Phase Evaluation and Segmentation

A Dissertation

Presented to the Faculty of the Graduate School

of

Yale University

in Candidacy for the Degree of

Doctor of Philosophy

by

Michel R. Nahon

Dissertation Director: Ronald R. Coifman

May 2000

Abstract

Phase Evaluation and Segmentation

Michel R. Nahon

Yale University

2000

This dissertation is organized in 3 parts. The first part is a study of instantaneous frequencies for one and two-dimensional signals. In 1946, Gabor proposed in [1] to analyze a one dimensional signal via its complexification. We refine this idea and study the Blaschke Factorization to separate frequential and amplitudal information. We demonstrate stability and invariance properties. We describe an iterative algorithm to decompose a signal in an orthogonal basis of finite Blaschke product. An extension of this algorithm is presented for the two dimensional case.

In part II, we deal with the problem of textural segmentation. Many applications in image analysis are based directly or indirectly on segmentation. Several algorithms give good results. We study the properties of the pyramidal algorithm developed by J.-M. Morel (and his collaborators), that is based on the Mumford-Shah functional, as its properties interest us. For textured images a preprocessing is mandatory. In [19], a vector image is obtained after filtering with an undecimated wavelet decomposition by Koepfler et al.. The problem is now to segment a vector image. The "good filters" have to be chosen to obtain a reasonable segmentation, regions with comparable sizes. We propose an algorithm to select the useful filters in a library for a chosen image. It enables to reduce the computational

time and give a more efficient segmentation. We present an extension of the Mumford-Shah functional and show how to reduce the dimension of our vector image. Different results and a counter-example, where the pyramidal algorithm is not optimal, are presented.
The last part is devoted to the application of undecimated wavelets. We believe that multiscale analysis is an important tool for image and signal processing. We represent our data with the wavelets and wavelet packets decomposition. We work with undecimated wavelets since they are grid's independent. We first summarize their properties before showing some applications. The first application is deconvolution, sharpening and smoothing signals using the subspaces obtained with the wavelet packets. The second one is about denoising radar images using the separation of the structures given by the wavelet representation, to solve the troubles generated by the the targets. And finally, by extracting the variations at various scales, we show results for the detection of brain activities in functional-MRI.

A ma grand-mere Renée.

Contents

Acknowledgments

I would like to thank my advisor, Professor Ronald Coifman for his constant support and enthusiastic conversations during this thesis. I was very fortunate to have the opportunity to work with him.
I am also very grateful to Professors Yves Meyer and Jean-Michel Morel who introduced me respectively to the wavelets and image processing during my master of science in Paris. They enable my first trip to the U.S.A. in 1995 and since then maintain enriching collaborations during my summer trips to Paris. Jean-Michel Morel also enabled me to use the software MEGAWAVE that contains the segmentation algorithm, developed by his collaborators of the CEREMADE (C.N.R.S., Paris).
Many thanks to Professors Gregory Beylkin and Peter Jones for serving as the reading committee members of my thesis.
On the personal side, I would also like to thank my former officemates Roland Guglielmi and Lionel Woog for their help, and also all the members of the applied mathematics team. Especially Artur, Francois, Maxim and Thomasz who proofread parts of my thesis.

MICHEL R. NAHON

Yale University

May 2000

Chapter 1

Phase Evaluation and Blaschke Product

1.1 Introduction

We develop an approach to signal analysis using instantaneous frequencies, for periodic function. We exploit an idea of Gabor's developed in [1] to use the phase of the complexified analytic signal to define the instantaneous frequency. In [8] and [9], Voeckler showed how to understand the instantaneous frequencies and envelope, based on the "product representation of signals" which he modeled as polynomials or rational functions. The Hilbert transform is used to compute this analytic signal. In case of a one-dimensional signal, its amplitude gives an "envelope" of the signal. Computing the frequential information from such a function is unstable in the presence of noise. For the one-dimensional case we will refer to the Canonical Factorization (see [2]) and the Blaschke product that gives us some invariance properties. This factorization can be assimilate with Voeckler idea of "product representation signals". In [3], Kumaresan proposed an extension of this work. He mentioned that the 'Min Phase', 'Max Phase' and 'All Phase' was not done by Voeckler but by Oppenheim and colleagues in chapter 12 of [6]. In the representation, the 'All-phase' is a Blaschke product and the 'Min-phase' is an outer function (that corresponds to our function G, up to a constant factor, that we will describe later). We observe, also shown by Loughin and Tacer in [4], that the sum of the derivative of the phase and the amplitude modulation can be interpreted as the instantaneous frequency.

We begin with an overview of results about the Canonical Factorization. We define an iterative algorithm to decompose an analytic signal in an orthogonal basis of finite Blaschke products. We effectively obtain the information contents in a wide variety of examples. This algorithm consists in the decomposition of a signal into different oscillatory modes.

In conclusion of our study, we attempt to extend the notion of Blaschke product to the two-dimensional case in a consistent way. We apply this algorithm to synthesized and real images.

1.2 Summary of results about Blaschke Products

In this section we define Blaschke product. We follow Zygmund [12] and Garnett [2] in this presentation.

1.2.1 Definitions

Our discussion will require the following basic definitions and properties, cf. [2] and [12] for details.

Theorem 1.2.1. *If $z \mapsto F(z)$ is regular for $|z| \leq 1$, then $\log|F(z)|$ is dominated in $|z| < 1$ by the Poisson integral of the function $\log\left|F(e^{ix})\right|$, i.e.*

$$\log\left|F(\rho e^{i\xi})\right| \leq \frac{1}{2\pi}\int_0^{2\pi} \log\left|F(\rho e^{i\theta})\right| \frac{1-\rho^2}{1-2\rho\cos(\theta-\xi)+\rho^2} d\theta. \tag{1.1}$$

Let $\xi_1, \xi_2, \xi_3, \cdots, \xi_m, \cdots$ be a sequence of points such that $0 < |\xi_m| < 1$, and that $\prod_m |\xi_m|$ converges. Then the product

$$\prod_m b(z, \xi_m) = \prod_m \frac{(z-\xi_m)}{(z-\xi_m^*)}\frac{1}{|\xi_m|}, \quad \xi_m^* = \frac{1}{\bar{\xi}_m}. \tag{1.2}$$

converges absolutely and uniformly in every disc $|z| < r < 1$ to a function $\beta(z)$, regular and bounded above by 1 in $|z| < r$, which has $\xi_1, \xi_2, \xi_3, \cdots, \xi_m, \cdots$ as its only zeros there.

Given $F \in N$, let $\xi_1, \xi_2, \xi_3, \cdots, \xi_m, \cdots$ be the zeros of F located in $\{z, |z| < 1\}$ and $\xi_k \neq 0$ for all k. If F has an additional zero of order $N \geq 0$ at $z = 0$ the expression

$$B(z) = e^{i\gamma} z^N \cdot \prod_m \frac{(z-\xi_m)}{(z-\xi_m^*)}\frac{1}{|\xi_m|} \tag{1.3}$$

where γ is any real number, is called the Blaschke product of F. If F has no zero for $\{z, 0 < |z| < 1\}$, then $B(z) = e^{i\gamma} z^N$ for such F. We have $|B(z)| \leq 1$ for $|z| < 1$ and the ratio

$$G(z) = \frac{F(z)}{B(z)} \tag{1.4}$$

is regular and has no zeros in $\{z, |z| < 1\}$. We shall always assume that γ is selected so that $G(0)$ is real and positive.

In [2] various properties of the Nevanlinna Class are described. In particular, one obtains the least harmonic upper bound of $\log|f(z)|$ of the form $\int P_z(\theta)d\mu(\theta)$, with the important relation:

$$d\mu(\theta) = \log|f(\theta)|\frac{d\theta}{2\pi} + d\mu_s(\theta) \tag{1.5}$$

with $d\mu_s$ singular i.e. orthogonal to $d\theta$. The canonical factorization theorem for functions in N is given by writing $d\mu_s$ as the difference of two positive measures $d\mu_2$ and $d\mu_1$, orthogonal to $d\theta$. With the previous definitions, we have the outer function:

$$G_1(z) = \exp\left(\int_0^{2\pi} \frac{e^{i\theta}+z}{e^{i\theta}-z} \cdot \log\left|f(e^{i\theta})\right| \frac{d\theta}{2\pi}\right) \tag{1.6}$$

And the singular inner analytic functions S_j are defined by:

$$S_j(z) = \exp\left(-\int_0^{2\pi} \frac{e^{i\theta}+z}{e^{i\theta}-z} d\mu_j(\theta)\right), \quad j = 1, 2 \tag{1.7}$$

That enables us to state the Canonical Factorization theorem:

Theorem 1.2.2. *Let $f \in N, f \not\equiv 0$. Then*

$$F(z) = C \cdot B(z) \cdot G_1(z) \cdot \frac{S_1(z)}{S_2(z)}, \quad |C| = 1 \tag{1.8}$$

where B is a Blaschke product, G_1 is an outer function and S_i are singular functions. The factorization is unique except for the choice of C.

1.3 Elementary computations of the phase of a signal

Throughout the paper, the signal f will always be a trigonometric polynomial with values in $\mathbb{R}$. We will compute an analytic function f_+ such that $Re(f_+|_{\mathbb{R}}) = f$. This function exists and can be obtained by canceling the negative frequencies of the signal $\hat{f}$, the Fourier Transform of f, so that $supp(\hat{f}_+) \subset [0, +\infty)$. The steps are the following:

- We compute $\hat{f}$, the Fourier transform of f, and preserve the positive frequencies: $\hat{f_+} = 2\,\mathcal{F}(f) \cdot 1_{(0,+\infty)} + \delta_0(\mathcal{F})(f)$. A multiplication by two is necessary to preserve the real part and we have $Re(f_+|_{\mathbb{R}}) = f$
- We compute the inverse Fourier transform: $f_+ = \mathcal{F}^{-1}(\hat{f}_+)$

Thus we have constructed a mapping from $L^2(\mathbb{R}) \mapsto L^2(\mathbb{C})$ that maps a real function f to an analytic function f_+ using the Hilbert transform:

$$f_+|_{\mathbb{R}} = f + iH(f) \tag{1.9}$$

Now we work on the unit circle, given a trigonometric polynomial f, we construct on $\mathbb{C}$ an analytic function f_+ (such that $Re(f_+|_{\mathbb{R}}) = f$). We then have holomorphic function F such that the restriction to the unit circle has the following properties:

$$\text{for each } \; \theta \in (-\pi, \pi], F(e^{i\theta}) = f_+(\theta) \tag{1.10}$$

This notation for functions will be kept throughout the paper. We have interests in the instantaneous frequencies. Thus, we will now explain how to compute it.

1.3.1 Evaluation of the instantaneous frequencies

We can obtain the Blaschke product by extracting the roots α_k of F, an analytic function on $\mathbb{C}$, with absolute value smaller or equal to 1 and denote N the order of zero as a root. We have the following identity:

$$F(z) = B(z) \cdot G(z) \,\text{with}\, B(z) = z^N \cdot \prod_{k=0}^{M} \left(\frac{z - \alpha_k}{1 - \bar{\alpha}_k z} \cdot \frac{\bar{\alpha}_k}{|\alpha_k|} \right) \tag{1.11}$$

We have, on the unit circle, $|B| = 1$ and $|G| = |F|$. Thus there exists a function $\phi : [-\pi, \pi) \to \mathbb{R}$, such that $e^{i\phi(\theta)} = B(e^{i\theta})$ and we obtain:

$$B(e^{i\theta}) \quad = \quad B(1) \cdot e^{i \int_0^\theta \phi'(t)dt} \text{ with } \phi'(\theta) = N + \sum_{k=0}^{M} \frac{1 - |\alpha_k|^2}{|e^{i\theta} - \alpha_k|^2} > 0. \tag{1.12}$$

The fact that the phase ϕ is non-decreasing means that B has the property of always turning around the origin in the same direction. The curve defined by B has a counter-clockwise trajectory. This property is not always verified by analytic functions even if by definition their Fourier Transform has only positive frequencies.

1.3.2 The classical method to compute the phase gradient

Given a trigonometric signal f_+, we compute its phase modulo (2π) and its amplitude. We "unwrap" the phase along the θ-axis and compute its derivative in order to obtain the instantaneous frequency of f. We assume that the phase does not have any jumps greater than π, or otherwise the phase could not be unique.

1.3.3 An alternative way to compute the phase gradient

Many numerical artifacts are introduced when F vanishes, making the phase computations sensitive to noise, and its phase has a discontinuity as supported on the interval $(-\pi, \pi]$. To solve these problems, we begin by computing directly the phase derivative and determine after the phase value by integration. As a complex signal, F, can be written as that follows:

$$F = |F|e^{i\phi} \in \mathbb{C}, \text{with } \phi \text{ the phase that we are interested in}$$

We have a determined ϕ if only $|F| > 0$, indeed $\phi = -i \log \frac{F}{|F|}$. We notice that a segmentation of the signal can be obtained using this criterion, the regions K_i are separated by points where $|F|$ cancels. When we have separated our connected components K_i contained in $K = \{x \in \Omega : |F(x)| > 0\}$, we compute the derivative of F according to θ and obtain the expression:

$$F' = |F|'e^{i\phi} + i\phi'|F|e^{i\phi}, \text{ where } u' \text{ corresponds to } \partial_\theta u$$

thus:

$$\frac{F'}{F} = \frac{|F|'}{|F|} + i\phi' \tag{1.13}$$

To obtain a signal with a meaningful phase, in the sense that the phase has some variations, we first extract the low frequencies from our original signal. The instantaneous frequency is considered as unchanged under this operation, since signals with low frequencies contents have very small fluctuation of their phase by definition.

Remark 1.3.1. *In the preprint "Lifting in Sobolev Spaces" the authors propose a multi-resolutions algorithm to determinate the phase of a "given function* $u : \Omega \mapsto S^1$ *(i.e.,* $u : \Omega \mapsto \mathbb{C}$ *and* $|u(x) = 1|$ *a.e.) we may write pointwise*

$$u(x) = e^{i\psi(x)}$$

for some function $\psi; \Omega \mapsto \mathbb{R}$*. The objective is to find a lifting* ψ *"as regular as u permits."*

This function ψ *is obtained by an iterative process, that computes the function at various scales from coarse to fine.*

1.4 An algorithm to compute the Blaschke Product

Let f_+ be a trigonometric polynomial of one variable, defined by the restriction of f to the positive frequencies in the Fourier space:

$$f_+ = 2 \cdot FT^{-1}\big(FT(f) \cdot \chi_{[0,+\infty)}\big), \text{ thus } Re(f_+|_{\mathbb{R}}) = f. \tag{1.14}$$

We write f_+ as a Fourier series with only positive frequencies on the circle:

$$f_+(\theta) = \sum_{k=0}^{k=M} a_k e^{ik.\theta} \tag{1.15}$$

And now on the unit disc, we have:

$$F_+(z) = \sum_{k=0}^{k=M} a_k z^k \tag{1.16}$$

We have shown that F_+ is analytic in $\mathbb{C}$. But F_+ has some zeros inside the unit circle. We want to build a new analytic function G_+ with same absolute value a F_+ on the unit circle C but no zeros inside the unit circle. Let's denote $l = \log|F_+|$, and obtain L_+ analytic such that $Re(L_+|_{\mathbb{R}}) = l$. Denote further $G_+ = e^{L_+}$ so that:

$$|G_+| = \left|e^{L_+}\right| = e^{Re(L_+)} = e^l = |F_+|\,, \text{ on } C.$$

Thus, we have now the following identity:

$$F_+ = B_+ \cdot G_+ \quad \text{with} \quad |B_+| = \left|\frac{F_+}{G_+}\right| = 1 \text{ on the unit circle } C. \tag{1.17}$$

We interpret now these two functions and show that B_+ corresponds to the Blaschke product. We will assume that F_+ has no zeros on the unit circle through the paper.

1.4.1 The interpretation of the "$B \cdot G$" decomposition

We have decomposed a holomorphic function F_+ on the complex plane as the product of two holomorphic functions B_+ and G_+. Just as F_+ the function B_+ has no zeros on the unit circle and G_+ has the same modulus as F_+ on the unit circle. We show now that this algorithm, based on the numerical values on the unit circle C, gives B_+ equal to the Blaschke product of F_+ (to a multiplicative constant c, with $|c| = 1$). We have a discrete signal F_+ with roots $\{a_i\}$ all distinct from zero. N is the order of zero as a root, thus:

$$F_+(z) \quad = \quad \lambda \cdot z^N \cdot \prod_{|a_i|<1} (z - a_i) \cdot \prod_{1<|a_i|} (z - a_i) \tag{1.18}$$

Then it is obvious that:

$$\log|F_+(z)| \quad = \quad \log|\lambda| + \sum_{|a_i|<1} \log|z - a_i| + \sum_{1<|a_i|} \log|z - a_i| \tag{1.19}$$

But we know that z belongs to the unit circle C, thus we write:

$$\log|F_+(z)| \quad = \quad \log|\lambda| + \sum_{|a_i|<1} \log|1 - z \cdot \bar{a_i}| + \sum_{1<|a_i|} \log|z - a_i| \tag{1.20}$$

And we know that inside the unit disc, for $|a| > 1$, we have $z \mapsto \log(z - a)$ analytic, thus we have:

$$Re\big(\log(z - \frac{1}{\bar{a_i}})\big) \quad = \quad \log\left|z - \frac{1}{\bar{a_i}}\right| \text{ for } |a_i| < 1 \tag{1.21}$$

$$Re\big(\log(z - a_i)\big) \quad = \quad \log|z - a_i| \text{ for } |a_i| > 1 \tag{1.22}$$

We conclude that:

$$G_+(z) \quad = \quad |\lambda| \cdot \prod_{|a_i|<1} (1 - \bar{a_i} \cdot z) \cdot \prod_{1<|a_i|} (a_i - z) \cdot \frac{\bar{a_i}}{|a_i|} \tag{1.23}$$

$$B_+(z) \quad = \quad \frac{\lambda}{|\lambda|} \cdot z^N \cdot \prod_{|a_i|<1} \frac{(z - a_i)}{(z - a_i^*)} . \frac{-1}{\bar{a_i}} \cdot \prod_{1<|a_i|} \frac{-\bar{a_i}}{|a_i|}, \text{ where } a_i^* = \frac{1}{\bar{a_i}} \tag{1.24}$$

We observe that this algorithm enables us to obtain the decomposition that conforms to the Blaschke product defined previously. As we will see below, we have thus separated the frequency and the amplitude contents of our signal F_+. The factorization above yields:

$$Phase(F_+) = Phase(B_+) + Phase(G_+) \tag{1.25}$$

In [2], Garnett shows that the same decomposition can be obtained for functions with infinitely many zeroes. After experimentation, we note that in some cases the phase of G_+

is contained in $(-\pi/2, \pi/2)$ but it is not a rule. Nevertheless, the curve defined by G_+ has the property of not surrounding the origin. On the other hand, B_+ contains complementary information. In fact its phase corresponds to the number of turns around the origin which is a topological invariant. Computing the instantaneous frequencies of B_+ is more stable as it never vanishes since $|B_+| = 1$. Moreover its phase is non-decreasing as we have seen in equation (1.12) and may lead to more stability to noise. The shortcoming of the algorithm is that we note that numerical artifacts appear when the roots of F_+ are close to the unit circle C, that corresponds to the limit case for a root to change from being a root of B_+ or G_+.

The absence of singular function

It is well known that an analytic function F is the product of an Inner and Outer function. The Inner function is composed of two factors: the Blaschke product and a ratio of two singular functions (Theorem 5.5 in [2]). Since we work with trigonometric polynomials, the measure $d\mu_s$ (that is the singular part of the weak limit of $\log|F(re^{i\theta})|$ as $r \to 1$) is equal to zero. Moreover, the trigonometric polynomial can be written as a polynomial of $z = e^{i\theta}$, as its has a Fourier series with only positive frequencies. It means that it is the product of B, with roots in the unit disc, and G with roots outside the unit disc. They are respectively rational functions and polynomials.

1.4.2 Properties of the Blaschke Product

We notice that this decomposition is similar to the Local Trigonometric bases as both decompositions enables us to have localized oscillations. For example, the local cosine transform has its oscillations parametrized (it corresponds to the spatial and frequential localization of the signal). Each term has a well defined support.

$$\forall x \in [a_k, a_{k+1})\; x \mapsto \cos\left(\frac{j \cdot (x - a_k)}{a_{k+1} - a_k}\right) \tag{1.26}$$

For the Blaschke product it will depend on the phase of the roots contained in the unit disc, as $\theta \mapsto \frac{e^{i\theta} - \lambda e^{i\theta_0}}{1 - \lambda e^{i(\theta - \theta_0)}}$ (with $\lambda \geq 0$). The oscillations are more or less concentrated depending on the distance of the root to the unit circle. The oscillations are around θ_0 and the frequencies depend on $(1-\lambda)$. If λ is equal to zero we have a low frequency by opposition to λ near one. The frequencies will be affected too, localization and oscillations are forced at the same time in both cases. We notice on figure 1.1, the localization properties of the Blaschke Product. The factorization process obtained from the family of Blaschke products is a non-linear

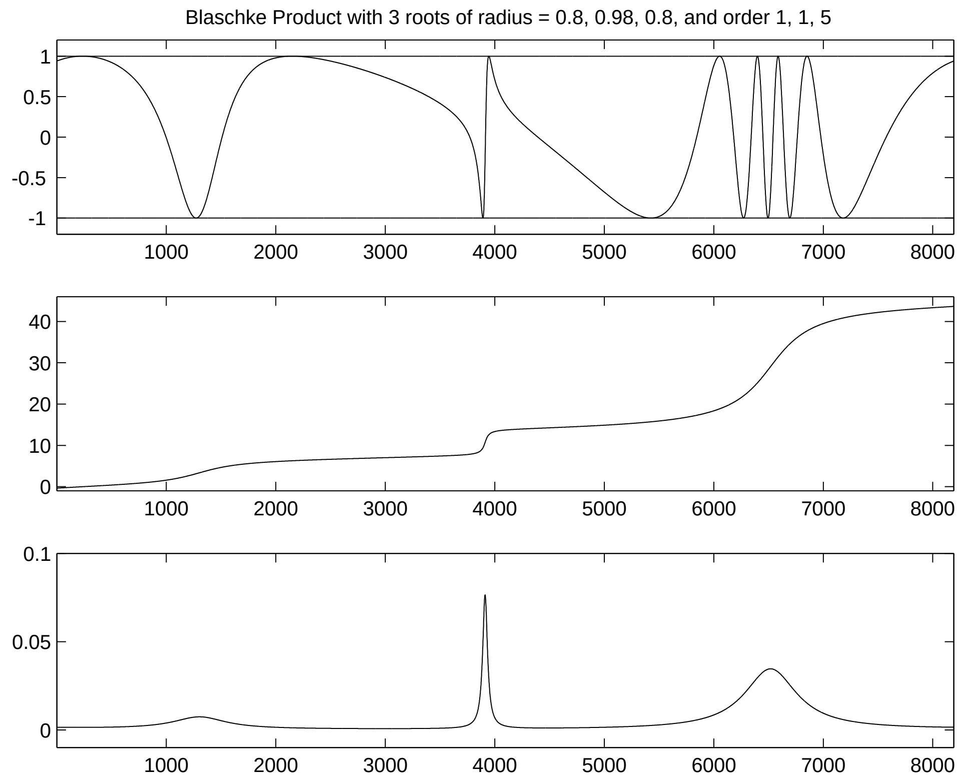


Figure 1.1: *An example of Blaschke product $B = e^{i\phi_B}$ that has three roots of different orders. The top graph shows the real part of B, while the second and third row show the phase ϕ_B and its derivative ϕ_B'. We observe that the "spread" of the oscillations depends on the distance of the root to the boundary (unit circle C). The order of the root increases the number of local oscillations.*

approximation as it will generate a function B, the Blaschke product, that is adapted to the signal. We observe that the oscillations, related to the derivative of ϕ_B (the phase of B), will depend on the local frequencies of our signal. We also notice that the localizations of the zeros can be obtained by observing the maxima of the derivative ϕ_B as shown in chapter 1.3.1. On the other hand, it has the drawback of being of absolute value equal to one. We will show in the next chapters some numerical results and necessary corrections to reduce the computational artifacts.

1.4.3 Numerical Experiments

In the previous algorithm, we apply the logarithm to the absolute value of F_+, It means that in every region where $|F_+|$ is close to zero the computations of G_+ and therefore of B_+ are going to be unstable and artifacts will appear. We can also notice that in some cases we loose analyticity of G_+ even if it corresponds to the exponential of an analytic function.

Corrections for regions where F_+ vanishes

To prevent this numerical difficulty we apply a correction, we have tried two different kinds of filtering. In the first case we smooth $\log|F_+|$ by choosing a parameter ϵ that we use as a threshold. It can be written as follows:

$$\left|F_+^{new}\right| = \sqrt{|F_+|^2 + (\epsilon \cdot \|F_+\|_\infty)^2} \tag{1.27}$$

And we have now on the unit circle:

$$|B_+| = \left(1 + \left(\epsilon \frac{\|F_+\|_\infty}{|F_+|}\right)^2\right)^{-\frac{1}{2}} \tag{1.28}$$

We conclude that

$$|B_+| \quad << \quad 1 \text{ for } F_+ << \epsilon\|F_+\|_\infty \tag{1.29}$$

$$\text{and } |B_+| \quad \sim \quad 1 \text{ for } F_+ >> \epsilon\|F_+\|_\infty \tag{1.30}$$

The result is then obvious on the left bottom graph of figure 1.2. Another possibility is to filter the function $|F_+|$ in order to “separate the function from zero”. The convolution by a positive function will raise the values in the regions where $|F_+|$ are equal to zero (as $|F_+|$ is a non-negative function), it has a similar effect to the heat kernel. We convolve $|F_+|$ with the Poisson kernel at $(1-\epsilon)$, this corresponds to multiply by $(1-\epsilon)^{|i|}$ each Fourier

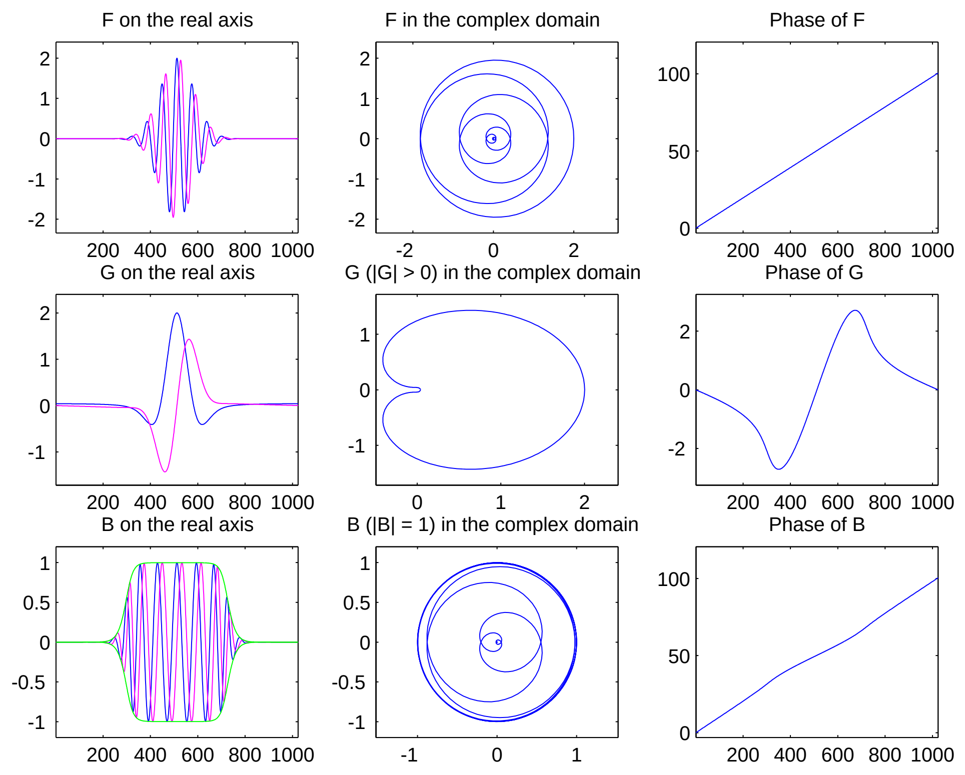


Figure 1.2: *The Blaschke decomposition for a "modulated Gaussian",* $\theta \mapsto e^{-(\theta-\theta_0)^2} \cdot e^{in\theta}$*, signal with an* ϵ *raised to show that* $|B| = 1$ *only where* F *is significantly different from zero.*

coefficient $\widehat{|F|}(i)$. The filtering effect is to "raise B_+ to one" on regions where $|F_+|$ is above the threshold and to "lower B_+ to zero" on the other regions. We observe this effect on the example where we have chosen F_+ to be a "modulated Gaussian" shown on figure 1.2. By choosing different values for ϵ we decide to "raise a smaller or bigger region for B_+". The effect of the first filtering is uniform everywhere by opposition to the application of the Poisson kernel. It means that the value of B_+ is fixed by equations (1.29) and (1.30) everywhere by the same value for ϵ. The second filtering is more adaptive as it corresponds to the first filtering but with an adaptive ϵ. Moreover, the first filtering enables us to keep the property that $|B_+|$ is smaller than one as opposed to the second filtering. The first filtering will be used in the one-dimensional case and in two dimensional case we will use the Poisson kernel.

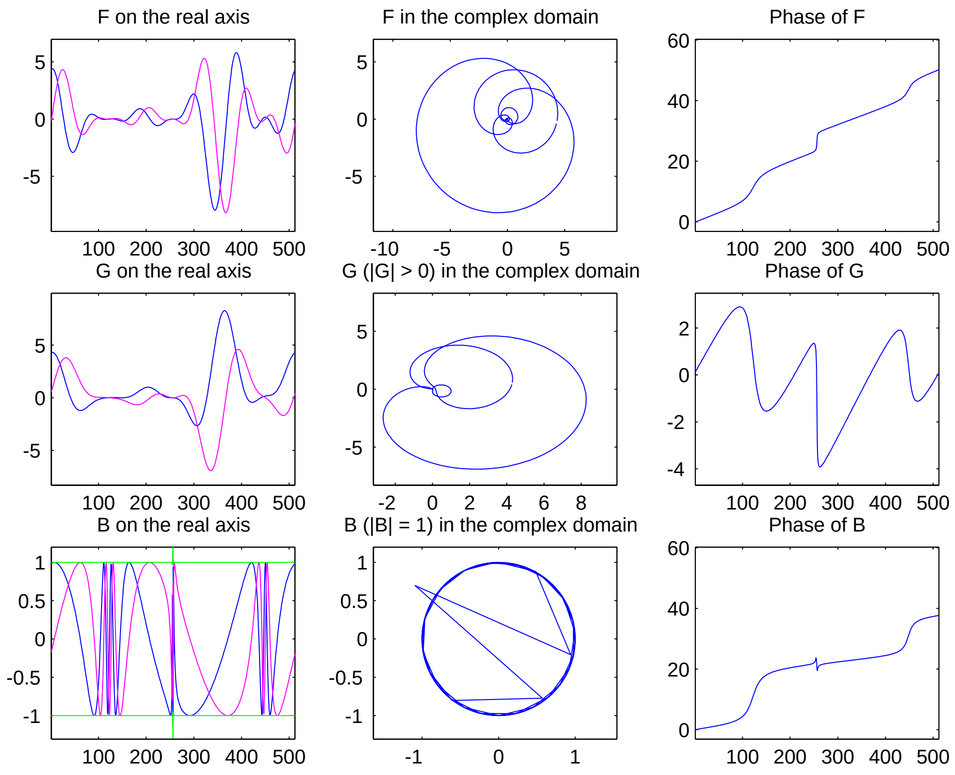


Figure 1.3: *Numerical artifacts appearing for* $\theta \mapsto (e^{i\theta} - z_1)^{p_1} \cdot (e^{i\theta} - z_2)^{p_2} \cdot (e^{i\theta} - z_3)^{p_3} \cdot g_1(\theta)$, *with* g_1 *an outer function, since the roots* z_i *are too close to the unit circle. We notice that* $|B| \neq 1$ *and the* ϕ'_B, *derivative of* B*'s phase, is not non-decreasing anymore.*

We know G_+ is an analytic function as the exponential of an analytic function. Nevertheless artifacts appear in this computation. We observe on the example shown on figure 1.3 the numerical artifacts that appear clearly in the phase of B_+ and modify the relation $|B_+| = 1$. A Fourier transform will confirm the loss of analyticity (we have high negative frequencies different from zero) and also we easily notice that B has an absolute value different from one in some points. This numerical errors interfere with our results especially for the iterative decomposition that we apply later.

Corrections to compute the logarithm and exponential

The Blaschke product is related to the roots of F_+ living in the unit disc D. These two functions have in common all the roots included in D. The poles of B_+ have an absolute value greater than one. We want to see how accurate is the computation of B_+ knowing F_+. Since B_+ is a product, we restrict our study to a simple real root in the unit disc D, and the problem is invariant by rotation so the root can be taken on the interval $[0, 1)$. We

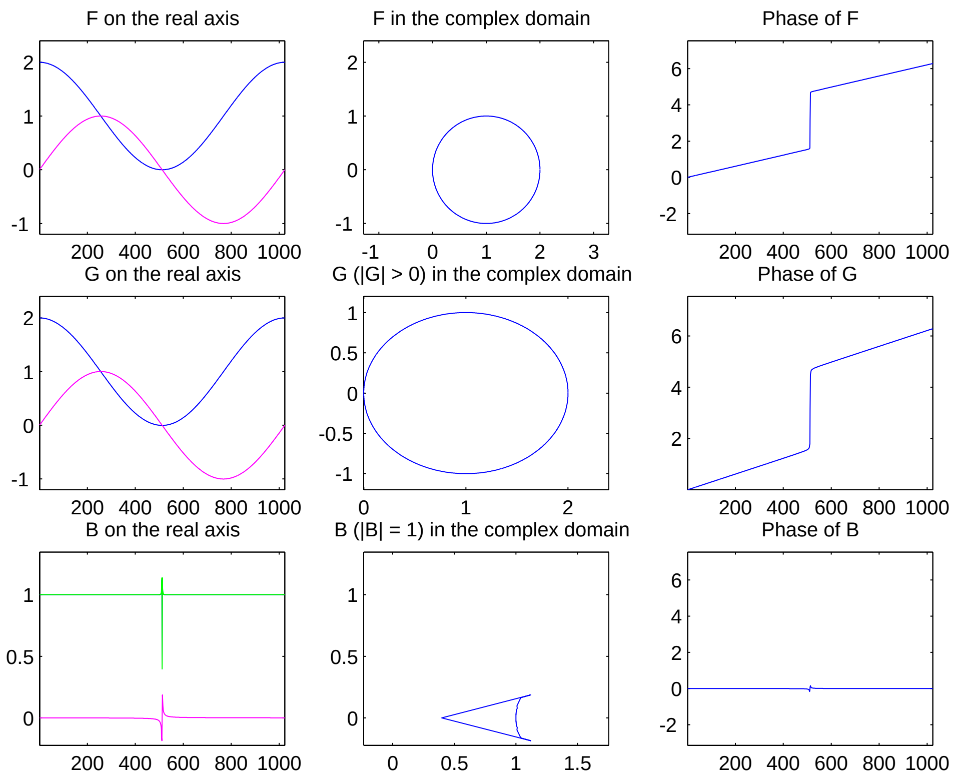


Figure 1.4: *The function $\theta \mapsto e^{i\theta} - r$, with r close to one, and its $B_+ \cdot G_+$ decomposition, where it appears that $|B_+|$ is not equal to one everywhere*

are working on a simple example, shown on figure 1.4, the binomial defined by:

$$F_+(z) = z - r, \text{ with } 0 \leq r < 1. \tag{1.31}$$

We know that the decomposition "$F_+ = B_+ \cdot G_+$" is obtained with:

$$B_+(z) = \frac{z - r}{1 - rz} \quad \text{and } G_+(z) = 1 - rz \tag{1.32}$$

We notice that if r is close to one the function $\log|F_+|$ goes to minus infinity and generates artifacts in the complexification process. It will create some high negative frequencies that should be equal to zero as an analytic signal. In fact, the signal is suddenly going to turn much faster around the origin and if the signal is not oversampled enough, it will mean a jump greater than π for the phase. Some artifacts are obtained in the Fourier decomposition of G_+. As we know the exponential of an analytic function is also analytic, so the non-zero coefficients appearing for the negatives frequencies are obviously artifacts. And as a consequence, we have $|B_+| \neq 1$ and its phase will not be non-decreasing We observe

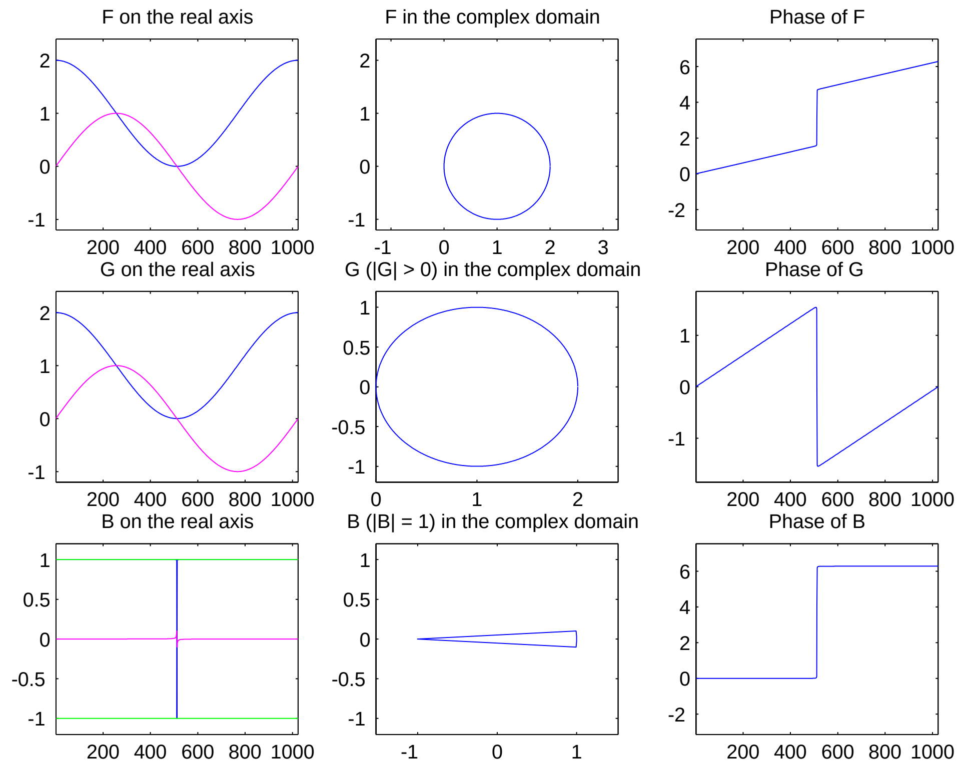


Figure 1.5: *The function $\theta \mapsto e^{i\theta} - r$ and its $B_+ \cdot G_+$ decomposition, where the artifacts disappear using an oversampling of four.*

on figure 1.5 that oversampling the function, applying a zero padding in the Fourier space, makes the negative frequencies tend to cancel. We keep the same trigonometric polynomial but oversample the signal to avoid artifacts. We notice that G_+ is a polynomial of degree smaller or equal than F_+, then we truncate the Fourier expansion of our signals L_+ and G_+ at a rank R (that we denote by $(\cdot)_R$) to obtain $(L_+)_R$ and $(G_+)_R$ according to the degree of F_+, and we have:

$$(L_+)_R = \mathcal{F}^{-1}(\mathcal{F}(L_+) \cdot 1_{[0,R]}) \tag{1.33}$$

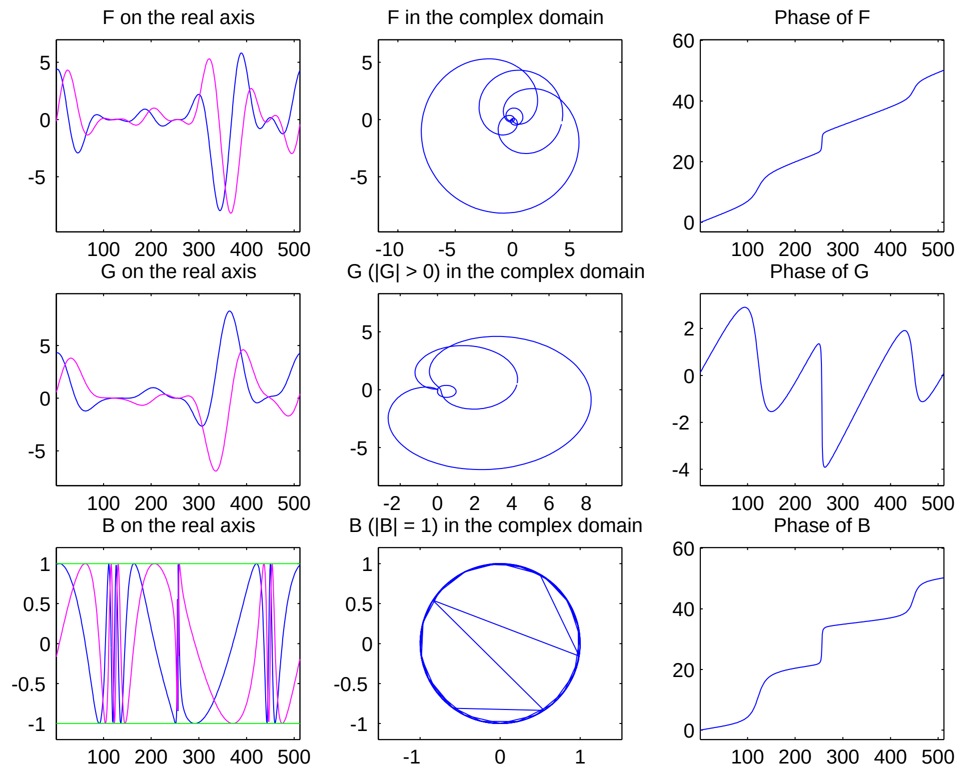


Figure 1.6: *The function $F_+ : \theta \mapsto (e^{i\theta} - z_1)^{p_1} \cdot (e^{i\theta} - z_2)^{p_2} \cdot (e^{i\theta} - z_3)^{p_3} \cdot g_1(\theta)$, from figure 1.3, oversampled has now the desired decomposition "$B_+ \cdot G_+$": $|B_+| = 1$ and $B_+ = e^{i\phi_B}$ and ϕ_B is non-decreasing (bottom row). We observe that ϕ_B has jump of 2π where the discontinuity was before on figure 1.3.*

The degree of G_+ is smaller or equal than the degree of F_+. As we write the decomposition of "$F_+ = B_+ \cdot G_+$", where we note N the order of zero (as a root):

$$\begin{aligned} B_+(z) &= \lambda z^N \cdot \prod_{0<|a_i|<1} \frac{(z-a_i)}{(1-\bar{a}_i z)} && (1.34)\\ \text{and } G_+(z) &= \prod_{|a_i|>1} (z-a_i) \cdot \prod_{0<|a_i|<1} (1-\bar{a}_i z) && (1.35)\\ \Rightarrow deg(G_+) &= deg(F_+) - N, \text{ with } N \geq 0. && (1.36)\end{aligned}$$

We truncate L_+ and G_+ at the same degree R. And we have:

$$(G_+)_R = \Big(\exp\big((\log|F_+|)_+\big)_R\Big)_R \qquad (1.37)$$

We conclude from this experience, figure 1.6, that we obtain a better result by oversampling a trigonometric polynomial function to compute L_+. Truncate the signals in the Fourier space to the same order as the original signal F_+, before and after applying the exponential to these two steps, is also necessary to keep the non-decreasing phase of B_+.

But the possible presence of zeroes close to boundary, the unit circle C, is a major problem for this algorithm. We have shown that oversampling solves it up to a point. No matter how much we oversample, if the root is close enough to the boundary, we will have aliasing ("close" of course depends on the amount of oversampling). So this "solution" to the problem is merely a convenience to allow computations for signals having roots closer to the boundary than we could with no oversampling, but it doesn't resolve the difficulty.

Stability of the zeroes contained in the unit disc

We choose, as an example, an analytic function with three distinct roots in the unit disc. We apply different bells (modifying the support of the signal) to quantify the stability of the positions of the zeroes. We have shown before that the derivative of B's phase is a sum of Poisson kernels (related to the root of B_+), in chapter 1.3.1. In chapter 1.4.2, we showed that the phase of the roots is related to the maxima of ϕ'_B,the first derivative of the phase. While we had the oversampling to eliminate artifacts, we observed that the position of the zeroes was stable with the oversampling of the signal. We also observe the following paradox: the computation of B_+ is global, as we use the Fourier transform, but

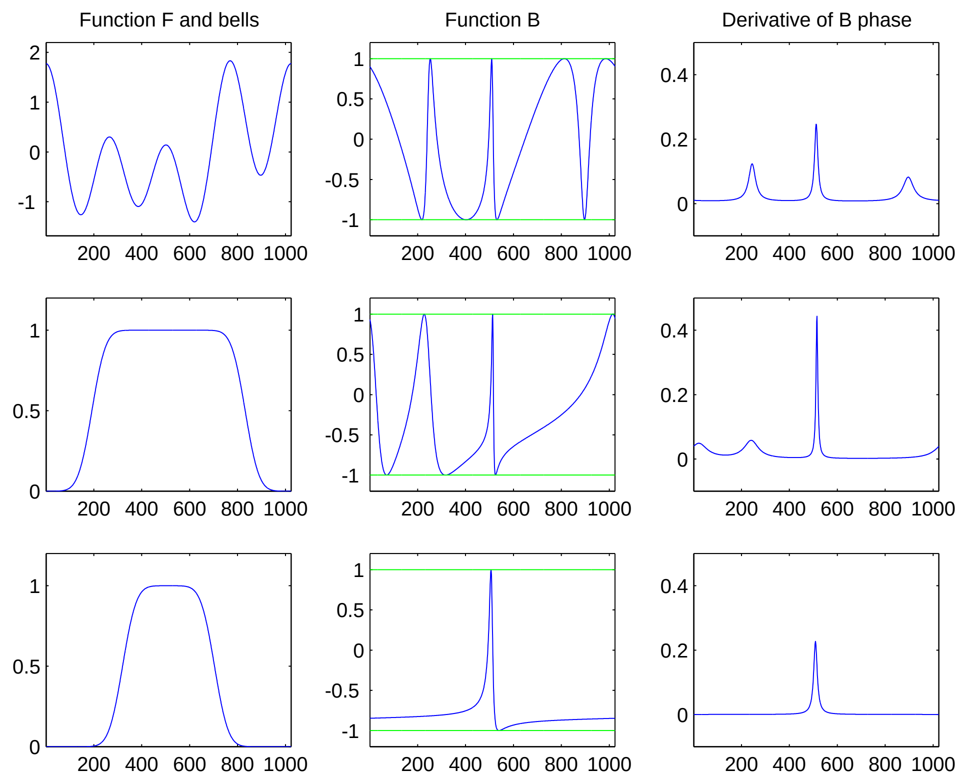


Figure 1.7: *The first row show the decomposition of the F the complexified version of the function f. We have $F : z \mapsto \prod_{i=1}^{3}(z - z_i)$, with three distinct zeroes z_i in the unit disc, represented by the real part of its values on the unit circle C. The second and third row show the decomposition of the complexification of $\theta \mapsto f(\theta) \cdot Bell(\theta)$, where the Bells are represented on the left columns below the original signal. The three zero locations appear obvious in the first graph representing the derivative of B's phase. We notice on the third column that the location of the central zero has not been affected by the bells.*

the information obtained on the phase is local as it corresponds to the roots' location. For example, as shown on figure 1.7, if we modify the support of a function by multiplying the signal by bells of different sizes (equal to zero outside and one in the center part, as shown on the two left graphs below the function F_+, for the first graph the bell is not shown as we do not apply any one). We have new function $\theta \mapsto f(\theta) \cdot Bell(\theta)$. We then observe that the phase, and furthermore the roots of B_+ are not very sensitive to the support of the bell (if the root occurs where the bell is 1).

1.5 Properties of $F_+ = B_+ \cdot G_+$

In the beginning of this chapter, ee look at the effects of an eventual filtering of F_+. For simplification we study the properties on the upper half plane that corresponds to another representation, knowing that there exists a conformal mapping between the unit disc and the upper half plane. After this study, we look at the properties of the numerical Blaschke product, defined as before, on some trigonometric polynomials to enable the reader to develop an intuition about the "$B \cdot G$" factorization. And we finish the chapter by showing the robustness of the algorithm and its property of invariance.

