varepsilon T \end{cases}$$

Notice that, from this definition naturally $v_\varepsilon \to 1$ a.e., also we have:

$$\begin{aligned}
G_\varepsilon(u_\varepsilon, v_\varepsilon) &= \int_{-1}^{1} v_\varepsilon^2|\nabla_w u_\varepsilon|^2 + \frac{\varepsilon}{2}v_\varepsilon'^2 + \frac{1}{2\varepsilon}(1-v_\varepsilon)^2 dt \\
&= \int_{-1}^{-\varepsilon^2} v_\varepsilon^2|\nabla_w u|^2 dt + \int_{\varepsilon^2}^{1} v_\varepsilon^2|\nabla_w u|^2 dt + \int_{-\varepsilon^2}^{\varepsilon^2} v_\varepsilon^2|\nabla_w u_\varepsilon|^2 dt \\
&\quad + \int_{-1}^{1} \frac{\varepsilon}{2}v_\varepsilon'^2 + \frac{1}{2\varepsilon}(1-v_\varepsilon)^2 dt \\
&\leq \int_{-1}^{-\varepsilon^2} |\nabla_w u|^2 dt + \int_{\varepsilon^2}^{1} |\nabla_w u|^2 dt + \int_{-\varepsilon^2}^{\varepsilon^2} |\nabla_w u_\varepsilon|^2 dt \\
&\quad + \int_{-1}^{1} \frac{\varepsilon}{2}v_\varepsilon'^2 + \frac{1}{2\varepsilon}(1-v_\varepsilon)^2 dt
\end{aligned}$$

In order to bound the terms of v_ε first notice that:

- The term $(1-v_\varepsilon)^2$ is 0 if $|t| \geq \varepsilon^2 + \varepsilon T$, is 1 if $|t| \leq \varepsilon^2$ and ranges values in $(0,1)$ if $\varepsilon^2 < |t| < \varepsilon^2 + \varepsilon T$.
- The term $|v_\varepsilon'|^2$ is 0 if $|t| < \varepsilon^2$ or $|t| > \varepsilon^2 + \varepsilon T$ otherwise is not 0 due to the regularity of v_ε.

From this, we can deduce that:

$$\begin{aligned}
\int_{-1}^{1} \frac{\varepsilon}{2}v_\varepsilon'^2 + \frac{1}{2\varepsilon}(1-v_\varepsilon)^2 dt &= 2\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T} \frac{\varepsilon}{2}v_\varepsilon'^2 dt + \frac{1}{2\varepsilon}\left(2\int_0^{\varepsilon^2} 1dt + 2\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T}(1-v_\varepsilon)^2 dt\right) \\
&= \varepsilon\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T} v_\varepsilon'^2 dt + \varepsilon + \frac{1}{\varepsilon}\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T}(1-v_\varepsilon)^2 dt
\end{aligned}$$

Notice that

$$v_\varepsilon'(t) = \frac{d}{dt}\left(v\left(\frac{t-\varepsilon^2}{\varepsilon}\right)\right) = \frac{1}{\varepsilon}v\left(\frac{t-\varepsilon^2}{\varepsilon}\right),\ t \in (0,1)$$

Then

$$\varepsilon\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T} v_\varepsilon'^2 dt + \frac{1}{\varepsilon}\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T}(1-v_\varepsilon)^2 dt = \int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T}\frac{1}{\varepsilon}v'^2\left(\frac{t-\varepsilon^2}{\varepsilon}\right)dt + \frac{1}{\varepsilon}\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T}\left(1-v\left(\frac{t-\varepsilon^2}{\varepsilon}\right)\right)^2 dt$$

By change of variables formula, we get:

$$\begin{aligned}
\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T}\frac{1}{\varepsilon}v'^2\left(\frac{t-\varepsilon^2}{\varepsilon}\right)dt &= \int_0^T v'^2(u)du \\
\frac{1}{\varepsilon}\int_{\varepsilon^2}^{\varepsilon^2+\varepsilon T}\left(1-v\left(\frac{t-\varepsilon^2}{\varepsilon}\right)\right)^2 dt &= \int_0^T (1-v(u))^2 du
\end{aligned}$$

Then

$$\int_{-1}^{1} \frac{\varepsilon}{2}v_\varepsilon'^2 + \frac{1}{2\varepsilon}(1-v_\varepsilon)^2 dt = \varepsilon + \int_0^T v'^2(u)du + \int_0^T (1-v(u))^2 du \leq \varepsilon + 1 + \nu$$

Therefore:

$$\begin{aligned}\limsup_{\varepsilon\to 0} G_\varepsilon(u_\varepsilon, v_\varepsilon) &\leq \limsup_{\varepsilon\to 0} \int_{-1}^{-\varepsilon^2} |\nabla_w u|^2 dt + \int_{\varepsilon^2}^{1} |\nabla_w u|^2 dt + \int_{-\varepsilon^2}^{\varepsilon^2} |\nabla_w u_\varepsilon|^2 dt + \varepsilon + 1 + \nu \\ &= \int_{-1}^{1} |\nabla_w u|^2 dt + 1 + \nu\end{aligned}$$

Where the last equality is just an application of dominated convergence theorem. Since this is valid for all $\nu > 0$ we conclude that:

$$\limsup_{\varepsilon\to 0} G_\varepsilon(u_\varepsilon, v_\varepsilon) \leq \int_{-1}^{1} |\nabla_w u|^2 dt + 1 = G(u,v)$$

Which is the lim sup inequality in this case.

With this two inequalities we conclude the desired Γ-convergence, and therefore we conclude the proof of the theorem. □

To conclude this chapter we prove the pending Theorem 4.5.1, using most of the same arguments as in the previous proof, recall that this theorem asserts that if we consider the original non-local gradient term then the Γ-convergence becomes trivial, in the sense that no discontinuities of u have been measured.

Proof of Theorem 4.5.1.

By the continuity of the fidelity term on $L^2(\Omega)$ (therefore on $L^1(\Omega)$) we just have to focus on the two later terms of the functional.

Following the same prodecure as the previous proof we get that, up to subsequence, $u_j \to u$, $v_j \to 1$ a.e. and in $L^2(\Omega)$, moreover, up to subsequence (again) we have that (v_j) is in $L^\infty(\Omega)$ and:

$$\lim_j F_{\varepsilon_j}(u_j, v_j) < \infty$$

As we have done before, thank to Lemma 4.4.1 we have the existence of a finite set S such that $\forall I \subset\subset (a,b) \setminus S$ we have

$$\sqrt{\frac{1}{2}} < v_j < 2 - \sqrt{\frac{1}{2}} \text{ on } I$$

The idea is to prove that S_u is not contained in S.

In virtue of this we get:

$$\begin{aligned}0 \leq \frac{m}{2} \sup_j \int_I \int_I (u_j(y) - u_j(x))^2 dy dx &= m \sup_j \left(|I| \|u_j\|^2_{L^2(I)} - \int_I u_j(x)dx \int_I u_j(y) dy \right) \\ &\leq \frac{1}{2} \sup_j \int_I |\nabla_w u_j|^2 dt \\ &\leq \sup_j \int_I v_j^2 |\nabla_w u_j|^2 dt \\ &\leq \sup_j \int_\Omega v_j^2 |\nabla_w u_j|^2 dt < \infty\end{aligned}$$

Therefore (noticing that the sequence u_j is in $L^1(\Omega)$), we have that there exists $u \in L^2(I)$ such that $u_j \rightharpoonup u$ in $L^2(I)$. Using that $v_j \to 1$ in $L^2(I)$ and dominated convergence theorem, we have:

$$\int_I |\nabla_w u|^2 dt \leq \liminf_j \int_I v_j^2 |\nabla_w u_j|^2 dt \leq \liminf_j \int_\Omega v_j^2 |\nabla_w u_j|^2 dt$$

Notice that in this case we can't conclude anything about the continuity of u in I, this just let us to deduce that:

$$\int_I |\nabla_w u|^2 dt \leq \liminf_j \int_I v_j^2 |\nabla_w u_j|^2 dt \leq \liminf_j F_{\varepsilon_j}(u_j, v_j; I) \tag{4.19}$$

Concluding the lim inf inequality for points of u outside the finite set S constructed by the lemma 4.4.1.

Since the last steps don't imply continuity of u in the interval I, this procedure is valid for all points on $(a, b) \setminus S$, even for ones in S_u.

Applying the same limit procedure as before we conclude the lim inf inequality.

The lim sup inequality is easier because we can take $v_\varepsilon \equiv 1$ and $u_\varepsilon \equiv u$, then the inequality becomes trivial. This concludes the desired proof. □

Chapter 5

Gabor Functions and Additional Texture Features

In this part we will try to improve the detection of textures by considering the so called **Gabor functions** additionally to just considering the non local segmentation of the image.

This extra step is motivated by the signal phase analysis that could be done easily when the signal is a linear combination of Gabor functions, this analysis can reveal the existence of some special kind of texture, the one based on putting togheter the same image structure, but displaced in some phase. It also adds a more realistic setting, because the Gabor functions are considered a good model for the way that the human vision process an image.

We will first detail what specifically what is a Gabor function, then we will explain our specific work: approximate a signal by 'Gabor-like' functions, and finally we will describe the process performed to include this new feature in our task, including some numerical examples in one dimension.

5.1 Gabor Functions - Mathematical Definition

The Gabor function, also known as Gabor atom, was proposed by Denis Gabor in 1946, the idea is to build a family of functions from translations and modulations of a generating function.

So, given a function $g \in L^2(\mathbb{R})$, and constants $a, b \in \mathbb{R}$ we consider the family, called Gabor Functions generated by the function g by:

$$g_{l,n}(x) = g(x - al)e^{2\pi ibnx}, \quad -\infty < l, n < \infty$$

Is important to notice that this family of functions could be a basis for $L^2(\mathbb{R})$ (but is not required) depending of the parameters g, a, b.

Unlike Fourier Series, this functions are not necessarily a basis for some function space, and also, the functions generated by g may have a compact support, which is highly realistic for image-signal representation purposes.

5.2 Approximating a Signal by Gabor Functions

Given a signal $u \in L^{\infty}(\mathbb{R})$ (with compact support, in fact), we would like to approximate it by the family of Gabor Functions defined above, in some norm. So, we would like to write:

$$u(x) \sim \sum_{l,n \in \mathbb{N}} c_{l,n} \cdot g_{l,n}(x) =: G_c(x)$$

where the coefficients $c_{l,n}$ are such that the distance in some norm between u and G is minimized, i.e. we choose the sequence of coefficients $\overline{c} = (\overline{c}_{l,n})$ such that:

$$||u - G_{\overline{c}}|| = \min_{c} ||u - G_c||$$

In a discretized setting (the one in we work actually) we will have the discretized domain: $(x_i)_{i=1}^{N}$ and the corresponding values of the signal: $u_i = u(x_i),\ i \in \{1, \ldots, N\}$, so, the problem becomes a distance between a finite number of points (instead of distances between functions in a infinite dimensional space), and the problem is now a finite dimensional one. For example, if we want the approximation on norm 2 (known as least squares approximation) we have:

$$||u - G_{\overline{c}}|| = \sum_{i=1}^{N} (u_i - G_{\overline{c}}(x_i))^2 = \min_{c} \sum_{i=1}^{N} (u_i - G_c(x_i))^2$$

Moreover, if we have a finite family of functions, that will be our usual setting, due to the compact support of the signal, we have a simpler version of the problem, because we have:

$$G_c(x) = \sum_{i=1}^{l} c_i \cdot g_i(x)$$

where g_i is indexed in some compatible way to order the subscripts $l \in \{1, \ldots, m_1\},\ n \in \{1, \ldots, m_2\}$.

With this notation, the problem can be rewritten in a explicit way as:

$$\min_{c=(c_i)} ||u - Ac||$$

where A is a matrix of size (Number of functions of Gabor function family × N) were each row have the structure:

$$A_i = (c_1 g_1(x_i), c_2 g_2(x_i), \ldots, c_M g_M(x_i))$$

where M is the number of functions in the Gabor function family.

This is the general setting for function approximation in our context, in our case, we will have a large number of functions (much larger than the number of samples from our signal), so, the approximation scheme should be taken carefully, because an unsmart choice of parameters and methods could lead to a computationaly expensive process. This specific problem is known as a basis pursuit denoising and we will discuss it in the next section.

5.3 Basis Pursuit Denoising

In this special scheme of approximation we have a large number of basis functions, the goal is to find a good fit of the given signal as a linear combination of a **small** number of the basis functions. This is called basis pursuit since we are selecting a much smaller basis to model the data.

So, if we call $\mathcal{F}$ the set of linear combinations of Gabor function family functions (i.e. $\mathcal{F} = span(\{g_i\})$), we want to find a function $f \in \mathcal{F}$ such that:

$$f \text{ fits the data well, i.e. } f(x_i) = u_i$$

and

$$f = \sum_{i \in B} c_i \cdot g_i, \quad \text{with } B = \{i \mid c_i \neq 0\}$$

B is the set of indices of the chosen basis elements, as a sparse description of the data.

This representation is useful in our case because with appropiate Gabor function associated we can study the predominant phases of a given signal, which could be very useful in order to detect some texture structure in a signal, additionaly with the non local segmentation.

In order to ensure that the number of non-zero coefficients is small (this is usually known as find a sparse description of the signal) we will have to perform a pre-approximation step, so, we first solve the following problem:

$$\min_c \sum_{i=1}^{N} (f(x_i) - u_i)^2 + \gamma ||c||_1 \Leftrightarrow \min_c \sum_{i=1}^{N} \left(\sum_{j=1}^{|\mathcal{F}|} c_j g_j - u_i \right)^2 + \gamma \|c\|_1 \tag{5.1}$$

where $\gamma > 0$ is a parameter used to trade off the quality of the fit to the data, and the sparsity of the coefficients. We can use the solution of this problem as a solution of the original problem, or perform a new minimization (a refinement of the first minimization), in order to ensure that we choose the best fit approximation. So, if $\hat{c}$ is the coefficient vector solution of the first minimization problem, then we set $\hat{B} = \{i \mid \hat{c}_i \neq 0\}$ and then we solve the least-squares problem:

$$\min \sum_{i=1}^{N} (f(x_i) - u_i)^2 \tag{5.2}$$

with variables c_i, $i \in \hat{B}$, and $c_i = 0$ if $i \notin \hat{B}$. A more detailed treatment of this kind of problems can be found in [12] in Chapter 6 and in the article [24].

5.4 Setting up our Problem, Exponential Splines

In our case, we will work with the following family of functions:

$$\phi_{k,p}(x) = \cos(\pi/2 \cdot (x-p)) \sin(\pi/2 \cdot k \cdot (x-p)) \cdot \mathbf{1}(|x-p| \leq 1/k)$$

$$\psi_{k,p}(x) = \cos(\pi/2 \cdot (x-p)) \cos(\pi/2 \cdot k \cdot (x-p)) \cdot \mathbf{1}(|x-p| \leq 1/k)$$

with $k = 2^2, 2^3, \ldots, 2^8$, $p = p(k) = 0, 1/k, \ldots, (k-1)/k$

This functions are the real and imaginary part of the complex Gabor function family:

$$\Psi_{k,p}(x) = \cos(\pi/2 \cdot (x-p)) e^{i\pi/2 \cdot k \cdot (x-p)} \cdot \mathbf{1}(|x-p| \leq 1/k) \tag{5.3}$$

which is a particular case of the so-called exponential splines. The choice of this functions is based on the good properties of this ones, for example: $\phi'_{k,p}(x) = C \cdot \psi_{k,p}(x)$. Moreover, this function family could be understood as a Gabor function family, so, it could be considered as a good modelling for human vision (i.e. how our eyes process-approximate the signals from the exterior).

The different values for k means that we allow to have different levels of resolution (higher the k, higher level of resolution for the approximations), this is a very important setting, because with this we are able to distinguish what level of resolution is predominant on the signal, and moreover, distinguish if some elements on the signal are on a different level of resolution, for example, some particular texture which can be considered as noise by usual methods. A more detailed discussion of the exponential splines can be found on the following references: [8], [28].

Chapter 6

Numerical Implementation

In this chapter we describe how to implement the diverse tasks that we focused in this work. We start implementing the Non Local denoising filter proposed by Buades that we studied in chapter 2 in order to study if this denoising filter does in fact work better with textures than the classical filters, after that we focus our attention in implementations for Segmentation, local and non-local in order to study how this tasks perform in textured signals and images. We describe in a separate section how to implement the construction of weight function $w(x, y)$ involved in computations for non local methods, showing two algorithms which makes the construction of this function, the most computationally demanding, faster than just considering a complete computations. Finally we study how to implement the approximation of signals by a family of specific Gabor functions, the so called (Complex Exponential)-Trigonometric Splines, in order to study how this approximations reveals more information about the texture on example signals.

6.1 Non Local Denoising Filter

As we reviewed on Chapter 2, the Non Local denoising filter is based on the equation 2.2, in order to perform a numerical implementation we need to consider a discrete version of this filter.

Then, in the discrete case we have the following formulation:

Consider a discrete noisy image $u_0 = \{u_0(i)/i \in \Omega\}$, in this context, the estimated value $NL(u_0)(i)$ is computed as a weighted average of all the pixels in the image, i.e. considering:

$$NL(u_0)(i) = u(i) = \frac{1}{C(i)} \sum_{j \in \Omega} w(i, j) u_0(j) \tag{6.1}$$

where $w(i, j)$ must be a sort of discretization of the continuous formula described before, it intends to measure the similarity between pixels i and j; in a general context we have to consider $w(i, j)$ such that:

$$\sum_{j \in \Omega} w(i, j) = 1 \quad 0 \leq w(i, j) \leq 1$$

The (sort of) discretization of the weight function is:

$$w(i,j) = \frac{1}{Z(i)} e^{-\frac{||u_0(\mathcal{N}_i)-u_0(\mathcal{N}_j)||^2_{2,a}}{h^2}}$$

where $Z(i)$ is a normalization factor, $||\cdot||_{2,a}$denotes the Euclidean weighted distance (by a Gaussian kernel of standard deviation a) and $\mathcal{N}_i$ denotes a "neighborhood" of the pixel i, centered at it, usually this neighborhood is a square of length $2m$, then $u_0(\mathcal{N}_i)$ is a vector, so, the idea of this discretization is to compare the patch centered on i between the one centered on j.

The important thing here is the weight function $w(x,y)$ this contains the formula which perform the "averaging" term. This function is considered in other context to perform segmentation (using it in the so called non-local gradient or weighted gradient as we seen before). Then, is important to follow the behaviour of this function in the task of denoising, because we have to check that, if we consider a new weight function (we may need to modify the original NL-means weight function for theorical or computational purposes), in fact still have the main characteristic we want: preservation of textures in denoising procedure. Then, is important to have an implementation of this algorithm, in order to have a checkpoint for next steps of this work.

In the case of one dimensional denoising a 'primitive' implementation can be constructed just by implementing the formula 6.1, but if we want to perform the denoising with a 2D images we need to try to reduce the number of calculations, as we seen before, the original filter involves a high number of calculations (at least $2N^2$ for each pixel if N is the number of pixels of the image). In order to do that, we will follow the "faster" version of NL-means proposed by Buades in [18] Section 3.7, which try to reduce calculations:

Let I a grid of pixels, choose a subset $\{i_1,\ldots,i_k\} \subset I$. Consider $B = \{i \ / \ ||i|| \leq m\}$, and then define: $W_k = i_k + B$. The idea is to divide I in non-disconnected regions such that: $I = \cup_i W_i$ and $W_i \cap W_{i+1} \neq \emptyset$. The idea is to define NL-means for the W_k objects (the so called vectorial NL-means) and then defining the NL-means for a fixed pixel as the average of the vectorial NL-means where this pixel belongs.

Let us define the vectorial NL-means, for each W_k as:

$$NL(W_k) = \frac{1}{C_k}\sum_{j\in I} u_0(W_j) e^{-\frac{||u_0(W_k)-u_0(W_j)||^2_2}{h^2}}$$

Where C_k is a normalization parameter.

Notice that in this case, the norm involved is the usual, since we restore at the same time the whole neighborhood and do not want to give any privilege to any point in particular.

Finally, in order to restore the value at a pixel i, we must consider all W_k containing

i, so, if we define: $A_i = \{k \ / \ i \in W_k\}$, we have to define:

$$NL(u_0)(i) = u(i) = \frac{1}{|A_i|} \sum_{k \in A_i} NL(W_k)(i) \tag{6.2}$$

Which is the final formula that we will consider in order to perform non local denoising in 2D setting. Examples are given in 1 and 2 Dimensions in the next chapter.

6.2 Segmentation using Ambrosio-Tortorelli Approximation

Let us consider the following generalized version of Ambrosio-Tortorelli functional with constants α, β in order to penalize more/less the regularization or the boundary search:

$$F_\varepsilon^{AT}(u,v) = \alpha \int_\Omega v^2 |\nabla u|^2 dx + \beta \int_\Omega (u-u_0)^2 dx + \frac{\nu}{2} \int_\Omega \varepsilon |\nabla v|^2 + \frac{(1-v)^2}{\varepsilon} dx \tag{6.3}$$

In order to minimize this functional we will try to search for local minima based in a fixed point iteration method, this method is based on solving numerically the first order conditions (Euler-Lagrange equations, which will lead to solve two vector equations) for the functional in an alternate scheme, which have the following basic idea:

1. Consider initial conditions (given by some usual condition, which we will explain later) u^0 and v^0
2. While the relative error of iterations (i.e. the relative error between the new iteration and the old one) is bigger than a tolerance, do the following:

   Consider (u_n, v_n) the values of u and v in the n-th iteration:

   (a) Solve the equation related to the first order condition in u, considering $v = v_n$ fixed. Call the solution $\hat{u}$.

   (b) Solve the equation related to the first order condition in v, considering $u = u_n$ fixed. Call the solution $\hat{v}$.

   (c) Assign: $u_{n+1} = \hat{u}$ and $v_{n+1} = \hat{v}$
3. Finally, return $(\hat{u}, \hat{v})$.

Notice that any numerical method to minimize this functional have a high dependence of the initial conditions, specially in our case, because F_ε^{AT} is not convex in (u, v) and the related Euler-Lagrange equations have in general multiple solutions. Then, the choice of initial conditions could be essential, we will detail later the choice of them. In order to define precisely the numerical algorithm we need to compute the Euler-Lagrange equations for this functional, which we are going to deduce in the following section.

6.2.1 Euler-Lagrange Equations

The idea is to compute the first order conditions for this functional, we will understand the 'first order condition' as the equations derived from imposing the Gateaux derivative for 6.3 on u and v as 0. Recall that this is a necessary condition to be a local minima.

We resume this equations in the following theorem:

Theorem 6.2.1. *The First order conditions (or Euler-Lagrange equations) for the functional defined by 6.3 with $(u, v) \in H^1(\Omega)^2$ are:*

$$div(v^2\nabla u) - \frac{\beta}{\alpha}(u - u_0) = 0 \text{ in } \Omega, \tag{6.4}$$

$$\Delta v + \frac{(1 - v)}{\varepsilon^2} - \frac{2\alpha \cdot v|\nabla u|^2}{\nu\varepsilon} = 0 \text{ in } \Omega. \tag{6.5}$$

Proof. Recall that the Gateaux derivative of a functional $J(u, v)$ at u in the direction φ is:

$$D_u J(u, v)[\varphi] = \lim_{t\to 0} \frac{J(u + t\varphi, v) - J(u, v)}{t}$$

analogously we can define $D_v J(u, v)[\varphi]$. So, in the case of $F_\varepsilon^{AT}(u, v)$ we have to compute:

$$D_u F_\varepsilon^{AT}(u, v)[\varphi] = \lim_{t\to 0} \frac{F_\varepsilon^{AT}(u + t\varphi, v) - F_\varepsilon^{AT}(u, v)}{t}$$

and

$$D_v F_\varepsilon^{AT}(u, v)[\varphi] = \lim_{t\to 0} \frac{F_\varepsilon^{AT}(u, v + t\varphi) - F_\varepsilon^{AT}(u, v)}{t}$$

So, let us compute in the first case:

$$\begin{aligned} F_\varepsilon^{AT}(u + t\varphi, v) - F_\varepsilon^{AT}(u, v) &= \alpha \int_\Omega (v^2|\nabla(u + t\varphi)|^2 - v^2|\nabla u|^2)dx \\ &\quad + \beta \int_\Omega ((u + t\varphi) - u_0)^2 - (u - u_0)^2 dx \\ \text{Notice that:} & \\ |\nabla(u + t\varphi)|^2 - |\nabla u|^2 &= |\nabla u|^2 + 2t\nabla u\nabla\varphi + \varepsilon^2 + t^2|\nabla\varphi|^2 - |\nabla u|^2 \\ &= 2t\nabla u\nabla\varphi + t^2|\nabla\varphi|^2 \\ \text{Also notice that:} & \\ ((u + t\varphi) - u_0)^2 - (u - u_0)^2 &= (u + t\varphi)^2 - 2(u + t\varphi)u_0 + u_0^2 - u^2 + 2uu_0 - u_0^2 \\ &= u^2 + 2tu\varphi + t^2\varphi^2 - 2uu_0 - 2t\varphi u_0 + u_0^2 - u^2 + 2uu_0 - u_0^2 \\ &= 2t(u - u_0)\varphi + t^2\varphi^2 \\ \text{Then we get:} & \\ F_\varepsilon^{AT}(u + t\varphi, v) - F_\varepsilon^{AT}(u, v) &= \alpha \int_\Omega v^2 \cdot (2t\nabla u\nabla\varphi + t^2|\nabla\varphi|^2)dx + \beta \int_\Omega 2t(u - u_0)\varphi + t^2\varphi^2 dx \\ \text{Therefore:} & \\ \frac{F_\varepsilon^{AT}(u + t\varphi, v) - F_\varepsilon^{AT}(u, v)}{t} &= \alpha \int_\Omega v^2 \cdot (2\nabla u\nabla\varphi + t|\nabla\varphi|^2)dx + \beta \int_\Omega 2(u - u_0)\varphi + t\varphi^2 dx \end{aligned}$$

Then, by a standard application of Dominated Convergence Theorem we have:

$$\lim_{t\to 0}\frac{F_\varepsilon^{AT}(u+t\varphi,v)-F_\varepsilon^{AT}(u,v)}{t} = \alpha\int_\Omega 2v^2\cdot\nabla u\nabla\varphi dx+\beta\int_\Omega 2(u-u_0)\varphi dx$$

But by Green Identity and noticing that $\varphi\in C_0^\infty$ in $\partial\Omega$

$$\begin{aligned}\lim_{t\to 0}\frac{F_\varepsilon^{AT}(u+t\varphi,v)-F_\varepsilon^{AT}(u,v)}{t} &= \alpha\int_\Omega -2div(v^2\cdot\nabla u)\cdot\varphi dx+\int_{\partial\Omega}2\alpha v^2\partial_n u\cdot\varphi dS\\ &+ \beta\int_\Omega 2(u-u_0)\varphi dx\\ &= \int_\Omega(-2\alpha div(v^2\cdot\nabla u)+2\beta(u-u_0))\varphi dx\end{aligned}$$

Then, because we are imposing the first order condition, we have:

$$D_uF_\varepsilon^{AT}(u,v)[\varphi]=0\ \ \forall\varphi\in\mathcal{C}_0^\infty(\Omega)$$

Noticing that this is valid for all $\varphi\in\mathcal{C}_0^\infty(\Omega)$, we can conclude using 2.3.3 that:

$$2\beta(u-u_0)-2\alpha div(v^2\cdot\nabla u)=0\text{ in }\Omega$$

which can be rewritten as:

$$div(v^2\nabla u)-\frac{\beta}{\alpha}(u-u_0)=0\text{ in }\Omega$$

Now, for $D_vF_\varepsilon^{AT}$ let us compute:

$$\begin{aligned}F_\varepsilon^{AT}(u,v+t\varphi)-F_\varepsilon^{AT}(u,v) &= \frac{\nu}{2}\int_\Omega\varepsilon(|\nabla(v+t\varphi)|^2-|\nabla v|^2)+\frac{(1-v-t\varphi)^2-(1-v)^2}{\varepsilon}dx\\ &+ \alpha\int_\Omega((v+t\varphi)^2-v^2)|\nabla u|^2dx\end{aligned}$$

Notice that:

$$\begin{aligned}(v+t\varphi)^2-v^2 &= 2vt\varphi+t^2\varphi^2\\ |\nabla(v+t\varphi)|^2-|\nabla v|^2 &= |\nabla v|^2+2t\nabla v\cdot\nabla\varphi+t^2|\nabla\varphi|^2-|\nabla v|^2\\ &= 2t\nabla v\cdot\nabla\varphi+t^2|\nabla\varphi|^2\\ (1-v-t\varphi)^2-(1-v)^2 &= -2(1-v)t\varphi+t^2\varphi^2\end{aligned}$$

Then we get:

$$\begin{aligned}F_\varepsilon^{AT}(u,v+t\varphi)-F_\varepsilon^{AT}(u,v) &= \alpha\int_\Omega(2vt\varphi+t^2\varphi^2)|\nabla u|^2dx+\frac{\nu}{2}\int_\Omega\varepsilon(2t\nabla v\cdot\nabla\varphi+t^2|\nabla\varphi|^2)dx\\ &- \frac{\nu}{2}\int_\Omega\frac{2(1-v)t\varphi+t^2\varphi^2}{\varepsilon}dx\end{aligned}$$

Therefore:

$$\begin{aligned}\frac{F_\varepsilon^{AT}(u,v+t\varphi)-F_\varepsilon^{AT}(u,v)}{t} &= \alpha\int_\Omega(2v\varphi+t\varphi^2)|\nabla u|^2dx+\frac{\nu}{2}\int_\Omega\varepsilon(2\nabla v\cdot\nabla\varphi+t|\nabla\varphi|^2)dx\\ &- \frac{\nu}{2}\int_\Omega\frac{2(1-v)\varphi+t\varphi^2}{\varepsilon}dx\end{aligned}$$

Using again the Dominated Convergence Theorem, we get that:

$$\begin{aligned}\lim_{t\to0}\frac{F_\varepsilon^{AT}(u,v+t\varphi)-F_\varepsilon^{AT}(u,v)}{t} &= \alpha\int_\Omega 2v|\nabla u|^2\varphi dx+\nu\int_\Omega\varepsilon\cdot\nabla v\cdot\nabla\varphi dx\\ &- \nu\int_\Omega\frac{(1-v)\varphi}{\varepsilon}dx\end{aligned}$$

Using Green Identity again and noticing that $\varphi\in C_0^\infty,$ in $\partial\Omega$:

$$\begin{aligned}\lim_{t\to0}\frac{F_\varepsilon^{AT}(u,v+t\varphi)-F_\varepsilon^{AT}(u,v)}{t} &= \alpha\int_\Omega 2v|\nabla u|^2\varphi dx-\nu\int_\Omega\varepsilon\cdot\Delta v\cdot\varphi dx\\ &+ \nu\int_{\partial\Omega}\varepsilon\cdot\varphi\cdot\partial_n v dS-\nu\int_\Omega\frac{(1-v)\varphi}{\varepsilon}dx\\ &= \int_\Omega 2\alpha v|\nabla u|^2\varphi dx-\nu\varepsilon\cdot\Delta v\cdot\varphi dx\\ &+ \int_\Omega -\nu\frac{(1-v)\varphi}{\varepsilon}dx\end{aligned}$$

Then, because we are imposing the first order condition, we have:

$$D_vF_\varepsilon^{AT}(u,v)[\varphi]=0\quad\forall\varphi\in\mathcal{C}_0^\infty(\Omega)$$

we can conclude using again 2.3.3 that:

$$2\alpha v|\nabla u|^2-\nu\varepsilon\Delta v-\nu\frac{(1-v)}{\varepsilon}=0\text{ in }\Omega$$

which can be rewritten as:

$$\Delta v+\frac{(1-v)}{\varepsilon^2}-\frac{2\alpha\cdot v|\nabla u|^2}{\nu\varepsilon}=0\text{ in }\Omega$$

□

6.2.2 Implementation

In order to implement a numerical method to solve the equations, we need to discretize them. Considering a classical finite differences derivatives [1] we are able to reduce each equation to a linear system. Considering the discretizations (with unit step and proper extension to the boundaries i.e. reflecting the image or signal):

$$\frac{\partial u}{\partial x}(i,j)=\frac{u_{i+1,j}-u_{i-1,j}}{2}\quad\frac{\partial u}{\partial y}(i,j)=\frac{u_{i,j+1}-u_{i,j-1}}{2}$$

$$\frac{\partial^2 u}{\partial x^2}(i,j) = u_{i+1,j} - 2u_{i,j} + u_{i-1,j} \quad \frac{\partial^2 u}{\partial y^2}(i,j) = u_{i,j+1} - 2u_{i,j} + u_{i,j-1}$$

where $u(i,j) = u_{i,j}$.

