INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

University of Alberta

Positive Wavelet Packet Transforms Theory

by

Jorge Pulido Salcedo

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

Department of Electrical and Computer Engineering

Edmonton, Alberta
Spring 2005

Happiness is like a butterfly. When you pursue it, it is always beyond your grasp. But when you sit down, it may alight upon you.

Nathaniel Hawthorne

To

my wife, my mom and the rest of my family. Without them I would not be here.

Abstract

In this project problems related to wavelet applications are studied. An analysis of the behaviour of a chain of decimators when its input is white Gaussian noise on the assumption of finite duration input signals is presented. A structure for the output correlation matrix of the system is obtained. In addition, an upper and lower bound for the unitary norms of the output correlation matrix are given. These bounds will provide information about the inverse of the output correlation matrix with respect to its condition number. After this characterization is done, the creation of a positive wavelet projection into wavelet subspaces to represent positive signals is shown. The positive wavelet projections are based on the creation of positive kernels as shown by Walter [1]. This theory is applied and simplified in order to obtain computationally feasible positive projections which can be applied in real life applications. Finally, an application where a positive wavelet projection is used to improve the computational speed of a QR factorization algorithm is showed.

Acknowledgements

The author wishes to express most sincere appreciation to Dr. Christopher J. Zarowski. He guided me through my research providing comments, suggestions and support in all the aspects he could. In addition I want to thank Dr. Behrouz Nowrouzian for their assistance, guidance and wise advices in the preparation of this manuscript and the final defense. Thanks also to Dr. Xiaoping Shen for the support given to comprehend her theory.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

For a long time the Fourier transform has been widely employed in engineering applications. However, the Fourier transform presents problems in some applications such as the processing of non-stationary signals.

In recent years the wavelet transform has become an alternative to the Fourier transform. The wavelet transform has already been used in problems like signal denoising, signal compression and signal detection with satisfactory results. Due to this fact, there is a lot of interest in using the wavelet transform in other applications. However, the wavelet transform theory remains under current development, and there are some parts of the theory that have not been completely developed.

In this thesis we are particularly interested in a group of wavelets known as the compact support wavelets. Generally we use the Daubechies [2] wavelet family which is the most well known compact support wavelet family.

We study two problems related to wavelet applications. The first one is the behaviour of a chain of decimators when its input is white Gaussian noise. The decimation operation is one of the basic operations to compute the coefficients of a wavelet projection. In many applications (i.e. signal compression , signal prediction) the signals to be processed with the wavelet transform are affected by noise. Hence, the behaviour of the system is modified by the presence of noise. The characterization of decimator outputs when the input is noise improves the results obtained in these applications.

There have been previous attempts to explain the behaviour of decimators when the input is noise [3] [4] [5]. However, previous authors made assumptions about the system that are not always common in actual applications. In Chapter 6 we present the characterization of a chain of decimators when the input is white Gaussian noise on the assumption of finite duration input signals, in contrast with previous works which as-

sume signals of infinite duration. We give a structure for the output correlation matrix of the system. This matrix can be used to predict the statistical properties of a noisy wavelet-based multirate system. In addition, we give an upper and a lower bound for the unitary norms of the output correlation matrix. These bounds provide information about the inverse of the output correlation matrix with respect to its condition number.

The second problem to be analyzed is the projection of positive signals into wavelet subspaces. When a positive signal is projected we would want the projection coefficients to be positive. Furthermore, if the input signal presents discontinuities we want to reconstruct the original signal from the projection coefficients without Gibbs' phenomenon. This is not normally the case for either the Fourier or wavelet projections. There are methods to modify the Fourier projection in such a way that these problems are avoided [1]. There have also been attempts to solve these problems for wavelet projections [6] [7], but they use their theory to denoise signals and do not present a mathematical explanation to support the elimination of the Gibbs' phenomenon. Walter [1] also presents a theory to solve these problems. He gives a complete mathematical explanation to support his findings. However, the processing of signals using Walter's theory present some computational problems. In Chapter 7 we reconsider the theory given by Walter and modify it in such a way that these problems are eliminated.
Finally in Chapter 8 we will present an application where we use the positive wavelet projection of a positive input signal combined with the minimum description length criterion of Rissanen to determine the rank of a matrix.

In order to understand the theory given in Chapters 6, 7, and 8 we give a review of the theory of function projections and Gibbs' phenomenon in Chapters 2 and 3 , orthogonal wavelet projections in Chapter 4 , and the minimum description length criterion in Chapter 5.

Chapter 2

Function Sequence Convergence and Series

2.1 Function Spaces

The mathematical representation of a certain physical phenomenon is called a signal. A signal will contain certain characteristic information about the represented phenomena and it is defined as a function of this information. Furthermore the functions used to represent a signal possess common mathematical characteristics. Hence we can group these functions into collections. These sets are called function spaces and the area of mathematics that studies the properties of function spaces is called functional analysis [8].
In this thesis we deal with some of these spaces. They are the $C^{(p)}$ differentiable functions, the integrable functions L^1 and the square integrable functions L^2.

Definition 2.1.1 ($C^{(p)}$ **Differentiable Functions).** Given a number $p \in \mathbb{N}$ [1] , we say that a function $f(t)$ defined on an interval of length T located anywhere is $C^{(p)}$ if it is p times differentiable on the interval T and the p^{th} derivative is also continuous.

Definition 2.1.2 ($L^1(T)$ **Space).** A piecewise continuous function $f(t)$ on an interval T is a $L^1(T)$ function if

$$\int_T |f(t)|dt < \infty.$$

Definition 2.1.3 ($L^2(T)$ **Space).** A piecewise continuous function $f(t)$ on an interval

[1] N is the set of natural numbers.

T is a $L^2(T)$ function if

$$\int_T |f(t)|^2 dt < \infty.$$

The L^2 inner product is given by

$$< f, g >= \int_T f(t)g^*(t)dt,$$

where $g^*(t)$ is the complex conjugate of the function $g(t)$. The L^2 norm is derived from the inner product and it is defined by

$$||f||_2 =< f, f >^{\frac{1}{2}} = \left(\int_T |f(t)|^2 dt\right)^{\frac{1}{2}}.$$

Note that these sets are not mutually exclusive, so then there are functions that can belong to two or more of these sets at the same time.
The norm operator in L^2 provides the space with a function to measure the size of its elements. Furthermore using the norm we can construct an operator to measure the distance between two functions. This operator is called a metric and it is given by:

$$d(x, y) = ||x - y||,$$

where x, y are elements of the function space.

2.2 Sequences

It is possible to form a sequence of elements in a space using an iterative process. These sequences usually have practical value only if there is a limiting element when the number of iterations goes to infinity. This concept is referred as the convergence of a sequence of functions.

Definition 2.2.1 (Convergence). A sequence of functions $x_n(t)$ with $n \in \mathbb{Z}$ [2] inside a space X converges if and only if there is a function $x(t) \in X$ such that

$$\lim_{n\to\infty} d(x_n(t), x(t)) = 0,$$

where $d(x, y)$ is a suitable metric on X, for $x, y \in X$. We say that

$$\lim_{n\to\infty} x_n = x.$$

[2] $\mathbb{Z}$ is the set of integer numbers.

This definition of convergence is very general, and it only limits itself to show the stability of the iterations. We can extend the definition of convergence of a sequence to account for some extra characteristics of the iteration process. We will introduce two particular cases of convergence.

Definition 2.2.2 (Pointwise Convergence). Given a sequence of functions $x_n(t)$ we say the sequence converges pointwise if and only if there is an $x(t)$ so that for all $\epsilon > 0$ there is an $N = N(\epsilon, t)$ such that

$$|x_n(t) - x(t)| \leq \epsilon$$

for $n \geq N$. We say x_n tends pointwise to x.

It is very important to note that the value N used in this definition can depend on both ϵ and t. This dependence implies that under this definition it is possible to find sequences where although the distance from a sequence element to the limit function tends to zero the error of approximation is not the same for all the points t.

Example 2.2.1. The function

$$x_n(t) = \begin{cases} 2nt & \text{if } t \in [0, \frac{1}{2n}) \\ 2 - 2nt & \text{if } t \in [\frac{1}{2n}, \frac{1}{n}) \\ 0 & \text{if } t \in [\frac{1}{n}, 1]. \end{cases}$$

belongs to $L_1(0,1]$ and $L_2(0,1]$. This function converges pointwise to the function $x(t) = 0$, and so we say $x_n(t) \to 0$ pointwise.
The behavior of this sequence can be seen in Figure 2.1. Note how as n increases most of the points t from the sequence element x_n tend to zero. However as the gray area in Figure 2.1 shows there are some points that are not going to zero.

Definition 2.2.3 (Uniform Convergence). Given a sequence of functions $x_n(t)$ we say the sequence converges uniformly if and only if there is an $x(t)$ so that for all $\epsilon > 0$ there is an $N = N(\epsilon)$ such that

$$|x_n(t) - x(t)| \leq \epsilon$$

for $n \geq N$. We say x_n converges uniformly to x.

It can be clearly seen that the difference between uniform and pointwise convergence

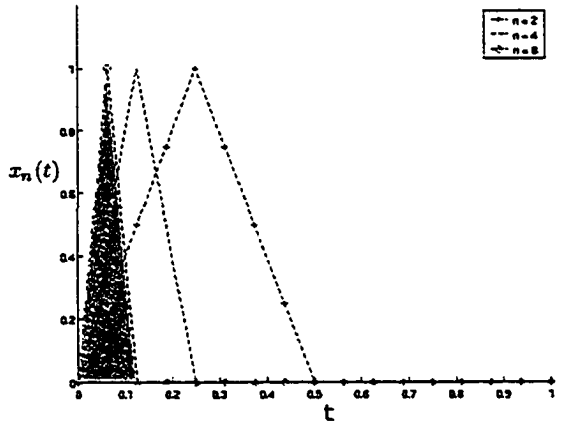

Figure 2.1: Sequence of functions with pointwise convergence

is the dependence of N or otherwise on the variable t. Moreover, due to this fact a uniformly convergent sequence is also pointwise convergent but the converse is not true.

Example 2.2.2. The function

$$x_n(t) = \frac{t}{1+nt^2}$$

belongs to $L_2(\mathbb{R})$ [3]. This function converges uniformly to the function $x(t) = 0$, and so we say $x_n(t) \to 0$ uniformly.
The behavior of this sequence can be seen in Figure 2.2. The gray area shows how as n increases all of the points t of x_n tend to the limit function.

2.3 Orthonormal Series Expansion

Using the inner product given in Definition 2.1.3 we can say a set of functions $(e_1 \dots e_n) \in L^2$ is an orthonormal set if and only if

$$< e_n, e_m >= \begin{cases} 0 & \text{if } m \neq n \\ 1 & \text{if } m = n, \end{cases}$$

for every e_m, e_n. The orthonormality between two functions e_n and e_m is often symbolized as $e_n \perp e_m$.

[3] $\mathbb{R}$ is the set of real numbers.

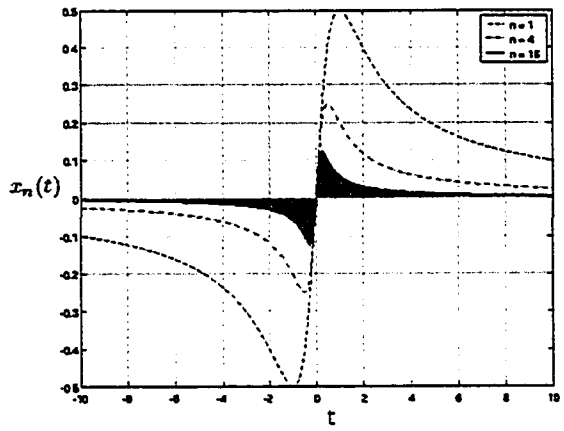

Figure 2.2: Sequence of functions with uniform convergence

It is easy to prove that the members of an orthonormal set are linearly independent. Using this fact a linear combination of orthonormal functions $[e_1 \dots e_n]$ will be an element of some subspace $S \in L^2$ that they span. Hence we can say span $[e_1 \dots e_n] = S$.
In addition, as it is shown by Kreyszig [8], a function $x \in L^2$ can be projected into the space S as

$$P_S x = \sum_{k=1}^{n} \underbrace{< x, e_k >}_{\alpha_k} e_k. \tag{2.1}$$

Note that if $x \in S$ then $P_S x = x$ and then the projection is called an orthonormal series expansion.
We can extend the concept of series expansions to the case where we have an infinite number of orthonormal functions $(\dots, e_{-1}, e_0, e_1, \dots)$ [9].
The projection of the signal using an infinite number of orthonormal functions is

$$P_S x = \sum_{n=-\infty}^{\infty} \underbrace{< x, e_n >}_{\alpha_n} e_n. \tag{2.2}$$

This projection can be seen as the limit of the sequence

$$x_m = \sum_{n=-m}^{m} < x, e_n > e_n,$$

when $m \rightarrow \infty$. Furthermore in the case $x \in \text{span}(\ldots, e_{-1}, e_0, e_1, \ldots)$ we expect

$$\lim_{m \rightarrow \infty} x_m = x.$$

Note the projection is based on a sequence and then we have to assure the sequence converges.

The convergence of the sequence depends on the characteristics of the function to be approximated. For some functions the series expansion converges uniformly, but for others pointwise, and for others the series does even converge at all.

In the orthonormal projections used in this thesis the coefficients α_n decrease as $|n| \rightarrow \infty$, and for this reason the projection can be approximately computed using a truncated version of the sum. The error introduced is known as the truncation error and can be decreased to a desirable value by increasing the number of terms used to compute the projection.

2.4 Conclusion

In this Chapter, the concept of a function space and a sequence has been introduced. The convergence of a sequence and two different types of convergence have been explained. In addition, the use of a set of orthonormal functions to obtain a sequence which give us a series expansion has been shown.

Chapter 3

Delta sequences and Fourier Series

3.1 Reproducing Kernel

As it was seen in Section 2.3 when a function is represented by means of infinite series expansions first we need to check if the series converges at all. Furthermore if the series converges then we need to find what type of convergence the series possesses. Checking the convergence of a series using Definitions 2.2.1,2.2.2 and 2.2.3 directly can be difficult. For this reason alternative methods have been developed. One of these methods uses the concept of the reproducing kernel.
If we truncate the series expansion for $x(t)$ given in Equation (2.2) we obtain

$$\begin{aligned} x_m(p) &= \sum_{n=-m}^{m} \langle x(t), e_n(t)\rangle e_n(p) \\ &= \sum_{n=-m}^{m} \left\{ \int_{-\infty}^{\infty} x(t) e_n(t)\, dt \right\} e_n(p) \\ &= \int_{-\infty}^{\infty} x(t) \underbrace{\left\{ \sum_{n=-m}^{m} e_n(t) e_n(p) \right\}}_{\delta_m(t,p)} dt \\ &= \int_{-\infty}^{\infty} \delta_m(t,p) x(t) dt \end{aligned} \tag{3.1}$$

where $\delta_m(t,p)$ is known as the reproducing kernel sequence for the space S using the orthonormal set $(\ldots, e_{-1}, e_0, e_1, \ldots)$.

Through Equation (3.1) we can note there is a resemblance between the function $\delta_m(t,p)$ inside the space S as $m \to \infty$ and the Dirac's delta function δ defined on the real number line. Due to this reason the reproducing kernel sequence is also known as a delta sequence.
Further analysis of the similarities between the Dirac's delta function and the delta sequences gives rise to two different types of delta sequences.

3.1.1 Quasi-positive delta sequences

A set of sequences $\delta_m(\cdot,t) \in L^1(\mathbb{R})$ is called a quasi-positive delta sequence if

$$\text{There is a } C > 0 \text{ such that } \int_{-\infty}^{\infty} |\delta_m(p,t)|dp \leq C,\ t \in \mathbb{R},\ m \in \mathbb{Z}^+.^1 \tag{3.2i}$$

$$\int_{-\infty}^{\infty} \delta_m(p,t)dp \to 1 \text{ uniformly in compact } ^2\text{subsets of } \mathbb{R} \text{ as } m \to \infty. \tag{3.2ii}$$

$$\text{For each } \gamma > 0,\ \sup_{|t-p|\geq\gamma} \delta_m(p,t) \to 0 \text{ as } m \to \infty, \tag{3.2iii}$$

where $\sup_{|t-p|\geq\gamma} \delta_m(p,t)$ denotes the maximum value of $\delta_m(p,t)$ on the region given by $|t-p| \geq \gamma$. The first property guarantees the delta sequence is bounded in magnitude for all values of m. This bound is made with respect to the absolute value of the delta sequence, making possible for the delta sequence to have negative values. This behaviour is different than the one presented by the Dirac's delta function, but as it will be seen later it is common to find orthonormal projections whose delta sequences have this behavior.
The second property ensures that the integral value for the delta sequence goes to one as $m \to \infty$, which is the same value possessed by the integral of the Dirac's delta function. The third property ensures that the delta sequence is concentrated in a small interval γ around t. Ideally we want to concentrate all the values of a delta sequence on a single point as $m \to \infty$.

3.1.2 Positive delta sequences

A positive delta sequence is a sequence where condition (3.2i) is replaced by

$$\delta_m(x,y) \geq 0, \quad x,y \in \mathbb{R} \tag{3.3}$$

[1] $\mathbb{Z}^+$ is the subset of nonnegative integers.
[2] A compact subset of $\mathbb{R}$ is a closed and bounded subset of real numbers.

This new condition makes these sequences more similar to Dirac's delta function than the quasi-positive sequences.
Equation(3.3) combined with (3.2iii) makes every positive delta sequence a quasi-positive delta sequence. However the converse is not necessarily true.

3.2 Fourier Series

The Fourier series is an infinite orthonormal series. The orthogonal functions that define the series are a basis for the space $L^2[-\pi, \pi]$.
This series is created with the orthonormal set of functions

$$e_n = e^{int} = \cos(nt) + i\sin(nt),$$

for $n \in \mathbb{Z}$.
The inner product $< x, e_n >$ for this system is defined as

$$< x, e_n >= \frac{1}{2\pi}\int_{-\pi}^{\pi} x(t)e^{-int}dt.^3$$

Using this inner product the series given by Equation (2.2) becomes

$$\begin{aligned} Px(t) &= \sum_{n=-\infty}^{\infty} \langle x(p), e_n(p)\rangle e_n(t) \\ &= \frac{1}{2\pi}\sum_{n=-\infty}^{\infty}\left\{\int_{-\pi}^{\pi} x(p)e^{-inp}dp\right\}e^{int} \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} x(p)\left\{\sum_{n=-\infty}^{\infty} e^{-inp}e^{int}\right\}dp \\ &= \int_{-\pi}^{\pi} x(p)\underbrace{\left\{\frac{1}{2\pi}\sum_{n=-\infty}^{\infty} e^{-in(p-t)}\right\}}_{\delta_\infty(p,t)}dp. \end{aligned} \tag{3.4}$$

[3]The reader will note that the inner product used in the Fourier series is different from the one we previously defined by a factor $\frac{1}{2\pi}$. We will limit ourselves to say this inner product is valid. For further discussion on the subject see Kreyszig [8].

Equation (3.4) shows the kernel sequence for the Fourier series is

$$\begin{aligned}
\delta_m(p,t) &= \frac{1}{2\pi}\sum_{n=-m}^{m} e^{-in(p-t)} \\
&= \frac{1}{2\pi}\left(1+\sum_{n=1}^{m} e^{-in(p-t)} + \sum_{n=-m}^{-1} e^{-in(p-t)}\right) \\
&= \frac{1}{2\pi}\left(1+\sum_{n=1}^{m} e^{-in(p-t)} + \sum_{n=1}^{m} e^{in(p-t)}\right) \\
&= \frac{1}{2\pi}\left(1+\sum_{n=1}^{m} \left(e^{-in(p-t)} + e^{in(p-t)}\right)\right) \\
&= \frac{1}{2\pi} + \frac{1}{\pi}\sum_{n=1}^{m} \frac{e^{-in(p-t)} + e^{in(p-t)}}{2} \\
&= \frac{1}{2\pi} + \frac{1}{\pi}\sum_{n=1}^{m} \cos n(p-t).
\end{aligned} \tag{3.5}$$

This kernel is known as the Dirichlet kernel and it is often represented as $D_m(p-t)$. In the case of the Dirichlet kernel as in many other kernel sequences the behavior of the delta sequence $\delta_m(p,t)$ depends on the time difference between t and p. Due to this reason in this case the delta sequence is analyzed with the term $\delta_m(k)$ where $k = p - t$. For the Dirichlet kernel it can be shown that

$$\begin{aligned}
D_m(t) &= \frac{1}{2\pi} + \frac{1}{\pi}\sum_{n=1}^{m} \cos nt \\
&= \frac{\sin(m+\frac{1}{2})t}{2\pi \sin\frac{t}{2}} \\
&= \frac{1}{2\pi}\left(\frac{\sin mt \cos\frac{t}{2}}{\sin\frac{t}{2}} + \cos mt\right).
\end{aligned}$$

This kernel can be seen in Figure 3.1. We can check Property 3.2ii computing the integral of the Dirichlet kernel and obtaining

$$\begin{aligned}
&\int_{-\infty}^{\infty} \delta_m(p,t)dp \\
&= \int_{-\pi}^{\pi} D_m(p,t)dp \\
&= \int_{-\pi}^{\pi} \left(\frac{1}{2\pi} + \frac{1}{\pi} \sum_{n=1}^{m} \cos n(p-t) \right) dp \qquad (3.6) \\
&= \frac{2\pi}{2\pi} + \frac{1}{\pi} \sum_{n=1}^{m} \frac{\sin n(p-t)}{n} \Big|_{p=-\pi}^{\pi} \\
&= 1.
\end{aligned}$$

Notice Equation (3.6) tells us the integral for the Dirichlet kernel is equal to one for all the values of m. Inspecting Figure 3.1 we notice the peak value of $\delta_m(p-t)$ increases when m increases. Notice also that the majority of non-zero values for the delta sequence are concentrated around the peak value as m increases. Then intuitively Property 3.2iii is satisfied. Note however this kernel is neither positive or quasi-positive.

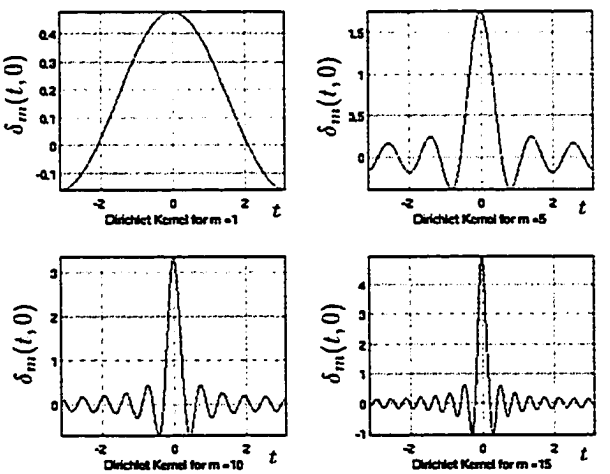

Figure 3.1: Dirichlet kernel for various values of m.

It is of interest to find the set of functions whose Fourier series converges and what type of convergence they present. Over the years various kinds of these sets have been

found.
One of these sets is the group of 2π-periodic functions that have a Lipschitz continuity condition.

Definition 3.2.1 (Lipschitz condition). A function $x(t) \in L^1$ satisfies the Lipschitz condition of order $\alpha > 0$ at t_0 if there is a constant C such that

$$|x(t) - x(t_0)| \leq C|t - t_0|^\alpha$$

for $|t - t_0| < \epsilon$, where α and C do not depend on ϵ.

It has been shown [1] that the functions $x(t) \in L^1(-\pi, \pi)$ that satisfy the Lipschitz condition converge pointwise in accordance with

$$Px(t_0) \rightarrow \frac{x(t_0+) + x(t_0-)}{2} = \overline{x}(t_0)$$

as $n \rightarrow \infty$. [4] If the function $x(t)$ is a continuous function that satisfies the Lipschitz condition, then $Px(t_0) \rightarrow x(t_0)$ as $n \rightarrow \infty$. On the other hand, if the function possesses a discontinuity, the Fourier series converges to the average of the right and left limit values of the function at the discontinuity.