1.5.1 Phase properties on the upper half plane

Let $f \in L^2(\mathbb{R})$, we have constructed f_+ in the Fourier space with the following formula:

$$f_+(x) = 2 \cdot \int_0^\infty e^{ix\cdot\xi} \hat{f}(\xi) d\xi, \quad x \in \mathbb{R}. \tag{1.38}$$

We can also define F_+ on the upper half plane as follows:

$$F_+(z) = 2 \cdot \int_0^\infty e^{iz\cdot\xi} \hat{f}(\xi) d\xi, \quad z \in \mathbb{C} \tag{1.39}$$

And we have $Re(f_+|_{\mathbb{R}}) = f$. Also we assume $|F_+(z)| > 0$ for real z and we define G_+ as before:

$$G_+(z) = \exp(\log |F_+(z)|)_+ \tag{1.40}$$

It can be written as $G_+ = \exp(L_+)$ where $L_+(z) = l(z) + i \cdot \tilde{l}(z)$ and $l(z) = \log |F_+(z)|$ for real z, and $B_+ = \frac{F_+}{G_+}$. As B_+ is defined as the Blaschke product for the decomposition of F_+, we have:

$$B_+(z) = \left(\frac{z-i}{z+i}\right)^k \cdot \prod_j \frac{z-a_j}{z-\bar{a}_j} \cdot \frac{\left|a_j^2+1\right|}{a_j^2+1} \tag{1.41}$$

where $a_j = \alpha_j + i\beta_j$ are the roots of F_+ in the upper half plane. ϕ, the phase of B_+ verifies the relation $i\phi' = \frac{B'_+}{B_+}$, with $u' = \frac{du}{d\theta}$, and a simple calculation shows that :

$$\phi'(x) = 2 \sum_j \frac{\beta_j}{|x-a_j|^2} > 0 \tag{1.42}$$

We denote the following projection:

$$P_\epsilon f(z) = F_+(z + i\epsilon) \tag{1.43}$$

We note F_ϵ, l_ϵ and K_ϵ the respective projections of the functions F, l and $K = \log|F|$. We observe that we have:

$$|F_+(x+i\epsilon)| \quad = \quad \prod_j \left| \frac{x+i\epsilon - a_j}{x+i\epsilon-\bar{a_j}} \right| \cdot \exp(l_\epsilon(x)) \tag{1.44}$$

And also:

$$\begin{aligned}
K_\epsilon(x) &= \frac{1}{2}\sum_j \log\left(\frac{(x-\alpha_j)^2+(\epsilon-\beta_j)^2}{(x-\alpha_j)^2+(\epsilon+\beta_j)^2}\right) + l_\epsilon(x) && (1.45)\\
\tilde{K}'_\epsilon(x) &= H(K'_\epsilon)(x) && (1.46)\\
\tilde{K}'_\epsilon(x) &= \sum_j \frac{|\epsilon+\beta_j|}{(x-\alpha_j)^2+(\epsilon+\beta_j)^2} - \frac{|\epsilon-\beta_j|}{(x-\alpha_j)^2+(\epsilon-\beta_j)^2} + \tilde{l_\epsilon}'(x) && (1.47)
\end{aligned}$$

Therefore we get that:

$$\begin{aligned}
Im\Big(\frac{F'(x+i\epsilon)}{F(x+i\epsilon)}\Big) &= \tilde{l}'(x+i\epsilon) + \sum_j \frac{(\beta_j+\epsilon)}{(x-\alpha_j)^2+(\beta_i+\epsilon)^2} && (1.48)\\
&- \sum_j \frac{|\beta_j-\epsilon|}{(x-\alpha_j)^2+(\beta_i-\epsilon)^2}\\
&+ 2\sum_j \frac{(\beta_j-\epsilon)}{(x-\alpha_j)^2+(\beta_i-\epsilon)^2}
\end{aligned}$$

Since $F(z+i\epsilon)$ has zeroes $(a_j - i\epsilon)_j$, for $\beta_j > \epsilon$.

$$\begin{aligned}
Im\Big(\frac{F'(x+i\epsilon)}{F(x+i\epsilon)}\Big) &= \tilde{l}'(x+i\epsilon) + \sum_{\beta_j>\epsilon} \frac{(\beta_j+\epsilon)}{(x-\alpha_j)^2+(\beta_j+\epsilon)^2} && (1.49)\\
&+ \sum_{\beta_j>\epsilon} \frac{(\beta_j-\epsilon)}{(x-\alpha_j)^2+(\beta_i-\epsilon)^2}\\
&- \sum_{\beta_j<\epsilon} \left[\frac{(\beta_j+\epsilon)}{(x-\alpha_j)^2+(\beta_j+\epsilon)^2} - \frac{|\beta_j-\epsilon|}{(x-\alpha_j)^2+(\beta_i-\epsilon)^2}\right]
\end{aligned}$$

And if we ϵ-filter the phase of B_ϵ we get :

$$2 \sum_{\beta_j > \epsilon} \frac{\beta_j}{(x-\alpha_j)^2 + \beta_j^2} \tag{1.50}$$

We have shown that by filtering we can eliminate the roots of B_ϵ that are to close to the real axis. They correspond to the roots contained in the unit disc D near the boundary C.

1.5.2 Examples of "$B \cdot G$" decomposition

In this chapter we present two examples showing the decomposition described before. These examples are created to show properties linked to this study and to give to the reader an better intuition about the Blaschke product. The first example is given by figure 1.8, that represents the factorization of two trigonometric monomials with disjoint supports, different amplitudes and frequencies. We easily see that the phase of B_+ and F_+ are similar but B_+ has the property of having a modulus equal to one.

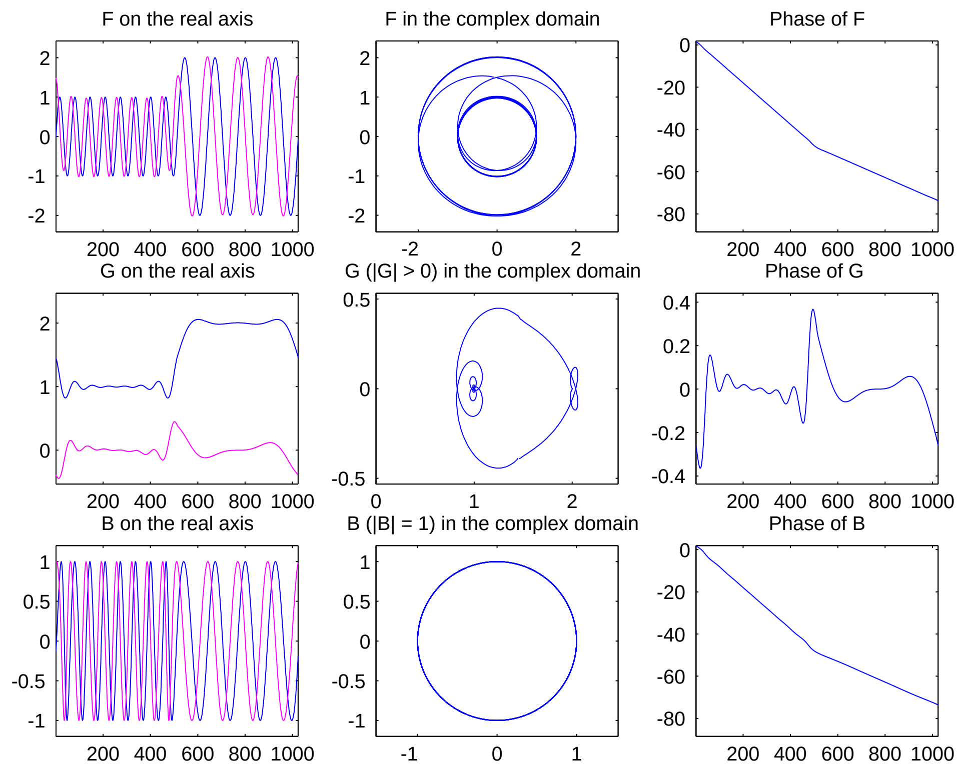


Figure 1.8: *A simple signal composed of two trigonometric monomials,* $\theta \mapsto e^{i2n_1\theta} \cdot 1_{I_1} + 2e^{in_1\theta} \cdot 1_{I_2}$*, where* I_1 *and* I_2 *correspond to the first and second half of the signal. On the bottom row, we observe that* B_+ *is keeping the frequential information given by* F_+*. The middle row shows* G_+ *living in the right half plane.*

B_+ can be compared to $\frac{F_+}{|F_+|}$ but it has the particularity of being an analytic function. Moreover we can observe on the figure that G_+ is living in the right half plane and has then it phase is in the small range $[-\pi/2, \pi/2]$ and is not surrounding the origin O as B_+ does. The winding information is contained by B_+. The second example, on figure 1.9 shows two chirps with distinct support, amplitudes and frequencies range as before. We observe as

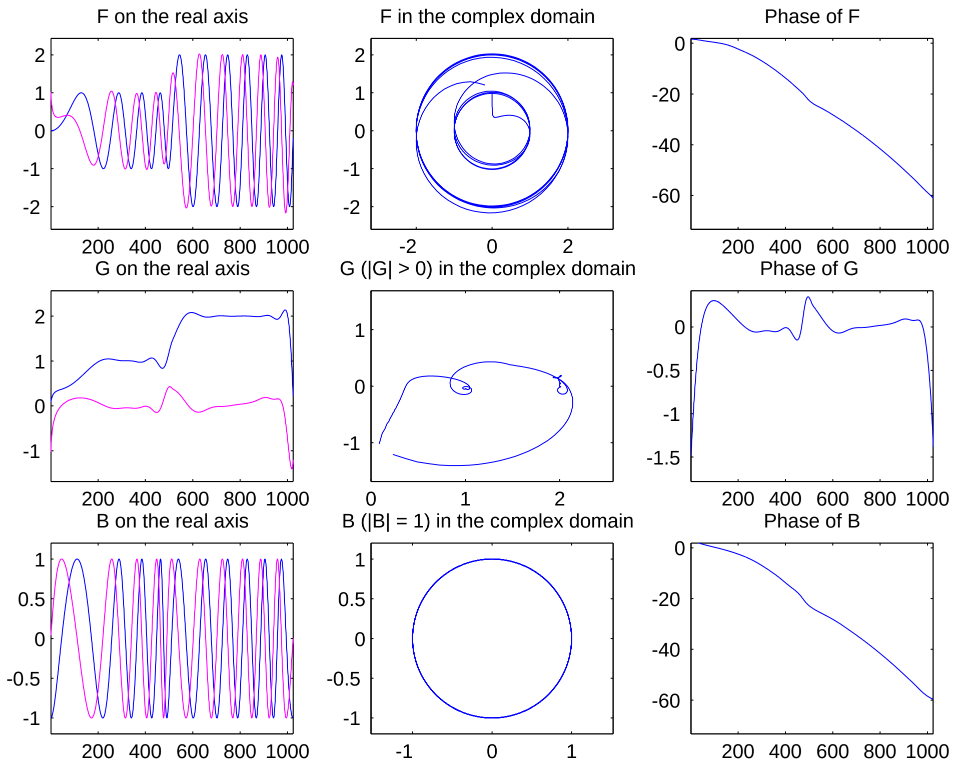


Figure 1.9: *A second signal composed of two chirps,* $\theta \mapsto e^{i2n_1\theta^2} \cdot 1_{I_1} + 2e^{in_1\theta^2} \cdot 1_{I_2}$, *where* I_1 *and* I_2 *correspond to the first and second half of the signal. As the previous figure showed* B_+ *contains all the frequential information while* G_+ *lives in the right half plane.*

before that G_+ is living in the right half plane and B_+ is similar to $\frac{F_+}{|F_+|}$. Using figures 1.8 and 1.9, as a first conclusion we can say that the effect of factorization on F_+ to give B_+ is quite intuitive, and we observe that the phases of F_+ and B_+ are similar in both cases.

1.5.3 Examples of robustness to noise

We show now on different examples that the factorization is robust to noise in the sense that the Blaschke product is invariant. We apply the decomposition algorithm on the examples previously shown on figure 1.8 and 1.9 but we generate additive or multiplicative noise. We easily notice, in the new figures 1.10 and 1.11, that B_+ stay the same while the noise affects F_+ and G_+. The function G_+ seems to "attract the noise" and leave B_+ clean. A possible

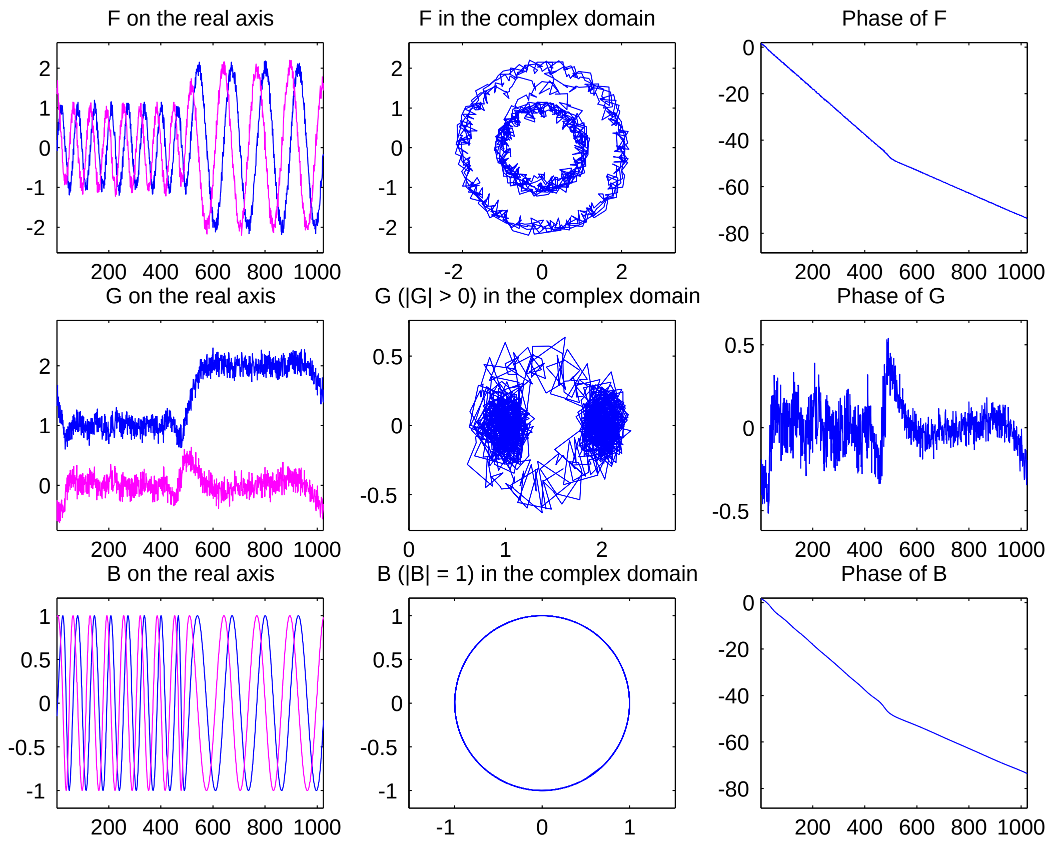


Figure 1.10: *Decomposition of the two trigonometric monomials from figure 1.8 with additive noise. B seems to not be affected by the noise while G_+ is keeping all the noise.*

explanation comes from the fact that the phase of B_+ is non-decreasing. But no reasonable explanation seems to be obvious. A possible interpretation is given by the fact that the winding number is a topological invariant. Since the Blaschke product is directly related to it, its robustness to noise is the consequence. Similar results appear on the chirps, B_+ seems

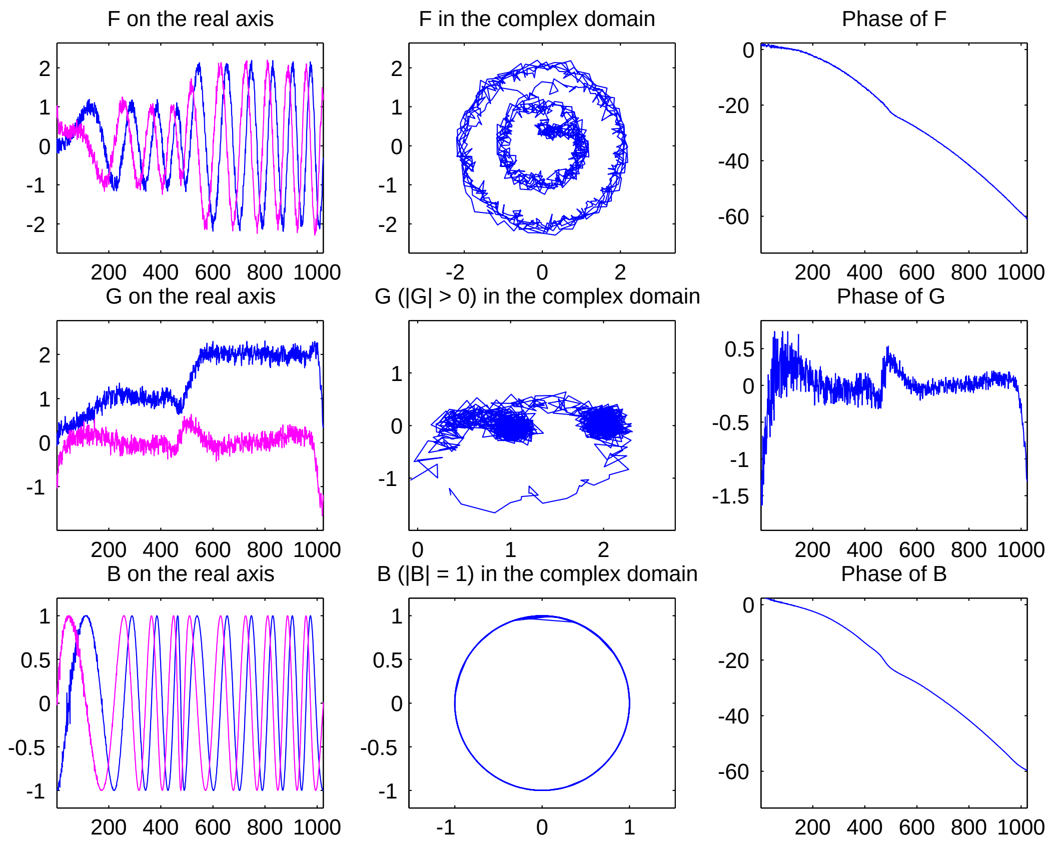
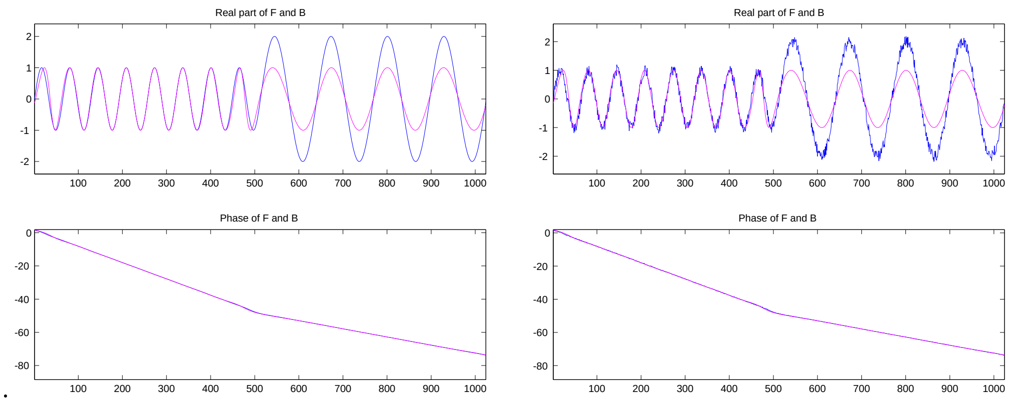


Figure 1.11: *Decomposition of the two chirps from figure 1.9 with additive noise. B seems to be invariant to this noise.*

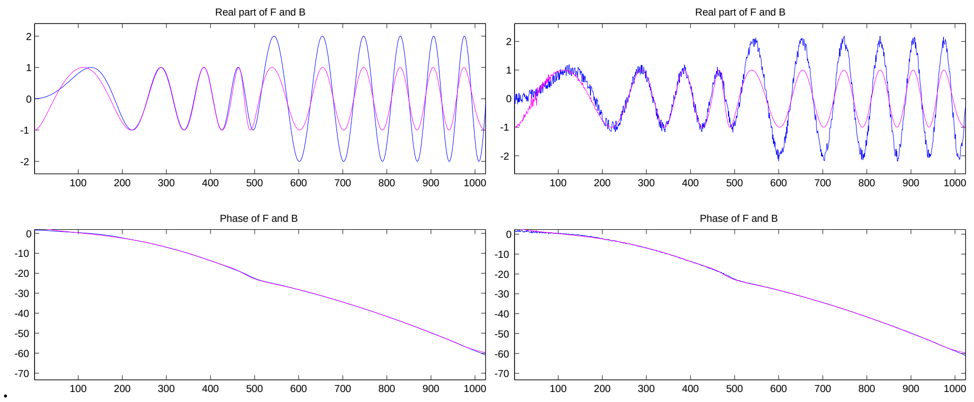
to be without any noise. If we represent F and B on the same graph we can observe that B_+ is like a metronome. The rhythm seems to be not sensitive to the noise as it appears on the following graphs that resume the study of the noise effect.

As a first conclusion, we can say that the effect of the noise is obvious on the functions G_+ by contrast to B_+ that seems to be identical to the function obtained before without noise. We conclude that B_+, obtained by factorization, is not sensitive to noise and so by consequence its phase either.



.

Figure 1.12: *The Real parts of the Blaschke products and the first signal with and without noise (shown in figures 1.8 and 1.10), and their phases.*



.

Figure 1.13: *The Real parts of the Blaschke products and the first signal with and without noise (shown in figures 1.9 and 1.11), and their phases.*

1.5.4 Examples of invariance

After showing some properties of stability with noise for the phase, we are taking the example of a family of deformed functions on the unit circle (parametrized by $\theta \in [-\pi, \pi)$ and $z = e^{i \cdot \theta}$) such as $\theta \mapsto sign(sin(k \cdot \theta^2))$.

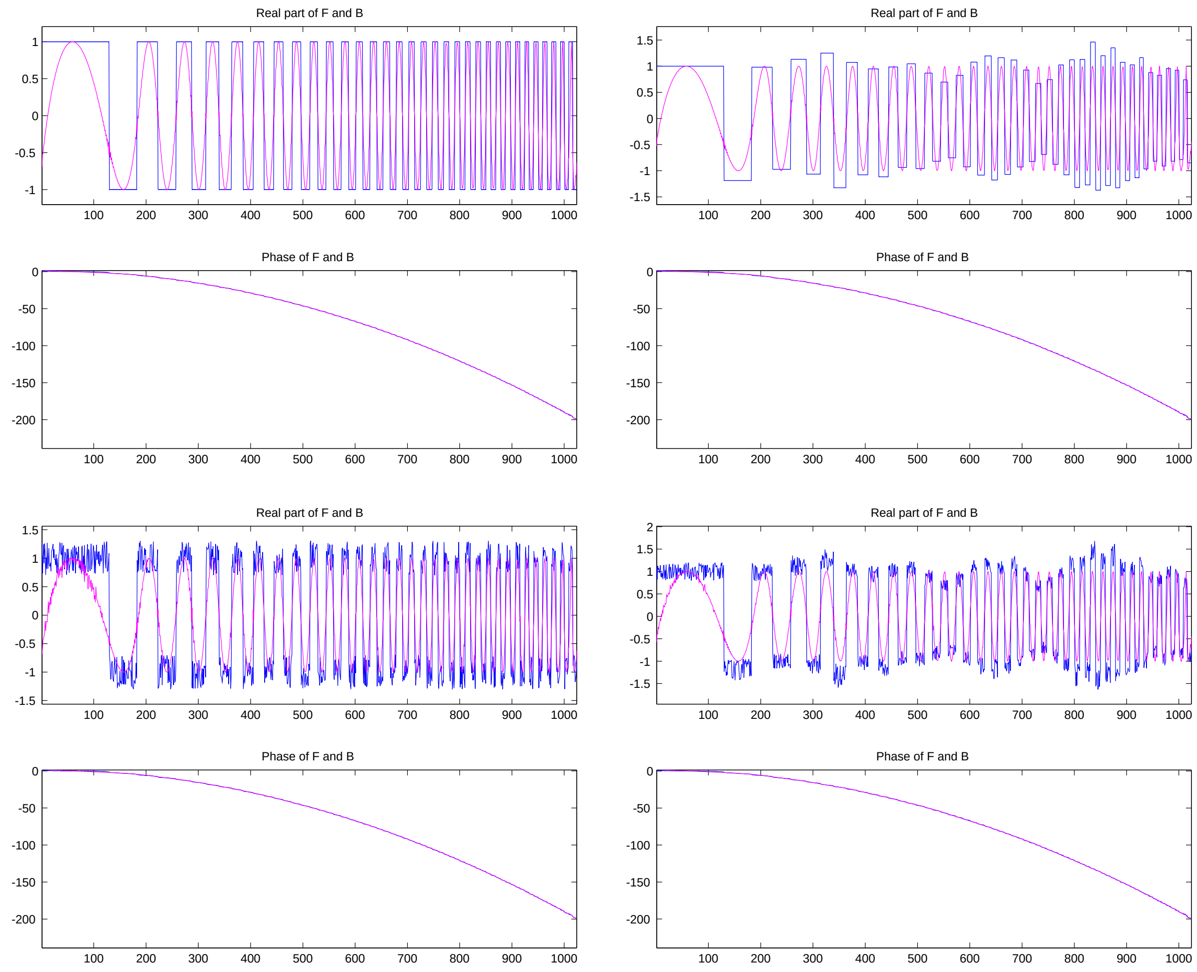


Figure 1.14: *The original real signal $\theta \mapsto sign(sin(k{\cdot}\theta^2))$ and its three different perturbations plotted separately with the real parts of the Blaschke products. The phase of the four signals are also plotted below*

We can observe that each B_+, for this family of functions, is an invariant. B_+ corresponds to the metronome of a zero crossing counter.

1.6 Orthogonal Decomposition into Blaschke products

In this section, we develop an idea to represent and understand analytic signals with a finite orthogonal decomposition. In [7], Szabo et al. presents an "orthonormal basis on $\mathcal{L}^2(\mathbb{R})$ generated by a finite Blaschke product". Our decomposition is based on an iterative method similar to the Gram-Schmidt orthogonal decomposition, some recalls and details are given in [11] by Walsh. We use his remarks to develop a non-linear algorithm: construct an orthonormal decomposition with Blaschke products. But the main difference is that the examples given by Walsh are related to a basis construction and we are interested in an orthonormal series, with a high rate of convergence and without interest in the completeness. This method can be interpreted as a matching pursuit that differs from a classical best basis approach. We present some simple examples to develop the intuition of the reader and explain how the algorithm works (pealing the different layers of the signal). We then apply the decomposition on a random trigonometric polynomial to show is effectiveness. For a family of functions, with a uniform distribution on the unit sphere, we compute a lower bound of the average projection on the Blaschke products family. We finish our examples with a case for which our algorithm is not adapted at all and present an alternative.

1.6.1 Rational Orthogonal Basis

One can orthogonalize the set of rational polynomials, as shown in chapter IX of [11]:

$$1, \frac{1}{z-\alpha_1}, \frac{1}{z-\alpha_2}, \cdots, \frac{1}{z-\alpha_n} \text{ with } |\alpha_i| > 1 \tag{1.51}$$

We have to recall that the functions f analytic on and within the unit circle C, vanishing at the point $z = \frac{1}{\bar{\alpha}}$ with $|\alpha| > 1$, are orthogonal on C to the function $z \mapsto \frac{1}{z-\alpha}$. And the Gram-Schmidt orthonormalisation process, in $L^2(|z| = 1, \frac{d\theta}{2\pi})$, gives the following decomposition:

$$1, \frac{z}{z-\alpha_1}, \frac{z \cdot (1-\bar{\alpha}_1 \cdot z)}{(z-\alpha_1)\cdot(z-\alpha_2)}, \cdots, \frac{z \cdot (1-\bar{\alpha}_1 \cdot z) \cdots (1-\bar{\alpha}_{n-1} \cdot z)}{(z-\alpha_1)\cdots(z-\alpha_n)} \tag{1.52}$$

We observe that the Blaschke Products are a key factor in this orthonormal decomposition. We are developing this idea in the following chapter to construct matching pursuit with Blaschke products.

1.6.2 Informal description of the method

We want to decompose an analytical signal f in an orthogonal sum of Blaschke products, with the factorization defined before. We develop an iterative algorithm that will extract the main oscillations of our signal in different steps.

Remark 1.6.1. *A good interpretation of this algorithm is given by looking at the rotation of the Moon turning around the Sun. We manage to separate the two rotations, the Moon around Earth and the Earth around the Sun, by running this iterative algorithm.*

At the first iteration we subtract the constant part of f on C (that is to $f(0) = \int_0^{2\pi} f(e^{i\cdot\theta})\frac{d\theta}{2\pi}$). We obtain $f_1 = f - f(0)$ that has obviously zero has a root, it enables us to decompose $z \mapsto f_1(z)/z$ (that is analytical) with the product of b_1 and g_1, where b_1 is the Blaschke product and g_1 an outer function. We iterate the previous two steps on g_1 and so forth. For notation reasons, we denote $g_0 = f$, and we have the following algorithm:

$$\begin{aligned} f = g_0 &= g_0(0) + z \cdot b_1 \cdot g_1 \\ g_1 &= g_1(0) + z \cdot b_2 \cdot g_2 \\ &\cdots \\ g_{n-1} &= g_{n-1}(0) + z \cdot b_n \cdot g_n \end{aligned} \tag{1.53}$$

Remark 1.6.2. *As g_i is an outer function, for all $i > 0$, we know that $g_i(0) \neq 0$ (for all $i > 0$) since g_i is an outer function.*

By summation we obtain:

$$\begin{aligned} g_0 = g_0(0) &+ g_1(0) \cdot z \cdot b_1 + \cdots + g_{n-1}(0) \cdot z^{n-1} \cdot b_1 \cdot b_2 \cdots b_{n-1} \\ &+ z^n \cdot b_1 \cdot b_2 \cdots b_n \cdot g_n \end{aligned} \tag{1.54}$$

Equation (1.54) is composed of a residual (the last right hand term) and an approximation at the level $(n-1)$ that is g_0 minus the residual. We will now prove that the approximation, just defined, is an orthonormal decomposition and that we have convergence in norm of the residual to zero. On the unit circle C, the scalar product of two terms of g_0's decomposition is:

$$\begin{aligned}&\forall (p,n) \in (\mathbb{N}^*)^2, p < n, \\ &< z^p \cdot b_1 \cdots b_p, z^n b_1 \cdots b_n >= \oint_C (\bar{z}^p \cdot \bar{b_1} \cdots \bar{b_p}) \cdot (z^n \cdot b_1 \cdots b_n) d\mu,\end{aligned} \tag{1.55}$$

But we know that $|b_j| = 1$, thus:

$$< z^p \cdot b_1 \cdots b_p, z^n \cdot b_1 \cdots b_n >= \frac{1}{2\pi i} \cdot \oint_C z^{n-p} \cdot b_{p+1} \cdots b_n \frac{dz}{z} \tag{1.56}$$

We know that the poles are outside C and $n - p \geq 1$, so it implies:

$$< z^p \cdot b_1 \cdots b_p, z^n \cdot b_1 \cdots b_n >= 0 \tag{1.57}$$

The L^2-norm is verified as we have:

$$\begin{aligned} \forall n \in \mathbb{N}^* < z^n \cdot b_1 \cdots b_n, z^n \cdot b_1 \cdots b_n > &= \frac{1}{2\pi i} \oint_C |b_1 \cdots b_n|^2 \frac{dz}{z} \\ \Rightarrow \quad \|z^n \cdot b_1 \cdots b_n\|_2 &= 1 \end{aligned} \tag{1.58}$$

Therefore we have the orthonormality of the series. We are now interested in the evaluation of the residual part of the decomposition. By construction we know that $|b_i| = 1$ (where f doesn't vanish numerically), so $\|b_i\| = 1$, and we want to show that $\|g_n\|$ converges to zero as n goes to infinity. And so if we denote $approx_n$ the approximation of f with n vectors we have :

$$\begin{aligned} approx_n(z) &= \sum_{i=0}^{n} c_i \cdot \prod_{k=1}^{i} z \cdot b_k(z), \text{ where } , c_i = g_i(0) > 0 \\ \Rightarrow \|f - approx_n(z)\|_2 &= \|\left(\prod_{k=1}^{n+1} z \cdot b_k(z)\right) \cdot g_{n+1}\|_2 \\ \Rightarrow \|f - approx_n(z)\|_2 &= \|g_{n+1}\|_2 \end{aligned} \tag{1.59}$$

We have shown before the orthogonality of the decomposition it means that we can apply Pythagore's Theorem:

$$\|f\|_2^2 = \sum_{i=0}^{n} c_i^2 + \|g_{n+1}\|_2^2 \tag{1.60}$$

Remark 1.6.3. *An alternative way to write this algorithm is to split the function f in two parts using the low frequency content instead of just the constant term. And we observe that the convergence rate is improved.*

1.6.3 Properties of the iterative algorithm

We have shown in chapter 1.4.2 some of the properties of the Blaschke product and its similarities to the local trigonometric basis. We observe that the iterative algorithm, that we just described, produces a lacunary series. Since, each time a new term is computed by induction, it contains the previous term and is multiplied by z and a Blaschke product. And two consecutive terms, the p^{th} and $(p+1)^{th}$ terms, of the decomposition are written as follows:

$$z^p \cdot b_1 \cdots b_p \quad \text{and } z^{p+1} \cdot b_1 \cdots b_p \cdot b_{p+1} \tag{1.61}$$

The ratio is obviously $z \cdot b_{p+1}$, where b_{p+1} corresponds to the roots inside the unit disc (b_{p+1} can be equal to one or corresponds to multiple roots inside the unit circle C). By building a lacunary series the algorithm creates a decomposition using fewer terms than what is necessary to have an orthonormal basis. The fast convergence and non-completeness of the algorithm are some of these consequences.

1.6.4 Some examples of decomposition

In this chapter we show the advantages and properties of such a decomposition for different examples. We choose them to explain how this algorithm works and more precisely the recursive algorithm that extracts the different signal layers. The three examples show how the algorithm works.

Three elementary examples

The first example is composed of a trigonometric binomial where each term has a different amplitude ($\theta \mapsto c_0 e^{im_0\theta} + c_1 e^{im_1\theta}$). In the complex plane, the signal can be interpreted as the

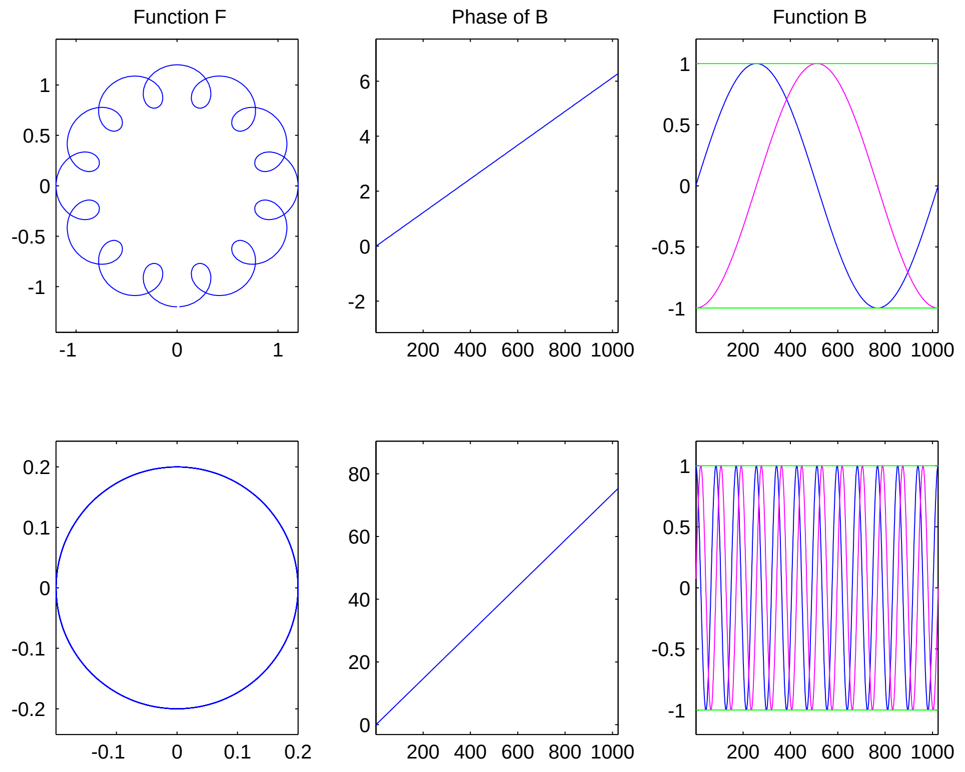


Figure 1.15: *Decomposition of $F_+ : z \mapsto z \cdot (1 + 0.2 \cdot z^n)$ with two main oscillations, B_+^1 and B_+^2 are two oscillating functions with different radius and speed. The main oscillation is extracted at the first iteration (top row), while the second step extracts the minor amplitude oscillation (bottom row).*

rotation of the Moon around Earth around the Sun. The first decomposition corresponds to the main oscillation (the one with the greatest amplitude). In the second step we obtain the minor oscillation. It can be interpreted as the rotation of the Moon around Earth (minor amplitude oscillation), the Earth turning around the Sun. The results are shown on figure 1.15 where we see the simplicity of the decomposition. Our signal is composed of only two oscillant signals, and we have extracted them of our signal in two iterations. Each time we iterate the decomposition, we peel one layer off our signal.

For the second case, we choose to study the decomposition on a simple function composed of two trigonometric monomials with distinct support (as one of the example used previously), shown on figure 1.16. The original signal has an average value equal

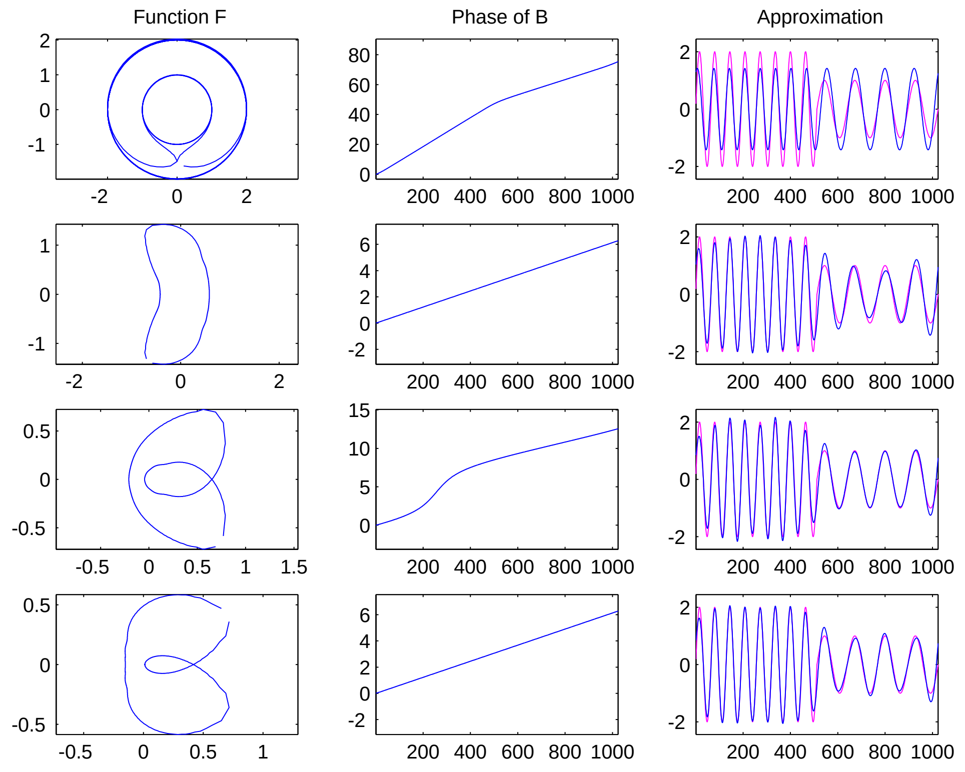


Figure 1.16: *Decomposition of the function* $F : \theta \mapsto e^{i2n_1\theta} \cdot 1_{I_1} + 2e^{in_1\theta} \cdot 1_{I_2}$ *and its approximations (right column).* B_+ *has a fixed amplitude in the first step, and oscillates mostly like* F_+. *After a few iterations the algorithm catches most of the signal energy*

to zero, so $c_0 = 0$, and the signal B_+ contains, as expected, only the signal frequential component. By this first approximation we manage to have more than half of the original signal energy. After few iterations, less than ten, we obtain more than 95% of the signal and the norm of g_n is converging to zero as an exponential function.

The last function has been created by using directly the formula that gives F_+ as the sum of the product of three Blaschke products: $z \mapsto B_+^1(z) \cdot (5 + B_+^2(z) \cdot (2 + B_+^3(z)))$. Then we just verify that we find the same zeroes and obtain the decomposition in three steps with the same terms. The figure 1.17 shows this obvious result. Under the condition that the residual, at the step n, has a smaller amplitude than the approximation, at the step n, we have unicity of the decomposition.

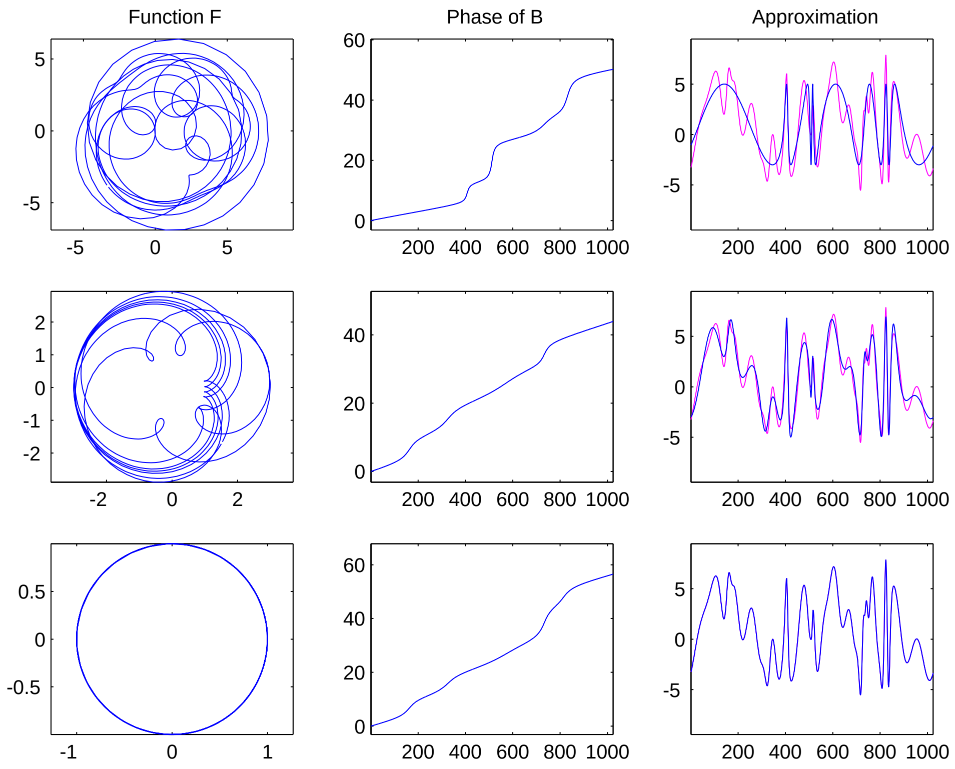


Figure 1.17: *An obvious decomposition of* $F_+ = B_+^1 \cdot (5 + B_+^2 \cdot (2 + B_+^3))$, *with* B_+^1, B_+^2 *and* B_+^3 *which are three polynomials with roots included in the unit disc and absolute value equal to one. It implies that they are Blaschke products. The oscillating functions are obtained in three iterations with a residual equal to zero numerically*

An extreme case

In this case we want to evaluate the "richness of our family of bases". So we generate random trigonometric polynomials (with an average value equal to zero) with degree N and coefficients given by a normal distribution. And we apply the same algorithm to decompose the function as before. We have the graphs on figure 1.18.

On figure 1.19, we can observe that the convergence rate is exponential as before. We can observe that the residual norm decreases as an exponential function. We can notice that G_+ is a always polynomial with a degree smaller than F_+. If we write f_+ as a Fourier series of order N on $-]\pi, \pi]$, it corresponds to write F_+ as a polynomial of degree N with

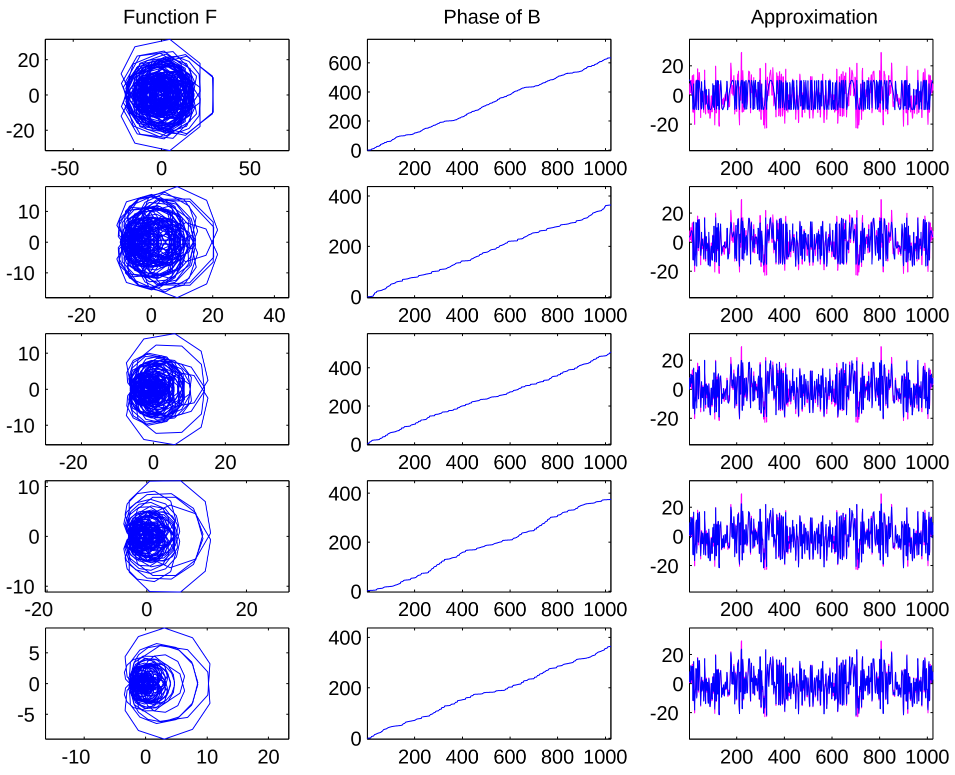


Figure 1.18: *Approximations of F_+ a random trigonometric polynomial of degree $N = 200$ using the orthogonal decomposition, $F : z \mapsto \sum_{j=1}^{N} a_j z^j$, $a_j \in N(0,\sigma)$*

the same coefficients. So the previous decomposition can be written as follow:

$$F_+(z) = F_+(0) + \lambda_1.z^k. \prod_{|a_i|<1} \frac{(z-a_i)}{(1-\bar{a}_i z)}. \prod_{|a_i|>1} (z-a_i). \prod_{|a_i|<1} (1-\bar{a}_i z) \tag{1.62}$$

Let's denote $\beta_i = \frac{1}{\bar{a}_i}$ for $|a_i| < 1$ and $\beta_i = a_i$ for $a_i > 1$, we have:

$$F_+(z) = F_+(0) + \lambda_1.z^k. \prod_{|a_i|<1} \frac{(z-a_i)}{(1-\bar{a}_i z)}. \prod (z-\beta_i), \text{ with } |\beta_i| > 0 \tag{1.63}$$

So G_+ has all its roots outside the unit disc and has a smaller degree than F_+ as $k > 0$. And we finally have, by letting B_+^1 and G_+^1 the terms coming from the first

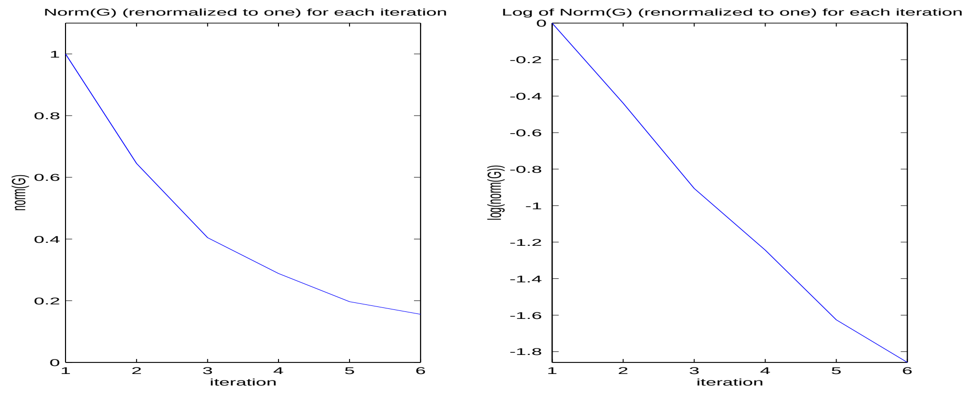


Figure 1.19: *Fast convergence of the residual (G) norm obtained from the decomposition of F_+, a random trigonometric polynomial of degree $N = 200$. On a logarithmic scale, this norm appears to be a linear function of the number of iterations*

decomposition:

$$\begin{aligned} F_+(z) &= F_+(0) + \lambda_1 \cdot B_+^1(z) \cdot G_+^1(z) \\ \text{with } G_+^1(z) &= \prod_{|\beta_i|>1} (z - \beta_i), \text{ and } deg(G_+) = deg(F_+) - k. \end{aligned} \tag{1.64}$$

The same algorithm is applied to G_+^1, we let B_+^j and G_+^j be the terms coming from the decomposition of G_+^{j-1}. B_{++}^j is the ratio polynomial obtained by extracting z^{k_j} from B_+^j, we have:

$$F_+(z) = F_+(0) + z^{k_1}.B_{++}^1(z)\big(\lambda_1 + z^{k_2}.B_{++}^2(\lambda^2 + z^{k_3}\cdots)\big) \tag{1.65}$$

We have to filter the Fourier transform of each G_+^i to obtain a trigonometric polynomial of same degree and be sure to not have any artifacts that can make the decomposition unstable. We treated in this chapter analytic functions without caring of the numerical values of the function on the unit circle. Analyticity on the unit circle means that the function will not vanish on a set of measure different from zero. But numerically, the function may have a compact support. We treat this case in the following chapter.