We have to solve numerically the following equations:

- For a fixed $v = v_n$, solve for u:

  In 1-Dimension:

  $$2vv'u' + v^2u'' - \frac{\beta}{\alpha}(u - u_0) = 0 \ \ \forall i \in \Omega$$

  In 2-Dimensions:

  $$\partial_x v^2 \cdot \partial_x u + \partial_y v^2 \cdot \partial_y u + v^2(\partial_{xx}u + \partial_{yy}u) - \frac{\beta}{\alpha}(u - u_0) = 0 \ \ \forall (i,j) \in \Omega$$

  which, in a discretized setting leads to:

  In 1-Dimension:

  $$2v_i\left(\frac{v_{i+1} - v_{i-1}}{2}\right)\left(\frac{u_{i+1} - u_{i-1}}{2}\right) + v_i^2(u_{i+1} - 2u_i + u_{i-1}) - \frac{\beta}{\alpha}u_i = \frac{\beta}{\alpha}u_{0i} \quad \forall i \in \Omega$$

  In 2-Dimensions:

  $$\begin{aligned} \left(\frac{v_{i+1,j}^2 - v_{i-1,j}^2}{2}\right)\left(\frac{u_{i+1,j} - u_{i-1,j}}{2}\right) + v_{i,j}^2(u_{i+1,j} - 2u_i + u_{i-1,j}) + \\ + \left(\frac{v_{i,j+1}^2 - v_{i,j-1}^2}{2}\right)\left(\frac{u_{i,j+1} - u_{i,j-1}}{2}\right) + v_{i,j}^2(u_{i,j+1} - 2u_{i,j} + u_{i,j-1}) - \frac{\beta}{\alpha}u_{i,j} &= \frac{\beta}{\alpha}u_{0i,j} \end{aligned}$$

  This last equation is valid $\forall (i,j) \in \Omega$.

  Notice that in both cases we can rewrite, recalling that $v = (v_i)$ is fixed, the equations in the form:

  $$Au = \frac{\beta}{\alpha}u_0$$

  for suitable matrix A, which in 1-D case belongs to $\mathcal{M}_{N\times N}(\mathbb{R})$ where N is the number of points of the discretized domain Ω, notice that in this case the matrix have a tri-diagonal structure. In 2-D the matrix A belongs to $\mathcal{M}_{NM\times NM}(\mathbb{R})$ where N, M are identified with the number of integers of the discretized domain Ω according to definition 1.1.1, is important to notice that in this case the matrix have a diagonal by blocks structure. Notice also that in 2D case we don't expand the derivative $\partial_x v^2 = 2v\partial_x v$ in order to reduce the number of computations involved.

- For a fixed $u = u_n$, solve for v:

  In 1-Dimension:

  $$v'' + \frac{(1-v)}{\varepsilon^2} - \frac{2\alpha}{\nu\varepsilon} \cdot v(u')^2 = 0 \ \ \forall i \in \Omega$$

  In 2-Dimensions:

  $$\partial_{xx} v + \partial_{yy} v + \frac{(1-v)}{\varepsilon^2} - \frac{2\alpha}{\nu\varepsilon} \cdot v((\partial_x u)^2 + (\partial_y u)^2) = 0 \ \ \forall (i,j) \in \Omega$$

  which, in a discretized setting leads to:

  In 1-Dimension:

  $$(v_{i+1} - 2v_i + v_{i+1}) + \frac{1}{\varepsilon^2}\mathbb{1} - \frac{1}{\varepsilon^2} v_i - \frac{2\alpha}{\nu\varepsilon} \cdot v_i \left(\frac{u_{i+1} - u_{i-1}}{2}\right)^2 = 0 \quad \forall i \in \Omega$$

  In 2-Dimensions:

  $$\begin{aligned} (v_{i+1,j} - 2v_{i,j} + v_{i-1,j}) - \frac{1}{\varepsilon^2} v_{i,j} - \frac{2\alpha}{\nu\varepsilon} \cdot v_{i,j} \left(\frac{u_{i+1,j} - u_{i-1,j}}{2}\right)^2 + \\ + (v_{i,j+1} - 2v_{i,j} + v_{i,j-1}) - \frac{2\alpha}{\nu\varepsilon} \cdot v_{i,j} \left(\frac{u_{i,j+1} - u_{i,j-1}}{2}\right)^2 &= -\frac{1}{\varepsilon^2}\mathbb{1} \end{aligned}$$

  Noticing now that $u = (u_i)$ is fixed, we can rewrite the two equations in the form:

  $$Bv = \frac{1}{\varepsilon^2}\mathbb{1}$$

  where $\mathbb{1}$ is a vector of the same dimension as v filled with ones.

  Recall that the matrix B in 1-D case belongs to $\mathcal{M}_{N\times N}(\mathbb{R})$ where N is the number of points of the discretized domain Ω, in this case the matrix have a tri-diagonal structure. In 2-D the matrix B belongs to $\mathcal{M}_{NM\times NM}(\mathbb{R})$ where N, M are identified as before, notice that B also have a diagonal by blocks structure.

Having a discretization defined, we just need to impose the initial conditions in order to define completely our iterative discrete scheme to solve the equations.

As we said before, the choice of the initial conditions u_0, v_0 is extremely important, due to the non-convexity of our functional. In image processing the usual choices as initial conditions for image segmentation are:

$$u^0 = u_0 \text{ (the input image)}$$

$$v^0 = \frac{1}{1+2\alpha\nu\varepsilon|\nabla u_0|^2}$$

The initial condition for u is quite obvious, but let us see how to understand the choice of v^0. Consider the Euler-Lagrange equation for v:

$$\Delta v + \nu\frac{(1-v)}{\varepsilon^2} - \frac{2\alpha\nu \cdot v|\nabla u|^2}{\varepsilon} = 0$$

which can be rewritten as:

$$v = \frac{1+\nu^{-1}\varepsilon^2\Delta v}{1+2\alpha\varepsilon|\nabla u|^2}$$

Since ε is small enough, then $\varepsilon^2 \sim 0$, and then we have, as initial state for the edge function v:

$$v^0 = \frac{1}{1+2\alpha\varepsilon|\nabla u^0|^2}$$

which in 1-Dimension discrete version is:

$$v_i^0 = \frac{2}{2+\alpha\varepsilon\left(u_{i+1}^0 - u_{i-1}^0\right)^2}$$

and in 2-Dimension discrete version is:

$$v_{i,j}^0 = \frac{2}{2+\alpha\varepsilon\left[(u_{i+1,j}^0 - u_{i-1,j}^0)^2 + (u_{i,j+1}^0 - u_{i,j-1}^0)^2\right]}$$

With this conditions, the problem is completely defined, in 1 and 2-Dimensional settings. Examples are provided in the next chapter.

6.3 Non Local Segmentation

Let us consider the following generalized version of the Non Local Ambrosio-Tortorelli functional (with constants α, β in order to penalize more/less the regularization or the boundary search):

$$F_\varepsilon^{NLAT}(u,v) = \alpha\int_\Omega v^2|\nabla_w u|^2 dx + \beta\int_\Omega (u-u_0)^2 dx + \frac{\nu}{2}\int_\Omega \varepsilon|\nabla v|^2 + \frac{(1-v)^2}{\varepsilon}dx \qquad (6.6)$$

recall that:

$$|\nabla_w u|^2(x) = \int_\Omega (u(y)-u(x))^2 w(x,y)dy$$

where $w \in L^\infty(\Omega\times\Omega)$ is a weight function which measure the similarity between neighborhoods of x and y. We also deal (in 1 Dimension only) with the case of:

$$|\nabla_w u|^2(x) = \int_\Omega \frac{(u(y)-u(x))^2}{|y-x|^{1+2s}} w(x,y)dy, \quad s \in (1/2, 1)$$

We have to derive (analogously to the case of the classic AT functional) the first order conditions (or Euler-Lagrange equations) to this functional in order to perform a alternating scheme via fixed point iterations in order to obtain numerically a local minima of this functional. In the next section we will prove the equations involved and in the next one we detail the implementation in order to obtain local minimas numerically.

6.3.1 Euler-Lagrange Equations

Noticing that the functional change only in the gradient of u (to the non-local gradient), is easy to convince that the Gateaux derivative for 6.6 in v is the same as the Gateaux derivative for 6.3 except for the term $|\nabla u|^2$, which changes to $|\nabla_w u|^2$, we resume the equations in the following theorem.

Theorem 6.3.1. *The First order conditions (or Euler-Lagrange equations) for the functional defined by 6.6 with $(u,v) \in L^2(\Omega) \times H^1(\Omega)$ and respectively $(u,v) \in H^s(\Omega) \times H^1(\Omega)$ are:*

$$L^{NL/MS}(u,v) + \frac{\beta}{\alpha}(u-u_0) = 0 \ \textit{in}\ \Omega, \tag{6.7}$$

$$\Delta v + \frac{(1-v)}{\varepsilon^2} - \frac{2\alpha \cdot v|\nabla_w u|^2}{\nu\varepsilon} = 0 \ \textit{in}\ \Omega. \tag{6.8}$$

where

$$L^{NL/MS}(u,v)(x) = -2\int_\Omega (u(y)-u(x))\tilde{w}(x,y)\cdot(v^2(y)+v^2(x))dy \quad x \in \Omega \tag{6.9}$$

and

$$\tilde{w}(x,y) = w(x,y) \quad \textit{if}\ |\nabla_w u|^2(x) = \int_\Omega (u(y)-u(x))^2 w(x,y)dy$$

$$\tilde{w}(x,y) = \frac{w(x,y)}{|y-x|^{1+2s}} \quad \textit{if}\ |\nabla_w u|^2(x) = \int_\Omega \frac{(u(y)-u(x))^2}{|y-x|^{1+2s}} w(x,y)dy$$

Proof. Notice that we just have to compute

$$D_u F_\varepsilon^{NLAT}(u,v)[\varphi] = \lim_{t\to 0} \frac{F_\varepsilon^{NLAT}(u+t\varphi, v) - F_\varepsilon^{NLAT}(u,v)}{t}$$

because the computation of $D_v F_\varepsilon^{NLAT}(u,v)[\varphi]$ is exactly the same as for the Ambrosio-Tortorelli functional, from which we get the second equation of the theorem.

Let us compute the derivative on u for the functional, for this, let us compute in the first case:

$$\begin{aligned} F_\varepsilon^{NLAT}(u+t\varphi,v)-F_\varepsilon^{NLAT}(u,v) &= \alpha\int_\Omega(v^2|\nabla_w(u+t\varphi)|^2-v^2|\nabla_w u|^2)dx \\ &+\beta\int_\Omega((u+t\varphi)-u_0)^2-(u-u_0)^2dx \end{aligned}$$

Notice that:

$$\begin{aligned} |\nabla_w(u+t\varphi)|^2-|\nabla_w u|^2 &= \int_\Omega w(x,y)\left[((u+t\varphi)(y)-(u+t\varphi)(x))^2-(u(y)-u(x))^2\right]dy \\ &= \int_\Omega w(x,y)[2tu(y)\varphi(y)+t^2\varphi^2(y)-2t\varphi(y)u(x)-2t\varphi(x)u(y) \\ &\quad -2t^2\varphi(x)\varphi(y)+t^2\varphi^2(x)+2tu(x)\varphi(x)]dy \end{aligned}$$

And:

$$((u+t\varphi)-u_0)^2-(u-u_0)^2 = 2t(u-u_0)\varphi+t^2\varphi^2$$

Then we get:

$$\begin{aligned} F_\varepsilon^{NLAT}(u+t\varphi,v)-F_\varepsilon^{NLAT}(u,v) &= \beta\int_\Omega[2t(u-u_0)\varphi+t^2\varphi^2]dx+\alpha\int_\Omega v^2(x)\int_\Omega w(x,y)\cdot \\ &\quad [2tu(y)\varphi(y)+t^2\varphi^2(y)-2t\varphi(y)u(x)-2t\varphi(x)u(y) \\ &\quad -2t^2\varphi(x)\varphi(y)+t^2\varphi^2(x)+2tu(x)\varphi(x)]dydx \end{aligned}$$

Therefore:

$$\begin{aligned} \frac{F_\varepsilon^{NLAT}(u+t\varphi,v)-F_\varepsilon^{NLAT}(u,v)}{t} &= \beta\int_\Omega 2(u-u_0)\varphi+t\varphi^2dx+\alpha\int_\Omega v^2(x)\int_\Omega w(x,y)\cdot \\ &\quad [2u(y)\varphi(y)+t\varphi^2(y)-2\varphi(y)u(x)-2\varphi(x)u(y) \\ &\quad -2t\varphi(x)\varphi(y)+t\varphi^2(x)+2u(x)\varphi(x)]dydx \end{aligned}$$

Then, by a standard application of Dominated Convergence Theorem we have:

$$\begin{aligned} \lim_{t\to 0}\frac{F_\varepsilon^{NLAT}(u+t\varphi,v)-F_\varepsilon^{NLAT}(u,v)}{t} &= \beta\int_\Omega 2(u-u_0)\varphi dx+\alpha\int_\Omega v^2(x)\int_\Omega w(x,y)\cdot \\ &\quad [2u(y)\varphi(y)-2\varphi(y)u(x)-2\varphi(x)u(y)+2u(x)\varphi(x)]dydx \end{aligned}$$

Therefore:

$$\begin{aligned} D_uF_\varepsilon^{NLAT}(u,v)[\varphi] &= \beta\int_\Omega 2(u-u_0)\varphi dx+\alpha\int_\Omega v^2(x)\int_\Omega w(x,y)\cdot[-2(\varphi(x)-\varphi(y)) \\ &\quad \cdot(u(y)-u(x))]dydx \end{aligned}$$

But, notice that:

$$\int_\Omega v^2(x)\int_\Omega w(x,y)\cdot[-2(\varphi(x)-\varphi(y))\cdot(u(y)-u(x))]dydx =$$

$$\int_\Omega \varphi(x)\int_\Omega v^2(x)\cdot[-2w(x,y)(u(y)-u(x))]dydx-\int_\Omega\int_\Omega v^2(x)\cdot[-2\varphi(y)w(x,y)(u(y)-u(x))]dydx$$

Using Fubini Theorem we get:

$$\begin{aligned}\int_\Omega\int_\Omega v^2(x)\cdot[-2\varphi(y)w(x,y)(u(y)-u(x))]dydx &= \int_\Omega\int_\Omega v^2(y)\cdot[-2\varphi(x)w(y,x)(u(x)-u(y))]dxdy\\ &= \int_\Omega \varphi(x)\int_\Omega v^2(y)\cdot[2w(x,y)(u(y)-u(x))]dydx\end{aligned}$$

Where we used that the weight function is symmetric: $w(x,y)=w(y,x)$. Finally:

$$\begin{aligned}D_uF_\varepsilon^{NLAT}(u,v)[\varphi] &= \beta\int_\Omega 2(u-u_0)\varphi dx+\alpha\int_\Omega\varphi(x)\int_\Omega v^2(x)\cdot[-2w(x,y)(u(y)-u(x))]dydx\\ &\quad+\alpha\int_\Omega\varphi(x)\int_\Omega v^2(y)\cdot[-2w(x,y)(u(y)-u(x))]dydx\end{aligned}$$

Then, because we are imposing the first order condition, we have:

$$D_vF_\varepsilon^{NLAT}(u,v)[\varphi]=0\ \ \forall\varphi\in\mathcal{C}_0^\infty(\Omega)$$

we can conclude using 2.3.3 that, $\forall x\in\Omega$:

$$\beta(u-u_0)+\alpha\int_\Omega v^2(x)\cdot[-2w(x,y)(u(y)-u(x))]dydx+\alpha\int_\Omega v^2(y)\cdot[-2w(x,y)(u(y)-u(x))]dydx=0$$

Which can be rewritten as:

$$\frac{\beta}{\alpha}(u-u_0)-2\int_\Omega(v^2(x)+v^2(y))\cdot[w(x,y)(u(y)-u(x))]dydx=0\ \ x\in\Omega$$

or, in the notation of the theorem:

$$L^{NL/MS}(u,v)+\frac{\beta}{\alpha}(u-u_0)=0\ \text{ in }\Omega$$

Noticing that this computation is independent of $w(x,y)$ (and of which $|\nabla_w u|$ we choose) we cover the two cases with this computation.

Therefore we conclude the proof of the theorem. □

6.3.2 Implementation

In this case we have to separate the implementation of 1-Dimensional and 2-Dimensional due to the non-local operators involved on Euler-Lagrange equations. In 1-Dimensional case we will follow the same prodecure to solve the equations: Alternating Minimization on the discretized systems.

In 2-Dimensional case the treatment should be different, basically due to the non-local terms present in the equations 6.7, if we try to follow the same structure (a direct finite differences discretization) the number of computations and the complex form of the matrices involved lead to fail a fixed point iterative scheme such as the used in classical Ambrosio-Tortorelli scheme and 1D version of Non-Local segmentation. An extension of a procedure used for non-local regularization (i.e. minimizing the functional without the length function v) based on a technique called 'Split-Bregman' was developed in MOTIV Laboratory from the Center of Mathematical Modelling, this method uses various simplifications in order to get a reasonable computation times for the segmentation. We detail this on further subsection.

Finally, is important to mention that in 1D case we deal with the two forms of $|\nabla_w u|$ just by considering different weight functions, just as how we mentioned in the theorem. Unfortunately in 2D we didn't get proper implementation with the redefined functional, then this experiment remains as a future work.

6.3.2.1 1D Implementation

Following the implementation of the Ambrosio-Tortorelli functional, the idea of an implementation in this case is quite similar: We have to solve two equations depending of u and v, this equations are not convex in the variable (u, v), but are convex when we consider separate variables, then, the idea is again apply a alternating minimization scheme. We will have some troubles dealing with the construction of the weight function w, because the construction of the discretized version of this function involves a high number of computations. We discuss in the next section briefly how to deal with this problem, based on the 'fast versions' proposed by Gilboa and Osher in [31] Section 5.

The discretization of the functions involved in this minimization are:

Let u_k the value of a pixel k in the image ($1 \leq k \leq N$) i.e. the discretization of $u(x)$, let $h_{k,l}$ be the discretized version of $h(x, y)$ with $x = k$, $y = l \in \Omega$.

Let $w_{k,l}$ the sparsely discrete version of the weight function w. We follow the same notation for neighborhood sets as in [31], [20]: $l \in \mathcal{N}_k = \{l : w_{k,l} > 0\}$.

Then, we define ∇_{wd} the discretization of ∇_w as:

$$\nabla_{wd}(u_k) := (u_l - u_k)\sqrt{w_{k,l}} \quad l \in \mathcal{N}_k$$

Recall that:

$$|h|_k = \sqrt{\sum_l (h_{k,l})^2} \text{ the magnitude of } h_{k,l} \text{ at } k$$

Therefore, the discretization of $|\nabla_w u|^2$ in the pixel $x = k \in \Omega$:

$$|\nabla_{wd} u_k|^2 = \sum_l (u_l - u_k)^2 w_{k,l}$$

Is important to notice that this discretizations are valid if $w(x, y)$ is well defined for each $x, y \in \Omega$, otherwise (for example, if we consider $\tilde{w}(x, y) = w(x, y)/(x - y)^{1+2s}$) we have to redefine the discretization.

Recall that we have to discretize the equations 6.7, we will follow the same scheme as in the case of the Ambrosio-Tortorelli approximation, and we will study separately the cases $\tilde{w}(x,y) = w(x,y)$ and $\tilde{w}(x,y) = w(x,y)/(x-y)^{1+2s}$, $s \in (1/2, 1)$.

- If $\tilde{w}(x,y) = w(x,y)$, for $u = u_n$ fixed, we have to solve:

  $$\Delta v + \frac{(1-v)}{\varepsilon^2} - \frac{2\alpha}{\nu\varepsilon} \cdot v|\nabla_w u|^2 = 0$$

  which becomes, for each $i \in \Omega$:

  $$(v_{i+1} - 2v_i + v_{i-1}) - \frac{1}{\varepsilon^2} v_i - \frac{2\alpha}{\nu\varepsilon} v_i |\nabla_{wd} u_i|^2 = -\frac{1}{\varepsilon^2}$$

  Therefore, by the discretization of non-local gradient given before:

  $$(v_{i+1} - 2v_i + v_{i-1}) - \frac{1}{\varepsilon^2} v_i - \frac{2\alpha}{\nu\varepsilon} v_i \cdot \sum_l (u_l - u_i)^2 w_{k,i} = -\frac{1}{\varepsilon^2}$$

  which can be written in the form:

  $$Av = \frac{1}{\varepsilon^2}\mathbb{1}$$

  Is important to notice that in this case, A again have a tri-diagonal structure, this is due that the non-local term is on u, which is fixed for this computation.

- If $\tilde{w}(x,y) = w(x,y) \in L^\infty$, for $v = v_n$ fixed, we have to solve:

  $$L^{NL/MS}(u,v) + \frac{\beta}{\alpha}(u - u_0) = 0$$

  first notice that:

  $$L^{NL/MS}(u,v)(x) = -2\int_\Omega (u(y) - u(x))w(x,y) \cdot (v^2(y) + v^2(x))dy$$

  is discretized in $x = i$ as (noticing that in the sum the term $l = i$ vanishes):

  $$L^{NL/MS}(u_i, v_i) = -2\sum_{l \neq i} (u_l - u_i) w_{li} \cdot (v_l^2 + v_i^2)$$

  therefore, we have that the discretized equation is, for each $i \in \Omega$:

  $$-2\sum_{l \neq i} (u_l - u_i) w_{li} \cdot (v_l^2 + v_i^2) + \frac{\beta}{\alpha} u_i = \frac{\beta}{\alpha} u_{0i}$$

this can be rewritten as, for each $i \in \Omega$:

$$u_i \cdot \left(\frac{\beta}{\alpha} + 2 \sum_{l \neq i} w_{li} \cdot (v_l^2 + v_i^2) \right) + \sum_{l \neq i} u_l \cdot \left(-2 w_{li} \cdot (v_l^2 + v_i^2) \right) = \frac{\beta}{\alpha} u_{0i}$$

Which can be finally written in the form:

$$Bu = \frac{\beta}{\alpha} u_0$$

Notice that in this case the non local term $L(u, v)$ have a high influence on the matrix structure of B, more specifically, the structure of B is highly determined by the weights w_{li}, thus, if we limitate this function to have a compact support around each fixed i, then, the matrix B will have many diagonals as the width of the support.

- In the case that $\tilde{w}(x, y) = w(x, y)/(x - y)^{1+2s}$ we just have to introduce the following changes:
For the equation with $u = u_n$ fixed, we have to redefine the term $|\nabla_{wd} u_i|^2$, because in its original form the terms of the sum may be singular when $l = i$, so, we define it in the following way:

$$\begin{aligned} |\nabla_{wd} u_i|^2 = \sum_l \frac{(u_l - u_i)^2}{|l - i|^{1+2s}} w_{l,i} &= \sum_{l \neq i} \frac{(u_l - u_i)^2}{|l - i|^{1+2s}} \cdot w_{l,i} + (u_i')^2 \cdot \frac{1}{\delta} \\ &= \sum_{l \neq i} \left(\frac{u_l - u_i}{l - i} \right)^2 \cdot p_{li} + \left(\frac{u_{i+1} - u_{i-1}}{2} \right)^2 \cdot \frac{1}{\delta} \end{aligned}$$

In this way this quantity is well defined, the idea behind this formula is noticing that in the limit $y \to x$ the function $w(x, y)$ becomes unbounded, but multiplying it by $(u(y) - u(x))^2$ we have, if u is regular enough that:

$$\lim_{y \to x} (u(y) - u(x))^2 \frac{w(x, y)}{(y - x)^2} = (u'(x))^2 w(x, x) = (u'(x))^2$$

the last equality is due that we usually consider $w(x, y)$ such that $w(x, x) = 1$ for all x. At the same time we regularize the extra power $\frac{1}{(y - x)^{2s-1}}$ by replacing it by the quantity $\frac{1}{\delta}$ with a small parameter δ.

Moreover, we conclude that, defined in this way, the non local gradient (in 1 dimension) is the classical local derivative plus some ‘non local’ terms, which is the precise idea of this kind of operator.

In a similar way, if $v = v_n$ is fixed, we redefine the discretized term $L^{NL/MS}(u, v)$ regularizing it, due to the unboundness in the term $l = i$, introducing again a small parameter δ (usually 0.1):

$$L^{NL/MS}(u_i, v_i) = -2 \sum_{l \neq i} \frac{u_l - u_i}{(l - i)^{1+2s}} \cdot w_{li} \cdot (v_l^2 + v_i^2) - 4 \frac{u_i}{\delta} \cdot w_{ii} \cdot v_i^2$$

With this small changes the operators are well defined in the case of our study.

Finally, in order to perform the alternating minimization, in the same way as the performed for local segmentation reviewed before, we strongly need a suitable initial conditions, in this case we choose for initialization: $u^0 = u_0$ the input image, and the same v^0 as before:

$$v^0 = \frac{1}{1 + 2\alpha\varepsilon|\nabla u_0|^2}$$

6.3.2.2 2D Implementation

In this case, the segmentation is performed minimizing the Non-Local Shah approximation of the Mumford-Shah functional also known as Total Variation Mumford Shah approximation, i.e. minimizing the functional:

$$\begin{aligned} F_\varepsilon^{NLTV}(u,v) &= \alpha \int_\Omega v^2|\nabla_w u|dx + \beta\int_\Omega (u-u_0)^2dx + \frac{\nu}{2}\int_\Omega \varepsilon|\nabla v|^2 + \frac{(1-v)^2}{\varepsilon}dx \\ &= \alpha\|v^2|\nabla_w u|\|_{L^1} + F^{AT}(v) + \beta\int_\Omega (u-u_0)^2dx \end{aligned} \tag{6.10}$$

The choice of this functional is due to the existence of efficient implementations on 2-D variational denoising methods (i.e. minimization on u of $F_\varepsilon^{NLTV}(u,1)$), this lead to propose an extension which presents very interesting results.

The method considered to perform the minimization is based on a technique known as 'Bregman iterations' defined in [15], specifically an extension due to Goldstein and Osher known as 'Split-Bregman method' [32], which is specially useful for problems where L^1-like norms are involved (which is our case in u).

The idea involved in this optimization method is to transform the total variation problem into an ℓ^1 norm minimization by introducing and auxiliary variable for the L^1 norm, and then use an efficient algorithm to solve this new problem. Consider the discrete problem:

$$\min_{u,v} \alpha\|v^2|\nabla_w u|\|_{\ell^1} + F_d^{AT}(v) + \beta\|u-u_0\|^2dx \tag{6.11}$$

We introduce the auxiliary variable d such that we solve the equivalent problem:

$$\min_{u,v,d} \alpha\|v^2 d\|_{\ell^1} + F_d^{AT}(v) + \beta\|u-u_0\|^2dx \quad s.t.\ d = |\nabla_w u| \tag{6.12}$$

To solve this problem, we first convert it into an unconstrained problem:

$$\min_{u,v,d} \alpha\|v^2 d\|_{\ell^1} + F_d^{AT}(v) + \beta\|u-u_0\|^2dx + \frac{\lambda}{2}\|d - |\nabla_w u|\|^2 \tag{6.13}$$

Finally, this problem can be solved using the Split Bregman Iteration [47], [16] scheme:

$$\begin{aligned} (u^{k+1}, v^{k+1}, d^{k+1}) &= argmin_{u,v,d}\ \alpha\|v^2 d\|_{\ell^1} + F_d^{AT}(v) + \beta\|u-u_0\|^2dx + \frac{\lambda}{2}\|d - |\nabla_w u| - b^k\|^2 \\ b^{k+1} &= b^k + |\nabla_w u^{k+1}| - d^{k+1} \end{aligned} \tag{6.14}$$

The solution of this problem is obtained by performing an alternating minimization process:

$$\begin{aligned}
u^{k+1} &= argmin_u \; \beta\|u-u_0\|^2dx + \frac{\lambda}{2}\|d-|\nabla_w u|-b^k\|^2 \\
v^{k+1} &= argmin_v \; \alpha\|v^2 d\|_{\ell^1} + F_d^{AT}(v) + \frac{\lambda}{2}\|d-|\nabla_w u|-b^k\|^2 \\
d^{k+1} &= argmin_d \; \alpha\|v^2 d\|_{\ell^1} + \frac{\lambda}{2}\|d-|\nabla_w u|-b^k\|^2 \\
b^{k+1} &= b^k + |\nabla_w u^{k+1}| - d^{k+1}
\end{aligned} \tag{6.15}$$

For more information of the method in a general context and convergence issues we refer to the works: [32], [47], [15].

6.4 Computing weights

The most important and demanding in computational terms part of the non local methods is the approximation of $w(x,y)$ into a sparse discrete version w_{kl}.

In [31] Section 3.1, the authors proposed two algorithms: the first, a semilocal one, is proposed for denoising purposes; the second, a fully nonlocal based on random choices instead of checking all the possibilities.

6.4.1 Semi-local version

Algorithm. For each pixel k:

1. Compute the similarity of all the patches in the window (Authors use 5×5 patch B_x and 11×11 window Ω_w). Construct $\mathcal{N}_k$ by taking the m (Authors use $m=5$) most similar and the four nearest neighbors (for a connectedness necessary condition) of the pixel.

2. Compute the weights w_{kl}, $l\in\mathcal{N}_k$ using the desired weight function and set to zero all the other connections. (i.e. $w_{kl}=0,\; l\notin\mathcal{N}_k$)

3. Set $w_{lk}=w_{kl}$ (symmetry of weight)

The complexity of this algorithm is $O(N\times Window_{size}\times(Patch_{size}+\log m))$.

6.4.2 Fast approximation for the fully nonlocal version

The following algorithm is based on ideas presented in [39]. It is simpler and faster but not accurate, anyway the results are better than the original fully nonlocal version.

Algorithm:

1. Compute the mean and the standard deviation of all patches in the image. Create a two dimensional bin table such that all patches in a bin are within a specific range of mean and s.d. from each other. Both types of bins are spaced in $h/2$ increments.

2. To construct the set $\mathcal{N}_k$: For each pixel k we consider the 9 bins around it (3×3 window in the table, this ensures that patches which are very similar are taken into account). Pick randomly $3m$ patches from these bins, check their similarity to the patch of pixel k and take the most similar m of them. Add to $\mathcal{N}_k$ also the four nearest neighbors (for connectedness necessary condition)

3. Compute w_{kl} as in the local algorithm.

6.5 Gabor Functions for Texture recognizion

Recall from 5.3 that we have to solve the problem 5.1, i.e.:

$$\min_{c=(c_i)} \sum_{i=1}^{N} (f(x_i) - u_i)^2 + \gamma ||c||_1$$

solving this problem we get a sparse solution which approximates/regularizes the known data (the input signal) by a given family of functions (in our case the Gabor family/Exponential spline family), after that we also have to refine the minimization, solving a least-square problem using only the non-null coefficients associated with the active functions of the family which approximates the input data, i.e. solving:

$$\min \sum_{i=1}^{N} (c_i \cdot g(x_i) - u_i)^2$$

fixing $c_i = 0$ as a constraint for c_i obtained as 0 in the previous minimization problem. The least-squares final problem is quite standard to solve numerically, due to its quadratic programming structure, we solve it by standard solvers.

Solve the first minimization problem that we have to deal is quite more complicated because its form is not (in principle) a known one. So, we first try to rewrite it on a known form. For this, let us consider first the following problem:

$$\min_x ||Ax - b||_1$$

this problem can be formulated as an LP of the form:

$$\begin{aligned} &\min_{x,y} && \mathbb{1}^T y \\ &\text{subject to} && -y \le Ax - b \le y \end{aligned}$$

which can be rewritten in a single variable as:

$$\begin{aligned} &\min_{\tilde{x}} && c^T \tilde{x} \\ &\text{subject to} && \tilde{A}\tilde{x} \le \tilde{b} \end{aligned}$$

where: $\tilde{x} = \begin{bmatrix} x \\ y \end{bmatrix}$, $\tilde{c} = \begin{bmatrix} 0 \\ \mathbb{1} \end{bmatrix}$, $\tilde{A} = \begin{bmatrix} A & -I \\ -A & -I \end{bmatrix}$, $\tilde{b} = \begin{bmatrix} b \\ -b \end{bmatrix}$

keeping this in mind, we can reformulate our basis pursuit problem as follows:

$$\min_{c=(c_i)} \sum_{i=1}^{N} (f(x_i) - u_i)^2 + \gamma \|c\|_1 \Leftrightarrow \min_{c=(c_i)} \|Ac - u\|_2^2 + \gamma \|c\|_1$$

where A is the matrix of evaluations of the Gabor Family functions as we seen before on 5.