3.3 Gibbs' Phenomenon

The Gibbs' phenomenon is the anomaly present in some orthonormal series when they try to approximate a piecewise continuous function with a countable number of discontinuities. The Gibbs phenomenon was first observed by Michelson [10] in 1898 when he build a machine to compute the Fourier series of a function. However it was Gibbs [11] in 1899 who explained the problem.
The phenomenon appears due to the lack of uniform convergence of the projection of piecewise continuous functions. The lack of uniform convergence does not at all imply a lack of some type of convergence in the projection. Generally, the orthonormal projection converges pointwise at the discontinuity. In spite of this convergence the value obtained using the projection presents an overshoot near the discontinuity that does not decrease as the number of terms in the sum used to compute the projection goes to infinity.
This overshoot introduces undesired effects in the projection. One of these effects is

[4] $x(t_0+)$ is the right limit value of the function $x(t)$ as $t \rightarrow t_0$. Similarly $x(t_0-)$ is the left limit value of the function $x(t)$ as $t \rightarrow t_0$.

the introduction of negative values in the projection of a positive function. This is undesirable in the approximation of non-negative functions such as gray-scale images and probability density functions. Also, "ringing artifacts " can appear in series-based signal compression schemes as a result of the Gibbs' phenomenon.

3.3.1 Gibbs' Phenomenon in Fourier series

We will show the Gibbs' phenomenon in Fourier series using an example extracted from [1]. Consider the function

$$x(t) = \begin{cases} \frac{\pi}{2} - t/2 & \text{if } 0 \leq t < \pi \\ -\frac{\pi}{2} - t/2 & \text{if } -\pi < t < 0. \end{cases}$$

This function is known as the sawtooth function, and it satisfies the Lipschitz condition at the point $t = 0$. Its Fourier series is

$$\begin{aligned} Px(t) &= \sum_{n=1}^{\infty} \frac{\sin nt}{n} \\ &= \sum_{n=1}^{\infty} \int_0^t \cos nsds \\ &= \int_0^t \sum_{n=1}^{\infty} \cos nsds \\ &= \int_0^t \left[\frac{1}{2} + \sum_{n=1}^{\infty} \cos ns - \frac{1}{2} \right] ds \\ &= \int_0^t \left[\frac{1}{2} + \sum_{n=1}^{\infty} \cos ns \right] ds - \frac{t}{2} \\ &= \lim_{n \to \infty} \pi \int_0^t D_n(s) ds - \frac{t}{2}. \end{aligned} \tag{3.7}$$

We are interested in finding the value of the projection close to the discontinuity point $t = 0$ when $n \to \infty$. When t is close to zero the term $\frac{t}{2}$ goes to zero and we only need to analyze the term

$$\pi \int_0^t D_n(s) ds.$$

Expanding this term we obtain

$$\begin{aligned}
&\pi \int_0^t D_n(s)ds \\
&= \int_0^t \frac{1}{2}\left(\frac{\sin nt \cos\frac{t}{2}}{\sin\frac{t}{2}} + \cos nt\right) ds \\
&= \int_0^t \frac{\sin nt}{t} dt + \int_0^t \sin nt \left(\frac{\cos(\frac{t}{2})}{2\sin(\frac{t}{2})} - \frac{1}{t}\right) dt + \int_0^t \frac{1}{2}\cos nt dt.
\end{aligned} \tag{3.8}$$

The second and third integral go to zero as $n \to \infty$, leaving only the first integral. Through a change of variable we obtain

$$\begin{aligned}
I(nt) &= \int_0^t \frac{\sin nt}{t} dt \\
&= \int_0^{nt} \frac{\sin t}{t} dt.
\end{aligned} \tag{3.9}$$

Using this Equation we note

$$Px(t) \approx I(nt).$$

If we choose $t = \frac{\pi}{n}$, then t goes to zero as $n \to \infty$ so

$$\frac{Px(0+)}{x(0+)} \equiv \frac{I(\pi)}{\pi/2} > 1,$$

and there is an overshoot in the projection as $t \to 0$. The Gibbs' phenomenon overshoot effect for the sawtooth Fourier series representation can be seen in Figure 3.2.

3.3.2 Gibbs' phenomenon for positive delta sequences

As we have seen in previous sections the Gibbs' phenomenon has a close relation with the convergence of the projection of a function. For this reason it is of interest to find what kind of convergence a positive delta sequence possesses. The following theorem is given by Walter [1]. However, he does not provide a complete proof of the theorem. Due to this reason we present a proof here.

Theorem 3.3.1. *Given a function $x(t) \in L^1(\mathbb{R}) \cap L^2(\mathbb{R})$, a positive delta sequence $\delta_m(p,t)$ which spans a space S and a projection $x_m(t)$ of the function $x(t)$ into the space S,*

i. If $M_1 \leq x(t) \leq M_2$ for $t \in \mathbb{R}$, then $M_1 \leq x_m(t) \leq M_2$ for $t \in \mathbb{R}$ and $m \in \mathbb{Z}$,

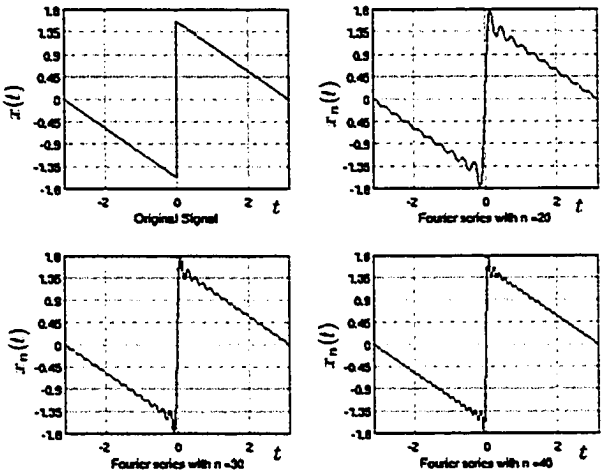

Figure 3.2: Sawtooth function and its Fourier series for various values of n.

ii. *If* $M_3 \leq x(t) \leq M_4$ *for* $t \in [a,b]$, *then for each* $\epsilon > 0, \eta > 0$ *there is an* m_0 *such that for* $t \in (a+\eta, b-\eta)$, $M_3 - \epsilon \leq x_m(t) \leq M_4 + \epsilon$, *for* $m \geq m_0$, *and hence*

$$x_m(t) \to x(t) \text{ uniformly as } m \to \infty$$

Proof. i. Recalling property (3.2ii), we note that for all $\varepsilon > 0$ there is a number $N = N(\varepsilon)$ such that

$$\left| \int_{-\infty}^{\infty} \delta_m(p,t)dp - 1 \right| \leq \varepsilon$$

for all $m \geq N$ and t on compact subsets of $\mathbb{R}$. This equation can be expressed as

$$1 - \varepsilon \leq \int_{-\infty}^{\infty} \delta_m(p,t)dp \leq 1 + \varepsilon.$$

Using this inequality and the function $x(t)$ bounds in the projection of the function $x(t)$ we obtain

$$\begin{aligned} x_m(t) &= \int_{-\infty}^{\infty} \delta_m(p,t)x(p)dp \\ &\leq \int_{-\infty}^{\infty} \delta_m(p,t)M_2 dp \\ &\leq M_2 \int_{-\infty}^{\infty} \delta_m(p,t)dp \\ &\leq M_2(1+\varepsilon). \end{aligned}$$

In a similar fashion,

$$\begin{aligned} x_m(t) &= \int_{-\infty}^{\infty} \delta_m(p,t)x(p)dp \\ &\geq \int_{-\infty}^{\infty} \delta_m(p,t)M_1 dp \\ &\geq M_1 \int_{-\infty}^{\infty} \delta_m(p,t)dp \\ &\geq M_1(1-\varepsilon). \end{aligned}$$

Then,

$$M_1 \leq x(t) \leq M_2,$$

as $\varepsilon \to 0$.

ii. For $t \in (a+\eta, b-\eta)$ we have

$$\begin{aligned} x_m(t) &= \int_{-\infty}^{\infty} \delta_m(p,t)x(p)dp \\ &= \left\{ \int_{-\infty}^{a+\eta} \int_{a+\eta}^{b-\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t)x(p)dp \\ &= \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t)x(p)dp + \int_{a+\eta}^{b-\eta} \delta_m(p,t)x(p)dp^{5} \\ &\leq \underbrace{\left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t)x(p)dp}_{I_3} + \underbrace{M_4 \int_{a+\eta}^{b-\eta} \delta_m(p,t)dp}_{I_4}. \end{aligned}$$

By properties 3.3 and 3.2ii we know

$$0 \leq \int_{a+\eta}^{b-\eta} \delta_m(p,t)dp \leq 1,$$

then $I_4 \leq M_4$. Since $x(t) \in L^1(\mathbb{R})$ we can find a real number U such that

$$\int_{-\infty}^{\infty} x(p)dp \leq U,$$

[5]Here we use the notation $\left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} x(p)dp$ to represent $\int_{-\infty}^{a+\eta} x(p)dp + \int_{b-\eta}^{\infty} x(p)dp$.

using this fact we obtain

$$\begin{aligned}
|I_3| &= \left| \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t) x(p) dp \right| \\
&\leq \left| \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t) dp \right| \left| \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} x(p) dp \right| \\
&\leq \left| \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t) dp \right| \left| \int_{-\infty}^{\infty} x(p) dp \right| \\
&\leq \left| \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t) dp \right| U.
\end{aligned}$$

By means of property 3.2iii we know

$$\sup_{|t-p| \geq \min(t-a-\eta, t-b+\eta)} \delta_m(p,t) \longrightarrow 0 \text{ as } m \longrightarrow \infty,$$

where $\min(x,y)$ gives the minimum value between x and y. In addition, since $\delta_m(p,t) \geq 0$, we can find a number J such that for every integer $m > J$

$$0 \leq \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t) dp \leq \kappa_J$$

where κ_J is a real number dependent on J. Hence, we can choose the value J in such a way that

$$\begin{aligned}
|I_3| &\leq \left| \left\{ \int_{-\infty}^{a+\eta} \int_{b-\eta}^{\infty} \right\} \delta_m(p,t) dp \right| U \\
&\leq \epsilon.
\end{aligned} \tag{3.10}$$

Using the bounds obtained we note

$$x_m(t) \leq M_4 + \epsilon,$$

for $t \in (a,b)$. In a similar form

$$\begin{aligned} x_m(t) &= \int_{-\infty}^{\infty} \delta_m(p,t)x(p)dp \\ &= \left\{\int_{-\infty}^{a+\eta}\int_{b-\eta}^{\infty}\right\}\delta_m(p,t)x(p)dp + \int_{a+\eta}^{b-\eta}\delta_m(p,t)x(p)dp \\ &\geq \underbrace{\left\{\int_{-\infty}^{a+\eta}\int_{b-\eta}^{\infty}\right\}\delta_m(p,t)x(p)dp}_{I_3} + \underbrace{M_3\int_{a+\eta}^{b-\eta}\delta_m(p,t)dp}_{I_5}. \end{aligned}$$

In this case we can find the lower bound for I_5 is M_3. Since the bound for the absolute value of the term I_3 is given in Equation (3.10), the lower bound for $x_m(t)$ is

$$x_m(t) \geq M_3 - \epsilon.$$

□

This characteristic of positive delta sequences will ensure that they do not present Gibbs' phenomenon since the overshoot in the projection of a function within a bounded interval cannot be bigger than ϵ, and this value can be controlled with the number of terms used in the delta sequence.

3.3.3 Positive sequences for Fourier analysis

As we have seen a positive delta sequence does not exhibit Gibbs' phenomenon. Due to this reason, there are some positive kernels created through modifications of the Fourier series.

One of these kernels is the Fejer kernel. It is given by

$$\begin{aligned} F_n(t) &= \frac{1}{n}\sum_{k=0}^{n-1} D_k(t) \\ &= \frac{\sin(\frac{nt}{2})^2}{2\pi n \sin(\frac{t}{2})^2}, \end{aligned}$$

The Fejer kernel and the projection of the sawtooth function using this kernel are shown in Figure 3.3. Another kernel based on the Fourier series is the Poisson kernel. This kernel is based on the Abel mean of the Fourier series. The Abel mean for the Fourier

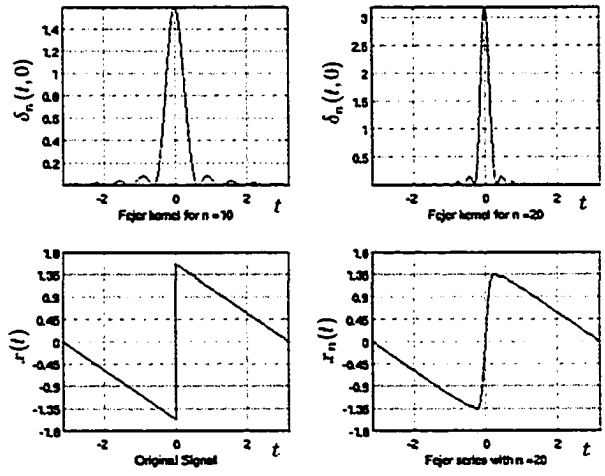

Figure 3.3: Fejer kernel for different values of n and the representation of the sawtooth function using this kernel.

series is given by

$$\sigma_r(t) = \sum_{n=-\infty}^{\infty} c_n r^{|n|} e^{int} \quad 0 < r < 1. \tag{3.11}$$

The Poisson kernel is

$$\begin{aligned} P_r(t) &= \frac{1}{2\pi} \sum_{n=-\infty}^{\infty} r^{|n|} e^{int} \\ &= \frac{1-r^2}{2\pi(1-2r\cos t + r^2)}. \end{aligned}$$

Note that the Poisson kernel has a closed form for the sum from minus infinity to infinity. Due to this reason, the dependence on n on the Dirichlet kernel is somehow replaced by a dependence on $0 < r < 1$ for this kernel. In spite of this change, the Poisson kernel behaves like a positive delta sequence except that the condition $n \longrightarrow \infty$ is replaced by $r \longrightarrow 1$.

3.4 Conclusion

In this Chapter, the concept of reproducing kernels has been introduced, and a few types of reproducing kernels have been discussed. Moreover, we have derived the reproducing kernel for a Fourier series. This kernel can be used to analyse the Gibbs' phenomenon

in a Fourier series expansion. In addition, some examples of positive kernels derivable from the Fourier series kernel have been given.

Chapter 4

Wavelets and Wavelet Packets

The wavelet transform has arisen as an alternative method to the Fourier representation of functions. The Fourier representation projects the original signal into the frequency domain. The problem with this projection is the loss of information about the time localization of the signal. This is not a problem if the signal to be represented is stationary. However, in many practical situations, we do not have stationary signals. For example, moving targets in radar and sonar, and voice signals and images have nonstationary characteristics. In this case we require a transform that provides us with information about the frequency and time localization of the signal at the same time.

The wavelet transform gives a solution to the above problem, and, in addition, it also brings other advantages such as higher levels of signal compression and better convergence behaviour. The first known wavelet family was created by Haar at the beginning of the twentieth century. However, his work was almost forgotten until around the year 1975 when Jean Morlet introduced the term "wavelets" to describe this type of function, and in 1981 teamed up with Alex Grossman to introduce the concept of continuous wavelet transforms and their inverses. Mallat [12] introduced the concept of multiresolution analysis which gave a new and easier way to compute the projection of a signal into a wavelet space. His work also contributed to the discovery of the discrete wavelet transform and wavelet packets. Around 1988, Daubechies [2] used the multiresolution analysis combined with other wavelet concepts to create her family of orthogonal wavelets with compact support. The Daubechies wavelet was the first wavelet function to possess compact time support,and that is why it is one of the wavelet families most often used.

4.1 Multiresolution Analysis (MRA) of $L^2(\mathbb{R})$

The multiresolution analysis theory is based on the existence of an orthogonal basis function. In the case of wavelets this function is called the scaling function $\phi(t)$.

Definition 4.1.1 (Multiresolution Analysis). The multiresolution analysis of $L^2(\mathbb{R})$ is a subspace sequence $\{V_j\}_{j\in\mathbb{Z}} \in L^2(\mathbb{R})$ such that

$$\{\phi(t-n) : n \in \mathbb{Z}\} \quad \text{is an orthonormal basis of } V_0 \tag{4.1i}$$

$$V_j \subset V_{j+1} \quad \text{for every } j \in \mathbb{Z} \tag{4.1ii}$$

$$\phi(t) \in V_j \text{ iff } \phi(2t) \in V_{j+1} \tag{4.1iii}$$

$$\bigcap_{j=-\infty}^{\infty} V_j = \{0\} \quad \text{span}\left\{\bigcup_{j=-\infty}^{\infty} V_j\right\} = L^2(\mathbb{R}). \tag{4.1iv}$$

The basis for the space V_m is given by

$$\phi_{m,n}(t) = 2^{m/2}\phi(2^m t - n),$$

with $m, n \in \mathbb{Z}$.
In addition to the properties given by the MRA, $\phi(t)$ is r times differentiable with continuous and rapidly decreasing functions [1]. Hence, for a suitable C_{pk},

$$|\phi^{(k)}(t)| \leq C_{pk}(1+|t|)^{-p}, \quad k = 0, 1, \cdots, r \quad p \in \mathbb{Z}^+, \quad t \in \mathbb{R}. \tag{4.2}$$

This property is related to the convergence of the projection of functions onto wavelet subspaces. There is also a function $\psi(t) \in W_0 \in L^2(\mathbb{R})$ called the wavelet function which forms a set of closed subspaces W_m, where

$$\psi_{m,n}(t) = 2^{m/2}\psi(2^m t - n)$$

with $m, n \in \mathbb{Z}$ is an orthonormal basis for W_m.
Some of the wavelet functions including those of Daubechies [2] are also orthonormal "between spaces", so that

$$\langle \psi_{m,n}(t), \psi_{p,k}(t) \rangle = \delta_{m-p}\delta_{n-k}, \tag{4.3}$$

where δ_m is the Kronecker delta.
The space W_m is a complementary space to V_m and the union of the two yields V_{m+1}.

This means

$$V_{m+1} = V_m \oplus W_m.^1 \tag{4.4}$$

Using Equation (4.4), we note $W_{-1} \subset V_0$ and since $\phi(t)$ yields a basis for the space V_0 we can find a set of coefficients $\{g_k\}$ such that

$$\frac{1}{\sqrt{2}}\psi\left(\frac{t}{2}\right) = \sum_k g_k \phi(t-k). \tag{4.5}$$

In a similar way, $V_{-1} \subset V_0$ so we can also find a set of coefficients $\{h_k\}$ such that

$$\frac{1}{\sqrt{2}}\phi\left(\frac{t}{2}\right) = \sum_k h_k \phi(t-k). \tag{4.6}$$

Equation (4.6) combined with the orthogonality properties of the basis function $\phi(t)$ gives rise to other properties. First they introduce an orthonormal-like relation for the coefficients $\{h_k\}$. This relation is

$$\sum_{n \in \mathbf{Z}} h_n h^*_{n+2k} = \delta_k. \tag{4.7}$$

Secondly they give rise to a property known as the "partition of unity". This property states

$$\sum_k \phi(t-k) = 1. \tag{4.8}$$

In addition, the dependence of the function $\psi(t)$ on $\phi(t)$ creates a relation between the coefficients $\{g_k\}$ and $\{h_k\}$ [13]. This relation is given by

$$g_k = (-1)^k h^*_{-k+1}. \tag{4.9}$$

It is sometimes called the "alternating flip" relationship (Strang and Nguyen [14]). Using

[1] The $A \oplus B$ operation gives as a result a subspace whose span is the union of the elements spanned by the subspaces A and B. The spaces are also orthogonally complementary, so if $a \in A$, $b \in B$, then $< a, b >= 0$ for a suitable inner product $< ., . >$.

Equation (4.6) the definition of the function $\phi_{j,k}(t)$ becomes

$$\begin{aligned}\phi_{j,k}(t) &= 2^{j/2}\phi(2^j t - k) \\ &= 2^{(j+1)/2}\sum_r h_r \phi(2^{j+1}t - 2k - r) \\ &= \sum_r h_r \phi_{j+1,2k+r}(t) \\ &= 2^{(j+1)/2}\sum_n h_{n-2k}\phi(2^{j+1}t - n) \\ &= \sum_n h_{n-2k}\phi_{j+1,n}(t).\end{aligned} \tag{4.10}$$

Similarly we obtain

$$\begin{aligned}\psi_{j,k}(t) &= \sum_r g_r \phi_{j+1,2k+r}(t) \\ &= \sum_n g_{n-2k}\phi_{j+1,n}(t).\end{aligned} \tag{4.11}$$

The Daubechies wavelet and scaling functions are special due to the fact that they have a compact support in time [2]. The scaling function is defined inside a given interval $[0, M-1]$, where M is a natural number which is always even. The scaling function is identically zero-valued outside of $[0, M-1]$. In this case the number of coefficients h_k is M. These coefficients are found in tables and are constructed from Daubechies theory.

4.2 Projections of $f(t) \in L^2(\mathbb{R})$ onto V_j and W_j

The projection of a function $f(t) \in L^2(\mathbb{R})$ on the space V_{j+1} is given by

$$P_{V_{j+1}} f = \sum_n \underbrace{< f, \phi_{j+1,n} >}_{c_n^{j+1}} \phi_{j+1,n}(t). \tag{4.12}$$

Since $V_{j+1} = V_j \oplus W_j$

$$\begin{aligned} P_{V_{j+1}} f &= P_{V_j} f + P_{W_j} f \\ &= \sum_k < f, \phi_{j,k} > \phi_{j,k}(t) + \sum_k < f, \psi_{j,k} > \psi_{j,k}(t) \\ &= \sum_k c_k^j \phi_{j,k}(t) + \sum_k d_k^j \psi_{j,k}(t) \\ &= \sum_k c_k^j \left[\sum_n h_{n-2k} \phi_{j+1,n}(t) \right] + \sum_k d_k^j \left[\sum_n g_{n-2k} \phi_{j+1,n}(t) \right] \\ &= \sum_n \underbrace{\left[\sum_k c_k^j h_{n-2k} + \sum_k d_k^j g_{n-2k} \right]}_{=c_n^{j+1}} \phi_{j+1,n}(t). \end{aligned} \tag{4.13}$$

The operations

$$F_0^* c^j = \sum_k c_k^j h_{n-2k} \ , \quad F_1^* d^j = \sum_k d_k^j g_{n-2k} \tag{4.14}$$

are known as the interpolation operators, and concisely $c^{j+1} = F_0^* c^j + F_1^* d^j$. Hence the interpolation operations allow us to compute the coefficients of projections of functions into the space V_{j+1} with only the knowledge of the coefficients from the spaces V_j and W_j. Note also the coefficients c_k^j can be expressed as

$$\begin{aligned} c_k^j &= < f, \phi_{j,k} > \\ &= < f, \sum_n h_{n-2k} \phi_{j+1,n}(t) > \\ &= < f, \sum_r h_r \phi_{j+1,r+2k}(t) > \\ &= \sum_r h_r^* < f, \phi_{j+1,r+2k}(t) > \\ &= \sum_r h_r^* c_{2k+r}^{j+1} \\ &= \sum_n c_n^{j+1} h_{n-2k}^*. \end{aligned} \tag{4.15}$$

Analogously, we can obtain

$$d_k^j = \sum_n c_n^{j+1} g_{n-2k}^*. \tag{4.16}$$

The decimation operators are defined as

$$F_0 c^j = \sum_n c_n^j h_{n-2k}^* , \quad F_1 d^j = \sum_n d_n^j g_{n-2k}^*, \tag{4.17}$$

and so concisely $c^j = F_0 c^{j+1}$, and $d^j = F_1 d^{j+1}$. The decimation and interpolation operators are related in accordance with the following properties:

$$F_0^* F_0 + F_1^* F_1 = I \tag{4.18i}$$

$$F_0 F_1^* = F_1 F_0^* = 0 \tag{4.18ii}$$

$$F_0^* F_0 = F_1^* F_1 = I, \tag{4.18iii}$$

where I is the identity operator.