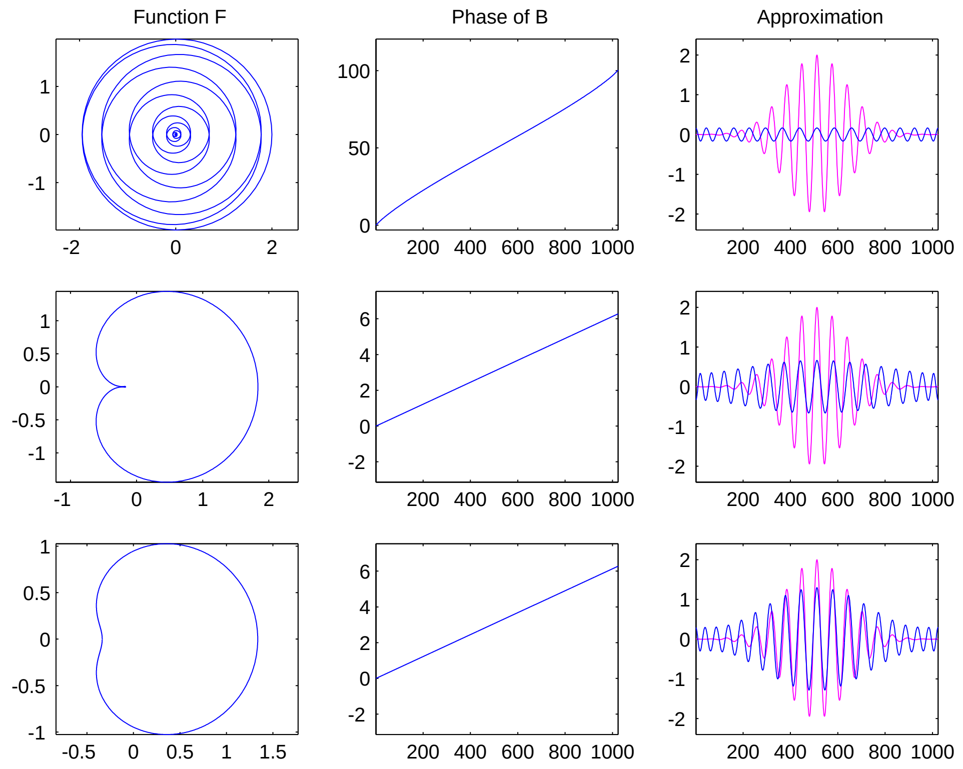


Figure 1.20: *Orthogonal decomposition of the modulated Gaussian signal* $F : \theta \mapsto e^{-(\theta-\theta_0)^2} \cdot e^{in\theta}$ *in three steps. The right column shows the evolution of the approximation.*

An analytic function with a numerical compact support different from the whole unit circle

We treat in this paragraph some special cases: analytical functions on the unit circle C with a "numerical compact support". It means that the functions have numerical values close to zero on a set of non-null measure. With a representation using the phase θ, it means that numerically the function vanishes numerically on an interval strictly included in $]-\pi,\pi]$. On figure 1.20, we have the case of a modulated Gaussian:

$$f(\theta) = \exp(-\lambda \cdot (\theta-\theta_0)^2) \cdot \cos(n(\theta-\theta_0)) \tag{1.66}$$

$$\Rightarrow f_+(\theta) = \exp(-\lambda(\theta-\theta_0)^2 + in(\theta-\theta_0)) \tag{1.67}$$

In this case, we know that we have to $\epsilon-$filter the function F to be able to compute B. We observe that "support(B)" can be adapted to F as it depends on the $\epsilon-$threshold. We observe that the norm of G_+ as a Gaussian distribution and converges to zero very fast on figure 1.21.

$$\|G_{iter}\| = \|F\| \cdot \exp(-\nu \cdot iter^2) \tag{1.68}$$

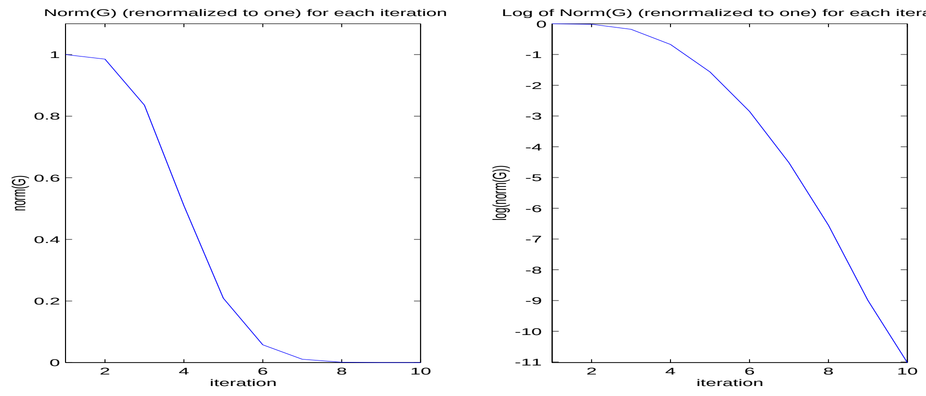


Figure 1.21: *Fast Convergence rate of the Gaussian approximation shown in figure 1.20*

1.6.5 Density of Blaschke product in the class of analytic functions

In the informal description of the method, we have written any signal f as a sum of Blaschke products were the coefficients c_n correspond to the scalar product of g_{n-1} and the family of Blaschke products. So once renormalized with the g_n and g_{n-1} norms, it gives the cosines of the angle between our library and the function.

A good approximation on average

Experiments ran on random trigonometric polynomials have been done, the coefficients of our polynomials have a Normal distribution centered in zero. We stored the value $g_0(0)$ obtained by the iterative decomposition algorithm. A mean value of 0.75 for the value of $g_1(0)$ (with g_0 renormalised to one) with a standard deviation of 0.03 is obtained for our data. The following theorem gives us a lower bound for this value.For simplicity we use a uniform distribution of the Fourier coefficients on the unit sphere.

Theorem 1.6.1. *Let $H_N^2 = \{F(\theta) = \sum_{k=0}^{N} a_k \cdot e^{ik\theta}, \quad ||a_k||_2 = 1\}$ equipped with denoted $d\sigma_F$ Lebesgue surface measure of Σ_{2N-1} in $\mathbb{C}^N$ normalized to one. There exists a universal constant c such that:*

$$E\Big(\exp\big[\int \log|F|\,d\theta\big]\Big) = \int_{\Sigma_{2N-1}} \exp\Big[\int_0^{2\pi} \log|F|\,d\theta\Big] d\sigma_F \geq c \tag{1.69}$$

where $c = \exp\Big(-\dfrac{\gamma + \log(2)}{2}\Big)$, *and* γ *is the Euler constant.*

Proof: See Appendix .1

Remark 1.6.4. *c corresponds to the scalar product between F and our family of Blaschke products. It corresponds to an angle around 58 degrees. We have shown a result that corresponds to an average value.*

The next chapter will show that the basis can be inappropriate in some cases.

A basis not adapted for polynomials

We show in this chapter that the polynomials $z \mapsto (z-\alpha)^n$ are not well represented in the basis using the family of Blaschke products. In this case, we observe on figure 1.22 that the Blaschke Product is unadapted to this polynomial. We show that using a variant rational function we considerably improve our basis, this idea is coming from the previous chapter 1.6.1. We study two cases to overcome this problem. We construct a polynomial f_n with the order-n root α in the unit disc. By rotation we simplify the study to the real positive case. If $\alpha = 0$ the case is trivial as $B_n = F_n$. We normalize f_n with c_n for the $\mathcal{L}^2(\mathbb{R})$-norm.

$$f_n(z) = c_n \cdot \left(\frac{z-\alpha}{1+\alpha}\right)^n, \text{ with } 0 < \alpha, \text{ and } c_n \sim \sqrt{n}. \tag{1.70}$$

We split our study in two main cases depending on the value of α.

- First case $\alpha < 1$

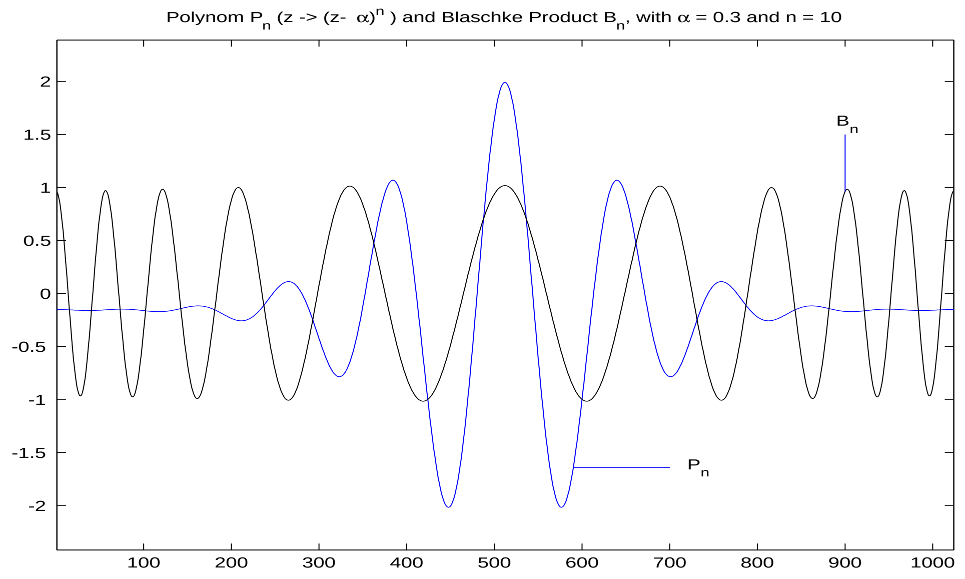


Figure 1.22: *The Real parts of the polynomial* P_n *(*$z \mapsto (z-\alpha)^n$, $|\alpha| < 1$*) and its Blaschke product* B_n

We have $f_n(0) = (\frac{\alpha}{1+\alpha})^n < \frac{1}{2^n} << 1$, and as the scalar product is linear, in this case we write $< f_n, \cdot > \sim < f_n - f_n(0), \cdot >$. Thus, the iterative algorithm gives its Blaschke product B_n associated:

$$B_n(z) = c_0 \cdot \left(\frac{z-\alpha}{1-\bar{\alpha}\cdot z}\right)^n, \text{ with } |c_0| = 1. \tag{1.71}$$

We compute the scalar product on the unit circle C between these two unit vectors:

$$\begin{aligned}
< f_n, B_n > &= c_0 \frac{c_n}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C (z-\alpha)^n \left(\frac{\bar{z}-\bar{\alpha}}{1-\alpha\cdot\bar{z}}\right)^n \frac{dz}{z} \\
< f_n, B_n > &= c_0 \frac{c_n}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C (z-\alpha)^n \left(\frac{1-\bar{\alpha}z}{z-\alpha}\right)^n \frac{dz}{z} \\
< f_n, B_n > &= c_0 \frac{c_n}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C (1-\bar{\alpha}z)^n \frac{dz}{z} \\
|< f_n, B_n >| &\sim \frac{\sqrt{n}}{(1+\alpha)^n} << 1
\end{aligned} \tag{1.72}$$

The scalar product decreases exponentially to zero as n tends to infinity and corresponds to the cosine of the minimum angle between f_n and any vectors from our family. We conclude that our family of functions is not very well adapted to this case. We generate a new family, more concentrated spatially (most of the energy is within a small region of C) to obtain a better result:

$$\psi_n(z) = B_n(z) \cdot \frac{(1-|\beta|^2)^{\frac{1}{2}}}{1-\bar{\beta}\cdot z}, \text{ with } \beta \in \mathbb{C}. \tag{1.73}$$

We then compute, in a similar way as before, the scalar product:

$$\begin{aligned}
< f_n, \psi_n > &= c_0 c_n \cdot \frac{(1-|\beta|^2)^{1/2}}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C (z-\alpha)^n \left(\frac{1-\bar{\alpha}z}{z-\alpha}\right)^n \cdot \frac{1}{1-\beta\bar{z}} \frac{dz}{z} \\
< f_n, \psi_n > &= c_0 c_n \cdot \frac{(1-|\beta|^2)^{1/2}}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C \frac{(1-\bar{\alpha}z)^n}{z-\beta} dz \\
< f_n, \psi_n > &= c_0 c_n \cdot \frac{(1-|\beta|^2)^{1/2}}{(1+\alpha)^n} (1-\alpha\beta)^n
\end{aligned} \tag{1.74}$$

We observe that $\beta = 0$ gives $\psi_n = B_n$. We use β to maximize this scalar product in order to obtain a better approximation. Thus, our iterative algorithm will be optimize as it gets more energy at each iteration of the decomposition. We observe that this scalar product can be written in a different way by using the $B \cdot G$ decomposition as follow:

$$< f_n, \psi_n > = < F(z), B_n(z) \cdot \frac{(1-|\beta|^2)^{1/2}}{1-\bar{\beta}\cdot z} > \tag{1.75}$$

And we can compute the Blaschke product on the unit circle C:

$$\begin{aligned}
< f_n, \psi_n > &= \frac{1}{2\pi i} \oint_C B_n(z) \cdot G_n(z) \cdot \bar{B}_n(z) \cdot \frac{(1-|\beta|^2)^{1/2}}{1-\beta\cdot\bar{z}} \frac{dz}{z} \\
< f_n, \psi_n > &= \frac{1}{2\pi i} \oint_C G_n(z) \cdot \frac{(1-|\beta|^2)^{1/2}}{z-\beta} dz \\
|< f_n, \psi_n >| &= |G_n(\beta)| \cdot (1-|\beta|^2)^{1/2}
\end{aligned} \tag{1.76}$$

In this case, we know that f_n has α for only root, with order n, $G_n(z) = (1-\bar{\alpha}\cdot z)^n$. We have to maximize the right term of 1.76: $\beta \mapsto \left|(1-\bar{\alpha}\cdot\beta)^n(1-|\beta|)^{1/2}\right|$. We suppose $\alpha \in [0,1]$, as a rotation will leave the problem unchanged, and we obtain a β real defined as:

$$\beta = -1 + \frac{1+\alpha}{\alpha\cdot(2n+1)} \tag{1.77}$$

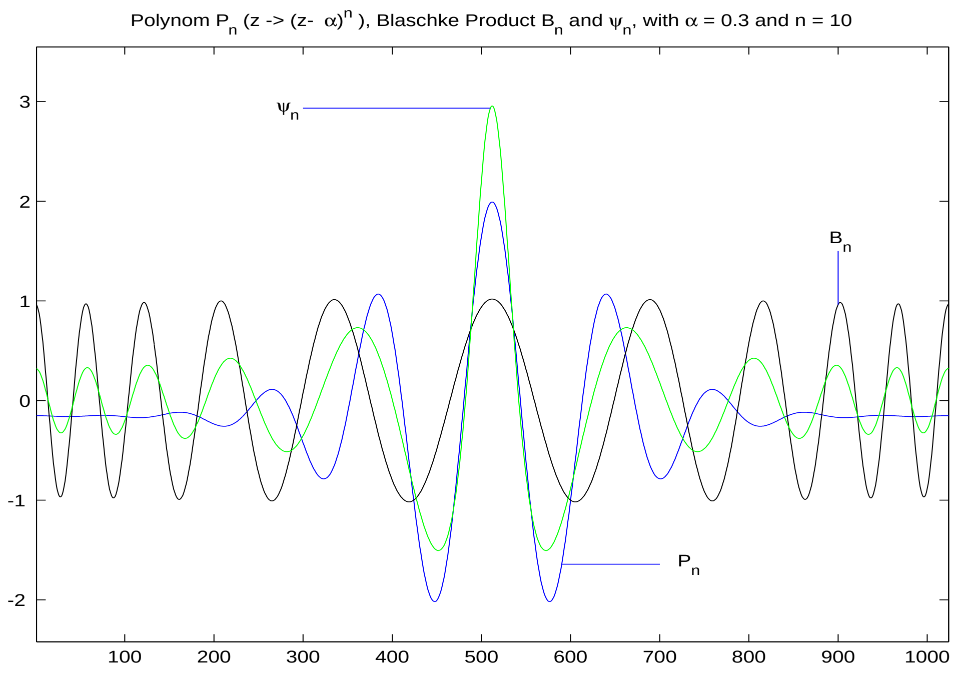


Figure 1.23: *The Real parts of the polynomial P_n $(z \mapsto (z-\alpha)^n$ $(|\alpha| < 1))$, its Blaschke product B_n and ψ_n*

We have the following scalar product:

$$\begin{aligned}
|< f_n, \psi_n >| &\sim \frac{\sqrt{n \cdot \frac{1+\alpha}{\alpha \cdot n}}}{(1+\alpha)^n} \Big(1 + \alpha(1 - \frac{1+\alpha}{\alpha \cdot 2n})\Big)^n \\
|< f_n, \psi_n >| &\sim \sqrt{\frac{1+\alpha}{\alpha \cdot e}} \sim 1 \qquad (1.78)
\end{aligned}$$

We observe on figure 1.23 that the function ψ_n is more localized than B_n and obviously a numerical computation confirms that the second scalar product is greater.

- Second case $\alpha > 1$

We have $f_n(0) = (\frac{-\alpha}{1+\alpha})^n$ that is not necessarily small. One can verifies easily that the roots of $f_n - f_n(0) = 0$ are $z_k = \alpha \cdot (1 - \omega_k)$ with $\omega_k = \exp(i2k\pi/n)$. The roots of B_n belong to the unit disc. Thus, we are only interested in the value of z_k such that $|z_k| < 1$. $z_0 = 0$, thus 0 is a root of B_n. We then have two sub-cases depending on the

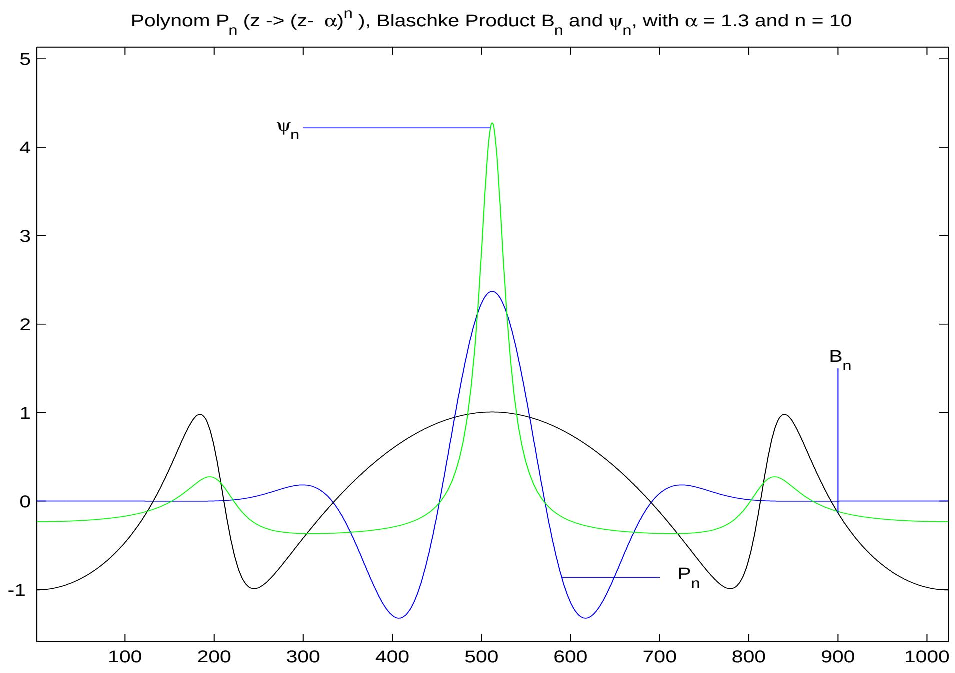


Figure 1.24: *The Real parts of the polynomial P_n ($z \mapsto (z-\alpha)^n$, $|\alpha| > 1$), its Blaschke product B_n and ψ_n*

value of $r = \min_{\omega_k \neq 1} |\alpha(1-\omega_k)|$, there is only zero as a root or not. A short calculation gives $r = 2\alpha \sin(\pi/n)$. But we know that if $\alpha > 1$, the signal is not really oscillant. Thus we are just computing a lower bound of our projection. We consider that 0 is the only root of B_n. Then we have $B_n(z) = z$ and $G_n(z) = (F_n(z) - F_n(0))/z$. The scalar product is the following:

$$\begin{aligned} < f_n, B_n > &= \frac{c_n}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C \left((z-\alpha)^n - (-\alpha)^n\right) \cdot \bar{z} \cdot \frac{dz}{z} \\ |< f_n, B_n >| &\sim \frac{n^{\frac{3}{2}}}{\alpha} \cdot \left(\frac{\alpha}{1+\alpha}\right)^n << 1 \end{aligned} \tag{1.79}$$

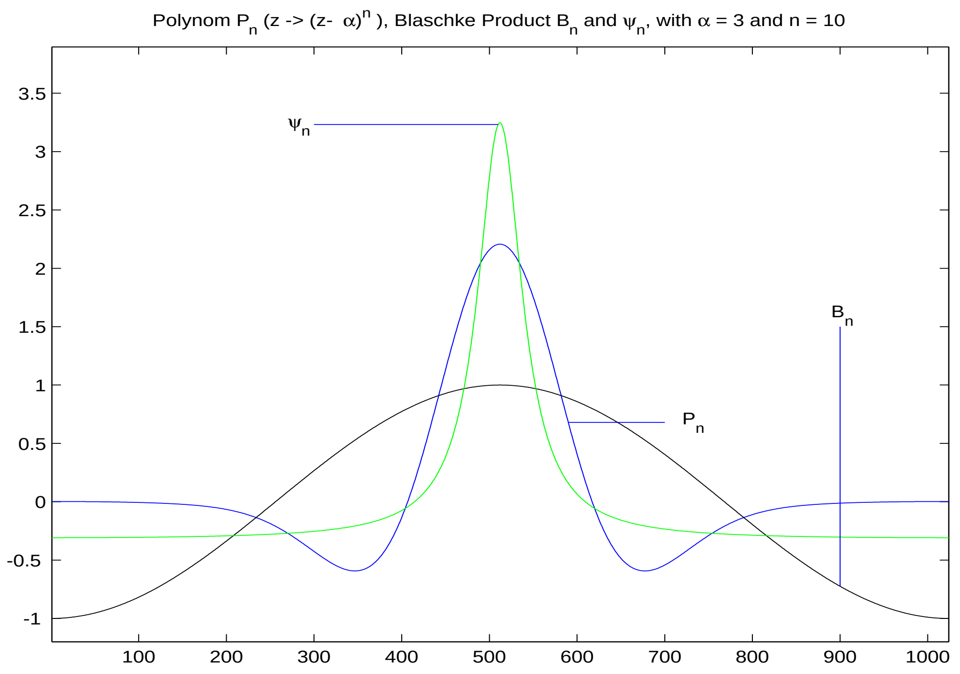


Figure 1.25: *The Real parts of the polynomial P_n ($z \mapsto (z-\alpha)^n$ $|\alpha| > 1$), its Blaschke product B_n and ψ_n*

This scalar product is small and we construct another family more concentrated spatially to obtain a better result. We have $\psi_n(z) = \frac{\sqrt{1-|\beta|^2}}{1-\bar{\beta}\cdot z}$, with $\beta \in \mathbb{C}$. The scalar product is:

$$\begin{aligned}
< f_n, \psi_n > &= \frac{c_n}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C \left((z-\alpha)^n - (-\alpha)^n\right) \cdot \frac{\sqrt{1-|\beta|^2}}{1-\beta\cdot\bar{z}} \cdot \bar{z} \cdot \frac{dz}{z} \\
< f_n, \psi_n > &= \frac{c_n}{(1+\alpha)^n} \cdot \frac{1}{2\pi i} \oint_C \left((z-\alpha)^n - (-\alpha)^n\right) \cdot \frac{(1-|\beta|^2)^{1/2}}{z-\beta} \cdot \frac{dz}{z} \\
< f_n, \psi_n > &= \frac{c_n \cdot (-\alpha)^{n-1}}{(1+\alpha)^n} \cdot \left(\sum_{p=0}^{n-1} \left(\frac{\beta-\alpha}{-\alpha}\right)^p \right) \cdot \sqrt{1-|\beta|^2} \\
< f_n, \psi_n > &= \frac{c_n \cdot (-\alpha)^n}{(1+\alpha)^n} \cdot \left(\left(1 - \frac{\beta}{\alpha}\right)^n - 1 \right) \cdot \frac{\sqrt{1-|\beta|^2}}{\beta} \qquad (1.80)
\end{aligned}$$

That corresponds as shown before to $G_n(\beta) \cdot (1-|\beta|^2)^{1/2}$. Then we study the extrema of $\beta \mapsto \frac{\sqrt{1-|\beta|^2}}{\beta} \cdot \left(\left(\frac{\beta-\alpha}{-\alpha}\right)^n - 1 \right)$. And for $\alpha > 1$, we approximate β:

$$\beta = -1 + \frac{\alpha}{2n} + o\left(\frac{1}{n}\right) \qquad (1.81)$$

The scalar product is then:

$$\begin{aligned} |< f_n, \psi_n >| &\sim \frac{c_n \cdot \alpha^n}{(1+\alpha)^n} \cdot \left(\left(\frac{\alpha+1}{\alpha} - \frac{1}{2n} \right)^n - 1 \right) \cdot \sqrt{1 - \left| 1 - \frac{\alpha}{2n} \right|^2} \\ |< f_n, \psi_n >| &\sim c_n \cdot \left(1 - \frac{\alpha}{(\alpha+1) \cdot 2n} \right)^n \cdot \sqrt{\frac{\alpha}{n}} \\ |< f_n, \psi_n >| &\sim \sqrt{\frac{\alpha}{e^{\frac{\alpha}{(\alpha+1)}}}} \sim 1 \end{aligned} \tag{1.82}$$

And we conclude that in this case the family $(\psi_n)_n$ is more adapted to our function than $(B_n)_n$ as observed on figure 1.25 and 1.24 that correspond to the two sub-cases for $\alpha > 1$.

Improvement of the iterative algorithm

We have shown that we obtain a much better projection by changing our projecting s-pace. The iterative algorithm given by equation (1.53) is modified. We are now using new notations and the iterative algorithm is the following:

$$r_n(z) = r_{n-1}(z) - < r_{n-1}, b_{n-1} \cdot \frac{\sqrt{1 - |\beta_n|^2}}{1 - \bar{\beta}_n \cdot z} > \cdot b_{n-1}(z) \cdot \frac{\sqrt{1 - |\beta_n|^2}}{1 - \bar{\beta}_n \cdot z} \tag{1.83}$$

Where b_{n-1} is the Blaschke Product of $r_{n-1} = b_{n_1} \cdot g_{n-1}$. r_n is orthogonal to r_{n-1}. We can notice also that

$$< r_{n-1}, b_{n-1} \cdot \frac{\sqrt{1 - |\beta_n|^2}}{1 - \bar{\beta}_n \cdot z} >= \frac{g_{n-1}(\beta_n)}{\sqrt{1 - |\beta_n|^2}} \tag{1.84}$$

β_n has been chosen such that the absolute value of equation (1.84) is maximum. The previous algorithm based on the Blaschke product only corresponds to $\beta_n = 0$. And we have obviously $r_n(\beta_n) = 0$. We then have the following iterative orthogonal decomposition.

$$\begin{aligned} r_0(z) &= c_0 \cdot b_0(z) \cdot \frac{1}{1 - \bar{\beta}_1 \cdot z} \\ &+ c_0 \cdot c_1 \cdot b_0(z) \cdot \frac{z - \beta_1}{1 - \bar{\beta}_1 \cdot z} \cdot b_1(z) \cdot \frac{1}{1 - \bar{\beta}_2 \cdot z} \\ &+ \cdots \\ \text{with } c_i &= < g_i, b_i \cdot \frac{1 - |\beta_{i+1}|^2}{1 - \bar{\beta}_{i+1}} > \end{aligned} \tag{1.85}$$

We still have an orthogonal decomposition, since we have a process similar to the Gram-Schmidt algorithm describes for rational functional in chapter 1.6.1 and each successive term of equation (1.85) are orthogonal to each other.

1.7 Application to Sound Signals

In this section, we want to study sound signals using the "$F = B{\cdot}G$" factorization properties. We record sounds on a microphone, that contain many times the word "Michel" for different speakers, as real one dimensional signals. We separate each word and compute the analytic signal corresponding and the "$B \cdot G$" product. We show that the Blaschke product gives a reasonable way to discriminate the different speakers.

1.7.1 Study of the word "Michel"

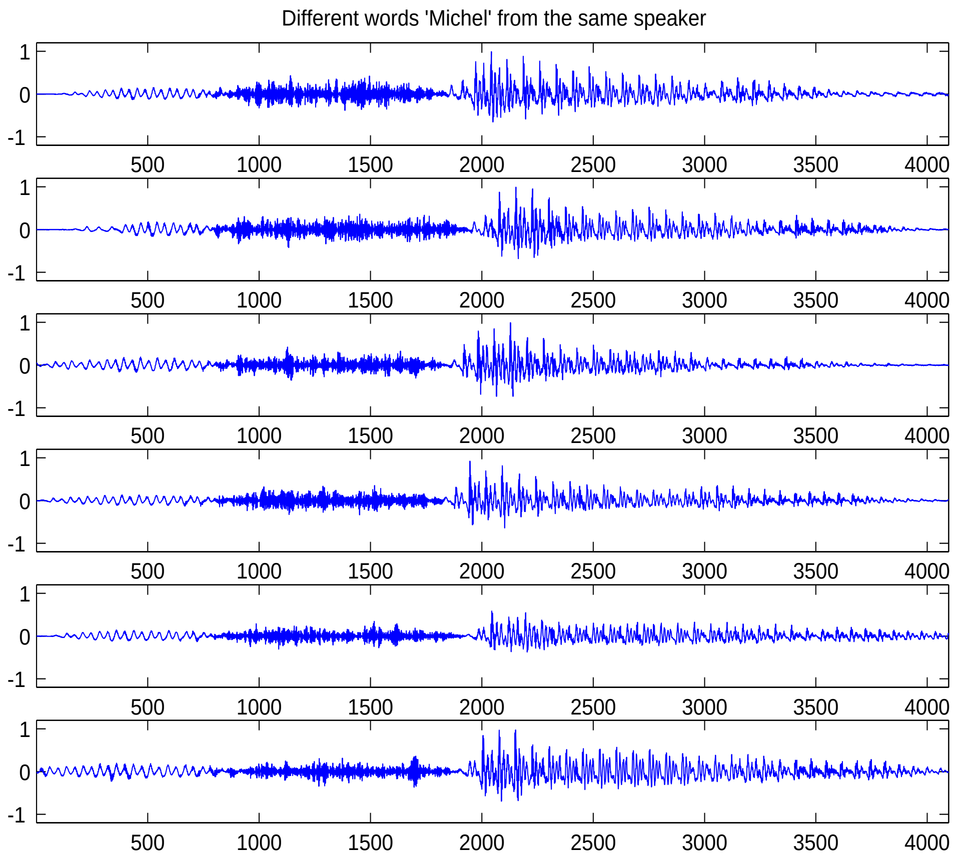


Figure 1.26: *The time series of the word "Michel" pronounced six times by the same speaker*

Different segments "Michel", from the same speaker, have been extracted from the complexified extension of our original one dimensional signal, each of them are denoted F_i. It is difficult to observe and study the signal as is. We observe that these signals can be segmented in three regions. We apply a post-treatment on our signal, as any analysis on them seems to be far from obvious as we can observe in figure 1.26. We decompose these signals, indexed with i, using the Blaschke product. We have for each signal the decomposition $F_i = B_i \cdot G_i$.

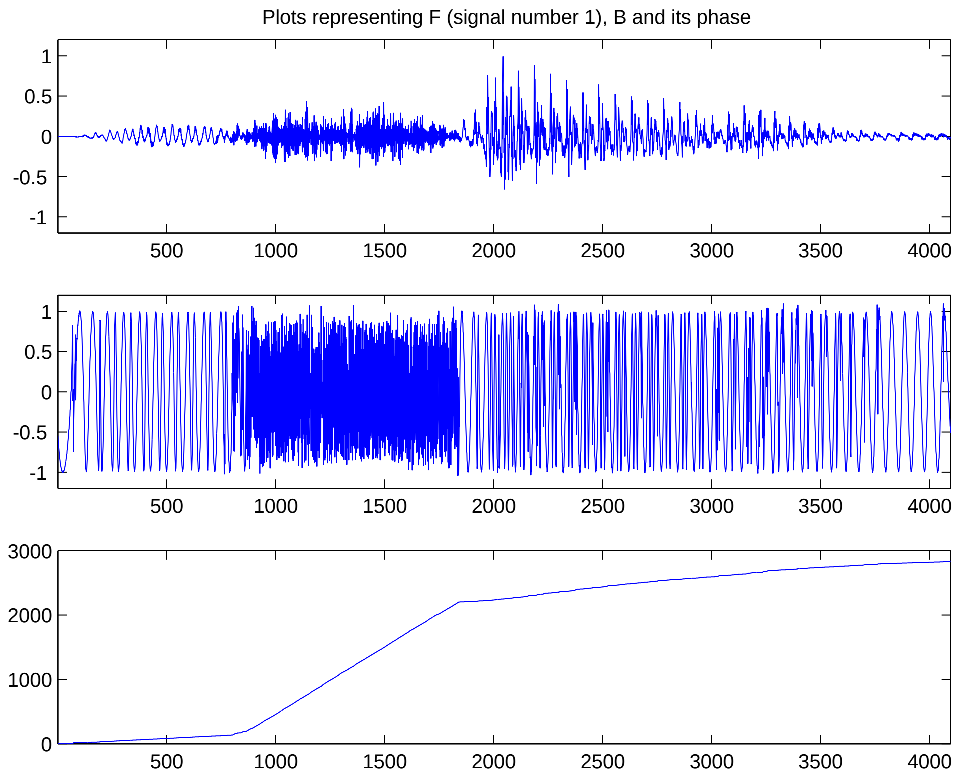


Figure 1.27: *Decomposition of the word "Michel": real part of F, B and the non-decreasing phase ϕ_B.*

We know that any Blaschke product B_i has a non-decreasing phase ϕ_{B_i} as seen previously, this characteristic is false for the signal F_i. The family of ϕ_{B_i} seems easier to study and classify than the F_i as we can observe on the figure 1.27. One can observe the three distinct intervals of ϕ_B, plotted on the bottom row of figure 1.27.

We now represent six phases ϕ_{B_i} corresponding to the Blaschke product B_i of the decomposition for the words “Michel” pronounced by the same speaker. It gives the interesting figure 1.28.

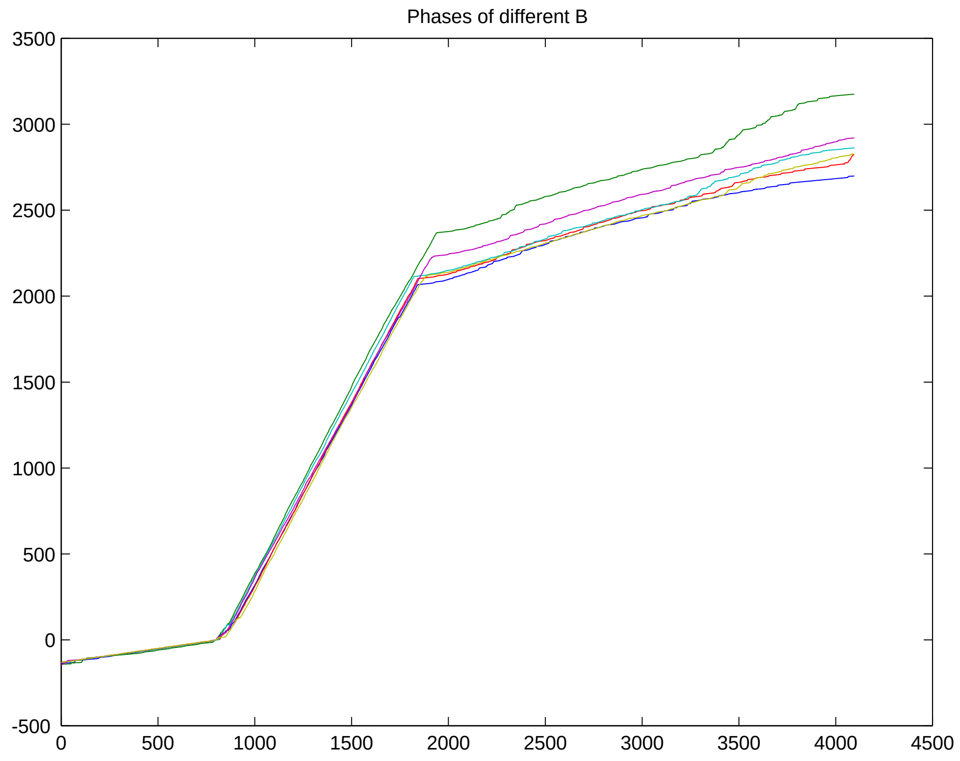


Figure 1.28: *Six phases ϕ_{B_i}, corresponding to the Blaschke product of the word “Michel”. One can look the slope of ϕ_{B_i} and separate the signal in three intervals. We also notice that this representation may be stable.*

We observe that the ϕ_{B_i} have a simple structure. At first sight, we note three regions, each of them characterized by a different slope. The slopes are linked to the syllables. It gives us the main frequencies or pitches of the word, and corresponds to the melody of the word. We obtain a basic segmentation of “Michel” using a polygonal line, by splitting F_i where ϕ_{B_i} has the same main frequency. These three parts have different lengths for each ϕ_{B_i} and correspond to the syllables of “Michel”. Figure 1.29 show the graphs for six words “Michel” pronounced by another speaker.

Surprisingly, if the word is pronounced more or less slowly we still have three intervals with slope of similar value. As a consequence the length of each interval will be obviously

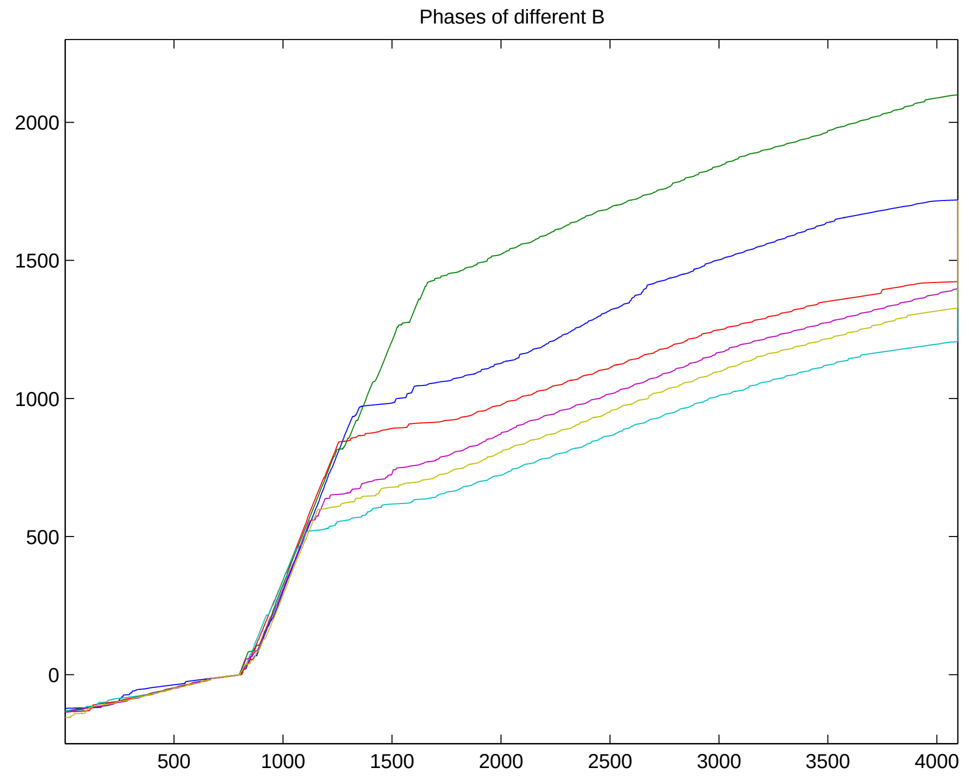


Figure 1.29: *Six phases ϕ_{B_i} (obtained from the Blaschke product) of "Michel", said by another speaker, showing the influence of the speech speed. One can obviously see that the length of each interval is not a stable parameter, as supposed in figure 1.28, but one can notice that the slopes on each interval may be quite stable.*

modified and the "winding number", as the phase will vary accordingly. We can also observe that the slopes are slightly different from one speaker to another. The same word "Michel" pronounced by two different persons has the same structure: one very slow slope at the beginning, one faster in the middle and a slow one at the end. Thus, we can think that for the fixed word "Michel", the three slopes depend on the speaker only and the three lengths on the speed of the speech. To determine if this property is true we represent now five different speakers using the two last slopes (and not three for an easier visualization) as parameters for the graphical representation. We obtain the figure 1.30. We observe that with only two parameters, in some cases, we can start to evaluate which speaker corresponds to the word "Michel".

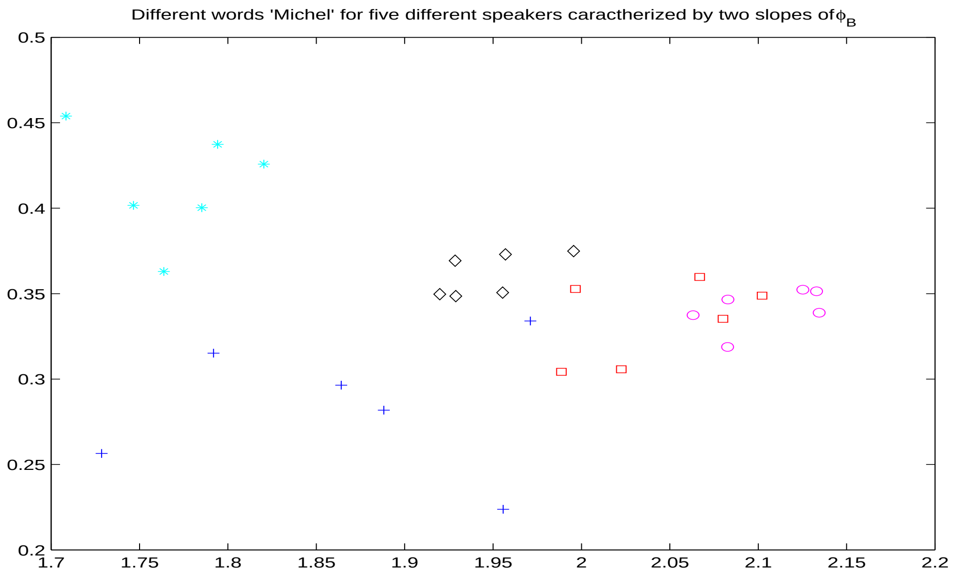


Figure 1.30: *The word "Michel" represented for five different speakers, who pronounced the word six times each, using the two last slopes of ϕ_B for five different speakers*

We observe that some speakers have their six points more or less concentrated. The main reason is that two of the five speakers have done variations of their tone and speech speed, as we observe the difference between the figures 1.28 and 1.29, that represent two different speakers. It makes their representation more difficult by contrast to the three others speakers. We compare now the instantaneous frequencies given by a classical phase-plane analysis with best basis (figure 1.31)on the signal F and the derivative of ϕ_F and ϕ_B (figure 1.32).

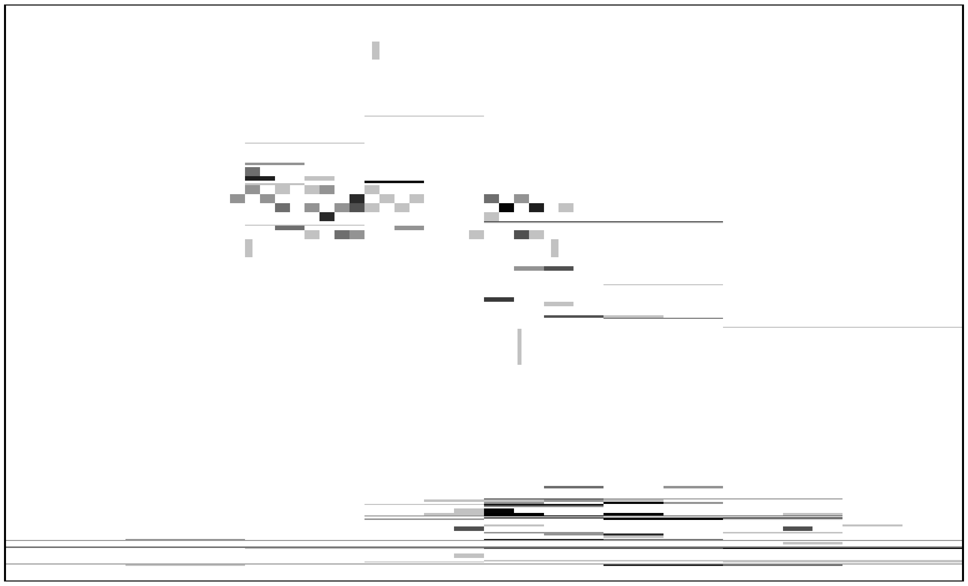


Figure 1.31: *The instantaneous frequencies of the word "Michel" using the phase plane analysis*

A change of variables between two different phases doesn't give too much result! And the registration should be done first!!