Notice then, that we have that the problem:

$$\min_{c=(c_i)} \|Ac - u\|_2^2 + \gamma \|c\|_1$$

without loss of generality we can consider $\gamma = 1$, notice that this problem is equivalent to:

$$\begin{array}{ll} \min\limits_{c,y} & \|Ac - u\|_2^2 + \mathbb{1}^T y \\ \text{subject to} & -y \leq c \leq y \end{array}$$

and this problem is equivalent to the following program:

$$\begin{array}{ll} \min\limits_{c,y} & c^T A^T A c - 2u^T A c + \mathbb{1}^T y \\ \text{subject to} & -y \leq c \leq y \end{array}$$

Which can be formulated as a classical quadratic cone program if we take $\tilde{x} = [c, y]^T$, because in that case we have:

$$\begin{array}{ll} \min\limits_{\tilde{x}} & \frac{1}{2} \tilde{x}^T P \tilde{x} + \tilde{c}^T \tilde{x} \\ \text{subject to} & G\tilde{x} \leq 0 \end{array} \tag{6.16}$$

where $\tilde{c} = \begin{bmatrix} -2u^T A \\ \mathbb{1} \end{bmatrix}$, $\tilde{G} = \begin{bmatrix} I & -I \\ -I & -I \end{bmatrix}$, $P = \begin{bmatrix} 2A^T A & 0 \\ 0 & 0 \end{bmatrix}$.

This problem is solvable by the standard quadratic program solvers, there are many choices for this and we choose the CVXOPT routines to perform this minimization.

Is important to mention that the computation can be strongly improved if we develop a 'customized' solver for the system involved when one impose the KKT conditions for this optimization problem, an example of this can be found in [26].

Summaring the structure of the implementation to perform the complete process of Gabor Function approximation in order to study textures is:

1. Construct que matrix A composed by the evaluation of each Gabor Function over the discretized points of the domain.

2. Solve the basis pursuit problem, i.e. Given data u, we look for the solution (the coefficients c) of: $\min_c \|Ac - u\|_2^2 + \|c\|_1$. This is performed by a computational solver considering the quadratic program structure studied before. From this, we get a solution $\hat{c}$.

3. Given the solution $\hat{c}$ of the step before, let $\hat{B} = \{i \mid \hat{c}_i \neq 0\}$. Consider the least squares problem:
$$\min_c \|Ac - u\|_2^2$$
subject to $c_i = 0$ if $i \notin \hat{B}$. This final step is optional but ensures the optimal choices for the coefficients in order to minimize the approximation in 2-norm. Call the solution $\hat{c}$. The solution can be computed directly with quadratic programming solvers.

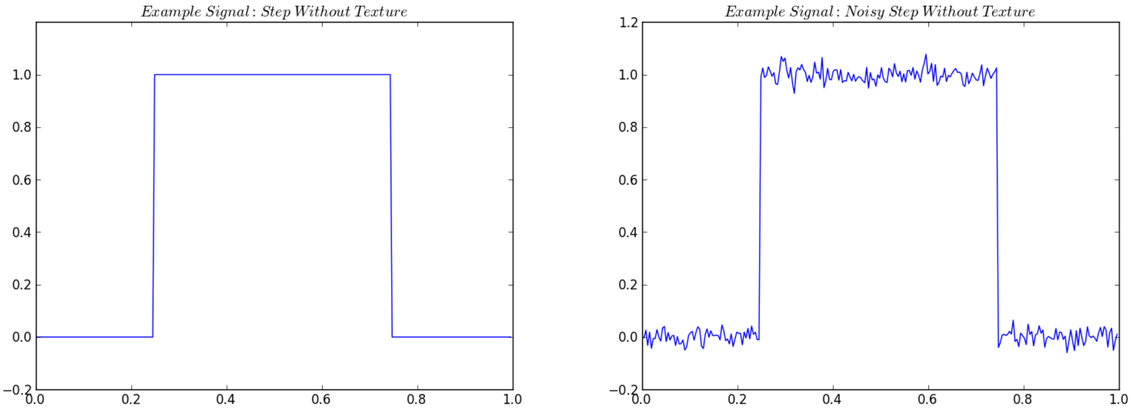
4. Finally, return $\hat{c}$ and reconstruct the signal computing the approximated signal $u_{app} = A\hat{c}$.

Chapter 7

Numerical Examples

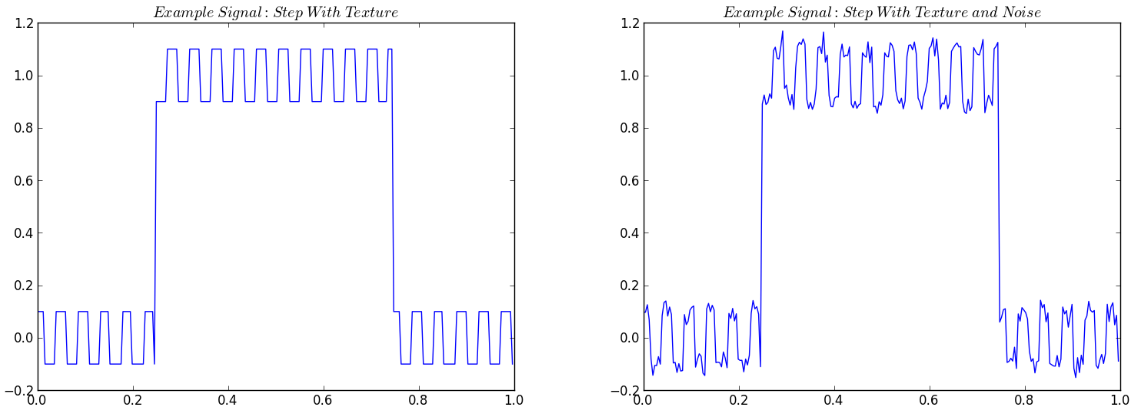
In this chapter we study some numerical examples in order to compare the performance of several methods with 'suitable signals/images' which are intended to exploit the most important properties of the studied methods.

In the 1-Dimensional setting, we will consider the following synthetic signals:



(a) (b)

Figure 7.1: Step signal without Texture. (a) Without noise. (b) With noise.



(a) (b)

Figure 7.2: Step signal with Texture. (a) Without noise. (b) With noise.

The choice of this test signals is in order to study how the methods perform in specific scenarios:

- In the signal without noise and texture: In order to experiment how the methods perform in a ‘basic’ problem, in which the noise and texture are not involved.
- In the signal with noise but without texture: In order to experiment how the methods perform in a ‘classical’ problem, no texture is included yet.
- In the signal with texture but without noise: In order to experiment how the methods perform in a ‘target’ problem, without including the noise yet to discover how the methods behave exclusively with the texture.
- In the signal with texture and noise: In order to experiemnt how the methods perform in the ‘real-targetted’ problem.

We also consider in some examples shifted versions of this signals, in order to verify the correctness of algorithms.

For the 2-Dimensional setting, we will consider the following ‘model’ images: a natural one of a tiger in a natural enviroment and two synthetic images, a lion and a moose in a textured space. Is important to mention that we also consider this images with its noisy versions.

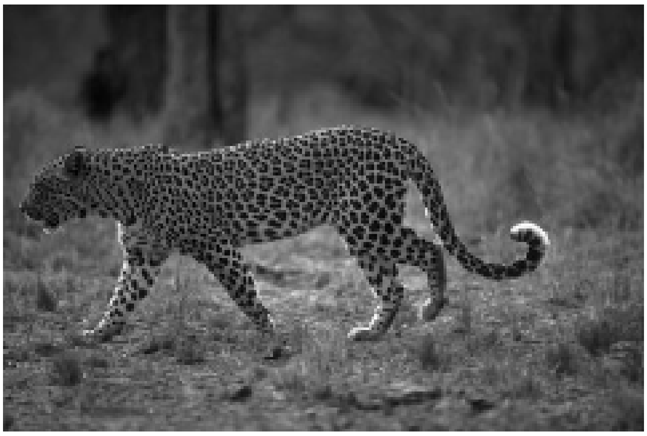
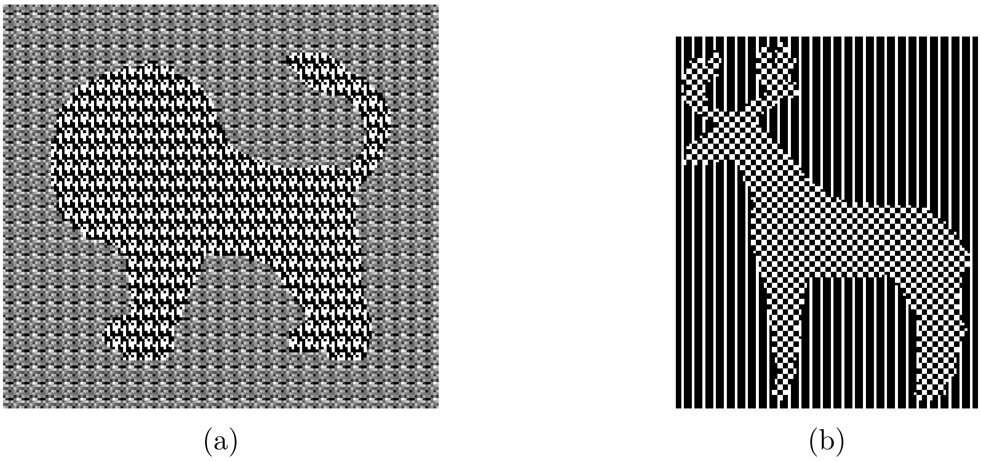


Figure 7.3: 2D Natural Test Image: Tiger in natural enviroment



(a)

(b)

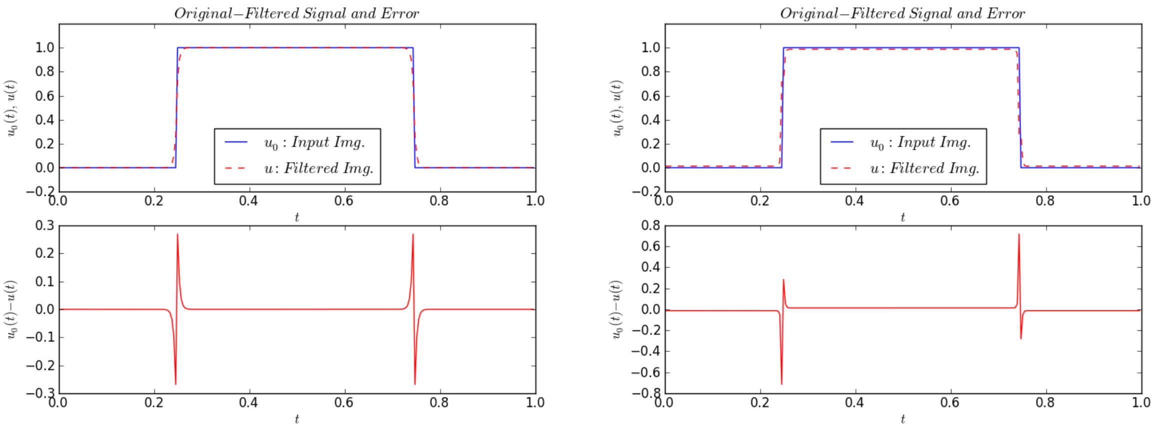
Figure 7.4: 2D Artificial Test Images. (a) Lion (b) Moose

The artificial examples are devoted to explore the behaviour of the methods in a very specific task, the recognizement of an object in a textured environment, the introduction of noise is to explore the behaviour of the methods in 'classical' cases, and finally the introduction of the natural image is to explore the full potential of the methods.

All the numerical examples have been performed in a PC with Microsoft Windows 7 powered by Intel i7 Processor. The coding language is Python bundled in Pythonxy distribution. The optimization routines for Gabor Functions involved processes are from the package CVXOPT.

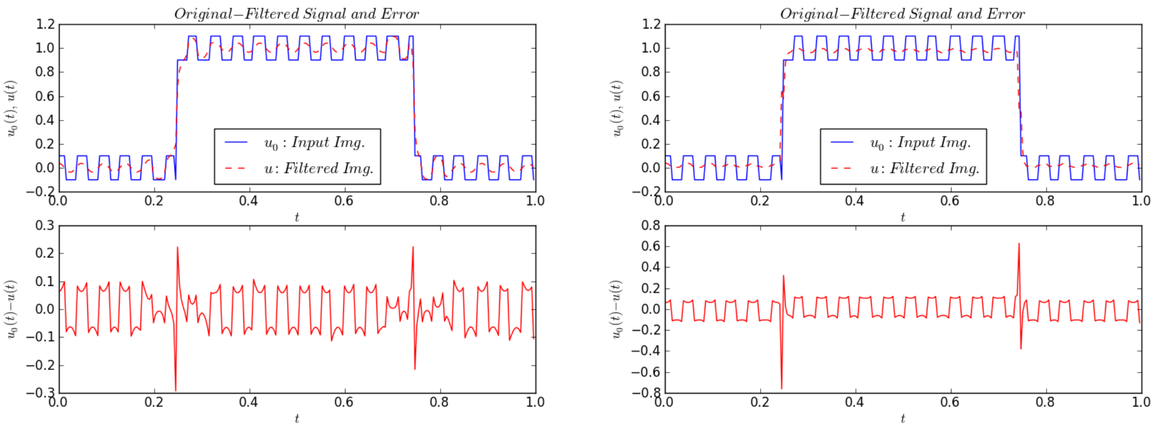
7.1 Non Local Denoising Filter

7.1.1 1-Dimensional Filter Examples



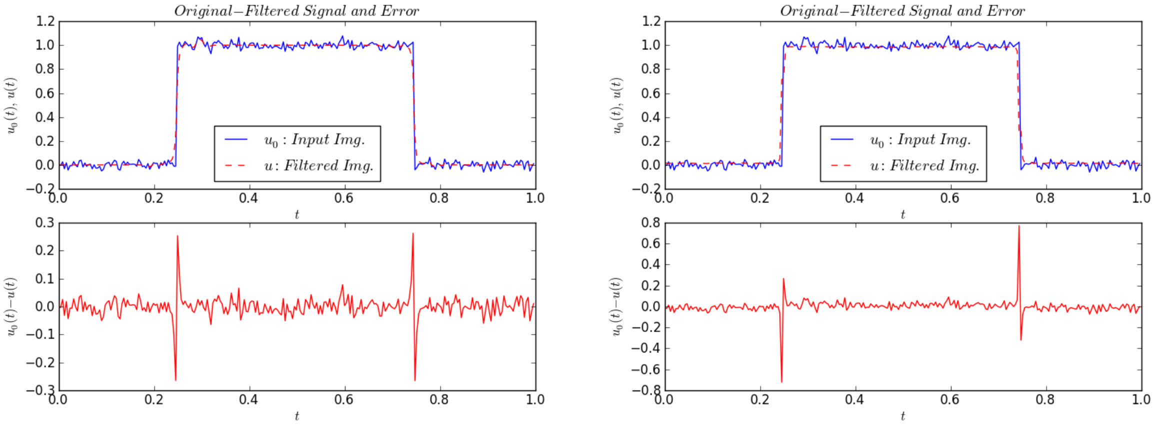
(a) (b)

Figure 7.5: 1D Example of Non Local denoising Filter: Signal without texture and noise. (a) Denoising with Rectangular search window. (b) Denoising with Gaussian search window.



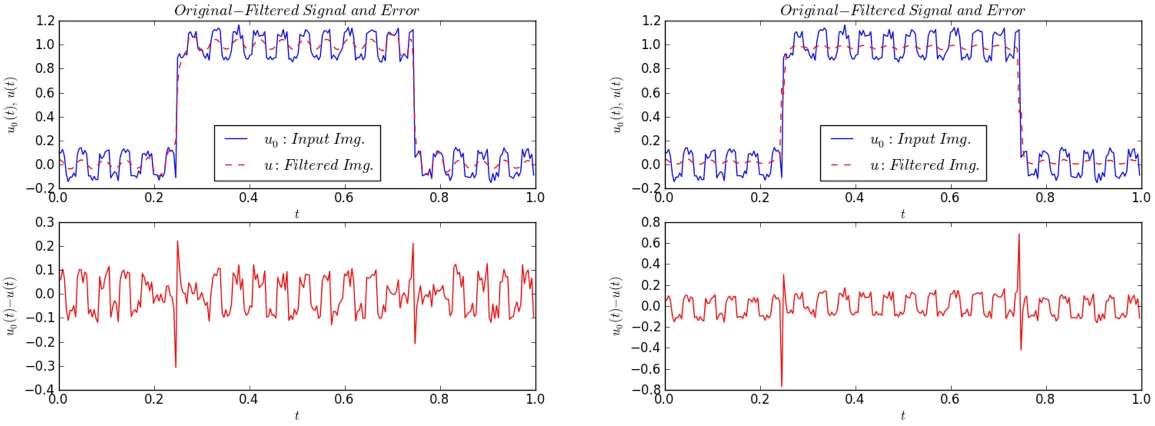
(a) (b)

Figure 7.6: 1D Example of Non Local denoising Filter: Signal with texture but without noise. (a) Denoising with Rectangular search window. (b) Denoising with Gaussian search window.



(a) (b)

Figure 7.7: 1D Example of Non Local denoising Filter: Signal with noise but without texture. (a) Denoising with Rectangular search window. (b) Denoising with Gaussian search window.



(a) (b)

Figure 7.8: 1D Example of Non Local denoising Filter: Signal with noise and texture.
(a) Denoising with Rectangular search window. (b) Denoising with Gaussian search window.

From this simulations we can conclude that:

- In the basic case, the signal without texture and noise, the filter gives desirable results with the two versions of search windows, in the sense that we recover the original signal almost completely, naturally we see that in edges we get a kind of ‘regularization’.

- In the case of noisy signal with no texture, the filter also gives desirable results with the two versions of search windows in the same sense as the before.

- In the case of textured signal the results let us discuss some important things. First of all we discover that the Gaussian search window is not suitable for textured signal denoising, since it ‘almost’ banish the amplitude of the texture, notice that it happens in both cases (with or without noise). We can see that in the case of using a rectangular search window the denoising is quite good, the remotion of noise is high with a little loss of amplitude of the texture, which is still recognizable.

Therefore, for the 1-Dimensional non-local denoising we can conclude that the better choice of search windows are the rectangular ones, and this method is very effective for textured images, even with noise we can recover the complete structure of the texture, and in more simpler cases (images without texture) the original image is almost completely recovered.

7.1.2 2-Dimensional Filter Examples

7.1.2.1 Artificial Example: Moose

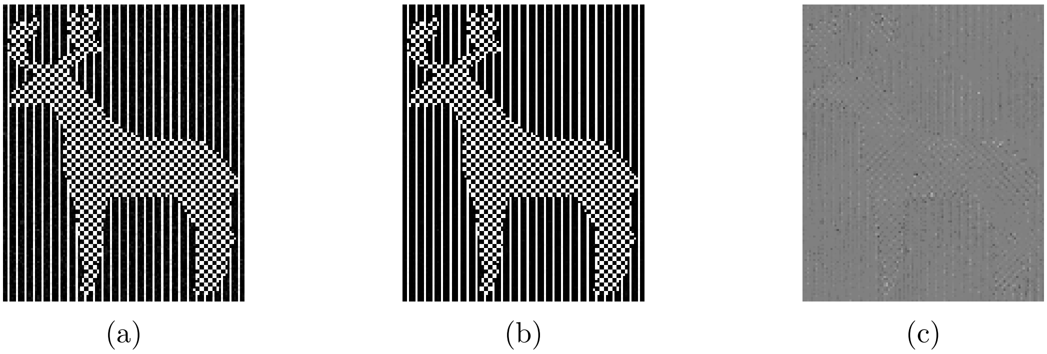

(a) (b) (c)

Figure 7.9: 2D Example of Non Local denoising Filter: Artificial Moose with Gaussian noise ($\sigma = 15$). (a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

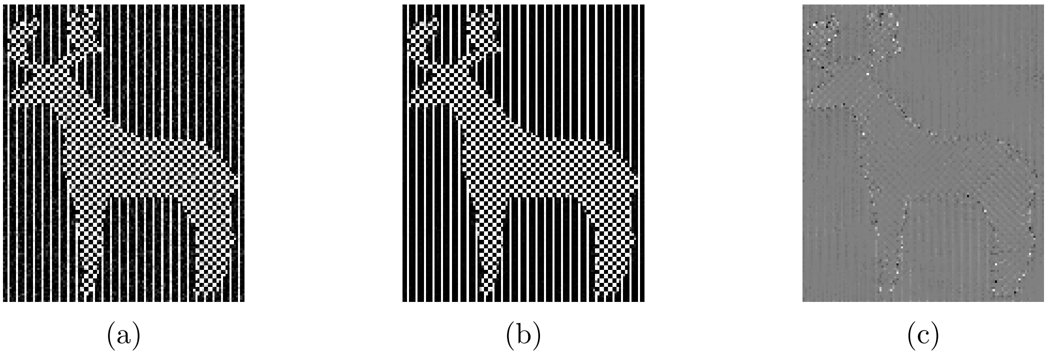

(a) (b) (c)

Figure 7.10: 2D Example of Non Local denoising Filter: Artificial Moose with Gaussian noise ($\sigma = 30$). (a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

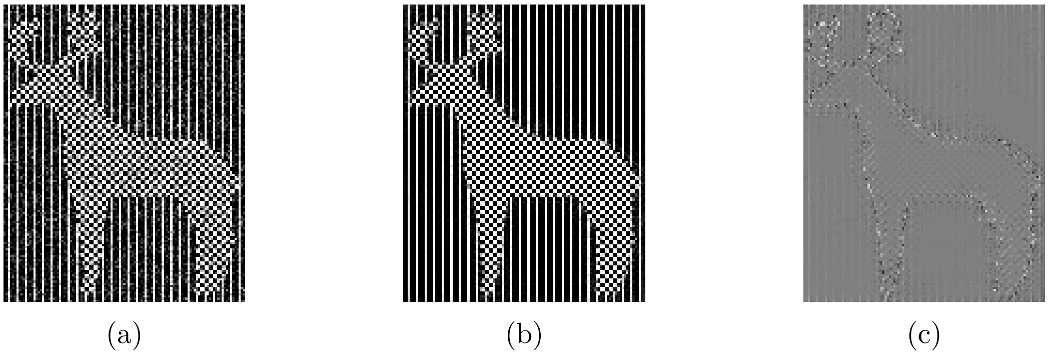

(a) (b) (c)

Figure 7.11: 2D Example of Non Local denoising Filter: Artificial Moose with Gaussian noise ($\sigma = 50$). (a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

7.1.2.2 Artificial Example: Lion

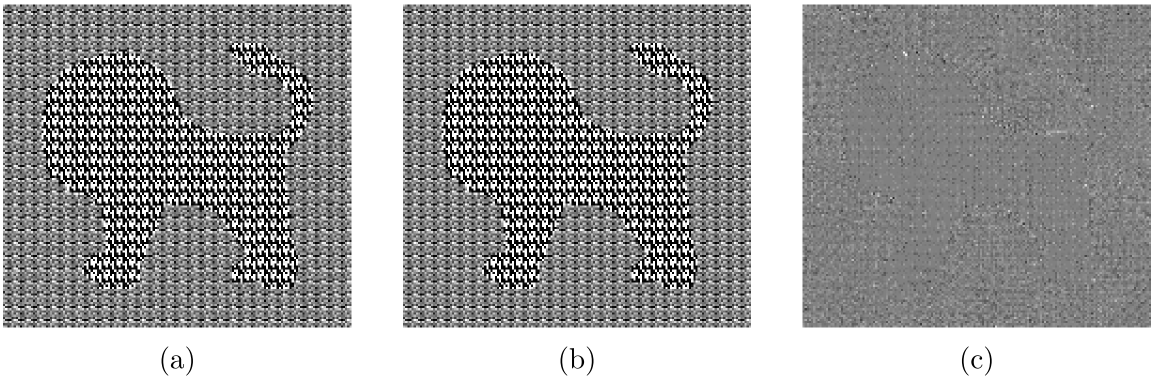

(a) (b) (c)

Figure 7.12: 2D Example of Non Local denoising Filter: Artificial Lion with Gaussian noise ($\sigma = 15$). (a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

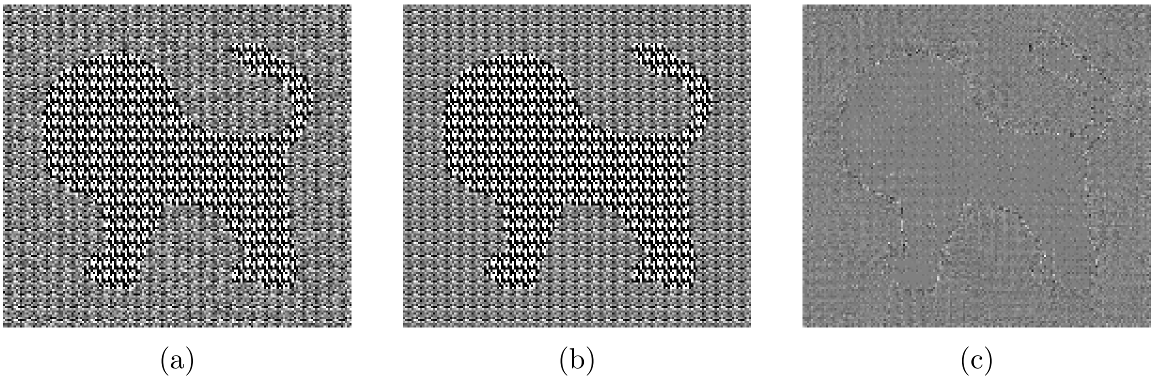

(a) (b) (c)

Figure 7.13: 2D Example of Non Local denoising Filter: Artificial Lion with Gaussian noise ($\sigma = 30$). (a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

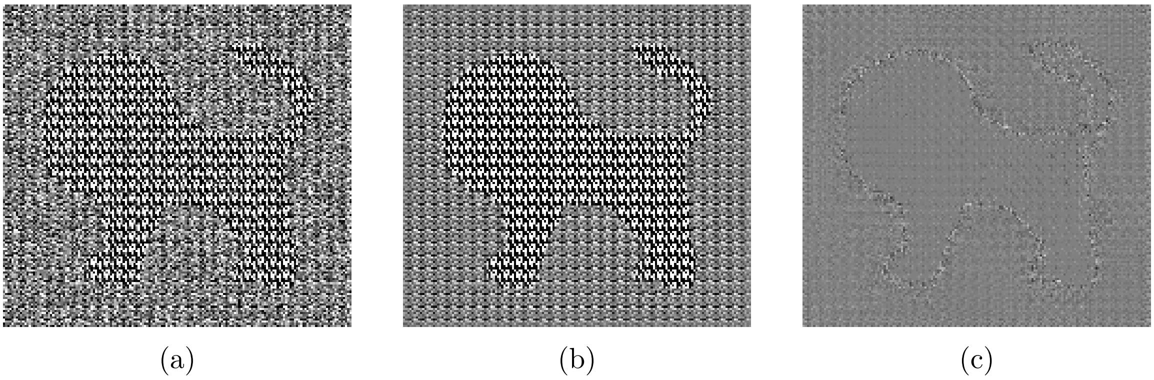


Figure 7.14: 2D Example of Non Local denoising Filter: Artificial Lion with Gaussian noise ($\sigma = 50$).
(a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

From this simulation of artificial images we can conclude that:

- In general, the quality of the denoising is good, we can see that most of the structure of the image is recovered completely, even with higher noise settings.
- The compromised structure is the one involved with the edges of objects, increasing the noise it becomes more notorious. Is important to notice that this structure is compromised in amplitude in a moderated way, since in the filtered image we can notice this structure anyways.

From this we can conclude that, at least in the case of artificial images, the non-local denoising filter has a great performance, even with a high amount of noise the filter is able to recover the original structure of the images.

7.1.2.3 Natural Example: Tiger

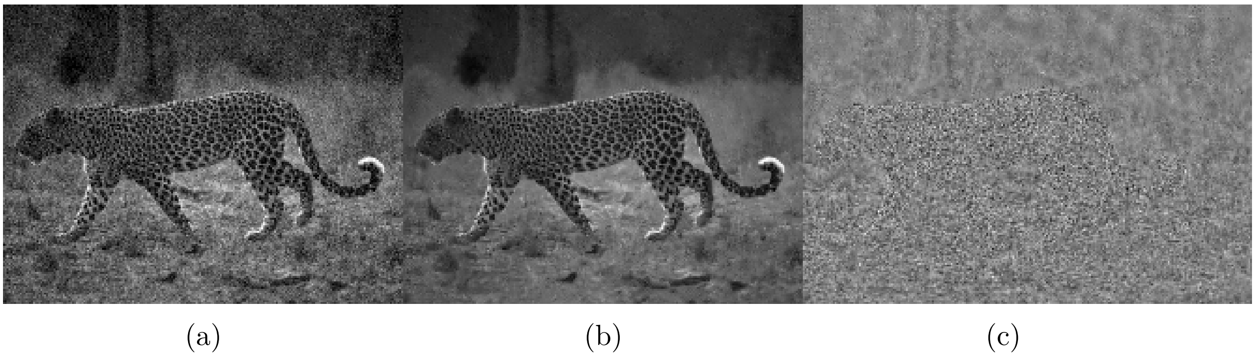


Figure 7.15: 2D Example of Non Local denoising Filter: Tiger with Gaussian noise ($\sigma = 15$).
(a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

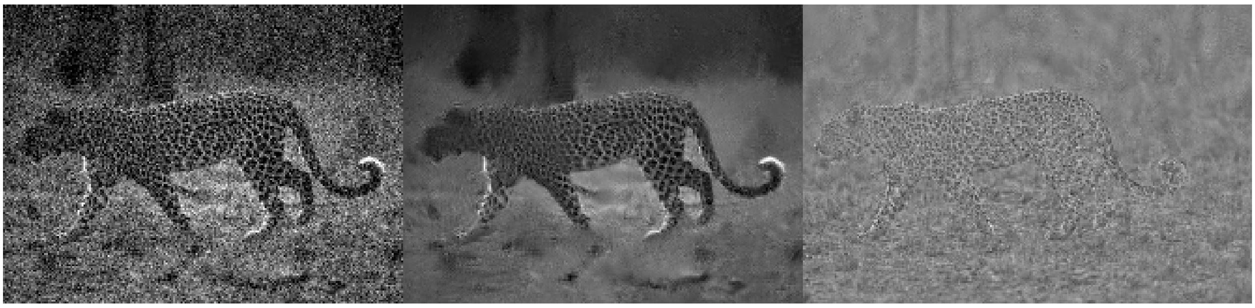


Figure 7.16: 2D Example of Non Local denoising Filter: Tiger with Gaussian noise ($\sigma = 30$). (a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

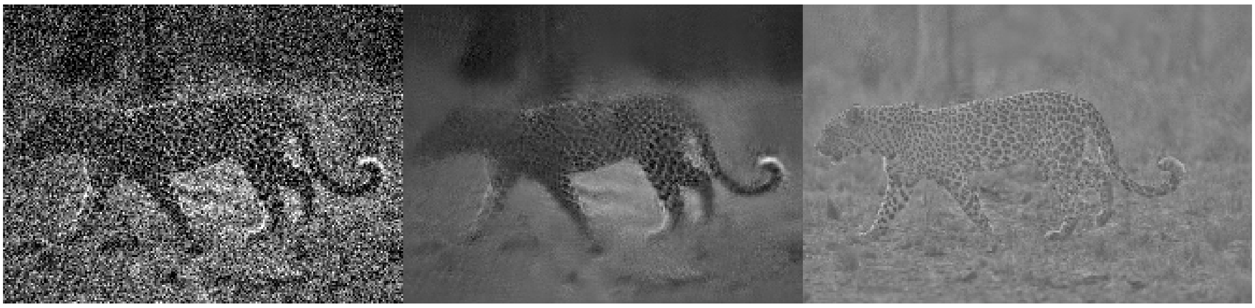


Figure 7.17: 2D Example of Non Local denoising Filter: Tiger with Gaussian noise ($\sigma = 50$). (a) Original Image with noise added. (b) Filtered Image. (c) Difference Image between Original input (without noise) and Filtered Image.