The above properties yield a method of the construction of a recursive tree to compute the coefficients associated with a set of spaces $\{V_j\}$ and $\{W_j\}$. In Figure 4.1 we notice how using the decimation operators we can decompose the projection coefficients c_k^L into different sets of coefficients. The original coefficients can be recovered from the new set of coefficients using the interpolation operators. This process will allow us to represent the projection of $f(t)$ onto V_j in different forms. The coefficients used in the alternative representations can have advantages over the original coefficients (i.e., become zero) allowing us to represent the signal more efficiently. This has applications in signal compression and denoising.

In the case shown in Figure 4.1, an alternative for the coefficients c_k^L is the set of coefficients c_k^{L-1}, d_k^{L-1}. Another choice can be the set $c_k^{L-2}, d_k^{L-2}, e_k^{L-2}, f_k^{L-2}$. In general we can form at least 2^{2^L} different sets to represent the coefficients from the projection of function $f(t)$ into the space V_L. These representations are known as the "wavelet packet basis" for the space V_L. The wavelet transform is a specific case of a wavelet packet basis. In this case the set of coefficients chosen to form this basis is $c_k^{L-p}, d_k^{L-p}, d_k^{L-p+1}, \ldots, d_k^{L-1}$ as it is shown in Figure 4.2.

4.3 Matrix Operators for Compactly Supported Wavelets

In Section 4.2 we noted that the decimation and interpolation operations are used in the computation of coefficients for wavelet packet bases. Assuming the scaling function $\phi(t)$ has compact support (such as Daubechies compact wavelets), it can be expanded by a finite set of coefficients h_k with $k \in [0...M-1]$, where M is an even number.

The system **H** will be defined as the system whose impulse response sequence is given

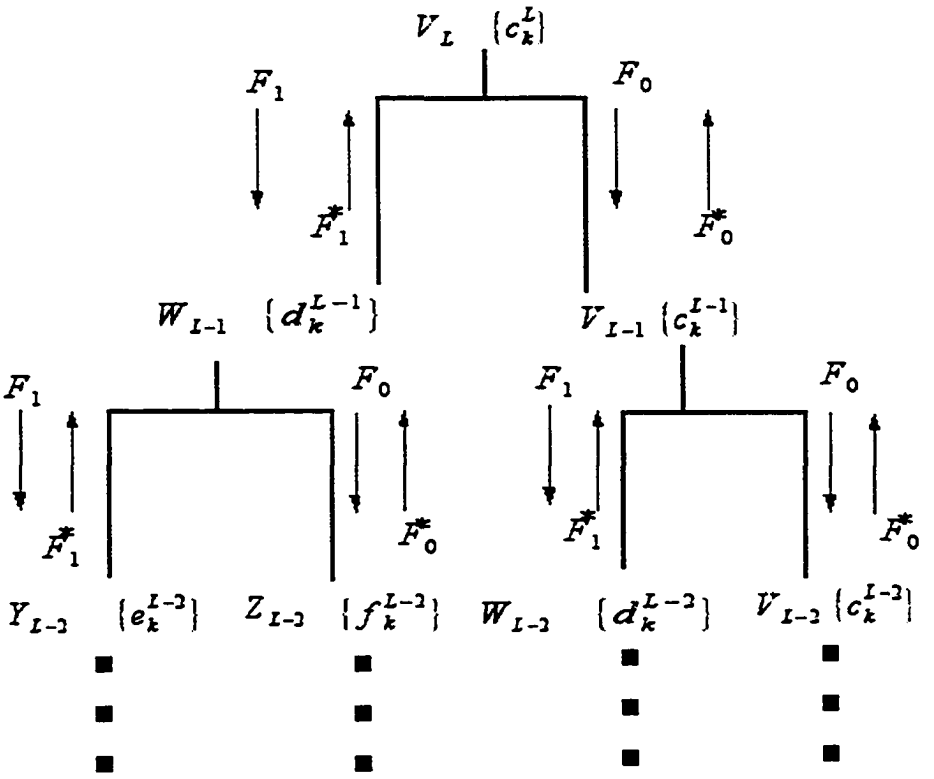

Figure 4.1: Decomposition tree for wavelet packets

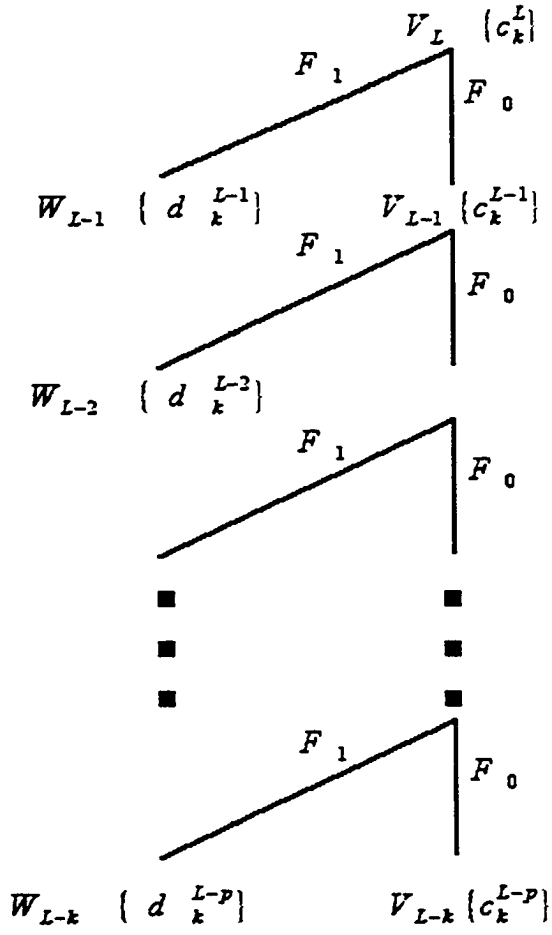

Figure 4.2: Decomposition tree for wavelet transform

by h_k . The asterisk superscript denotes complex conjugate, but we will usually assume $h_k \in \mathbb{R}$ for all k in what follows, and so the conjugate will often be dropped.

The decimation operation can be divided into two stages:

1. Convolution with the time inverse and conjugate of system **H**.
2. Down-sampling by a factor of two.

This operation is depicted in Figure 4.3.

If the input signal is of finite length N, the decimator operation becomes

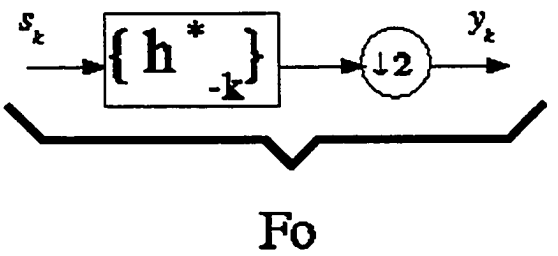

Figure 4.3: Decimation operation (decimator).

$$y_n = F_0\{s_k\} = \sum_{k=0}^{N-1} s_k h^*_{k-2n}. \tag{4.19}$$

Since the scaling function has compact support, then $k - 2n \in [0...M-1]$. For $k = 0$

$$-(M-1) \leq 2n \leq 0$$

so that

$$-\left\lfloor \frac{M-1}{2} \right\rfloor \leq n \leq 0.$$

For $k = N-1$

$$-(M-1) \leq -N+1+2n \leq 0$$

so that

$$N - M \leq 2n \leq N-1.$$

Therefore,

$$\left\lfloor \frac{N-M}{2} \right\rfloor \leq n \leq \left\lfloor \frac{N-1}{2} \right\rfloor .$$

Combining these inequalities we find that

$$\operatorname{supp} y_n = \left[-\left\lfloor \frac{M-1}{2} \right\rfloor , \left\lfloor \frac{N-1}{2} \right\rfloor \right] .^{2} \tag{4.20}$$

The decimation can be represented as a matrix operation [15] of the form $\mathbf{y} = \mathbf{DH}\mathbf{s}$, where

$$\mathbf{H} = \begin{pmatrix} h_{M-1}^* & 0 & \cdots & 0 & 0 \\ h_{M-2}^* & h_{M-1}^* & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_1^* & h_2^* & \cdots & h_{M-1}^* & 0 \\ h_0^* & h_1^* & & h_{M-2}^* & h_{M-1}^* \\ 0 & h_0^* & \ddots & \vdots & h_{M-2}^* \\ \vdots & \vdots & \ddots & h_1^* & \vdots \\ 0 & 0 & \cdots & h_0^* & h_1^* \\ 0 & 0 & \cdots & 0 & h_0^* \end{pmatrix} \in \mathbb{C}^{(N+M-1)\times N}$$

and

$$\mathbf{D} = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 \end{pmatrix} \in \mathbb{C}^{L\times(N+M-1)}$$

with

$$L = \left\lfloor \frac{N-1}{2} \right\rfloor + \left\lfloor \frac{M-1}{2} \right\rfloor + 1 = \left\lfloor \frac{N+M-1}{2} \right\rfloor .$$

For the interpolation operation the output for a finite signal s of length N is

$$y_n = F_0^*\{s_k\} = \sum_{k=0}^{N-1} s_k h_{n-2k}. \tag{4.21}$$

For $k = 0$

$$M-1 \geq n \geq 0$$

[2] $\operatorname{supp} x_n = [A, B]$ means that x_n is nonzero only for $A \leq n \leq B$

and for $k = N-1$

$$M-1 \geq n-2(N-1) \geq 0,$$

or

$$M-1 \geq n-2N+2 \geq 0$$

so that

$$M+2N-3 \geq n \geq 2N-2.$$

Combining these inequalities we find that

$$\operatorname{supp} y_n = [0, M+2N-3]\,. \tag{4.22}$$

This function can be represented as a matrix operation of the form $\mathbf{y} = \mathbf{HU}\,\mathbf{s}$, where

$$\mathbf{H} = \begin{pmatrix} h_0 & 0 & \cdots & 0 & 0 \\ h_1 & h_0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{M-2} & h_{M-3} & \cdots & h_0 & 0 \\ h_{M-1} & h_{M-2} & & h_1 & h_0 \\ 0 & h_{M-1} & \ddots & \vdots & h_1 \\ \vdots & \vdots & \ddots & h_{M-2} & \vdots \\ 0 & 0 & \cdots & h_{M-1} & h_{M-2} \\ 0 & 0 & \cdots & 0 & h_{M-1} \end{pmatrix} \in \mathbb{C}^{(M+2N-2)\times(2N-1)}$$

and

$$\mathbf{U} = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{pmatrix} \in \mathbb{C}^{(2N-1)\times N}.$$

4.4 Wavelet Kernel

In previous sections we showed that we are able to approximate a function $f(t) \in L^2(\mathbb{R})$ through a projection into a space V_m. This projection can be expressed as

$$\begin{aligned} P_{V_m} f(t) &= \sum_{n=-\infty}^{\infty} \langle f, \phi_{m,n} \rangle \phi_{m,n}(t) \\ &= \sum_{n=-\infty}^{\infty} \left\{ \int_{-\infty}^{\infty} f(x) 2^{m/2} \phi(2^m x - n)\, dx \right\} 2^{m/2} \phi(2^m t - n) \\ &= \int_{-\infty}^{\infty} f(x) \underbrace{\left\{ 2^m \sum_{n=-\infty}^{\infty} \phi(2^m x - n) \phi(2^m t - n) \right\}}_{q_m(x,t)} dx \\ &= \int_{-\infty}^{\infty} q_m(x,t) f(x) dx \end{aligned} \tag{4.23}$$

$q_m(x,t)$ is known as the reproducing kernel for V_m; note $q_m(x,t) = 2^m q_0(2^m x, 2^m t)$. Walter [1] shows this kernel is a quasi-positive delta sequence.

4.5 Gibbs' Phenomenon for Wavelets

The wavelet kernel is not a positive delta sequence. Hence we must check if this kernel gives rise to Gibbs' phenomenon. In addition, the wavelet series kernel $q_m(x,t)$ given in Equation (4.23) does not depend on the time difference $x - t$. Due to this reason the behavior of the wavelet series for a function with discontinuities must be analysed at a general point.

The function used to test for Gibbs' phenomenon will be:

$$g(x) = f(x-b) = \begin{cases} (b-1) - x & (b-1) \le x \le b \\ (b+1) - x & b < x \le (b+1) \\ 0 & \text{otherwise.} \end{cases}$$

This function has a discontinuity at the point $x = b$.
The projection of this function into the space V_m is given by

$$\begin{aligned}
P_{V_m} g(x) &= \int_{-\infty}^{\infty} g(y) q_m(y,x) dy \\
&= \int_{b-1}^{b} [(b-1)-y]\, q_m(y,x) dy + \int_{b}^{b+1} [(b+1)-y]\, q_m(y,x) dy \\
&= \int_{b-1}^{b} [(b-1)-y]\, 2^m q_0(2^m y, 2^m x) dy \\
&\quad + \int_{b}^{b+1} [(b+1)-y]\, 2^m q_0(2^m y, 2^m x) dy \\
&= \int_{b-1}^{b} [(b-1)-y]\, 2^m q_0(2^m y, 2^m x) dy \\
&\quad + \int_{b}^{b+1} [(b+1)-y]\, 2^m q_0(2^m y, 2^m x) dy \\
&= \int_{2^m(b-1)}^{2^m b} \left[(b-1) - 2^{-m} t\right] q_0(t, 2^m x) dt \\
&\quad + \int_{2^m b}^{2^m (b+1)} \left[(b+1) - 2^{-m} t\right] q_0(t, 2^m x) dt.
\end{aligned} \tag{4.24}$$

We are interested in the behavior of the projection close to the discontinuities, so we will choose $x = 2^{-m} a + b$, where a is a fixed real number, and we will allow $m \to \infty$.

Substituting this choice for x into Equation (4.24) we obtain

$$\begin{aligned}
P_{V_m} g(2^{-m}a+b) &= \int_{2^m(b-1)}^{2^m b} \left[(b-1)-2^{-m}t\right] q_0(t, a+2^m b)dt \\
&\quad + \int_{2^m b}^{2^m(b+1)} \left[(b+1)-2^{-m}t\right] q_0(t, a+2^m b)dt \\
&= -\int_{2^m}^{0} \left[(b-1)-(-2^{-m}u+b)\right] q_0(-u+2^m b, a+2^m b)du \\
&\quad + \int_{0}^{2^m} \left[(b+1)-(2^{-m}u+b)\right] q_0(u+2^m b, a+2^m b)du \\
&= -\int_{2^m}^{0} \left[-1+2^{-m}u\right] q_0(-u+2^m b, a+2^m b)du \\
&\quad + \int_{0}^{2^m} \left[1-2^{-m}u\right] q_0(u+2^m b, a+2^m b)du \\
&= \int_{2^m}^{0} \left[1-2^{-m}u\right] q_0(-u+2^m b, a+2^m b)du \\
&\quad + \int_{0}^{2^m} \left[1-2^{-m}u\right] q_0(u+2^m b, a+2^m b)du \\
&= -\int_{0}^{2^m} \left[1-2^{-m}u\right] q_0(-u+2^m b, a+2^m b)du \\
&\quad + \int_{0}^{2^m} \left[1-2^{-m}u\right] q_0(u+2^m b, a+2^m b)du \\
&= \int_{0}^{2^m} \left[1-2^{-m}u\right] \left[-q_0(-u+2^m b, a+2^m b) + q_0(u+2^m b, a+2^m b)\right] du \\
&= \int_{0}^{2^m} \left[1-2^{-m}u\right] \left[q_0(u+2^m b, a+2^m b) - q_0(-u+2^m b, a+2^m b)\right] du.
\end{aligned} \tag{4.25}$$

Note that

$$\begin{aligned}
q_0(x,t) &= \sum_{n=-\infty}^{\infty} \phi(x-n)\phi(t-n) \\
&= \sum_{n=-\infty}^{\infty} \phi(x-(n+n_1))\phi(t-(n+n_1)) \\
&= q_0(x-n_1, t-n_1),
\end{aligned}$$

with $n_1 \in \mathbb{Z}$. Then,

$$\begin{aligned} q_0(u+2^m b, a+2^m b) &= q_0(u+\underbrace{2^m b - \lfloor 2^m b \rfloor}_{b_m}, a+\underbrace{2^m b - \lfloor 2^m b \rfloor}_{b_m}) \\ &= q_0(u+b_m, a+b_m). \end{aligned} \tag{4.26}$$

Using Equation (4.26) in Equation (4.25) we obtain

$$P_{V_m} g(2^{-m}a+b) = \int_0^{2^m} \left[1-2^{-m}u\right] \left[q_0(u+b_m, a+b_m) - q_0(-u+b_m, a+b_m)\right] du. \tag{4.27}$$

When $m \to \infty$, Equation (4.27) becomes

$$\begin{aligned} &\int_0^{\infty} \left[q_0(u+b_m, a+b_m) - q_0(-u+b_m, a+b_m)\right] du \\ &= \int_0^{\infty} q_0(u+b_m, a+b_m) du - \int_0^{\infty} q_0(-u+b_m, a+b_m) du \\ &= \int_{b_m}^{\infty} q_0(u, a+b_m) du + \int_{b_m}^{-\infty} q_0(u, a+b_m) du \\ &= \int_{b_m}^{\infty} q_0(u, a+b_m) du - \int_{-\infty}^{b_m} q_0(u, a+b_m) du \\ &= \int_{b_m}^{\infty} q_0(u, a+b_m) du - \int_{-\infty}^{b_m} q_0(u, a+b_m) du + 1 - 1 \\ &= \int_{b_m}^{\infty} q_0(u, a+b_m) du - \int_{-\infty}^{b_m} q_0(u, a+b_m) du + \int_{-\infty}^{\infty} q_0(u, a+b_m) du - 1 \\ &= \int_{b_m}^{\infty} q_0(u, a+b_m) du + \int_{b_m}^{\infty} q_0(u, a+b_m) du - 1 \\ &= 2\int_{b_m}^{\infty} q_0(u, a+b_m) du - 1. \end{aligned} \tag{4.28}$$

Using this equation we note that there is Gibbs' phenomenon near the point b if there is a number $a > 0$ such that

$$\int_{b_m}^{\infty} q_0(u, a+b_m) du > 1,$$

or a number $a < 0$ such that

$$\int_{b_m}^{\infty} q_0(u, a+b_m) du < 0.$$

Using this test, Kelly [16] showed that the Haar scaling functions which are given by

$$\phi(t) = \begin{cases} 1 & \text{if } 0 \leq t < 1 \\ 0 & \text{otherwise,} \end{cases} \tag{4.29}$$

do not present Gibbs' phenomenon. On the other hand, Daubechies' scaling functions present the Gibbs phenomenon on the points where $b_m = 0$.

4.6 Conclusion

In this Chapter, the basic multiresolution analysis concepts have been reviewed. These concepts have been used to introduce the concept of wavelet and wavelet packet transforms. The decimation and interpolation operations which are used in these transforms have been illustrated also. Finally, a study has been undertaken of the Gibbs' phenomenon for wavelet and wavelet packet transforms.

Chapter 5

Minimum Description Length (MDL) Criterion

The use of wavelet packets and the wavelet transform have increased considerably. However, the variety of existent wavelet functions and basis sets to choose from is a problem when we want to apply the wavelet theory to a problem. Sometimes the previous knowledge of the problem is enough to find the ideal wavelet function and projection coefficients. Nevertheless, it is of interest to find an automated criterion to choose the best basis among a given set of possible bases. A short introduction to some of these methods is given by Merhav [17]. One of these criteria is the Minimum Description Length (MDL) criterion proposed by Risannen [18] [19] [20].

5.1 The Model-Order Selection Problem

Consider the noisy function

$$y = x + \varepsilon, \tag{5.1}$$

where $y, x, \varepsilon \in \mathbb{R}$. The signal y is the noisy function, x is the signal to be estimated and ε is noise. In addition, x, y, ε are functions of time. We often assume ε is Gaussian noise. For the model-order selection problem we use a number n of sample points obtained from the signal y.

We have a group of models built using orthonormal bases. These bases can be wavelet packets, projections into different wavelet spaces V_j or any orthonormal basis capable of representing the data y and x. Suppose we have a set of orthonormal bases $B_1, B_2, \ldots, B_m \ldots, B_d$. The model m is constructed by projecting the noisy data y using the basis B_m from the set of orthonormal bases. The number of coefficients different

than zero for the projection into the space B_m will be k_m. The number k_m is known as the order given by the model m.

The MDL criterion will pick the model with the basis B_t and the model order k_t that represents the signal x in the best form. Note the MDL criterion is just a model-order selector. Hence it will try to find the best model-order to represent the signal among the models at hand. However, if the bases chosen from the orthonormal basis set do not represent the signal x well, the solution obtained will not be satisfactory, but it will be the best possible solution from the given ones.

5.2 Selection Criterion

The MDL criterion measures the quality of a certain model as the complexity to represent the data with the model. If the model does not represent the data well, we will need a lot of information (i.e. bits) in order to obtain the desired representation. Hence, the complexity of representation for that model will be large. On the other hand, if the model used represents the data well, the complexity is small.

In the case the model fits the data well, we need to ensure that the model is not overfitting. This means the basis represents the data well but any small variation (i.e. noise) will cause the model to be inaccurate. The possibility of overfitting is related to the level of complexity of the basis used to represent the signal. If the elements of the basis are highly elaborate functions or the number of coefficients used to represent the data is large compared to other bases, then the model is more complex and hence this model will be less desirable than a simpler model.

The MDL criterion measures the data representation complexity assuming the data obtained by representing the signal using a model m is a symbol which has to be coded and transmitted. The coding scheme used to represent the symbols will be a prefix scheme. This scheme creates the codewords in such a way that no codeword is a prefix of any other codeword. The codewords are related to the symbols according to the probability of appearance of the symbols. Hence the word with the shortest codelength will be given to the symbol with the greatest probability. The MDL criterion does not care about the codewords. It only selects the codelength value used to assign the codewords. The MDL criterion proposes to measure the codelength for the model H using the formula

$$L(H) + L(D|H), \tag{5.2}$$

where $L(H)$ is the codelength, in bits, of the description for the model H and $L(D|H)$ is the codelength, in bits, of the residual D (i.e. the difference) between the actual data y and the model prediction of the data using the model H. The MDL criterion says the best model to represent the data D will be the one with the smallest codelength.

5.3 $L(D|H)$ Codelength Computation

As we have seen the prefix coding scheme codelength is closely related to the probability of occurrence of the symbols. Equation (5.1) shows us the data to be represented by the models is a random function. Hence it is straightforward to use this random function to obtain $L(D|H)$.

By definition, $L(D|H)$ is the codelength of the data D when the model H is used, then we can use the probability $p(D|H)$ of obtaining the data D given the model H to get the codelength. In order to obtain the minimum possible codelength we need to maximise the probability $p(D|H)$. This maximization is accomplished using the maximum likelihood estimate (MLE) [21] $p(D|\hat{H})$ of $p(D|H)$.

Using the MLE $L(D|H)$ is given by

$$L(D|H) = -\log_2 \left[p(D|\hat{H}) \right]. \tag{5.3}$$

5.4 $L(H)$ Codelength Computation

The definition of term $L(H)$ is harder due to the fact we cannot relate a probabilistic value directly to the model. However, this term can be related to the order of the model. The order of the model will be used to measure the grade of compression the model H is capable of delivering. It has been shown [20] [22] that this can be accomplished as

$$L(H) = \frac{k}{2} \log_2(n) + c_k, \tag{5.4}$$

where k is the number of coefficients used to express the data using the model H, c_k is a constant that depends on k, and n is the number of data points obtained from the signal y. Furthermore, Saito [22] suggests that the term c_k has an almost constant value when we are using wavelet bases, and then it can be ignored.

5.5 Conclusion

In this Chapter, an introduction has been given of the use of the minimum description length criterion. This criterion has subsequently been used to find the best basis functions to represent a positive-valued signal.

Chapter 6

Gaussian Noise as a Decimator Input

The wavelet transform is used in many applications. Generally, the signal to be processed is projected into a scaling function subspace V_j and then the transform coefficients are computed using the decimator operation as it was shown in Figure 4.3. The transform coefficients amount to a representation of the original signal. After the coefficients are obtained, they can be processed, perhaps for purposes such as denoising, compression, or signal detection. However, the original signal is often corrupted by additive noise. This noise can create problems when we want to recover the signal from the transform coefficients.