1.7.2 Application to compression

We observed previously that in the factorization "$F = B \cdot G$" a form of noise is mostly contained in the phase of G. We decide to study $F^* = |F| \cdot B$, than can be also written as $F \cdot e^{-i \cdot \phi_G}$, and use the properties of B. The sound of $Re(F^*)$ doesn't seem to be different from $Re(F)$. It means that the ear has difficulty to compare $|F| \cdot \cos(\phi_B)$ and $|F| \cdot \cos(\phi_F)$. Reducing ϕ_B to lines on each segment, as observed before, corresponds to a major compression and what we hear has lost most of the information. Having the main frequency gives us just a whispering by itself, it corresponds to the melody of the word. We need to have information on the instantaneous frequency at every point. But if we combine the polygonal phase with F amplitude we can almost guess the sound "Michel", but we still have a lot of noise even if it sounds better than just the F amplitude. Furthermore, we have to keep the smaller oscillations of the phase that contains the instantaneous frequencies of

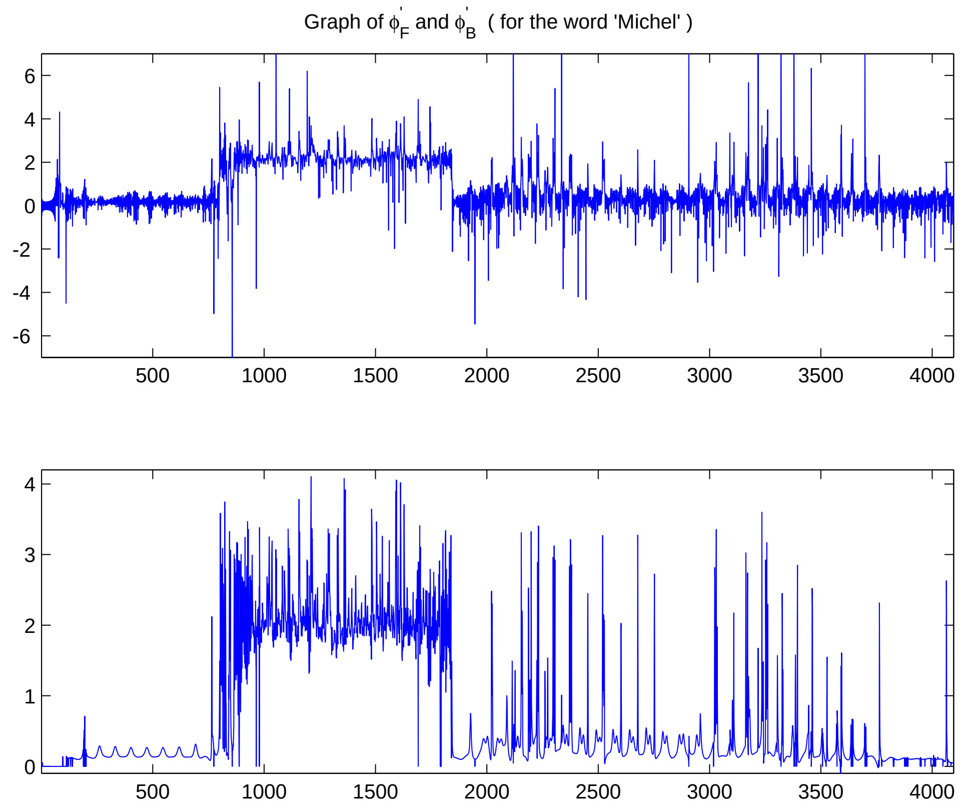


Figure 1.32: *First derivative of the F and B phases for the same sound "Michel".*

the signal. We compressed the amplitude of F and the phase of B, using a Best Basis decomposition. And we observe that both signals have more than 99.9% of their energy with only one percent of the signal as opposed to $Re(F)$ that needs around five percent of the signal to keep the same amount of energy. But we observe the following paradox that the second compression, that retains a lower percentage of energy, gives us a better result. This paradox can be explained as we approximate ϕ_B and not B, thus we do not have any control on the approximations.

1.8 A two-dimensional extension of the Blaschke Product

In the previous chapters, we computed the Blaschke product for the one-dimensional case. We complexified a real function defined on the unit circle C, using the Hilbert transform, to extend it to the whole complex plane $\mathbb{C}$. The Blaschke product is known to only exist in the one dimensional case, but we want to extend this approach to the two-dimensional case by developing a similar algorithm, knowing that the canonical factorization associated is obtained without searching for the zeroes. We will would like for example to re-normalize the "three circles" on figure 1.33. Where each circle is defined by the center, the radius, the

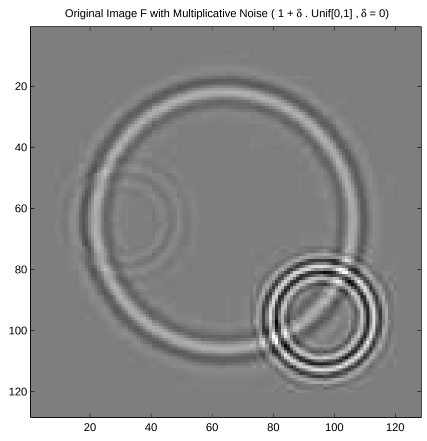


Figure 1.33: *Three circles with different frequencies, amplitudes and radii. The amplitudes are proportional to* $1, 5, 10$. *We use this example to show later on how we can re-normalize to the same amplitude these three circles.*

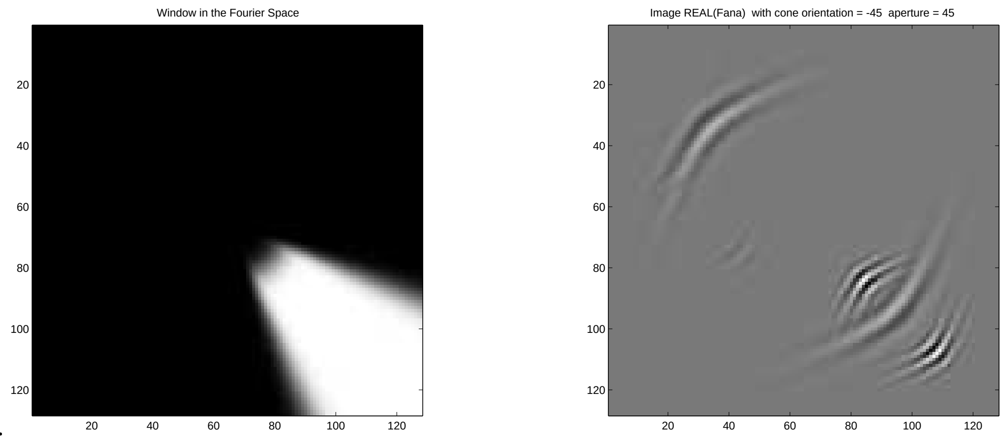
amplitude, frequency modulation and the variance of the Gaussian that supports it. These four variables are noted $(z_i, r_i, \alpha_i, \lambda_i)$ and thus the equations of each circle $Circle_i$ and the image I are as follows:

$$\begin{aligned} Circle_i(z) &= \alpha_i \cdot \sin(\lambda_i\,|z - z_i|) \cdot e^{-\frac{(|z-z_i|-r_i)^2}{\sigma_i}} && (1.86) \\ I &= \sum_{i=1}^{3} Circle_i && (1.87) \end{aligned}$$

For this example, we chose the amplitude α_i proportional to $1, 5, 10$.

1.8.1 Artifact related to the two dimensional case

During the complexification process of f in the one-dimensional case, we apply the Hilbert transform and thus the Fourier transform. For the two-dimensional case, this step is sensitive as there is not a canonical extension. But we are interested in the different structures contained in images using the oscillatory contents (as many structures seem to be related to the oscillations). The extraction of the oscillations is obtained by a division of the Fourier space with "cones" (we keep the regions in the Fourier space that corresponds to a common



.

Figure 1.34: *A possible cone defined for a partition of the Fourier Space, left figure, and* F_{C_i}*, right image, obtained after filtering of the "three circles" shown on figure 1.33. We notice that the present oscillations of* F_{C_i} *are obviously contained in the cone defined in the Fourier space.*

main oscillation direction), using a partition of unity. For each cone "C_i", we note F_{C_i} the subimage obtained after filtering in the Fourier space. The division is obtained to give priority to the direction and not to the frequency. One of the partition of unity functions can be chosen with a C_i as seen on the figure 1.34.

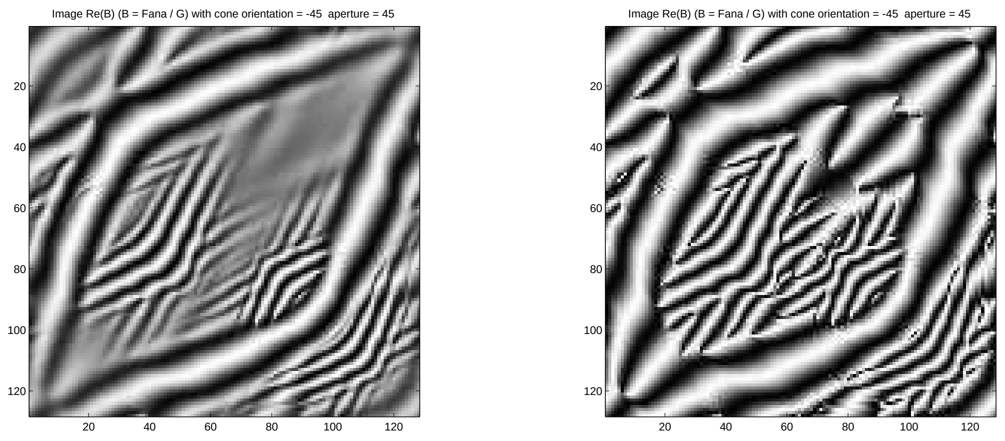
1.8.2 Analytic extension with cone

We divide the Fourier space in "cones" centered in the origin, and isolated the low frequencies for a separate study, as seen on figure 1.34. The number of cones has to be adapted to the image composition, we choose usually eight cones and the low frequencies as the partition of the Fourier space. We use the classic Cartesian Fourier transform. We start from a real signal F from $\mathbb{R}^2$ to $\mathbb{R}$, we compute F_{C_i}, that corresponds to the frequencies living in the region C_i, letting C_0 be the central region containing the low frequencies component. Each F_{C_i} is analytic on $C_i^* x \mathbb{R}^2$, where C_i^* is the dual cone of C_i. An example is shown on figure 1.34. We have the following formulae:

$$\forall i > 0, F_{C_i} = FT^{-1}\big(FT(f) \cdot 1_{C_i}\big) \quad \text{and} \sum_i 1_{C_i} = 1 \tag{1.88}$$

$$f = Re\big(\sum_{C_i} F_{C_i}\big) \tag{1.89}$$

To avoid artifacts, the regions C_i can be defined using smooth bells. The functions F_{C_i} are comparable to the brushlets developed by F. Meyer [5]. The similarity is due to the fact that we have a partition of the Fourier space and separate the oscillations in different directions, but we do not fold the projection since the orthogonality is not our concern. We present now the factorization process for the two-dimensional case. After the first step corresponding to



.

Figure 1.35: *Two B_{C_i} obtained for the "three circles" (shown on figure 1.33) with two different ϵ-filtering. The ϵ on the left is smaller than the right one. Then the right figure shows an "upper B" than on the left.*

the complexification, explained in the previous chapter, we obtain $l_i = \log|F_{C_i}|$. We have to extend l_i to $L_{i,+}$, to obtain an analytic function. We compute $L_{i,+}$ using the same cone C_i in the Fourier space and obtain the equivalent of the one dimensional outer function that we note G_{C_i,C_i}. We lose the properties on B_+, obtained in the one dimensional case, shown before as we do not preserve $|G_{C_i,C_i}| = |F_{C_i}|$ (we use the double notation "C_i, C_i", for index, as we use the cone C_i for the Fourier transform at each iteration). It means that $|B_{C_i,C_i}|$ is not equal anymore to one. An alternative way presents to us, filtering in the Fourier space using the half plane H_i centered with the angle chosen before. In this case, we have by construction $|B_{C_i,H_i}| = \left|\frac{F_{C_i}}{G_{C_i,H_i}}\right| = 1$ since the filtering in the Fourier space gives $|G_{C_i,H_i}| = |F_{C_i}|$. For figure 1.33, we obtain B as shown on figure 1.35. This transformation has the particularity to make $|B_{C_i,H_i}| = 1$. But unfortunately it is not necessarily an advantage as oscillations appear all over the image.

We filter our signal with the ϵ-threshold, as defined in the one dimensional case in the chapter 1.4.3, to obtain with a new notation $G_{C_i,H_i,\epsilon}$ (for $\epsilon = 0$, we omit the parameter ϵ in our notation), to lower to zero the regions where $|F_{C_i}|$ is smaller than $\epsilon\|F_{C_i}\|_\infty$. And we obtain $B_{C_i,H_i,\epsilon} = \frac{F_i}{G_{C_i,H_i,\epsilon}}$. In the other regions of the image, where $|F_{C_i}| >> \epsilon\|F_{C_i}\|_\infty$, the absolute value is then re-normalized to one as seen in the one-dimensional case. For the two-dimensional case we have choose the Poisson filtering as defined previously. By decreasing, the value of ϵ we only keep the oscillations in the regions where F_{C_i} has more energy.

The main difference between the one-dimensional and two-dimensional case is coming from the fact that F_{C_i} has numerically a compact support in most of the case as opposed to F_+ in the one-dimensional case. It comes from the fact that by selecting a cone C_i in the Fourier space, instead of the half plane H_i, to compute F_{C_i} we lose a main part of the energy of f. Therefore it does not make sense to raise $|B_{C_i,H_i,\epsilon}|$ to one everywhere by lowering the value of ϵ. A solution that one can choose is to set to zero $B_{C_i,H_i,\epsilon}$ in the regions where $F_{C_i} > \epsilon\|F_{C_i}\|_\infty$. From the decomposition we obtain:

$$f = Re\big(F_{C_0} + \sum_{C_i, i>0} B_{C_i,H_i,\epsilon} \cdot G_{C_i,H_i,\epsilon}\big) \tag{1.90}$$

Since by construction of $B_{C_i,H_i,\epsilon} = \frac{F_{C_i}}{G_{C_i,H_i,\epsilon}}$. If we use an ϵ-threshold we have:

$$\forall i > 0, |B_{C_i,H_i}| = 1 \text{ if } |F_{C_i}| > \epsilon, \text{ and } |B_{C_i,H_i}| = 0 \text{ else.} \tag{1.91}$$

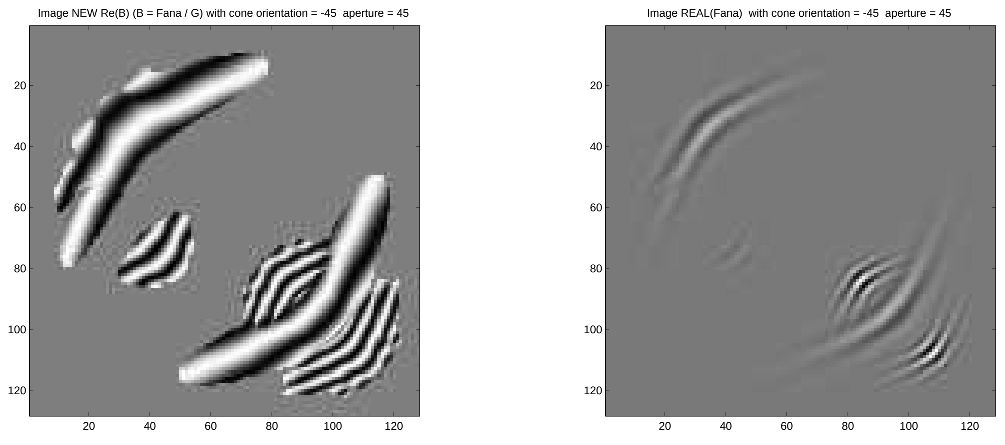


Image NEW Re(B) (B = Fana / G) with cone orientation = -45 aperture = 45
Image REAL(Fana) with cone orientation = -45 aperture = 45
20
40
60
80
100
120

.

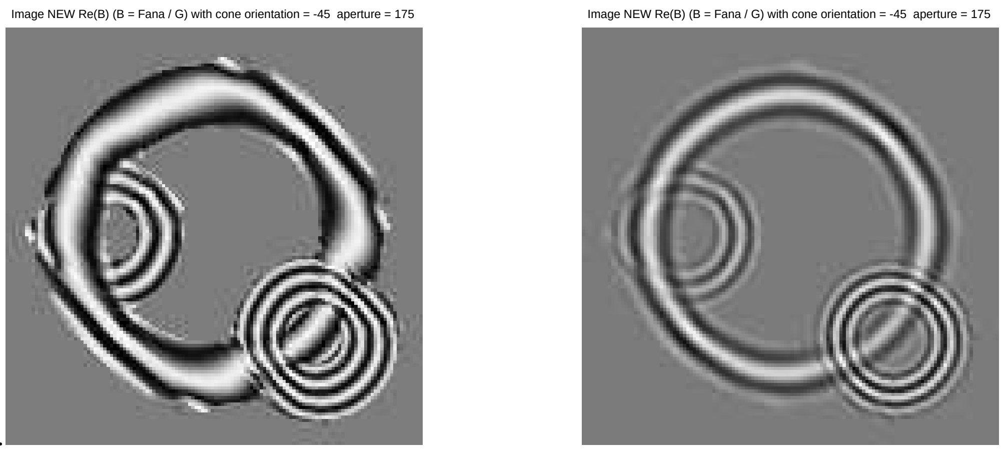
Figure 1.36: *The left figure represents B_{C_i} obtained for the "three circles" (shown on figure 1.35) with an ϵ-threshold. We observe the extension from the right figure F_{C_i}*

We represent on figure 1.36 the function $B_{C_i,H_i,\epsilon}$ with an ϵ-threshold and set to zero for $|F_{C_i}| < \epsilon\|F_{C_i}\|_\infty$. We observe that with this new $B_{C_i,H_i,\epsilon}$ (with a threshold to set to zero) we have extended in some sense the function F_{C_i} shown on figure 1.34, the region where the oscillations are visible is now greater than previously. The value of the ϵ, chose for the ϵ-filtering, will affect much more our two-dimensional signal during the extension, than in the one-dimensional case.

1.8.3 Image enhancement of "the three circles"

$|B_{C_i,H_i,\epsilon}|$ has the particularity to be similar to $1_{F_{C_i}}$, as we apply a ϵ-threshold, renormalizing the function to one where $F_{C_i} >> \epsilon\|F_{C_i}\|_\infty$, and setting to zero the regions where $F_{C_i} < \epsilon\|F_{C_i}\|_\infty$. $B_{C_i,H_i,\epsilon}$ and F_{C_i} have similar oscillations but a different amplitude. We can interpret $B_{C_i,H_i,\epsilon}$ as a re-normalization of F_{C_i} such that all its oscillations have the same amplitude one.

It can be in some cases an advantage as seen in the example of the three "circles", each one them have a different frequency and amplitude. So by using the "$B \cdot G$" factorization we manage to restore the three circles with the same contrast. On figure 1.37, we present two results with a different ϵ-filtering. If the ϵ is chosee quite small it gives a much



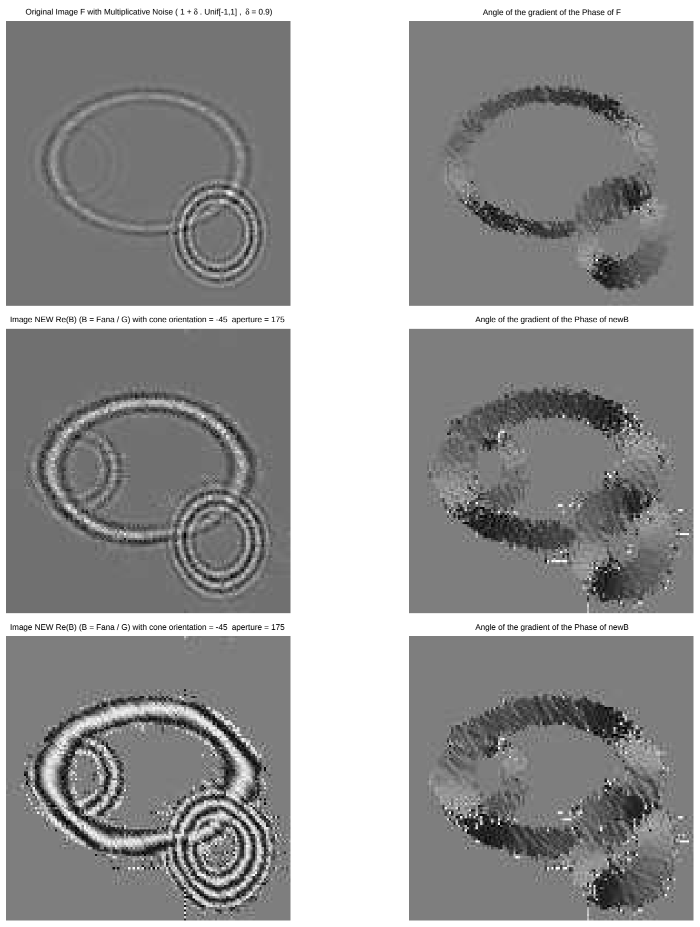
.

Figure 1.37: *Two B obtained for the "three circles" (shown on figure 1.33) with two different ϵ-filterings. The left image has a smaller ϵ and then a better contrast than the right image. But we can observe that some artifacts appear when ϵ starts to be too small as on the left figure.*

better contrast. We also observe that the more we gain on the contrast the more we loose on the spatial localization of the oscillations. There, we can conclude that a trade-off has to be done.

1.9 Stability to noise

In this chapter, we test the algorithm with a similar image as before except that we made modifications to transform the circles on ellipses, to avoid rotation symmetries in the image. We also add a multiplicative noise to observe the stability of the transform as we see on figure 1.38. We are interested in the gradient phase of the signal, and we observe that we increased the regions where we compute it. We observe that a large value for the multiplicative noise is not a main problem for the process. We still have an efficient result. Artifacts appear in region where $|F_{H_i}|$ is small.

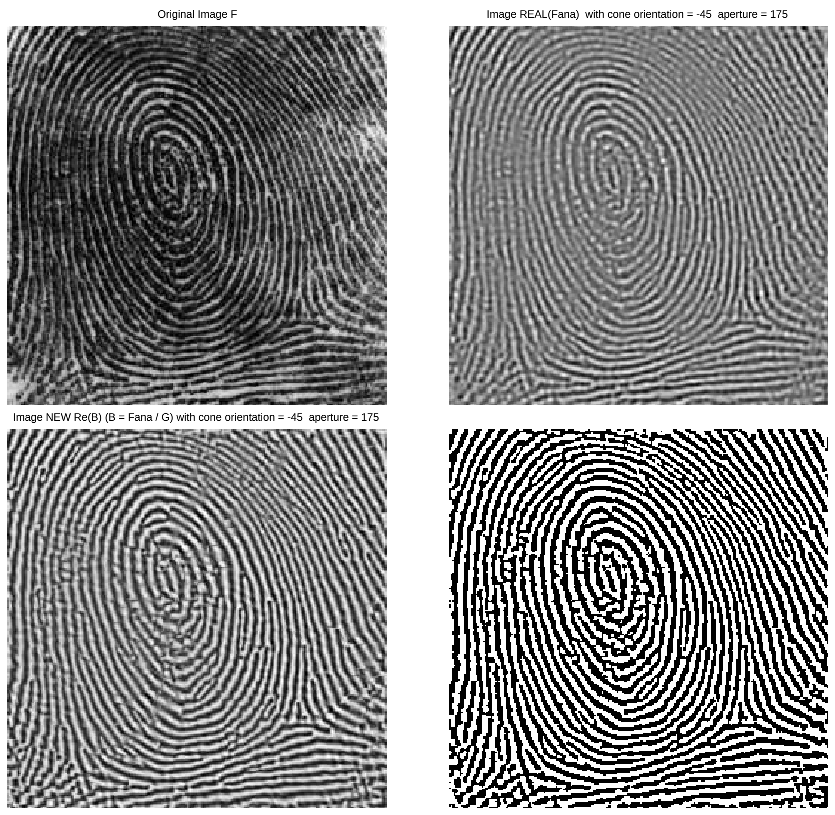

Original Image F with Multiplicative Noise (1 + δ . Unif[-1,1] , δ = 0.9)
Angle of the gradient of the Phase of F
Image NEW Re(B) (B = Fana / G) with cone orientation = -45 aperture = 175
Angle of the gradient of the Phase of newB
Image NEW Re(B) (B = Fana / G) with cone orientation = -45 aperture = 175
Angle of the gradient of the Phase of newB

.

Figure 1.38: *The top row shows the original image $Re(F_{H_i})$ (on the left) "one circle and two ellipses" and the corresponding angle of $\nabla\phi_{F_{H_i}}$. The second and third rows represent $B_{H_i,H_i,\epsilon}$ with two ϵ-filtering and the angle of the $\nabla\phi_B$. The value of ϵ is lower for the bottom row, thus it raises $|B_{H_i,H_i,\epsilon}|$ to one in wider regions than the second row*

1.9.1 Real Images

We can observe the result on a real image. Figure 1.39 shows the improvement between the analytic image that contains half of the Fourier information and the extension to the Blaschke product.

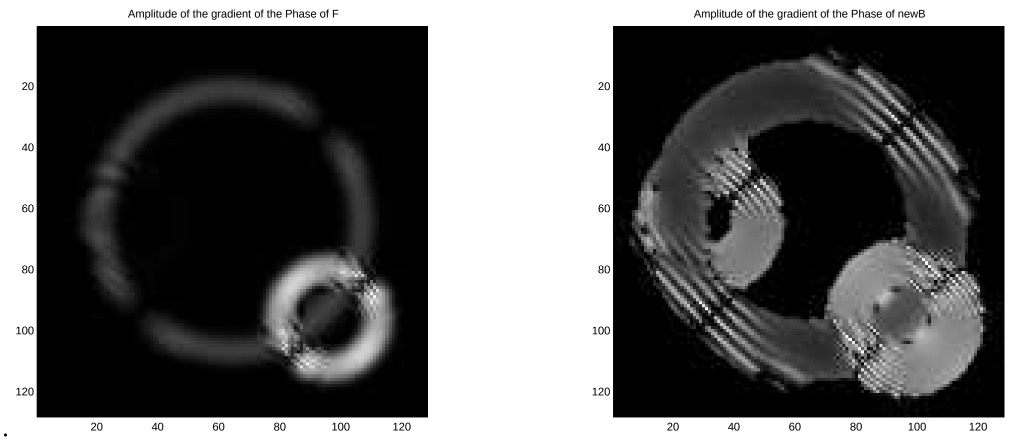

Original Image F
Image REAL(Fana) with cone orientation = -45 aperture = 175
Image NEW Re(B) (B = Fana / G) with cone orientation = -45 aperture = 175

.

Figure 1.39: *The top row shows the original fingerprint and the real part of the complexified signal without high and low frequencies. We mainly keep the frequencies corresponding to the "stripes oscillations". On the bottom row, the left figure represents the "Blaschke Product" while the right one is its signum. We observe that the contrast has been increased from the top to the bottom.*

1.9.2 Eventual segmentation

As observed in the one dimensional case, B_{C_i,H_i} enables us to compute the instantaneous frequencies of our signal that belongs to a region C_i. After thresholding, we compute the gradient of the phase of B_{C_i,H_i} on the region C_i selected on wider regions than for F_{C_i} as we have seen before. For each point, pixel, we can then attribuate its $\nabla\phi F_{C_i}$ and $\nabla\phi B_{C_i,H_i,\epsilon}$



.

Figure 1.40: *The amplitude of the gradient of the phase of F_{H_i} and B_{H_i,H_i} obtained for the "three circles" (shown on figure 1.33) with an half plane centered at -45 degres and an ϵ-threshold. We observe artifact due to the fact that there exist no analytic signal with a symmetry of rotation. The right image, obtained from B, has obviously extended the domain where the phase's gradient can be computed.*

related to each C_i. As we have seen before we have increased the area of the regions where the phase gradient exists or has a meaningful value. Using these different gradients and the low frequencies components we obtain a vector image that can be used for vectorial segmentation

1.10 Conclusion

In the one dimsenional case, we have shown invariance and stability properties of the Blaschke product using our factorization representation. We have shown how to work with "numerical compact" signal by adapting of ϵ for the ϵ-threshold. We presented an orthogonal decomposition based on the Blaschke products can be done and gives good results in many cases. An optimisation obtained by extension with a localized factor, $z \mapsto \frac{\sqrt{1-|\beta|^2}}{1-\bar{\beta}\cdot z}$, is also possible but need an algorithm to find the optimal β.

The two dimensional extension of the Blaschke product, based on the same steps, enables a similar representation. It renormalizes our function to an absolute value equals to one, in the regions containing the oscillations that have frequencies in the region selected in the Fourier space. More has to be studiedin the two dimensional case, especially to test the stability to noise and how to choose the partition of the Fourier space for the two steps. The pseudo polar Fourier transform, described and implemented in [10], is also an option to the cartesian Fourier transform as it may give a representation more adapted, depending on which filter has to be created.

.1 A rich class

In this chapter, we provide a detailed proof of the Theorem 1.6.1. The Jensen inequality enables us to write:

$$E(\exp\Big[\int \log(|F|)d\theta\Big]) \geq \exp\left[\frac{1}{2\pi}\int_0^{2\pi} d\theta \cdot \int_{\Sigma_{2N-1}} \log(|F|)d\sigma\right] \tag{92}$$

As we know that F is a random trigonometric polynomial, we can write:

$$F(\theta) = \sum_{k=0}^{N} a_k \cdot e^{ik\theta} \quad \Rightarrow \quad F(\theta) = A \cdot E_\theta \sqrt{N} \tag{93}$$

$$\text{where } E_\theta = \{\frac{1}{\sqrt{N}} e^{-ik\theta}\}, k = 0 \cdots N, \quad \text{and } A = \{a_k\}$$

E_θ is a unit vector which can be rotated to the first coordinate axis without changing $E(\log|F|)$. So we have:

$$E(\log|F|) \;=\; E\Big(\log|A \cdot E_\theta| + \frac{\log(N)}{2}\Big) \tag{94}$$

$$E(\log|F|) \;=\; E(\log|A \cdot E_\theta|) + \frac{\log(N)}{2} \tag{95}$$

The Lebesgue measure on $\Sigma_{2N-1} = \{z : |z_1|^2 + \cdots + |z_N|^2 = 1\}$ can be written as the following:

$$E(\phi(z_1, \cdots, z_N)) = \int \phi d\sigma_{N-1}(r) d\theta_1 \cdots d\theta_1, \tag{96}$$

$$\text{where } z_i = r_i \cdot e^{i\theta_i}, \quad r_i \geq 0, \quad \sum_{i=0}^{N} r_i^2 = 1.$$

$d\sigma_{N-1}(r)$ is renormalized to 1 on Σ_{2N-1} to have E(1) = 1. We obtain:

$$E(\log|A \cdot E_\theta|) \;=\; c_N \cdot \int_0^{\pi} \log|\cos(\psi)| \sin^{N-1}(\psi) d\psi \tag{97}$$

Where $c_N = \frac{\sqrt{\pi}\cdot\Gamma(\frac{N}{2})}{\Gamma(\frac{1+N}{2})}$. And if we let γ be the Euler constant we have:

$$E(\log|A \cdot E_\theta|) = -\left(\gamma + \log(4) + \frac{\Gamma'(\frac{1+N}{2})}{\Gamma(\frac{1+N}{2})}\right)/2 \tag{98}$$

There is an asymptotic formula for the digamma function:

$$\begin{aligned}
\frac{\Gamma'(N)}{\Gamma(N)} &= \log(N) + o(1) && (99)\\
\Rightarrow \frac{\Gamma'(\frac{N+1}{2})}{\Gamma(\frac{N+1}{2})} &= \log(N) - \log(2) + o(1) && (100)
\end{aligned}$$

Finally we have

$$\begin{aligned}
E(\log|F|) &= -\frac{\gamma + \log(2)}{2} + o(1) && (101)\\
\Rightarrow \quad c &= \exp\big(-(\gamma + \log(2))/2\big) && (102)\\
\Rightarrow \quad c &= 0.52\ldots && (103)
\end{aligned}$$

Chapter 2

A Vectorial Segmentation Algorithm

2.1 Introduction

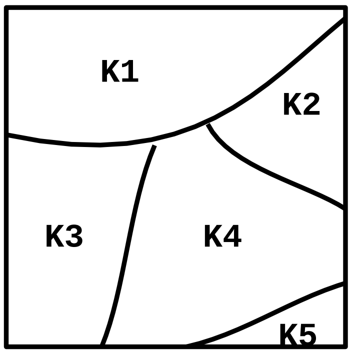
Segmentation is a key factor for image processing, it enables "to extract in homogeneous regions separated by edges". The term homogeneous has to be understood in a very broad sense. The regions can be piecewise constant, have a repetitive pattern or texture as seen in Brodatz book [13]. Works on the subject have led to a better understanding: Textons by Julesz [17], Wavelets representation with Mallat [21], Functional with Mumford and Shah [25]. These different theories generated recently many algorithms. We know the efficiency of the pyramidal algorithm presented by J.-M. Morel, and the CEREMADE (University Paris IX-Dauphine). This algorithm is based on the Mumford-Shah functional, a decreasing function of the number of regions, and is part of the Megawave platform [15]. We study the pyramidal algorithm that gives a segmentation for a predetermined number of regions. Some properties of convexity are shown for this algorithm. We use these properties to introduce an extra term, related to the number of regions, in the functional. This new functional has now a minima depending on the number of regions. We also show a counter example for the non-optimality of the algorithm. The algorithm approximates regions by a constant, so it implies that for textured images a preprocessing has to be done. Koepfler et al. showed some good results in [19] for textural image using a vectorial segmentation after a wavelets preprocessing. A summary of wavelets properties is then given as we will use them for our multi-scale representation, we also show the fast decomposition done by Mallat and present the Wavelet packets. These algorithms use decimation. But we want a grid independent algorithm so we introduce the notion of undecimated wavelets and present different preprocessing. A cost function is then applied to evaluate the usefulness of the filtering, we want to avoid "uninteresting filters" to shorten computations time and improve efficiency. We obtain the "Cost Subspaces" on which the segmentation algorithm will run. Our aim is to find a criterion to determinate the number of regions and the efficiency of the segmentation for each subspace. This approach is based on the fact that we are interested in finding a segmentation with regions of reasonable sizes and we are not looking, for example, for "targets" that are typical of small regions. Thus we create a criterion of "segmentation efficiency" for each component and discard the insignificant filtered images. We run the pyramidal algorithm on synthetic and real images, applying different improvements.

2.2 Segmentation Algorithm

J.-M. Morel and his collaborators from the CEREMADE developed Megawave [15], a software that works as a platform containing C programs, essentially for image processing. For segmentation, a pyramidal algorithm has been developed, it is based on Mumford-Shah functional. This merging algorithm is very efficient in the case of piecewise constant images. For textured images, a vector image is obtained after preprocessing before evaluating the cost function that enables to select the useful filters. Then we can reduce computational time and errors by choosing the appropriate component of our vector image. We describe the pyramidal algorithm and explain its advantages and show problems that can occur. This description is done with the purpose to find a criterion for a good segmentation.

2.2.1 Mumford-Shah Functional

There is an obvious relation between the piecewise constant approximation on each region and the region itself. Therefore, we define Mumford-Shah functional with a parameter λ, that can be compared to the scale, by using the regions (the function corresponding is then defined by the average value on the region). To determinate the regions is equivalent to obtain the piecewise constant functions. We define $P(\Omega)$ partition of Ω, and $N(\lambda)$ number of regions for a fixed λ. And we obtain $\forall K \subset P(\Omega)$, with $K = \cup_{i=1}^{N(\lambda)} K_i$ where the domains K_i are connected, a segmentation similar to figure 2.1.



.

Figure 2.1: *Example of a segmentation in five regions for an image.*

And we have the following representation:

$$\forall u \in L^2(\Omega), \quad \exists! g \in L^2(\Omega), \text{such that } g_{|K_i} = u_{K_i} = \frac{1}{|K_i|} \cdot \int_{K_i} u \tag{2.1}$$

And we denote $\partial K = \cup_{i=1}^{N(\lambda)} \partial K_i$

$$E_\lambda(u, K) = \int_\Omega \|u - g\|^2 + \lambda.length(\partial K) \tag{2.2}$$

$$E_\lambda(u, K) = \sum_{i=1}^{N(\lambda)} \int_{K_i} \|u - u_{K_i}\|^2 + \lambda.length(\partial K) \tag{2.3}$$

We note the optimal result:

$$E_\lambda(u) = \min_{K \subset P(\Omega)} E_\lambda(u, K) \tag{2.4}$$

And we want to find K such that $E_\lambda(u, K)$ is minimum. We now present the characteristics of a pyramidal algorithm developed and programmed by J.-M. Morel and his team.

2.2.2 A pyramidal algorithm

We know that a way to solve Mumford-Shah functional, with piecewise constant function, is to use an iterative algorithm. This algorithm has been developed by J.-M. Morel and his team the CEREMADE. They define this algorithm by recursion in Koepfler's thesis [18]. This algorithm starts with a segmentation at the pixel level (we obviously have N^2 regions for a image of size (N, N)) and λ is equal to zero. A merging algorithm runs until the desired number of regions λ is reached. The pyramidal algorithm constructing 2-normal affine segmentations is defined as followed in [18]:

"We now consider the problem of defining and computing a 2-normal segmentation. Notice that not all 2-normal segmentation are interesting: for instance, the empty segmentation, where Ω is the single region is clearly a 2-normal segmentation. If the scale parameter λ is very large, it is also a reasonable segmentation since one "pays" a too large energy amount for having any boundary. However, it is obvious from the definition that the empty segmentation is 2-normal for every λ, which certainly proves that the assertion that a

segmentation is 2-normal is not enough to ensure that it is "good". But if we follow the main idea of the region growing methods, we shall see that what they compute is precisely a 2-normal sub-segmentation of a fine initial segmentation, obtained by recursive merging.

Assume that the datum g is defined on a rectangle. This rectangle is divided in small squares of constant size (the pixels) and g is assumed to be constant on each pixel. Here are the properties which we require for the segmentation computed by a region growing algorithm, defined as an application associating to g and λ a segmentation (u, K).

a) *"correctedness"(Fixed point property): Assume that g is piecewise constant on some areas of the rectangle. Then there exists a value λ_0 of the parameter λ such for that for every $\lambda < \lambda_0$, the segmentation (u, K) obtained by the algorithm verifies $u = g$ and K is the union of the boundaries of the areas where g is constant. This property has been proved to be asymptotically true for the segmentations which are global minima of the energy E as λ tends to zero. But we impose it here as a non-asymptotic property.*

b) *"Causality"(Pyramidal segmentation property): If $\lambda > \lambda^{'}$, then the boundaries provided by the algorithm for λ are contained in those obtained for $\lambda^{'}$and the areas of segmentation associated to λ are the unions of some the areas obtained for $\lambda^{'}$.*

The last Property ensures that a fast pyramidal algorithm can be implemented, computing a hierarchy of segmentations from fine to coarse scales. Moreover the coarser segmentation will be deduced from the finer by "merging" operations, with a pyramidal structure for the computation. Note that, as a consequence of the fixed point property, if λ is very small, the computed segmentation is attained with (u_0, K_0) where $U_0 = u$ and K_0 consists of all the boundaries of all the pixels and therefore coincides with the global minimum as λ is zero. We shall call this segmentation, where each pixel is a region, the "trivial segmentation". It is easy to see that recursive merging algorithm which we present now verifies all the above mentioned properties." We believe that the algorithm does not define the segmentation that we are looking for as the number of regions is not inserted in the functional. We will define later a new functional with an extra term. But before we present the concavity and convexity characteristics of the algorithm.

2.2.3 Convexity and Concavity

In this section we consider the case of the pyramidal algorithm where only two regions merge at a time. We will show later that this detail has its importance. Let's show some properties of our initial cost function $E_\lambda(u, K)$ We have a pyramidal algorithm to segment

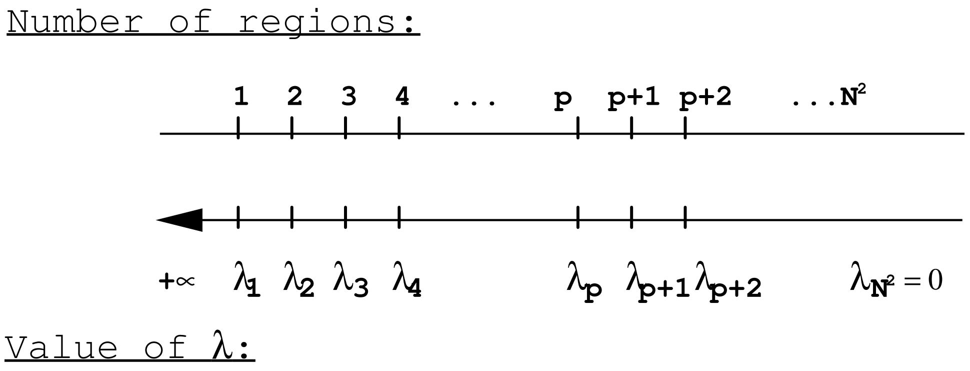

Number of regions:
1 2 3 4 ... p p+1 p+2 ...N²
+∝ λ1 λ2 λ3 λ4 λp λp+1 λp+2 λN² = 0
Value of λ:

Figure 2.2: *Diagram showing relations between λ and $N(\lambda)$ (the corresponding number of regions for the image segmentation)*

our image, and we start from a fine grid and merge neighborhood regions two by two in order to satisfy our criterion. For each couple of regions, we have a different value for λ that enables us to merge. The pyramidal algorithm merges, at each step, the couple of regions that needs the smallest λ over all the possible values, we note it λ_k at the step k. Then the number of regions is a decreasing function of λ, and stay constant between two consecutive λ_n:

$$\forall n \in N^*, \quad \forall \lambda \in [\lambda_{n+1}, \lambda_n) \quad N(\lambda) = n + 1. \tag{2.5}$$

We have $N(\lambda) \leq n$ only for $\lambda \geq \lambda_n$ as showed on the figure 2.3. And λ_n is the critical value to pass from the segmentation K to K'. Let's define the two variations:

$$\Delta E = E_{\lambda_{n+1}}(u) - E_{\lambda_n}(u) < 0 \tag{2.6}$$

$$\Delta \lambda = \lambda_{n+1} - \lambda_n \quad < 0 \tag{2.7}$$

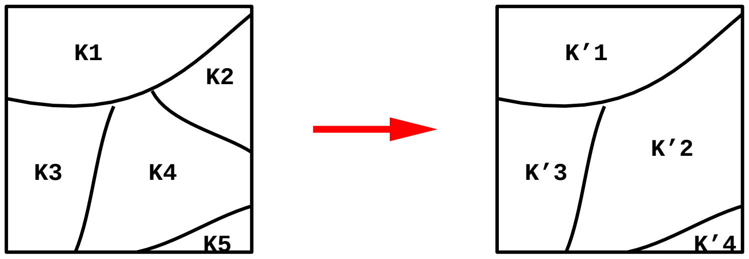
.


Figure 2.3: *Constructing $K^{'}$ from K by merging two regions for $\lambda = \lambda_4$*

Thus,

$$\boxed{\frac{\partial E}{\partial \lambda} \geq 0} \tag{2.8}$$

By construction we have $\lambda_{n+1} \leq \lambda_n$ and the number of regions is constant on each interval $[\lambda_{n+1}, \lambda_n)$ then

$$\boxed{\frac{\partial n}{\partial \lambda} \leq 0} \tag{2.9}$$

We note K and $K^{'}$ the two segmentations that minimize our functional for their respective λ_{n+1} and λ_n. We note K_i and K_j the two regions merging, for $\lambda = \lambda_n$, in order to obtain $K^{'}_p = K_i \cup K_j$ (in the figure 2.3, the indices are $i = 2$, $j = 4$, $p = 2$ and $n = 4$). E_n and E_{n+1} have terms in common, we define, for the merging region couple (K_i, K_j) related to the value of λ, the integral term corresponding to the difference:

$$I(K_i, K_j) = \int_{K^{'}_p} \|u - u_{K^{'}_p}\|^2 - \left[\int_{K_i} \|u - u_{K_i}\|^2 + \int_{K_j} \|u - u_{K_j}\|^2\right] \tag{2.10}$$

$$I(K_i, K_j) = \int_{K^{'}_p} \|u - u_{K^{'}_p}\|^2 - \|u - (u_{K_i}.\chi_{K_i} + u_{K_j}.\chi_{K_j})\|^2. \tag{2.11}$$

By definition of u_{K_i}, we have $u = (u - u_{K_i}) \oplus u_{K_i}$, Pythagoras theorem enables to write:

$$\begin{aligned} \|u - u_{K^{'}_p}\|^2 &= \|u - u_{K_i}\|^2 + \|u_{K_i} - u_{K^{'}_p}\|^2 \\ \Rightarrow I(K_i, K_j) &= \int_{K_i} \|u_{K_i} - u_{K^{'}_p}\|^2 + \int_{K_j} \|u_{K_j} - u_{K^{'}_p}\|^2 > 0 \end{aligned} \tag{2.12}$$

with $K^{'}_p = K_i \cup K_j$. After simplification, we easily obtain the relation:

$$\Delta E = I(K_i, K_j) + \lambda_{n+1} \cdot length(\partial K) - \lambda_n \cdot length(\partial K^{'}) \tag{2.13}$$

The pyramidal algorithm constructs $K^{'}$ from K, by merging two regions, K_i and K_j separated by a common boundary $\partial K_i \cap \partial K_j$, that disappears in the merging process:

$$\Delta E = I(K_i, K_j) + \Delta\lambda \cdot length(\partial K) + \lambda_n \cdot length(\partial K_i \cap \partial K_j) \tag{2.14}$$

For $\lambda \in [\lambda_{n+1}, \lambda_n)$ we have n regions, K_i and K_j merge at $\lambda = \lambda_n$. Then we have the following properties:

1. $\lambda \mapsto E_\lambda(u, K)$ is strictly increasing on $[\lambda_{n+1}, \lambda_n[$, as $E_\lambda(u, K)$ evolves linearly like $\lambda \mapsto \lambda \cdot length(\partial K)$ on the interval.

2. For $\lambda = \lambda_n$, $K^{'}$ minimizes $K \mapsto E_{\lambda_n}(u, K)$, as there are n regions.

3. For $\lambda \in [\lambda_{n+1}, \lambda_n[$, K minimizes $K \mapsto E_\lambda(u, K)$ for $(n+1)$ regions.

We deduce from these previous properties that $\forall \lambda \in [\lambda_{n+1}, \lambda_n)$:

$$\boxed{E_\lambda(u, K) = E_{\lambda_{n+1}}(u, K) + (\lambda - \lambda_{n+1}) \cdot length(\partial K)} \tag{2.15}$$

But $\forall n \in N^*, \lambda_n > \lambda_{n+1}$ and $E_{\lambda_n}(u, K) > E_{\lambda_{n+1}}(u, K)$. We also have:

$$\begin{aligned} E_{\lambda_n}(u, K) &\geq E_{\lambda_n}(u, K^{'}) \\ \Rightarrow \quad \lambda_n \cdot length(\partial K_i \cap \partial K_j) &\geq I(K_i, K_j) \\ \Rightarrow \quad \lambda_n &\geq \lambda_{K_i, K_j} = \frac{I(K_i, K_j)}{length(\partial K_i \cap \partial K_j)} \end{aligned} \tag{2.16}$$

But for $\lambda \in [\lambda_{n+1}, \lambda_n)$, we have a segmentation with $(n+1)$ regions. We can conclude that $\exists \delta > 0, \forall 0 < \epsilon < \delta$:

$$\begin{aligned} E_{(\lambda_n - \epsilon)}(u, K^{'}) &\geq E_{(\lambda_n - \epsilon)}(u, K) \\ \lambda_n - \epsilon &\leq \lambda_{K_i, K_j} \\ \Rightarrow \quad \lambda_n &= \frac{I(K_i, K_j)}{length(\partial K_i \cap \partial K_j)} \end{aligned} \tag{2.17}$$

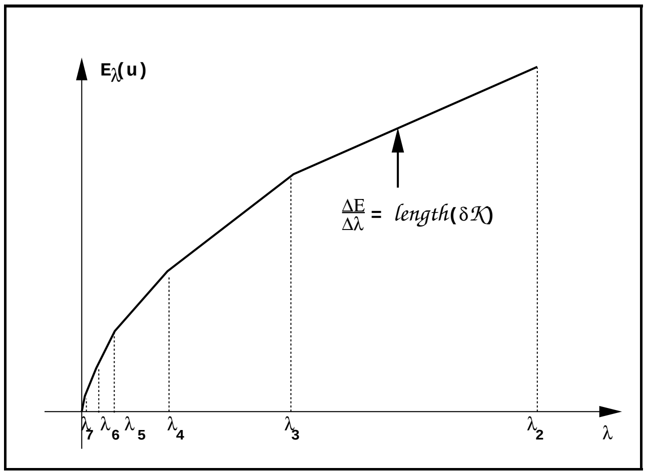
We obtain then the following formula that determines the λ necessary for two regions to merge:

$$\boxed{\lambda_n = \frac{\int_{K_i} \|u_{K_i} - u_{K_p^{'}}\|^2 + \int_{K_j} \|u_{K_j} - u_{K_p^{'}}\|^2}{length(\partial K_i \cap \partial K_j)}} \tag{2.18}$$

The energy's variation between two consecutive merging is then:

$$\begin{aligned} \Delta E &= \Delta\lambda \cdot length(\partial K) \\ \Rightarrow \quad \frac{\Delta E}{\Delta\lambda} &= length(\partial K) \end{aligned} \tag{2.19}$$

We know that $length(\partial K) > length(\partial K')$, as one boundary between two regions has disappeared. So $\frac{\Delta E}{\Delta\lambda}$ is a decreasing function of the number of regions. It means that we have a graph similar to figure 2.4:



.