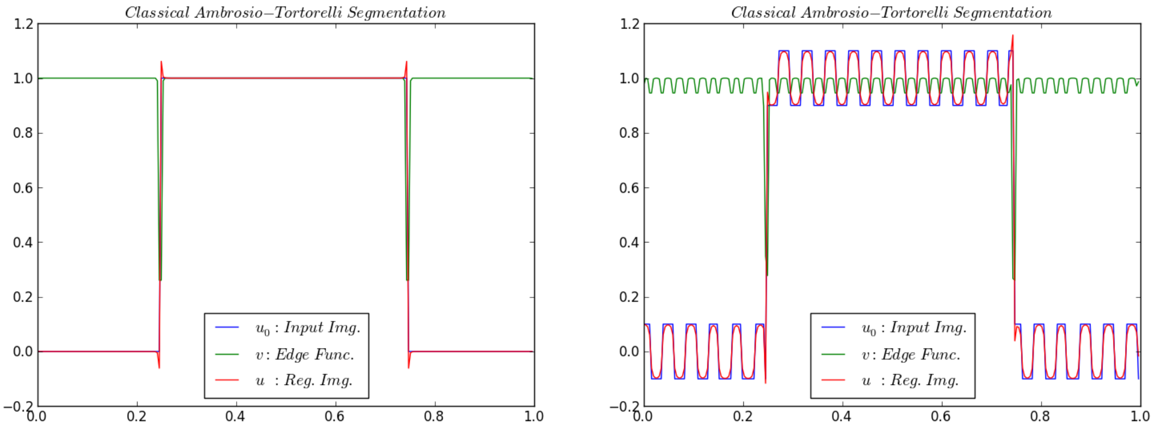
From this simulation of natural images we can conclude that:

- The denoising is good for low-noise images, it recovers the most part of image and regularize the boundary of objects (therefore, the amplitude of the image in the boundary is lower), background objects are still recognizable.
- The denoising is bad for high-noise images, we can see that in this case the 'principal object' is still recognizable, but almost all the background objects are lost, we also have a lot of blurring in the filtered image. The difference image let us see that basically 'the whole image details' are lost.

Finally, from the simulations we can conclude that the non-local denoising filter performs a high quality recovery of the image with controlled levels of noise, in high-noise situations the filter tends to lose all the background objects and introduce blurring in the denoised image, for artificial images the filter even works great in high-noise situations, and for real-enviroment images, with noise and texture it works successfully with reasonable levels of noise, in all this situations the filter tend to lose amplitude at the boundary of objects but the results are satisfactory.

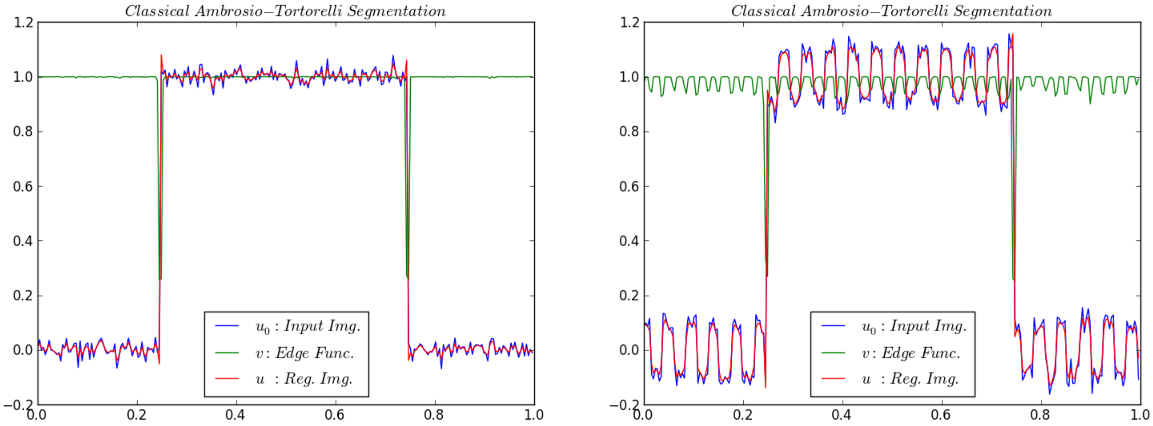
7.2 Segmentation using Ambrosio-Tortorelli Approximation

7.2.1 1-Dimensional Segmentation Examples



(a) (b)

Figure 7.18: 1D Example of Numerical Ambrosio Tortorelli Segmentation: Signals without Noise. (a) No Textured Signal. (b) Textured Signal.



(a) (b)

Figure 7.19: 1D Example of Numerical Ambrosio Tortorelli Segmentation: Noisy Signals. (a) No Textured Signal. (b) Textured Signal.

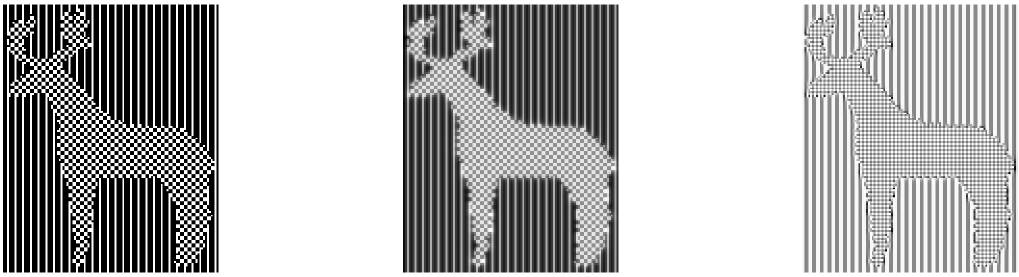
From this simulation of 1D signals we can conclude that:

- In non-noisy signals the segmentation and regularization gives good results, each discontinuity point is catched by the v function, anyway in the case of textured signal v detect the discontinuity but it didn't goes to zero as one could expect, this phenomena is not isolated and is well known from 2D image segmentation.

- In noisy signals the segmentation gives good results, the v function is very similar to the non-noisy v obtained before, the regularization is poor in the case without texture, we can see that regularized version is very 'sharp'. In the case with texture the regularization is a little better, the regularized version in this case don't have too much edges, but is not 'optimal', considering the original signal.

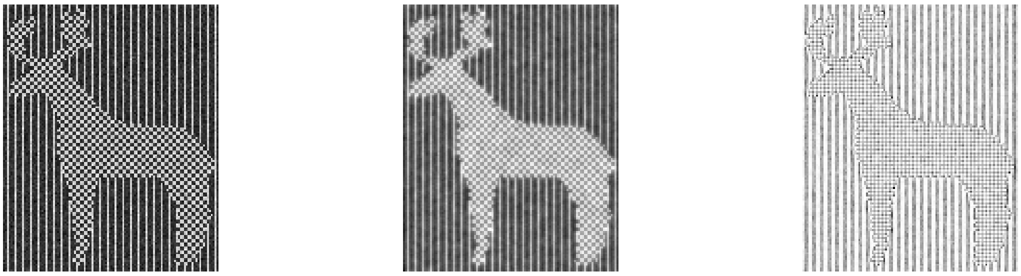
Although this is a very well known process we prefer to perform our own testing and we can conclude that this task allows to get well regularized and segmented signal in the case of non-noisy signals, and in the case of signals with noise we get some problems in the regularized output, also, in the case of texture, the v function does not recognize very well the discontinuities due to the texture in the sense that this 'edge set function' didn't goes to zero as one could expect.

7.2.2 2-Dimensional Segmentation Examples



(a) (b) (c)

Figure 7.20: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Artificial Moose without Noise. (a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).



(a) (b) (c)

Figure 7.21: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Artificial Moose with Noise. (a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

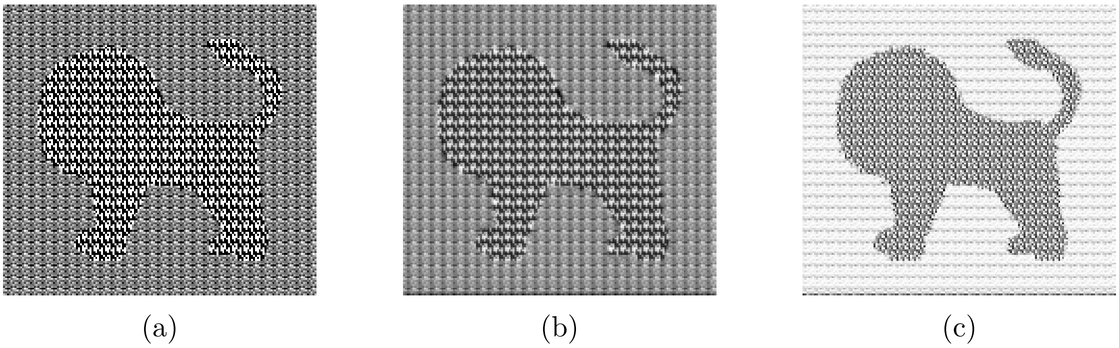

(a) (b) (c)

Figure 7.22: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Artificial Lion without Noise. (a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

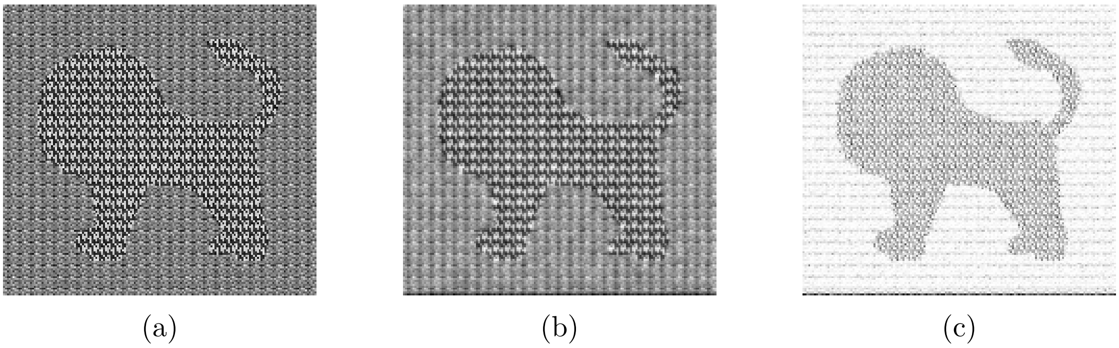

(a) (b) (c)

Figure 7.23: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Artificial Lion with Noise. (a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

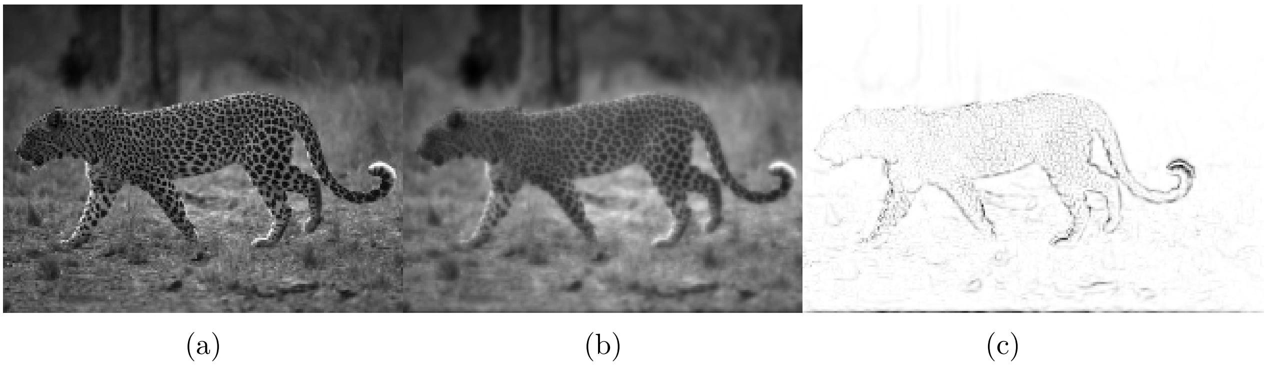

(a) (b) (c)

Figure 7.24: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Artificial Tiger. (a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

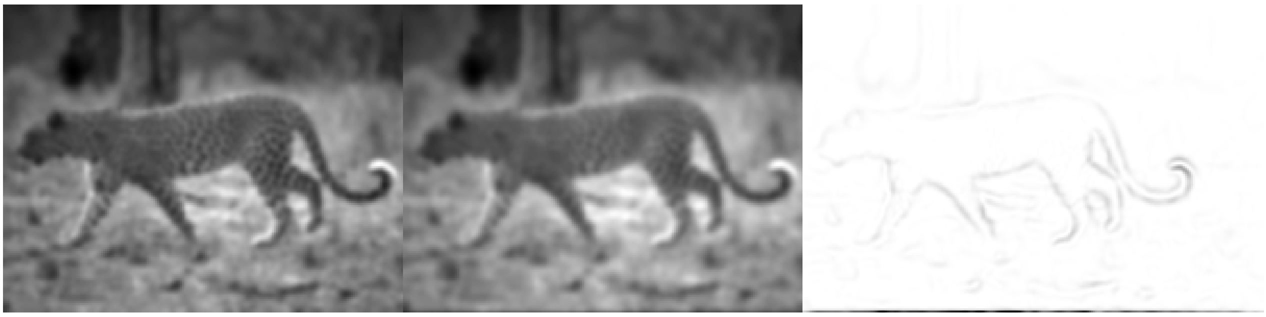


Figure 7.25: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Tiger with blur. (a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

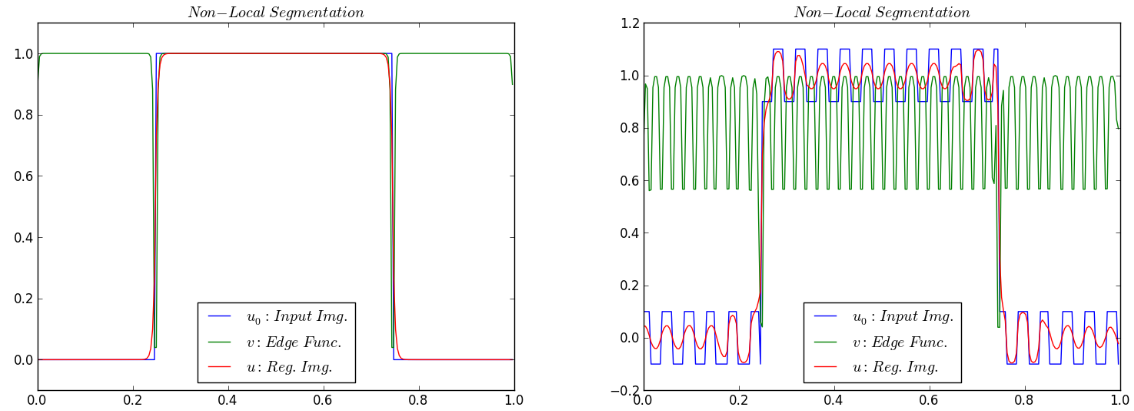
From this simulation of 2D signals we can conclude that:

- In artificial images, in noisy and non-noisy images the segmentation process didn't perform the segmentation in the sense we look for: The desired behaviour is to detect the object 'hidden' in the texture, unfortunately in this case the segmentation process only detect the change of intensity on the texture, no object is segmented at all.
- In the natural image, the segmentation process results are very good in the natural image without noise, the edge set function v is able to capture the main object and some details of the texture, anyway some background objects are lost and a big part of texture is also lost, the regularized image have good quality, with a little blurring. Things become worse on the noisy (in this case blurry) image, the set function v poorly get the boundary of the main object and the regularized image is more blurry than the original version.

Therefore, from this simulations we can conclude that Ambrosio-Tortorelli segmentation is suitable for signals and images without noise, we can recover the most important features of the images and the edge set function v recover the boundaries of the objects and also of the textures, in general v takes values not closer to zero as one could want. On the other hand, this segmentation is poor in noisy images, specially in noisy images with texture, this kind of segmentation is unable to detect texture properly, the edge set function v detect only main boundaries and the regularization is very poor. Unfortunately this segmentation process is unable to 'detect' objects 'hidden' in a textured environment, the process forces to segment each structure locally, therefore it segment each component of the texture independently.

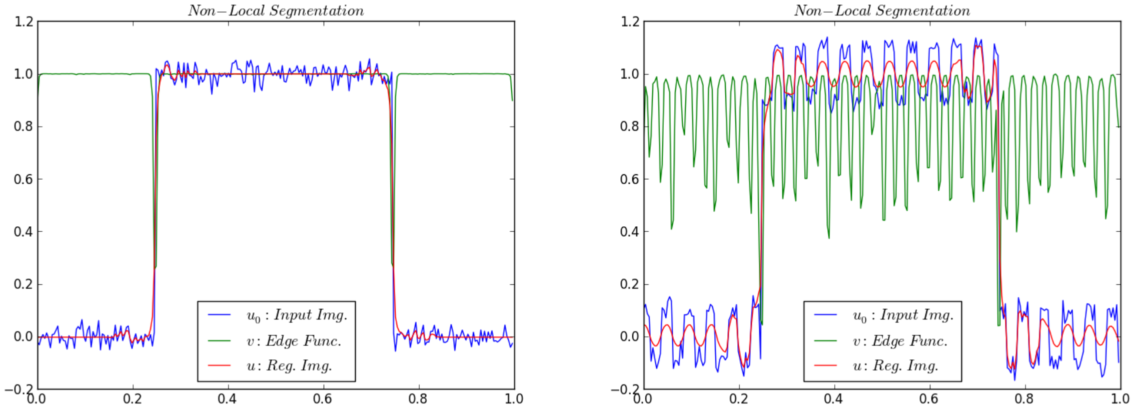
7.3 Non Local Segmentation

7.3.1 1-Dimensional Segmentation Examples



(a) (b)

Figure 7.26: 1D Example of Numerical Non Local Segmentation: Signals without Noise. (a) No Textured Signal. (b) Textured Signal.



(a) (b)

Figure 7.27: 1D Example of Numerical Non Local Segmentation: Noisy Signals. (a) No Textured Signal. (b) Textured Signal.

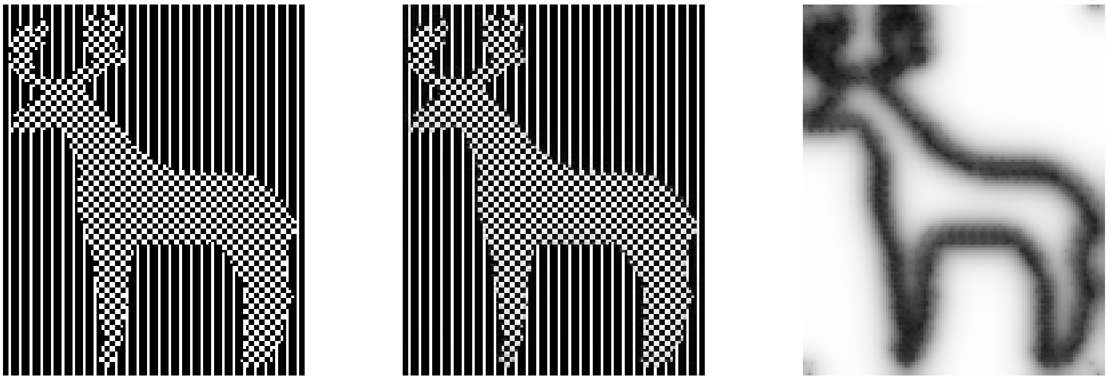
From this simulation of 1D signals we can conclude that:

- In the ‘basic examples’ (the ones without noise) we get similar results as for the classical segmentation, in the examples without noise we get almost identical results for regularized signals, excepting for the textured signal in which the regularized version also have lost amplitude in the texture, anyway, the edge set function v recognize better the discontinuities associated with the texture, in the sense that in this case the values of v are closer to zero compared to the original model.

- In noisy examples the results are much more better, the regularized version is much better than the obtained by the Ambrosio-Tortorelli segmentation, it is very similar to the original signal, we're still losing amplitude in the texture (which seems a lot like the non local filtering that we have seen before), anyway the results are much more better in texture recognition on edge set function v in the case of textured images.

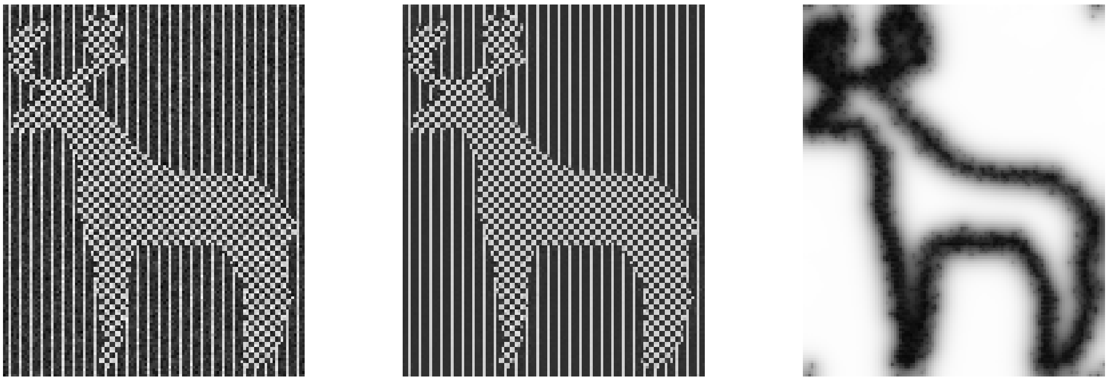
Therefore, we can see that this segmentation have some features that classical Ambrosio-Tortorelli don't have: It recognize better textured signals and performs the regularization better than the original model, also the edge set function v recognize better the discontinuities in the sense that the function is more closer to zero in the discontinuities than the Ambrosio-Tortorelli segmentation. However it have some flaws like the 'lose of amplitude' in the textured zone, this is specially relevant in signals without noise, in this case the original Ambrosio-Tortorelli segmentation performs better.

7.3.2 2-Dimensional Segmentation Examples



(a) (b) (c)

Figure 7.28: 2D Example of Non Local Segmentation: Artificial Moose without Noise.
(a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).



(a) (b) (c)

Figure 7.29: 2D Example of Non Local Segmentation: Artificial Moose with Noise.
(a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

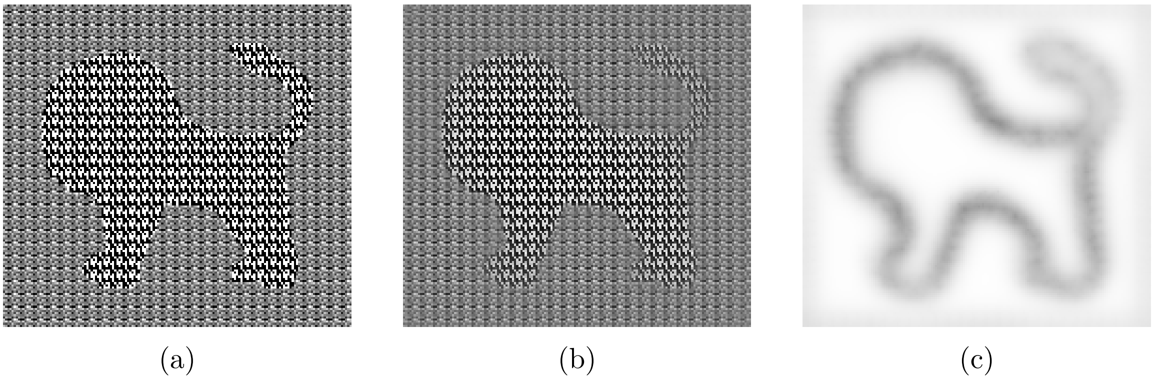


Figure 7.30: 2D Example of Non Local Segmentation: Artificial Lion without Noise.
(a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

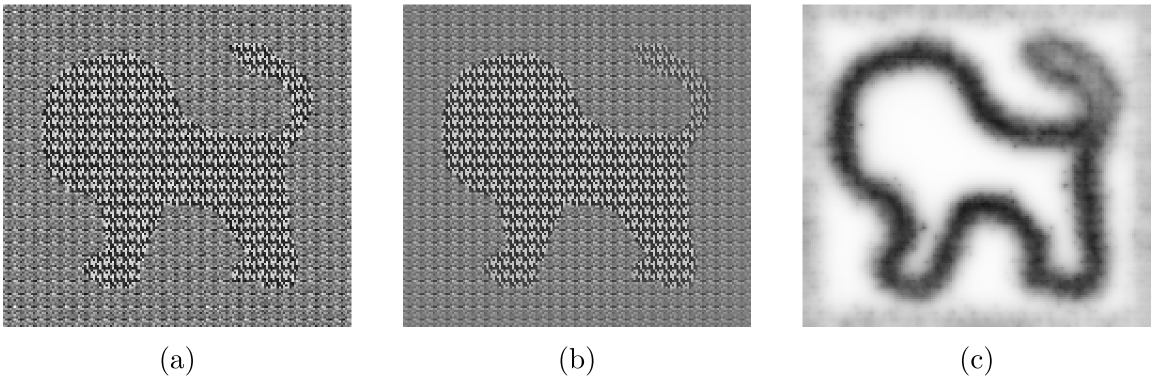


Figure 7.31: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Artificial Lion with Noise.
(a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

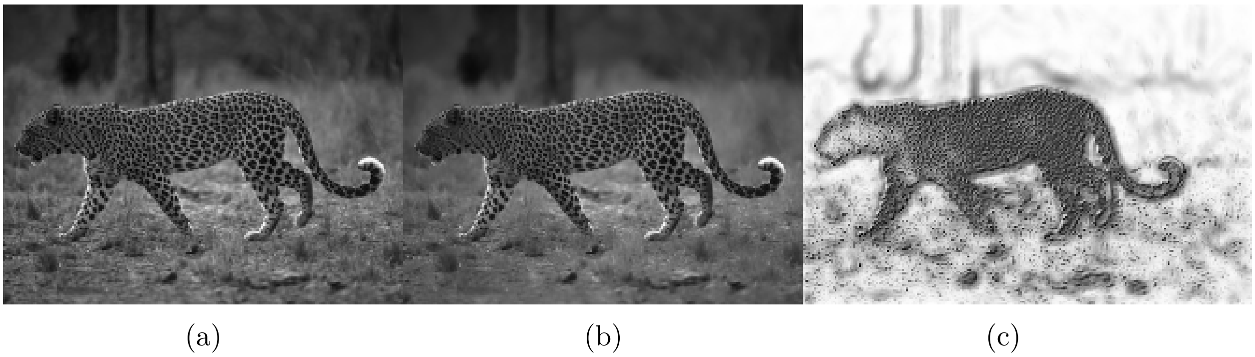


Figure 7.32: 2D Example of Non Local Segmentation: Artificial Tiger.
(a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

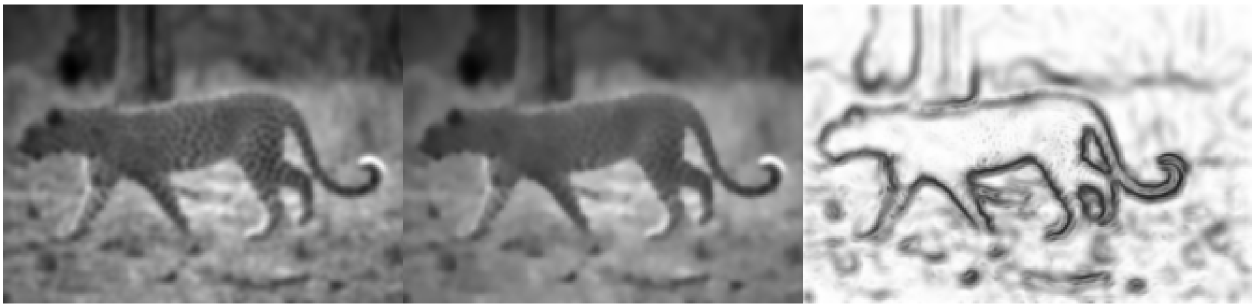


Figure 7.33: 2D Example of Classical Ambrosio-Tortorelli Segmentation: Tiger with blur. (a) Original Image. (u_0). (b) Regularized Image. (u). (c) Segmentated Image (set function v).

From this simulation of 2D signals we can conclude that:

- The segmentation in the case of artificial images brings very good results in all cases, the segmentation is as expected: The 'hidden' object is properly segmented, the non-local comparison between 'image patches' allows the process to recognize the similarity of the repeated texture and only segment the transition zone when the texture changes. The regularization is good excepting for some lose small structures (specially in more noisy examples) which tend to be vanished in the background.
- In the natural example we get an excellent segmentation results also: the main object is completely segmented, also its own texture is segmented, various objects of the environment are segmented, specially the ones in the background. The regularization is very good in the 'clear' image where we only get a small (like on the grass and rocks) vanished structures.
- In the blurry example we still get the main object segmented, also we can still segment the main objects of the background, unfortunately the regularization increases the blurring and the texture is not segmented anymore.

Finally, we can conclude that this non-local segmentation procedure have several advantages over the classical Ambrosio-Tortorelli segmentation, it brings much better results segmentating objects which are embedded in a textured environment, it also brings much better results when dealing with images in natural environment and in 1 dimension it is able to segmentate and regularize the texture in this case. It is important to recall that in 1D we used a different functional that in 2D case, however, in both cases we are dealing with non-local models, so we can state that non-local segmentation functionals perform better than the local ones.

7.4 Gabor Functions for Texture recognition

This numerical testing will be divided in two parts:

- In the first round of examples we run the optimization routine for the basis pursuit problem in order to obtain the approximation of the signal and division of the approximated signal in different levels of resolution for the family of complex exponential spline functions given by:

  $$\Psi_{k,p}(x) = \cos(\pi/2 \cdot (x-p))e^{i\pi/2\cdot k\cdot(x-p)} \cdot \mathbf{1}(|x-p| \leq 1/k)$$

  with $k = 2^2, 2^3, \ldots, 2^8$, $p = p(k) = 0, 1/k, \ldots, (k-1)/k$.

  We also compute this approximation with shifted signals, the purpose of this computation is to estimate how much the associated coefficients change between the original signal and the shifted signal.

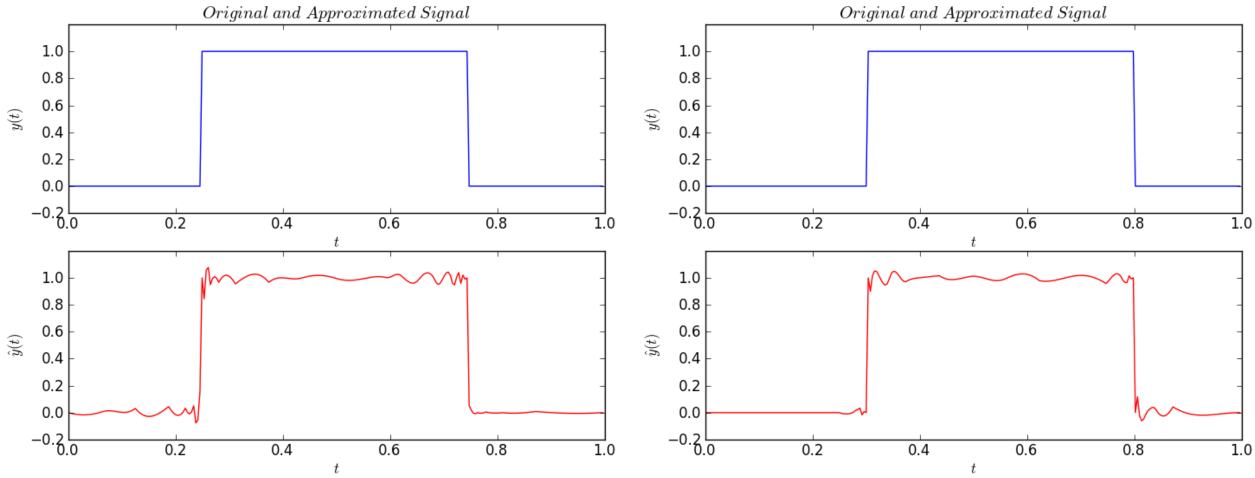
- In the second round of examples we run the optimization routine just with the family of the real part of the functions, i.e., with the family of the functions given by:

  $$\phi_{k,p}(x) = \Re\left(\Psi_{k,p}(x)\right) = \cos(\pi/2 \cdot (x-p)) \sin(\pi/2 \cdot k \cdot (x-p)) \cdot \mathbf{1}(|x-p| \leq 1/k)$$

  where $\Re(z)$ denotes the real part of the complex number z, in this case we also consider $k = 2^2, 2^3, \ldots, 2^8$, $p = p(k) = 0, 1/k, \ldots, (k-1)/k$.