In Section 4.3 we noticed that the decimator operation can be implemented as a matrix operation when the scaling function has compact support. In this chapter we will characterize the behavior of white Gaussian noise (WGN) introduced into a system composed of a chain of decimators. The scaling function $\phi(t)$ in this system will have compact support. Hence it will be associated with a finite number of coefficients h_k with $k \in [0 \dots M-1]$.

There have been previous works [3] [4] [5] that analyzed the behaviour of a function with noise when it is introduced into a chain of decimators. However, in all these works the input signal used has a support of $(-\infty, \infty)$. In many practical applications, this is generally not the case. Furthermore, if the support of the function to be processed gets close in size to the size of the scaling function's support, then the statistical behavior is very different from that considered in these works. In this chapter we will consider some of these difficulties.

The analysis of a random signal as an input to a system is very important in various

fields (i.e. estimation theory, detection theory). For example, if we want to detect the presence of a known deterministic signal in WGN, then consider the following simple case. In this case the detection process can be converted into a hypothesis selection problem where the hypotheses are:

$$\mathcal{H}_0 : x[n] = w[n] \quad n = 0, 1, \ldots, P-1$$
$$\mathcal{H}_1 : x[n] = s[n] + w[n] \quad n = 0, 1, \ldots, P-1,$$

where $s[n]$ is a known deterministic signal and $w[n]$ is zero-mean WGN with variance σ^2. In this model the hypothesis $\mathcal{H}_1$ represents the presence of a known signal in the system and hypothesis $\mathcal{H}_0$ represents the presence of just WGN. Hence a procedure which chooses between $\mathcal{H}_1$ and $\mathcal{H}_0$ will perform the detection. A procedure suggested in [23] to perform the detection is to choose the hypothesis $\mathcal{H}_1$ if

$$T(\mathbf{x}) = \mathbf{x^T R^{-1} s} > v,$$

where $\mathbf{x} = [x[0]x[1]\ldots x[P-1]]$, $\mathbf{s} = [s[0]s[1]\ldots s[P-1]]$, $\mathbf{R}$ is the output correlation matrix of the system when the input is WGN, and v is a selected threshold level. The threshold level would affect the detection accuracy of the model.

As we can notice, the correlation matrix inverse $\mathbf{R^{-1}}$ plays an important role in the detection problem. For this reason we have to study the behavior of matrix $\mathbf{R}$ to check if it possesses an inverse and if the inversion process is ill-conditioned.

In our case the system to analyze is a chain of n decimators since this is associated with wavelet based detectors. Hence, we will analyze its output correlation matrix when the input is WGN. We will find a predictable structure for the output correlation matrix of this system. After that, upper and lower bounds for the unitary norms for the output correlation matrix norm of the same system will be given. These bounds will be based on the the eigenvalues of the output correlation matrix. They will give us some information about the condition number of the correlation matrix. As it is shown for example by Horn [24], the condition number is related to the ill-conditioning of a matrix inversion process. Hence, we will obtain some information about the ill-conditioning in our problem.

The matrix correlation structure and matrix norm bound study are the new contributions that we have made in the analysis of Gaussian noise as an input into a decimator chain.

6.1 Random Process as an Input to the System

The output of a decimator given a nonstationary real random input can be described (partially) by the first and second order statistical moments as

$$m_{y_n} = \mathrm{E}\{y_n\} = \sum_{k=0}^{N-1} h^*_{k-2n}\,\mathrm{E}\{s_k\} \tag{6.1}$$

$$\mathrm{r}_{yy}(n_1, n_2) = \mathrm{E}\{(y_{n_1} - m_{y_{n_1}})(y_{n_2} - m_{y_{n_1}})^*\} \tag{6.2}$$

for $-\lfloor \frac{M-1}{2} \rfloor \leq n_1\,, n_2 \leq \lfloor \frac{N-1}{2} \rfloor$ [1], where $\mathrm{E}\{.\}$ denotes statistical expectation. In matrix form we have [15]

$$\mathrm{E}\{\mathbf{y}\} = \mathbf{DH}\,\mathrm{E}\{\mathbf{s}\} \tag{6.3}$$

$$\begin{aligned} \mathbf{R}_y &= \mathrm{E}\{(\mathbf{y} - \mathbf{m}_y)(\mathbf{y} - \mathbf{m}_y)^*\} \\ &= \mathbf{DH} \quad \mathrm{E}\{(\mathbf{s} - \mathbf{m}_s)(\mathbf{s} - \mathbf{m}_s)^*\} \quad \mathbf{H}^H\mathbf{D}^H \\ &= \mathbf{DH}\,\mathbf{R}_s\,\mathbf{H}^H\mathbf{D}^H \end{aligned} \tag{6.4}$$

Note this model does not assume any kind of stationarity in the input signal. This approach is different from the one used in [3] [4] [5]. In these works, the authors constructed stationary-like random functions (i.e. circularly stationary) and used them as input into the system. We do not like this approach since the model then becomes dependent on the stationarity of the input signal.

6.2 Gaussian noise (GN) as input to a Wavelet-Based Decimator Chain

6.2.1 Decimator Output Characterization

A white Gaussian random process has a correlation matrix $\mathbf{R}_s = \sigma^2\mathbf{I}$, and so the mean of the output of the system in Section 6.1 can be expressed as

$$\mathbf{m}_y = \mathrm{E}\{\mathbf{y}\} = \mathbf{DH}\,\mathrm{E}\{\mathbf{m}_s\} = \mathbf{DHm}_s \tag{6.5}$$

[1] $\lfloor x \rfloor$ is defined as the greatest integer that is less than or equal to $x \in \mathbb{R}$.

and the output correlation matrix is

$$\mathbf{R}_y = \sigma^2 \mathbf{D}\mathbf{H}\mathbf{H}^H\mathbf{D}^H \tag{6.6}$$

In addition, if the input mean is zero, then Equation (6.2) can be reduced to

$$\begin{aligned}
r_{yy}(n_1, n_2) &= \mathrm{E}\{y_{n_1} y_{n_2}^*\} \\
&= \mathrm{E}\left\{ \sum_{k=0}^{N-1} h_{k-2n_1}^* s_k \sum_{m=0}^{N-1} h_{m-2n_2} s_m^* \right\} \\
&= \sum_{k=0}^{N-1} \sum_{m=0}^{N-1} h_{k-2n_1}^* h_{m-2n_2} \mathrm{E}\{s_k s_m^*\} \\
&= \sum_{k=0}^{N-1} \sum_{m=0}^{N-1} h_{k-2n_1}^* h_{m-2n_2} \sigma^2 \delta(k-m) \\
&= \sigma^2 \sum_{k=0}^{N-1} h_{k-2n_1}^* h_{k-2n_2} \\
&= \sigma^2 \sum_{p=-2n_2}^{N-1-2n_2} h_p h_{p+2(n_2-n_1)}^* \quad \text{or} \\
&= \sigma^2 \sum_{p=-2n_1}^{N-1-2n_1} h_p^* h_{p+2(n_1-n_2)}
\end{aligned} \tag{6.7}$$

Note how the lower and upper limits on the output correlation can depend either on n_1 or on n_2.

Due to the fact that the coefficients for the scaling function $\phi(t)$ have a support $\operatorname{supp} h = [0...M-1]$, Equation (4.20) shows the intervals of n_1 and n_2 where the auto-correlation is defined are $n_1 \in \left[-\lfloor \frac{M-1}{2} \rfloor ... \lfloor \frac{N-1}{2} \rfloor\right]$ and $n_2 \in \left[-\lfloor \frac{M-1}{2} \rfloor ... \lfloor \frac{N-1}{2} \rfloor\right]$.
For the sake of clarity, the terms of the correlation will be displaced in time, where this displacement is not going to affect the structure of the correlation; it will only modify its time location which is not important in our problem. The time localization can be re-established with a displacement in time of the correlation after it has been obtained. This effect is achieved with a change of variable. We will define $m_1 = n_1 + \lfloor \frac{M-1}{2} \rfloor$ and

$m_2 = n_2 + \lfloor \frac{M-1}{2} \rfloor$. With this change, Equation (6.7) becomes

$$\begin{aligned} \mathrm{r}'_{yy}(m_1, m_2) &= \mathrm{E}\left\{ y_{m_1 - \lfloor \frac{M-1}{2} \rfloor} y^*_{m_2 - \lfloor \frac{M-1}{2} \rfloor} \right\} \\ &= \sigma^2 \sum_{p=-2m_2+2\lfloor \frac{M-1}{2} \rfloor}^{N-1-2m_2+2\lfloor \frac{M-1}{2} \rfloor} h_p h^*_{p+2d} \quad \text{or} \\ &= \sigma^2 \sum_{p=-2m_1+2\lfloor \frac{M-1}{2} \rfloor}^{N-1-2m_1+2\lfloor \frac{M-1}{2} \rfloor} h^*_p h_{p-2d}. \end{aligned} \tag{6.8}$$

where $d = m_2 - m_1$.

Equation (6.8) is only defined on the interval $0 \leq m_1, m_2 \leq L_1 - 1 = \lfloor \frac{N+M-1}{2} \rfloor - 1$.

If the system **H** describes an orthogonal wavelet transform with compact support, then the system's coefficients satisfy the orthogonality condition

$$\sum_{n=0}^{M-1} h_n h^*_{n+2k} = \delta_k, \tag{6.9}$$

where supp $h = [0...M-1]$. This equation is similar to the one obtained in Equation (6.8), where the equivalence between both depends on the upper and lower limits of the sum in Equation (6.8). If the lower limit $p = -2m_1 + 2\lfloor \frac{M-1}{2} \rfloor$ or $p = -2m_2 + 2\lfloor \frac{M-1}{2} \rfloor \leq 0$ and the upper limit $p = N-1-2m_1+2\lfloor \frac{M-1}{2} \rfloor$ or $p = N-1-2m_2+2\lfloor \frac{M-1}{2} \rfloor \geq M-1$ these two equations are equivalent. Then, for $\lfloor \frac{M-1}{2} \rfloor \leq m_1 \leq \lfloor \frac{N-M}{2} \rfloor + \lfloor \frac{M-1}{2} \rfloor$ and $\lfloor \frac{M-1}{2} \rfloor \leq m_2 \leq \lfloor \frac{N-M}{2} \rfloor + \lfloor \frac{M-1}{2} \rfloor$, Equation (6.8) can be reduced to

$$\mathrm{r}'_{yy}(m_1, m_2) = \sigma^2 \delta_{m_2 - m_1}. \tag{6.10}$$

Note also that if $|d| = |m_2 - m_1| > \lfloor \frac{M-1}{2} \rfloor$ Equation (6.8) is equal to zero.

Putting together the correlation values obtained, we form the correlation matrix for the decimator where m_1 is the row index, and m_2 is the column index. This matrix depends on the time difference $d = m_2 - m_1$ as it is shown in Equation (6.8), so then it will possess a Toeplitz-like structure. This structure will subdivide the correlation matrix into certain areas as shown in Figure 6.1; [2] as it has been seen with the previous arguments there are only three areas that are different from zero. The upper left corner, the middle and the bottom right corner, where the middle submatrix is equal to a diagonal matrix as given by Equation (6.10).

[2] In order to save space on the matrix representation, the elements of the form $h_a h_b + h_c h_d + \cdots + h_e h_f$ are represented as $h_{(a,b)+(c,d)+\cdots+(e,f)}$.

If the coefficients $h_k \in \mathbb{R}$ as it is the case for the Daubechies' wavelets [2], then the correlation matrix is real and symmetric. Due to this structure, the elements of this matrix are defined by their position relative to the main diagonal.

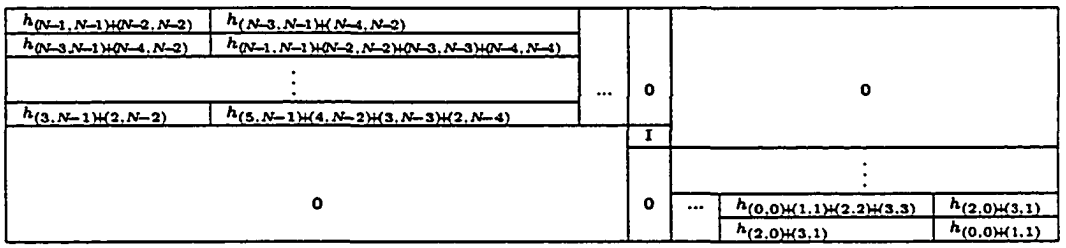

Figure 6.1: Matrix structure for $\mathbf{R}_y$ (assuming $\sigma^2 = 1$)

The elements in the upper left corner submatrix are given by Equation (6.8), with m_1 and m_2 in the interval $[0 \ldots \lfloor \frac{M-1}{2} \rfloor - 1]$. Note that in this case the lower limit of the sum $-2m_1 + 2\lfloor \frac{M-1}{2} \rfloor$ is greater than zero and approaches zero when we approach m_1 or $m_2 = \lfloor \frac{M-1}{2} \rfloor$. As a result, the number of terms in the sum is less than the number needed to obtain the equivalence with Equation (6.9). The sum values obtained in this case for different values of $d = m_1 - m_2$ are shown in Figure 6.2. In this figure it can be seen how the correlation coefficients tend to zero rapidly as d becomes larger. This effect is produced by the rapidly decreasing coefficients of system **H**.

In a similar form, the upper limit sum in Equation (6.8) gets smaller than $M-1$ for the elements on the lower right corner. In this case the number of elements in the sum would get smaller as we get close to the lower right corner. The result of the sum for this case is shown in Figure 6.3. In this figure we can see how the elements on the diagonal, $d = 0$, will remain close to one until they are close to the corner. Away from the diagonal, where $d \neq 0$, the elements will have a small overshoot as they approach the corner and then they return to zero.

Note the size of the upper left corner matrix is $LC_1 = \lfloor \frac{M-1}{2} \rfloor$, and it is independent of the size of the input signal. Analyzing Equation (6.10), we notice the size of the identity submatrix is given by $IM_1 = \lfloor \frac{N-M}{2} \rfloor + 1$, where N is the size of the input signal. Note how this value is dependent on the size of the input signal N. Subsequently the bottom right corner is also dependent on the length of the input signal and its size is $RC_1 = \lfloor \frac{M-1}{2} \rfloor + 1$ if N is odd and $RC_1 = \lfloor \frac{M-1}{2} \rfloor$ if N is even. The structure of this matrix is shown in Figure 6.4, where Q_1 is the upper left corner submatrix and W_1 is the lower right corner submatrix.

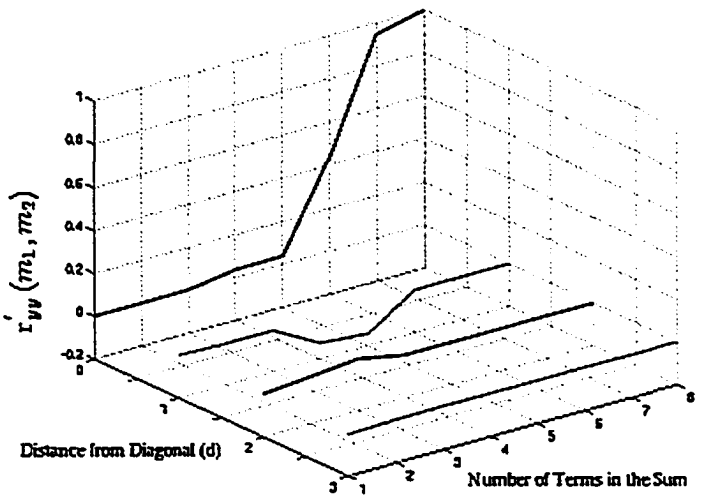

Figure 6.2: Correlation terms for which the sum lower limit of $r'_{yy}(m_1, m_2)$ is greater than zero, and here M=8.

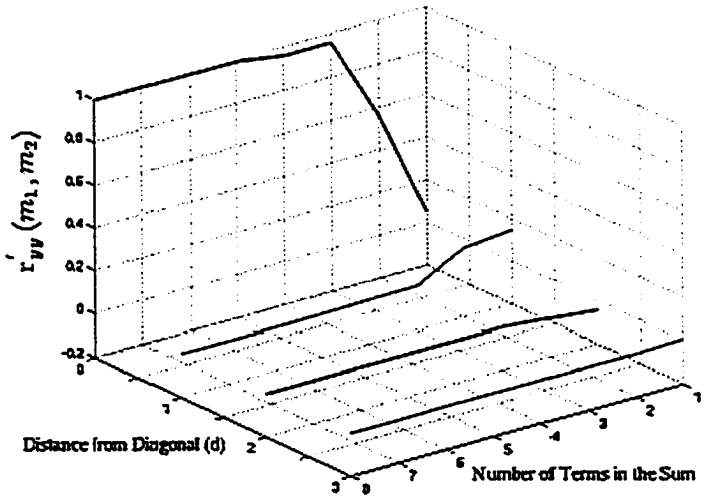

Figure 6.3: Correlation terms for which the sum upper limit of $r'_{yy}(m_1, m_2)$ is less than M, and here M=8.

Q_1	0	0
0	I_1	0
0	0	W_1

Figure 6.4: $D_1H_1H_1^HD_1^H$ structure

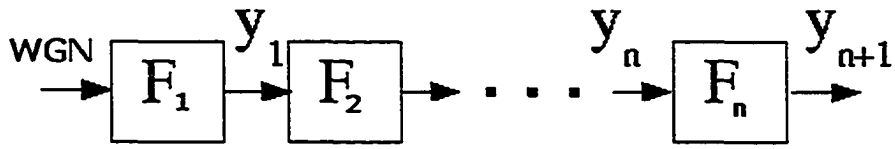

Figure 6.5: Decimators connected in a chain.

6.2.2 Chain of Decimators Correlation

Suppose we have a chain of decimators as illustrated in Figure 6.5. In this case the input correlation matrix of the next decimator will be the output of the previous one, and we continue to assume the input to the system is white Gaussian noise with zero mean. Papoulis [25] shows that the output of this system will be Gaussian noise as the system is linear.

The system correlation can be represented in a recursive form as

$$\mathbf{R}_{y_{n+1}} = \mathbf{D_n H_n R}_{y_n} \mathbf{H_n}^H \mathbf{D_n}^H, \tag{6.11}$$

where $\mathbf{R}_{y_n}$ is the n^{th} decimator correlation matrix and

$$\mathbf{R}_{y_1} = \sigma^2 \mathbf{D_1 H_1 H_1}^H \mathbf{D_1}^H \tag{6.12}$$

gives the behaviour of y_1 as was analyzed in Section 6.2.1. The size of $\mathbf{R}_{y_n}$ is given by

$$L_n = \left\lfloor \frac{L_{n-1} + M - 1}{2} \right\rfloor, \tag{6.13}$$

with

$$L_1 = \left\lfloor \frac{N + M - 1}{2} \right\rfloor,$$

where as before N the length of the signal to be input into the chain of decimators and $[0 \ldots M-1] \in \mathbb{Z}^+$ is the support of the scaling function $\phi(t)$. In Section 6.2.1 we analyzed the structure for the first correlation matrix $\mathbf{R}_{\mathbf{y_1}}$. The structure of this matrix was shown in Figure 6.4. To obtain the second correlation matrix $\mathbf{R}_{\mathbf{y_2}}$ the matrix $\mathbf{R}_{\mathbf{y_1}}$ is premultiplied by the matrix **DH** and postmultiplied by the matrix $\mathbf{(DH)^H}$. The matrix **DH** is shown in Figure 6.6.

We will find the conditions needed to obtain a n^{th} correlation matrix $\mathbf{R}_{\mathbf{y_n}}$ for a chain of n decimators with the structure shown in Figure 6.7.

For $n = 1$, the values LC_1, IM_1 and RC_1 can be obtained by analyzing the matrix **DH**. The size of the identity matrix IM_1 will be equal to the number of rows that contain all the M elements h_0 to h_{M-1} inside them; then the value of LC_1 is the number of rows from the first one until the one whose first element is either zero or h_0, and the value of RC_1 is the number of rows from the last one until the one that has the element h_{M-1}. When there are no rows that contain all the M elements there will be an overlap between the upper left corner and the bottom right corner. This case appears when the length of the input signal is very close to M, and then the size of the transient in the system will

h_{M-2}	h_{M-1}	0	0	0	0	...	0	0	0	0	0	0
h_{M-4}	h_{M-3}	h_{M-2}	h_{M-1}	0	0	...	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮	⋮	⋮	⋮
h_0	h_1	h_2	h_3	...	h_{M-1}	...	0	0	0	0	0	0
0	0	h_0	h_1	h_2	h_3	...	h_{M-1}	0	0	0	0	0
0	0	0	0	h_0	h_1	...	h_{M-3}	h_{M-2}	h_{M-1}	0	0	0
0	0	0	0	0	0	...	h_0	h_1	...	h_{M-2}	h_{M-1}	0
⋮	⋮	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮	⋮	⋮	⋮
0	0	0	0	0	0	...	0	h_0	h_1	h_2	h_3	h_4
0	0	0	0	0	0	...	0	0	0	h_0	h_1	h_2
0	0	0	0	0	0	...	0	0	0	0	0	h_0

Figure 6.6: **DH** matrix structure

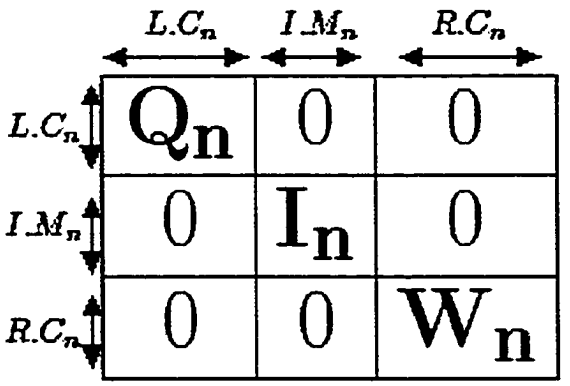

Figure 6.7: $\mathbf{D_n H_n H_n^H D_n^H}$ structure

make the analysis of the signal very difficult. In this case it is better to use a smaller M or a bigger signal length N. In accordance with this, from now on we will assume we have an initial identity matrix of size IM_1. Figure 6.8 shows LC_1 for $M = 12$.

For $n \geq 2$ we have to multiply $\mathbf{R}_{\mathbf{y}_{n-1}}$ with the matrix $\mathbf{DH}$ and its conjugate to get $\mathbf{R}_{\mathbf{y}_n}$.

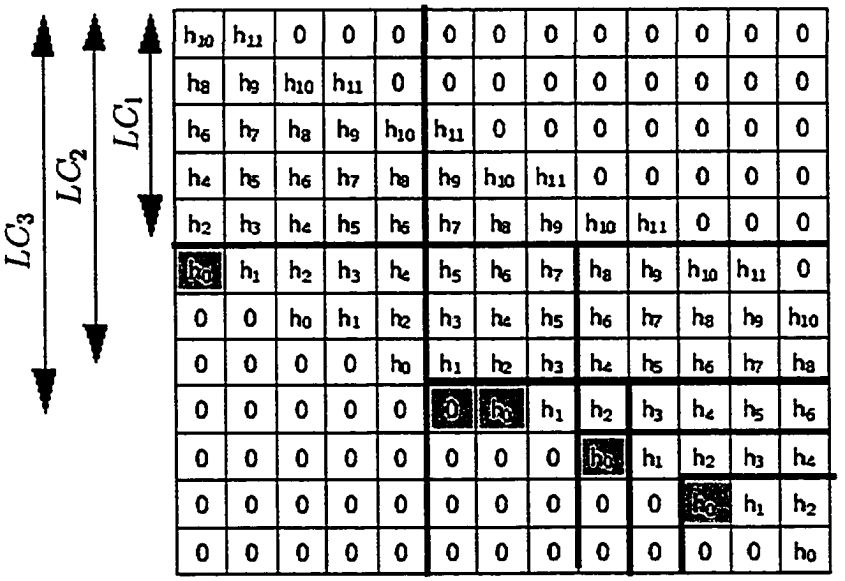

Figure 6.8: DH upper left corner for M=12.

We want to obtain an output matrix with the structure show in Figure 6.7. In order to get this result we will partition the matrix $\mathbf{D_nH_n}$ into nine submatrices as shown in Figure 6.9.