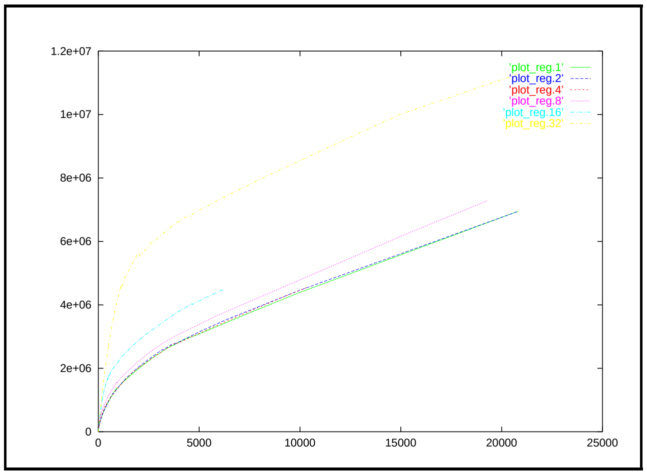
Figure 2.4: *Graph of the theoretical* $\lambda \mapsto E_\lambda(u)$

We compare this result to our experiments (image + different levels of noise) on figure 2.5, and we observe that experimentally we also have concavity of $\lambda \mapsto E_\lambda(u)$ as shown on the theoretical figure 2.4. We observe that the slopes of the asymptote are the same for the different images. We deduce that the length of the boundaries for the image at the last segmentation are similar.

$$\frac{\Delta E}{\Delta\lambda}(\lambda_{n+1}) > \frac{\Delta E}{\Delta\lambda}(\lambda_n) \tag{2.20}$$

and $\lambda \mapsto E_\lambda(u)$ is piecewise linear on each interval $[\lambda_{n+1}, \lambda_n]$. The concavity of E_λ is then obvious as:

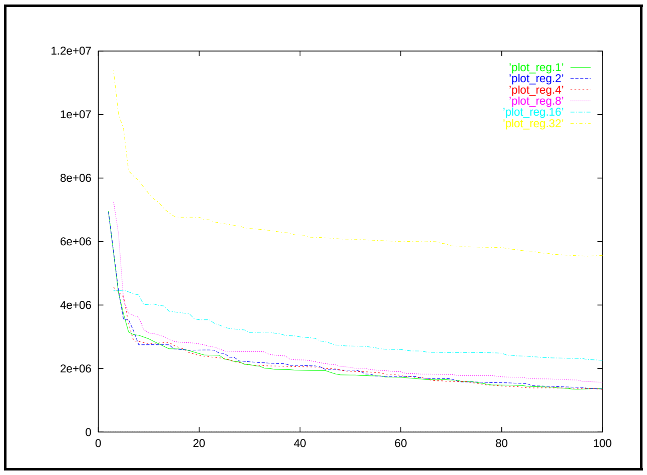
$$\boxed{\frac{\partial^2 E_\lambda(u)}{\partial_2 \lambda} < 0} \tag{2.21}$$



.

Figure 2.5: *Graphs of experimental* $\lambda \mapsto E_\lambda(u)$*, an image with different Gaussian noises added.*

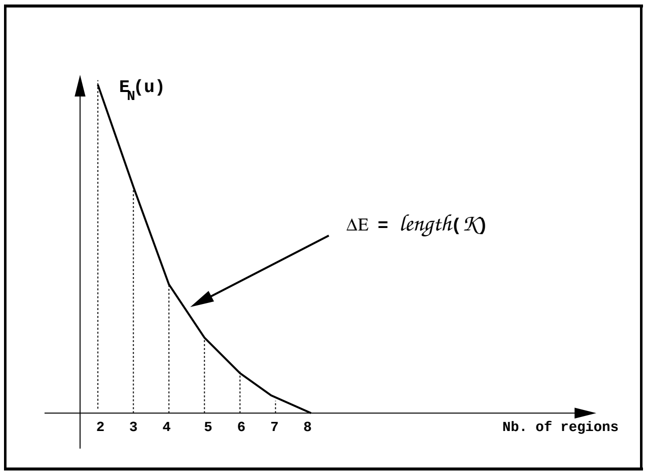
We showed some properties for $\lambda \mapsto E_\lambda(u)$ and we would like to have a similar relation for $n \mapsto E_{\lambda_n}(u)$. We can first look at the graph of this function. The experiments seem to



.

Figure 2.6: *Graph of experimental* $n \mapsto E_n(u) = E_{\lambda_n}(u)$*, an image with different Gaussian noises added*

agree on the convexity of $n \mapsto E_{\lambda_n}(u)$. But we have to verify this property more carefully. Especially the way the first derivate evolves when the regions merge.



.

Figure 2.7: *Graph of the theoretical* $n \mapsto E_n(u) = E_{\lambda_n}(u)$

We have an obvious property coming from the study of $\lambda \mapsto E_\lambda(u)$:

$$\Delta E = \Delta\lambda \cdot length(\partial K) < 0, \text{ thus } \frac{\partial E}{\partial \lambda} < 0 \tag{2.22}$$

We know that $n \mapsto length(\partial K_n)$ is increasing and $\Delta\lambda < 0$. We would like to have a convexity property for $n \mapsto E_{\lambda_n}(u)$. We will see later that this property is linked to the structure of the image.

2.2.4 Mumford-Shah and Extra Term

We note that the Mumford-Shah functional does not depend on the number of regions, but only on the length of these boundaries. It means that there will be no penalties for a large number of small regions. But we are mostly interested in segmented images with a small number of regions, for example three to a dozen of regions seems like a reasonable number. We add an extra term to the Mumford-Shah functional that will take care of the complexity of the image. This idea is similar to the compression problem. We want, for a fixed budget, to define a fixed number of regions, with a certain complexity, and a minimal error. We define an extension of Mumford-Shah functional with an extra term, in this case we force our functional to minimize the number of regions also.

$$E^{'}_{\lambda,\nu}(u,K) = \int_{\Omega} \|u-g\|^2 + \lambda.length(\partial K) + \nu.N(\lambda) \tag{2.23}$$

$$E^{'}_{\lambda,\nu}(u,K) = \sum_{i=1}^{N(\lambda)} \int_{K_i} \|u-u_{K_i}\|^2 + \lambda \cdot length(\partial K) + \nu \cdot N(\lambda) \tag{2.24}$$

We can notice that by increasing the value of λ, we indirectly decrease the number of regions $(N(\lambda))$. Our new functional is the sum of a concave and convex functions.

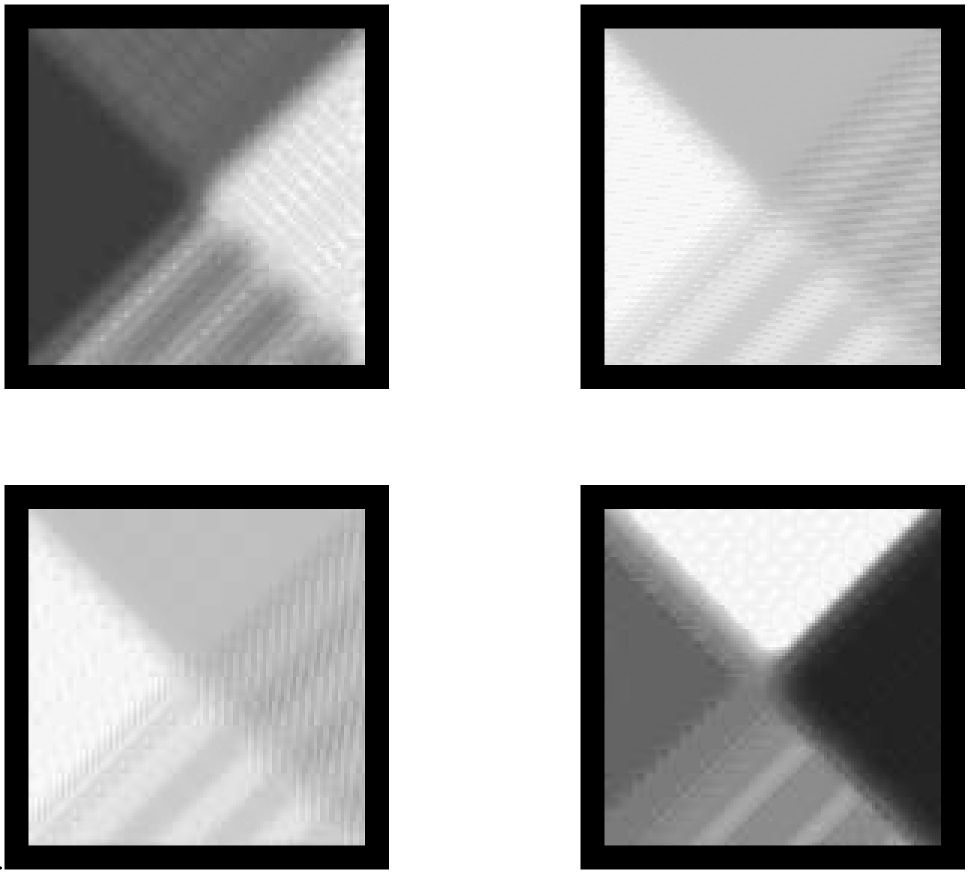


Figure 2.8: *Four images coming from the same image after filtering*

Let's apply this algorithm on the four filtered images showed on figure 2.8, coming from a "wave image" described in chapter 2.6.2. We added to these filtered images a crown for some boundaries problem, the reason will be explained in the same chapter. We want to know which one can give a reasonable segmentation. We mean by that reasonable: no small region, average values in each regions not too close. And we obtain the graph on figure 2.9, it gives us two informations. The first one is that “Image1” and “Image4” should be better for segmentation since they have a minima . The five regions can be seen on the four images but we observe that the contrast is much better in these two images. The second remark is that the canonical segmentation should be done with five or six regions as the minima are obtained for these number of regions.

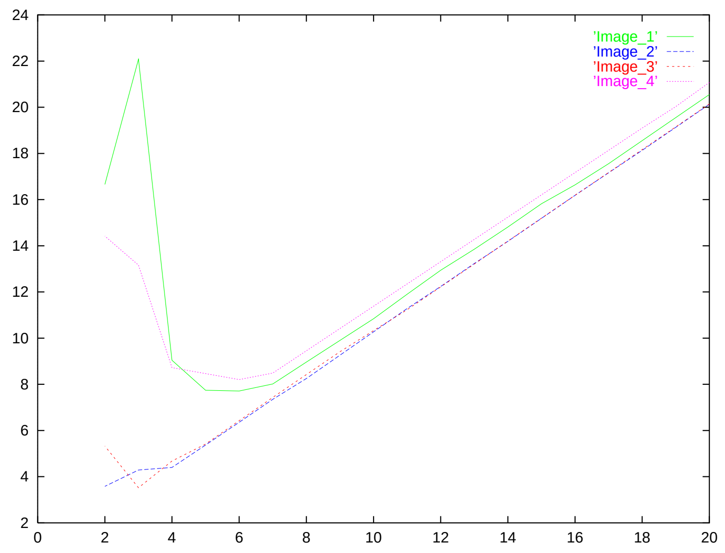


Figure 2.9: *Graphs of the “Mumford Shah + Extra Term” for the Images on figure 2.8. Two of the four images seem to give an optimal segmentation for five or six regions. It confirms our visual impression.*

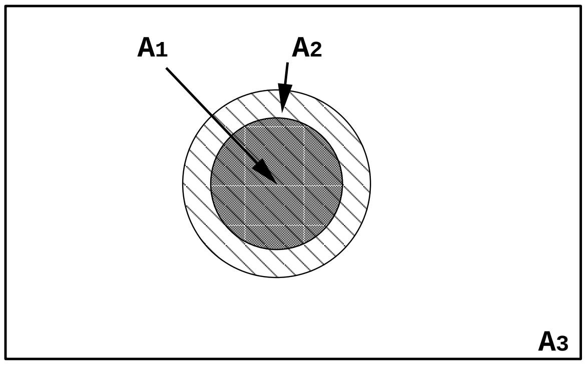
2.2.5 Counter-example for the pyramidal algorithm

We ran the pyramidal segmentation algorithm on different examples to test its robustness to noise. In some cases, we note that the function $N_\lambda \mapsto E_\lambda$ is not decreasing or $\lambda \mapsto E_\lambda$ is only piecewise non-decreasing and concave, as some discontinuity appear and make the function not globally non-decreasing. It would mean that the pyramidal algorithm is maybe not optimal. In this chapter, we build a simple example to understand what is happening.

Let’s take an image (N by N pixels), containing three regions. A disk A_1 with a crown A_2 around of respective radius r_1 and r_2 with $r_2 = r_1\sqrt{2}$, obviously $mes(A_1) =$

$mes(A_2)$. We decide to have $mes(A_3) = (4\pi - 2) \cdot mes(A_2)$, that implies $N^2 = (4\pi - 2 + 1 + 1) mes(A_1)$. And we obtain $N = 2\pi r_1$. Each region A_i can be characterized by a piecewise constant function. We represent the function on the figure 2.10 and we have the intensity of the pixel value that define the function I:

$$I(x) = -1_{A_1}(x) + 1_{A_2}(x) - \gamma \cdot 1_{A_3}(x), \text{ with } \gamma > 0. \tag{2.25}$$



.

Figure 2.10: *Image containing three regions for our counter-example*

We run the Pyramidal algorithm on our image (figure 2.10). For λ equals to zero we have the "trivial segmentation", with N^2 regions, and we obtain:

$$E_0 = \int_\Omega 0 + 0 \cdot (2 \cdot N^2 - 4 \cdot N) \tag{2.26}$$

Since we have N^2 regions, the error on each region is equal to zero, and we don't count the outside boundaries of the image. When λ starts to be greater than zero, if there is no regions merging we have:

$$E_\lambda = \lambda \cdot (2 \cdot N^2 - 4 \cdot N) \tag{2.27}$$

But we observe that by merging most of the regions we obtain our three regions without increasing the term corresponding to the approximations error for each region (the integral term) in the functional. We then have a new relation for E_λ :

$$\begin{aligned} E_\lambda &= 0 + \lambda \cdot (l(\partial A_1) + l(\partial A_2)) \\ \Rightarrow \quad E_\lambda &= \lambda \cdot 2\pi r_1 (1 + \sqrt{2}) \end{aligned} \tag{2.28}$$

We conclude that the region merging will appear as soon as $\lambda \neq 0$. This segmentation will be valid until merging two of the three regions. If we merge A_1 and A_2, it is obvious that the average value on $A_1 \cup A_2$ is $\gamma_{1-2} = 0$. Formula 2.18 enables us to compute the threshold value for λ:

$$\begin{aligned} \lambda_{1-2} &= \frac{1^2.mes(A_1) + 1^2.mes(A_2)}{l(\partial A_1)} && (2.29) \\ \Rightarrow \quad \lambda_{1-2} &= r_1 && (2.30) \end{aligned}$$

In the case where A_2 and A_3 merge, the average value γ_{2-3} on the new region $A_2 \cup A_3$ and the λ_{2-3} necessary for merging A_2 and A_3 are easily obtained:

$$\begin{aligned} \gamma_{2-3} &= \frac{1 - (4\pi - 2) \cdot \gamma}{4\pi - 1} && (2.31) \\ \lambda_{2-3} &= r_1 \cdot \frac{(\gamma + 1)^2 \cdot (4\pi - 2)}{2\sqrt{2}(4\pi - 1)} && (2.32) \end{aligned}$$

We observe that if γ is close enough to zero ($0 < \gamma < 0.5$), $\lambda_{2-3} < \lambda_{1-2}$, it means that we have to merge A_2 and A_3 before merging A_1 and A_2. Until $\lambda \leq \lambda_{2-3}$, we then have three regions as a result of our segmentation, and E_λ defined by the relation 2.28. And for $\lambda = \lambda_{2-3}$ we have the following equation for the functional:

$$E_{\lambda_{2-3}} = \pi r_1^2 \cdot \frac{(\gamma + 1)^2 \cdot (4\pi - 2)}{(4\pi - 1)} (1 + \frac{1}{\sqrt{2}}) \qquad (2.33)$$

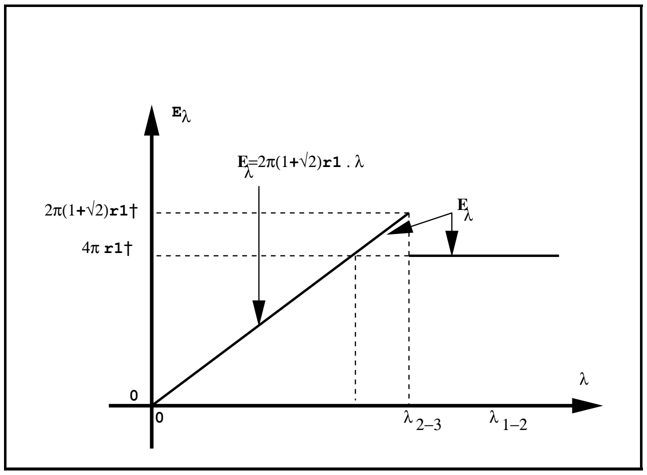
We now compute the functional for only one region ($A_1 \cup A_2 \cup A_3$), we easily have that the average value $\gamma_{1-2-3} = -\frac{\gamma(2\pi - 1)}{2\pi}$. And then we obtain the value of our functional, knowing that the second term corresponding to the boundaries term is equal to zero:

$$E_{\lambda_{1-2-3}} = \pi r_1^2 \cdot (1 + \gamma^2 \frac{(2\pi - 1)^2}{(2\pi)^2}) \qquad (2.34)$$

To obtain an interesting example, we set $\gamma = \frac{1}{4}$ and obtain:

$$\begin{aligned} E_{\lambda_{2-3}} &= \pi r_1^2 \cdot \frac{25(1 + \sqrt{2})}{16\sqrt{2}} \cdot \frac{4\pi - 2}{4\pi - 1} && (2.35) \\ E_{\lambda_{1-2-3}} &= \pi r_1^2 \cdot (1 + \frac{(2\pi - 1)^2}{(8\pi)^2}) && (2.36) \end{aligned}$$

Since $N \mapsto E_N$ is increasing, as we have seen before, it means that the merging of the three regions should have been done before the merging of any two regions individually. We deduce from this experience that in some case where a given value of λ make merge more than one region at a time, the pyramidal algorithm is not optimal anymore. We now



.

Figure 2.11: *Graph showing the non-optimality of the pyramidal algorithm, Energy as a function of* λ

summarize the properties of the wavelet representation.

2.3 Continuous Wavelets Recapitulatory

In this section, we present some of the main properties and theorems about the wavelets decomposition. We follow Mallat [22] in his study, show the difference between undecimated and decimated wavelets. We introduce also the concept of wavelet packets decomposition [14].

2.3.1 Theorems and Properties on Wavelets

Theorem 2.3.1. *In the case of a wavelet multi-resolution approximation, we have a sequence of closed subspaces $\{V_j\}_{j\in Z} \subset L^2(\mathbb{R})$, that verifies the followings properties*

$$\forall (j,k) \in \mathbb{Z}^2, \quad f(\cdot) \in V_j \Leftrightarrow f(\cdot - 2^j k) \in V_j \tag{2.37}$$

$$\forall j \in \mathbb{Z}, \quad f(\cdot) \in V_j \Leftrightarrow f(\frac{\cdot}{2}) \in V_{j+1} \tag{2.38}$$

$$\ldots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \ldots \tag{2.39}$$

$$\lim_{j\to -\infty} V_j = \overline{\cup_{j\in Z} V_j} = L^2(\mathbb{R}) \tag{2.40}$$

$$\lim_{j\to +\infty} V_j = \cap_{j\in Z} V_j = \{0\} \tag{2.41}$$

And we know that there exists θ such that $\{\theta(. - n)\}_{n\in Z}$ is a Riesz basis of V_0. For example in the Haar case, it corresponds to the piecewise constant approximation. We have $\theta = \chi_{[0,1]}$. And so V_j represents the functions f such that f is constant on $[k \cdot 2^j, (k+1) \cdot 2^{(j+1)})$.

Theorem 2.3.2. *Let $\{V_j\}_{j\in Z}$ be a multi-resolution approximation and ϕ a scaling function such that:*

$$\widehat{\phi}(\xi) = \frac{\widehat{\theta}(\xi)}{\left(\sum_{-\infty}^{+\infty} |\widehat{\theta}(\xi + 2k\pi)|^2\right)^{\frac{1}{2}}} \tag{2.42}$$

And we let

$$\phi_{j,n}(\cdot) = \frac{1}{2^j}\phi(\frac{\cdot - 2^j n}{2^j}) \tag{2.43}$$

For all $j \in Z$, $\{\phi_{j,n}\}_{n\in Z}$ is an orthonormal basis of V_j

Remark 2.3.1. *We define an approximation, at a level j, over V_j by using an expansion in the scaling orthogonal basis:*

$$P_{V_j} f = \sum_{-\infty}^{+\infty} \langle \phi_{j,n}, f \rangle \phi_{j,n} \tag{2.44}$$

We can easily see that $\langle \phi_{j,n}, f \rangle = f * \phi_j(2^j n)$, where $*$ represents the convolution.

Theorem 2.3.3. *(Mallat, Meyer)*

Let $\phi \in \mathcal{L}^2(\mathbb{R})(\mathbb{R})$ be an integrable scaling function. The Fourier series of $h[n] = \langle \frac{1}{\sqrt{2}} \phi(\frac{\cdot}{2}), \phi(\cdot - n) \rangle$ satisfies:

$$\forall \xi \in \mathbb{R}, \quad |\widehat{h}(\xi)|^2 + |\widehat{h}(\xi + \pi)|^2 = 2 \tag{2.45}$$

and

$$|\widehat{h}(0)|^2 = 2. \tag{2.46}$$

Conversely, if $\widehat{h}$ is a 2π-periodic and continuously differentiable in a neighborhood of zero, if it satisfies the two precedent properties and if

$$\inf_{\xi \in [-\frac{\pi}{2}, \frac{\pi}{2}]} |\widehat{h}(\xi)| > 0 \tag{2.47}$$

and

$$\widehat{\phi}(\xi) = \prod_{p=1}^{+\infty} \frac{\widehat{h}(2^{-p}\xi)}{\sqrt{2}} \tag{2.48}$$

Then $\widehat{\phi}$ is the Fourier transform of a scaling function $\phi \in L^2(\mathbb{R})$.

Remark 2.3.2. *For piecewise constant approximations, $\phi = \chi_{[0,1]}$. Since $h[n] = \langle \frac{1}{\sqrt{2}} \phi(\frac{\cdot}{2}), \phi(\cdot - n) \rangle$ it follows that*

$$h[n] = \begin{cases} \frac{1}{\sqrt{2}} & \text{if} \quad n = 0, 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.49}$$

With the approximations of f, at the scales 2^j and 2^{j-1}, that are equal to their orthogonal projections on V_j and V_{j-1}. We know that $V_j \subset V_{j-1}$. Let W_j be the orthogonal complement of V_j in V_{j-1}, i.e.

$$V_{j-1} = V_j \oplus W_j. \tag{2.50}$$

So the orthogonal projection of f on V_{j-1} can be decomposed as the sum of two orthogonal projections:

$$P_{V_{j-1}} f = P_{V_j} f + P_{W_j} f. \tag{2.51}$$

$P_{W_j} f$ provides the "details" of f that appear at the scale 2^{j-1} but which disappear at the coarser scale 2^j. Furthermore from the previous relations we can easily show that $\oplus_{j \in \mathbb{Z}} W_j = L^2(\mathbb{R})$.

Theorem 2.3.4. *(Mallat, Meyer):*

Let ϕ be a scaling function and h the corresponding conjugate mirror filter. Let φ be the function whose Fourier transform is:

$$\widehat{\varphi}(\xi) = \frac{1}{\sqrt{2}} \widehat{g}\left(\frac{\xi}{2}\right) \widehat{\varphi}\left(\frac{\xi}{2}\right), \tag{2.52}$$

with

$$\widehat{g}(\xi) = e^{-i\xi} \overline{\widehat{h}}(\xi + \pi), \tag{2.53}$$

And we note

$$\varphi_{j,n}(\cdot) = \frac{1}{2^j} \varphi\left(\frac{\cdot - 2^j n}{2^j}\right) \tag{2.54}$$

for any scale 2^j, $\{\varphi_{j,n}\}_{n \in Z}$ is an orthonormal basis of W_j. For all scales, $\{\varphi_{j,n}\}_{j,n \in Z^2}$ is an orthonormal basis of $L^2(\mathbb{R})$.

We describe now a fast algorithm to compute the wavelets decomposition.

2.3.2 Fast Algorithm

We are going to describe a fast filter bank algorithm designed by S. Mallat. This algorithm computes the orthogonal wavelet coefficients of a discrete signal $(a_0[n])_n$. That corresponds to a decimate wavelets decomposition. Let us define:

$$f = \sum_{n=-\infty}^{+\infty} a_0[n]\phi(. - n) \in V_0. \tag{2.55}$$

Since $\{\phi(. - n)\}_{n\in Z}$ is orthonormal, we have:

$$a_0[n] = \langle f(.), \phi(. - n)\rangle \tag{2.56}$$

Each $a_0[n]$ is thus a weighted average of f in the neighborhood of n. The discrete wavelet coefficients of a_0 are defined to be the wavelet coefficients of f:

$$d_j[n] = \langle f, \varphi_{j,n}\rangle \tag{2.57}$$

And we denote $\bar{x}[n] = x[-n]$ and

$$\check{x}[n] = \begin{cases} x[p] & \text{if} \quad n = 2p \\ 0 & \text{otherwise} \end{cases} \tag{2.58}$$

So the following theorem shows how to compute the wavelet decomposition and reconstruction with discrete convolutions.

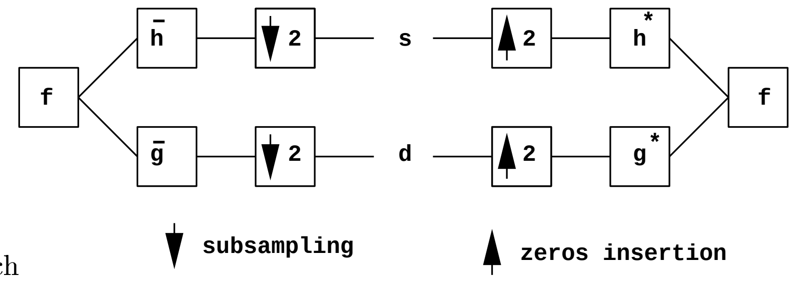
Theorem 2.3.5. *(Mallat)*

For the decomposition we have:

$$a_{j+1}[p] = \sum_{n=-\infty}^{+\infty} a_j[n]h[n-2p] = a_j * \bar{h}[2p], \tag{2.59}$$

and

$$d_{j+1}[p] = \sum_{n=-\infty}^{+\infty} a_j[n]g[n-2p] = a_j * \bar{g}[2p]. \tag{2.60}$$



chch

Figure 2.12: *A signal f is filtered by a low-pass and high-pass filter respectively to give s and d after subsampling, while an insertion of zero with dual filters reconstruct f*

Concerning the reconstruction we have:

$$\begin{aligned} a_j[p] &= \sum_{n=-\infty}^{+\infty} a_{j+1}[n]h[p-n] + \sum_{n=-\infty}^{+\infty} d_{j+1}[n]g[p-n] \\ &= \check{a}_{j+1} * h[n] + \check{d}_{j+1} * g[n]. \end{aligned} \tag{2.61}$$

The perfect decomposition is ensured by the next theorem.

Theorem 2.3.6. *(Vetterli)*

The filter bank performs an exact reconstruction for any input signal if and only if :

$$\overline{\widehat{h}}(\xi+\pi).\widehat{\tilde{h}}(\xi) + \overline{\widehat{g}}(\xi+\pi).\widehat{\tilde{g}}(\xi) = 0 \tag{2.62}$$

and

$$\overline{\widehat{h}}(\xi).\widehat{\tilde{h}}(\xi) + \overline{\widehat{g}}(\xi).\widehat{\tilde{g}}(\xi) = 2 \tag{2.63}$$

So we obtain a perfect decomposition reconstruction by using convolutions and decimations as we can see on the figure 2.12. By sub-sampling, we modify the relations between our wavelet decomposition coefficients and the original signal. Since we have $\widehat{x}(\xi) = \sum_{n=-\infty}^{\infty} x[n]e^{-in}$, the Fourier series of the subsampled signal, $y[n] = x[2n]$, is going to be such that:

$$\begin{aligned}
\widehat{y}(\xi) &= \sum_{n=-\infty}^{\infty} x[2n]e^{-in\xi} && (2.64)\\
\Rightarrow \quad \widehat{y}(2\xi) &= \sum_{n=-\infty}^{\infty} x[2n]e^{-2in\xi} && (2.65)\\
\Rightarrow \quad \widehat{y}(2\xi) &= \sum_{n=-\infty}^{\infty} (\frac{1+(-1)^n}{2})x[n]e^{-in\xi} && (2.66)\\
\Rightarrow \quad \widehat{y}(2\xi) &= \frac{1}{2}(\widehat{x}(\xi)+\widehat{x}(\xi+\pi)) && (2.67)
\end{aligned}$$

And by interpolating with zero, for reconstruction we have a similar relation. The insertion is defined by:

$$\check{y}[n] = \begin{cases} x[p] & \text{if} \quad n = 2p \\ 0 & \text{otherwise} \end{cases} \qquad (2.68)$$

whose gives us :

$$\widehat{y}(\xi) = \sum_{n=-\infty}^{\infty} x[n]e^{-2in\xi} \qquad (2.69)$$

$$\Rightarrow \quad \widehat{y}(\xi) = \widehat{x}(2\xi) \qquad (2.70)$$

For a first level decomposition, we denote s and d the wavelet decomposition coefficients. We have

$$\widehat{s}(\xi) = \frac{1}{2}\left[\widehat{h}\cdot\widehat{f}\left(\frac{\xi}{2}\right) + \widehat{h}\cdot\widehat{f}\left(\frac{\xi+2\pi}{2}\right)\right] \qquad (2.71)$$

and

$$\widehat{d}(\xi) = \frac{1}{2}\left[\widehat{g}\cdot\widehat{f}\left(\frac{\xi}{2}\right) + \widehat{g}\cdot\widehat{f}\left(\frac{\xi+2\pi}{2}\right)\right] \qquad (2.72)$$

and for the reconstruction we have:

$$\widehat{f}(\xi) = \frac{1}{2}\left[\overline{\widehat{h}}\cdot\widehat{s}(2\xi) + \overline{\widehat{h}}\cdot\widehat{d}(2\xi+\pi)\right] \qquad (2.73)$$

If we do not subsample, we obtain an undecimated wavelets decomposition. It will more computational time as at each level i we have compute $N = 2^p$ points and not 2^{p-i} points for the decimate case. Although this decomposition is not orthonormal anymore but it has the advantage of being grid independent. We now do a recall on the Wavelet packets

2.3.3 Wavelet Packets

The wavelets decomposition is obtained by projection of each subspace V_{j-1} on the direct sum of two orthogonal spaces: $V_{j-1} = V_j \oplus W_j$. Instead of dividing only V_{j-1} we can decide to operate the same division on the details space W_{j-1} to obtain a binary wavelet packet decomposition. We then have a recursive symmetric splitting algorithm as opposed to the wavelet decomposition. For so we have to define $w_{lev,j}$, the filter used for the j^{th} space at the lev^{th} level to project on $W_{lev,j}$. That enables to have $W_{lev-1,j} = W_{lev,2j} \oplus W_{lev,2j+1}$. To obtain the two components, also called children nodes, we convolve our signal either by the scaling or the wavelet functions. We obtain the two wavelet packet orthogonal basis:

$$\psi_{lev}^{2j} = \sum_n h[n] \cdot \psi_{lev-1}^{j}(. - 2^{lev-1}n) \tag{2.74}$$

$$\text{and } \psi_{lev}^{2j+1} = \sum_n g[n] \cdot \psi_{lev-1}^{j}(. - 2^{lev-1}n) \tag{2.75}$$

We denote $m_0, m_1 \in L^2(\mathbb{R})$, in the Fourier domain such that:

$$m_0(\xi) = \widehat{h}(\xi) \quad \text{and} \quad m_1(\xi) = \widehat{g}(\xi), \tag{2.76}$$

and we obtain $w_{lev,j}$ in the Fourier domain such that:

$$\widehat{w}_{lev,j}(\xi) = \prod_{i=0}^{lev-1} m_{\alpha_i}(2^{-i}\xi) \tag{2.77}$$

where α corresponds to the decomposition of j in the dyadic decomposition, $j = \sum_{i=0}^{lev-1} \alpha_i.2^i$ And we have $\forall l \in N \quad \oplus_{j=0}^{2^l-1} W_j^l = V_0$. We represent this tree for the one dimensional case on figure 2.13. This algorithm enables to build a large family of spaces that will be used for our preprocessing.

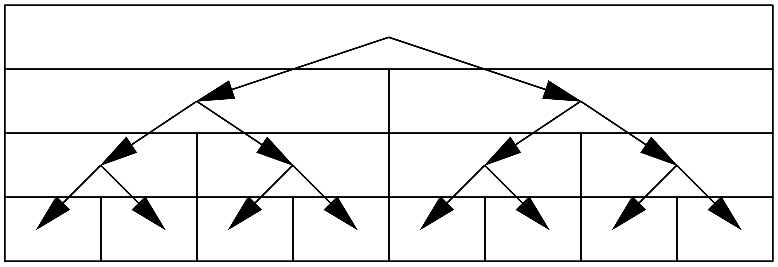


Figure 2.13: *One dimensional Wavelet Packets Decomposition*

2.4 Preprocessing with Undecimated Wavelet Transform

We have defined multi-scale analysis based on the Orthogonal Wavelet Decomposition in the previous chapter. But we want a segmentation to be grid independent and this property cannot be obtained with decimated wavelets representation. Thus we are going to use the undecimated wavelet representation, and more precisely the Wavelet packets. Our preprocessing will be based on this formalization, that will be done at different scales.

2.4.1 Undecimated Wavelet Representation

Let's apply a one level decomposition to a discrete signal $f \in L^2([0, 2^N - 1])$, that we periodize over $\mathbb{R}$. We have $s(i) = \langle f, \varphi(\cdot - i)\rangle$ and $d(i) = \langle f, \phi(\cdot - i)\rangle$. For a decimate decomposition we have:

$$f = \sum_{i=0}^{2^{N-1}-1} \langle f, \varphi(\cdot - 2i)\rangle \cdot \varphi(\cdot - 2i) + \sum_{i=0}^{2^{N-1}-1} \langle f, \phi(\cdot - 2i)\rangle \phi(\cdot - 2i) \tag{2.78}$$

and by translation of the signal we easily obtain:

$$\begin{aligned} f &= \sum_{i=0}^{2^{N-1}-1} \langle f, \varphi\big(\cdot - (2i+2)\big)\rangle \cdot \varphi\big(\cdot - (2i+2)\big) \\ &+ \sum_{i=0}^{2^{N-1}-1} \langle f, \phi\big(\cdot - (2i+2)\big)\rangle \cdot \phi\big(\cdot - (2i+2)\big) \end{aligned} \tag{2.79}$$

And then we can write:

$$f = \frac{1}{2}\Big(\sum_{i=0}^{2^N-1} \langle f, \varphi(\cdot - i)\rangle \cdot \varphi(\cdot - i) + \sum_{i=0}^{2^N-1} \langle f, \phi(\cdot - i)\rangle \phi(\cdot - i)\Big) \tag{2.80}$$

That means in the Fourier domain, we have the following decomposition:

$$\widehat{s}(\xi) = \widehat{h}(\xi).\widehat{f}(\xi) \quad \text{and} \quad \widehat{g}(\xi) = \widehat{g}(\xi) \cdot \widehat{f}(\xi) \tag{2.81}$$

And for the reconstruction:

$$\widehat{f}(\xi) = \frac{1}{2}\left[\overline{\widehat{h}}(\xi) \cdot \widehat{s}(\xi) + \overline{\widehat{g}}(\xi) \cdot \widehat{d}(\xi)\right] \tag{2.82}$$

We are using now the undecimated Wavelet Packets decomposition to be free of any troubles related to the dyadic grid. We have continuity of the decomposition. Of course we do not have anymore a basis, because of the non-orthogonality of the decomposition. We apply the algorithm developed by Roland Guglielmi in [16]. In the case of a three level decomposition, for a one dimensional signal we have shown before on figure 2.13, the representation as an array. In a similar way, we can represent the two-dimensional wavelet packets decomposition as seen on figure 2.14. But as we do not have any decimation in the wavelets and wavelets

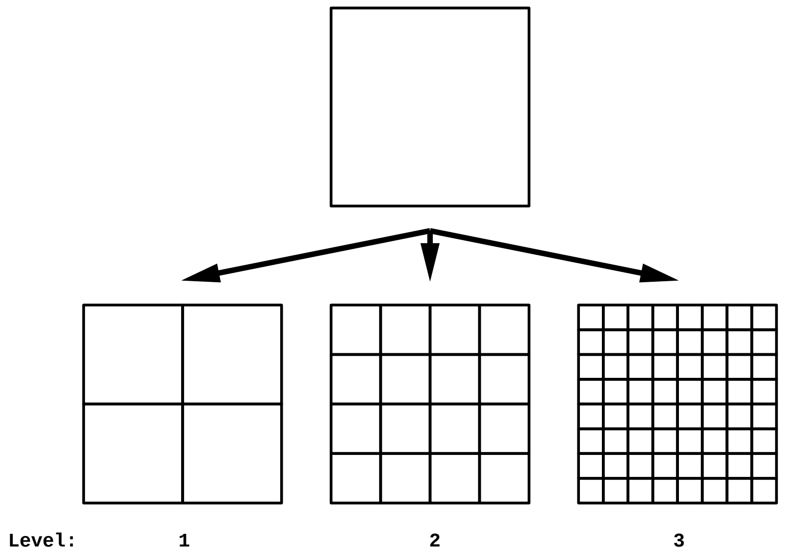


Figure 2.14: *Two dimensional Wavelet Packets Decomposition.*

packets decomposition, we loose the property relatives to basis. And also instead of having always a constant size for the data, we have a growing size of elements, to be more precise in $\mathbb{R}^n$ the data size is multiply by 2^n at each level. These representation has the advantages of creating “a more adapted partition” of the frequential domain than with a wavelets decomposition.

2.4.2 Wavelets Packets Preprocessing

The data set is composed of one and two dimensional signals of different lengths. We will not work directly on the raw data set, however, we will apply the following transformation to the data. For each signal we subtract the low frequencies, to do so, we applied an undecimated wavelet decomposition until level three or four depending on the size of the signals. And we set to zero all the wavelets coefficients but the low frequencies of the last level. We show on figure 2.15 the representation as an array for an one dimensional signal. For a two-dimensional signal, we obtain a comparable representation as seen on figure 2.16.

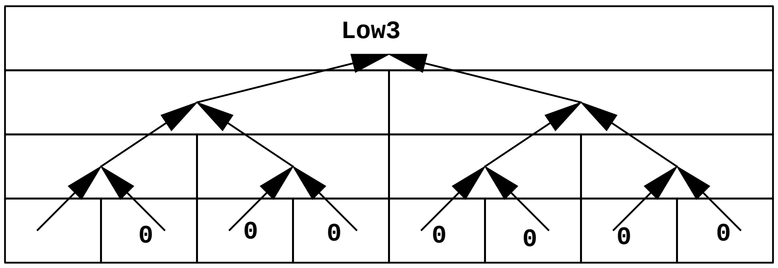


Figure 2.15: *One dimensional Preprocessing using Undecimated Wavelet Representation*

We extract the low frequencies of our signal and we obtain $Signal = Low_i + High_i$, with $Low_i = \boldsymbol{V_i}$ and $High_i = Signal - Low_i$. Therefore the working data set is composed of two signals, and we are going to work first on the high frequencies part. A direct segmentation can be done on the low frequencies components with Mumford-Shah, but we are not interested by this part right now.

Remark 2.4.1. *We work with a one dimensional signal, as seen on figure 2.17, composed of a parabola and three different subcomponents: low frequency sinusoid, a chirp and a high frequency sinusoid. And we add some noise to the whole signal.*

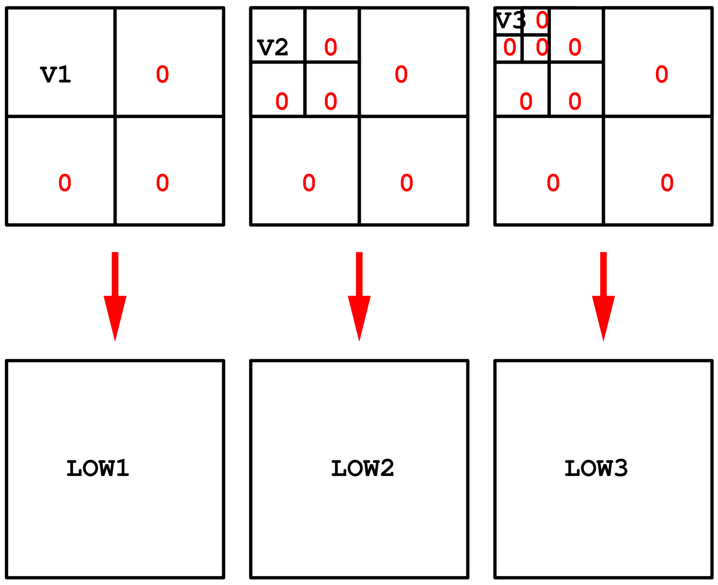


Figure 2.16: *Two dimensional Preprocessing with a Undecimated Wavelet Representation. Three levels of reconstruction are shown.*

When the first step has been applied, we can then start the computation of the "Cost Subspaces".

2.4.3 "Cost Subspaces" computations

We know that a segmentation done directly after wavelets or wavelet packets decomposition does not work. In fact, all the subspaces but the low frequencies have an average equals to zero. The Mumford-Shah algorithm has then a low probability to be efficient. Koepfler, Lopez and Morel [19] solve this trouble by separating the positive and negative part of the decompositions and convolving them. They obtain good results but this method presume a preselection of the filters. We want to have an automatic selection and a post-processing has to be done on the decomposition. We want to apply a cost function on each subspace to be able to determine either the filter can "see the different structures" in the signal or not. And we will call "Cost Subspace" the subspace corresponding to a signal that has the same size has this subspace and gives us locally the efficiency of our filter to "see structures". So for each subspace we are applying the same operations on every pixels. The steps are in

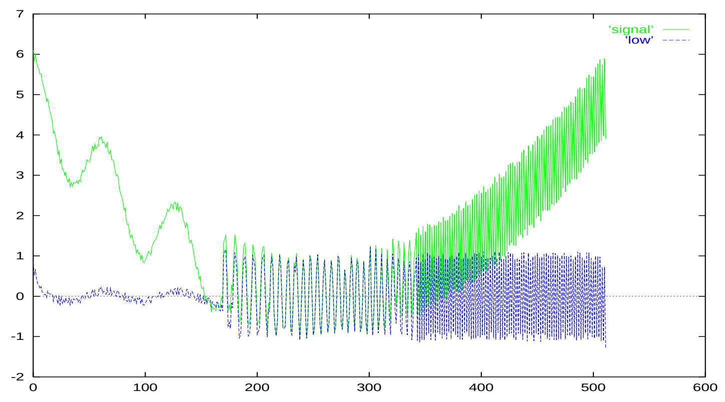


Figure 2.17: Signal and Low components

the one and two-dimensional case:

Step 1 Extraction of neighborhood window for each pixel i or (i, j).

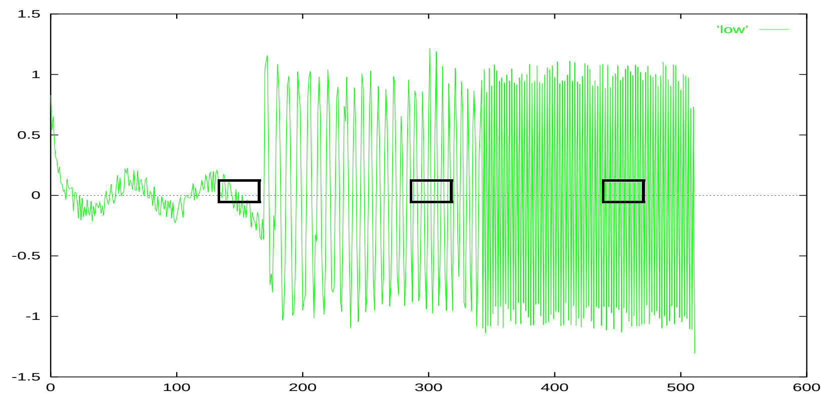
Step 2 Histogram of each window.

Step 3 Computation of the histogram cost and association of the cost to the point i or pixel (i, j).

Extraction of neighborhood windows

We compare now the information contained in different regions of the signal (its size will be denoted M and (M, M) for the one and two-dimensional case) to apply a segmentation. We affect to each pixel a neighborhood of a determined size (N) or (N, N) $(N \ll M)$ respectively for the one or two dimensional case. This size will determine the scale of the segmentation that interests us.

If N is small, it will give a fine segmentation (a high number of regions), else we will have a coarse segmentation (only few regions). We can assimilate N with the segmentation's scale. For a one dimensional case we have an example on figure 2.18. In the two dimensional case an image of size (M, M), and choosing some window of size (N, N), we have the figure 2.19. After extracting the neighborhood window, we compute the histogram and try to select the useful filters.



.

Figure 2.18: *Three windows of size N represented on the signal used for the extraction neighborhood points.*

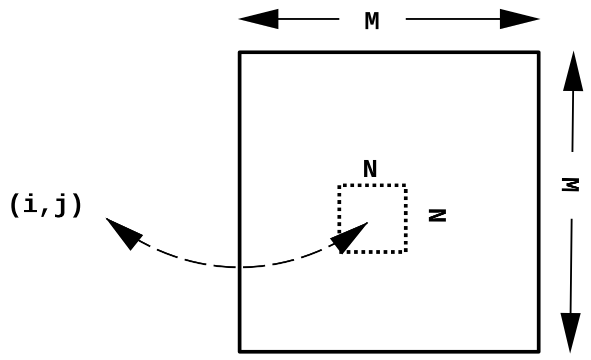


Figure 2.19: *Window Extraction for the pixel* (i, j) *in the two dimensional case*

Histogram of each window

After having extracted a window for each pixel, we sort the values to be able to have an histogram. This histogram is going to enable us to determine if the information contained in the neighborhood pixel is relevant or not. As a matter of fact if there is no information contained in this subspace around this pixel, most of the coefficients are going to be equal to zero. As we want to sort the filter with their ability to discriminate textures, we are looking for some particular histograms. We will have as example three histograms, from a neighborhood window of size equals to 32, taken for three different filters at three different points. We have choose three characteristic points, each of them filtered by a specific filters.

For each point, we sort each neighborhood window obtained after filtering. We

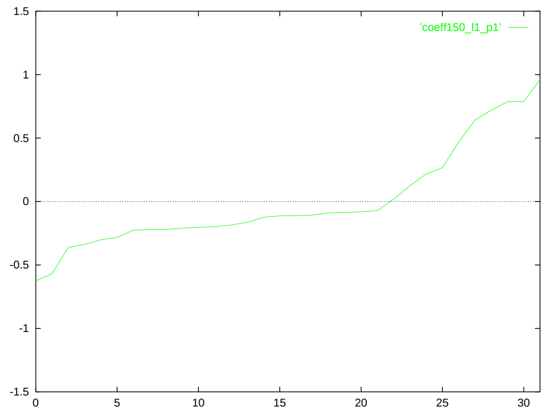


Figure 2.20: (point = 150, level = 1, position = 1), low frequency filter in a neighborhood with low energy, most of the coefficients are concentrated in zero.

obtained array with indices from 0 to 31, sorted by decreasing value, and we obtain the three following graphs on figures 2.20,2.21 and 2.22.