The idea of this division is to have a comparative example for the variation of the coefficients due to the shift of a given signal, we want to see that our complex functions led to have a more 'stable' approximation when we shift a signal than the functions considered when we don't introduce a complex counterpart, like Kingsbury have done in his work about complex Wavelets in [36].

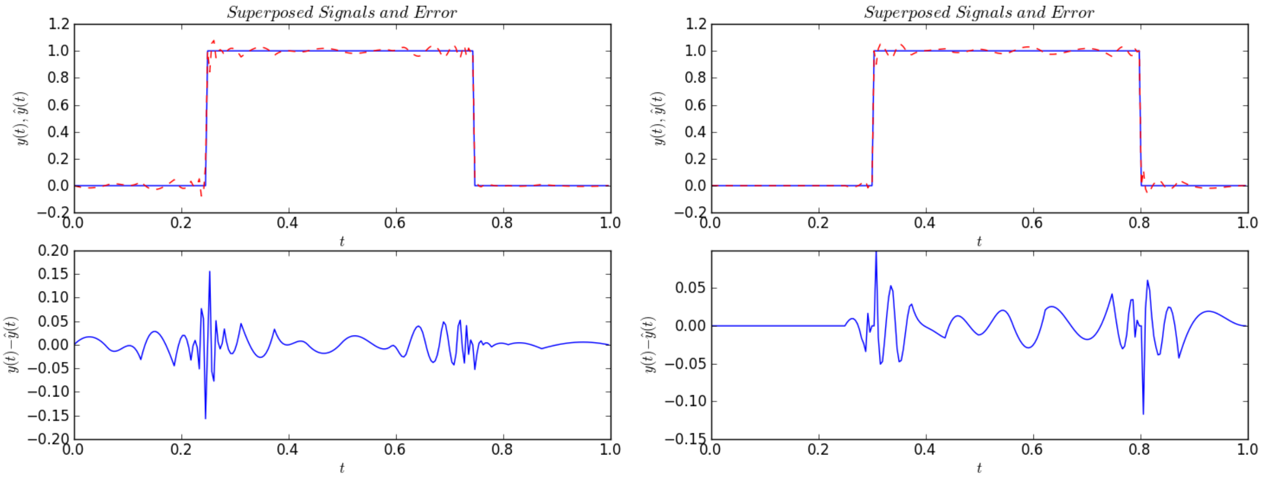
7.4.1 Approximation and Decomposition for No Texture-No Noise Image: Complex Exponential Spline



(a) Original Signal data

(b) Shifted Signal data

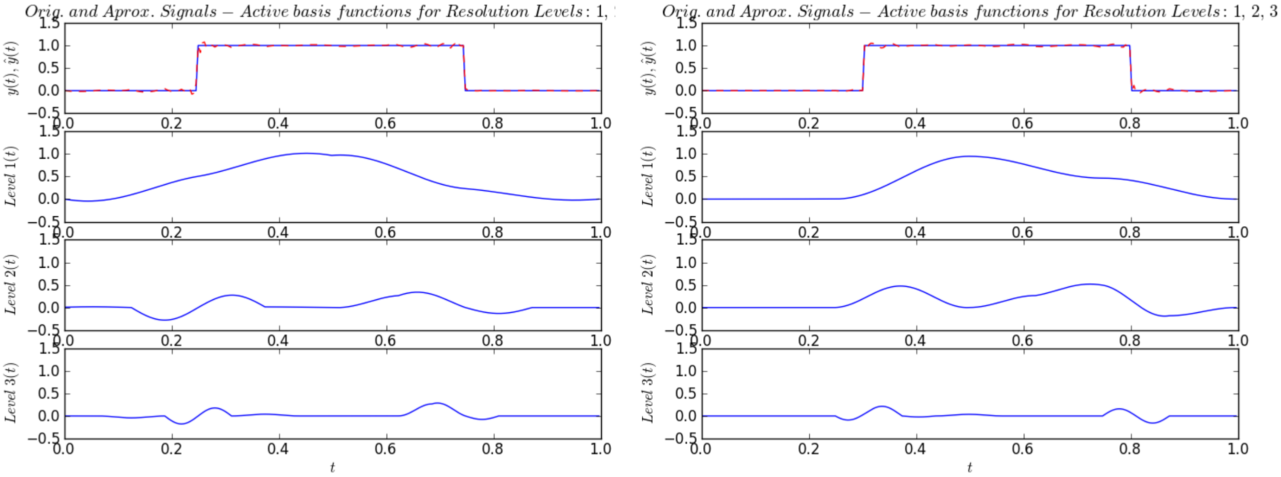
Figure 7.34: Original and Approximated Signal - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

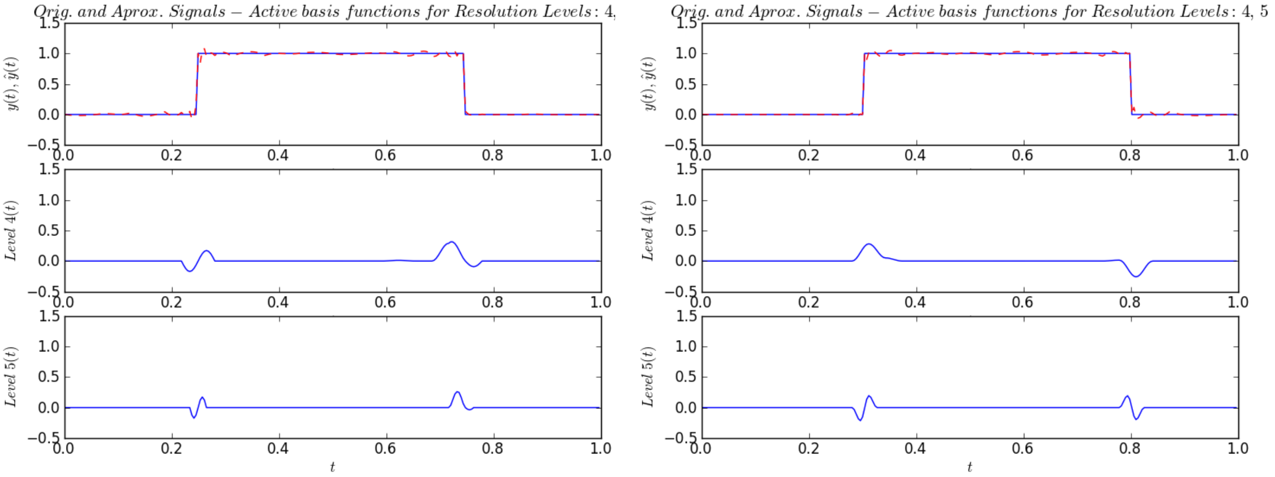
Figure 7.35: Superposed Signals and Error - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

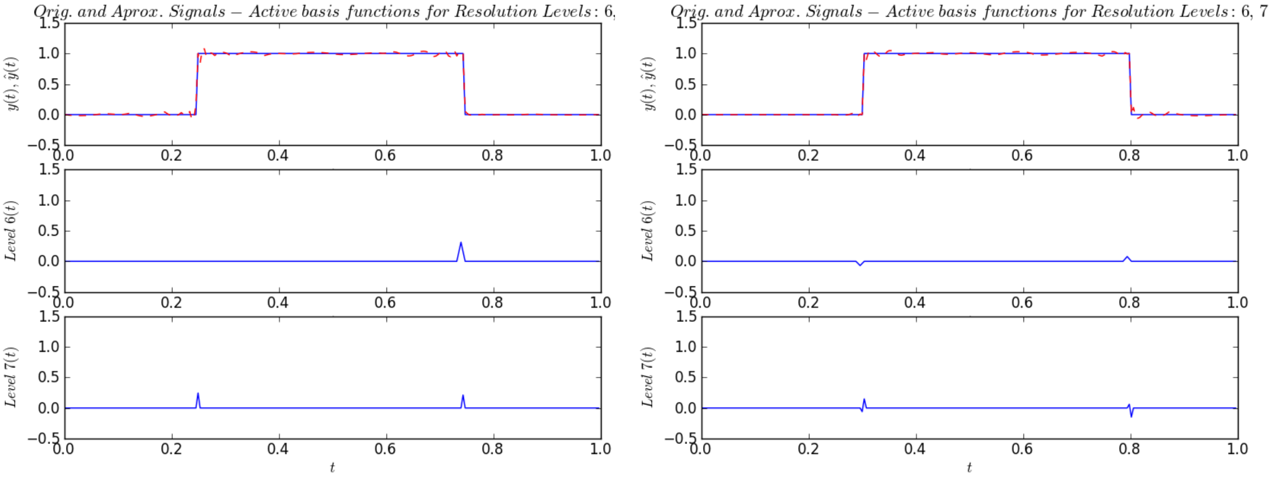
Figure 7.36: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

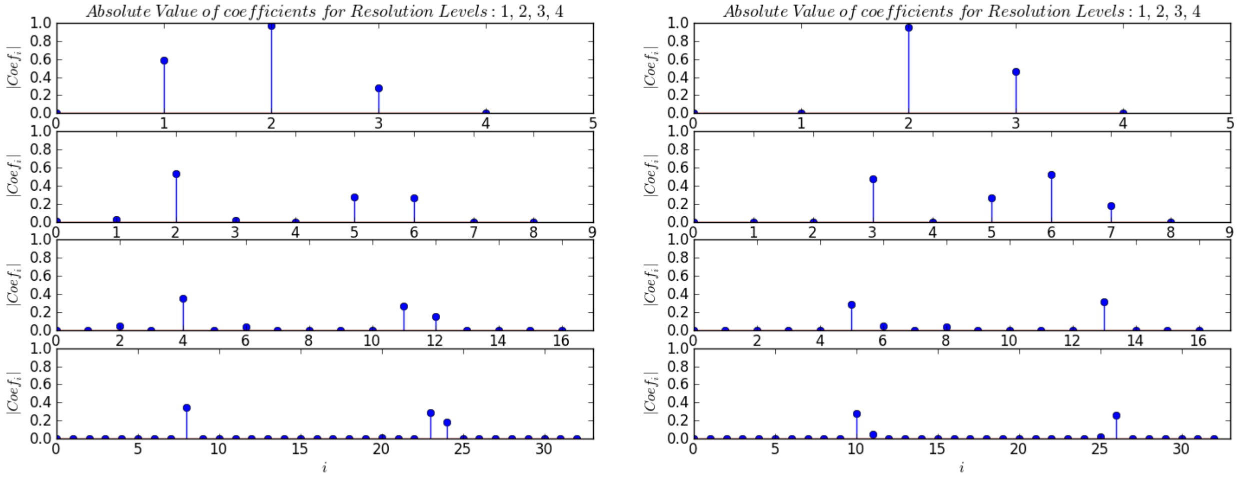
Figure 7.37: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

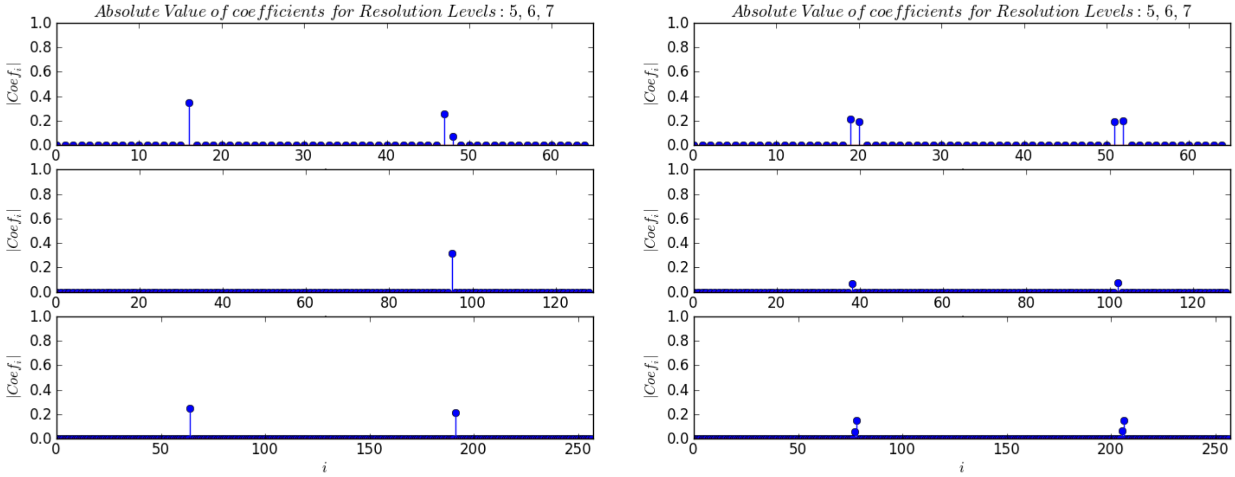
Figure 7.38: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

Figure 7.39: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal without Texture and without Noise

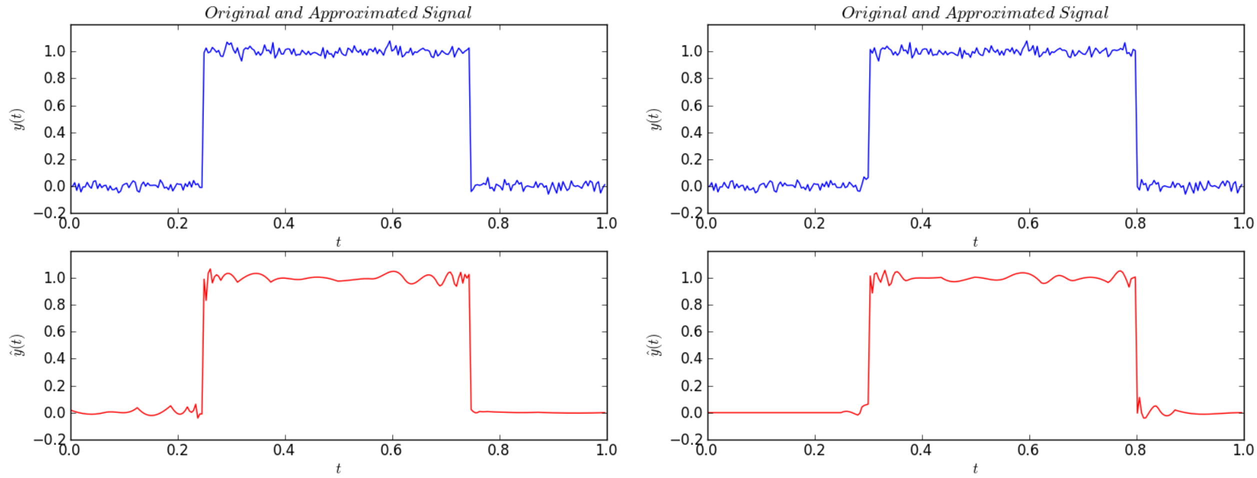


(a) Original Signal data

(b) Shifted Signal data

Figure 7.40: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal without Texture and without Noise

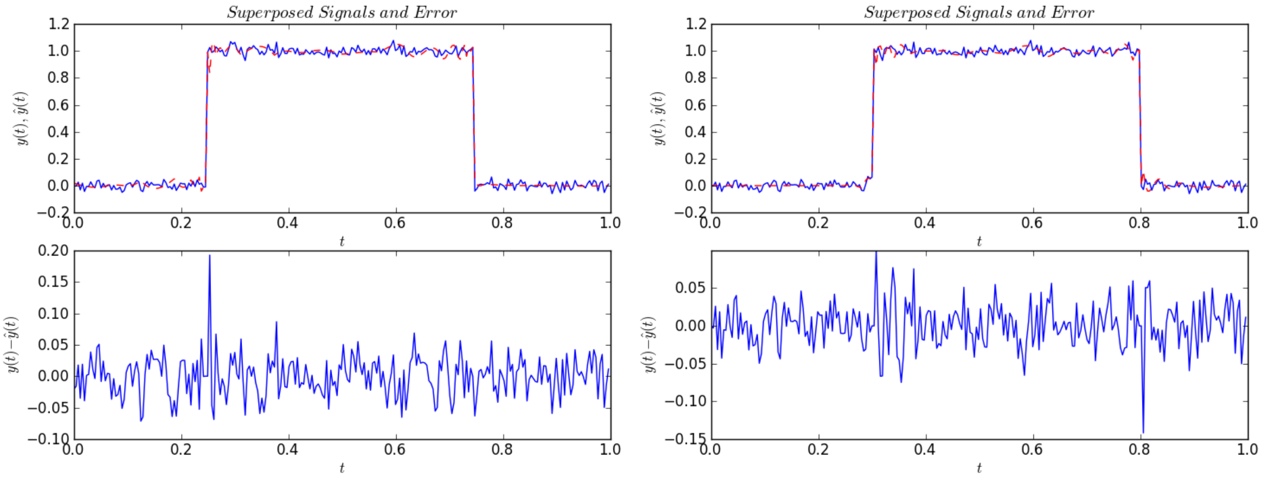
7.4.2 Approximation and Decomposition for No Texture-Noisy Image: Complex Exponential Spline



(a) Original Signal data

(b) Shifted Signal data

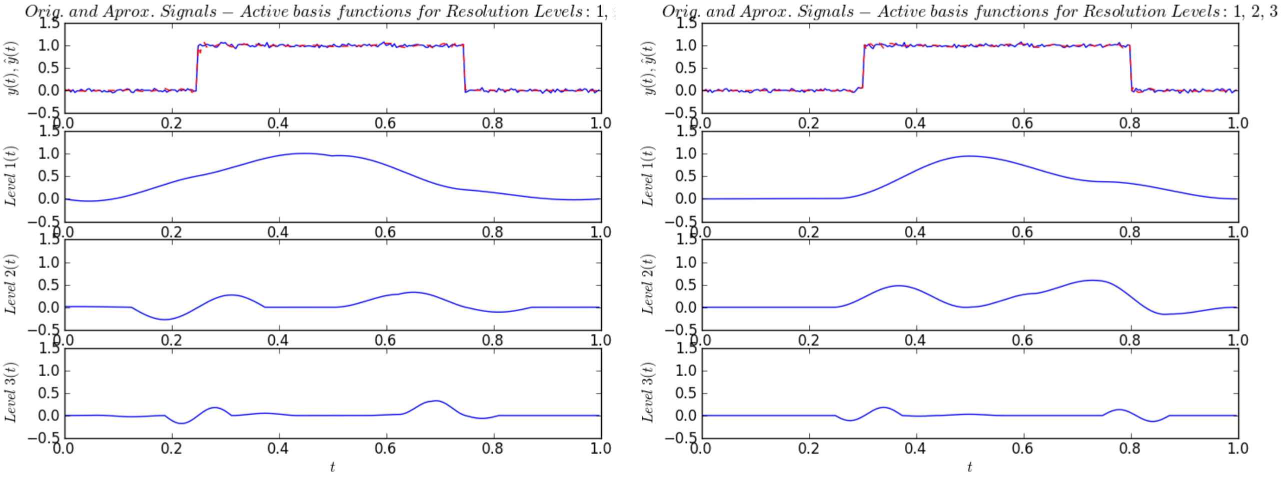
Figure 7.41: Original and Approximated Signal - Input Signal with Noise but without Texture



(a) Original Signal data

(b) Shifted Signal data

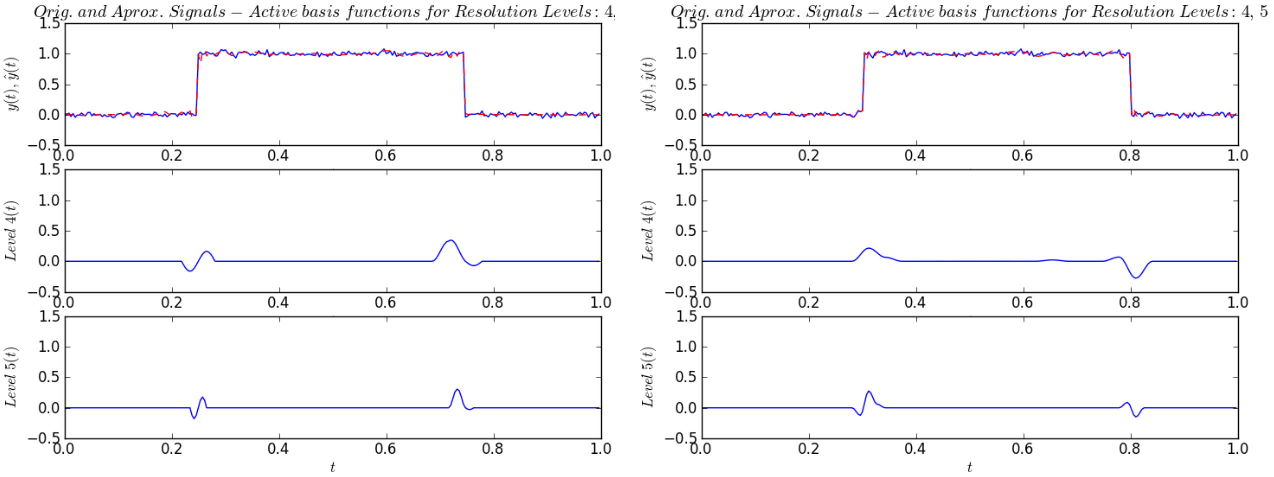
Figure 7.42: Superposed Signals and Error - Input Signal with Noise but without Texture



(a) Original Signal data

(b) Shifted Signal data

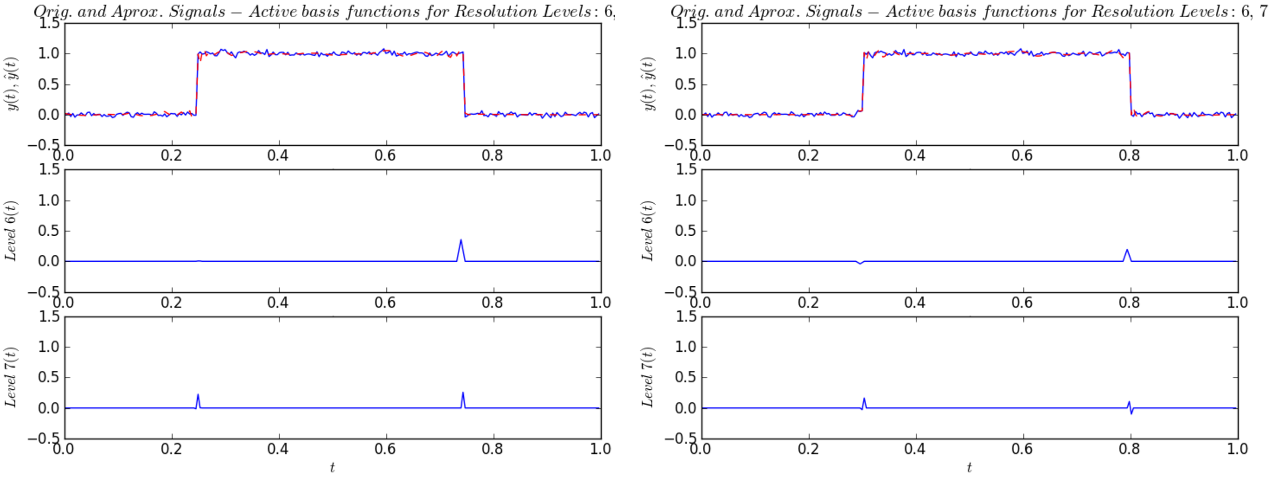
Figure 7.43: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal with Noise but without Texture



(a) Original Signal data

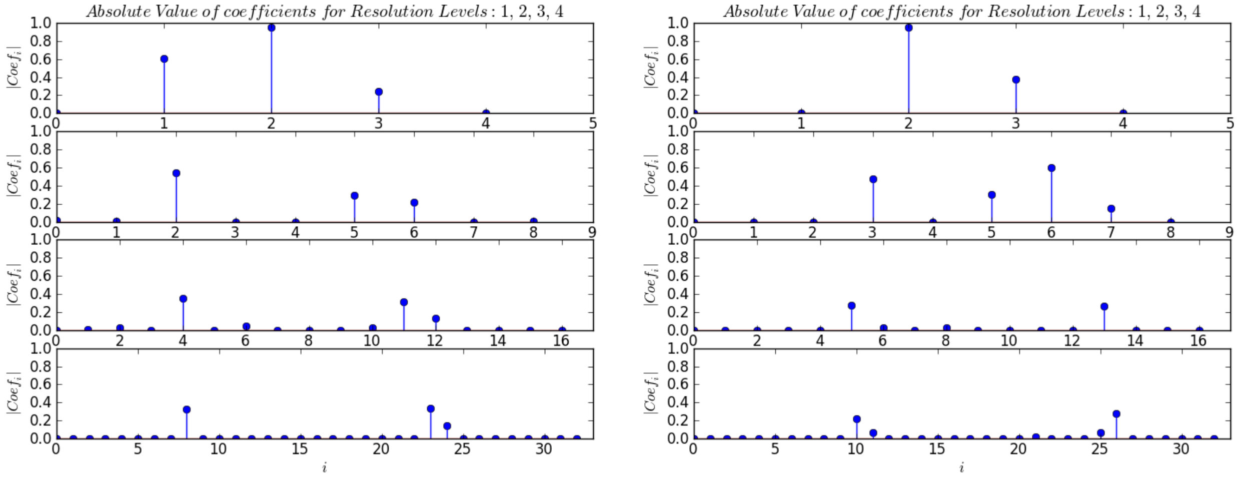
(b) Shifted Signal data

Figure 7.44: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal with Noise but without Texture



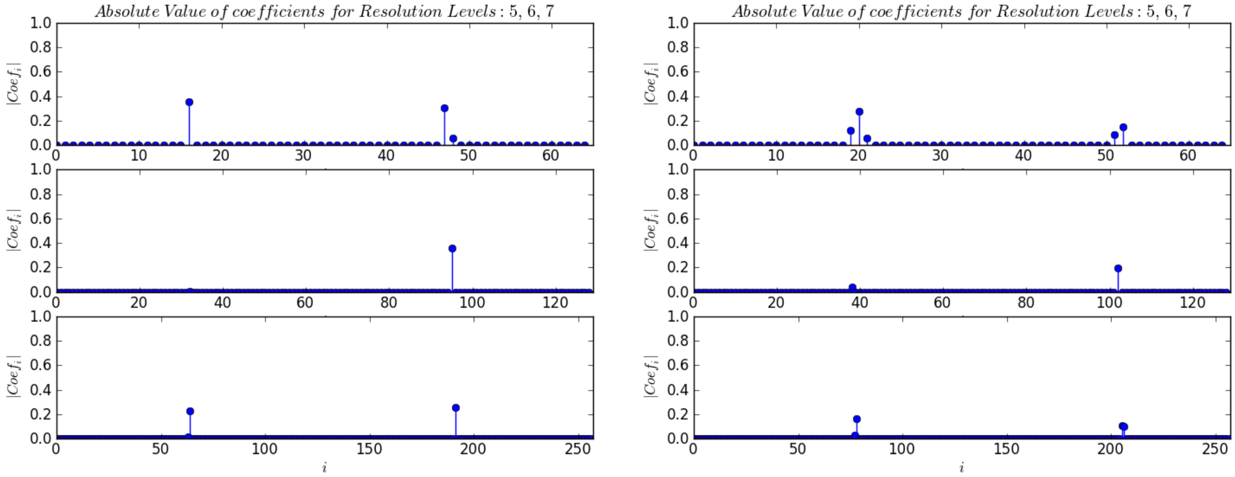
(a) Original Signal data (b) Shifted Signal data

Figure 7.45: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal with Noise but without Texture



(a) Original Signal data (b) Shifted Signal data

Figure 7.46: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal with Noise but without Texture

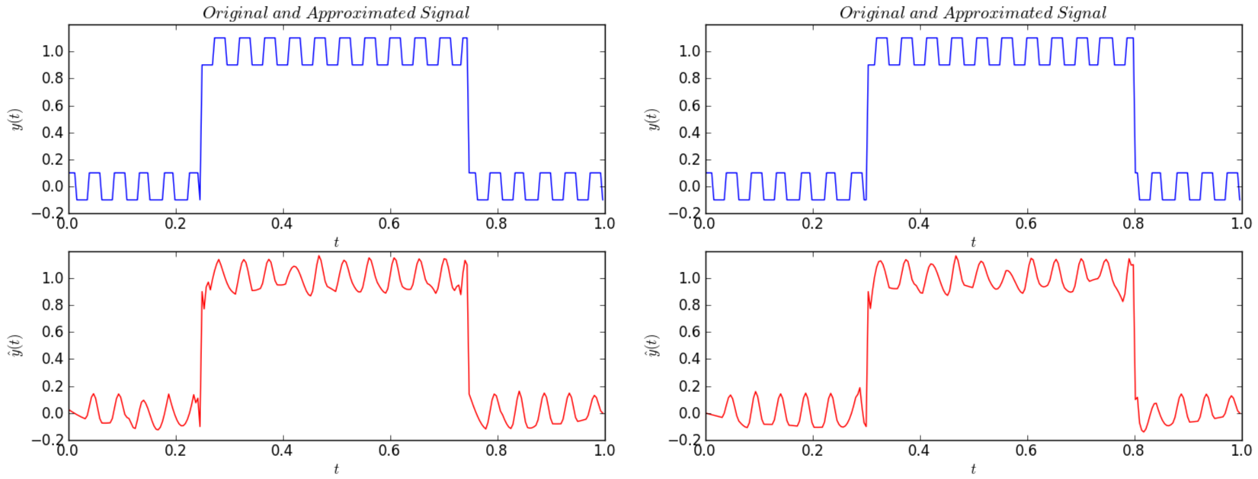


(a) Original Signal data

(b) Shifted Signal data

Figure 7.47: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal with Noise but without Texture

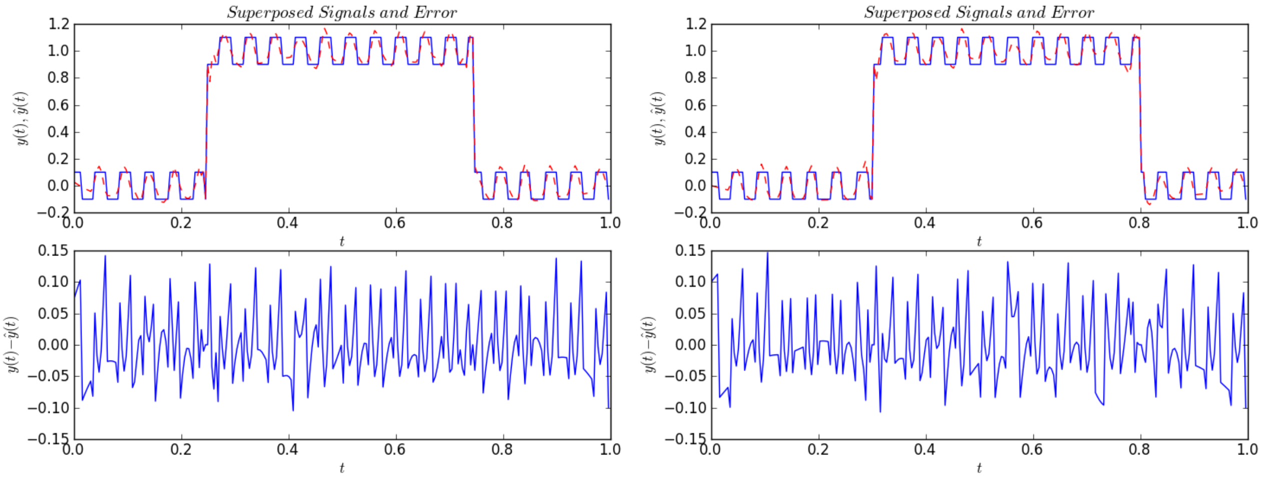
7.4.3 Approximation and Decomposition for Texture-No Noise Image: Complex Exponential Spline