We will start the iteration process with $n = 2$, then for $n-1 = 1$ we have $LC_1 = \lfloor \frac{M-1}{2} \rfloor$,

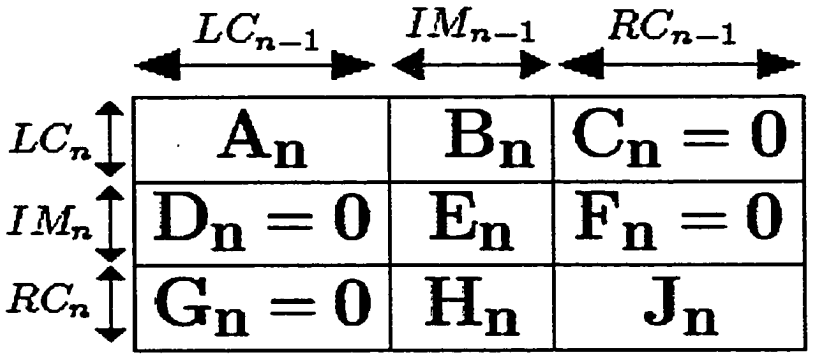

Figure 6.9: $\mathbf{D_nH_n}$ submatrix division

$IM_1 = \lfloor \frac{N-M}{2} \rfloor + 1$ and $RC_1 = \lfloor \frac{M-1}{2} \rfloor + 1$ if N is odd, and $RC_1 = \lfloor \frac{M-1}{2} \rfloor$ if N is even. As it was shown in Equation (6.12), when $n = 1$ we pre-multiply a diagonal matrix by $\mathbf{D_1H_1}$ and post-multiply it by $\mathbf{H_1^H D_1^H}$ which is the conjugate transpose of $\mathbf{D_1H_1}$. In the subsequent cases instead of a diagonal matrix we now have to pre-multiply $\mathbf{R_{y_{n-1}}}$ by $\mathbf{D_nH_n}$ and post-multiply by its conjugate transpose. If the matrix $\mathbf{R_{y_{n-1}}}$ has the structure shown in Figure 6.7, then only the middle submatrix is equal to the identity matrix. Due to this reason, the structure of the matrix $\mathbf{R_{y_n}}$ given by the recursion is subject to the following rules

$$LC_n \geq LC_{n-1} \tag{6.14a}$$

$$RC_n \geq RC_{n-1} \tag{6.14b}$$

$$0 < IM_n \leq IM_{n-1}, \tag{6.14c}$$

the recursion will be finished when $IM_n \leq 0$.

The value of RC_n is an integer chosen in such a way that the submatrix $\mathbf{G_n}$ is equal to the zero matrix. Analogously, LC_n is an integer that makes $\mathbf{C_n}$ equal to zero, and finally IM_n makes $\mathbf{D_n}$ and $\mathbf{F_n}$ also equal to zero. The structure of the partitioned $\mathbf{D_nH_n}$ matrix using these values is shown in Figure 6.9. Using this matrix partition for $\mathbf{D_nH_n}$ and the matrix partition shown in Figure 6.7 for $\mathbf{R_{y_{n-1}}}$ we will obtain the matrix structure shown in Figure 6.10 for matrix $\mathbf{R_{y_n}} = \mathbf{D_nH_nR_{y_{n-1}}(D_nH_n)^H}$. The structure of the

$\mathbf{Q_n = A_nQ_{n-1}A_n^H + B_nB_n^H + 0}$	$\mathbf{0 + B_nE_n^H + 0 = 0}$	$\mathbf{0 + B_nH_n^H + 0 = 0}$
$\mathbf{0 + E_nB_n^H + 0 = 0}$	$\mathbf{0 + E_nE_n^H + 0}$	$\mathbf{0 + E_nH_n^H + 0 = 0}$
$\mathbf{0 + H_nB_n^H + 0 = 0}$	$\mathbf{0 + H_nE_n^H + 0 = 0}$	$\mathbf{0 + H_nH_n^H + J_nW_{n-1}J^H}$

Figure 6.10: n^{th} correlation matrix structure.

autocorrelation matrix $\mathbf{R_{y_n}}$ given in Figure 6.10 will have an upper left corner given by $\mathbf{Q_n = A_nQ_{n-1}A_n^H + B_nB_n^H}$.The lower right corner will be $\mathbf{W_n = J_nW_{n-1}J_n^H + H_nH_n^H}$. In addition to the previous conditions imposed upon LC_n, RC_n and IM_n, the values of LC_n and RC_n will be chosen in such a way that $\mathbf{E_nE_n^H = I_n}$. This is accomplished when all the matrix rows from matrix $\mathbf{E_n}$ contain all the elements h_k, $k \in [0 \ldots M-1]$. Furthermore, using Equation (6.8) it can be shown that this choice will make the matrices $\mathbf{B_nE_n^H}$, $\mathbf{B_nH_n^H}$ and $\mathbf{E_nH_n^H}$ equal to zero.

Suppose as an example we have a chain of two decimators. The value of LC_2 can be obtained by examining the submatrix of $\mathbf{D_2H_2}$. The recursion rules given in Equation (6.14) tell us LC_2 has to be greater than or equal to LC_1. This characteristic can also

be seen in Figure 6.7 due to the fact that the submatrix $\mathbf{Q_1}$ is different than the identity matrix. We need to find the number of extra rows starting from row LC_1 needed to construct LC_2. In order to find these rows we will create a new matrix by getting rid of the first LC_1 rows and the same number of columns of matrix $\mathbf{D_2H_2}$. Then, we will perform the same analysis used to obtain LC_1 in this new matrix. That is to find the number of rows starting from the upper left corner that do not include the element h_0. As it can be seen in Figure 6.11, the number of rows is given by

$$\left\lfloor \frac{M - LC_1 - 1}{2} \right\rfloor. \tag{6.15}$$

Finally, as it is shown in Figure 6.11 LC_2 is given by:

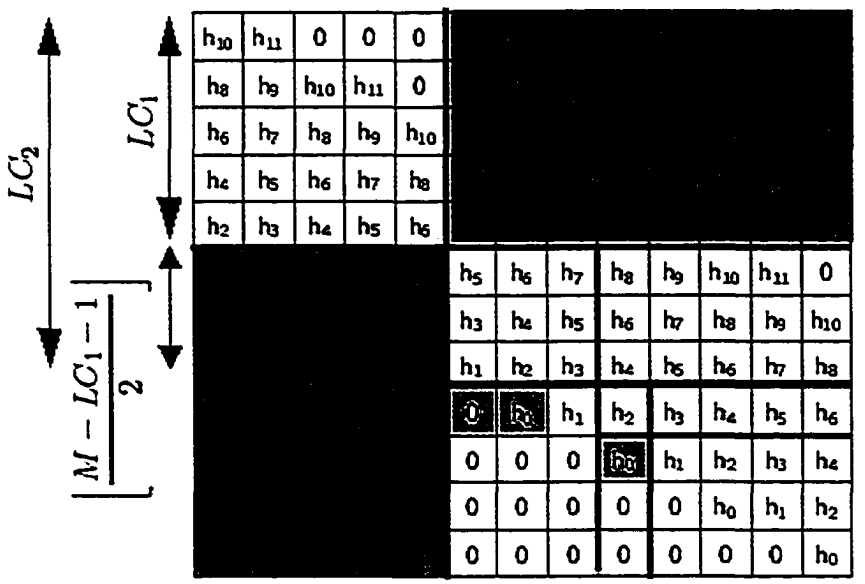

Figure 6.11: $\mathbf{D_2H_2}$ left corner sub matrix, for $M = 12$.

$$LC_2 = LC_1 + \left\lfloor \frac{M - LC_1 - 1}{2} \right\rfloor. \tag{6.16}$$

The argument used to compute LC_2 can be extended for a chain of three or more decimators. Hence as it is shown in Figure 6.8 the size of the upper left corner matrix which represents the transient effect of the system will be given by

$$LC_n = \begin{cases} LC_{n-1} + \left\lfloor \frac{M - LC_{n-1} - 1}{2} \right\rfloor & \text{, if } \left\lfloor \frac{M - LC_{n-1} - 1}{2} \right\rfloor > 0 \\ LC_{n-1} & \text{, otherwise.} \end{cases} \tag{6.17}$$

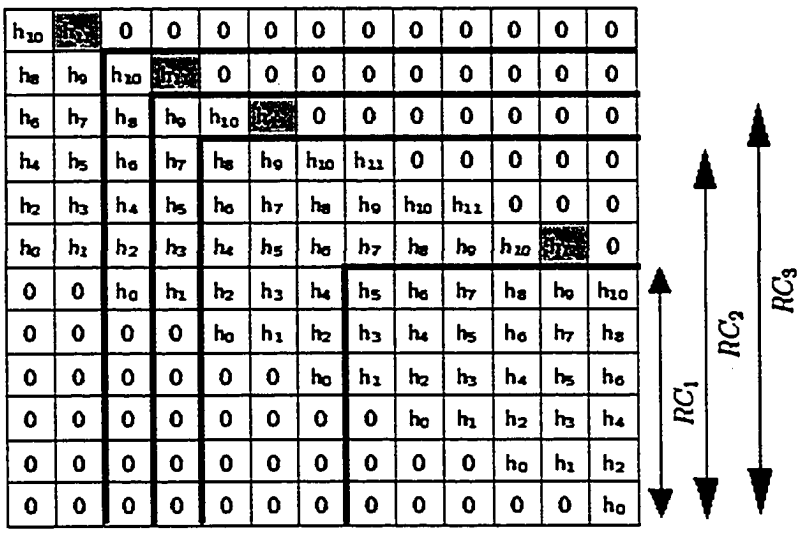

Figure 6.12: $\mathbf{D_nH_n}$ lower right corner with N odd.

The analysis of the lower right corner is similar. However, remember the size of RC_1 depends on the length of the input signal. If the input signal length N is odd the last element in the lower right corner is h_0. This is shown for $M = 12$ in Figure 6.12. The recursion in this case will be given by:

$$RC_n = \begin{cases} RC_{n-1} + \left\lfloor \frac{M - RC_{n-1}}{2} \right\rfloor & \text{,if } \left\lfloor \frac{M - RC_{n-1}}{2} \right\rfloor > 0 \\ RC_{n-1} & \text{,otherwise,} \end{cases} \tag{6.18}$$

with $RC_1 = \lfloor \frac{M-1}{2} \rfloor + 1$.
For N even the last element in the lower right corner is h_2. This is shown for $M = 12$ in Figure 6.13.The recursion in this case will be given by:

$$RC_n = \begin{cases} RC_{n-1} + \left\lfloor \frac{M - RC_{n-1} - 2}{2} \right\rfloor & \text{if , } \left\lfloor \frac{M - RC_{n-1} - 2}{2} \right\rfloor > 0 \\ RC_{n-1} & \text{Otherwise,} \end{cases} \tag{6.19}$$

with $RC_1 = \lfloor \frac{M-1}{2} \rfloor$.

6.2.3 Matrix Structure Examples

To illustrate the structure of the correlation matrix $\mathbf{R}_{y_1}$, we will look at some examples. [3] Suppose $\sigma^2 = 1, N = 8$ and $M = 8$, so $L = 7$, $LC_1 = 3$, $IM_1 = 1$ and $RC_1 = 3$. Note

[3]Numerical calculations of these examples are shown in the Appendix A.

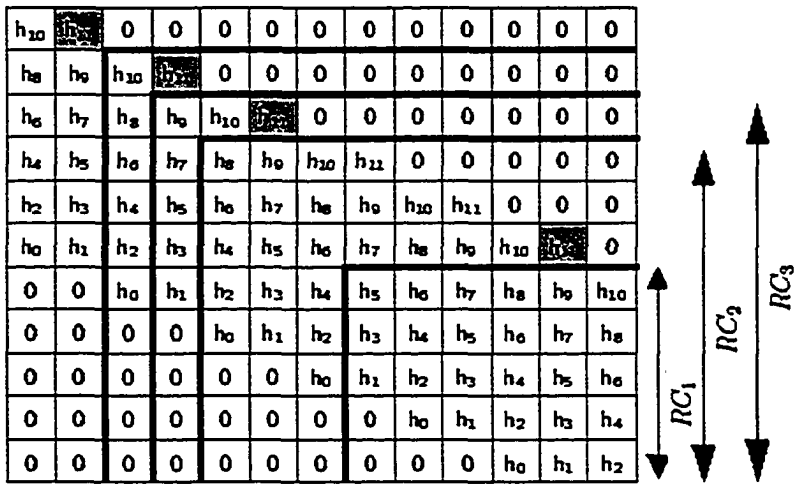

Figure 6.13: $\mathbf{D_n H_n}$ lower right corner with N even.

in this case if we want to connect the system to another decimator we will get an overlap of LC_2 and RC_2. The next matrices will show the structure of matrix $\mathbf{D_1 H_1 H_1}^H \mathbf{D_1}^H$.

$$\mathbf{D_1 H_1 H_1{}^H D_1{}^H} =$$

$$\begin{bmatrix}
h_{(6,6)+(7,7)} & h_{(6,4)+(7,5)} & h_{(6,2)+(7,3)} & h_{(6,0)+(7,1)} = (0) & \cdots \\
h_{(4,6)+(5,7)} & h_{(4,4)+(5,5)+(6,6)+(7,7)} & h_{(4,2)+(5,3)+(6,4)+(7,5)} & h_{(4,0)+(5,1)+(6,2)+(7,3)} = (0) & \cdots \\
h_{(2,6)+(3,7)} & h_{(2,4)+(3,5)+(4,6)+(5,7)} & h_{(2,2)+(3,3)+(4,4)+(5,5)+(6,6)+(7,7)} & h_{(2,0)+(3,1)+(4,2)+(5,3)+(6,4)+(7,5)} = (0) & \cdots \\
h_{(0,6)+(1,7)} = (0) & h_{(0,4)+(1,5)+(2,6)+(3,7)} = (0) & h_{(0,2)+(1,3)+(2,4)+(3,5)+(4,6)+(5,7)} = (0) & h_{(0,0)+(1,1)+(2,2)+(3,3)+(4,4)+(5,5)+(6,6)+(7,7)} = (1) & \cdots \\
0 & h_{(0,6)+(1,7)} = (0) & h_{(0,4)+(1,5)+(2,6)+(3,7)} = (0) & h_{(0,2)+(1,3)+(2,4)+(3,5)+(4,6)+(5,7)} = (0) & \cdots \\
0 & 0 & h_{(0,6)+(1,7)} = (0) & h_{(0,4)+(1,5)+(2,6)+(3,7)} = (0) & \cdots \\
0 & 0 & 0 & h_{(0,6)+(1,7)} = (0) & \cdots
\end{bmatrix}$$

$$\begin{bmatrix}
0 & 0 & 0 \\
h_{(6,0)+(7,1)} = (0) & 0 & 0 \\
h_{(4,0)+(5,1)+(6,2)+(7,3)} = (0) & h_{(6,0)+(7,1)} = (0) & 0 \\
h_{(2,0)+(3,1)+(4,2)+(5,3)+(6,4)+(7,5)} = (0) & h_{(4,0)+(5,1)+(6,2)+(7,3)} = (0) & h_{(6,0)+(7,1)} = (0) \\
h_{(0,0)+(1,1)+(2,2)+(3,3)+(4,4)+(5,5)} & h_{(2,0)+(3,1)+(4,2)+(5,3)} & h_{(4,0)+(5,1)} \\
h_{(0,2)+(1,3)+(2,4)+(3,5)} & h_{(0,0)+(1,1)+(2,2)+(3,3)} & h_{(2,0)+(3,1)} \\
h_{(0,4)+(1,5)} & h_{(0,2)+(1,3)} & h_{(0,0)+(1,1)}
\end{bmatrix},$$

The numbers surrounded by parentheses are obtained by applying (6.9) to the matrix. Now suppose $\sigma^2 = 1, N = 8$ and $M = 4$, so $L = 5$, and the structure of matrix $\mathbf{D_1 H_1 H_1}^H \mathbf{D_1}^H$ is

$$\mathbf{D_1 H_1 H_1}^H \mathbf{D_1}^H =$$

$$\left[\begin{array}{c:ccc:c} h_{(2,2)+(3,3)} & (0) & 0 & 0 & 0 \\ \hdashline (0) & (1) & (0) & 0 & 0 \\ 0 & (0) & (1) & (0) & 0 \\ 0 & 0 & (0) & (1) & (0) \\ \hdashline 0 & 0 & 0 & (0) & h_{(0,0)+(1,1)+(2,2)} \end{array}\right]$$

In this case $LC_1 = 1$, $IM_1 = 3$ and $RC_1 = 1$. Connecting the system to another decimator we will get $LC_2 = LC_1 + \lfloor \frac{M-LC_1-1}{2} \rfloor = 2$ and $RC_2 = RC_1 = 1$ since $\lfloor \frac{M-RC-1-2}{2} \rfloor = 0$.

6.2.4 Correlation Matrix Inverse

If we have a chain of n decimators with WGN as an input the output correlation matrix will be given by:

$$\mathbf{R}_{y_n} = \sigma^2 \mathbf{D_n H_n} \cdots \mathbf{D_2 H_2 D_1 H_1 H_1}^H \mathbf{D_1}^H \mathbf{H_2}^H \mathbf{D_2}^H \cdots \mathbf{H_n}^H \mathbf{D_n}^H. \tag{6.20}$$

It is of interest to find if this matrix possesses an inverse. In order to solve this question we will start with the matrix

$$\sigma^2 \mathbf{D_1 H_1 I_1 H_1}^H \mathbf{D_1}^H, \tag{6.21}$$

where $\mathbf{I_1}$ is the identity matrix. Notice matrices $\mathbf{D_1}$ and $\mathbf{H_1}$ are full rank matrices. This means all their rows are independent. Notice also that for all nonzero vectors $\mathbf{x}$

$$\mathbf{x}^H \mathbf{I_1 x} > 0.$$

Due to this property, matrix $\mathbf{I_1}$ is part of a group of matrices known as positive definite matrices. It is important to note that a positive definite matrix always has an inverse

matrix [24]. Horn [24] also showed that the product

$$\mathbf{C}\mathbf{A}\mathbf{C}^H,$$

where $\mathbf{A}$ is a positive definite matrix and $\mathbf{C}$ is a full rank matrix is a positive definite matrix. Hence, the product

$$\sigma^2\mathbf{H}_1\mathbf{I}_1{\mathbf{H}_1}^H$$

is positive definite. Using this argument we can also conclude (6.21) is positive definite. Furthermore, the argument can be used in a recursive manner to show that the correlation matrix is positive definite, and then has an inverse.

6.2.5 Correlation Matrix Eigenvalue Bounds

In Section 6.2.2 we showed the matrix given in Equation (6.20) is usually real and symmetric as it is derived from the Daubechies construction method for wavelets. Horn and Johnson [24] show that this matrix can be represented as

$$\mathbf{D}_1\mathbf{H}_1{\mathbf{H}_1}^H{\mathbf{D}_1}^H = \mathbf{U}\Lambda\mathbf{U}^H, \tag{6.22}$$

where here $n = 1$, and $\mathbf{U}$ is a square unitary matrix and Λ is a diagonal matrix with the eigenvalues of $\mathbf{D}_1\mathbf{H}_1{\mathbf{H}_1}^H{\mathbf{D}_1}^H$ Horn and Johnson also show that these eigenvalues are real and can be arranged from minimum to maximum as

$$\lambda_{min} = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{L_1} = \lambda_{max} \tag{6.23}$$

Using (6.23) and a unitary invariant matrix norm [4] , we can put a bound on the norm of the matrix given in Equation (6.22) as

$$\lambda_{min} = \lambda_{min}\,|||\mathbf{U}\mathbf{U}^H||| \leq |||\mathbf{U}\Lambda\mathbf{U}^H||| \leq \lambda_{max}\,|||\mathbf{U}\mathbf{U}^H||| = \lambda_{max}. \tag{6.24}$$

The output of the first decimator will be the input of the second one, and so upon applying (6.24) we can determine a bound on the norm of the correlation matrix of the

[4] A unitary invariant matrix norm has the property $|||\mathbf{U}\mathbf{A}||| = |||\mathbf{A}|||$. Some examples of these norms are the Frobenius and the spectral norms.

output of a two decimator chain as

$$\begin{aligned}
|||\mathbf{R}_y||| &= \sigma^2 |||\mathbf{D_2 H_2 D_1 H_1 H_1}^H \mathbf{D_1}^H \mathbf{H_2}^H \mathbf{D_2}^H||| \\
&= \sigma^2 |||\mathbf{D_2 H_2 U_1 \Lambda_1 U_1}^T \mathbf{H_2}^H \mathbf{D_2}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} |||\mathbf{D_2 H_2 U_1 U_1}^H \mathbf{H_2}^H \mathbf{D_2}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} |||\mathbf{D_2 H_2 I_1 H_2}^H \mathbf{D_2}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} |||\mathbf{D_2 H_2 H_2}^H \mathbf{D_2}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} |||\mathbf{U_2 \Lambda_2 U_2}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} \lambda_{max}^{\mathbf{H_2}} |||\mathbf{U_2 U_2}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} \lambda_{max}^{\mathbf{H_2}},
\end{aligned} \tag{6.25}$$

where $\lambda_{xxx}^{\mathbf{H_p}}$ represents the xxx eigenvalue of the matrix $\mathbf{D_p H_p H_p}^H \mathbf{D_p}^H$. If we consider a chain of n decimators, the output bound will be given by

$$\begin{aligned}
|||\mathbf{R}_y||| &= \sigma^2 |||\mathbf{D_n H_n} \ldots \mathbf{D_2 H_2 D_1 H_1 H_1}^H \mathbf{D_1}^H \mathbf{H_2}^H \mathbf{D_2}^H \ldots \mathbf{H_n}^H \mathbf{D_n}^H||| \\
&= \sigma^2 |||\mathbf{D_n H_n} \ldots \mathbf{D_2 H_2 U_1 \Lambda_1 U_1}^T \mathbf{H_2}^H \mathbf{D_2}^H \ldots \mathbf{H_n}^H \mathbf{D_n}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} |||\mathbf{D_n H_n} \ldots \mathbf{D_2 H_2 U_1 I_0 U_1}^H \mathbf{H_2}^H \mathbf{D_2}^H \ldots \mathbf{H_n}^H \mathbf{D_n}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} |||\mathbf{D_n H_n} \ldots \mathbf{D_2 H_2 I_1 H_2}^H \mathbf{D_2}^H \ldots \mathbf{H_n}^H \mathbf{D_n}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} |||\mathbf{D_n H_n} \ldots \mathbf{U_2 \Lambda_2 U_2}^H \mathbf{H_n}^H \mathbf{D_n}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} \lambda_{max}^{\mathbf{H_2}} \ldots \lambda_{max}^{\mathbf{H_{n-1}}} |||\mathbf{D_n H_n H_n}^H \mathbf{D_n}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} \lambda_{max}^{\mathbf{H_2}} \ldots \lambda_{max}^{\mathbf{H_{n-1}}} |||\mathbf{U_n \Lambda_n U_n}^H||| \\
&\leq \sigma^2 \lambda_{max}^{\mathbf{H_1}} \lambda_{max}^{\mathbf{H_2}} \lambda_{max}^{\mathbf{H_3}} \ldots \lambda_{max}^{\mathbf{H_{n-1}}} \lambda_{max}^{\mathbf{H_n}}.
\end{aligned} \tag{6.26}$$

In a similar manner the lower bounds can be calculated. Using these results, lower and upper bounds for the unitary invariant matrix norm of the correlation matrix of the output of a chain of decimators can be obtained as

$$\sigma^2 \lambda_{min}^{\mathbf{H_n}} \cdots \lambda_{min}^{\mathbf{H_1}} \leq \sigma^2 |||\mathbf{D_n H_n} \cdots \mathbf{D_1 H_1 H_1}^H \mathbf{D_1}^H \cdots \mathbf{H_n}^H \mathbf{D_n}^H||| \leq \sigma^2 \lambda_{max}^{\mathbf{H_n}} \cdots \lambda_{max}^{\mathbf{H_1}} \tag{6.27}$$

In practice the upper bound is close to σ^2, and the lower bound goes to zero as the number of decimators increases due to the fact that the elements of the upper left corner submatrix of the correlation matrix are close to zero. In Figure 6.14 we can see the lower bound for a chain of decimators with $\sigma^2 = 1$.