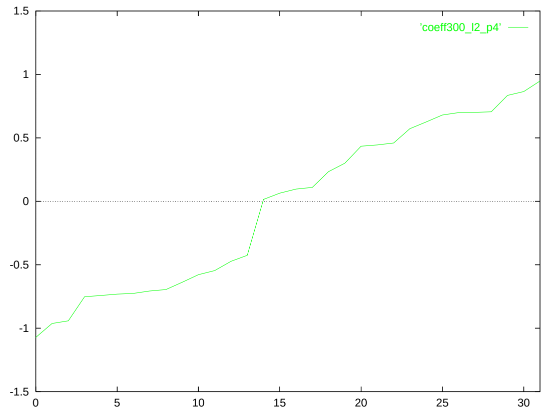


Figure 2.21: point = 300, level = 2, position = 4), high frequency filter in a neighborhood where a chirp is living, we almost have a uniform repartition over $[-1, 1]$.

We can conclude that the first histogram shows us that most of the window's coefficients are concentrated in zero. Our filter is not sensitive to our signal in the neighborhood of this point, the signal is living in a subspace orthogonal to the one spanned by this filter. By contrast, a filter containing information will give us an histogram like the second one. But the best result is obtained when the filtering gives only two modes, as we can see on third histogram. The goal of the next step is to characterize significant histogram, the two

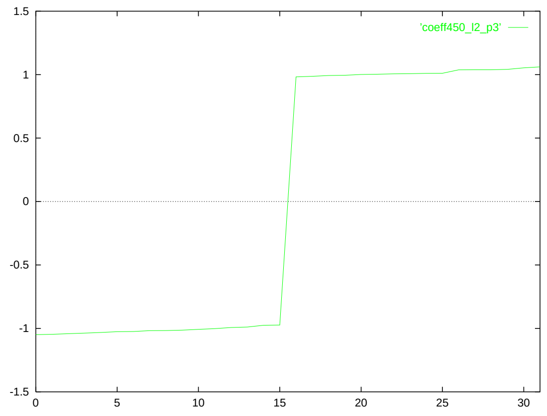


Figure 2.22: (point = 450, level = 2, position = 3), high frequency filter in a neighborhood where high frequencies are in higher density than the others frequencies, concentration of the coefficients in two modes.

last one as opposed to the first one. We show now how to compute some cost functions to determine how well a filter "can see structures".

Cost Computation

We have an histogram and we want to determine its significance. A first and simple cost function is the energy function $\sum_{i=1}^{N} {x_i}^2$. An histogram concentrated in zero will give us a low energy by opposition to one composed of two modes far from zero. This cost function has the conservation property. If we sum all the subspaces from a same level, we have for each pixel equality of the energy contained in the level zero. An example on figure 2.23 shows this property. We can see how the cost function evolves. On the first third of the signal, there is almost no energy, because in the preprocessing we had extracted the low frequencies, so the two signals are almost equal to zero. On the second part we have a chirp, so the frequencies domains is contained in the low and high frequencies domains. Both signals are different from zero. The last part containing mainly high frequencies, only one of the two signals (high pass) is obviously different from zero. We loose the energy conservation property but we will see later the advantage of this cost function. We tried also to compute different cost functions using the density probability over the window.

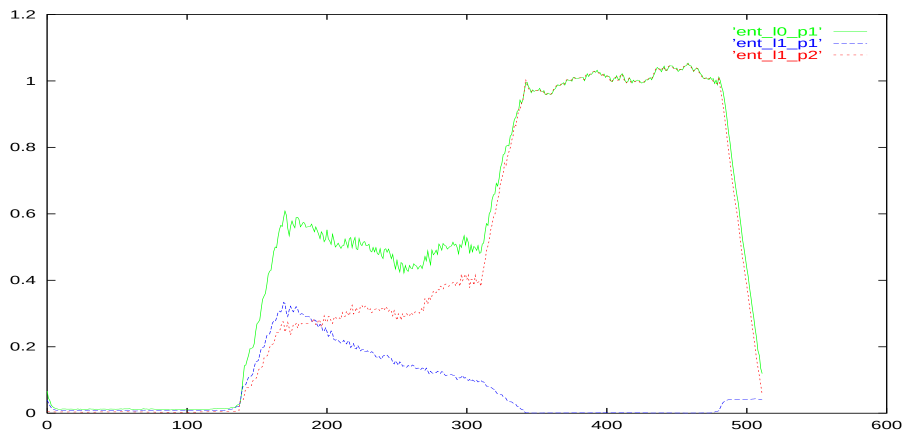


Figure 2.23: *Local Energy at the first level (low and high pass).*

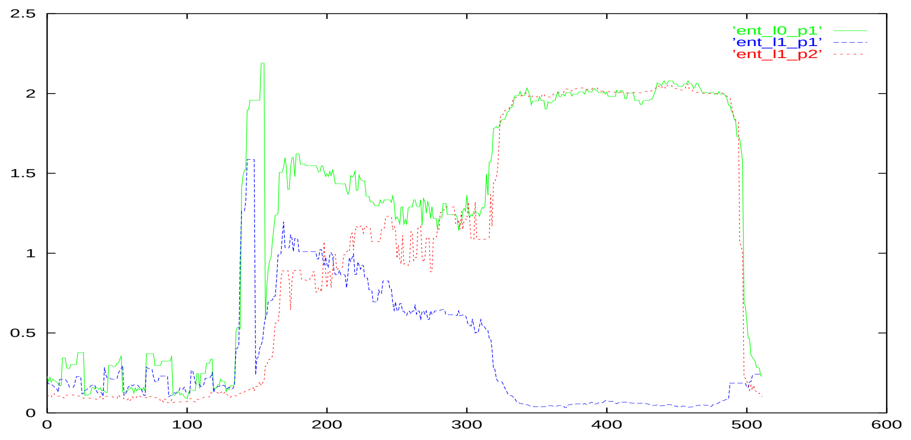


Figure 2.24: *Local Cost at the first level (low and high pass).*

Another possibility is to determine the distance between the two modes when they exist. If the two modes exist, different from zeros, we assume that each of them contains at least a quarter of the data contained in the histogram, if not it means that these two modes are negligible. We are affecting to this histogram the distance between the first quarter and the last quarter. An histogram with concentration in zero will give us a cost almost equal to zero compared to an histogram composed of two separated modes. We have such cost function:

$$Cost(x_1, \ldots, x_n) = sort(x)_{\frac{3n}{4}} - sort(x)_{\frac{n}{4}} \tag{2.83}$$

where $sort(x)$ is the vector containing the x's sorted values (non-decreasing sort). So for the three previous examples, seen in chapter 2.4.3. We note Δ the value defined in equation 2.83, we have $N = 32$ then $\Delta = |x_{23} - x_7|$ (as our array is sorted by decreasing value from 0 to 31). The value of Δ for the examples is then almost equal to zero, equals to one and bigger than one. And it corresponds to the fact that the first filter is not as adapted and does not see as much the structures as the two others one.

2.5 Some Options

We present in this section different options to select the efficient filters and a way to shorten the computational time for segmentation. We look at different entropies of segmented images to determinate if a segmentation is reasonable based on our criterion. Then we select the components of our vector image that carry information. Furthermore, we shorten the computation time. We finish this chapter by summarizing our algorithm on a single figure.

2.5.1 Best Filter Choice and Reduction of Dimension

For each subspace, we computed a "Cost Subspace" according to a cost function. From these new subspaces, we have to decide which one are the most adapted for segmentation, our criterion is to have no small regions and not too many. We will put a part the non significant. Therefore we apply a global cost function on each "Cost Subspace". We compute the entropy of this subspace by using the following formula:

$$Ent(X) = -\sum_{i=1}^{q} p_i \cdot log_2(p_i) \tag{2.84}$$

where $(p)_{i=1}^{q}$ is the probability density function of the whole subspace, the interval of value range is divided in q regions. We compute the ratio of the $\|.\|_1$ and $\|.\|_2$ that gives us information about the coefficient repartition. These two global functions have the drawback to not give any spatial informations. In fact, they only give informations about the values repartition. Therefore we do not know when a filter is efficient for segmentation. An ideal cost function may have to be global to make sense. The trouble that we have with the Mumford-Shah Functional is that it does not make any sense to look at its value for a fixed number of regions. Henceforth we have:

$$\forall (N_1, N_2) \in N^2, \quad N_1 < N_2 \quad \Rightarrow \quad E_{N_1}(u) > E_{N_2}(u) \tag{2.85}$$

with $E_N(u)$ equals to the cost of a segmentation in N regions, approximate by a piecewise constant function. We have seen that an extra term can be useful. We choose another criterion to establish when a filtered image is useful or not for our segmentation. In the sense that we are interested in finding regions of equivalent sizes and not a segmentation with small regions, that can be related to noisy regions or targets. We compute a cost function based on the entropy of the renormalized areas A_i (such that the measure of the image I is one) of the different segmented regions i:

$$Ent(I) = -\sum_{i=1}^{N} A_i \cdot log_2(A_i) \text{ with } \sum_{i=1}^{N} A_i = 1 \text{ and } A_i > 0. \tag{2.86}$$

We observe, in different experiments, that there is a region i_0 that is much larger than the other regions. We assimilate it to a background in many cases. We consider now two cases. For a given image, a part of "Barbara", that we have filtered in two different ways, We obtain the two segmentations, in eleven regions, shown on figure 2.25. We consider for

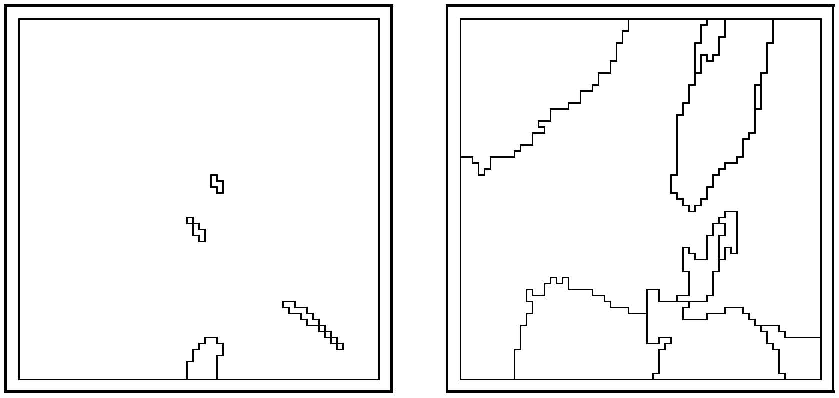

.

Figure 2.25: *Two segmentations obtained from different filters applied to "Barbara". Obviously the right segmentation gives, with our criterion, a better result than the left one with its very small regions (related to small regions with high contrast like noise or targets).*

our study that a segmentation is viable when the regions correspond to the average scale. We mean that there are no small regions, and that they all have similar sizes. Since small regions, for this algorithm, are synonym of noise or targets, point or small region with a high contrast, as they can be seen as targets on military images. Thus we prefer the right image to the left one. We now try to explain how to select them.

For the left image on figure 2.25, we model an image with one main region and N small regions of area A fixed (close to the pixel size), such that $1 >> N \cdot A$, and a main region i_0 with $A_{i_0} = 1 - N \cdot A$. We have the following entropies function:

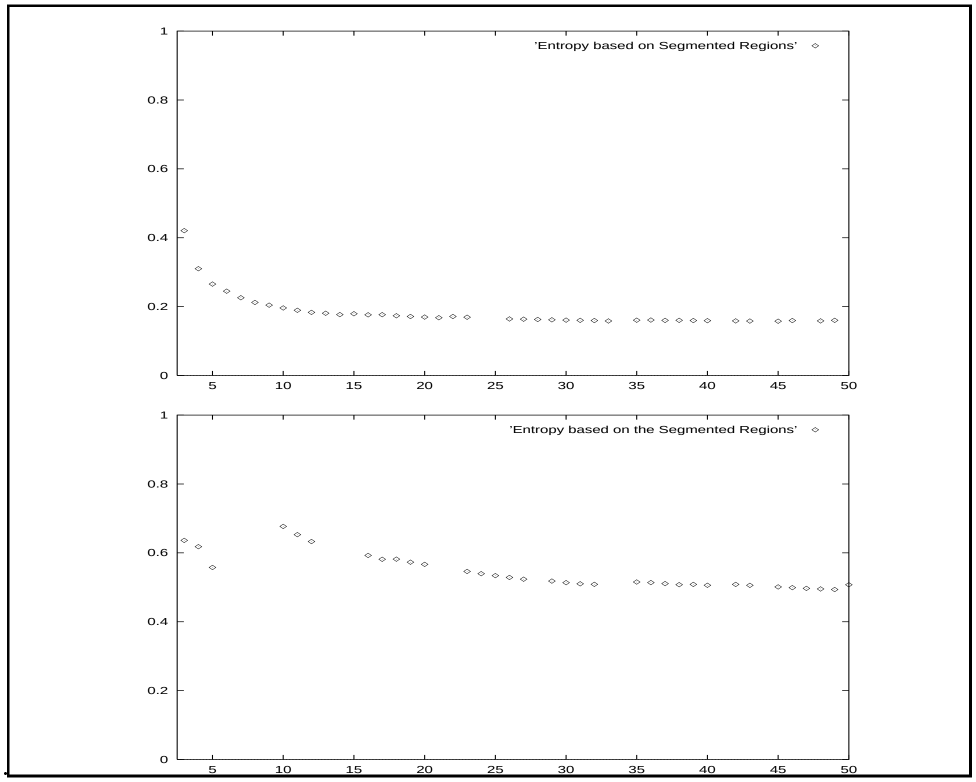


Figure 2.26: *Graph of Plot of the Entropies as a function of the areas segmented, renormalized to one for the optimal segmentation in N regions of same size. The top graph corresponds to segmentations with many small regions, segmented on the left image of figure 2.25, as opposed to the bottom case that is obtained for a segmentation with "average size regions", right image of the figure 2.25. The second case is the one that interests us as the entropy level is higher and there is a discontinuity in the decrease for the number of regions equals to 9, that corresponds to the "optimal number of regions" for the right image.*

$$Ent(I) = -(1 - N \cdot A) \cdot \log(1 - N \cdot A) - N \cdot A \cdot \log(A) \tag{2.87}$$

Thus after simplifications, we have:

$$Ent(I) \sim -N \cdot A \cdot \log(A), \text{ with } A \text{ fixed.} \tag{2.88}$$

We conclude that $N \mapsto Ent(I)$ is linear for value of N such that $1 >> N \cdot A$, with A fixed corresponding a small region close to the pixel size. This example is shown on figure 2.26

(top graph). This possible segmentation does not interest us as we are looking for regions with intermediate sizes. In the case of the right image in figure 2.25, we model our image as N regions of sizes A_i similar to A. Thus we have $N \cdot A = 1$. After obvious simplifications, we have the following formula:

$$Ent(I) = \log(N) \tag{2.89}$$

We have an experimental result corresponding to this case on figure 2.26, bottom graph. The optimal number of regions found for this case is nine, as we have change in the decrease of the entropy for this particular value. Then the second filter will have a degree of significance higher than the first one. Hence, we sort the subspaces, obtained after filtering, based on this criterion and the Mumford-Shah with Extra term. We defined a way to choose the best filters and reduce the dimensionality. We will apply the Mumford-Shah segmentation to a few number of subspaces. We now present a sub-sampling that enables to shorten computation time before summarizing the global algorithm on figure 2.28.

2.5.2 Optional Speed Improvement

We want to improve the speed of this algorithm by reducing the “cost subspace” computations. In fact, we subsample our “cost subspaces” and thus shorten computational time. The “cost subspaces” are smaller than before, and will enable more and higher levels for decomposition as they reduce the computational costs. The sub-sampling is shown on figure 2.27 The sub-sampling is possible as the cost subspace are obtained by using window of size at least $(8, 8)$. Their scale is then greater than the pixel size, a ratio of two or four will not change the result. In a first step, we compute the “cost subspaces” by this fast algorithm. Then we apply our criterion to decide if the filtering is good or not, we assimilate this as a preprocessing to reduce dimensionality problem. In the second step we compute only the interesting “cost subspaces” at their full size and then apply the vectorial segmentation on them.

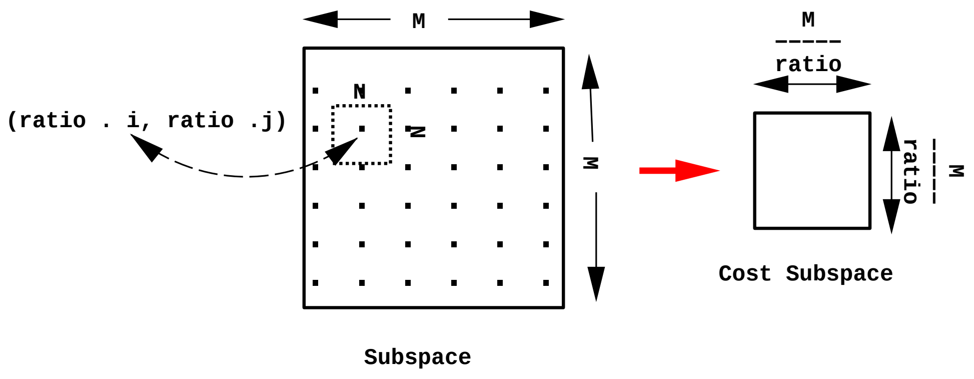


Figure 2.27: *Subsample "Cost Subspace" with a ratio to improve speed. , and a neighborhood window of size* (N, N) *with generally* $N = 8$ *or* 16. *A ratio of* 4 *is then acceptable*

2.5.3 Segmentation Summary

We present on figure 2.28 the main steps of our segmentation algorithm:

- Wavelet Packets Decomposition
- "Cost Subspaces" Computation
- Sorting of the Significant Subspaces
- Vectorial Segmentation

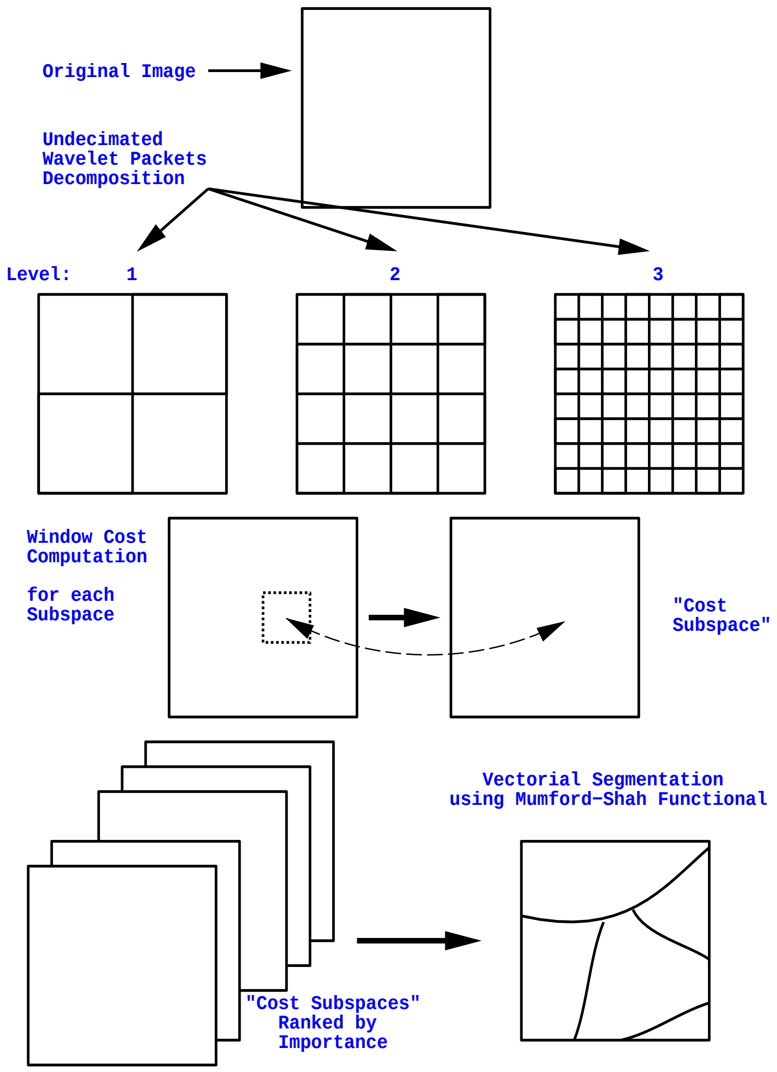

Original Image
Undecimated Wavelet Packets Decomposition
Level: 1
2
3
Window Cost Computation
for each Subspace
"Cost Subspace"
Vectorial Segmentation using Mumford–Shah Functional
"Cost Subspaces" Ranked by Importance

Figure 2.28: *The different steps of our segmentation algorithm, seen in the previous chapters.*

2.6 Results for two-dimensional segmentation

We present the different versions of this algorithm that we tried. We first worked with man-made images that are easily segmented in the frequencies space, thus a wavelets preprocessing is well adapted. The first image is a two-dimensional signals composed of one dimensional sine products. After, we rotate our image by forty five degrees to have a real two dimensional problem. And we started to be confronted with artifacts, due to periodicity problems at the boundaries. We solved them by using a mask (like a crown). The next chapters also show the different results on real images, some troubles and the way we solved them.

2.6.1 Wave image

Our first trial is on an image composed of four regions, each region corresponding to a quarter. Each one is composed of sine products. We choose an easy image to start, since it is almost having to segment a one dimensional signal. Moreover each region has a specific local frequency and a wavelet representation is obviously well adapted. The original image and its segmentation are shown in figure 2.29. We notice that the algorithm gives a good

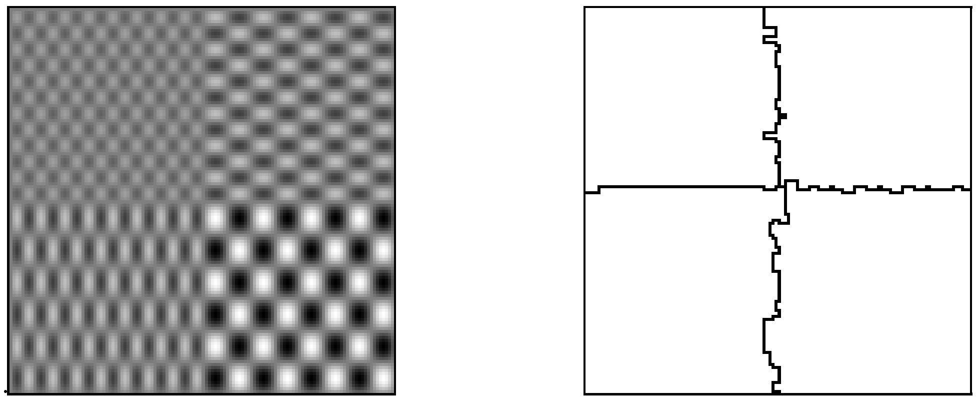
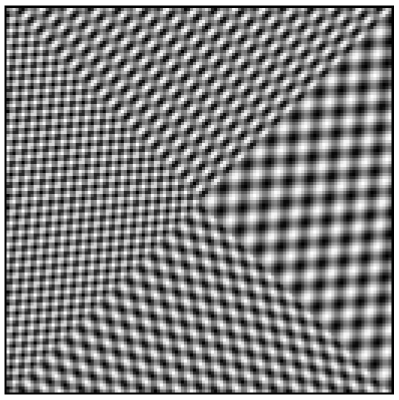


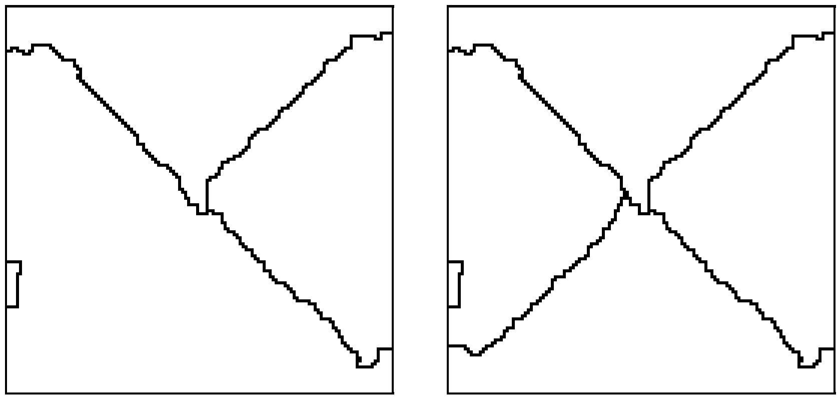
Figure 2.29: *The original image* (128, 128) *and its segmentation in four regions obtained after wavelets preprocessing. We obtain the desired segmentation in four regions.*

result but as it corresponds to an easy one-dimensional problem. In the next chapter, we will rotate our image by an angle of forty five degrees to obtain a true two-dimensional problem.

2.6.2 Wave image rotated

We rotated of forty five degrees the previous image, shown on figure 2.29 to have a real two dimensional problem. The new image is shown on figure 2.30 with two possible segmentations. We can see that we have artifacts close to the boundaries, there are related to the





.

Figure 2.30: *The original image* (128, 128)*, obtained by rotation of* 45 *degrees of the image shown on figure 2.29, its segmentation in four and five regions.*

non-periodicity of the image. This trouble is due to the way we compute the wavelet packets decomposition. Since we assume that the image is periodic, some artifacts will appear when we compute the neighborhood window at the side of the images. We will now segment the window included in a crown, since we decide to set to zero the pixels closer than $(\frac{N}{2})$ to the boundaries where N is the size of the neighborhood window (for the cost computation as defined in chapter 2.4.3) for each "cost subspace". On figure 2.31, we obtain a better result for the same image, by masking with a crown. We force the algorithm segment separately the region where the points closer to $N/2$ pixels from the boundaries. The problem related to the non-periodicity of the image has then disappeared. We observe that the result is

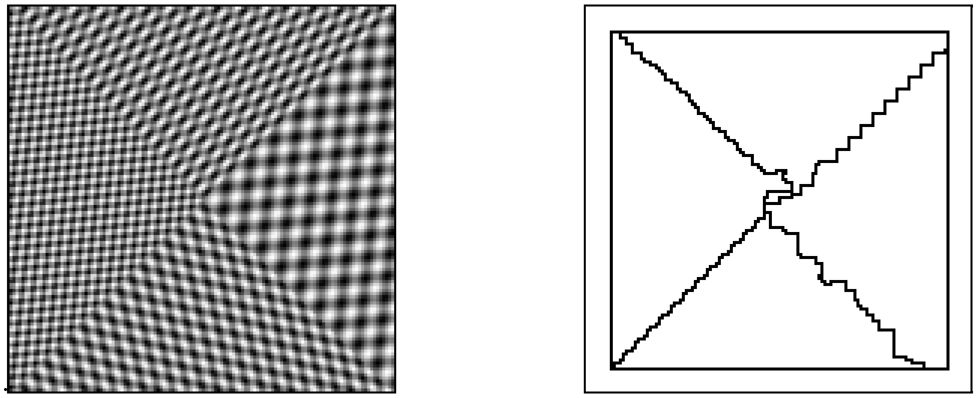


Figure 2.31: *Original Image (*$128, 128$*), its segmentation in four regions and a crown*

significantly better, as it corresponds now to what we are looking for, and the artifacts disappear.

2.6.3 A simple small real image

We extract a sub-image of "Barbara", seen on figure 2.32. It contains two major structures:

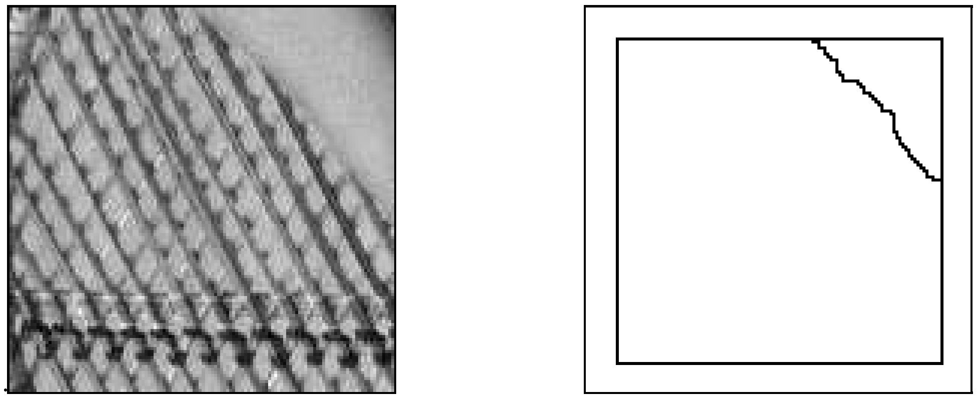
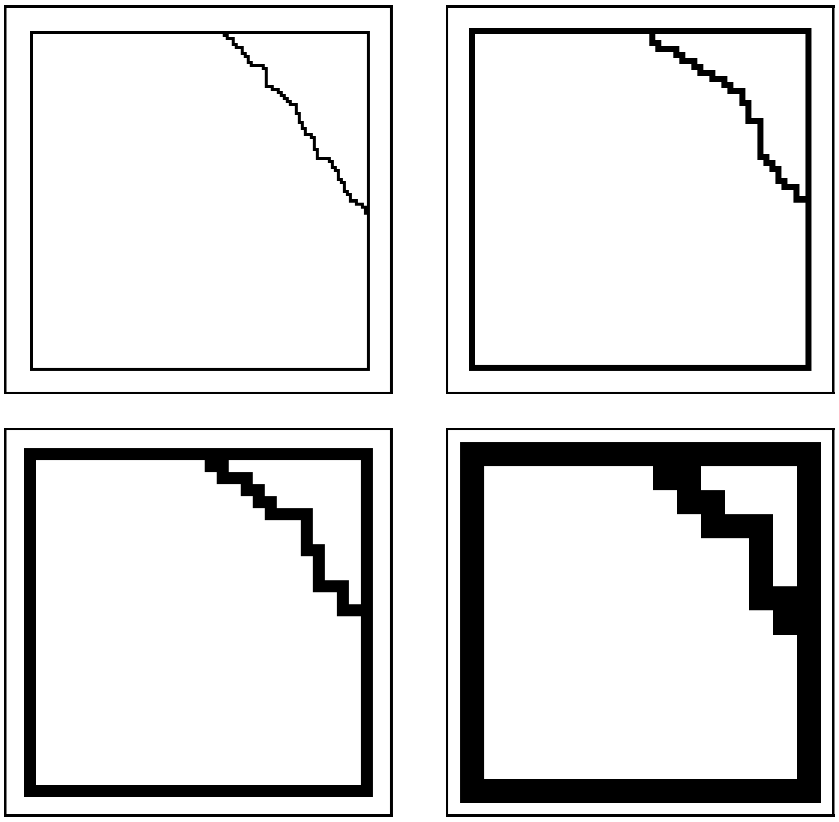


Figure 2.32: *Original Image* $(128, 128)$*, its segmentation in two regions and a crown*

a part of the arm and the back of the chair. One part has almost a constant value and the other part is composed of high frequencies, due to the stripes on the chair. We have a good result in this case. We lost as before the information in the boundaries neighborhood but the result corresponds to our expectations. We observe that the computations time can be important if the image is large. Thus we will use the improvement defined in chapter 2.5.2 to shorten computation time.

2.6.4 Real images with "speed improvement"

We work on a sub-image of "Barbara", shown on figure 2.32, but this time we divide our subspace by different ratio: 1, 2, 4, 8. We divide "cost subspaces" time computations by



.

Figure 2.33: *Results of the segmentations for four different ratios:* $1, 2, 4$ *and* 8*. They all four corresponds to the case where we have two regions and a crown for the image on figure 2.32*

the square of the corresponding ratio. Of course we loose precision in the segmentation, but we can see on the following results that we still have efficiency of our algorithm. For an easier comparison, we have decided to represent them at the same scale by using a zoom factor equal to the ratio used.

In a first step, this ratio is used for a faster selection of the significant "cost subspaces". And after we have selected the interesting filter using the Mumford-Shah and extra term, we then run the algorithm on the full size image.

2.6.5 Real images

We ran our algorithm on a part of Barbara image. The preprocessing obtained from the

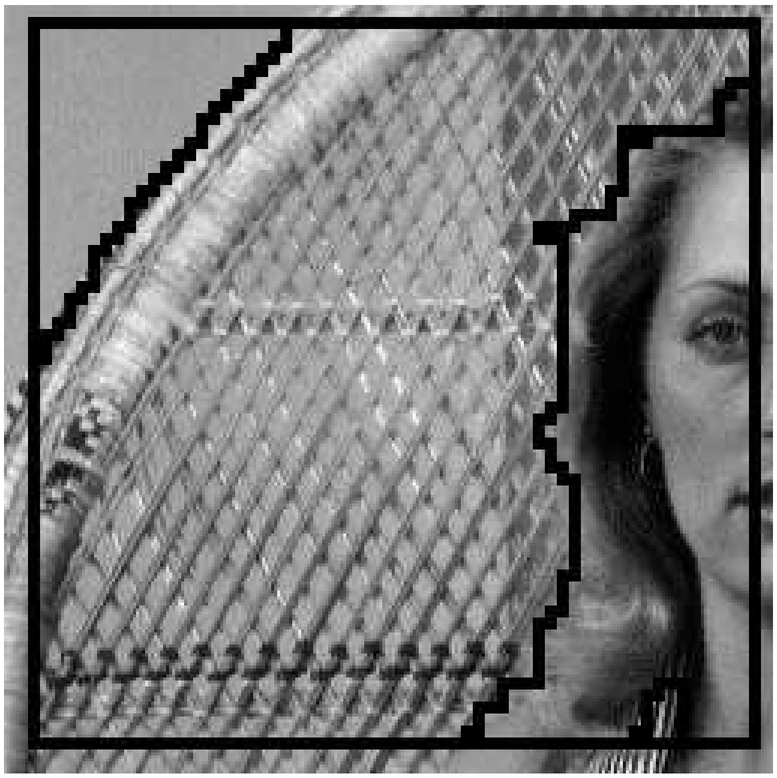


Figure 2.34: *A part of Barbara image (*$256, 256$*) segmented in four regions and a crown. A ratio of four has been used for speed improvement.*

wavelet packets analysis and cost function ranks the filter depending on their pratical worth for the considered image. In this case, we segment our image based on the information given by the best filter. We obtain result on figure 2.34 with a ratio of four for speed improvement. We can conclude that the choice of the utility of filter has been correctly found based on our criterion of segmented region sizes.

2.7 Conclusion

We have shown properties of the pyramidal algorithm designed by J.-M. Morel and his collaborators for the Mumford-Shah functional and how an extra term can be added to find an optimal number of regions for a given image. We have a preprocessing based on the undecimated wavelet transform to build a vector image. We create a cost function that should correspond to our criterion of having a segmentation with regions of similar sizes, as opposed to segmentation with very small regions corresponding to targets or noise. We can then decide which components are the most important and use them for our vectorial segmentation with the pyramidal algorithm. The algorithm is modular and thus some of the steps can be changed as the algorithm used for the segmentation in itself. Some refinement can be certainly done in the choice of the cost function or the preprocessing, as using the two-dimensional extension of the Blaschke product.

Chapter 3

Undecimated Wavelets and Applications

3.1 Introduction

In this paper, we propose to improve some traditional methods for image and signal processing using the wavelets representation. In [28], Mallat presents new ideas on multi-scale analysis, providing an approach to work and see the structures at various scales. As opposed to other methods that only work at the pixel level, this representation enables an orthogonal projection on various subspaces at different scales. In [26], the wavelet packets are introduced by Coifman to enable a better representation using a binary wavelet packet tree. The decimated wavelets can be computed with a fast algorithm but they have the objectionable feature of being grid dependent. To solve this trouble, we use the algorithms developed by Guglielmi in [27]. We will preprocess our data using these different tools.

Our first application is deconvolution, smoothing and sharpening one and two-dimensional signals. By choosing various subspaces of the signal, one can obtain similar results on the signal to a sound equalizer, without artifacts. Each component can be modified separately. We apply this process to one-dimensional chirps (since frequential and spatial informations are linked, it is easier to observe the frequential effect of the algorithm), two-dimensional man-made signals, deconvolution of one-dimensional signal before showing how to remove the stripes of Barbara's vest without erasing the shadows of her arms.

The second application is the denoising for radar images. There are some difficulties since these images have a high contrast due to the targets and also thre are variations in the background (structures at various scales). We extract the background with a multi-resolution approach, the targets are extracted with a threshold and a "γ-correction" is applied to enhance the contrast before denoising, with a multipass algorithm as implemented by L. Woog in [30]. This algorithm has been inserted in a platform.

The third application is within the medical field, we work on brain images obtained with the functional-MRI. We have two datasets corresponding to an active task and a baseline. We are interesting in the regions of the brain that have a distinct activation in the two datasets. We do not only mean a variation of the average value but much more two distinct probability density functions. The relative entropy is computed at various scales to determinate the dissimilarities between the two states. After reconstruction, we then have for each scale the level of activation at every points.

3.2 Continuous Wavelets Recapitulatory

In this section, we present some of the main properties and theorems about the wavelets decomposition. We follow Mallat [29] in his study, show the difference between undecimated and decimated wavelets. We introduce also the concept of wavelet packets decomposition first developed by Coifman in [26].

3.2.1 Theorems and Properties on Wavelets

Theorem 3.2.1. *In the case of a wavelet multi-resolution approximation, we have a sequence of closed subspaces $\{V_j\}_{j\in\mathbb{Z}} \subset L^2(\mathbb{R})$, that verifies the following properties*

$$\begin{aligned}
\forall (j,k)\in\mathbb{Z}^2,\quad f(\cdot) &\in V_j \Leftrightarrow f(\cdot - 2^j k) \in V_j && (3.1)\\
\forall j\in\mathbb{Z},\quad f(\cdot) &\in V_j \Leftrightarrow f(\frac{\cdot}{2}) \in V_{j+1} && (3.2)\\
\dots V_2 \subset V_1 \subset V_0 &\subset V_{-1} \subset V_{-2}\dots && (3.3)\\
\lim_{j\to-\infty} V_j &= \overline{\cup_{j\in Z} V_j} = L^2(\mathbb{R}) && (3.4)\\
\lim_{j\to+\infty} V_j &= \cap_{j\in Z} V_j = \{0\} && (3.5)
\end{aligned}$$

And we know that there exists θ such that $\{\theta(.-n)\}_{n\in Z}$ is a Riesz basis of V_0. For example in the Haar case, it corresponds to the piecewise constant approximation. We have $\theta = \chi_{[0,1]}$. And so V_j represents the functions f such that f is constant on $[k\cdot 2^j, (k+1)\cdot 2^{(j+1)})$.

Theorem 3.2.2. *Let $\{V_j\}_{j\in\mathbb{Z}}$ be a multi-resolution approximation and ϕ a scaling function such that:*

$$\widehat{\phi}(\xi) = \frac{\widehat{\theta}(\xi)}{\left(\sum_{-\infty}^{+\infty} |\widehat{\theta}(\xi + 2k\pi)|^2\right)^{\frac{1}{2}}} \tag{3.6}$$

And we note

$$\phi_{j,n}(\cdot) = \frac{1}{2^j}\phi\left(\frac{\cdot - 2^j n}{2^j}\right) \tag{3.7}$$

For all $j\in\mathbb{Z}$, $\{\phi_{j,n}\}_{n\in Z}$ is an orthonormal basis of V_j

Remark 3.2.1. *We define an approximation, at a level j, over V_j by using an expansion in the scaling orthogonal basis:*

$$P_{V_j} f = \sum_{-\infty}^{+\infty} \langle \phi_{j,n}, f \rangle \phi_{j,n} \tag{3.8}$$

We can easily see that $\langle \phi_{j,n}, f \rangle = f * \phi_j(2^j n)$, where $*$ represents the convolution.

Theorem 3.2.3. *(Mallat, Meyer)*

Let $\phi \in L^2(\mathbb{R})$ be an integrable scaling function. The Fourier series of $h[n] = \langle \frac{1}{\sqrt{2}} \phi(\frac{\cdot}{2}, \phi(\cdot - n) \rangle$ satisfies:

$$\forall \xi \in \mathbb{R}, \quad |\widehat{h}(\xi)|^2 + |\widehat{h}(\xi + \pi)|^2 = 2 \tag{3.9}$$

and

$$|\widehat{h}(0)|^2 = 2. \tag{3.10}$$

Conversely, if $\widehat{h}$ is a 2π-periodic and continuously differentiable in a neighborhood of zero, if it satisfies the two precedent properties and if

$$\inf_{\xi \in [-\frac{\pi}{2}, \frac{\pi}{2}]} |\widehat{h}(\xi)| > 0 \tag{3.11}$$

and

$$\widehat{\phi}(\xi) = \prod_{p=1}^{+\infty} \frac{\widehat{h}(2^{-p}\xi)}{\sqrt{2}} \tag{3.12}$$

is the Fourier transform of a scaling function $\phi \in L^2(\mathbb{R})$.

Remark 3.2.2. *For piecewise constant approximations, $\phi = \chi_{[0,1]}$. Since $h[n] = \langle \frac{1}{\sqrt{2}} \phi(\frac{\cdot}{2}), \phi(\cdot - n) \rangle$ it follows that*

$$h[n] = \begin{cases} \frac{1}{\sqrt{2}} & \textit{if} \quad n = 0, 1 \\ 0 & \textit{otherwise} \end{cases} \tag{3.13}$$

With the approximations of f, at the scales 2^j and 2^{j-1}, that are equal to their orthogonal projections on V_j and V_{j-1}. We know that $V_j \subset V_{j-1}$. Let W_j be the orthogonal complement of V_j in V_{j-1}, i.e.

$$V_{j-1} = V_j \oplus W_j. \tag{3.14}$$

So the orthogonal projection of f on V_{j-1} can be decomposed as the sum of two orthogonal projections:

$$P_{V_{j-1}} f = P_{V_j} f + P_{W_j} f. \tag{3.15}$$

$P_{W_j} f$ provides the "details" of f that appear at the scale 2^{j-1} but which disappear at the coarser scale 2^j. Furthermore from the previous relations we can easily show that $\oplus_{j \in \mathbb{Z}} W_j = L^2(\mathbb{R})$.

Theorem 3.2.4. *(Mallat, Meyer):*

Let ϕ be a scaling function and h the corresponding conjugate mirror filter. Let φ be the function whose Fourier transform is:

$$\widehat{\varphi}(\xi) = \frac{1}{\sqrt{2}} \widehat{g}\left(\frac{\xi}{2}\right) \widehat{\varphi}\left(\frac{\xi}{2}\right), \tag{3.16}$$

with

$$\widehat{g}(\xi) = e^{-i\xi} \overline{\widehat{h}}(\xi + \pi), \tag{3.17}$$

And we note

$$\varphi_{j,n}(\cdot) = \frac{1}{2^j} \varphi\left(\frac{\cdot - 2^j n}{2^j}\right) \tag{3.18}$$

for any scale 2^j, $\{\varphi_{j,n}\}_{n \in Z}$ is an orthonormal basis of W_j. For all scales, $\{\varphi_{j,n}\}_{j,n \in Z^2}$ is an orthonormal basis of $L^2(\mathbb{R})$.

We describe now a fast algorithm to compute the wavelets decomposition.

3.2.2 Fast Algorithm

We are going to describe a fast filter bank algorithm designed by S. Mallat. This algorithm computes the orthogonal wavelet coefficients of a discrete signal $(a_0[n])_n$. That corresponds to a decimate wavelets decomposition. Let us define:

$$f = \sum_{n=-\infty}^{+\infty} a_0[n]\phi(. - n) \in V_0. \tag{3.19}$$

Since $\{\phi(. - n)\}_{n\in Z}$ is orthonormal, we have:

$$a_0[n] = \langle f(.), \phi(. - n)\rangle \tag{3.20}$$

Each $a_0[n]$ is thus a weighted average of f in the neighborhood of n. The discrete wavelet coefficients of a_0 are defined to be the wavelet coefficients of f:

$$d_j[n] = \langle f, \varphi_{j,n}\rangle \tag{3.21}$$

And we denote $\bar{x}[n] = x[-n]$ and

$$\check{x}[n] = \begin{cases} x[p] & \text{if} \quad n = 2p \\ 0 & \text{otherwise} \end{cases} \tag{3.22}$$

So the following theorem shows how to compute the wavelet decomposition and reconstruction with discrete convolutions.

Theorem 3.2.5. *(Mallat) For the decomposition we have:*

$$a_{j+1}[p] = \sum_{n=-\infty}^{+\infty} a_j[n]h[n - 2p] = a_j * \bar{h}[2p], \tag{3.23}$$

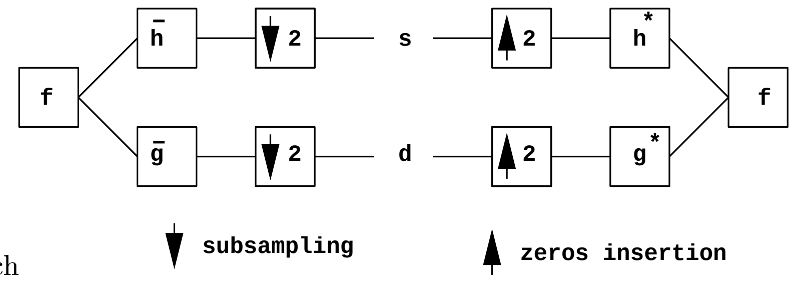
and

$$d_{j+1}[p] = \sum_{n=-\infty}^{+\infty} a_j[n]g[n - 2p] = a_j * \bar{g}[2p]. \tag{3.24}$$

Concerning the reconstruction we have:

$$\begin{aligned} a_j &= \sum_{n=-\infty}^{+\infty} a_{j+1} \cdot h[. - n] + \sum_{n=-\infty}^{+\infty} d_{j+1} \cdot g[. - n] \\ &= \check{a}_{j+1} * h + \check{d}_{j+1} * g \end{aligned} \tag{3.25}$$

The perfect decomposition is ensured by the next theorem.



chch

Figure 3.1: *A signal f is filtered by a low-pass and high-pass filter respectively to give s and d after sub-sampling, while an insertion of zero with dual filters reconstruct f*

Theorem 3.2.6. *(Vetterli)*

The filter bank performs an exact reconstruction for any input signal if and only if :

$$\overline{\widehat{h}}(\xi+\pi)\cdot\widehat{\tilde{h}}(\xi)+\overline{\widehat{g}}(\xi+\pi)\cdot\widehat{g}(\xi)=0 \tag{3.26}$$

and

$$\overline{\widehat{h}}(\xi)\cdot\widehat{\tilde{h}}(\xi)+\overline{\widehat{g}}(\xi)\cdot\widehat{g}(\xi)=2 \tag{3.27}$$

We obtain a perfect decomposition reconstruction by using convolutions and decimations as we can see on the figure 3.1.