(a) Original Signal data

(b) Shifted Signal data

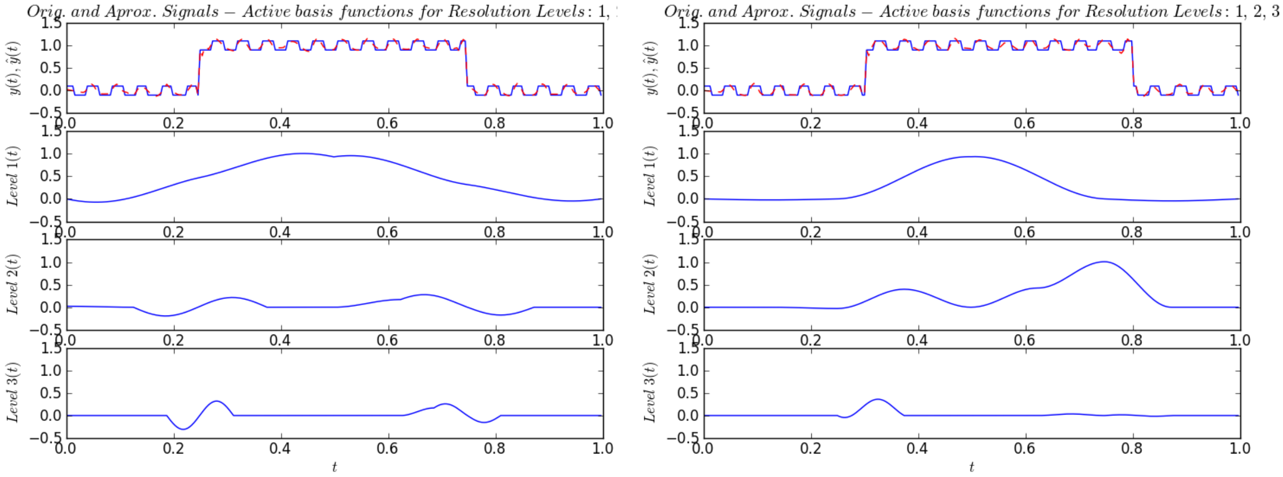
Figure 7.48: Original and Approximated Signal - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

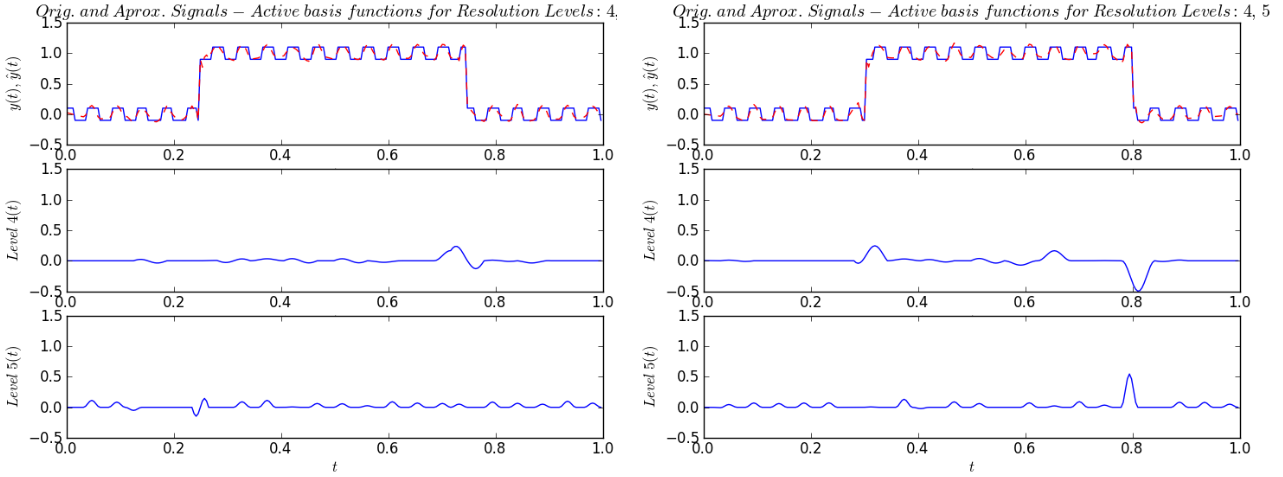
Figure 7.49: Superposed Signals and Error - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

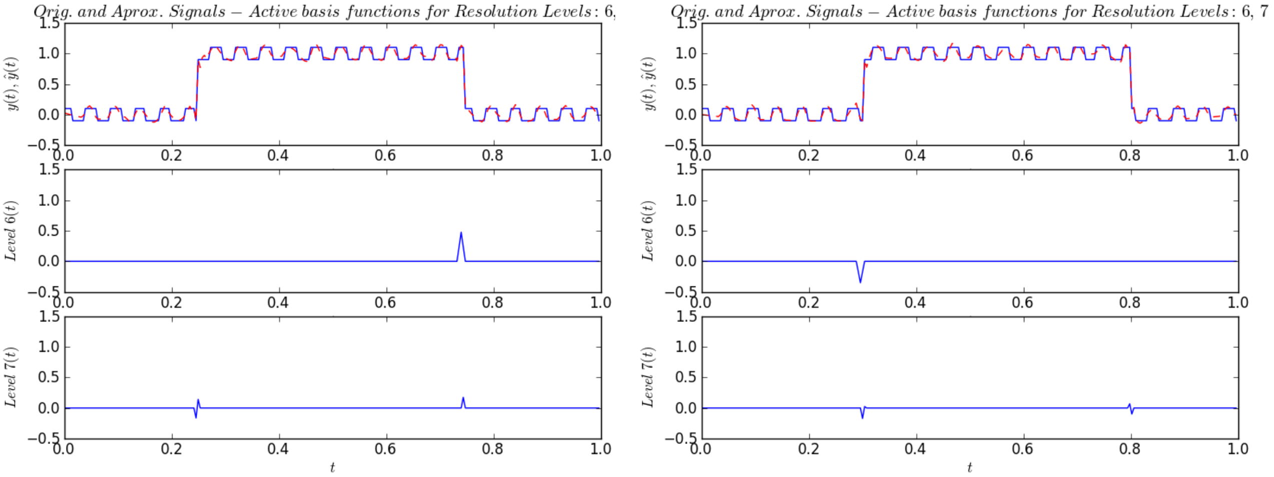
Figure 7.50: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal with Texture and without Noise



(a) Original Signal data

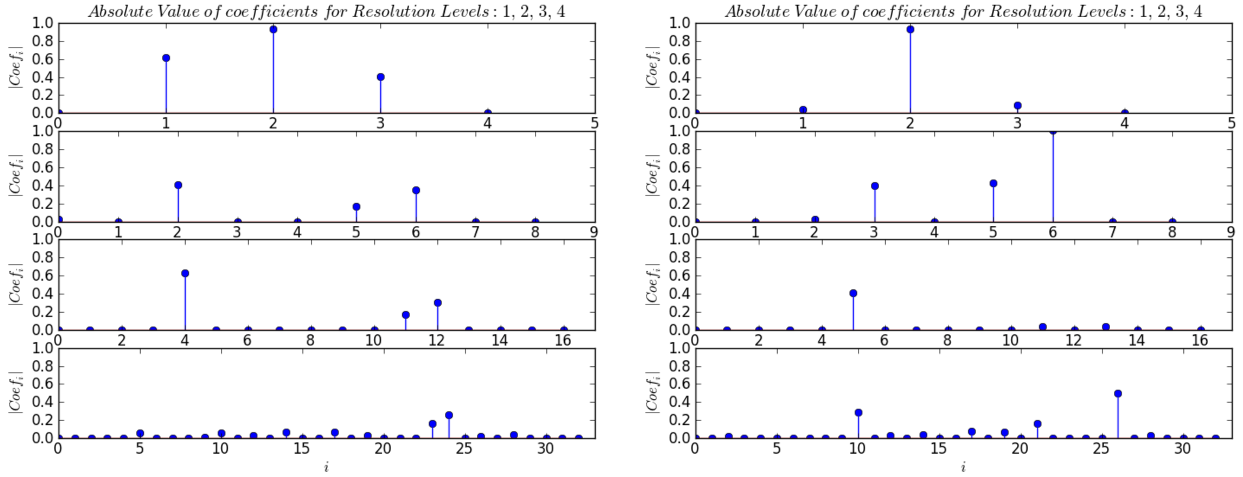
(b) Shifted Signal data

Figure 7.51: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal with Texture and without Noise



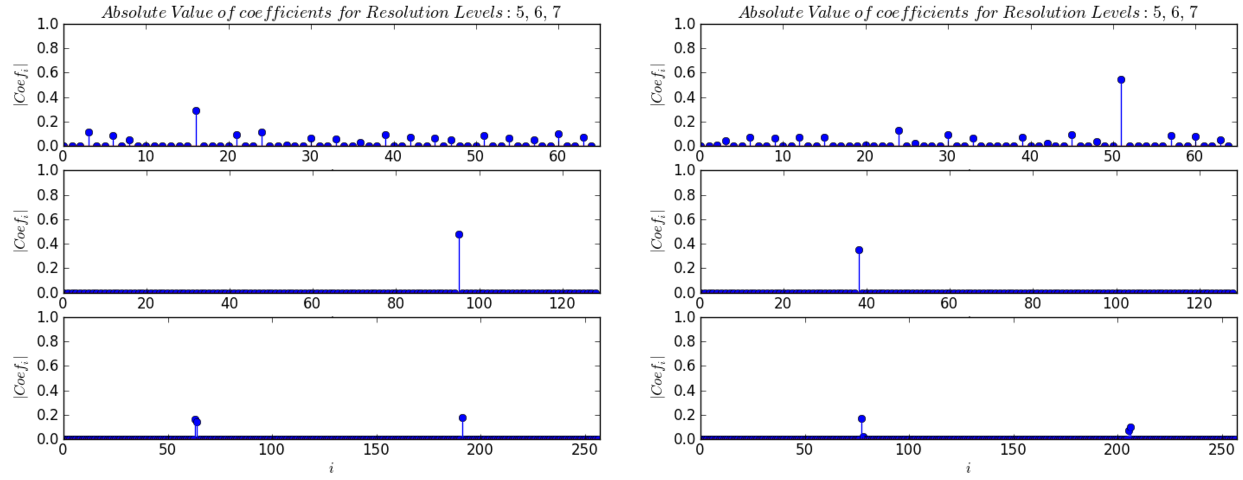
(a) Original Signal data (b) Shifted Signal data

Figure 7.52: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal with Texture and without Noise



(a) Original Signal data (b) Shifted Signal data

Figure 7.53: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal with Texture and without Noise

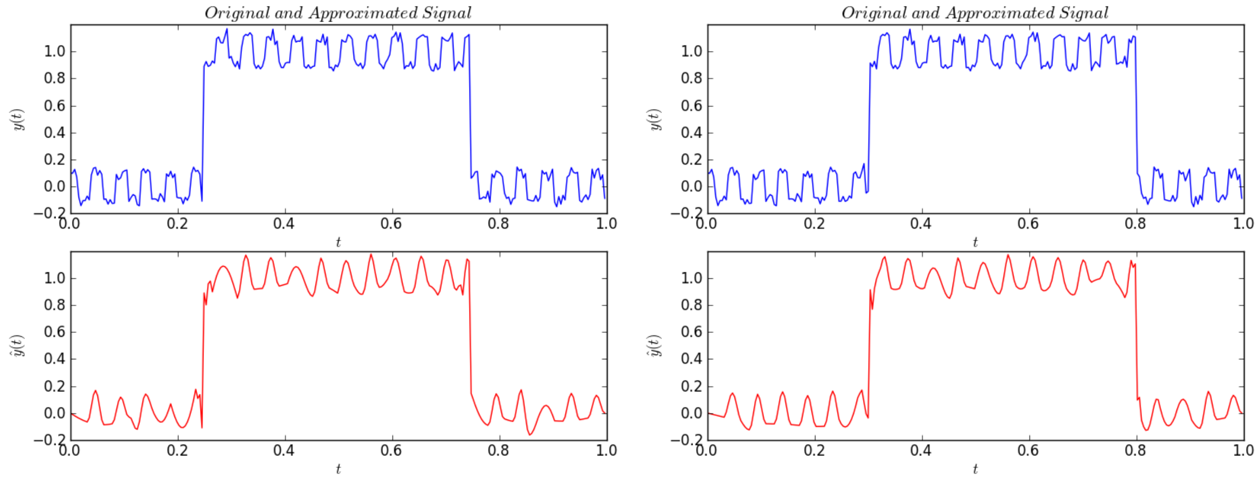


(a) Original Signal data

(b) Shifted Signal data

Figure 7.54: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal with Texture and without Noise

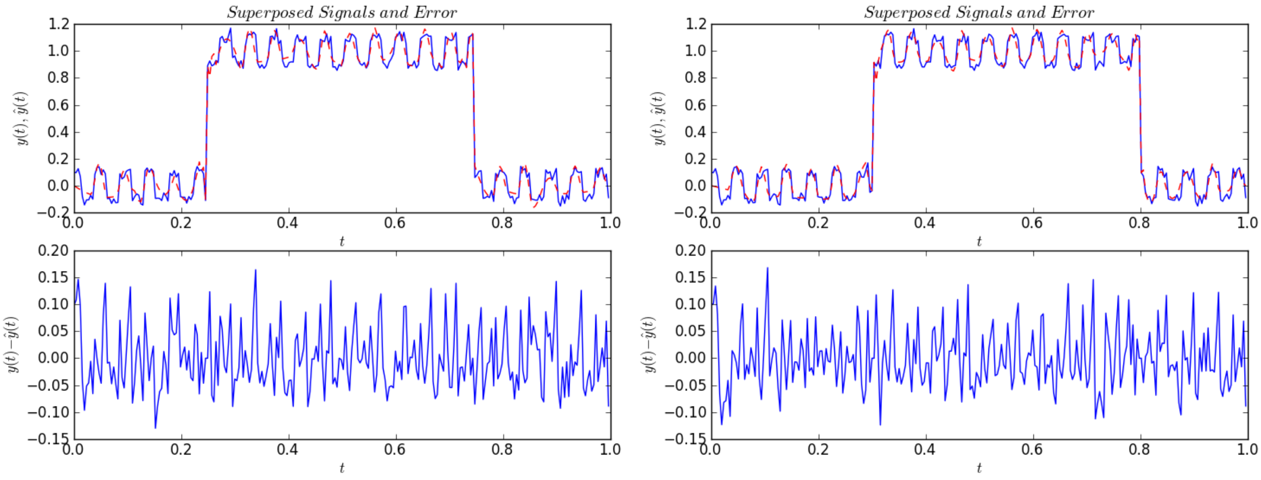
7.4.4 Approximation and Decomposition for Texture-Noise Image: Complex Exponential Spline



(a) Original Signal data

(b) Shifted Signal data

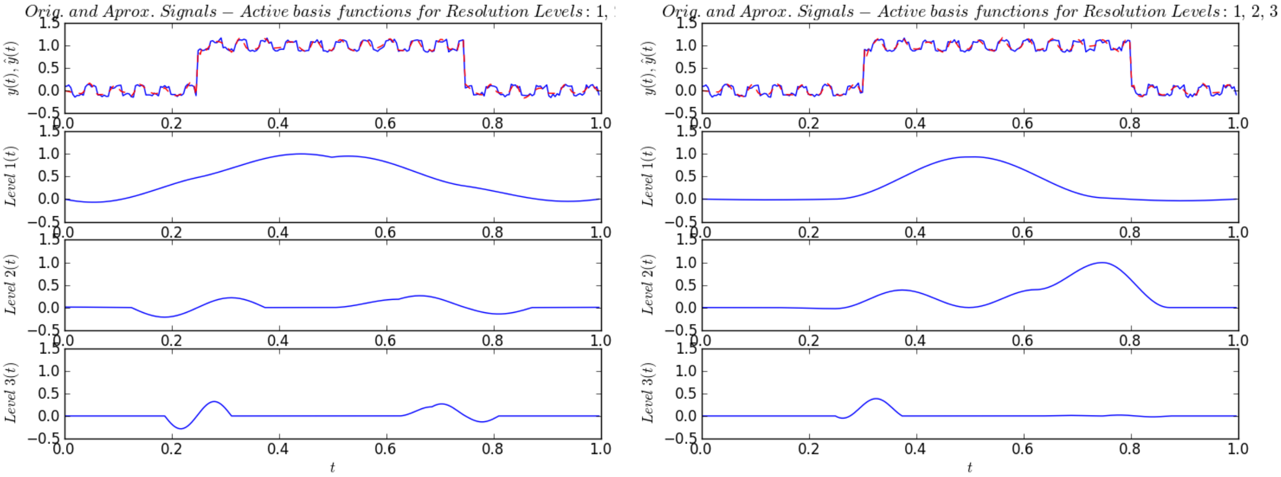
Figure 7.55: Original and Approximated Signal - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

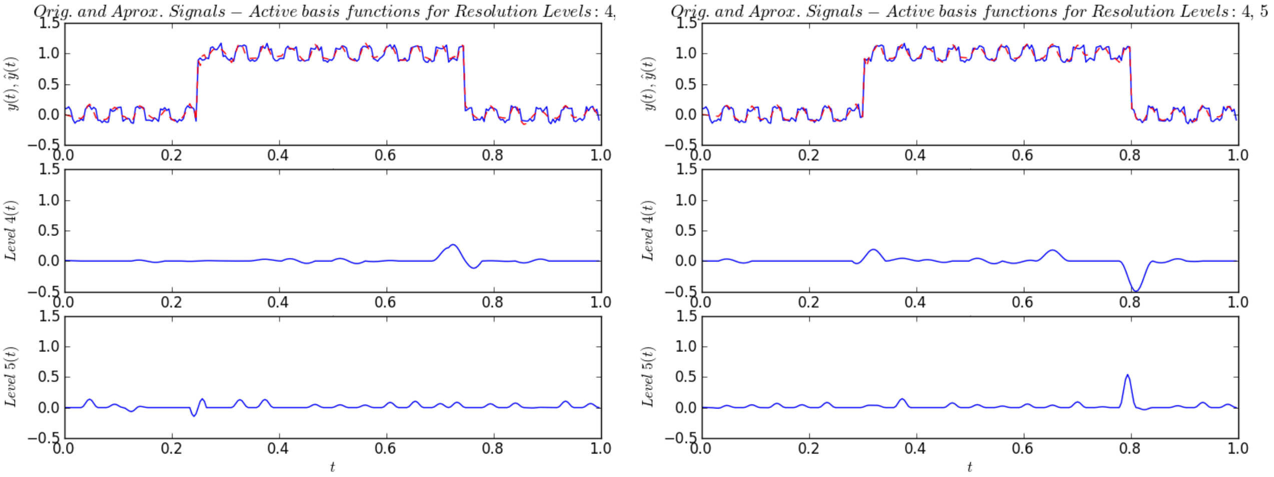
Figure 7.56: Superposed Signals and Error - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

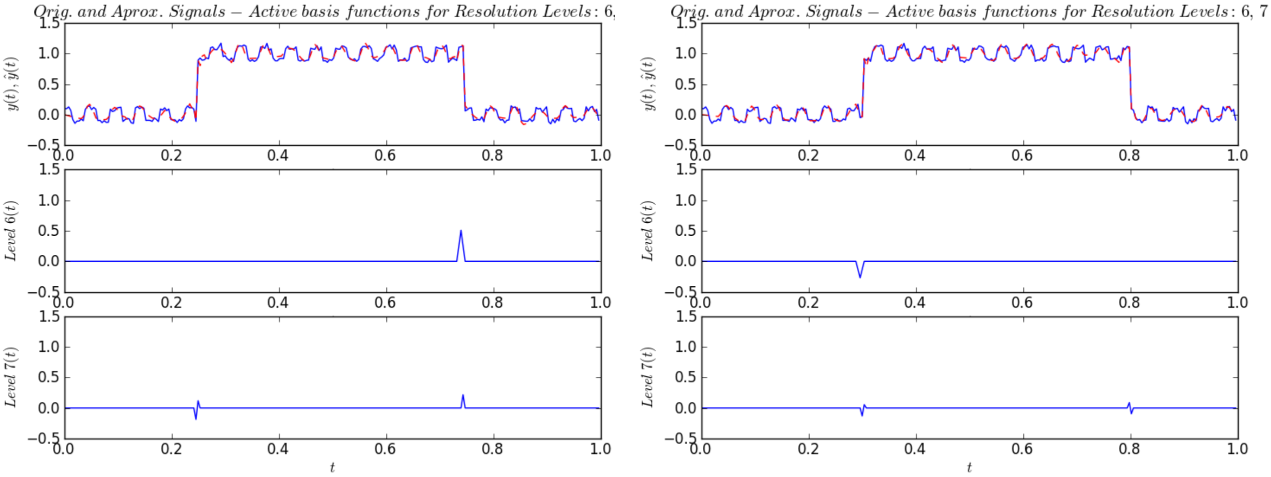
Figure 7.57: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

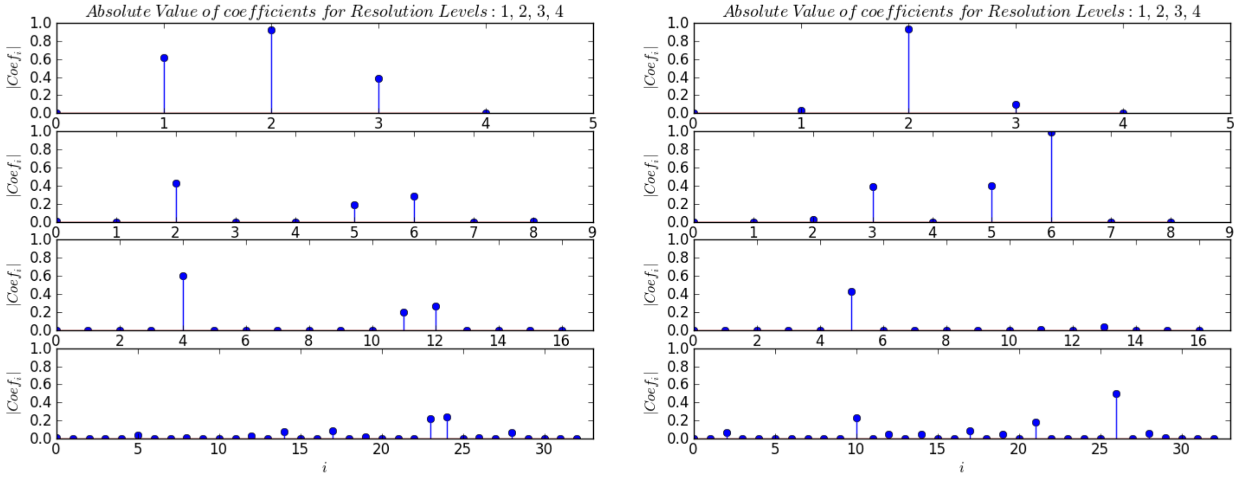
Figure 7.58: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

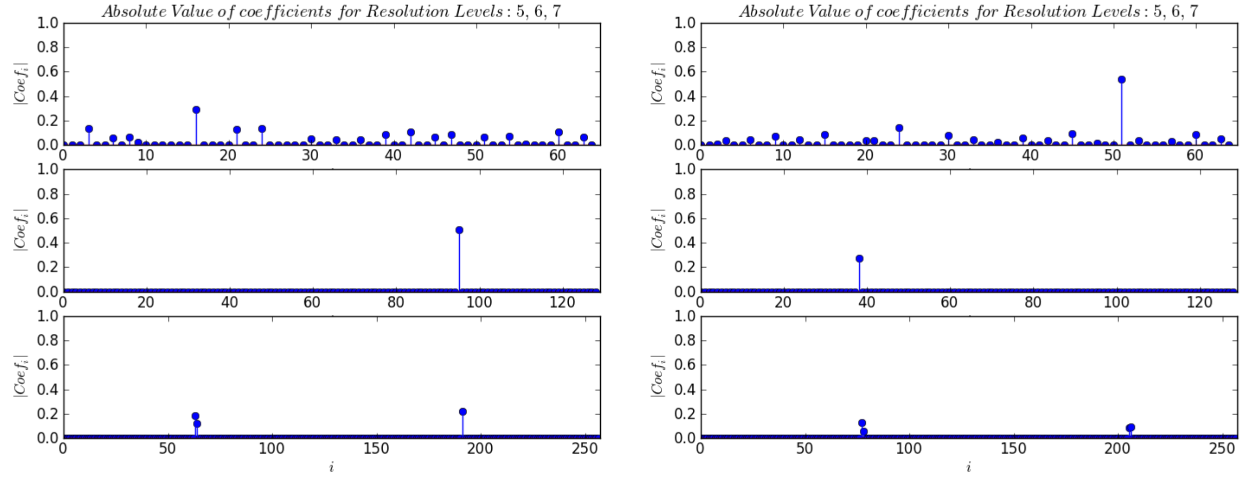
Figure 7.59: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

Figure 7.60: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

Figure 7.61: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal with Texture and Noise

From this first series of examples we can observe that:

- In general, for each example, the approximation is good, we can see that the pointwise error is small on all the examples, the error becomes higher in discontinuity points (which is natural due to the continuity of the approximating functions), this last observation is specially notorious in the non-textured examples, for textured ones the error seems to be equally distributed in the whole signal.

- Is interesting how the signal is decomposed on each level of resolution: the 'general features' (like a flat section of the signal) are represented essentially in the lower levels of resolution, we can notice this specially in no textured examples. On the other hand, the 'details' of the signal (like the discontinuities) are represented essentially on a higher levels of resolution, we can notice in the textured examples that the texture structure is essentially distributed along specific levels of resolutions, and is differentiable from the 'strong discontinuities' by noticing that this ones are represented in some levels in which texture is not represented.

- We observe by comparing the noisy and non-noisy examples that this method is able to ignore the noise of the signal, we get a very similar decomposition on noisy and non-noisy examples, this is a very desirable behaviour because this imply that the method is able to differentiate noise from texture, and therefore ignore the noise from the decomposition of the signal in resolution levels.

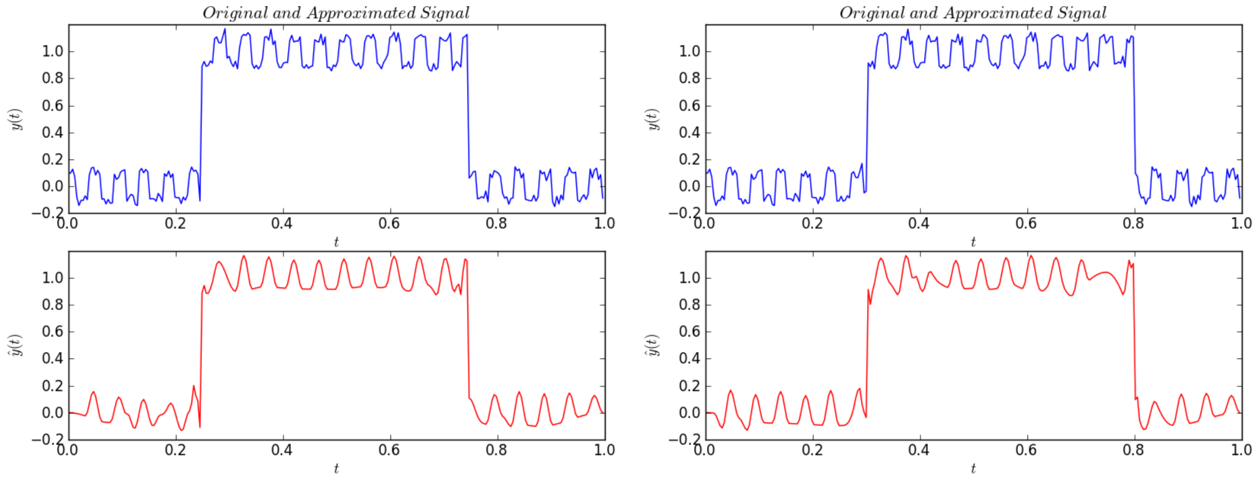
- From the comparison of original signal and shifted signal, we get that, the coefficients vary in a expected way: if we move the signal to the right, the coefficients on all the resolution levels move to the right and the amplitude of the functions remain 'relatively' stable. Unfortunately this is not accure as one could expect: A shifted signal may not have the same number of active functions on this process, i.e. the method is not completely shift invariant. This phenomena could be explained by noticing that this method is an approximation and the computational methods for computation applies some approximations which tend to have a different number of active functions for each simulation.

From this observations we can conclude that this method is able to approximate with a small error a signal, and we are able to decompose this signal on representative levels of resolution which lead to ‘discover’ the structure of the original signal: We are able to distinguish and separate the small details (like discontinuities, and texture) from the ‘general structure’ (flat or monotone regions), also, this approximation is stable, in the sense of how the coefficients of each ‘basis function’ vary in the expected way when we introduce a shift on the original signal. Unfortunately, the method is not shift invariant because the number of active functions for a shifted signal in general is not the same as the original signal.

7.4.5 Approximation and Decomposition for Texture-Noise Image: Real part of Complex Exponential Spline

The purpose of this particular example, in which we only consider the family of functions given by the real part of our exponential splines to perform the basis pursuit problem, is to compare 'how much' shift invariant is this standard family of functions in the basis pursuit problem. We will measure this setting by studying how the coefficients vary in the original signal approximation relative to the coefficients in the shifted signal approximation.

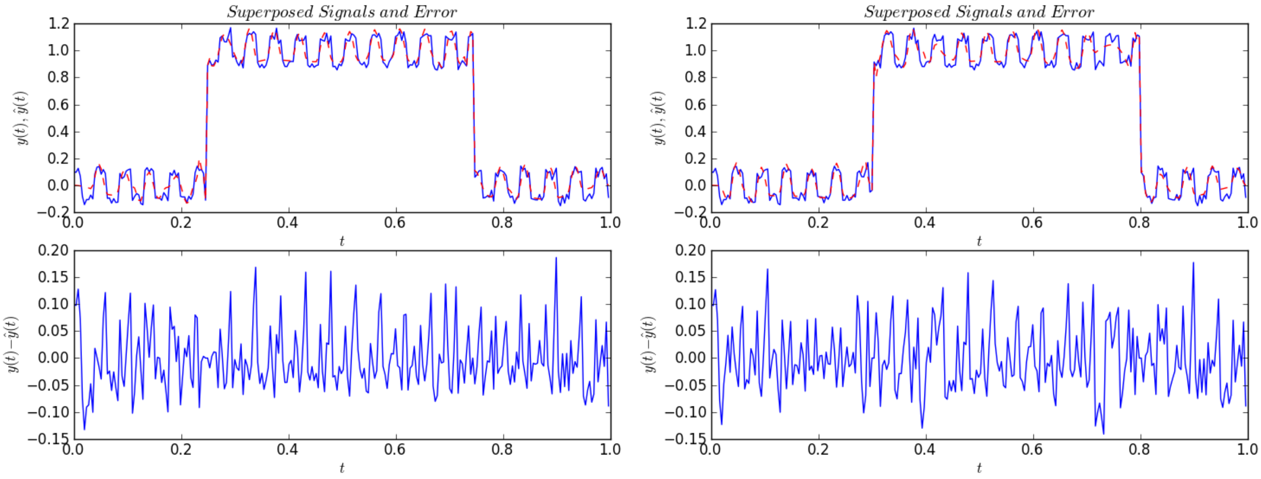
We only consider in this part only a signal with texture and noise, more examples can be found in the Appendix A.