Note how the ratio between the maximum and the minimum bound goes to infinity as the number of connected decimators increases. This behavior suggests the condition

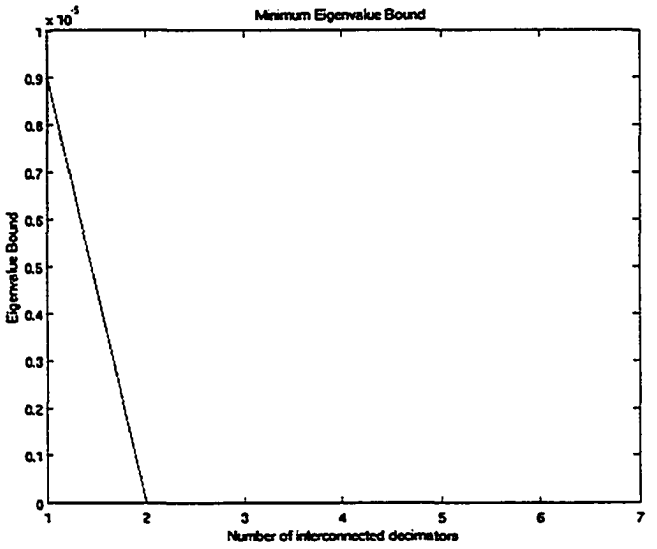

Figure 6.14: Minimum eigenvalue bound.

number is big. Hence, it is possible that the inversion problem is ill-conditioned. Figure 6.15 shows the structure of the correlation matrix $\mathbf{R}_{y_7}$ with $N = 320$ and $M = 8$. In this case the upper left corner and the lower right corner overlapped. Notice how a lot of the terms in the upper left corner have a zero or close to zero value. In this case the correlation matrix $\mathbf{R}_{y_7}$ inverse cannot be computed using Matlab due to the ill-conditioning.

6.3 Conclusion

In this Chapter, the structure of the output correlation matrix for a chain of decimators when the input is white Gaussian noise has been analyzed. The structure obtained has three diagonal submatrices. A recursive formula has been given to obtain the size of these submatrices. In addition, the existence of an inverse correlation matrix has been discussed. However, it has also been shown that the matrix inversion problem is ill-conditioned.

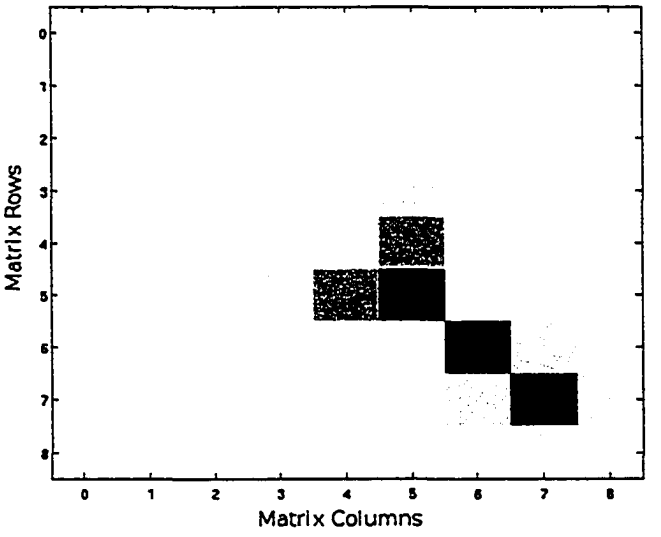

Figure 6.15: Correlation matrix for a chain of seven decimators in gray-scale with $M = 8$, and $N = 320$.

Chapter 7

Quasi-positive Sampling in Wavelet Subspaces

In general, the approximation error between a signal f and its projection f_n decreases more rapidly as $n \to \infty$ when we use a wavelet projection than when we use a Fourier series representation [1]. However, even with this behavior a lot of the wavelet families constructed present Gibbs' phenomenon. In general, as it was shown by Shim and Volkmer [26], any wavelet scaling function $\phi(t)$ with $\phi'(t) \neq 0$ for a dyadic number [1] t that satisfies

$$\phi(t) \leq C(1 + |t|)^{-\beta},$$

with t and $C \in \mathbb{R}$, and $\beta > 3$. Then the wavelet projection presents the Gibbs phenomenon on the right or the left of the point $t = 0$. Most of the wavelets used presently satisfy this condition. In particular, Daubechies' wavelets present this behavior. Furthermore, as it was shown in Section 4.5, the Daubechies wavelet projections present the Gibbs' phenomenon on the right or the left of any discontinuity located at a dyadic point.

There have been some attempts to eliminate the Gibbs' phenomenon from the wavelet projections using thresholding and averaging of the projection coefficients from various translations of the input signal [6] [7]. We will present another method to eliminate the Gibbs' overshoot in wavelet projections using compact support wavelets based in the work of Walter and Shen [27].

[1] A dyadic number is a number that can be written as $k2^p$ where $k, p \in \mathbb{Z}$.

7.1 Positive Delta Wavelet Sequences

Every scaling function with compact support possesses the partition of unity property. Recall this property is

$$\sum_n \phi(t-n) = 1, \quad t \in \mathbb{R}.$$

Definition 7.1.1 (Abel Summability Function). The Abel summability function of a compact support scaling function $\phi(t)$ is given by

$$P_r(t) = \sum_n r^{|n|}\phi(t-n) \text{ for } 0 < r \leq 1. \tag{7.1}$$

Walter and Shen [27] built the positive delta sequence

$$k_{r,m}(s,t) = 2^m k_r(2^m s, 2^m t), \tag{7.2}$$

where

$$k_r(s,t) = \left(\frac{1-r}{1+r}\right)^2 \sum_{n=-\infty}^{\infty} P_r(s-n)P_r(t-n). \tag{7.3}$$

With this kernel we obtain a nonnegative and uniformly convergent approximation to $f(t) \in L^1(\mathbb{R}) \cap L^2(\mathbb{R})$ when $m \to \infty$. This approximation is given by

$$f_m^r(t) = \int_{-\infty}^{\infty} k_{r,m}(s,t)f(s)ds. \tag{7.4}$$

7.2 f_m^r Series Expansion

By construction $f_m^r \in V_m$ and due to this fact there is a b-sequence b_k such that

$$f_m^r(t) = \sum_{n=-\infty}^{\infty} b_n \phi_{m,n}(t). \tag{7.5}$$

In [28] these coefficients are found to be

$$b_n = \left(\frac{1-r}{1+r}\right)^2 \sum_{l=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} r^{|l|}r^{|k|}\langle f(t), \phi_{m,n+k-l}(t)\rangle. \tag{7.6}$$

We may only compute b_n approximately. A truncation in the sums is necessary to calculate these coefficients directly from (7.6). A feasible truncation is proposed in [28]. However, this truncation will give rise to an error in the computation of these coefficients.

7.3 Bounded Kernel Definition

As it has been seen in section 7.1, Walter gave a method to obtain a positive sampling delta sequence using wavelet scaling functions. The projection of a function using this delta sequence will not present Gibbs' phenomenon. However, the only practical form of computing the projection coefficients is through a truncation in a sum as it is shown in section 7.2. The computation of the projection using this method is a computationally expensive task. Due to this reason we modified the theory of Walter and created a new delta sequence which does not present Gibbs' phenomenon for functions that are defined on the interval $t \in [0 \ldots \infty)$. This new delta sequence can be computed in a recursive manner. Then, the computational complexity will be reduced.
If a function $f(t)$ to be projected is such that it starts at a given time $t = 0$, the behavior of $f(t)$ for $t < 0$ is no longer an issue and can be discarded. For this reason the following modification to the positive delta sequence definition is proposed:

$$\delta_m(x,y) \geq 0, \text{ as } m \to \infty \text{ and } x, y \in \mathbb{R}^{+2} \tag{7.7i}$$

$$\int_0^\infty \delta_m(x,y)dx \to 1 \text{ uniformly in compact subsets of } \mathbb{R}^+ \text{ as } m \to \infty \tag{7.7ii}$$

$$\text{For each } \gamma > 0, \ \sup_{|x-y|\geq\gamma} \delta_m(x,y) \to 0 \text{ as } m \to \infty \text{ and } x, y \in \mathbb{R}^+. \tag{7.7iii}$$

We seek to modify Walter's kernel to fulfill the constraints imposed in (7.7), and in this way create a new kernel $G_{r,m}(s,t)$.

Expanding Equation (7.2) we have

$$k_{r,m}(s,t) = \left(\frac{1-r}{1+r}\right)^2 2^m \sum_{n=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} r^{|k|}\phi(2^m t - k - n) \sum_{j=-\infty}^{\infty} r^{|j|}\phi(2^m s - j - n). \tag{7.8}$$

If the scaling function has compact support, that is $\operatorname{supp}\phi(t) = [0, M-1]$, then

$$\operatorname{supp}\phi(2^m t - k - n) = \left[\frac{k+n}{2^m}, \frac{k+n+M-1}{2^m}\right], \tag{7.9}$$

and using this equation we can put an upper and lower bound for k and n at a given time t. This bound is given by

$$2^m t \geq k + n \geq 2^m t - M + 1. \tag{7.10}$$

Since we want to characterize functions that are only defined for $t \geq 0$ we will obtain an

[2] $\mathbb{R}^+$ is the subset of nonnegative real numbers.

upper and lower bound for $k+n$ for this case. When $t=0$, (7.10) becomes

$$0 \geq k+n \geq -M+1,$$

and if $t \to \infty$ then $\infty > k+n$, and putting these two inequalities together the bounds for $k+n$ are

$$\infty > k+n \geq -M+1. \tag{7.11}$$

In a similar fashion, we can obtain

$$\infty > j+n \geq -M+1. \tag{7.12}$$

We will introduce the sum

$$\sum_{k=0}^{\infty} \phi(2^m t-k-n) \quad \text{for } t \in \mathbb{R}^+ \text{ and } \infty > k+n \geq -M+1. \tag{7.13}$$

If

$$0 \leq 2^m t-k-n < M \tag{7.14}$$

for a given $t \in \mathbb{R}^+$ the partition of unity property (4.8) tells us the sum in Equation (7.13) is equal to one. This is the case when $m \to \infty$. On the other hand, if for certain $t \in \mathbb{R}^+$ the inequality given in Equation (7.14) is not fulfilled we can still put an upper bound to the sum in Equation (7.13) due to the fact $\phi(t)$ is compactly supported. Combining these results we can conclude

$$\sum_{k=0}^{\infty} \phi(2^m t-k-n) \leq S \quad \text{for } t, S \in \mathbb{R}^+ \text{ and } \infty > k+n \geq -M+1. \tag{7.15}$$

If we only consider Equation (7.13) for a given t in a closed interval $t \in \left[0, \frac{M-1+n}{2^m}\right]$ with $m \to \infty$ the sum limits can be changed obtaining

$$\sum_{k=\lfloor 2^m t-n-M+1 \rfloor}^{\lceil 2^m t-n \rceil} \phi(2^m t-k-n) \to 1 \quad \text{for } \infty > k+n \geq -M+1 \text{ as } m \to \infty^3. \tag{7.16}$$

Since this is a finite function series which converges to a constant value, we can say that Equation (7.16) converges uniformly for $t \in \left[0, \frac{M-1+n}{2^m}\right]$ as $m \to \infty$.

The Abel summability function of series (7.15) for $t \in \mathbb{R}^+$ and $\infty > k+n \geq -M+1$

[3] $\lceil x \rceil$ is the smallest integer such that $\lceil x \rceil \geq x$

is

$$P_{r,n}(2^m t) = \sum_{k=0}^{\infty} r^k \phi(2^m t - k - n) \quad \text{for } 0 < r \leq 1. \tag{7.17}$$

Analogously we can define the Abel summability function for $t \in \mathbb{R}^+$ and $\infty > j + n \geq -M + 1$ as

$$P_{r,n}(2^m t) = \sum_{j=0}^{\infty} r^j \phi(2^m t - j - n) \quad \text{for } 0 < r \leq 1. \tag{7.18}$$

For $j \geq 0$ and $r \in (0,1]$, $f_j(r) = r^j$ is a bounded decreasing function [4] , and for $t \in \left[0, \frac{M-1+n}{2^m}\right]$ Equation (7.16) converges to one when $m \to \infty$. Therefore, the Abel summability functions (7.17) and (7.18) converge uniformly for $r \in (0,1]$, $t \in \left[0, \frac{M-1+n}{2^m}\right]$, and $m \to \infty$. Consequently there exists a $0 < r_0 \leq 1$ such that $P_{r,n}(2^m t) \geq \frac{1}{2}$ for $r \geq r_0$, $t \in \left[0, \frac{M-1+n}{2^m}\right]$, and $m \to \infty$. In addition, if now $t = \frac{M-1+n}{2^m} + \frac{l}{2^{m+v}}$, where $l \geq 0$, $v \geq 0$ $(l, v \in \mathbb{Z})$, then

$$\begin{aligned}
P_{r,n}(2^m t) &= \sum_{j=0}^{\infty} r^j \phi\left(2^m t - j - n\right) \\
&= \sum_{j=0}^{\infty} r^j \phi\left(2^m \left(\frac{M-1+n}{2^m} + \frac{l}{2^{m+v}}\right) - j - n\right) \\
&= \sum_{j=0}^{\infty} r^j \phi\left(M - 1 + n + \frac{l}{2^v} - j - n\right) \\
&= \sum_{j=0}^{\infty} r^j \phi\left(M - 1 + \frac{l}{2^v} - j\right) \\
&= \sum_{p=-\lfloor \frac{l}{2^v} \rfloor}^{\infty} r^{p+\lfloor \frac{l}{2^v} \rfloor} \phi\left(M - 1 + \frac{l}{2^v} - \left\lfloor \frac{l}{2^v} \right\rfloor - p\right) \qquad p = j - \left\lfloor \frac{l}{2^v} \right\rfloor \\
&= \sum_{p=-\lfloor \frac{l}{2^v} \rfloor}^{-1} r^{p+\lfloor \frac{l}{2^v} \rfloor} \phi\left(\underbrace{M - 1 + \frac{l}{2^v} - \left\lfloor \frac{l}{2^v} \right\rfloor - p}_{\geq M}\right) \\
&\quad + \sum_{p=0}^{\infty} r^{p+\lfloor \frac{l}{2^v} \rfloor} \phi\left(\underbrace{M - 1 + \frac{l}{2^v} - \left\lfloor \frac{l}{2^v} \right\rfloor - p}_{< M}\right),
\end{aligned} \tag{7.19}$$

[4] A bounded decreasing function is a function where $f_{j+1}(r) \leq f_j(r)$ for all j and also $0 \leq f_j(r) \leq B$ for all $r \in [a,b]$ for some $a, b \in \mathbb{R}$.

since $\operatorname{supp}\phi = [0 \ldots M-1]$ then this equation becomes

$$\begin{aligned} P_{r,n}(2^m t) &= \sum_{p=-\lfloor \frac{l}{2^v} \rfloor}^{-1} r^{p+\lfloor \frac{l}{2^v} \rfloor} \underbrace{\phi\left(M-1+\frac{l}{2^v}-\left\lfloor \frac{l}{2^v} \right\rfloor - p\right)}_{=0} \\ &+ \sum_{p=0}^{\infty} r^{p+\lfloor \frac{l}{2^v} \rfloor} \phi\left(M-1+\frac{l}{2^v}-\left\lfloor \frac{l}{2^v} \right\rfloor - p\right) \\ &= \sum_{p=0}^{\infty} r^{p+\lfloor \frac{l}{2^v} \rfloor} \phi\left(M-1+\frac{l}{2^v}-\left\lfloor \frac{l}{2^v} \right\rfloor - p\right) \\ &= r^{\lfloor \frac{l}{2^v} \rfloor} \sum_{p=0}^{\infty} r^p \phi\left(M-1+\frac{l}{2^v}-\left\lfloor \frac{l}{2^v} \right\rfloor - p\right) \\ &= r^{\lfloor \frac{l}{2^v} \rfloor} P_{r,n}\left(2^m\left(\frac{M-1+n}{2^m}+\frac{l}{2^{v+m}}-\left\lfloor \frac{l}{2^v} \right\rfloor \frac{1}{2^m}\right)\right). \end{aligned} \tag{7.20}$$

Since $\left(\frac{M-1+n}{2^m}+\frac{l}{2^{v+m}}-\left\lfloor \frac{l}{2^v} \right\rfloor \frac{1}{2^m}\right) \in \left[0, \frac{M-1+n}{2^m}\right]$, we can conclude that $P_{r,n}(2^m t) \geq 0$ for $t > \frac{M-1+n}{2^m}$, and $m \to \infty$. Combining this result with the Abel mean convergence for $t \in [0, \frac{N+n}{2^m}]$, we obtain $P_{r,n}(2^m t) \geq 0$ for $\infty > j+n \geq -M+1, \infty > k+n \geq -M+1$, $t \in \mathbb{R}^+$, and $m \to \infty$.

Equations (7.17) and (7.18) are linear combinations of the scaling function $\phi(2^m t)$ which depend on the value of n. Due to this fact, as shown in [27], they are rapidly decreasing functions which obey:

$$|P_{r,n}(2^m t)| \leq C_p (1+|2^m t - n|)^{-p}, \quad p \in \mathbb{Z}^+, \quad t \in \mathbb{R}^+ \tag{7.21}$$

for a suitable constant C_p.

We will define a new delta sequence $G_{r,m}(s,t)$ as

$$G_{r,m}(s,t) = 2^m (1-r)^2 \sum_{n=-M+1}^{\infty} \sum_{k=0}^{\infty} r^k \phi(2^m t - k - n) \sum_{j=0}^{\infty} r^j \phi(2^m s - j - n) \tag{7.22}$$

Moreover, the new delta sequence is constructed from Equations (7.17) and (7.18), since these sequences have the positive property (7.7i).

The proof of property (7.7iii) though similar to the one presented in [27] will be given

here for the sake of clarity and completeness:

$$
\begin{aligned}
G_{r,m}(s,t) &= 2^m(1-r)^2 \sum_{n=-M+1}^{\infty} P_{r,n}(2^m t) P_{r,n}(2^m s) \\
&\le C^2 2^m \sum_{n=-M+1}^{\infty} \frac{1}{(1+|2^m t-n|)^4} \frac{1}{(1+|2^m s-n|)^4} \text{ via (7.21) with } p=4 \\
&\le C^2 2^m \sum_{n=-M+1}^{\infty} \frac{1}{(1+|2^m t-2^m s|)^2} \frac{1}{(1+|2^m t-n|)^2} \frac{1}{(1+|2^m s-n|)^2}^{5} \\
&\le \frac{C^2 2^m}{(1+|2^m t-2^m s|)^2} \underbrace{\sum_{n=-M+1}^{\infty} \frac{1}{(1+|2^m t-n|)^2} \frac{1}{(1+|2^m s-n|)^2}}_{=H(t,s)\le H(0,0)} \\
&\le \frac{C^2 2^m H(0,0)}{(1+2^m|t-s|)^2} \le \frac{C' 2^m}{(1+2^m|t-s|)^2} \\
&\le \frac{C' 2^m}{(1+2^m\gamma)^2} \to 0 \text{ for } |t-s| \ge \gamma \text{ as } m \to \infty.
\end{aligned}
\tag{7.23}
$$

Now, let us see if property (7.7ii) holds. We want

$$\int_0^{\infty} G_{r,m}(s,t)ds = 1.$$

Expanding the left side of this equation we have

$$2^m(1-r)^2 \int_0^{\infty} \sum_{n=-M+1}^{\infty} \sum_{k=0}^{\infty} r^k \phi(2^m t-k-n) \sum_{j=0}^{\infty} r^j \phi(2^m s-j-n)ds$$

Changing the order of the integral and the sum we obtain

$$
\begin{aligned}
&(1-r)^2 \sum_{n=-M+1}^{\infty} \sum_{k=0}^{\infty} r^k \phi(2^m t-k-n) \sum_{j=0}^{\infty} r^j 2^m \int_{-\infty}^{\infty} \phi(2^m s-j-n)ds \\
&= (1-r)^2 \sum_{n=-M+1}^{\infty} \sum_{k=0}^{\infty} r^k \phi(2^m t-k-n) \sum_{j=0}^{\infty} r^j \int_{-\infty}^{\infty} \phi(p)dp,
\end{aligned}
\tag{7.24}
$$

where $p = 2^m s - j - n$ and $dp = 2^m ds$.

[5] The identity $\frac{1}{(1+|t-s|)} \ge \frac{1}{(1+|t-n|)} \frac{1}{(1+|s-n|)}$ is used here.

As it is known that $\int_{-\infty}^{\infty}\phi(p)dp = 1$, Equation (7.24) becomes:

$$\begin{aligned}
&(1-r)^2 \sum_{n=-M+1}^{\infty}\sum_{k=0}^{\infty} r^k \phi(2^m t-k-n)\sum_{j=0}^{\infty} r^j \\
&= (1-r)^2 \sum_{n=-M+1}^{\infty}\sum_{k=0}^{\infty} r^k \phi(2^m t-k-n)\frac{1}{(1-r)} \\
&= (1-r)\sum_{k=0}^{\infty} r^k \underbrace{\sum_{n=-M+1}^{\infty}\phi(2^m t-k-n)}_{=1 \text{ as } m\to\infty} \text{ for } k+n\in[-M+1,\infty) \text{ and } t\in\mathbb{R}^+ \qquad (7.25)\\
&= (1-r)\sum_{k=0}^{\infty} r^k \\
&= \frac{(1-r)}{(1-r)} = 1,
\end{aligned}$$

and property (7.7ii) holds.