By sub-sampling, we modify the relations between our wavelet decomposition coefficients and the original signal. Since we have $\widehat{x}(\xi)=\sum_{n=-\infty}^{\infty}x[n]e^{-in}$, the Fourier series of the subsampled signal, $y[n]=x[2n]$, is going to be such that:

$$\begin{aligned}
\widehat{y}(\xi) &= \sum_{n=-\infty}^{\infty} x[2n]e^{-in\xi}\\
\Rightarrow\quad \widehat{y}(2\xi) &= \sum_{n=-\infty}^{\infty} x[2n]e^{-2in\xi}\\
\Rightarrow\quad \widehat{y}(2\xi) &= \sum_{n=-\infty}^{\infty} (\frac{1+(-1)^n}{2})x[n]e^{-in\xi}\\
\Rightarrow\quad \widehat{y}(2\xi) &= \frac{1}{2}(\widehat{x}(\xi)+\widehat{x}(\xi+\pi)) \qquad (3.28)
\end{aligned}$$

And by interpolating with zero, for reconstruction we have a similar relation. The insertion is defined by:

$$\check{y}[n] = \begin{cases} x[p] & \text{if} \quad n = 2p \\ 0 & \text{otherwise} \end{cases} \tag{3.29}$$

whose gives us :

$$\begin{aligned} \widehat{y}(\xi) &= \sum_{n=-\infty}^{\infty} x[p] e^{-2ip\xi} \\ \Rightarrow \quad \widehat{y}(\xi) &= \widehat{x}(2\xi) \end{aligned} \tag{3.30}$$

For a first level decomposition, we denote s and d the wavelet decomposition coefficients. We obtain

$$\widehat{s}(\xi) = \frac{1}{2}\left[\widehat{h} \cdot \widehat{f}\left(\frac{\xi}{2}\right) + \widehat{h} \cdot \widehat{f}\left(\frac{\xi + 2\pi}{2}\right)\right] \tag{3.31}$$

and

$$\widehat{d}(\xi) = \frac{1}{2}\left[\widehat{g} \cdot \widehat{f}\left(\frac{\xi}{2}\right) + \widehat{g} \cdot \widehat{f}\left(\frac{\xi + 2\pi}{2}\right)\right] \tag{3.32}$$

and for the reconstruction we have:

$$\widehat{f}(\xi) = \frac{1}{2}\left[\overline{\widehat{h}} \cdot \widehat{s}(2\xi) + \overline{\widehat{h}} \cdot \widehat{d}(2\xi + \pi)\right] \tag{3.33}$$

If we do not subsample, we obtain an undecimated wavelets decomposition. It will increase computational time, for a signal of length $N = 2^p$, since at each level i we have compute 2^p points and not 2^{p-i} points for the decimate case. Although this decomposition is not orthonormal anymore but it has the advantage of being grid's independent. We now do a recall on the Wavelet packets decomposition.

3.2.3 Wavelet Packets Representation

The wavelets decomposition is obtained by projection of each subspace V_{j-1} on the direct sum of two orthogonal spaces: $V_{j-1} = V_j \oplus W_j$. Instead of dividing only V_{j-1} we can decide to operate also the same division on the details space W_{j-1} to obtain a binary wavelet packets decomposition since there is symmetry now. We then have a recursive symmetric splitting algorithm as opposed to the wavelet decomposition. For so we have to define $w_{lev,j}$, the filter used for the j^{th} space at the lev^{th} level to project on $W_{lev,j}$. That enables to have the new iteration relation for the projection $W_{lev-1,j} = W_{lev,2j} \oplus W_{lev,2j+1}$. To obtain the two components, also called children nodes, we convolve our signal either by the scaling or the wavelet functions. We obtain the two wavelet packet orthogonal basis:

$$\psi_{lev}^{2j} = \sum_n h[n] \cdot \psi_{lev-1}^{j}(. - 2^{lev-1}n) \tag{3.34}$$

$$\text{and } \psi_{lev}^{2j+1} = \sum_n g[n] \cdot \psi_{lev-1}^{j}(. - 2^{lev-1}n) \tag{3.35}$$

We denote $m_0, m_1 \in L^2(\mathbb{R})$, in the Fourier domain such that:

$$m_0(\xi) = \frac{\widehat{h}(\xi)}{\sqrt{2}} \quad \text{and} \quad m_1(\xi) = \frac{\widehat{g}(\xi)}{\sqrt{2}}, \tag{3.36}$$

and we obtain $w_{lev,j}$ in the Fourier domain such that:

$$\widehat{w}_{lev,j}(\xi) = \prod_{i=0}^{lev-1} m_{\alpha_i}(2^{-i}\xi) \tag{3.37}$$

where α corresponds to the decomposition of j in the dyadic decomposition, $j = \sum_{i=0}^{lev-1} \alpha_i.2^i$. And we have $\forall l \in \mathbb{N}$, $\oplus_{j=0}^{2^l-1} W_j^l = V_0$.We represent this tree for the one dimensional case on figure 3.2. This algorithm enables to build a large family of spaces that will be used for our

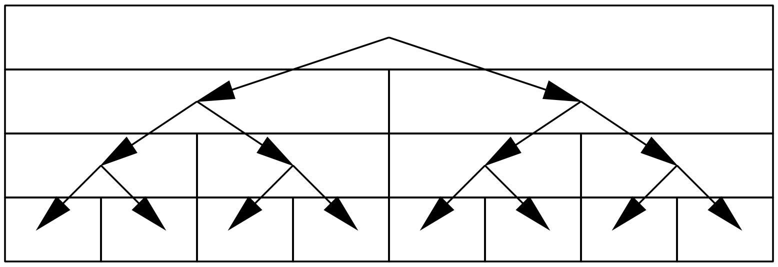


Figure 3.2: *One dimensional Wavelet Packets Decomposition*

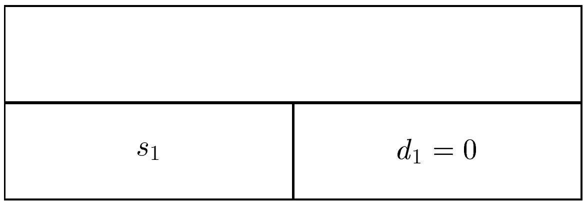
preprocessing.

3.3 Deconvolutions, sharpening and smoothing

In this chapter, we present various applications to deconvolve, sharpen or smooth one and two-dimensional signals. All these examples are based on the wavelets representation and iterative algorithms. In the first example, we are interested in restoring signals that have been blurred because of a lost of informations. Instead of having a signal f, for example we only have is low pass decomposition s. The second example is about sharpening and smoothing signal components. And finally, we present numerical results of these examples for one and two-dimensional data, including the removing of Barbara's vest stripes.

3.3.1 Applications to deconvolution

We used in this chapter the notations used before. We start from an original signal f_{orig} which we only have a blurred version f as only the low pass decomposition "s" has been kept. It can be written as $\widehat{f} = \bar{m}_0 \cdot \widehat{s}$, with $\widehat{s} = m_0 \cdot \widehat{f_{orig}}$ and $\widehat{d} = m_1 \cdot \widehat{f_{orig}}$, where "$d$ information" has been removed. We have to compute the high pass decomposition coefficients d to reconstruct f_{orig}. We can apply a division in the Fourier domain but it will imply instability problems. Let's define $f_1 = f, \quad s_1 = s \quad$ and $\quad d_1 = d$. We will use the capital letter to define the Fourier transform of a signal, $F = \widehat{f}$.



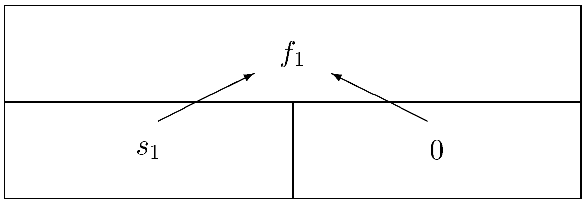
.

Figure 3.3: *Initial state, virtual tree showing high pass coefficients set to zero*

Remark 3.3.1. *The term "virtual" and "real" are used to define the different trees construct for the undecimated wavelet packets representation. "Real trees" correspond to the representation that have a true meaning like any decimated representations. Some wavelet packets trees do not correspond to a real tree. This feature is due to the fact that we do not have anymore an orthogonal decomposition by using undecimated wavelets. Thought,* *the low and high frequencies components are not anymore independent as in the decimated case. There now exists some conditions linking these two subspaces and at each level of the decomposition the dimension subspaces are reduced by one instead of being divided by two like in the decimated case.*

Presentation of the algorithm

For our example, we have by construction $d_1 = 0$ and recompose the signal to obtain the signal f_1:

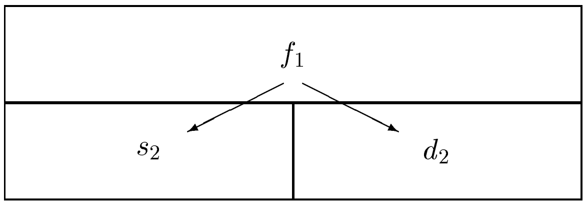


.

Figure 3.4: *Virtual tree showing reconstruction of $f_1 = f$ from s_1 and $d_1 = 0$*

$$\begin{aligned} F_1(\xi) = \widehat{f_1}(\xi) &= S_1(\xi).\overline{m_0}(\xi) + 0.\overline{m_1}(\xi) && (3.38) \\ \Rightarrow F_1(\xi) &= F(\xi) \cdot |m_0(\xi)|^2 && (3.39) \end{aligned}$$

We iterate a process that we explain now to obtain f_2. We obtain s_2 and d_2 by decomposition of f_1 such that $S_2(\xi) = m_0(\xi) \cdot F_1(\xi)$ and $D_2(\xi) = m_1(\xi) \cdot F_2(\xi)$ in the Fourier domain.

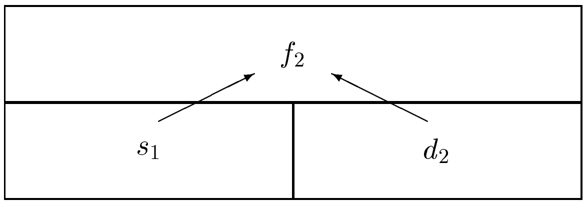


.

Figure 3.5: *Real tree for the decomposition of f_1 in s_2 and d_2*

We set $s_2 = s_1$, and we reconstruct f_2:

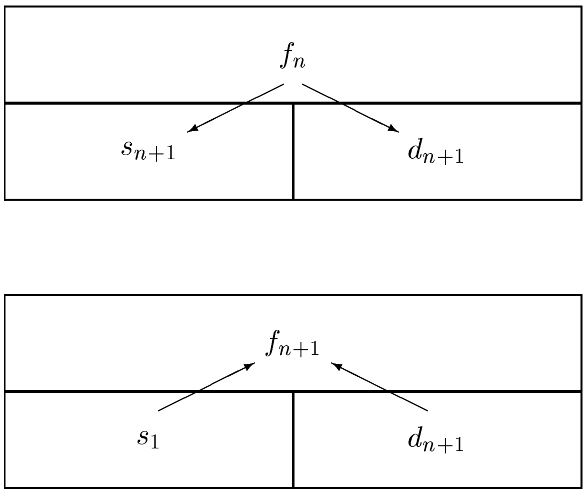
$$F_2(\xi) = S_1(\xi).\overline{m_0}(\xi) + D_2(\xi).\overline{m_1}(\xi) \tag{3.40}$$



.

Figure 3.6: *Virtual tree showing reconstruction of f_2 from s_1 and d_2*

We iterate this step. For each n, we decompose f_n in s_n and d_n and reconstruct f_{n+1} , setting $s_n = s_1$, we obtain the following trees:



.

Figure 3.7: *Real and virtual trees showing decomposition of f_n in s_n and d_n, and reconstruction of f_{n+1} setting $s_{n+1} = s_1$*

$$\forall n \geq 2 \qquad F_{n+1}(\xi) = S_1(\xi).\overline{m_0}(\xi) + D_{n+1}(\xi).\overline{m_1}(\xi) \tag{3.41}$$

with $S_1(\xi) = m_0(\xi) \cdot F(\xi)$ and $D_{n+1}(\xi) = m_1(\xi) \cdot F_n(\xi)$. We can write the following relation:

$$\forall n \geq 2 \qquad F_{n+1}(\xi) = F(\xi) \cdot |m_0(\xi)|^2 + F_n(\xi) \cdot |m_1(\xi)|^2 \tag{3.42}$$

As we know that $F_1(\xi) = F(\xi) \cdot |mo(\xi)|^2$, by induction we easily show that:

$$\begin{aligned} \forall n \geq 2 \qquad F_n(\xi) &= F(\xi) \cdot |m_0(\xi)|^2 \cdot (\sum_{i=0}^{n} |m_1(\xi)|^2) \\ \forall n \geq 2 \qquad F_n(\xi) &= F(\xi) \cdot |m_0(\xi)|^2 \cdot (\frac{1 - |m_1(\xi)|^{2n+2}}{1 - |m_1(\xi)|^2}) \end{aligned} \tag{3.43}$$

As we have $|m_0(\xi)|^2 + |m_1(\xi)|^2 = 1$, then

$$\forall n \geq 2 \qquad F_{n+1}(\xi) = F_1(\xi) \cdot (1 - |m_1(\xi)|^{2n+2}) \tag{3.44}$$

We assume that the comportment of m_0 around zero is such that $|m_0(\xi)|^2 = 1 - \lambda|\xi|^\gamma + o(\xi^\gamma)$, we have

$$\lim_{n \to \infty} F_n(\pi) = 0 \tag{3.45}$$

And we can conclude:

$$\forall \epsilon > 0, \exists \delta < 1 / \forall \xi \in [-\pi + \epsilon, \pi - \epsilon] \quad |m_1(\xi)| < \delta \tag{3.46}$$

$$\Rightarrow \forall \xi \in [-\pi + \epsilon, \pi - \epsilon], \lim_{n \to \infty} F_n(\xi) = F_1(\xi) \tag{3.47}$$

And we also have $F_n(\pi) = 0$. By letting δ_{x_0} be the Dirac function in x_0 we can write that:

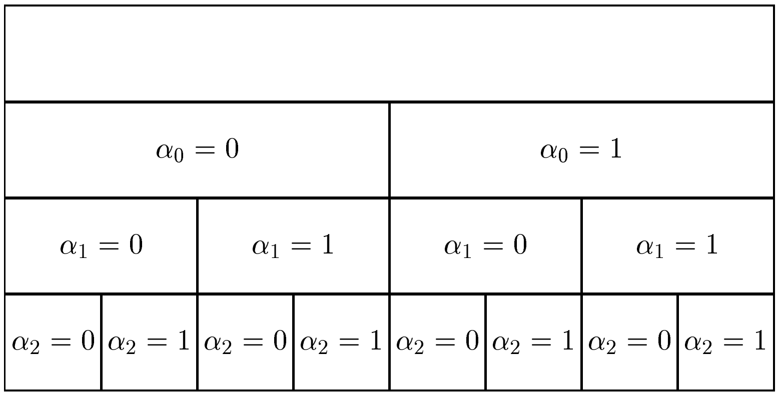
$$\lim_{n \to \infty} F_n = F_1 \cdot (1 - \delta_\pi). \tag{3.48}$$

The convergence of this algorithm is given by the fixed point theorem.

3.3.2 Application to sharpening and smoothing

In this chapter we show how to apply our algorithm to enhance or smooth some frequency bands without creating artifacts. We are working with discrete signal living in $L^2([0, N-1])$. At each level lev, for all filters $\omega_{lev,\alpha}$, $O \leq \alpha < 2^{lev}$, we can write its Fourier transform as:

$$\widehat{\omega}_{lev,\alpha}(\xi) = \prod_{i=0}^{lev-1} m_{\alpha_i}(2^{-i}\xi) \quad \text{where} \quad \alpha = \sum_{i=0}^{\infty} \alpha_i . 2^{(lev-1)-i}, \quad \alpha_i = 0 \text{ or } 1 \tag{3.49}$$



.

Figure 3.8: *Tree Classification on p levels with* $\alpha = (\alpha_0, \alpha_1, \cdots, \alpha_p)$

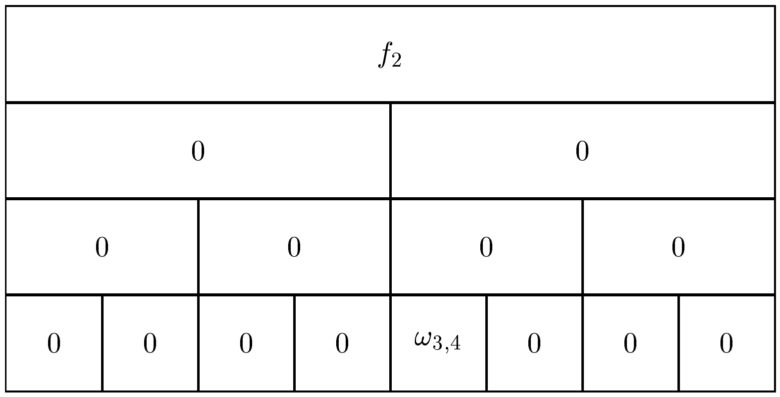
For a given signal f we compute its projections on the different subspace given by its wavelet packets decomposition, and obtain the following formula:

$$f_{\omega_{lev,\alpha}} = \sum_{i=0}^{N-1} \langle \omega^i_{lev,\alpha}, f \rangle \omega^i_{lev,\alpha} \quad \text{where} \quad \omega^i_{lev,\alpha} = \omega_{lev,\alpha}(. - i). \tag{3.50}$$

Similarly at the previous chapter, we can write in the Fourier space the relation given by the "construction-decomposition" of the wavelet packets $\omega_{lev,\alpha}$.We obtain f_1 by using its wavelet packets decomposition until the level lev, we set to zero all the wavelet packets but $\omega_{lev,\alpha}$:

$$\widehat{f}_1(\xi) = |\widehat{\omega}_{lev,\alpha}(\xi)|^2 . \widehat{f}(\xi) \tag{3.51}$$

And then we reconstruct our virtual tree to obtain f_2. By induction we obtain f_n with the



.

Figure 3.9: *Virtual tree with all components set to zero but* $\omega_{3,4}$

same procedure. And we have:

$$\widehat{f_n}(\xi) = |\widehat{\omega}_{lev,\alpha}(\xi)|^{2n}.\widehat{f}(\xi) \tag{3.52}$$

So if we have $\gamma \in \mathbb{R}^{p+1}$, we can construct

$$h_p(\xi) = \sum_{i=0}^{p} \gamma_i \cdot f_i(x) \tag{3.53}$$

that we can write in the Fourier space:

$$\widehat{f}_p(\xi) = \sum_{i=0}^{p} \gamma_i \cdot \widehat{f}_i(\xi) \Rightarrow \widehat{f}_p(\xi) = [\sum_{i=0}^{p} \gamma_i |\widehat{\omega}_{lev,\alpha}|^{2i}] \cdot \widehat{f}(\xi) \tag{3.54}$$

We fix $\gamma_0 = 1$ and we obtain

$$\Rightarrow \widehat{f}_p(\xi) = [1 + \sum_{i=1}^{p} \gamma_i |\widehat{\omega}_{lev,\alpha}|^{2i}]\widehat{f}(\xi) \tag{3.55}$$

By using the limited development of $x \mapsto (1+x)^\beta$ in zero:

$$(1+x)^\beta = 1 + \beta \cdot x + \frac{\beta(\beta-1)}{2} \cdot x^2 + \ldots + \begin{pmatrix} \beta \\ n \end{pmatrix} \cdot x^n + o(x^n) \tag{3.56}$$

And we also have for $x \mapsto (1-x)^\beta$ a similar result:

$$(1-x)^\beta = 1 - \beta \cdot x + \frac{\beta(\beta-1)}{2} \cdot x^2 + \ldots + (-1)^n \begin{pmatrix} \beta \\ n \end{pmatrix} \cdot x^n + o(x^n) \tag{3.57}$$

And then, when we choose $\gamma_i = (-1)^i \begin{pmatrix} \beta \\ i \end{pmatrix}$ we obtain:

$$\widehat{f_n}(\xi) = \widehat{f}(\xi).(1 - |\widehat{\omega}_{lev,\alpha}|^2)^\beta + O(|\widehat{\omega}_{lev,\alpha}(\xi)|^{2n}) \tag{3.58}$$

As we saw in the precedent chapter, we have convergence everywhere but at $\xi = \pi$. It corresponds to the Nyquist frequency. In the Haar case, it's the vectorial space $[\ 1\ -1\ 1\ -1 \cdots 1\ -1]\cdot \mathbb{R}$ that is orthogonal to the subspace spanned by the low frequency component. And it is the reason why we cannot recover it. We observe that $\beta = 1$ is a key factor. And β's value will determinate if we enhance or blur the corresponding wavelet packets.

3.3.3 Numerical Examples of one dimensional deconvolution

We have shown in the previous chapters that for a given one-dimensional signal f, with no energy at the Nyquist frequency, we can deconvolve it using the algorithm defined before. We start with a periodic chirp ($N = 128$ points) $x \mapsto \sin(cx \cdot (N - x))$ that does not have any high frequencies. This chirp has been chosen to have low frequencies concentrated in the center of the signal as figure 3.10 shows. We iterate the deblurred algorithm described

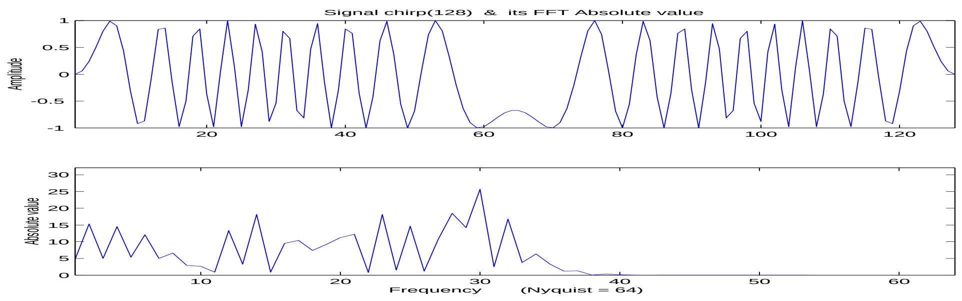


Figure 3.10: *A first chirp and its Fourier transform*

before, and observe the evolution of the signal on figure 3.11. The convergence is quite fast as planned, as $support(\widehat{f})$ is limited (no high frequencies). After an hundred iterations, there is almost no variations. We observe that the differences only appear where we have high frequencies as expected, as $|m_0|$ has more energy in the low frequencies than among the high frequencies.

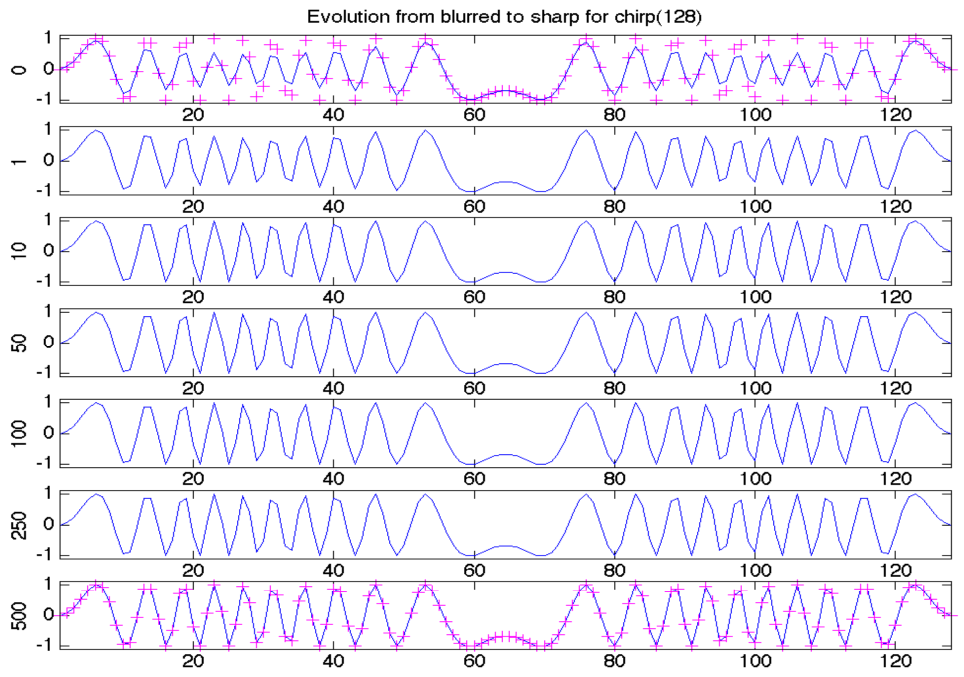


Figure 3.11: *A first signal, starting from a blurred version converging to the original*

We construct a second example with more energy in the high frequencies, as we can now notice on figure 3.12. We observe that the blurred signal has lost much more

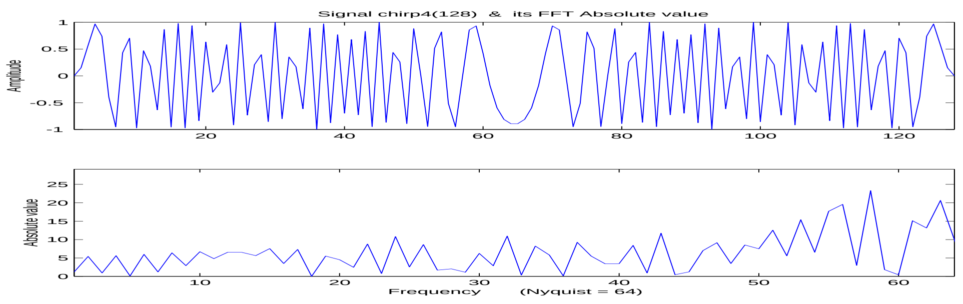


Figure 3.12: *A second chirp and its Fourier transform*

informations and we see the same kind of result but with a much slower convergence.

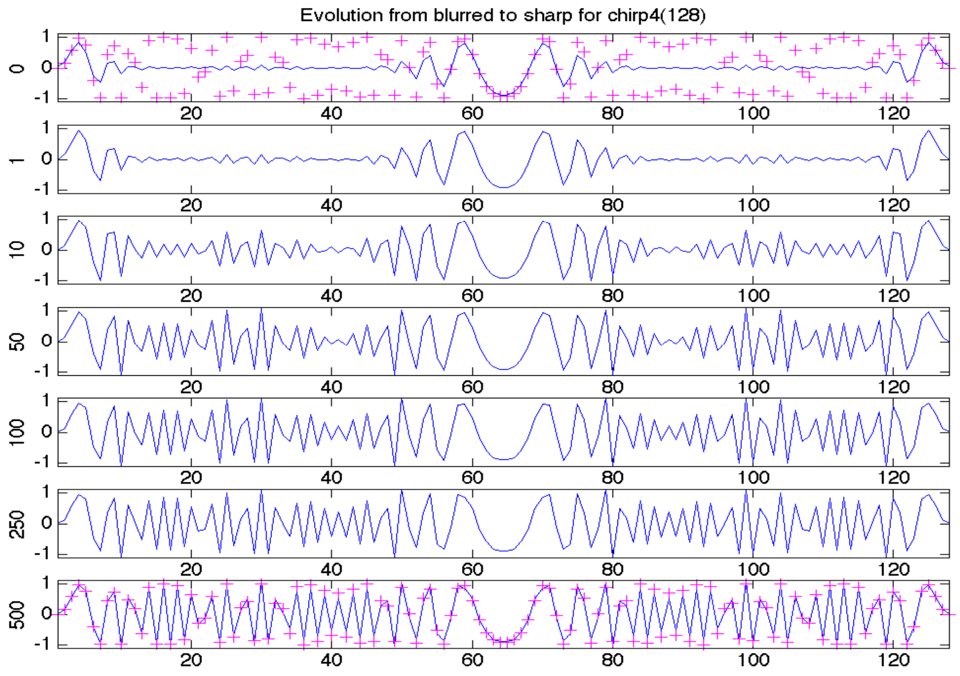


Figure 3.13: *Starting from a blurred version converging to the original*

3.3.4 Numerical examples of sharpening and smoothing

In this chapter we present numerical results for one and two-dimensional man made signals and also for a real image. For the one-dimensional case, we create chirps that allow a better understanding of the algorithm, since spatial and frequential locations are linked. We then show result for two-dimensional signals made of chirp and sine products. And we finish with the removing of Barbara's vest stripes.

Examples in the one-dimensional case

To visualize our sharpening or smoothing we are going to use our previous chirp (1024 points) and we are going to enhance or smooth each frequency band on the second level. So we obtain two examples for each sub-band. We can see on figure 3.14 and 3.15 the

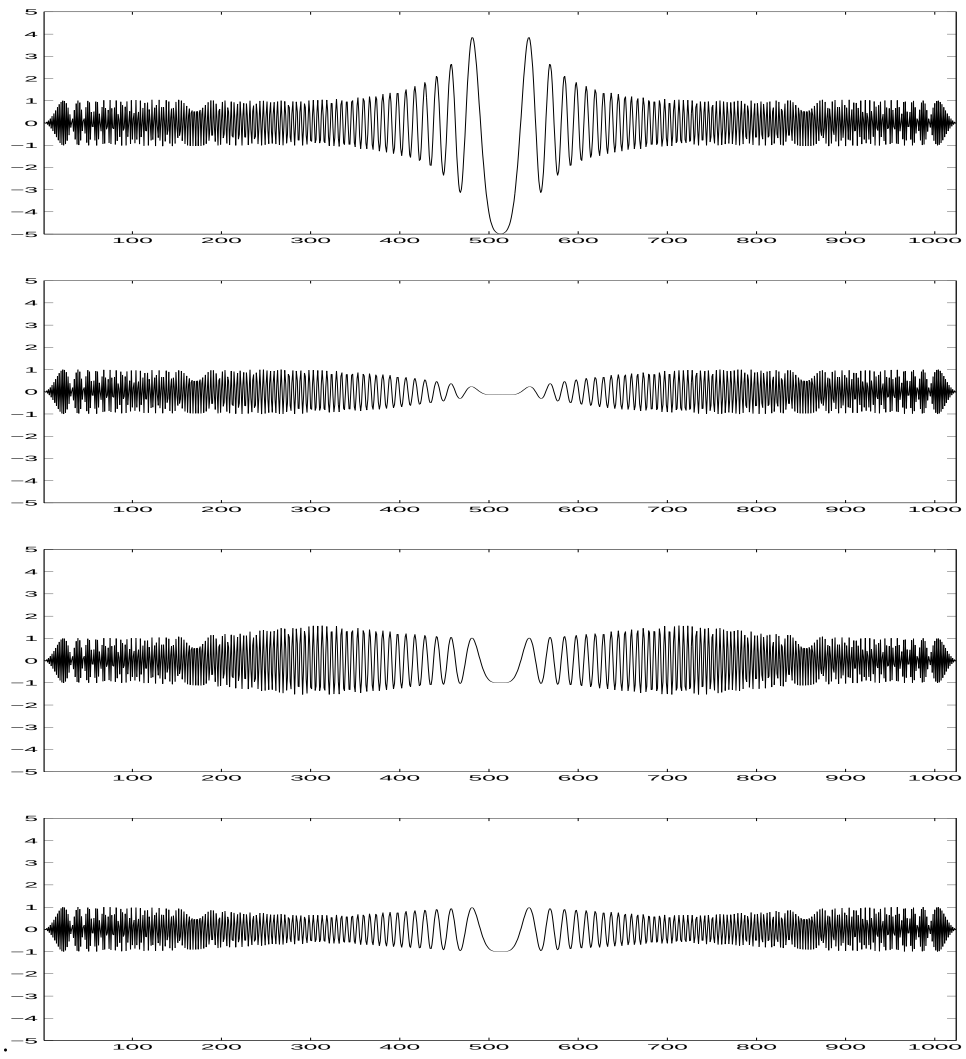


Figure 3.14: *Sharpening and smoothing of* $\omega_{2,0}$ *and* $\omega_{2,1}$ *on a chirp of length* $N = 1024$

successive effects of the sharpening and smoothing. In each case, we manage to modify the energy level without creating any artifacts. The effects on the signal is more or less important depending on how much energy is contained in the sub-band.

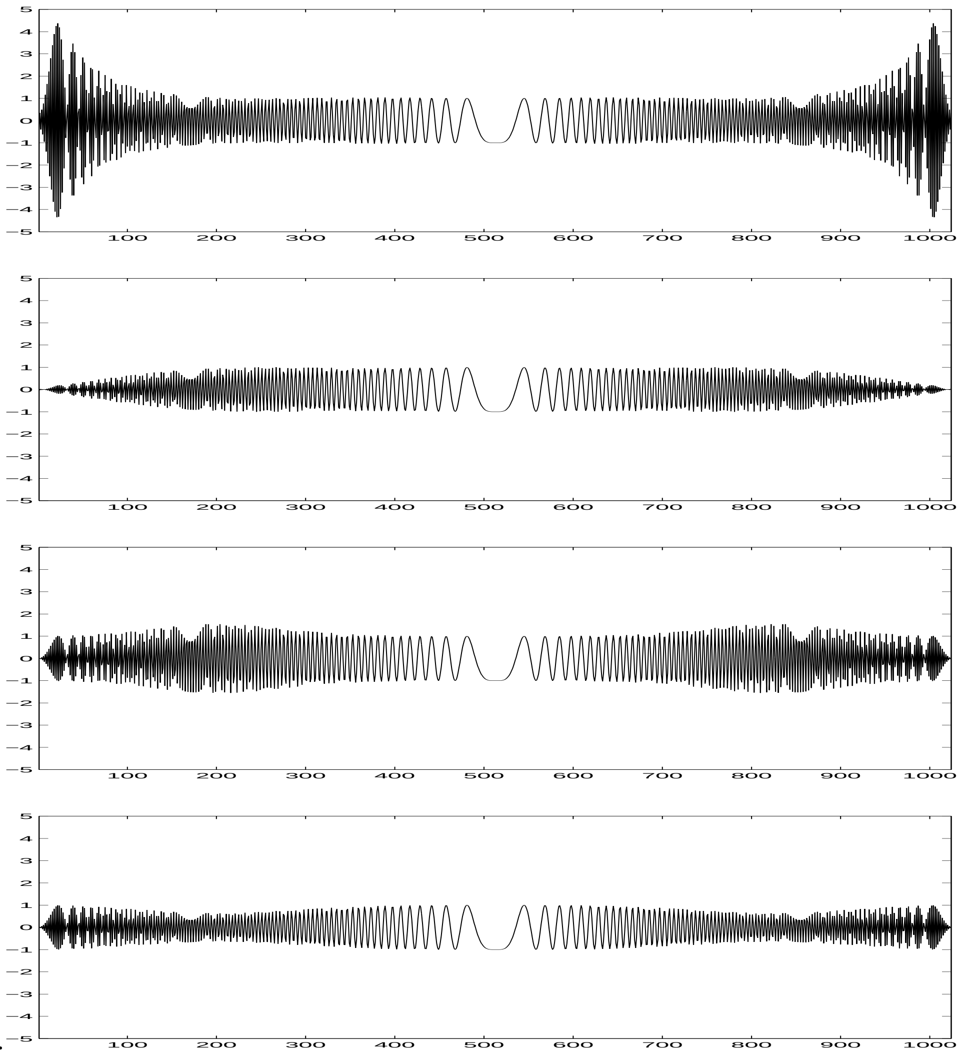


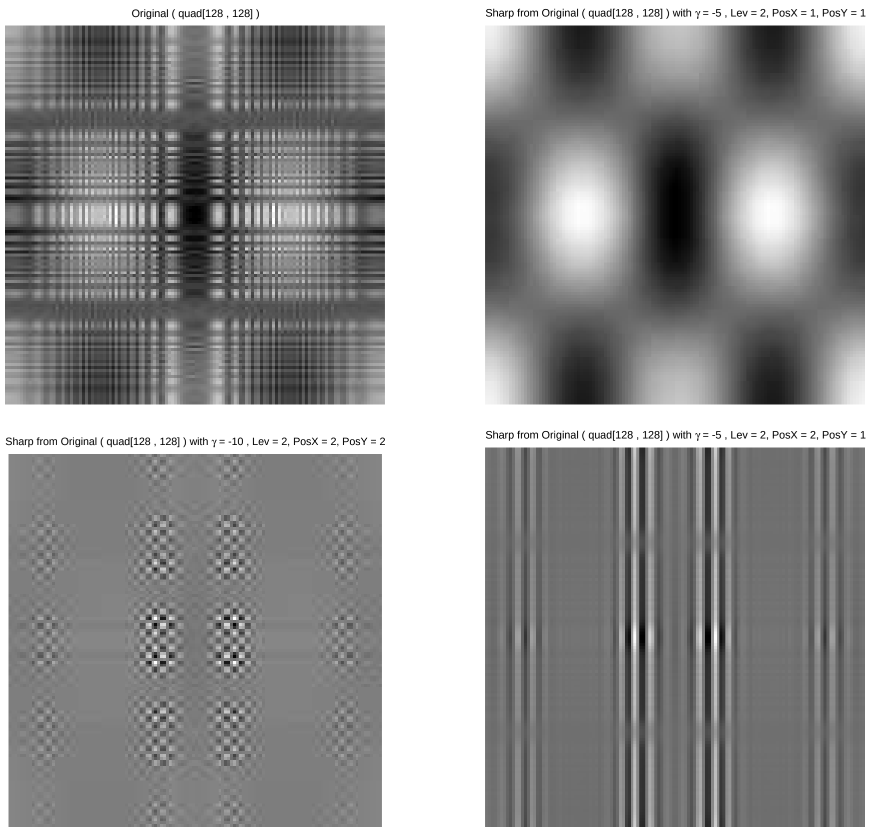
Figure 3.15: *Sharpening and smoothing of* $\omega_{2,2}$ *and* $\omega_{2,3}$ *on a chirp of length* $N = 1024$

Examples on two dimensional signal

We create a two-dimensional signal which is the sum of two functions: a product one dimensional chirp that allows localization of the frequencies (as we have with the one-dimensional chirp) for a better understanding, the product of two one-dimensional low frequency cosines. For $(x, y) \in [1, N]^2$, the intensity I of the signal can be written as follow:

$$\begin{aligned} I(x,y) &= \left[50 + \left(80 - \left|\frac{N}{2} - x\right|\right) \cdot \cos\left(\frac{\pi x \cdot (N-x)}{64}\right)\right] \\ &\cdot \left[50 + \left(80 - \left|\frac{N}{2} - y\right|\right) \cdot \cos\left(\frac{\pi y \cdot (N-y)}{32}\right)\right] \\ &+ 5000 \cdot \cos(\frac{\pi x}{32}) \cdot \cos(\frac{\pi y}{64}); \qquad (3.59) \end{aligned}$$

The two dimensional signal is represented on figure 3.16, the upper left image.


Original (quad[128 , 128])
Sharp from Original (quad[128 , 128]) with γ = -5 , Lev = 2, PosX = 1, PosY = 1
Sharp from Original (quad[128 , 128]) with γ = -10 , Lev = 2, PosX = 2, PosY = 2
Sharp from Original (quad[128 , 128]) with γ = -5 , Lev = 2, PosX = 2, PosY = 1

.

Figure 3.16: *Sharpening and smoothing of the image given by equation 3.59 for different subspaces and values of λ*

Example on a real image

We now work on a real image and we present result obtained with Barbara, an image $(512, 512)$ showed on figure 3.17. We want to remove the vertical lines appearing on "Bar-

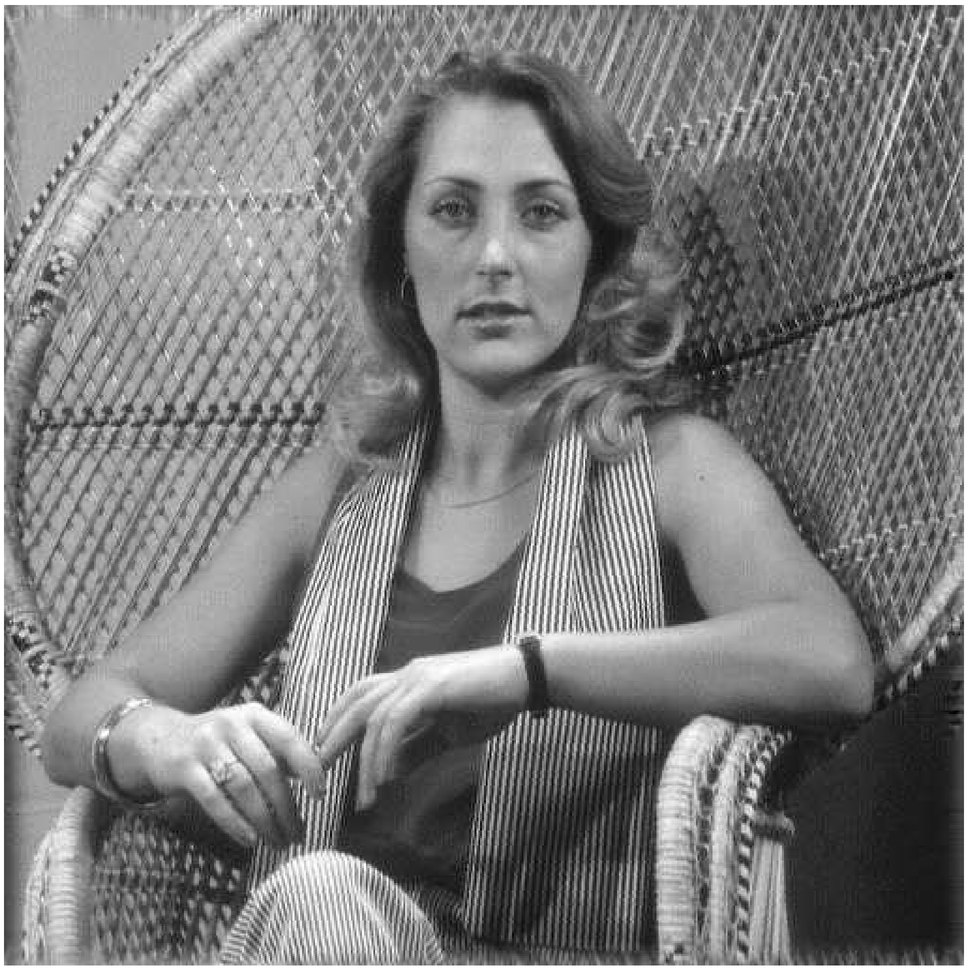


Figure 3.17: *Original Image "Barbara"* $(512, 512)$

bara's vest", just below her left arm without erasing the shadows. The stripes correspond to the wavelets of the first level, with low frequency in the vertical direction and high frequency in the horizontal direction. The wavelet representation is adapted for such kind of signals. The result can be observed on figure 3.18. We deconvolved only a small part of the vest and left the other part as it was. We can observe that the arm's shadow is still on the

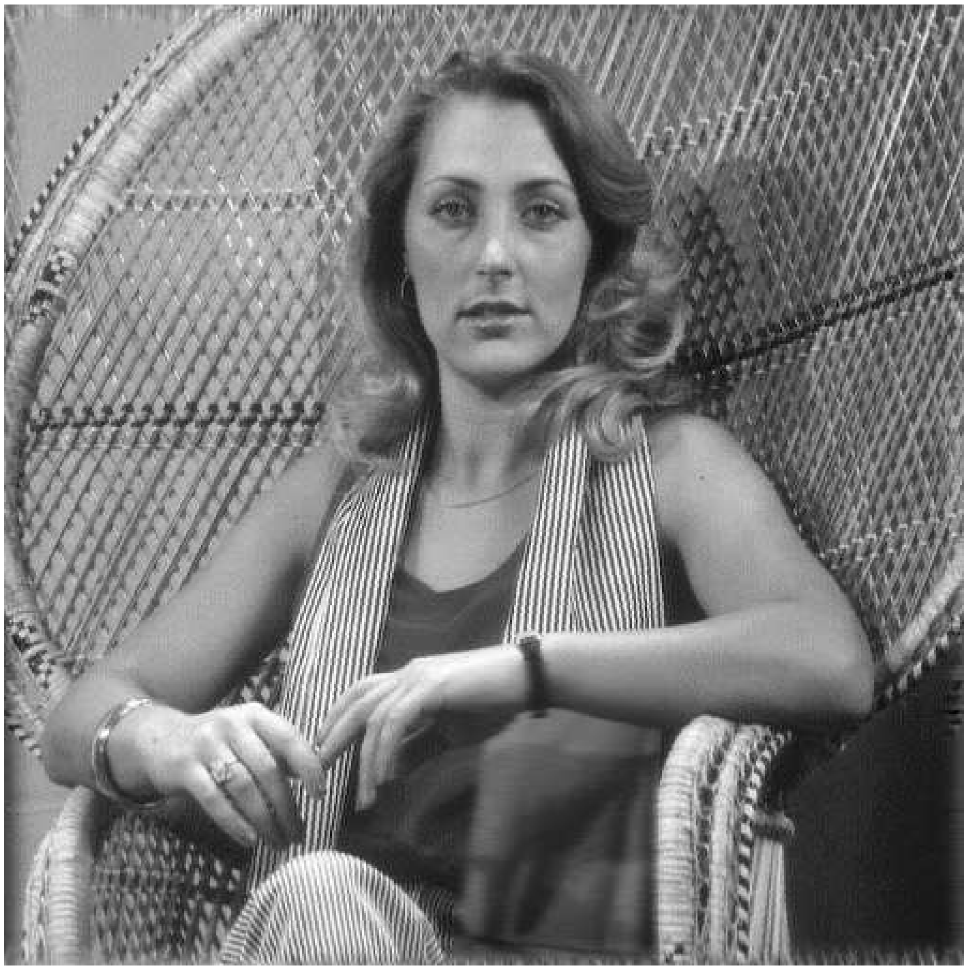


Figure 3.18: *Image "Barbara"* (512, 512) *partially deconvolved, the stripes of the vest have been removed*

vest, it didn't disappear with the algorithm. We observe that we do not loose resolution and sharpness as in the heat equation process or with a convolution.

3.4 Applications to Denoising

S.A.R. images with targets have some particular properties, more precisely the range of value (due to the targets) compresses the structures. We apply a preprocessing to be able to have a better denoising. In this report, we develop an algorithm to denoise radar images in four steps:

- Decomposition on Wavelets Packets Undecimated and Preprocessing,
- Applying a Threshold for targets separation and a γ correction,
- Denoising using multi-pass wavelet packets decomposition,
- Reconstruction of the complete signal

This algorithm has been implemented inside an existing platform for image and signal processing.

3.4.1 Preprocessing and Goals

Wavelets have been used for denoising. But we know that the results are not always good. Too many artifacts are introduced due to the data structure. In the case of S.A.R. images, we have a bad representation, as we can see on the figure 3.19 their statistical representation. Targets can be assimilate to Dirac functions or noise. To avoid the artifacts generated by this repartition, that essentially compress the structures and gives discontinuities due to the targets, we apply a preprocessing to our data set. In the first part we are going to extract the low frequencies, using undecimated wavelets. We know that the noise is almost inexistent in this component. When we have done this, we re-normalize our data. And in the second step we are extracting the targets, assimilated as high values. These targets modify the range and make impossible most of the denoising based on wavelet analysis. Since they compress the other data in a small range, by renormalisation. So we decide to extract them before denoising. But we have to be careful to not add artifacts to our data. And we are able to apply an optional "γ correction".