(a) Original Signal data

(b) Shifted Signal data

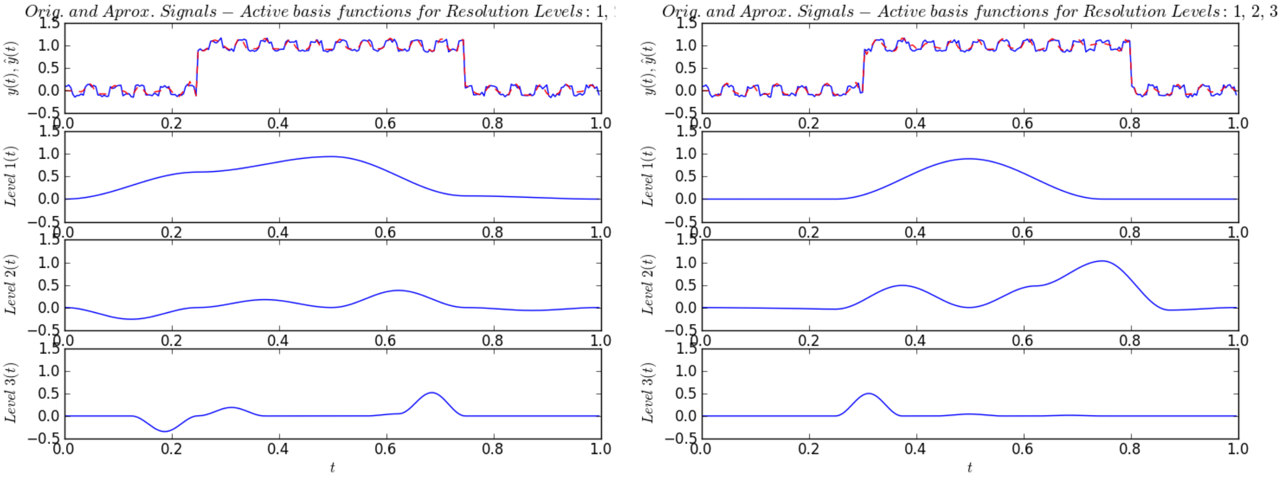
Figure 7.62: Original and Approximated Signal - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

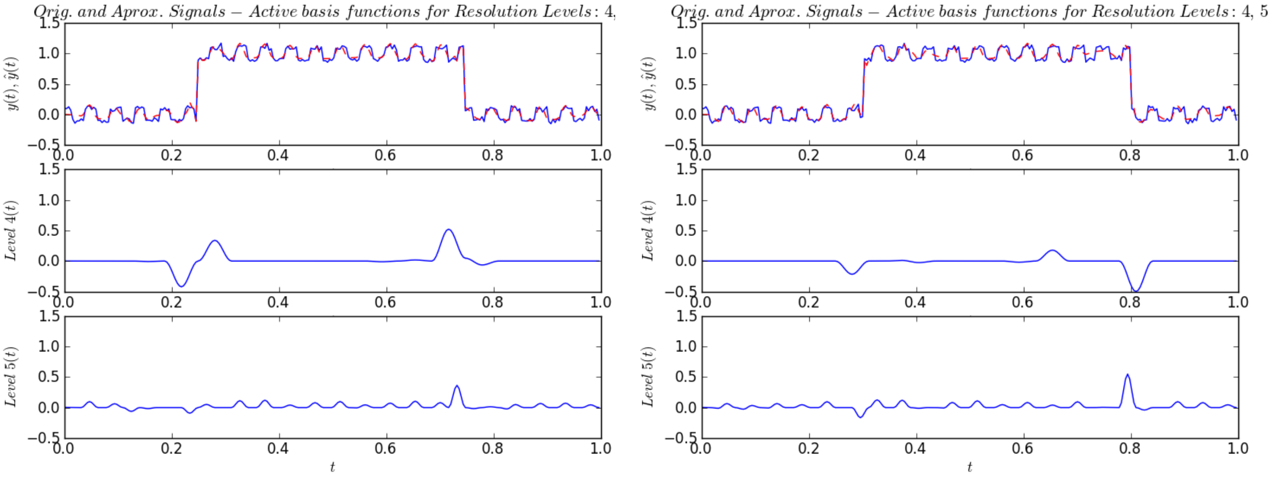
Figure 7.63: Superposed Signals and Error - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

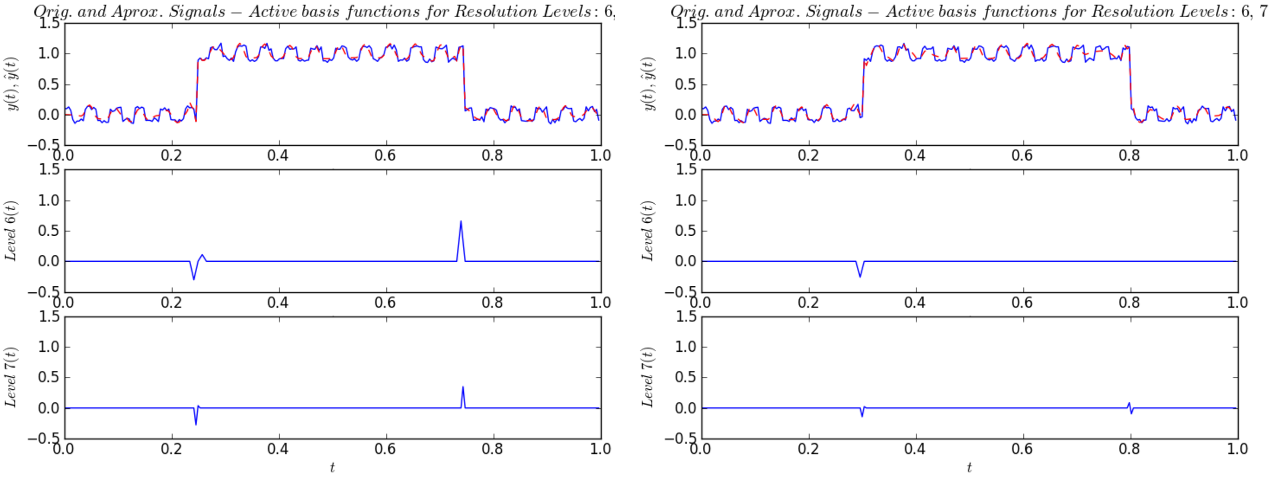
Figure 7.64: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

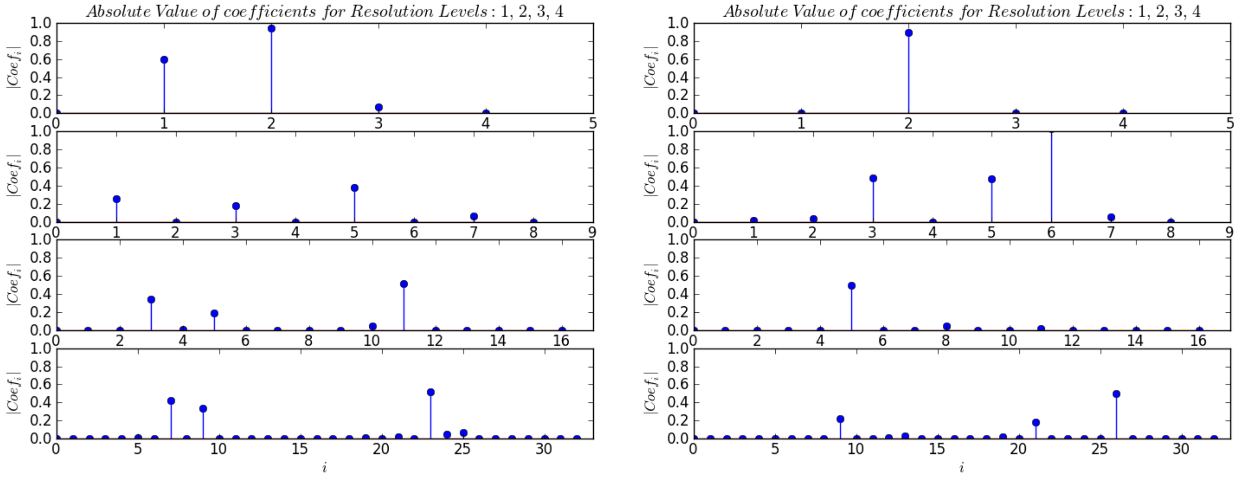
Figure 7.65: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

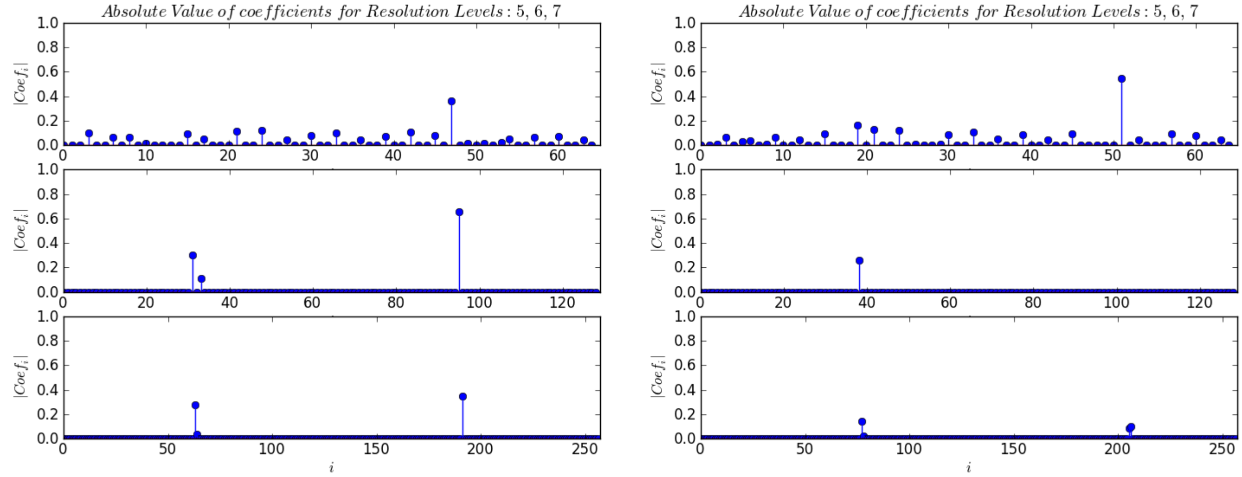
Figure 7.66: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

Figure 7.67: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal with Texture and Noise



(a) Original Signal data

(b) Shifted Signal data

Figure 7.68: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal with Texture and Noise

We also give the following resume table, in order to compare how does change the number of active functions on each approximation when we shift a signal, from this we can compare objectively the approximations with our proposed functions and a simpler (and classical) family of functions.

Table 7.1: Comparison table for two families of functions.

	No Noise Signal				Noisy Signal			
	No Textured		Textured		No Textured		Textured	
	Exp Spl	$Re(\cdot)$	Exp Spl	$Re(\cdot)$	Exp Spl	$Re(\cdot)$	Exp Spl	$Re(\cdot)$
Level 1 - Orig	3	3	3	3	3	3	3	3
Level 1 - Shifted	2	2	3	1	2	2	3	1
Level 2 - Orig	5	5	4	4	4	4	3	4
Level 2 - Shifted	4	4	4	6	4	4	4	6
Level 3 - Orig	5	7	3	5	6	6	3	4
Level 3 - Shifted	4	2	3	3	4	3	3	3
Level 4 - Orig	4	3	10	8	3	4	8	7
Level 4 - Shifted	4	8	9	6	5	7	10	6
Level 5 - Orig	3	3	18	23	3	4	18	23
Level 5 - Shifted	4	4	17	22	5	5	20	22
Level 6 - Orig	1	4	1	3	1	4	1	3
Level 6 - Shifted	2	2	1	1	2	2	1	1
Level 7 - Orig	2	4	3	3	3	4	3	3
Level 7 - Shifted	4	4	4	4	4	4	4	4
Variation	**7**	**15**	**3**	**12**	**9**	**10**	**6**	**10**

For this family of functions we can notice that:

- This functions are suitable to approximate the signals accurately, similarly to the case of considering the full complex exponential splines. We can see that we get a sort of decomposition of details and general parts of the signal in the different levels of resolution.
- Unlike the previous family of functions, in this case the introduction of shifted signal reveals that the coefficients become more 'disorted', the amplitudes tend to change more than in the previous case, and we can see that for several resolution levels the number of active functions changes much more as one could see on the case of considering the family of complex exponential splines. Moreover, from the last table we can see that the variation of the number of active functions on each level of resolution in all considered cases, the 'simpler' family gets a higher variation than the Exponential complex spline family.

Therefore, from this examples we can see that our proposed family of functions used to approximate a signal via the basis pursuit problem are suitable for this task. The approximation bring a small pointwise error in general and is a convenient way to decompose the signal in several resolution levels in which we can clearly identify the several components of a signal such that discontinuity points, textured structures and 'main structures' such as flat sections.

Moreover, this family of functions have a good behaviour when we shift a signal, unfortunately this approximation is not perfectly shift-invariant in the sense that when we shift the signal, the approximating functions have some changes respecting to the original approximating functions: the number of active functions slightly change and also the amplitude of this functions change. Anyway, we have seen that our proposed functions are better, in shift-invariance sense, in comparison with the family of only the real part of this complex exponential splines.

Chapter 8

Conclusions and Future Work

8.1 Main Results

The main topics and results given in this work are:

We studied a new way to perform segmentation of signals and images, specially focused on treatment of signals and images with texture: The non-local segmentation. We took the original non-local segmentation functional given by [34] and shown that this functional is not suitable in the theorical sense of Γ-convergence for segmentation purposes.

We have redefined in the 1-dimensional case the non-local segmentation functional in order to obtain an interesting Γ-limit, this functional is defined for the function u in the fractional Sobolev space H^s for $s \in (1/2, 1)$, we successfully proved, under several assumptions given in 4.2.1, that the functional given by 4.13 Γ-converges in L^1 topology to functional 4.14.

We have performed several numerical testings devoted to explore the potential of non-local methods in image processing. We started testing the non-local denoising filter developed by Buades in order to study how this filter (which is a key component on non-local segmentation) behaves in 1D and 2D, after that we perform numerical simulations for Ambrosio-Tortorelli segmentation and Non-local segmentation to compare them, from this we have deduced that non-local methods gives great results on textured image, specially segmentating 'hidden objects' in textured environments and segmentating 'natural environment' images; in this cases the non-local segmentation performs much better than the classical Ambrosio-Tortorelli method, in 1D case we also get much better results in signals with texture, getting a better edge set v function and better regularized signal.

We also explored a method to 'approximately separate' a signal into its different levels of resolution introducing a particular family of functions of the so-called complex-exponential splines, an interesting behaviour is obtained: We are able to identify the texture of a signal (even in a signal with noise) by its approximated version, in which we can clearly identify the texture component in a specific level of resolution. We also compare how much sensitive to shifted signals are the complex-exponential spline representation against the single (and classical) real part representation.

8.2 Future Work

The main open problem is how to extend the non-local functional defined in Fractional Sobolev space to higher dimensions, this is not trivial since the definition of this spaces relies strongly on the dimension, therefore, the functional may need to be changed in order to work in an appropriately space. Naturally after this a study about the Γ-convergence is needed.

The natural following step is to develop an efficient numerical implementation for higher dimension segmentation in Fractional Sobolev space, it may be follow the same procedure as the one shown in this work but this should be done with caution since it may imply the computation of singular terms.

About the exponential splines, a generalization of this procedure could be a big problem of study, we only focused in a particular case in order to explore its possibilities in order to 'detect and classify' texture and approximate signals in a fast and light way.

Bibliography

[1] G. Allaire. *Analyse numérique et optimisation: une introduction à la modélisation mathématique et à la simulation numérique.* École polytechnique, 2005.

[2] F. Álvarez. Introducción al cálculo de variaciones, 2011.

[3] L. Ambrosio. *A compactness theorem for a special class of functions of bounded variation.* Scuola normale superiore, 1988.

[4] L. Ambrosio. A compactness theorem for a new class of variational problems. *Boll. Un. Mat. It*, 3:857–881, 1989.

[5] L. Ambrosio. Variational problems in sbv and image segmentation. *Acta Applicandae Mathematicae*, 17(1):1–40, 1989.

[6] L. Ambrosio and V. M. Tortorelli. On the approximation of functionals depending on jumps by elliptic functionals via Gamma-convergence. *Comm. Pure Appl. Math*, 43:999–1036, 1990.

[7] L. Ambrosio and V.M. Tortorelli. *On the approximation of free discontinuity problems.* Scuola Normale Superiore, 1990.

[8] T. Asahi, K. Ichige, and R. Ishii. A Computationally Efficient Algorithm for Exponential B-Splines Based on Difference/IIR Filter Approach. *IEICE Trans Fundam Electron Commun Comput Sci (Inst Electron Inf Commun Eng)*, E85-A:1265–1273, 2002.

[9] H. Attouch. *Variational convergence for functions and operators.* Pitman Advanced Pub. Program, 1984.

[10] G. Aubert and P. Kornprobst. *Mathematical problems in image processing: partial differential equations and the calculus of variations*, volume 147. Springer, 2006.

[11] L. Bar, A. Brook, N. Sochen, and N. Kiryati. Deblurring of color images corrupted by impulsive noise. *Image Processing, IEEE Transactions on*, 16(4):1101–1111, 2007.

[12] S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[13] A. Braides. *Approximation of free-discontinuity problems*, volume 1694. Springer, 1998.

[14] A. Braides. *Gamma-convergence for Beginners.* Oxford University Press Oxford:, 2002.

[15] L.M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217, 1967.

[16] X. Bresson. A short note for nonlocal tv minimization. *http://www.cs.cityu.edu.hk/~xbresson/ucla/index.html*, 2009.

[17] H. Brézis. How to recognize constant functions. connections with sobolev spaces. *Russian Mathematical Surveys*, 57(4):693, 2007.

[18] A. Buades. *Image and movie denoising by non local means.* PhD thesis, Universitat de les Illes Balears, 2005.

[19] A. Buades, B. Coll, and J.M. Morel. A non-local algorithm for image denoising. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 60–65. IEEE, 2005.

[20] A. Buades, B. Coll, and J.M. Morel. Nonlocal image and movie denoising. *Int. J. Computer Vision*, 76:123–139, 2008.

[21] A. Chambolle. Image Segmentation by Variational Methods: Mumford and Shah Functional and the Discrete Approximations. *Siam Journal on Applied Mathematics*, 55:827–863, 1995.

[22] A. Chambolle. Finite-differences discretizations of the mumford-shah functional. *Mathematical Modelling and Numerical Analysis*, 33:261–288, 1999.

[23] T.F. Chan, J. Shen, and L. Vese. Variational pde models in image processing. *Notices of AMS*, 50(1), 2003.

[24] S. Chen and D. Donoho. Basis pursuit. In *Signals, Systems and Computers, 1994. 1994 Conference Record of the Twenty-Eighth Asilomar Conference on*, volume 1, pages 41–44. IEEE, 1994.

[25] B. Dacorogna. *Introduction to the Calculus of Variations.* World Scientific Publishing Company, 2004.

[26] J. Dahl and L. Vandenberghe. Cvxopt documentation, 2007. http://abel.ee.ucla.edu/cvxopt/documentation/index.html.

[27] E. De Giorgi, M. Carriero, and A. Leaci. Existence theorem for a minimum problem with free discontinuity set. *Archive for Rational Mechanics and Analysis*, 108:195–218, 1989.

[28] R. Delgado-Gonzalo, P. Thévenaz, and M. Unser. Exponential splines and minimal-support bases for curve representation. *Computer Aided Geometric Design*, 2011.

[29] E. Di Nezza, G. Palatucci, and E. Valdinoci. Hitchhiker's guide to the fractional sobolev spaces. *Bulletin des Sciences Mathématiques*, 2011.

[30] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6(6):721–741, 1984.

[31] G. Gilboa and S. Osher. Nonlocal linear image regularization and supervised segmentation. *Multiscale Modeling and Simulation*, 6:595–630, 2007.

[32] T. Goldstein and S. Osher. The split bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009.

[33] J. Jost and X. Li-Jost. *Calculus of variations*, volume 64. Cambridge University Press, 1999.

[34] M. Jung, X. Bresson, T. Chan, and L. Vese. Nonlocal mumford-shah regularizers for color image restoration. *IEEE Transactions on Image Processing*, 20:1583–1598, 2011.

[35] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

[36] N. Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Applied and Computational Harmonic Analysis*, 10(3):234–253, 2001.

[37] J.S. Lee. Digital image enhancement and noise filtering by use of local statistics. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-2(2):165–168, 1980.

[38] J.S. Lee. Digital image smoothing and the sigma filter. *Computer Vision, Graphics, and Image Processing*, 24(2):255–269, 1983.

[39] M. Mahmoudi and G. Sapiro. Fast image and video denoising via nonlocal means of similar neighborhoods. *Signal Processing Letters, IEEE*, 12(12):839–842, 2005.

[40] S. Mallat. *A wavelet tour of signal processing*. Academic press, 1999.

[41] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.

[42] L.I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

[43] J. Simon. Sobolev, Besov and Nikolskii fractional spaces: imbeddings and comparisons for vector valued spaces on an interval. *Annali di Matematica Pura ed Applicata*, 157(1):117–148, 1990.

[44] S.M. Smith and J.M. Brady. Susan-a new approach to low level image processing. *International journal of computer vision*, 23(1):45–78, 1997.

[45] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on*, pages 839–846. IEEE, 1998.

[46] L. Yaroslavsky. Digital picture processing. an introduction. *Digital picture processing. An introduction. Springer Series in Information Sciences, Vol. 9. Springer-Verlag, Berlin-Heidelberg-New York-Tokyo. 12+ 276 pp. Price DM 112.00 (1985). ISBN 3-540-11934-5 (FR Germany), ISBN 0-387-11934-5 (USA).*, 1, 1985.

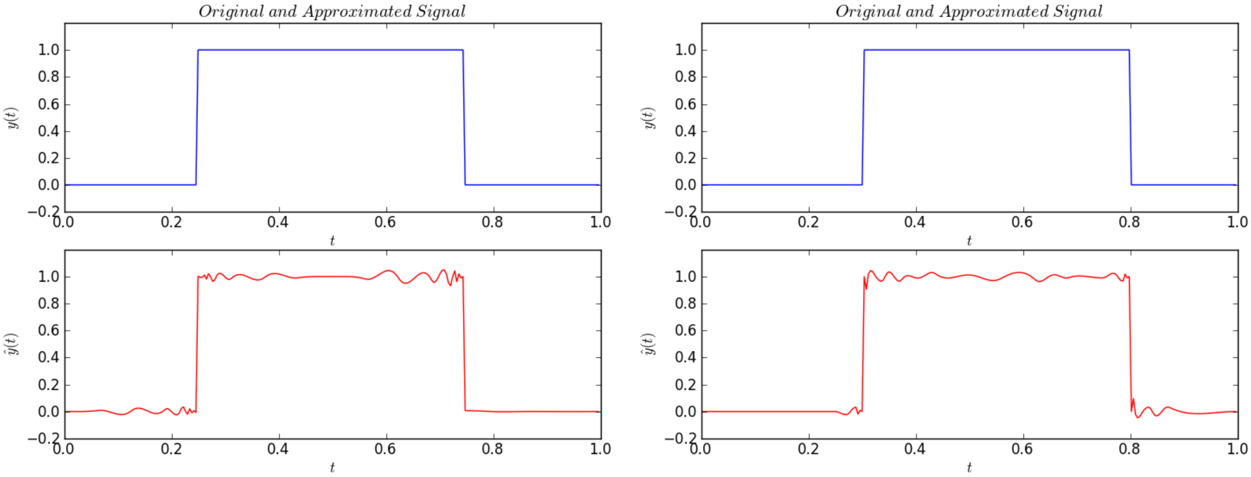
[47] X. Zhang, M. Burger, X. Bresson, and S. Osher. Bregmanized nonlocal regularization for deconvolution and sparse reconstruction. *SIAM Journal on Imaging Sciences*, 3(3):253–276, 2010.

Appendix A

Basis Pursuit for the Real Part of Exponential Splines: More Examples

In this appendix we give more examples of the basis pursuit problem when we consider the family of real part of complex exponential splines, for the sake of completeness and in order to observe the same phenomena that we just have analyzed in 7.4.5.

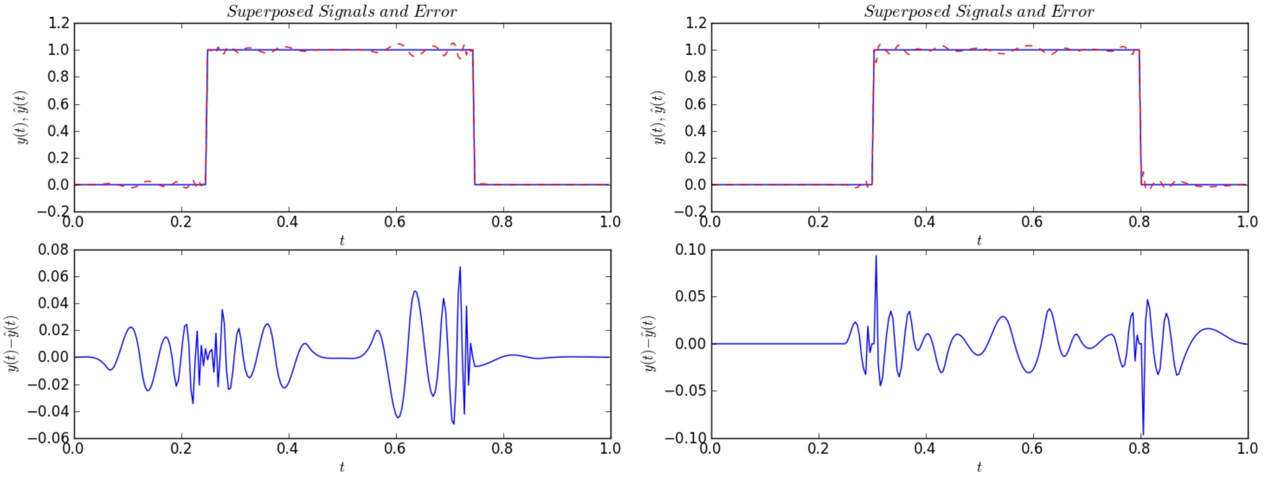
A.1 Approximation and Decomposition for No Texture-No Noise Image: Real part of Complex Exponential Spline



(a) Original Signal data

(b) Shifted Signal data

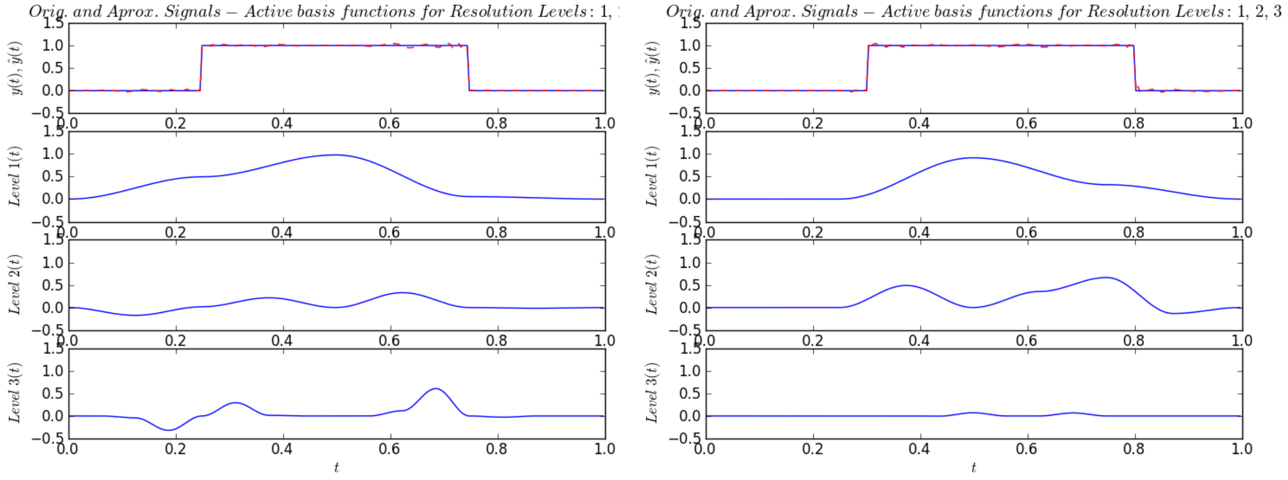
Figure A.1: Original and Approximated Signal - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

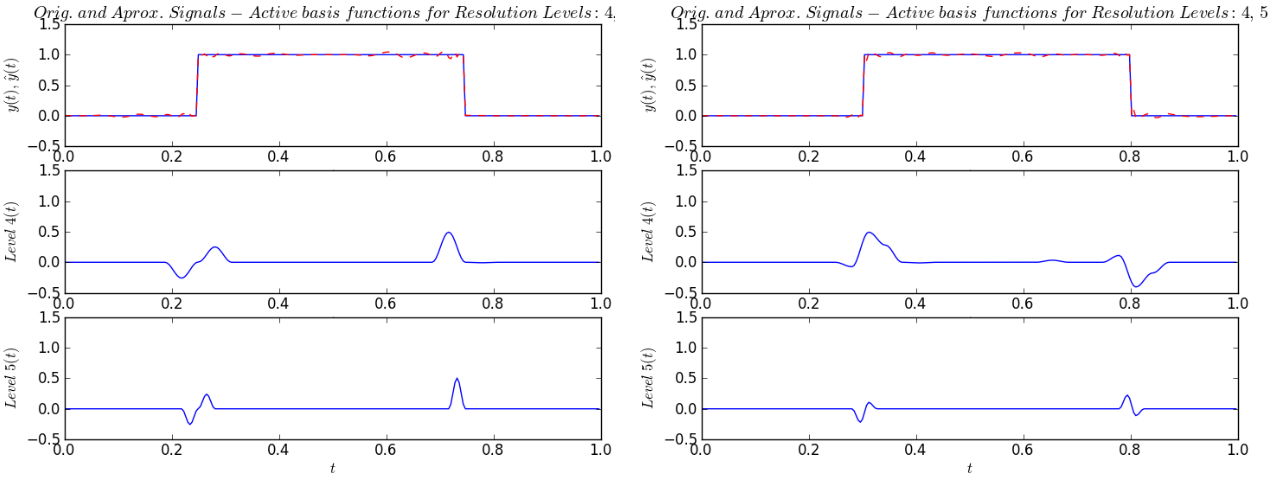
Figure A.2: Superposed Signals and Error - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

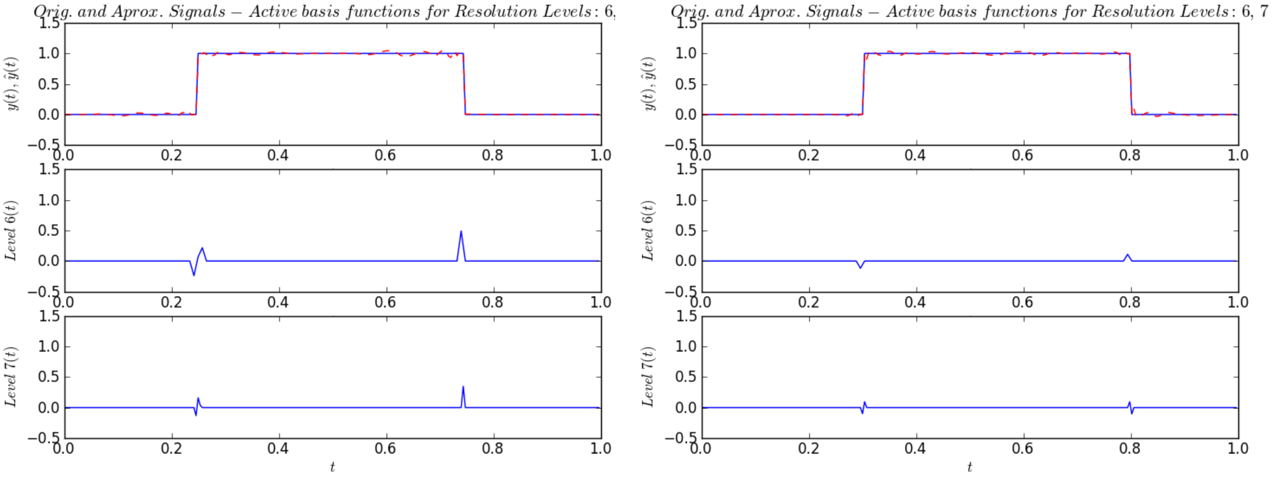
Figure A.3: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

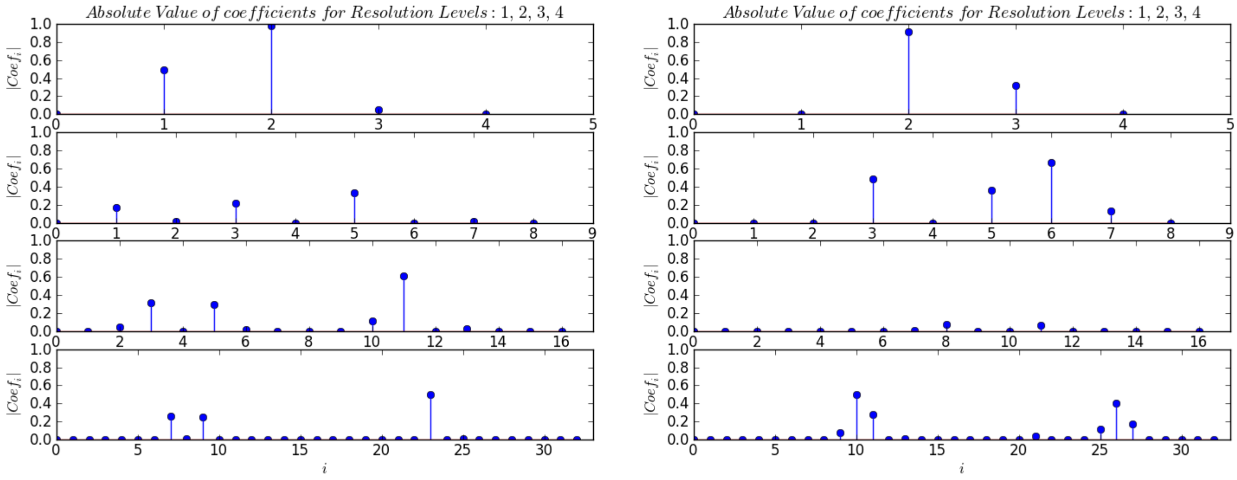
Figure A.4: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