7.4 Computation of b_n

Using Equation (7.4) with $G_{r,m}$ instead of $k_{r,m}$ we have

$$f_m^r(t) = \int_0^{\infty} G_{r,m}(s,t)f(s)ds. \qquad (7.26)$$

By construction $f_m^r \in V_m$, and due to this fact there is a b-sequence b_k such that

$$f_m^r(t) = \sum_{n=-\infty}^{\infty} b_n \phi_{m,n}(t). \qquad (7.27)$$

The coefficients b_n are given by

$$\begin{aligned} b_n &= \langle f_m^r(t), \phi_{m,n}(t) \rangle \\ &= \int_{-\infty}^{\infty} f_m^r(t)\phi_{m,n}(t)dt \\ &= \int_{-\infty}^{\infty}\int_{0}^{\infty} G_{r,m}(s,t)f(s)ds \quad \phi_{m,n}(t)dt \\ &= \int_{0}^{\infty}\int_{-\infty}^{\infty} G_{r,m}(s,t)f(s)ds \quad \phi_{m,n}(t)dt \\ &= \int_{0}^{\infty} \underbrace{\left[\int_{-\infty}^{\infty} G_{r,m}(s,t)\phi_{m,n}(t)dt\right]}_{=g(s)} f(s)ds \end{aligned} \tag{7.28}$$

Using (7.22) we have

$$\begin{aligned} g(s) &= 2^{3m/2}(1-r)^2 \sum_{l=-M+1}^{\infty}\sum_{k=0}^{\infty} r^k \left[\int_{-\infty}^{\infty} \phi(2^m t - k - l)\phi(2^m t - n)dt\right] \sum_{j=0}^{\infty} r^j \phi(2^m s - j - l) \\ &= 2^{m/2}(1-r)^2 \sum_{l=-M+1}^{\infty}\sum_{k=0}^{\infty} r^k \delta_{n,k+l} \sum_{j=0}^{\infty} r^j \phi(2^m s - j - l) \\ &= 2^{m/2}(1-r)^2 \sum_{k=0}^{\infty} r^k \sum_{l=-M+1}^{\infty} \delta_{n,k+l} \sum_{j=0}^{\infty} r^j \phi(2^m s - j - l) \\ &= 2^{m/2}(1-r)^2 \sum_{k=0}^{\infty} r^k \sum_{j=0}^{\infty} r^j \phi(2^m s - j - n + k) \text{ for } n \in [-M+1, \infty). \end{aligned} \tag{7.29}$$

Note that Equation (7.29) limits b_n to the interval $n \in [-M+1, \infty)$ which implies supp $f_m^r(t) = [0, \infty)$. Since supp $f(t) = [0, \infty)$ this result is expected.
Substituting (7.29) into (7.28) yields

$$\begin{aligned} b_n &= \int_0^{\infty} 2^{m/2}(1-r)^2 \sum_{k=0}^{\infty} r^k \sum_{j=0}^{\infty} r^j \phi(2^m s - j - n + k) f(s)ds \\ &= 2^{m/2}(1-r)^2 \sum_{k=0}^{\infty} r^k \sum_{j=0}^{\infty} r^j \int_0^{\infty} \phi(2^m s - j - n + k) f(s)ds \\ &= (1-r)^2 \sum_{k=0}^{\infty} r^k \underbrace{\sum_{j=0}^{\infty} r^j \underbrace{\langle f(t), \phi_{m,j+n-k}(t)\rangle}_{=f_{m,j+n-k}}}_{g_{n-k}}, \end{aligned} \tag{7.30}$$

7.5 Truncation Error Bound

Calculating the coefficients b_n directly from Equation (7.30) is not computationally feasible. However, applying a truncation in the series can generate an acceptable approximation. We propose the following approximation:

$$b_n \approx (1-r)^2 \sum_{k=0}^{D} r^k \underbrace{\sum_{j=0}^{\infty} r^j \underbrace{\langle f(t), \phi_{m,j+n-k}(t)\rangle}_{=f_{m,j+n-k}}}_{g_{n-k}}. \tag{7.31}$$

Assume that $\operatorname{supp} f(t) = [0,T]$.[6] The series coefficients $\langle f(t), \phi_{m,k}(t)\rangle$ are

$$\begin{aligned} f_{m,k} = \langle f(t), \phi_{m,k}(t)\rangle &= \int_{-\infty}^{\infty} f(t)\phi_{m,k}(t)dt \\ &= 2^{m/2}\int_0^T f(t)\phi(2^m t - k)dt. \end{aligned} \tag{7.32}$$

Since $\operatorname{supp}\phi_{m,k}(t) = \left[\frac{k}{2^m}, \frac{k+M-1}{2^m}\right]$, $\phi_{m,k}(t)$ overlaps $f(t)$ when

$$\frac{k+M-1}{2^m} \geq 0, \text{ and } \frac{k}{2^m} \leq T.$$

Putting together these two inequalities, we obtain

$$-M+1 \leq k \leq \lfloor 2^m T \rfloor = P. \tag{7.33}$$

From (7.33) and (7.30) the function $f_{m,j+n-k}$ is only defined in the region $j+n-k \in [-M+1, P]$. However, by construction, g_{n-k} is only defined for $j,k \in [0,\infty)$. Hence, if we let $p = n-k$, then g_{n-k} in (7.31) can be simplified as:

$$g_p = \begin{cases} \displaystyle\sum_{j=-M+1-p}^{P-p} r^j \langle f(t), \phi_{m,j+p}(t)\rangle, & \text{if } p \leq M-1 \\ \displaystyle\sum_{j=0}^{P-p} r^j \langle f(t), \phi_{m,j+p}(t)\rangle, & \text{if } p > M-1. \end{cases} \tag{7.34}$$

When $p \leq M-1$, the number of terms in the sum that determines g_p is

$$(P-p) - (-M+1-p) + 1 = P+M,$$

[6] This type of signal is known as a finite duration signal.

and if $p > M-1$ this number is

$$(P-p)+1 < P+M.$$

We will assume that $f_{m,j+n-k}$ is bounded such that $f_{m,j+n-k} \leq B < \infty$. Then, if $p \leq M-1$

$$\begin{aligned} |g_p| &\leq \sum_{j=-M+1-p}^{P-p} r^j \langle f(t), \phi_{m,j+p}(t) \rangle \\ &\leq B \sum_{j=-M+1-p}^{P-p} r^j \\ &< B \sum_{j=-M+1-p}^{P-p} 1 \\ &< B(P+M), \end{aligned} \tag{7.35}$$

and if $p > M-1$

$$\begin{aligned} |g_p| &\leq \sum_{j=0}^{P-p} r^j \langle f(t), \phi_{m,j+p}(t) \rangle \\ &\leq B \sum_{j=0}^{P-p} r^j \\ &< B \sum_{j=0}^{P-p} 1 \\ &< B(P+M). \end{aligned} \tag{7.36}$$

We see that the same bound can be applied in both cases.

Using Equations (7.35) and (7.36), we can find a bound on the truncation error generated by the approximation (7.31). Equation (7.30) can be expressed as

$$b_n = (1-r)^2 \sum_{k=0}^{D} r^k \underbrace{\sum_{j=0}^{\infty} r^j \langle f(t), \phi_{m,j+n-k}(t) \rangle}_{g_{n-k}} + e_n, \tag{7.37}$$

where e_n is given by

$$e_n = (1-r)^2 \sum_{k=D+1}^{\infty} r^k \underbrace{\sum_{j=0}^{\infty} r^j \langle f(t), \phi_{m,j+n-k}(t) \rangle}_{g_{n-k}}. \tag{7.38}$$

Using Equations (7.35) and (7.36), we obtain

$$\begin{aligned} |e_n| &\leq (1-r)^2 \sum_{k=D+1}^{\infty} |r^k||g_{n-k}| \\ &< (1-r)^2 B(P+M) \sum_{k=D+1}^{\infty} r^k \\ &< (1-r)^2 B(P+M) \frac{r^{D+1}}{1-r} = (1-r)B(P+M)r^{D+1}. \end{aligned} \tag{7.39}$$

7.6 Recursive Computation of the b sequence

From Equation (7.30) we have

$$\begin{aligned} g_{n-k} &= \sum_{j=0}^{\infty} r^j f_{m,n-k+j} \\ g_{n-k} &= \sum_{j=-\infty}^{0} r^{-j} f_{m,n-k-j} \\ g_p &= \sum_{j=-\infty}^{\infty} \underbrace{r^{-j} u[-j]}_{=h_j} f_{m,p-j}, \end{aligned} \tag{7.40}$$

where $p = n-k$, $u[n]$ is the unit step function, and $f_{m,k} = \langle f(t), \phi_{m,k}(t) \rangle = 0$ for $k < -M+1$ and $k > P$.

The z-transform of h_j, $H(z) = \mathcal{Z}\{h_j\}$ for $|z| < \frac{1}{r}$ is:

$$\begin{aligned} H(z) &= \sum_{j=-\infty}^{\infty} r^{-j} u[-j] z^{-j} \\ &= \sum_{j=-\infty}^{0} r^{-j} z^{-j} \\ &= \sum_{j=0}^{\infty} r^{j} z^{j} \\ &= \frac{1}{1-rz} \\ &= \frac{-(rz)^{-1}}{1-(rz)^{-1}}. \end{aligned} \tag{7.41}$$

The system $H(z)$ is a noncausal system as can be seen in (7.41), so it is not possible to realize this system in a causal recursive form since it will be unstable. However, g_p can be calculated directly using (7.34) without introducing any error in the computation. This is due to the fact that the sum in Equation (7.34) is finite and possesses a correlation like structure. Hence, algorithms such as the FFT can be implemented to compute g_p in an efficient way. In addition, from Equation (7.30) we have

$$b_n = (1-r)^2 \sum_{k=0}^{\infty} r^k g_{n-k}. \tag{7.42}$$

The z-transform for $c_k = r^k u[k]$, $C(z) = \mathcal{Z}\{c_j\}$ for $|z| > r$ is:

$$\begin{aligned} C(z) &= \sum_{k=-\infty}^{\infty} r^k u[k] z^{-k} \\ &= \frac{1}{1-rz^{-1}} \end{aligned} \tag{7.43}$$

Taking the z-transform of Equation (7.30) with $B(z) = \mathcal{Z}\{b_n\}$ we obtain

$$\begin{aligned} B(z) &= \underbrace{(1-r)^2}_{=\beta} C(z) G(z) \\ &= \beta \frac{1}{1-rz^{-1}} G(z) \end{aligned} \tag{7.44}$$

so

$$B(z) = \beta G(z) + rz^{-1} B(z), \tag{7.45}$$

and upon taking the inverse z-transform we obtain

$$b_{n+1} = \beta g_{n+1} + r b_n. \tag{7.46}$$

If $\operatorname{supp} f(t) = [0,T]$, we may assume $\operatorname{supp} f_m^r(t) = [0,T]$. Since $\operatorname{supp} \phi_{m,n}(t) = \left[\frac{n}{2^m}, \frac{n+M-1}{2^m}\right]$, $\phi_{m,n}(t)$ overlaps $f_m^r(t)$ when

$$\frac{n+M-1}{2^m} \geq 0, \text{ and } \frac{n}{2^m} \leq T.$$

However by construction, as shown by Daubechies [2], $\phi(0) = \phi(M-1) = 0$. Therefore, without loss of generality, we can say

$$\frac{n+M-1}{2^m} > 0 \text{ and } \frac{n}{2^m} < T.$$

Combining these inequalities we obtain

$$-M+2 \leq n \leq \lceil 2^m T \rceil - 1 = R. \tag{7.47}$$

This means that we only need to calculate the coefficients b_n for $n \in [-M+2, R]$. Using Equation (7.31), we can compute b_{-M+2}. Using this result as the initial condition in the recursion (7.46) for $n \in [-M+2, R-1]$, we can obtain the values of $b_n \in [-M+3, R]$.

7.7 Examples

The algorithm was implemented to calculate the approximation to a unit square pulse, shown in Figure 7.1, defined as:

$$u(t) = \begin{cases} 1, & \text{if } 0 < t \leq 1 \\ 0, & \text{otherwise,} \end{cases}$$

and a unit triangle pulse which is shown in Figure 7.5, defined as:

$$f(t) = \begin{cases} t, & \text{if } 0 < t \leq 1 \\ 0, & \text{otherwise.} \end{cases}$$

Our results are shown on Figures 7.3, 7.7, 7.4 and 7.8. For comparison, the results obtained in [28] are also shown on Figures 7.2 and 7.6.

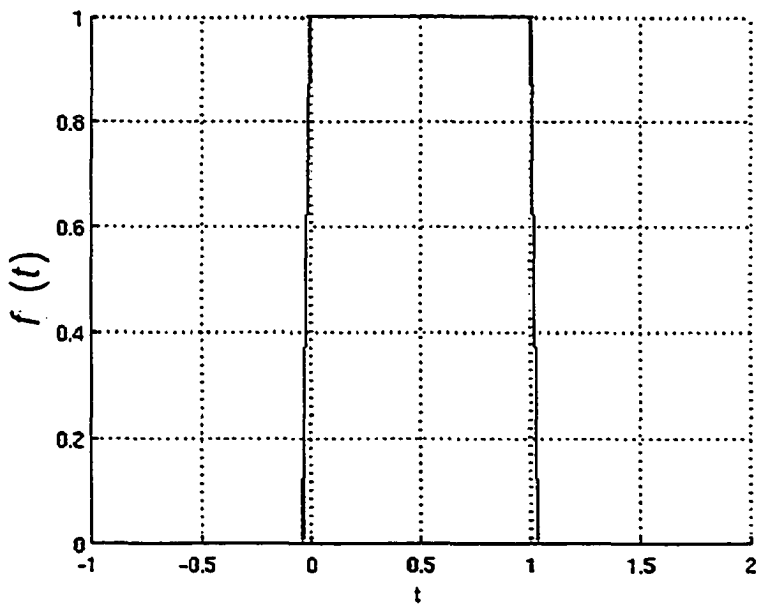

Figure 7.1: Unit square pulse.

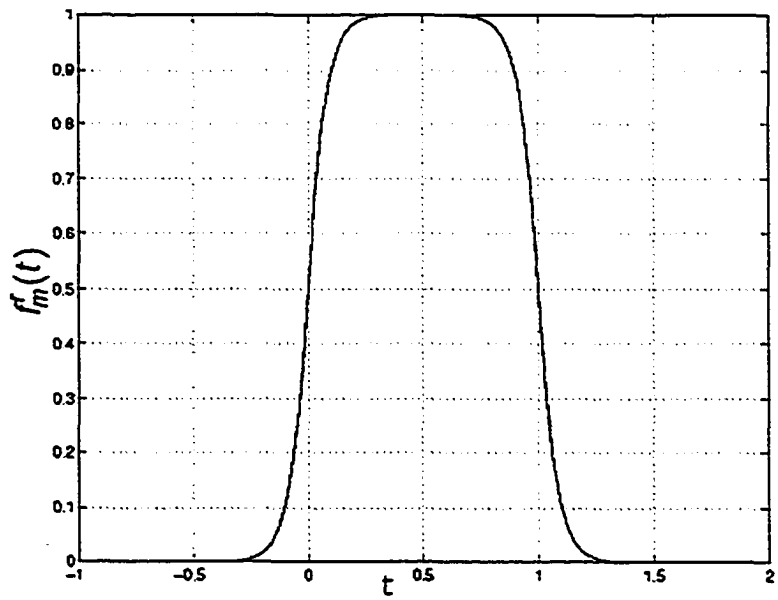

Figure 7.2: Unit square (Walter's approximation m=5, r=0.5).

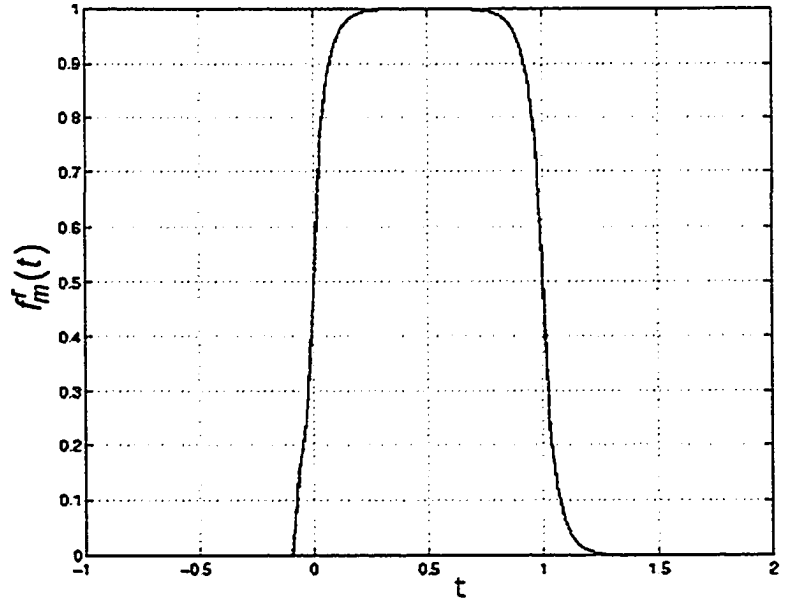

Figure 7.3: Unit square (new truncated approximation m=5, r=0.5).

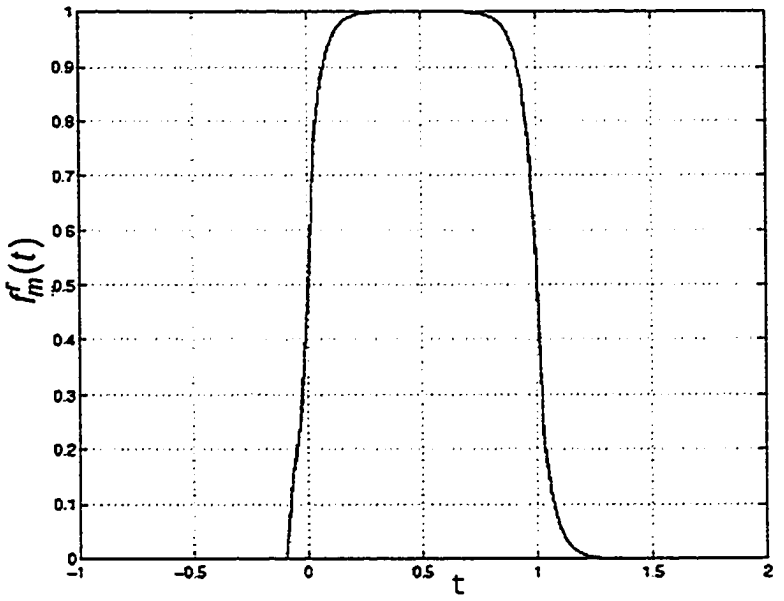

Figure 7.4: Unit square (new recursive approximation m=5, r=0.5).

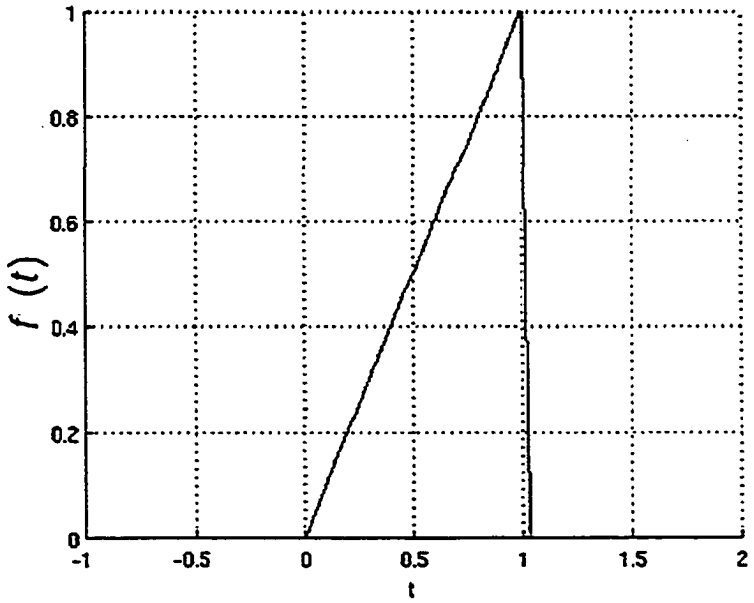

Figure 7.5: Unit triangle pulse.

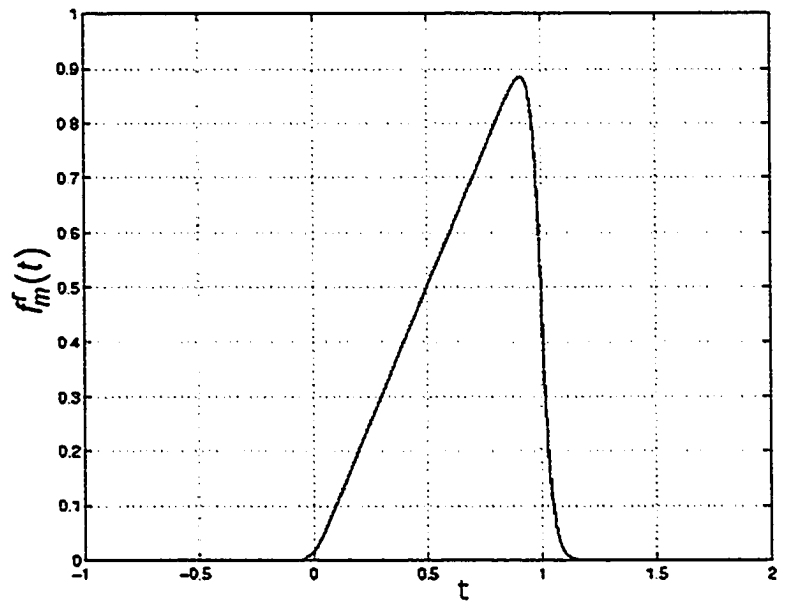

Figure 7.6: Unit triangle (Walter's approximation m=6, r=0.5).

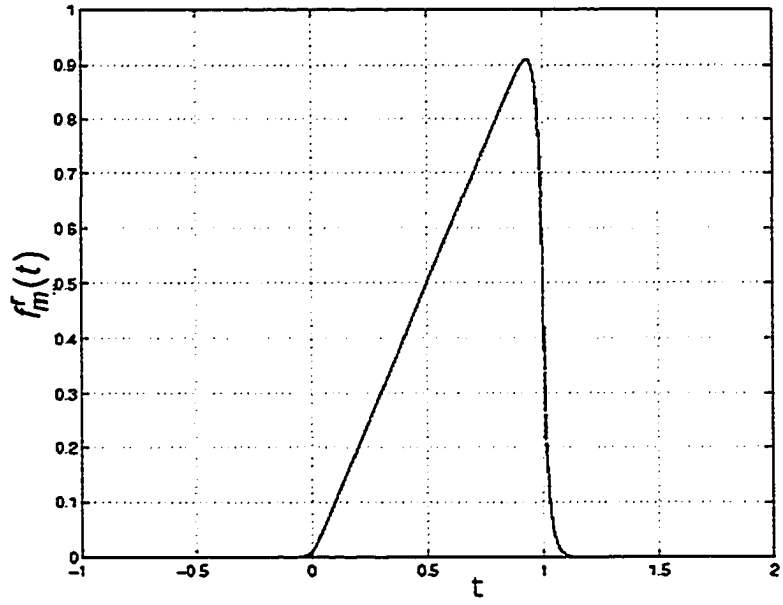

Figure 7.7: Unit triangle (new truncated approximation m=6, r=0.5).

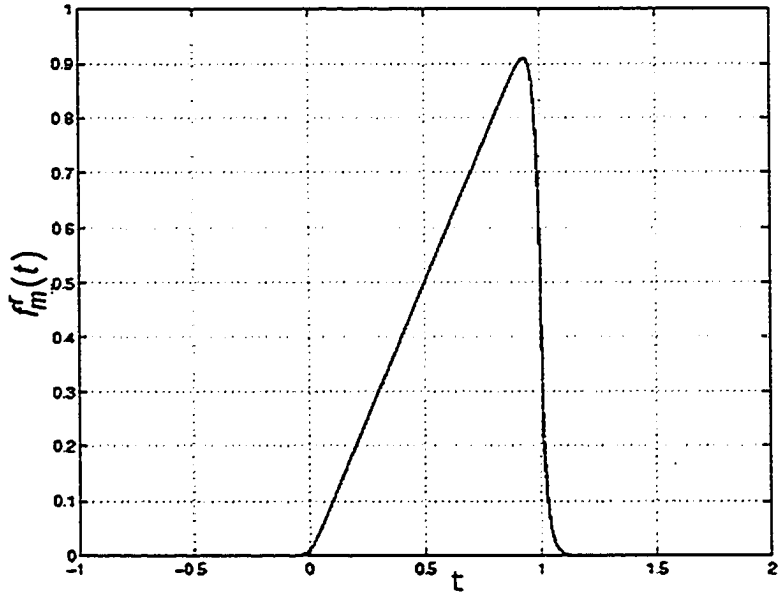

Figure 7.8: Unit triangle (new recursive approximation m=6, r=0.5).

7.8 Positive Sampling Without Integration

The bounded kernel projection solves the problem of Gibbs' phenomenon on wavelet projections. However, we have to compute the inner product integral in order to obtain the projection. Walter [29] proposes the following delta kernel sequence to avoid this problem

$$G_{r,m}(s,t) = 2^m G_r(2^m s, 2^m t) \tag{7.48}$$

where

$$G_r(s,t) = \left(\frac{1-r}{1+r}\right) \sum_{k=-\infty}^{\infty} P_r(t-k)\delta(s-k), \tag{7.49}$$

and $\delta(t)$ is the Dirac delta function. With this kernel, we obtain a nonnegative and uniformly convergent approximation to $f(t) \in L^1(\mathbb{R}) \cap L^2(\mathbb{R})$ when $m \to \infty$. We can compute this kernel approximation using Equation (7.5).