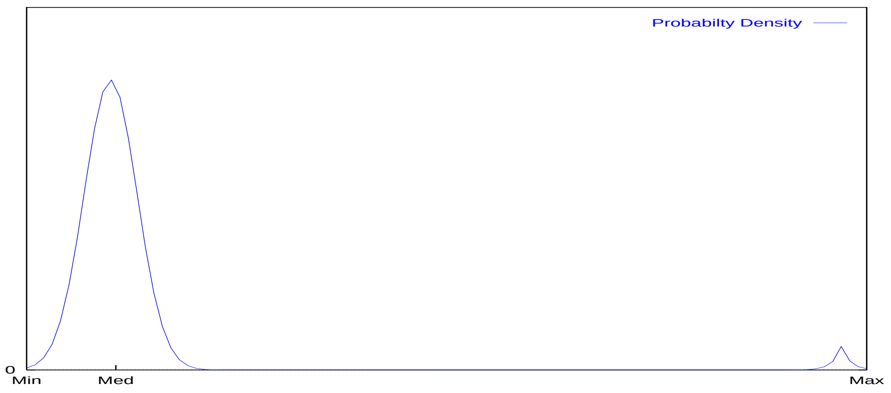


Figure 3.19: *Probability density function of a radar image with targets, showing the range of data.*

3.4.2 Extraction of Low frequencies

For each signal we subtract the low frequencies. We apply an undecimated wavelet decomposition until level three or four depending on the size of the signals. And we set to zero all the wavelets coefficients but the low frequencies components. And we rebuild our signal. This algorithm enables us to apply a soft threshold on our data. It corresponds to a multiplication by a bell in the Fourier space. We have extraction of low frequencies (at a fixed level) that is obtained by keeping the coefficients of the left box of the corresponding level and setting the other elements to zero. The reconstruction of the signal will give a signal containing the low frequency components. This scheme can be understood with the figure 3.20

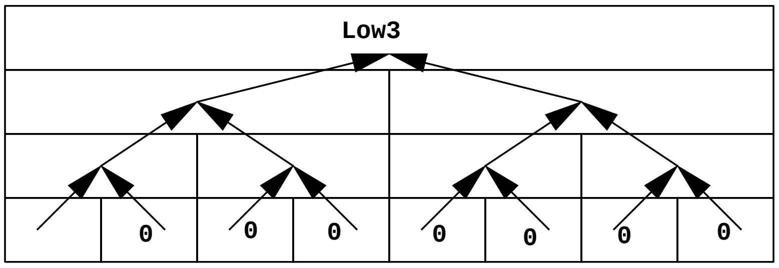


Figure 3.20: *One dimensional preprocessing using wavelet packets reconstruction.*

Remark 3.4.1. *After Wavelet Packets Decomposition undecimated, for a one dimensional* *signal (size N) we obtain ($2^{level} * N$) coefficients at each level. So we have ($\Sigma 2^{level} * N$) elements, it means ($(2^{(maxlevel+1)} - 1) * N$) coefficients if we have decomposition until level $maxlevel$. Instead of the N coefficients with decimated decomposition. For a two dimensional signal, the amount of data is even bigger as we have ($4^{(maxlevel+1)} - 1) * N$ elements.*

We have shown in the previous chapter that a similar computation can be done by keeping any wavelet packets subspace and setting to zero the others elements. We subtract this component to the original signal and the working data set is therefore composed of two components. We are keeping a side the low pass part and process the high pass part. We have a similar result for two-dimensional signals, like images that we will denoise later. The scheme for the decomposition-reconstruction is shown on figure 3.21. After the extraction of

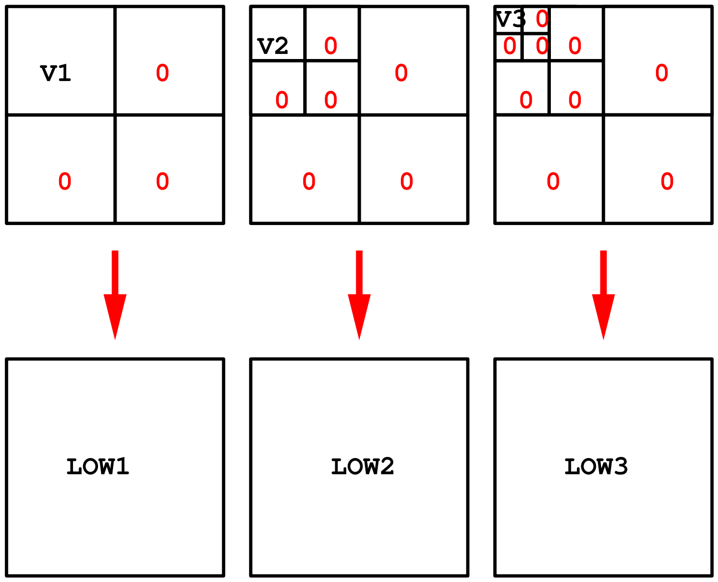


Figure 3.21: *Two-dimensional preprocessing using the wavelet packets reconstruction.*

the low frequencies, we obtain two components as previously. The low frequency component is kept as is, its complementary part needs processing. We apply a renormalisation using the a linear renormalisation that proceeds the “γ-correction”.

3.4.3 Renormalisation

We want to apply a renormalisation that keep some properties of our signal f to apply different operators. We re-normalize our signal between -1 and 1, and we also force the median to have a value equal to zero. To be able to apply a γ function on each side of the median. We apply a linear transformation, that we define as follows:

$$\begin{aligned} min &= \min_{i \in [1 \ldots N]} f(i) & (3.60) \\ med &= median_{i \in [1 \ldots N]} f(i) & (3.61) \\ max &= \max_{i \in [1 \ldots N]} f(i) & (3.62) \end{aligned}$$

And we have the following affine transformations:

$$\begin{aligned} if x \in [min, med] \Rightarrow f(x) = \frac{x - med}{med - min} & \quad (3.63) \\ if x \in [med, max] \Rightarrow f(x) = \frac{x - med}{max - med} & \quad (3.64) \end{aligned}$$

We represent the linear transformation on figure 3.22

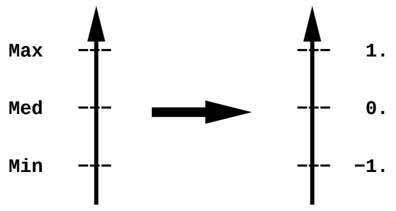


Figure 3.22: *Affine renormalisation on each side of the median value, that will enable to apply the "γ-correction", $x \mapsto sign(x)\,|x|^\gamma$, without lost of symmetry. The median is invariant by this transformation.*

3.4.4 Targets extraction

We are now working on the high amplitude part of the signal, as opposed to the previous part that was centered on the frequencies domain. This part is constituted of small structures, targets (high values that we compare to Dirac or characteristic function) and noise. Targets are represented by high positive values. We are extracting them to denoise the small structures without being influenced by the targets. We have different possibility to do so. But many artifacts can be introduced. The first and the easiest way is to apply a

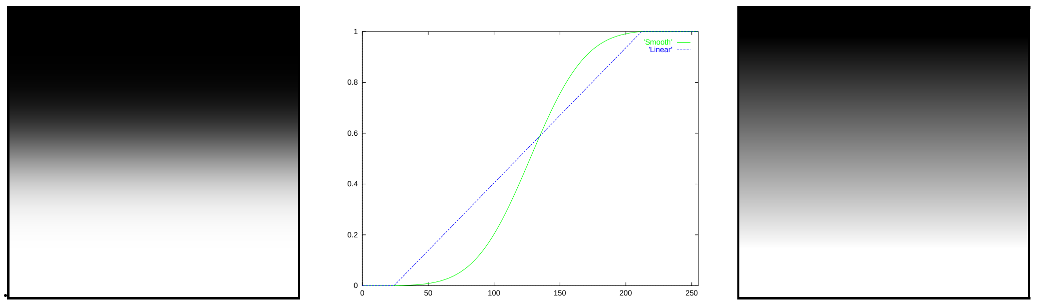


Figure 3.23: *Influence of regularity on vision. The central graph show two one-dimensional signals. One is C^∞ while the other is piecewise C^1. We generate two-dimensional signals from them to observe the importance of the C^1 property in vision.*

hard threshold. We separate the signal in two parts by assimilating targets to the highest few pour-cents. By iteration we can extract one, two, three or more pour-cents. And the separation between targets and simple structures is done but it is a manual procedure. The second method is based on the probability density. We know that our data stretch the range of value and we assimilate it to a bi-modal density. Computing the quantiles will enable us to find the separation of these two modes. As a matter of fact, there is a big jump of value between the two modes and that is going to generate a jump in our quantiles. These two methods have the drawbacks of applying hard threshold. The last possibility that we consider is based on the same idea at the extraction of the low frequencies. We want to apply an operator without adding artifacts. As we know image regularity is important as figure 3.23 shows. As the matter of fact, when we extract our data by hard threshold we obtain signal without enough regularity. Our signal can be assimilate to a “hat function”, and we know that it implies artifacts. We want to apply a soft threshold to separate targets from the small structures. Moreover, we know that targets have positive values so we only apply this soft threshold to $[0, 1]$, that corresponds to the value range contains between the median and maximal values. Our function should be in $C^2(\mathbb{R})$, as the one proposed on the graph on figure 3.24. We apply now a “γ-correction” that will help to concentrate or spread

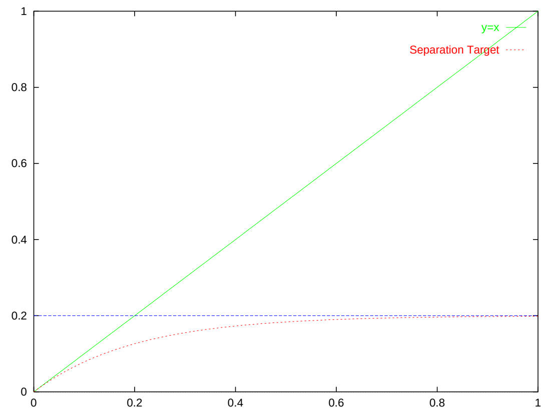


Figure 3.24: *Target extraction using soft threshold to avoid artifacts as shown on figure 3.23*

the different probability densities.

3.4.5 γ-Correction

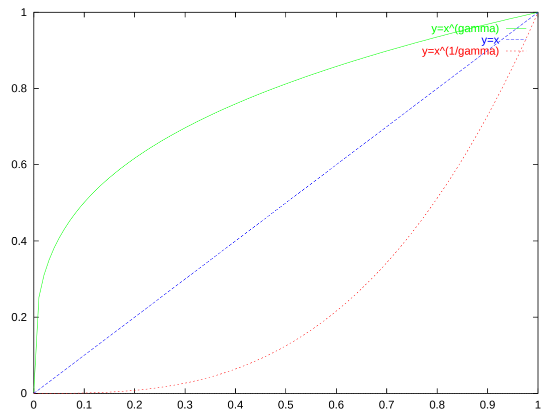


Figure 3.25: *Three different γ functions, $x \mapsto x^\gamma$, to emphasize or reduce the various different structures by modifying the probability density functions. A value of γ between 0 and 1 will create a function above the identity function and then will "push" the data to greater value as opposed to $\gamma > 1$ that will "pull" the data to zero*

After renormalisation and extraction of the targets, we can apply a γ-function,$x \mapsto x^\gamma$, to emphasize the structures compared to the noise. On each side of origin we apply the "*gamma*-functions" with different values for γ. We show their graphs on figure 3.25. We apply this transformation on $[-1, 0]$ and $[0, 1]$ separately. Denoising can now be running on this preprocessed signal. The median is not modified with the "γ-function", as it is a non-decreasing function. We present a recapitulatory of our processing before applying a denoising.

3.4.6 Preprocessing recapitulatory

We have shown in the last few chapters how to extract parts of our data to be able to have favorable conditions for denoising. We represent this summary as a diagram on figure 3.26. We have separated our data in three parts: low frequencies, high amplitudes called

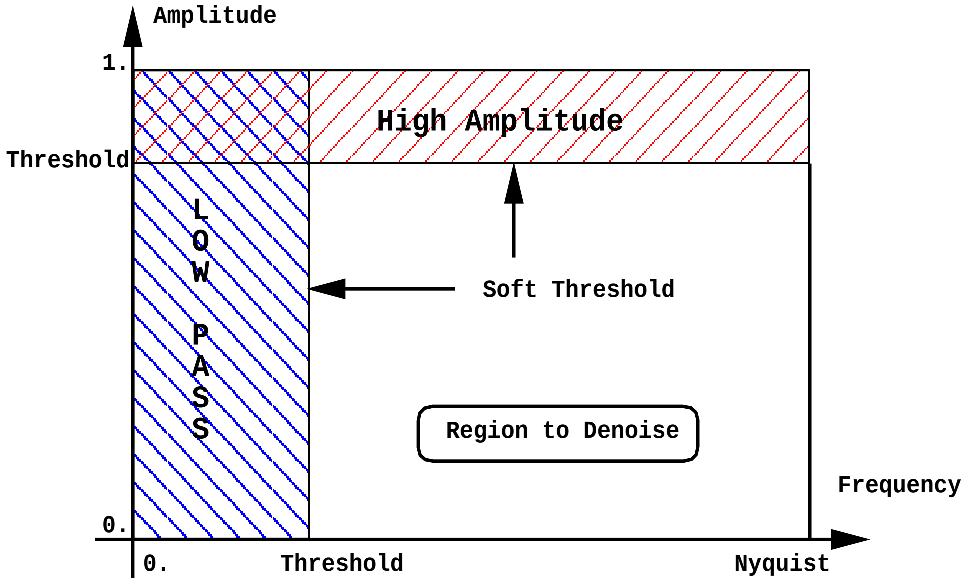


Figure 3.26: *Diagram showing the preprocessing steps before denoising*

targets and small structures to denoise. We know that the low frequencies do not contain any noise, we keep this apart and add it later. The targets are considered as high positive values, white dots, the noise does not have a veritable influence on their values. Targets will be added later too. We have to concentrate our denoising work on the small structures that correspond to the last part. The next chapter is going to show how to apply a multi-pass denoising using wavelet packets.

3.4.7 Denoising using multi-pass wavelet packets decomposition.

We apply here an algorithm developed by Lionel Woog, in his Ph.D. thesis [30], to denoise our data. Decimated wavelet packets are used for this step for fast computations. This algorithm is composed of :

- Multi-pass wavelet packets decomposition-recomposition with a hard threshold (as shown on figure 3.27)
- Spatial Spin Cycle
- Spin Cycle on the filters

Spin Cycle is used to compensate the dependence on dyadic grid or on the filter with an average of the results. To do so, we translate the data. For the spatial spin cycle, we shift the data of one pixel in each direction and repeat the algorithm, and take the average value. While the for the filters, we repeat the algorithm with various filters, before average. Usually four spatial translations and four filters are adequate. The Multi-pass algorithm

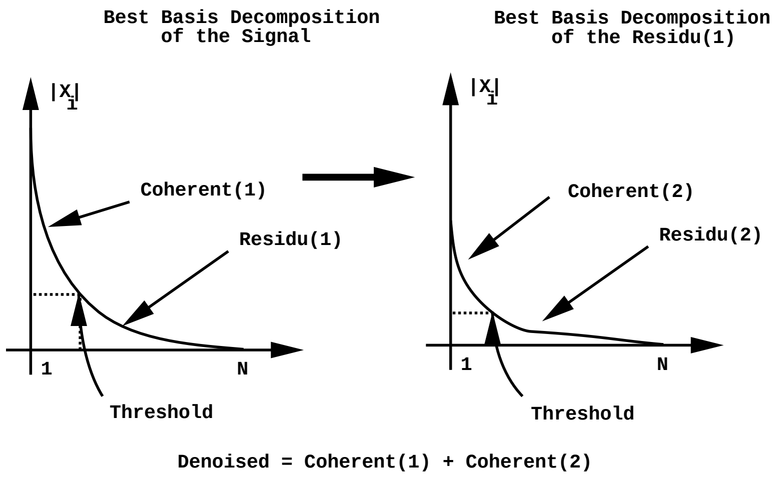

Best Basis Decomposition of the Signal
Best Basis Decomposition of the Residu(1)
|X_i|
|X_i|
Coherent(1)
Residu(1)
Coherent(2)
Residu(2)
1
N
1
N
Threshold
Threshold
Denoised = Coherent(1) + Coherent(2)

Figure 3.27: *Two-passes algorithm for denoising using a hard threshold*

is composed of the following steps (the first three corresponds to the one pass best basis thresholding):

- Wavelet Packets decompositions and search of Best Basis.
- Application of a hard threshold after data sorting by absolute values.
- Wavelet packets reconstruction to obtain the coherent part.
- Iterations of these three steps on the residue, until the residue is considered as only noise.
- Summation of the coherent parts

A two pass algorithm is represented on figure 3.27. The advantage of this algorithm is that it enables to "see" structures in different steps. The second and third pass enable to catch the small structures living in each respective residues.

3.4.8 Summary and Results

Our algorithm can be seen as a separation of our image in three parts and a denoising of one of them. This summary can be seen as the diagram shown on figure 3.28. The com-

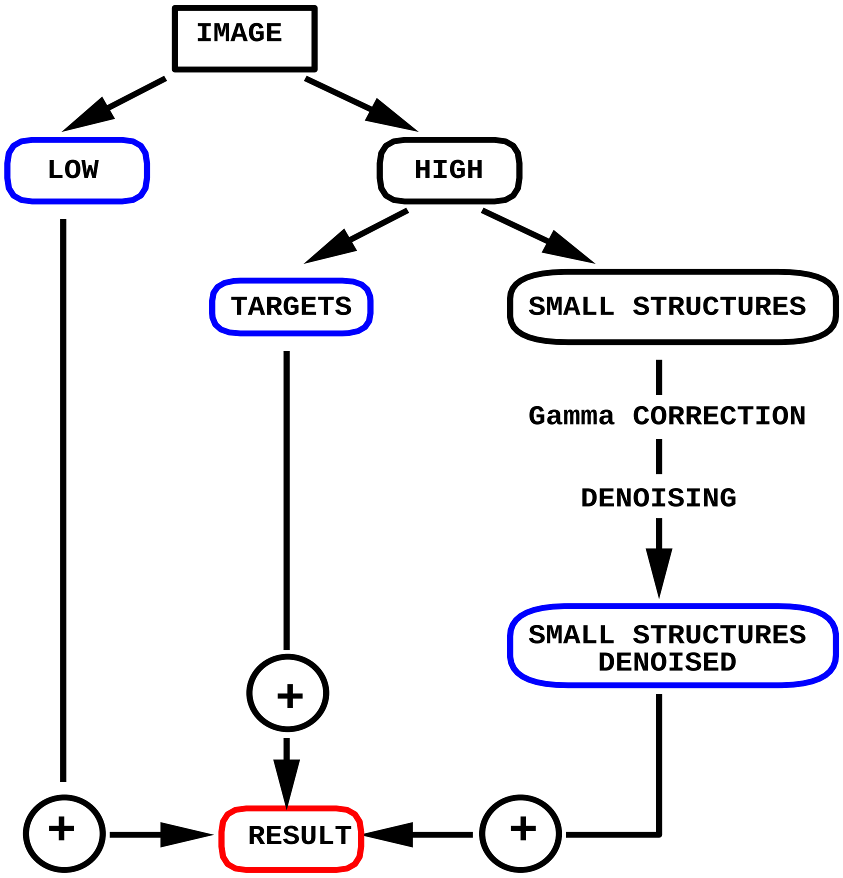
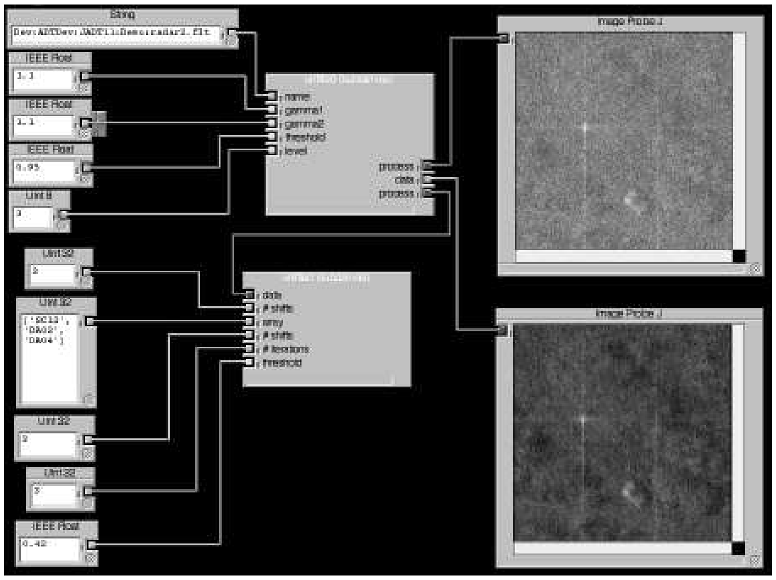


Figure 3.28: *Diagram showing the different steps of the global algorithm used for denoising*

bined algorithm has been implemented on a platform that contains others image processing programs. The canvas corresponding to this structure is shown on figure 3.29. And by applying the algorithm to the original image, shown on the left of figure 3.30, we obtain a interesting result on the right of figure 3.30.



```
String
Dev:ADTDev:JADTFli:Demo:radar2.flt
IEEE Float
1.1
IEEE Float
1.1
IEEE Float
0.95
UInt8
3
UInt32
2
UInt32
['SC12',
'DA02',
'DA04']
UInt32
9
UInt32
3
IEEE Float
0.42
```

Figure 3.29: *Canvas of the algorithm implemented in the platform*

.

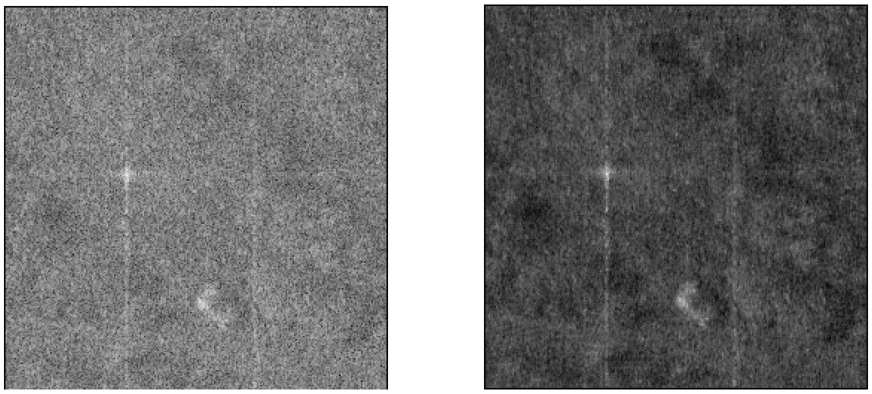


Figure 3.30: *Original and denoised SAR image, with our preprocessing and multi-pass algorithm*

3.5 Application to brain activation detection

In this chapter, we are confronted to the following medical problem: detection and localization of activation regions in human brain using functional MRI. Two different MRI scans were acquired: one baseline during which the subject was not performing any task, and another set during which the subject was performing a motor task. We would like to detect regions of activation. We will represent the data as two three dimensional data sets. The images should have the same spatial size but their respective number may differ. The third dimension corresponds to the temporal dimension, different realizations of the same state (active task or baseline). Each data set is a temporal sequence of one single slice of the brain. We assume that the slices in the activated data set and baseline are registered in the z-dimension as shown on figure 3.31.

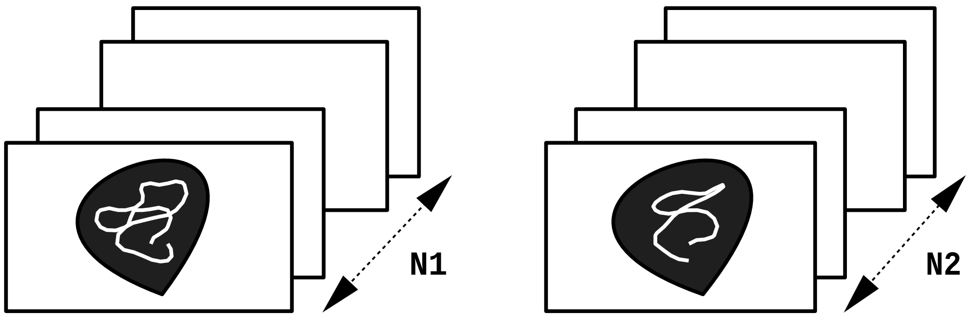


Figure 3.31: *Two data sets of brain given by the functional MRI, baseline and active task.*

The steps of the algorithm are the following:

- Decomposition at various scales of each image using the undecimated wavelets decomposition.
- For each wavelet coefficient location, estimation of the two probability density functions.
- Computation of a "Cost Subspace" for each subspace, that interprets the dissimilarity of the two probability density functions.

- Reconstruction of the "Cost Subspaces" to obtain the dissimilarity at a fixed level and merging of the various levels.

3.5.1 Preprocessing with Wavelets Decomposition

We represent our data with the multi scale analysis obtained with undecimated wavelets as explained before to have representation grid's independent. We use the algorithm developed by Roland Guglielmi. We represent this decomposition as a table. Each box has the same size as there is no decimation. But for an easier visualization we show them as a decimated decomposition on figure 3.32. The first raw is the original signal and the following correspond to the different levels of decomposition. We represent our decomposition on

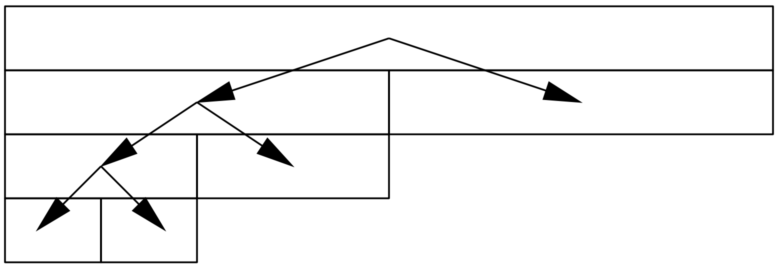


Figure 3.32: *One dimensional wavelets decomposition*

figure 3.33 for our two datasets. We apply this decomposition. For a fixed scale and a

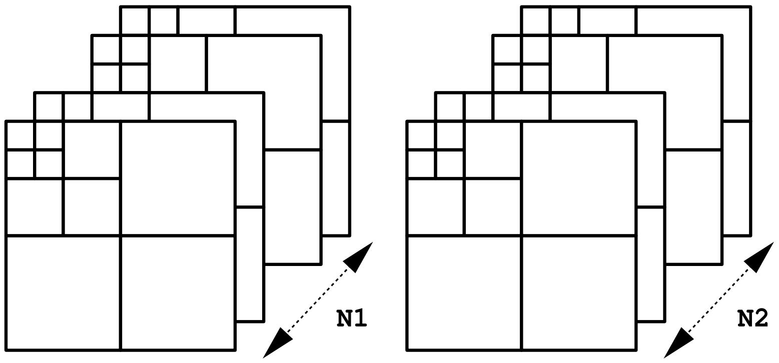


Figure 3.33: *Two dimensional wavelets decomposition of the two sets*

given location, we will compute for each subspace data set (that has the same size), the probability density function given by the respective N_1 and N_2 terms. We then determinate the dissimilarity of these two functions to compute the "cost subspaces" that will allow the reconstruction of an image that corresponds to the difference between our two data sets at this fixed scale.

Remark 3.5.1. *During the wavelet decomposition, we have a growing size of our data, for high level of decomposition, that is a real problem for two dimensional signals. It is the reason why our study does not use the wavelet packet decomposition as previously.*

3.5.2 Computations on each subspace

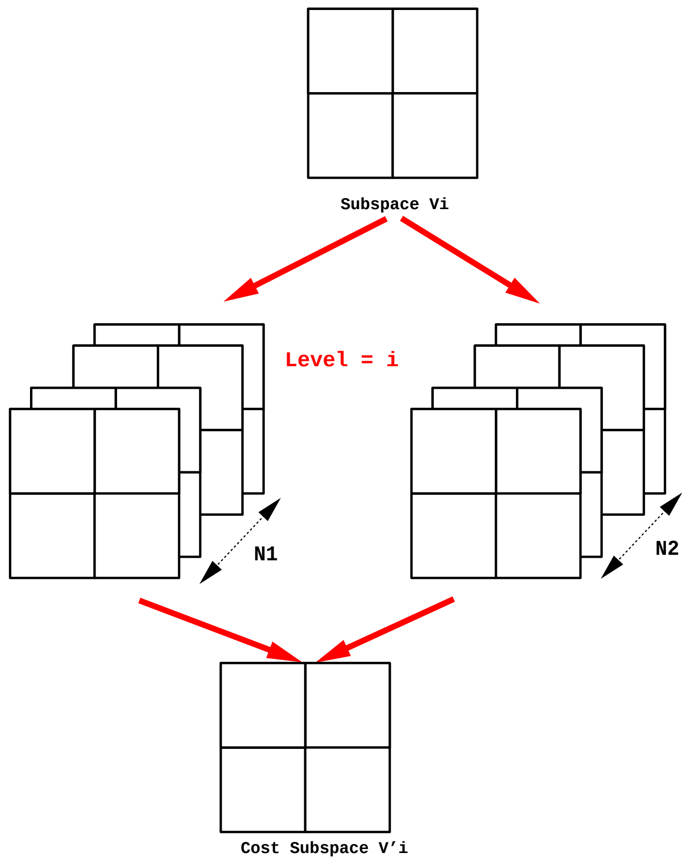


Figure 3.34: *Cost Subspaces computed for a fixed level (level $= i$).*

We want to determine if we can discriminate locally different activations between this two modes. We make a discrimination at each scale. We have four subspaces corresponding to each level, as we do not use the wavelet packets representation. We work in each fixed subspace, and we have respectively N_1 and N_2 realizations for each mode. For each level we compute the four cost subspaces as seen on figure 3.34. Therefore we have to compute a local cost according to the dissimilarity behavior of our data. We will call "Cost Subspace", the subspace corresponding to a signal (it has same dimension as this subspace) and gives us the local difference between our two data sets. For each pixel, of each subspace, we apply the same operations. The following steps are applied:

Step 1 Extraction of the N_1 and N_2 realizations with identical pixel location and Estimation of the two probability density functions

Step 2 Cost Computation using the differential entropy:

$$E(X_1, X_2) = \sum_{i=1}^{i=N} (p_i - q_i) \cdot \log(\frac{p_i}{q_i}) \geq 0 \tag{3.65}$$

Step 3 Creation of the Cost Subspace and reconstruction.

We give details of these steps in the next few chapters.

Estimation of the probability density function

We look for activation zones using a bin probability for each scale. We have now N_1 and N_2 set of images, and their respective undecimated wavelet decompositions. Therefore, we are interested in activation difference in the brain and not in the background. So later, when we build our probability density we will use this knowledge to compare the data. A noisy background will not affect our study. We want to detect the active parts of our data. We are working at various scales meaning, in our context, different levels of decomposition. Our study is done at each level separately. We work on the decomposition coefficients at fixed levels before reconstruction. We have two data sets, each containing images with identical sizes. These images are 256 by 128, and they have been obtained by M.D. Kevin Johnson (School of Medicine, Yale University). They correspond to slices of the brain in a

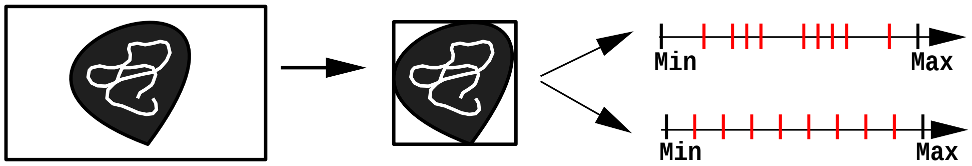


Figure 3.35: *Preprocessing of measure for the cost computation*

noisy background. We are only interested in what happens inside the brain. Therefore we consider the smallest rectangular region that contained the brain. Within this region, we estimate the probability density function using the first image of each data set, to obtain a global measure. The probability density is approximated with an histogram. We can calculate the histogram based on quantiles or linear distribution as shown on figure 3.35. This probability density function is computed to establish the dissimilarity between the two sets of realizations.

Extraction of data with identical spatial localization.

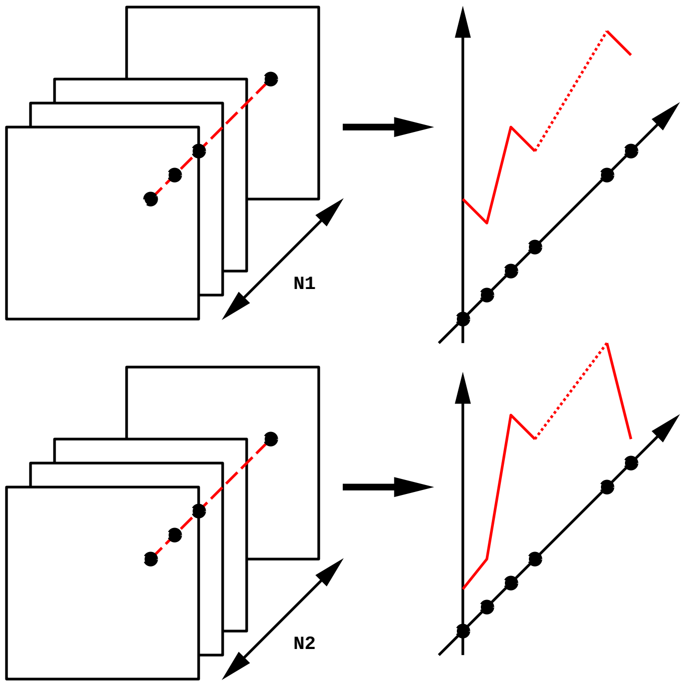


Figure 3.36: *Extraction of the two realizations sets for each pixel*

We have two different realizations of our data, baseline and activated. For each pixel (i, j), we create two sets corresponding to the N_1 and N_2 realizations, as seen on figure 3.36. These realizations do not have any temporal relations, and we are only interested in the statistical properties of our data. We then compute their respective histograms, using the measure computed previously during the preprocessing. For each subspace, the size and location of the bins (shown on figure 3.37) have been estimated previously using all the pixels of the subspace. Then we use the same bin sizes for every pixel of this subspace, related to the measure previously, to estimate the histogram of the time series. The different measure have been chosen globally, they depends on the pixel values of the whole subspace. In order

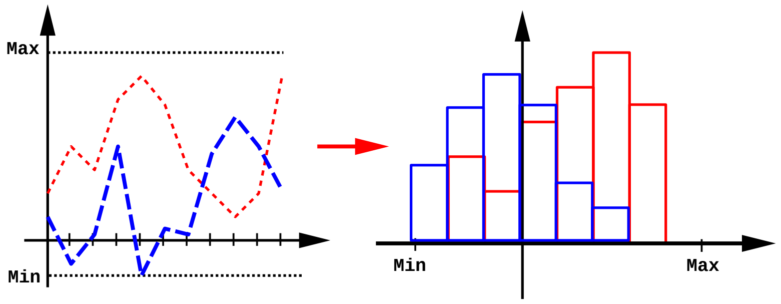


Figure 3.37: *Probability density function computed with the measure defined with the pre-processing*

to compare the two time series we compute their relative entropy. A “Cost subspace” is obtained by allocation to every pixel of this value.

Cost Computation and affectation of this value

For each pixel (i, j), we estimated the probability density functions using the two histograms. We decide to compare them by computing the differential entropy E as follow:

$$E(X_1, X_2) = \sum_{n=1}^{n=N} (p_n - q_n) \cdot \log(\frac{p_n}{q_n}) \geq 0 \qquad (3.66)$$

where p_n and q_n represent the probability of X_1 and X_2 in the interval I_n, with $[Min, Max] = \cup_{l=n}^{N} I_n$, for a given pixel (i, j). We compute the differential entropy using probability density functions and not the values. It give a better idea about how different are the probabilities, as seen on figure 3.38. We then affect this value cost to the spatial location (i, j) to create

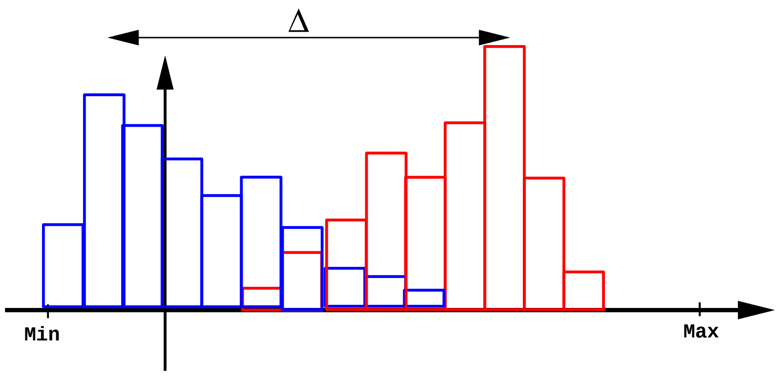


Figure 3.38: *Differential Entropy as an estimation of the discrimination of two probability density functions*

the cost subspace. For each level (or scale), we have obtained four "Cost subspaces" that enables us to reconstruct an image corresponding to the difference at this fixed level. This step is now follow by the reconstruction.

Reconstruction

For each decomposition level we have four subspaces (except at the pixel level). We rebuilt as many images as we have levels of decomposition, as represented on figure 3.39. Each reconstruction gives us a different scale for classification. We are then able to visualize the differences between our two data sets for different scales. We note on figure 3.39 by V_i the cost subspace at the level i. We then are able to multiply these result to visualize only the pixels seen as important at every scale. Since our cost function is positive, multiplying the various values is equivalent to a smooth version of the logic operator "AND".

3.5.3 Results

We present a result for the functional MRI data on figure 3.41. From these experiments, we note some important properties. First, we have consistency through the various scales. In fact, after only two levels the products obtained by this algorithm is almost the same.

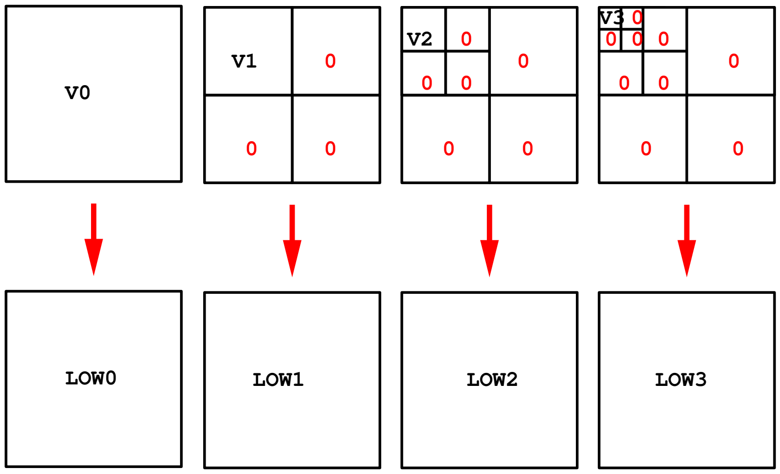


Figure 3.39: *Wavelet reconstruction from various levels*

We have a convergence of our algorithm through the scale which allows us to think that the result is stable. The second property is due to the multi-scale approach. We combine the result of different scales, we refine our result. By multiplying the various at different scales we go from coarse to fine, and so by iteration we discard the invalid “activated pixels” (informal conversation with Francois G. Meyer). For easier comprehension, we explain this property on a one dimensional example shown on figure 3.40.

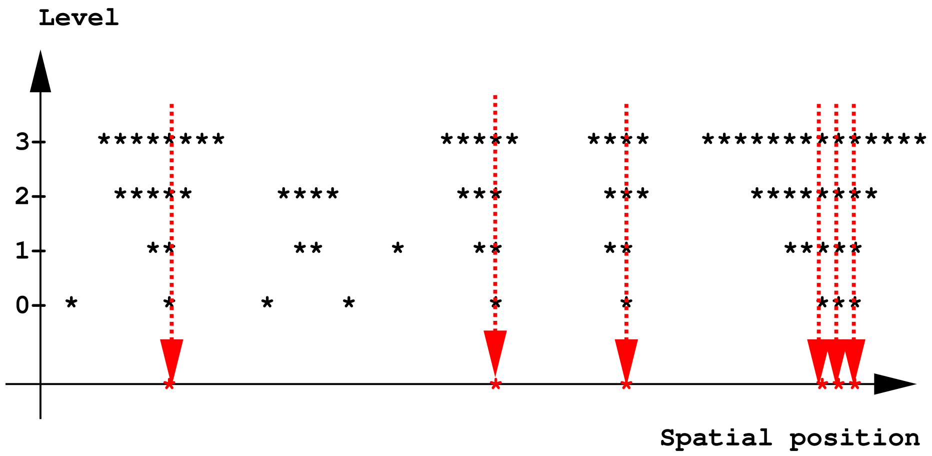


Figure 3.40: *Multi-scale property of the algorithm. At each scale, activation is detected. The product of the result of the various scales enable a better localization*

We observe on this result the consistency of the multi-resolution analysis. After two levels, the product of the reconstructed images does not have significant changes.

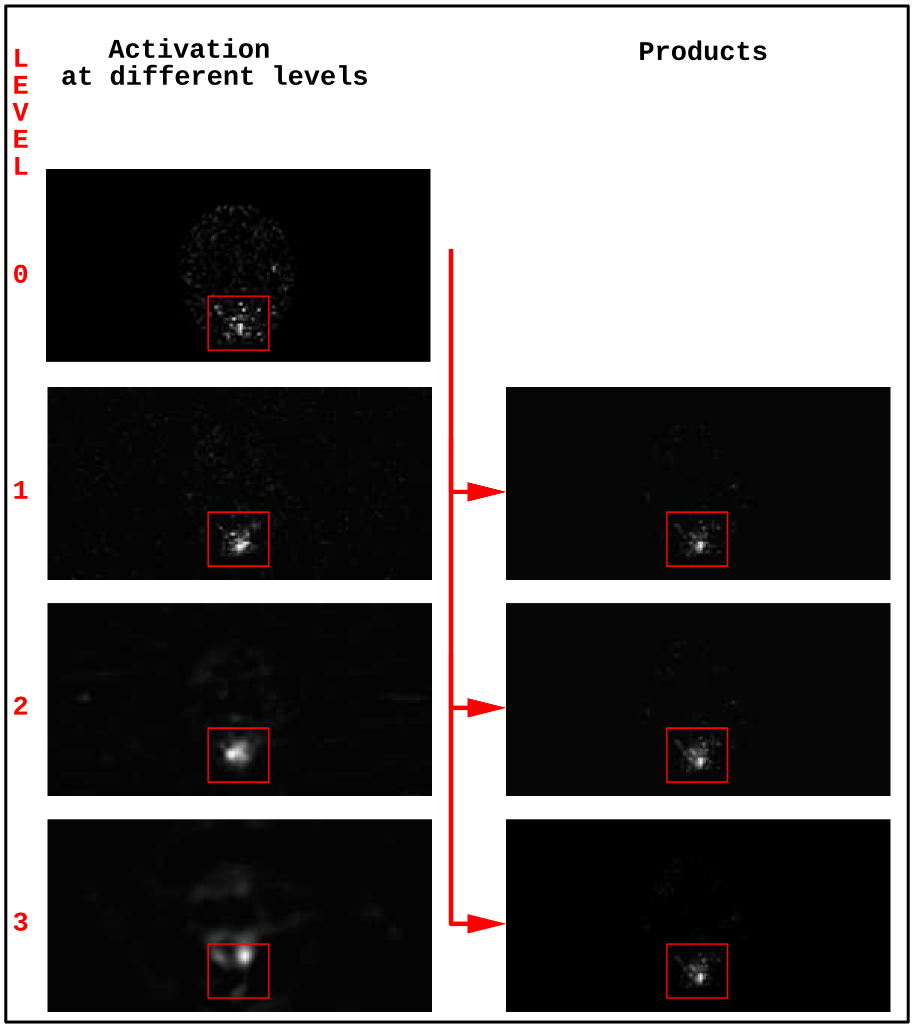


Figure 3.41: *Result of the brain activation detection at various scale. The left row shows the activation at level* 0 *(pixel level) to* 3*, while the left row shows the products of the various levels starting at* 0*.*

3.6 Conclusion

The three applications developed in our reports were based multi-resolution. Since we worked at different scale and not only at the pixel scale, we obtain a better analysis and result. We have shown in this report how to preprocess data with undecimated wavelets. Non-decimation is crucial as it gives grid independence as opposed to the classic decimated wavelets representation. Our preprocessing is followed by classical engineering methods. It also means that most of the steps, as the cost function or γ-correction are examples, of our algorithm can be modified. Improvement can be done in many directions since the various parts of the algorithms are independent.

Chapter 4

Conclusion

In this thesis, we have presented properties and results about the factorization representation based on the Blaschke Product. Stability to noise and invariance have been shown for the phase of the Blaschke product. The non-linear approximation gives good results and fast convergence. Further work have to be done to optimize the algorithm with the localized term $z \mapsto \frac{\sqrt{1-|\beta|^2}}{1-\bar{\beta}z}$. The two dimensional extension, that we develop, has properties of stability to noise to but many parameters, such as the partition unity of the Fourier space or the value of ϵ for the threshold, have to be tested to improve the algorithm. It also gives an alternative for computing phase for segmentation.

In the second part we have worked on segmentation using preprocessing based on the wavelet packets representation. the choice of a cost function and the addition of an extra term to the Mumford-Shah functional gave us the canonical number of regions, based on our criterion of regions of similar size, for segmentation. Efficient segmentation can then be obtained by keeping only the useful filters. By conception this algorithm is modular and then many improvement can be done on each bloc, as the choice of the cost function or the filters used in the preprocessing.

In the third part, we worked with mutli-resolution algorithms. We have shown application of the undecimated wavelets and more precisely how to create an equalizer for one and two dimensional signal that does not suffer of artifacts. An implementation on Matlab has been started but more can be done to make the tool more operational. The detection of brain activities has good results as it uses standard engineering techniques with the multi-scale analysis. Improvement can be done at various steps since the algorithm is also modular.

Bibliography

References for "Phase Evaluation and Blaschke Product"

[1] D. Gabor, *"Theory of Communication,"* Proc. IEE, vol. 93,pp 429-457,1946.

[2] J. B. Garnett, *"Bounded Analytic Functions,"* Academic Press 1981

[3] R. Kumaresan and A. Rao, *"Model-based approach to envelope and positive instantaneous frequency estimation of signals with speech applications,"* J. Acoust. Soc. Am. 105 (3), March 1999, pp. 1912-1924

[4] P. J. Loughlin and B. Tarcer, *"On the amplitude- and frequency- modulation decomposition of signals,"* Journal Acoustical society AM 100: (3) 1594-1601 sept 1996

[5] F. Meyer, R. Coifman *"Brushlets: A tool for directional image analysis and image compression,"* Applied and Computational Harmonical Analysis, 4:(2), page 147-187, Apr 1997

[6] A.V. Oppenheim and R.W. Schafer, *"Discrete-Time Signal Processing,"* Prentice-Hall, Englewood Cliffs, NJ, 1989.

[7] Z. Szabo, J. Bokor, and F. Schipp, *"Identification of Rational Approximate Models in H^∞ Using Generalized Orthonormal Basis,"* IEEE Transactions of Automatic Control, Vol. 44, No. 1,pp. 153-158, Jan. 1999.

[8] H.B. Voelcker, *"Towards a unified theory of modulation part I: Phase-envelope relationships,"* Proc. IEEE 54, 340-354 (1966).

[9] H. B. Voelcker, *"Towards a unified theory of modulation part II: Phase-envelope relationships,"* Proc. IEEE 54(5), 735-755 (1966).

[10] J. Walden, *"The Pseudopolar FFT and its applications,"* Research report YALEU/DCS/RR-1178, May 1999.

[11] J. L. Walsh, *"Interpolation and Approximation by Rational Functions in the Complex Domain,"* American Mathematical Society, Colloquium Publications Vol. XX 5th edition 1969

[12] A. Zygmund, *"Trigonometric Series,"* Cambridge Univ. Press 1959.

References for "A Vectorial Segmentation Algorithm for Textural Images"

[13] P. Brodatz *"Textures for artists and designers,"* Dovers Publications Inc. NY 1966

[14] R. Coifman and M. Wickerhauser *"Best-adapted wavelet packet bases,"* preprint Yale Univ., Feb. 1990.

[15] J. Froment *"Megawave2 System Library,"* CEREMADE, University Paris IX-Dauphine, May 1996

[16] R. Guglielmi *"Wavelet Feature Definition and Extraction For Classification and Image Processing,"* Mathematics Department, Yale University, New haven, May 1997

[17] B. Julesz *"Textons, the elements of texture perception, and their interactions,"* Nature, 290, 12 March 1981

[18] G. Koepfler *"Formalisation et Analyse Numerique de la Segmentation d'Images,"* Ph.D. Thesis, University Paris IX-Dauphine, 1991

[19] G. Koepfler, C. Lopez and J.-M. Morel *"A Multiscale Algorithm for Image Segmentation by Variational Method,"* SIAM JOURNAL On Numerical Analysis, 31:(1), page 282-299, Feb 1994

[20] J. Malik and P. Perona *"Preattentive texture discrimination with early vision mechanisms,"* Journal of the Optical Society of America A, 923-932, volume 7, n 5, 1991.

[21] S. Mallat *"A theory for multiresolution signal decomposition: the wavelet representation,"* IEEE Transaction on Pattern Analysis and Machine Intelligence, page 674-693, July 1989.

[22] S. Mallat *"A wavelet tour of signal processing,"* Academic Press 1998.

[23] F. Meyer, R. Coifman *"Brushlets: A tool for directional image analysis and image compression,"* Applied and Computational Harmonical Analysis, 4:(2), page 147-187, Apr 1997

[24] J.-M. Morel and S. Solimini *"Variational Methods on Image Segmentation,"* Birkhauser, 1995

[25] D. Mumford and J. Shah *"Boundary detection by minimizing functionals,"* IEEE Conference on Computer Vision and Pattern Recognition, San Franscico 1985

References for "Undecimated Wavelets and Applications"

[26] R. Coifman and M. Wickerhauser *"Best-adapted Wavelet Packet Bases,"* preprint Yale Univ., Feb. 1990.

[27] R. Guglielmi *"Wavelet Feature Definition and Extraction For Classification and Image Processing,"* Mathematics Department, Yale University, New Haven, May 1997

[28] S. Mallat *" A Theory for Multi-resolution Signal Decomposition: the Wavelet Representation,"* IEEE Trans. Patt. Recogn. and Mach. Intell., 11(7):pp674-693, July 1989

[29] S. Mallat *"A Wavelet Tour of Signal Processing,"* Academic Press 1998.

[30] L. Woog *"Adapted Waveform Algorithms for Denoising,"* Ph.D. Thesis, Computer Science, Yale University, New Haven, May 1997.