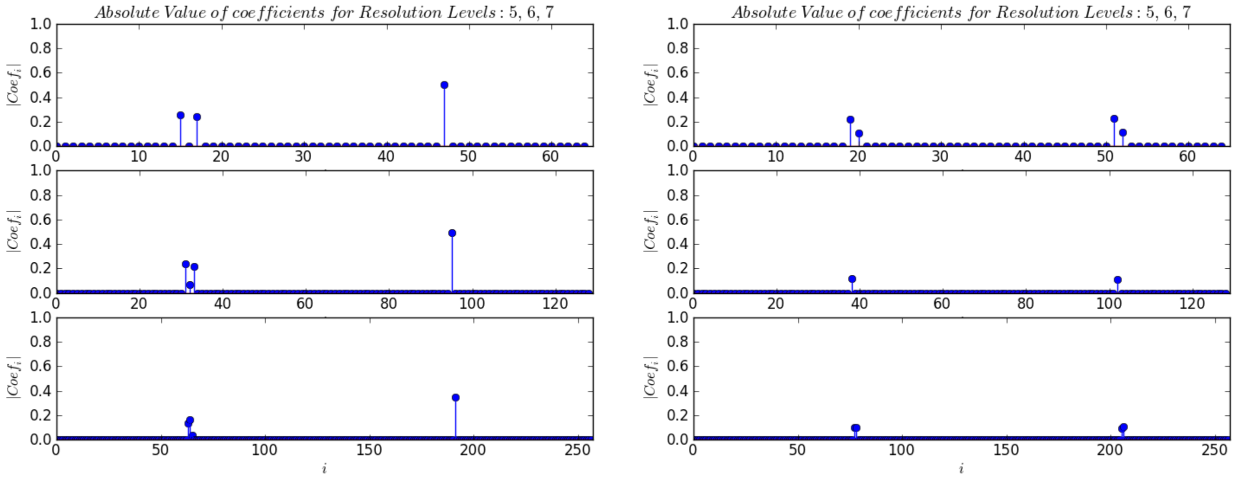
Figure A.5: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal without Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

Figure A.6: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal without Texture and without Noise

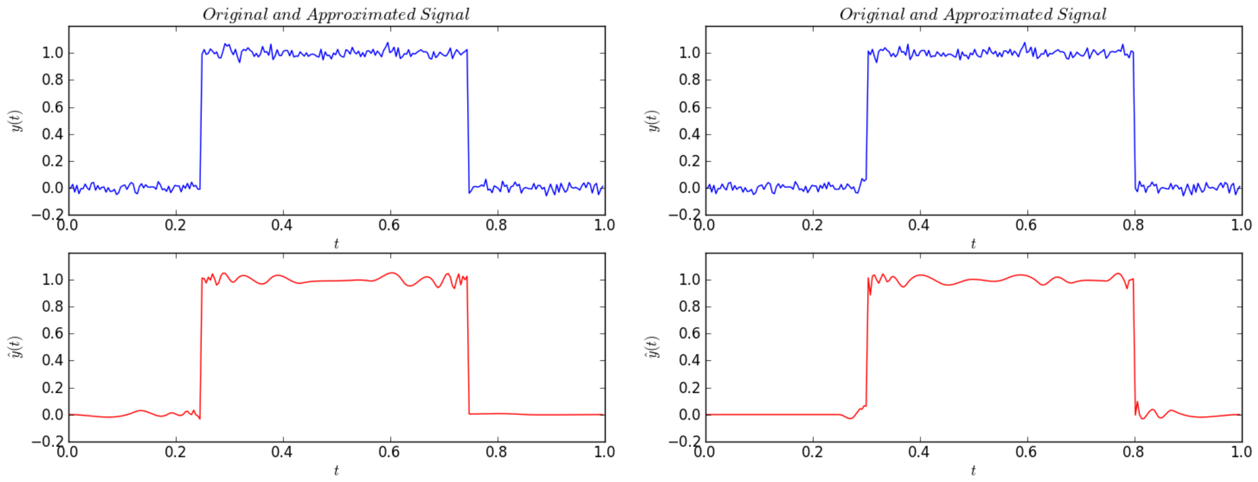


(a) Original Signal data

(b) Shifted Signal data

Figure A.7: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal without Texture and without Noise

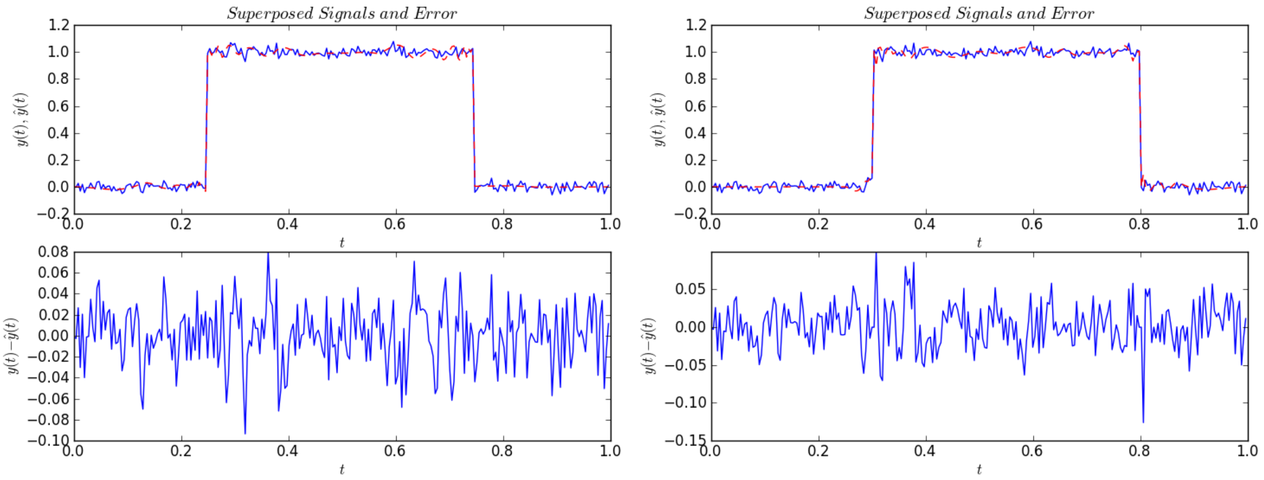
A.2 Approximation and Decomposition for No Texture-Noisy Image: Real part of Complex Exponential Spline



(a) Original Signal data

(b) Shifted Signal data

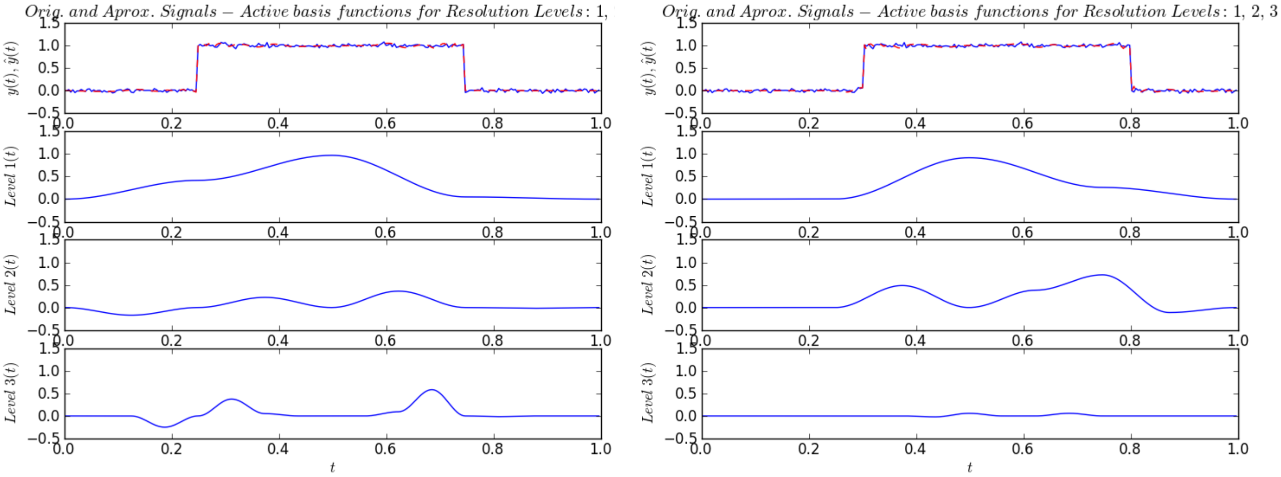
Figure A.8: Original and Approximated Signal - Input Signal with Noise but without Texture



(a) Original Signal data

(b) Shifted Signal data

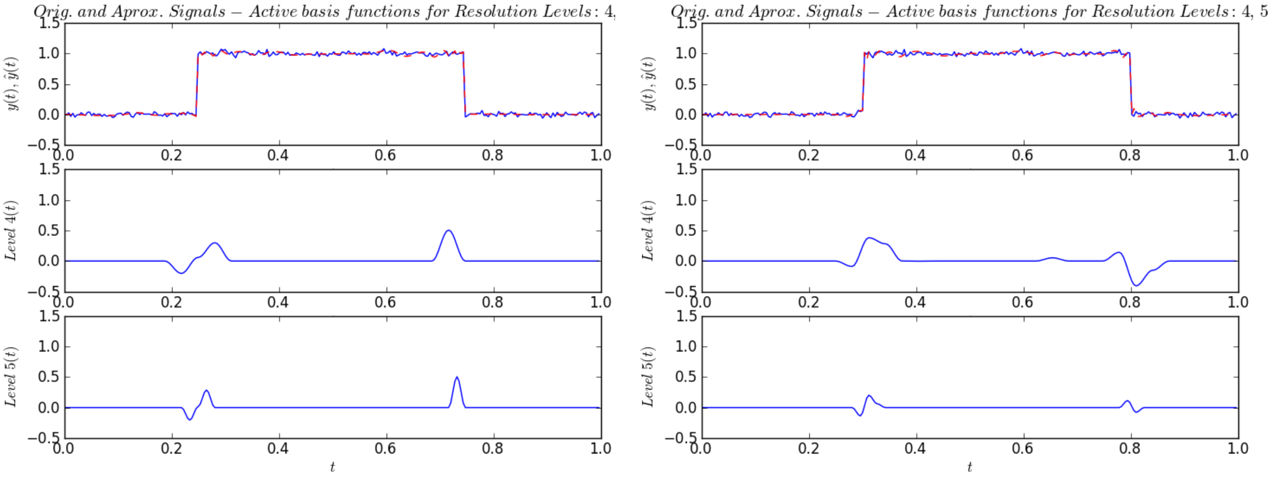
Figure A.9: Superposed Signals and Error - Input Signal with Noise but without Texture



(a) Original Signal data

(b) Shifted Signal data

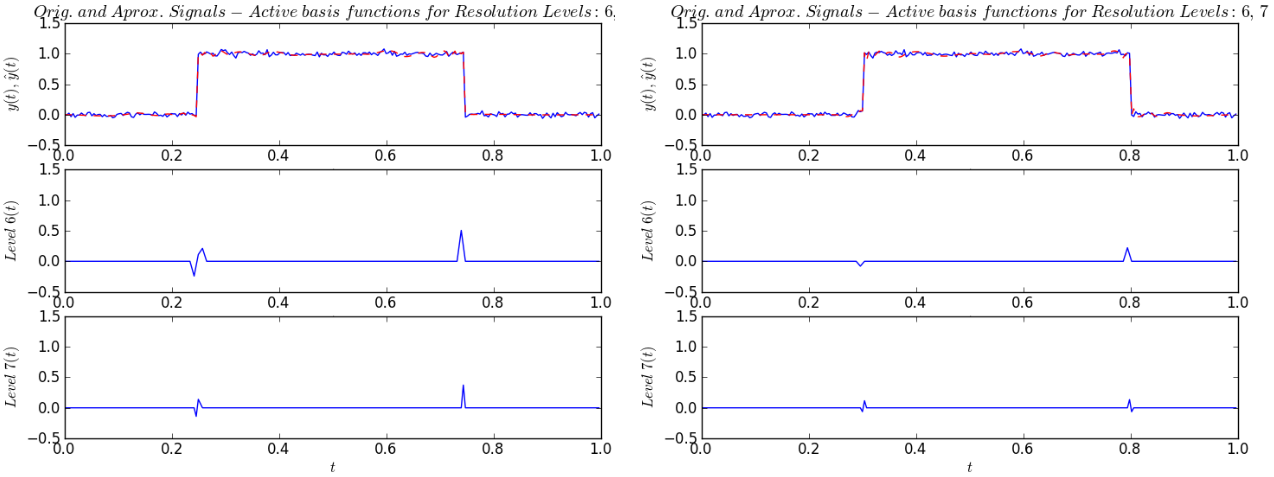
Figure A.10: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal with Noise but without Texture



(a) Original Signal data

(b) Shifted Signal data

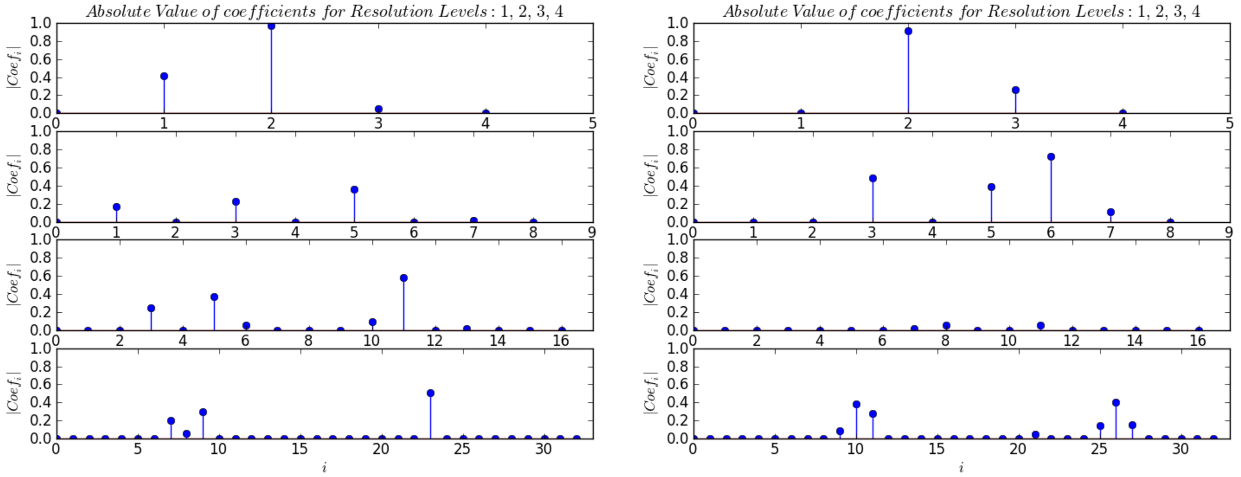
Figure A.11: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal with Noise but without Texture



(a) Original Signal data

(b) Shifted Signal data

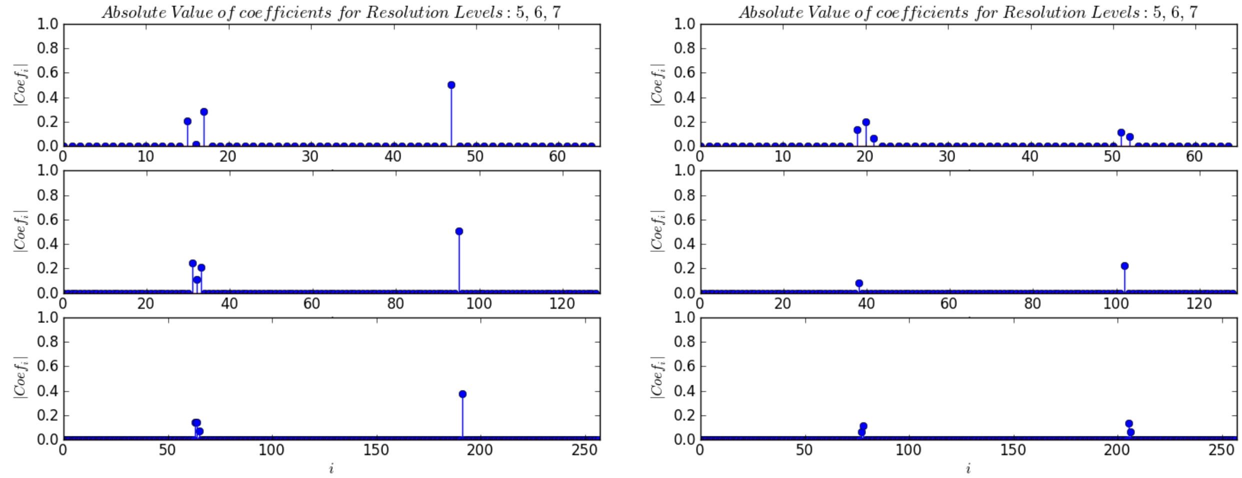
Figure A.12: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal with Noise but without Texture



(a) Original Signal data

(b) Shifted Signal data

Figure A.13: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal with Noise but without Texture

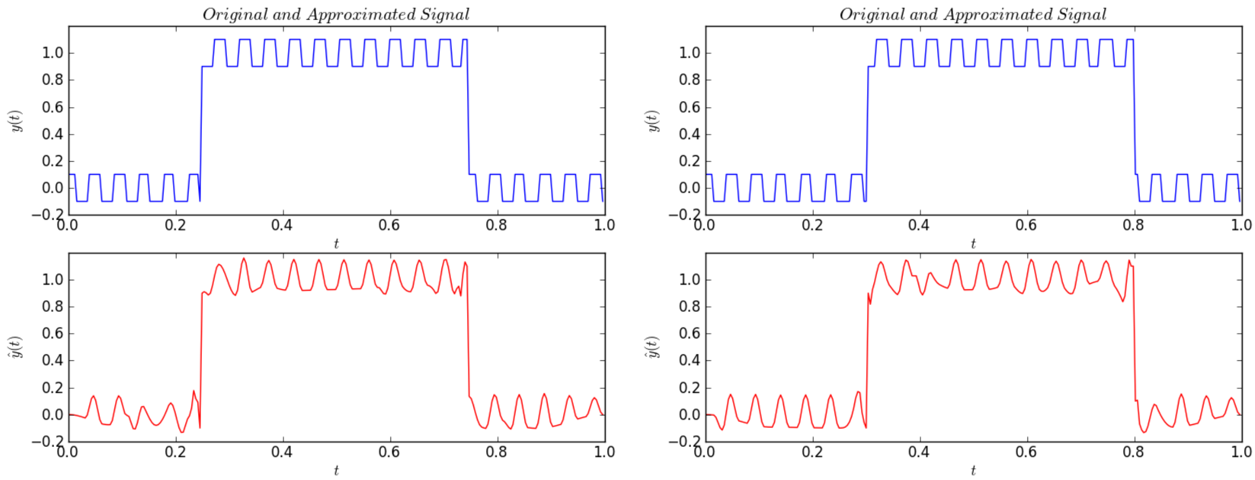


(a) Original Signal data

(b) Shifted Signal data

Figure A.14: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal with Noise but without Texture

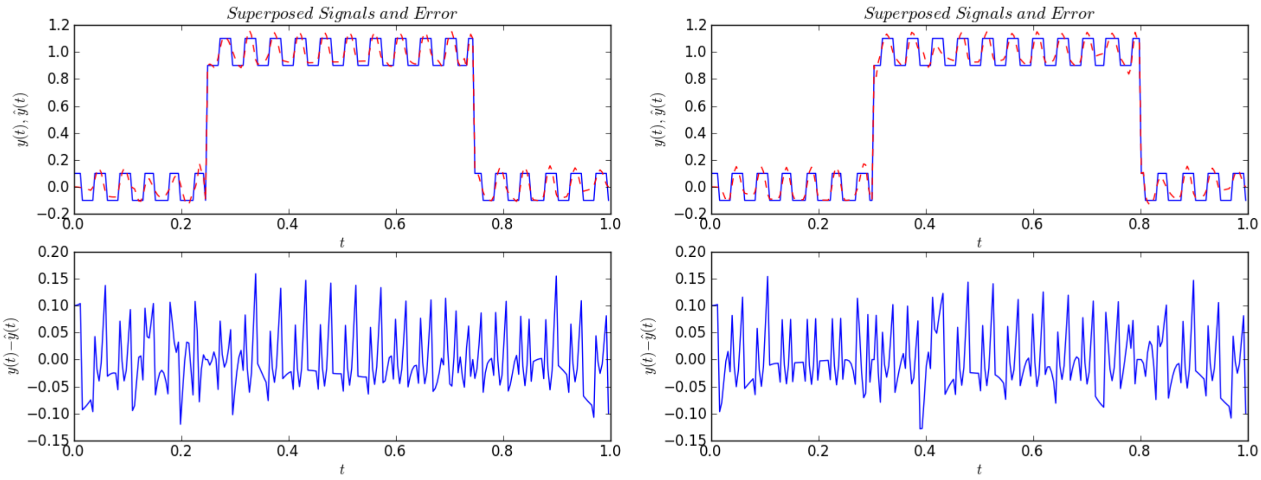
A.3 Approximation and Decomposition for Texture-No Noise Image: Real part of Complex Exponential Spline



(a) Original Signal data

(b) Shifted Signal data

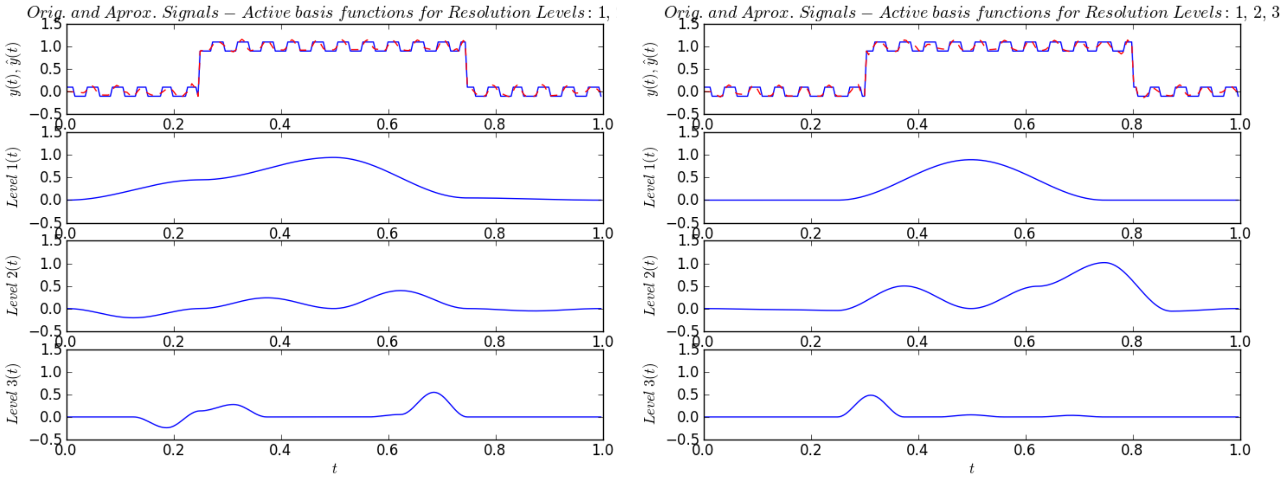
Figure A.15: Original and Approximated Signal - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

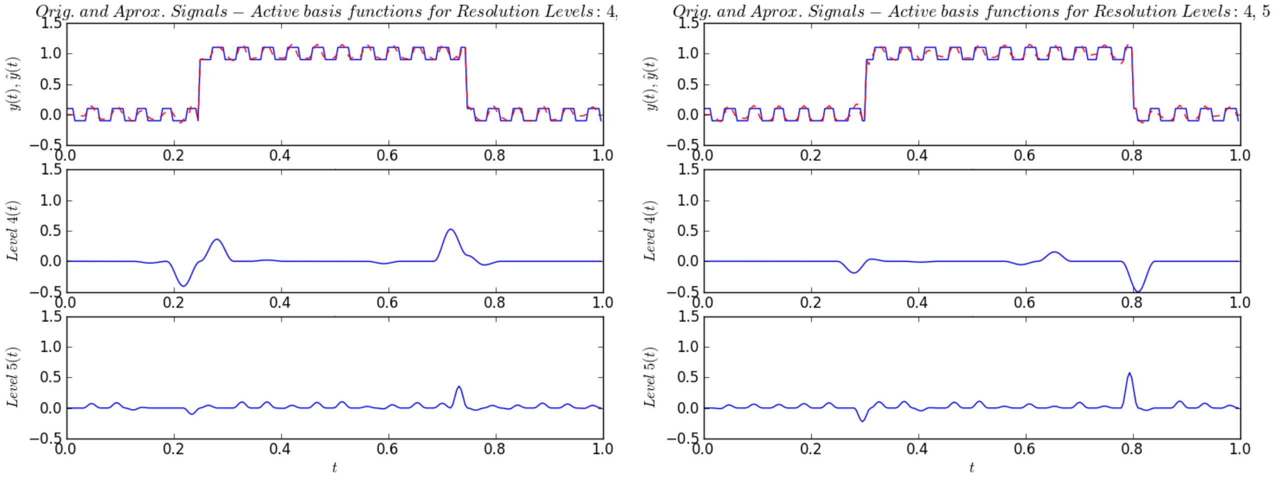
Figure A.16: Superposed Signals and Error - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

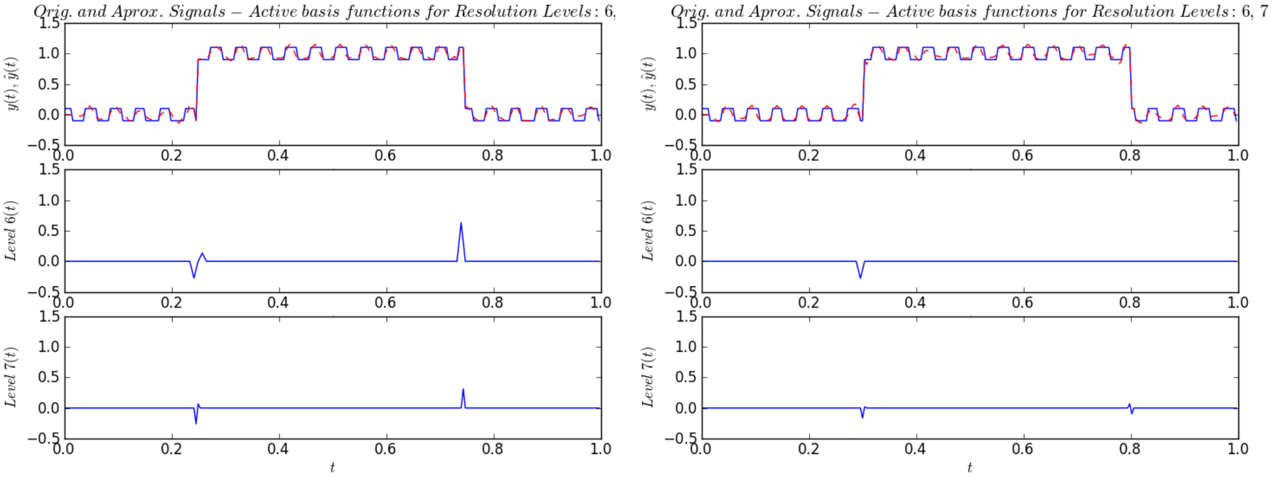
Figure A.17: Reconstruction of the Signal for Resolution Levels: 1, 2, 3 - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

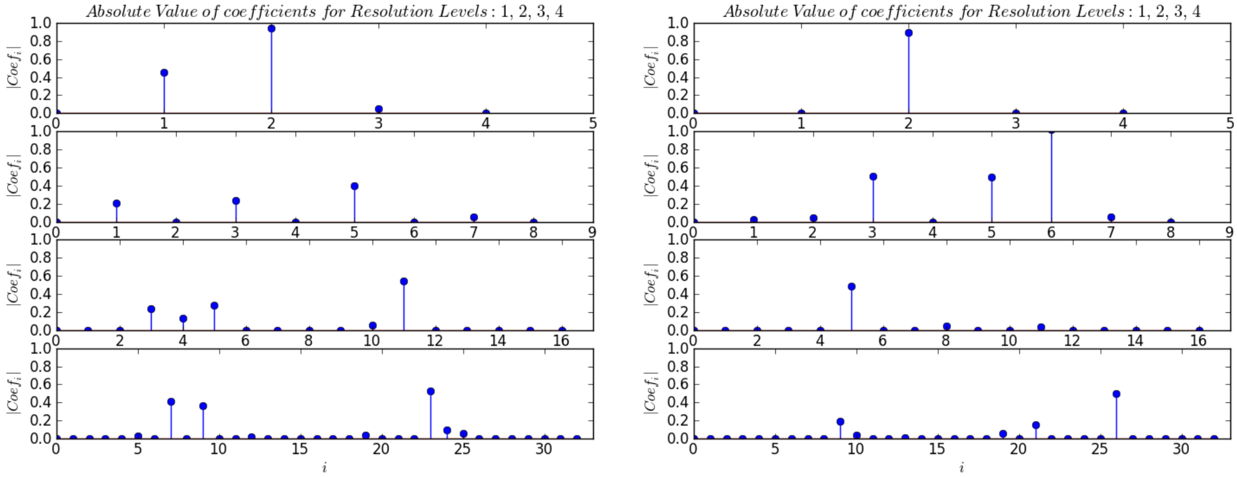
Figure A.18: Reconstruction of the Signal for Resolution Levels: 4, 5 - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

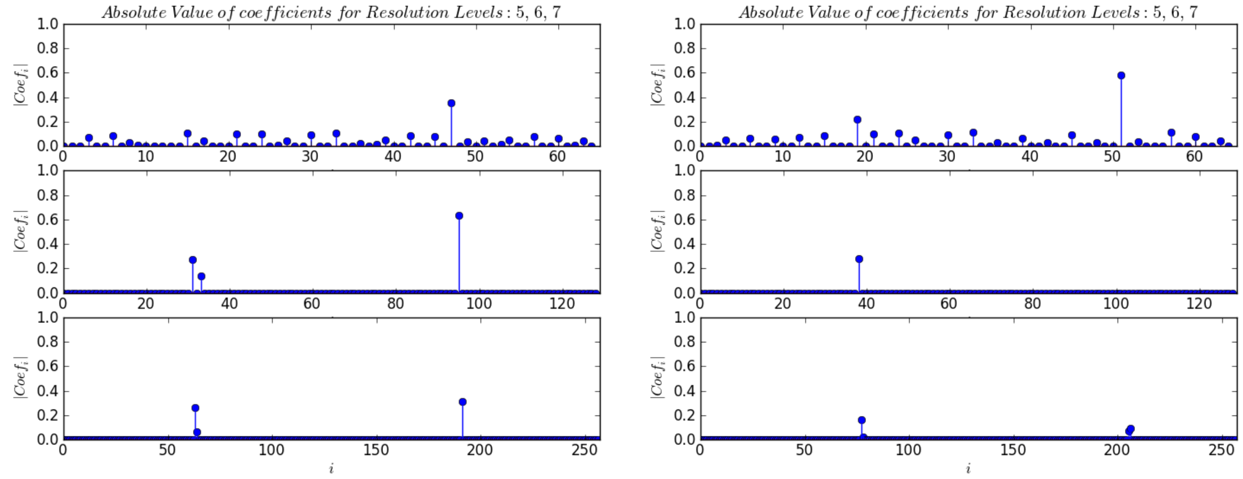
Figure A.19: Reconstruction of the Signal for Resolution Levels: 6, 7 - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

Figure A.20: Amplitude of Active Functions in Resolution Levels: 1, 2, 3, 4 - Input Signal with Texture and without Noise



(a) Original Signal data

(b) Shifted Signal data

Figure A.21: Amplitude of Active Functions in Resolution Levels: 5, 6, 7 - Input Signal with Texture and without Noise