7.9 Positive Sampling f_m^r Series Expansion

By construction $f_m^r \in V_m$. Therefore, we can use Equation (7.6) to obtain the coefficients of the projection. The coefficients b_n are given by (similarly to (7.30))

$$\begin{aligned} b_n &= \langle f_m^r(t), \phi_{m,n}(t) \rangle \\ &= \int_{-\infty}^{\infty} f_m^r(t)\phi_{m,n}(t)dt \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G_{r,m}(s,t)f(s)ds \quad \phi_{m,n}(t)dt \\ &= \int_{-\infty}^{\infty} \underbrace{\left[\int_{-\infty}^{\infty} G_{r,m}(s,t)\phi_{m,n}(t)dt\right]}_{=g(s)} f(s)ds \end{aligned} \tag{7.50}$$

Using (7.48) and (7.49) we have

$$
\begin{aligned}
& g(s) \\
& = 2^m \left(\frac{1-r}{1+r}\right) \int_{-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \delta(2^m s-k) P_r(2^m t-k) \phi_{m,n}(t) dt \\
& = 2^{3m/2} \left(\frac{1-r}{1+r}\right) \sum_{k=-\infty}^{\infty} \delta(2^m s-k) \int_{-\infty}^{\infty} P_r(2^m t-k) \phi(2^m t-n) dt \\
& = 2^{m/2} \left(\frac{1-r}{1+r}\right) \sum_{k=-\infty}^{\infty} \delta(2^m s-k) \sum_{l=-\infty}^{\infty} r^{|l|} \left[2^m \int_{-\infty}^{\infty} \phi(2^m t-k-l) \phi(2^m t-n) dt\right] \text{ (via (7.1))} \\
& = 2^{m/2} \left(\frac{1-r}{1+r}\right) \sum_{k=-\infty}^{\infty} \delta(2^m s-k) \sum_{l=-\infty}^{\infty} r^{|l|} \delta_{n-k-l} \\
& = 2^{m/2} \left(\frac{1-r}{1+r}\right) \sum_{l=-\infty}^{\infty} r^{|l|} \sum_{k=-\infty}^{\infty} \delta(2^m s-k) \delta_{n-k-l} \\
& = 2^{m/2} \left(\frac{1-r}{1+r}\right) \sum_{l=-\infty}^{\infty} r^{|l|} \delta(2^m s-n+l),
\end{aligned}
\tag{7.51}
$$

where δ_t is the Kronecker's delta, while $\delta(t)$ is the Dirac delta function. Substituting (7.51) into (7.50) yields

$$
\begin{aligned}
b_n &= \int_{-\infty}^{\infty} 2^{m/2} \left(\frac{1-r}{1+r}\right) \sum_{l=-\infty}^{\infty} r^{|l|} \delta(2^m s-n+l) f(s) ds \\
&= 2^{m/2} \left(\frac{1-r}{1+r}\right) \sum_{l=-\infty}^{\infty} r^{|l|} \int_{-\infty}^{\infty} \delta(2^m s-n+l) f(s) ds \\
&= 2^{m/2} \left(\frac{1-r}{1+r}\right) \sum_{l=-\infty}^{\infty} r^{|l|} 2^{-m} \int_{-\infty}^{\infty} \delta(p-n+l) f(p/2^m) dp \\
&= 2^{-m/2} \left(\frac{1-r}{1+r}\right) \sum_{l=-\infty}^{\infty} r^{|l|} f\left(\frac{n-l}{2^m}\right).
\end{aligned}
\tag{7.52}
$$

Note $f\left(\frac{n-l}{2^m}\right)$ is $f(t)$ sampled at a rate of $\frac{1}{2^m}$, so b_n can be interpreted as a lowpass filtered version of the sampled function.

It is not possible to put an upper and lower limit on the summation used to calculate b_n without involving approximation to the exact value of b_n. However, due to the factor $r^{|l|}$ the sum can be truncated depending on the index of the coefficient b_n that one wants to

calculate.

Assume that supp $f(t) = [0, T]$. We can obtain an upper and lower limit on l for each n in Equation (7.52) as

$$\begin{aligned} 0 \leq \frac{n-l}{2^m} \leq T \\ n - \lfloor 2^m T \rfloor \leq l \leq n. \end{aligned} \tag{7.53}$$

In addition, in Section 7.6 we found that we only need to calculate the coefficients b_n for $n \in [-M+2, R]$. Using this fact and Equation (7.53), we can obtain limits for l in Equation (7.52) as

$$-M + 2 - \lfloor 2^m T \rfloor \leq l \leq R. \tag{7.54}$$

The results obtained can be viewed in Figure 7.10, where a series representation for the function depicted in Figure 7.9 is shown.

7.10 Digital Signal Series Representation

Digital signals can be obtained by sampling real-time analog signals at certain sampling frequencies. The digital information is processed and after that the result may or may not be converted back into a real-time signal. When the data needs to be converted back to a real-time signal, the sampling frequency is of importance. However, nowadays there is a tendency to preserve the data in digital form in case further processing is desired. That is why in some cases it is necessary to process digital data without proper knowledge of the underlying real-time signal or the method used to convert it into a digital signal.

If the real-time signal is known we can apply any of the positive sampling transforms previously considered. In the case where we only have the discrete data we can associate a real-time signal to this data.

Suppose we have finite duration digital data $f_d[n]$ defined in the interval $n \in [0 \ldots q-1]$. We will assume without loss of generality that the real-time signal used to obtain the digital data was sampled with a sampling period $t_s = 2^{-m}$ and has supp $f(t) = [0, T = q-1]$. Since we have a finite digital signal we will assume the real-time signal is zero outside the interval given by the digital signal. We can use Equation (7.5) to obtain a projection of the associated real-time function into the space V_m. The digital data

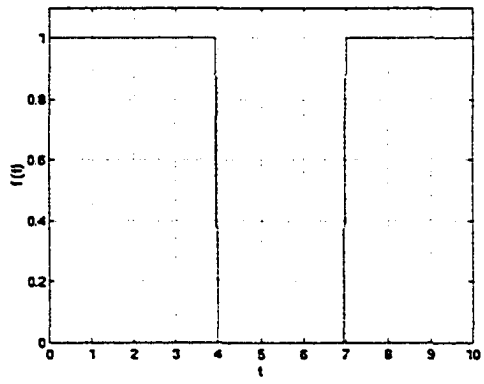

Figure 7.9: Original square signal

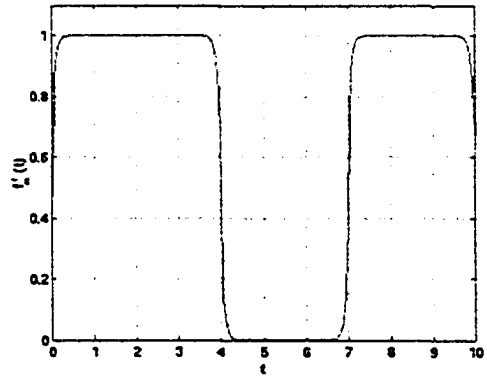

Figure 7.10: Square signal series approximation m=4, r=0.4

projection can be obtained by

$$f_d[k] = f_m^r(2^{-m}k) = \sum_{n=-N+1}^{2^m(q-1)-1} b_n \phi_{m,n}(2^{-m}k). \tag{7.55}$$

Note also the coefficients b_n are obtained by modifying Equation (7.52) into

$$\begin{aligned} b_n &= 2^{-m/2}\left(\frac{1-r}{1+r}\right) \sum_{l=n-2^m(q-1)}^{n} r^{|l|} f\left(\frac{n-l}{2^m}\right) \\ &= 2^{-m/2}\left(\frac{1-r}{1+r}\right) \sum_{l=n-2^m(q-1)}^{n} r^{|l|} f_d[n-l]\,. \end{aligned} \tag{7.56}$$

7.11 Conclusion

In this Chapter, a wavelet projection has been developed for representing compact support positive-valued real signals. The new projection has uniform convergence in compact subsets, and avoids Gibbs' phenomenon. The computation of the projection coefficients by means of a series truncation and recursion has been given. A similar projection which can be used to represent positive-valued digital signals has also been introduced. This projection also avoids the Gibbs' phenomenon.

Chapter 8

MDL Criteria for Rank Estimation Using Singular Values

In this chapter we will use positive wavelet projections to estimate the rank of a matrix using singular values. This will be done in a simple way using the Haar scaling function. Suppose we have the overdetermined system function

$$H(z) = \frac{B(z)}{A(z)} = \frac{\overline{B}(z)d(z)}{\overline{A}(z)d(z)}.$$

This system can represent for example a plant model transfer function. It is of interest to find the common factor $d(z)$ between the numerator and the denominator when the system function coefficients are perturbed by noise. A procedure to compute this factor is given in [30]. This procedure obtains the common factor from the last nonzero row of an upper triangular matrix $\mathbf{R}$ obtained from a QR factorization of a near-to-Toeplitz matrix. The most commonly used method to compute the last nonzero row of the matrix $\mathbf{R}$ is its singular value decomposition (SVD). The number of singular values bigger than zero will determine the number of rows with nonzero coefficients. However, the SVD method is computationally expensive and generally requires manual adjustment to compensate for the noise in a given problem. An alternative method to the SVD is given by Zarowski [30]. This method uses the incremental condition estimator (ICE) [31]. The ICE will estimate the smallest singular values of the leading principal submatrices of $\mathbf{R}$. Finally the degree of the polynomial $\overline{A(z)}$ is the number of singular values whose value is different than zero. Hence, the rank determination problem is translated into a test to find the number of nonzero smallest singular values of the leading principal submatrices.
The method used by Zarowski in [30] to find the number of nonzero smallest singular

values is to use the MDL criterion in an orthogonal polynomial projection of the smallest singular values. This procedure delivered good results. However, the computational complexity of the orthogonal polynomial projection is still high and was the bottleneck of the rank prediction algorithm. In this section we will use the MDL criteria in a positive wavelet projection of the smallest singular values to replace the orthogonal polynomial projection given by Zarowski. The goal is to reduce computational complexity without greatly sacrificing the statistical performance of the rank estimator.

Suppose we have a set of smallest singular values of the leading submatrices [1] of a real valued matrix $R = [r_{i,j}]$

$$\sigma_0 \geq \cdots \sigma_r \geq \cdots \geq \sigma_{n-1}.$$

where $\sigma_j > 0$ for $j = 0, 1, \cdots, q-1$ and $\sigma_j = 0$ for $j = q, \cdots, n-1$. In practice, it is difficult to access the values σ_j directly, and so usually an estimate $\hat{\sigma}_j$ is given. We will assume the estimated value is

$$\hat{\sigma}_j = \sigma_j + z_j, \tag{8.1}$$

where z_j is from a set of statistically independent random variables whose probability density function (pdf) is defined as

$$p(z_j) = \begin{cases} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z_j^2}{2\sigma^2}}, & j = 0, \cdots, q-1 \\ \alpha e^{-\alpha z_j}, & j = q, \cdots, n-2, n-1. \end{cases} \tag{8.2}$$

8.1 Haar Scaling Function Representation of the Singular Values

We will suppose the singular values can be expressed with the real distribution function

$$\begin{aligned} v(t) &= \sum_{j=0}^{q-1} \sigma_j \left(u(t-j) - u(t-(j+1))\right) \\ &= \sum_{j=0}^{q-1} \sigma_j \phi(t-j), \end{aligned} \tag{8.3}$$

where $\phi(t)$ is the Haar scaling function (recall the definition in Equation (4.29)).

Note with the definition given by Equation (8.3) $\operatorname{supp} v(t) = [0, q]$. Hence, using wavelet

[1] A leading submatrix of a matrix R is an square submatrix that starts at the first row and first column.

packet theory we can project $v(t)$ into the space V_m as

$$Pv(t) = \sum_p b_p \phi_{m,p}(t), \tag{8.4}$$

where b_p is given by

$$\begin{aligned} b_p &= \int v(t)\phi_{m,p}(t)dt \\ &= \int \sum_{j=0}^{q-1} \sigma_j \phi(t-j)\phi_{m,p}(t)dt \\ &= 2^{\frac{m}{2}} \int \sum_{j=0}^{q-1} \sigma_j \phi(t-j)\phi(2^m t - p)dt \\ &= 2^{\frac{m}{2}} \sum_{j=0}^{q-1} \sigma_j \int \phi(t-j)\phi(2^m t - p)dt \\ &= \sum_{j=0}^{q-1} \sigma_j < \phi_{0,j}, \phi_{m,p} > . \end{aligned} \tag{8.5}$$

If we have $m \geq 0$, Equation (8.5) can be reduced to

$$\begin{aligned} b_p &= 2^{\frac{m}{2}} \int_{\infty}^{\infty} \sum_{j=0}^{q-1} \sigma_j \left(u(t-j) - u(t-(j+1))\right) \phi(2^m t - p)dt \\ &= 2^{\frac{m}{2}} \int_{\frac{p}{2^m}}^{\frac{p+1}{2^m}} \sigma_{\lfloor \frac{p}{2^m} \rfloor} dt \\ &= 2^{\frac{m}{2}} \frac{\sigma_{\lfloor \frac{p}{2^m} \rfloor}}{2^m} \\ &= 2^{-\frac{m}{2}} \sigma_{\lfloor \frac{p}{2^m} \rfloor} \end{aligned} \tag{8.6}$$

for $0 \leq p \leq 2^m q$, and $b_p = 0$ otherwise.
On the other hand, if we have $m < 0$, Equation (8.5) is reduced to

$$\begin{aligned} b_p &= 2^{\frac{m}{2}} \int_{\infty}^{\infty} \sum_{j=0}^{q-1} \sigma_j \left(u(t-j) - u(t-(j+1))\right) \phi(2^m t - p)dt \\ &= 2^{-\frac{m}{2}} \sum_{j=\frac{p}{2^m}}^{\frac{p+1}{2^m}-1} \sigma_j \end{aligned} \tag{8.7}$$

for $0 \leq p \leq 2^m q$, and $b_p = 0$ otherwise. If $m < 0$, the computation of the coefficients b_p becomes more complex. Since one of the goals of the rank estimation algorithm is speed, from now on we will assume that $m \geq 0$.

8.2 MDL Criterion for the Singular Values Using the Haar Scaling Function

We can obtain a representation of the individual singular values from Equation (8.4) as

$$\sigma_j = v(j) = \sum_{p=0}^{2^m q} b_p \phi_{m,p}(j).$$

In accordance with this model, $\hat{\sigma}_j$ is a random variable with joint pdf given by

$$\begin{aligned} p(\hat{\sigma}_j | b_p, \sigma^2, \alpha, m, q) &= \prod_{j=0}^{q-1} \left[\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left(\frac{\left[\hat{\sigma}_j - \sum_{p=0}^{2^m q} b_p \phi_{m,p}(j) \right]^2}{2\sigma^2} \right)} \right] \prod_{j=q}^{n-1} \left[\alpha e^{-\alpha \hat{\sigma}_j} \right] \\ &= \frac{\alpha^{n-q}}{(2\pi\sigma^2)^{\frac{q}{2}}} e^{\left(-\frac{\sum_{j=0}^{q-1} \left[\hat{\sigma}_j - \sum_{p=0}^{2^m q} b_p \phi_{m,p}(j) \right]^2}{2\sigma^2} - \alpha \sum_{j=q}^{n-1} \hat{\sigma}_j \right)}. \end{aligned} \tag{8.8}$$

Note that this distribution is conditioned on the unknown parameters b_p, σ^2, α, m and q. The codelength (recall Equation (5.3)) for the residual between the real and the predicted data can be obtained from Equation (8.8) as

$$\begin{aligned} L(\hat{\sigma}_j | b_p, \sigma^2, \alpha, m, q) &= -\log_2 \left[p(\hat{\sigma}_j | b_p, \sigma^2, \alpha, m, q) \right] \\ &= -(n-q) \log_2 \alpha + \frac{q}{2} \log_2 [2\pi\sigma^2] \\ &+ \log_2 e \underbrace{\left(\frac{1}{2\sigma^2} \sum_{j=0}^{q-1} \left[\hat{\sigma}_j - \sum_{p=0}^{2^m q} b_p \phi_{m,p}(j) \right]^2 + \alpha \sum_{j=q}^{n-1} \hat{\sigma}_j \right)}_{= V(b^{(q)})} \end{aligned} \tag{8.9}$$

We will define

$$
\begin{aligned}
V(b^{(q)}) &= \frac{1}{2\sigma^2}\left[\sum_{j=0}^{q-1}\left[\hat{\sigma}_j - \sum_{p=0}^{2^m q} b_p \phi_{m,p}(j)\right]^2\right] + \alpha\sum_{j=q}^{n-1}\hat{\sigma}_j \\
&= \frac{1}{2\sigma^2}\sum_{j=0}^{q-1}\left[\hat{\sigma}_j^2 - 2\hat{\sigma}_j\sum_{p=0}^{2^m q} b_p\phi_{m,p}(j) + \sum_{d=0}^{2^m q}\sum_{p=0}^{2^m q} b_d b_p \underbrace{\phi_{m,d}(j)\phi_{m,p}(j)}_{2^m\delta_{d-p}}\right] \\
&\quad + \alpha\sum_{j=q}^{n-1}\hat{\sigma}_j \\
&= \frac{1}{2\sigma^2}\left[\sum_{j=0}^{q-1}\hat{\sigma}_j^2 - 2\sum_{p=0}^{2^m q} b_p \sum_{j=0}^{q-1}\hat{\sigma}_j\phi_{m,p}(j) + 2^m q\sum_{p=0}^{2^m q} b_p^2\right] + \alpha\sum_{j=q}^{n-1}\hat{\sigma}_j \\
&= \frac{1}{2\sigma^2}\left[\rho_q - 2[s^{(q)}]^T b^{(q)} + b^{(q)T}\mathbf{H}^{(q)}b^{(q)}\right] + \alpha\sum_{j=q}^{n-1}\hat{\sigma}_j,
\end{aligned}
\tag{8.10}
$$

where $\rho_q = \sum_{j=0}^{q-1}\hat{\sigma}_j^2$, $b^{(q)} = [b_0 \cdots b_{2^m q}]^T$, $s_l^{(q)} = \sum_{j=0}^{q-1}\hat{\sigma}_j\phi_{m,l}(j)$, $s^{(q)} = [s_0^{(q)}, s_1^{(q)}, \cdots, s_{2^m q}^{(q)}]$ and $\mathbf{H}^{(q)} = 2^m q\mathbf{I}^{(q)}$.

Note also that for $m > 0$

$$
\begin{aligned}
s_l^{(q)} &= \sum_{j=0}^{q-1}\hat{\sigma}_j\phi_{m,l}(j) \\
&= \begin{cases} 2^{\frac{m}{2}}\hat{\sigma}_{\frac{l}{2^m}} & \text{if } \frac{l}{2^m} \in \mathbb{Z} \\ 0 & \text{i.o.c.} \end{cases}
\end{aligned}
\tag{8.11}
$$

8.3 Maximum Likelihood Estimates

The MLE (Maximum Likelihood Estimate) for $b^{(q)}$ is obtained minimizing the codelength given in Equation (8.9). The minimum value can be obtained taking the gradient $\nabla_{b^{(q)}}$ of $b^{(q)}$ and finding where it is zero-valued. The gradient of Equation (8.9) is

$$
\nabla_{b^{(q)}}\left[L(\hat{\sigma}_j|b_p,\sigma^2,\alpha,m,q)\right] = \frac{\log_2 e}{2\sigma^2}\left[2b^{(q)T}\mathbf{H}^{(q)} - 2[s^{(q)}]^T\right], \tag{8.12}
$$

and this equation is zero-valued for $b^{(q)} = \hat{b}^{(q)}$,where

$$
\hat{b}^{(q)} = \mathbf{H}^{(q)-T}s^{(q)}. \tag{8.13}
$$

Furthermore, since in our case $\mathbf{H}^{(q)} = \mathbf{H}^{(q)T}$, we have

$$\mathbf{H}^{(q)-T} = \mathbf{H}^{(q)-1}.$$

When $\hat{b}_p$ replaces b_p for all p, one obtains

$$V(b^{(q)}) = \rho_q - [s^{(q)}]^T \mathbf{H}^{(q)-1} s^{(q)},$$

and hence Equation (8.9) becomes

$$\begin{aligned} & L(\hat{\sigma}_j | \hat{b}_p, \sigma^2, \alpha, m, q) \\ & = -(n-q)\log_2 \alpha + \frac{q}{2}\log_2[2\pi\sigma^2] \\ & + \frac{\log_2 e}{2\sigma^2}\left[\rho_q - [s^{(q)}]^T \mathbf{H}^{(q)-1} s^{(q)}\right] + \alpha \log_2 e \left[\sum_{j=q}^{n-1} \hat{\sigma}_j\right]. \end{aligned} \tag{8.14}$$

Note also that

$$\begin{aligned} \frac{\partial L(\hat{\sigma}_j | \hat{b}_p, \sigma^2, \alpha, m, q)}{\partial \sigma^2} &= \frac{q}{2}\log_2 e \left[\frac{1}{\sigma^2}\right] - \frac{\log_2 e}{2[\sigma^2]^2}\left[\rho_q - [s^{(q)}]^T \mathbf{H}^{(q)-1} s^{(q)}\right] \\ &= \frac{\log_2 e}{2\sigma^2}\left[q - \frac{1}{\sigma^2}\left[\rho_q - [s^{(q)}]^T \mathbf{H}^{(q)-1} s^{(q)}\right]\right], \end{aligned}$$

which implies that the MLE $\hat{\sigma}^2$ for σ^2 is

$$\hat{\sigma}^2 = \frac{1}{q}\left[\rho_q - [s^{(q)}]^T \mathbf{H}^{(q)-1} s^{(q)}\right],$$

and that

$$\begin{aligned} \frac{\partial L(\hat{\sigma}_j | \hat{b}_p, \sigma^2, \alpha, m, q)}{\partial \alpha} &= -(n-q)\log_2 e \frac{1}{\alpha} + \log_2 e \left[\sum_{j=q}^{n-1} \hat{\sigma}_j\right] \\ &= \log_2 e \left[\frac{-(n-q)}{\alpha} + \sum_{j=q}^{n-1} \hat{\sigma}_j\right], \end{aligned}$$

which implies that the MLE $\hat{\alpha}$ for α is

$$\hat{\alpha} = \frac{n-q}{\sum_{j=q}^{n-1} \hat{\sigma}_j}.$$

[2] In this equation we use the fact $\frac{d}{dx} log_a(u) = log_a(e)\frac{1}{u}\frac{du}{dx}$.

Substituting the MLEs into Equation (8.14) we obtain

$$
\begin{aligned}
& L(\hat{\sigma}_j|\hat{b}_p, \hat{\sigma}^2, \hat{\alpha}, m, q) \\
& = -(n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] + \frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q - [s^{(q)}]^T\mathbf{H}^{(q)-1}s^{(q)}\right)\right] \\
& + \frac{q\log_2 e}{2} + (n-q)\log_2 e \\
& = -(n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] + \frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q - [s^{(q)}]^T\mathbf{H}^{(q)-1}s^{(q)}\right)\right] \\
& + \left(n-\frac{q}{2}\right)\log_2 e \\
& = \left(n-\frac{q}{2}\right)\log_2 e - (n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] \\
& + \frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q - [s^{(q)}]^T\mathbf{H}^{(q)-1}s^{(q)}\right)\right] \\
& = \left(n-\frac{q}{2}\right)\log_2 e - (n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] \\
& + \frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q - [s^{(q)}]^T[2^m q\mathbf{I}^{(q)}]^{-1}s^{(q)}\right)\right] \\
& = \left(n-\frac{q}{2}\right)\log_2 e - (n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] \\
& + \frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q - \frac{2^{-m}}{q}[s^{(q)}]^T s^{(q)}\right)\right].
\end{aligned}
\tag{8.15}
$$

Using Equation (8.11) we note

$$
\begin{aligned}
& [s^{(q)}]^T s^{(q)} \\
& = 2^{\frac{m}{2}}[\hat{\sigma}_0, 0, \ldots, \hat{\sigma}_1, 0, \ldots, \hat{\sigma}_q, 0, \ldots, \hat{\sigma}_{q-1}]^T 2^{\frac{m}{2}}[\hat{\sigma}_0, 0, \ldots, \hat{\sigma}_1, 0, \ldots, \hat{\sigma}_q, 0, \ldots, \hat{\sigma}_{q-1}]^T \\
& = 2^m \sum_{j=0}^{q-1}\hat{\sigma}_j^2 \\
& = 2^m \rho_q.
\end{aligned}
\tag{8.16}
$$

Then, Equation (8.15) can be transformed into

$$\begin{aligned} L(\hat{\sigma}_j|\hat{b}_p,\hat{\sigma}^2,\hat{\alpha},m,q) &= \left(n-\frac{q}{2}\right)\log_2 e-(n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] \\ &\quad+\frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q-\frac{2^{-m}}{q}2^m\rho_q\right)\right] \\ &= \left(n-\frac{q}{2}\right)\log_2 e-(n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] \\ &\quad+\frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q\left(1-\frac{1}{q}\right)\right)\right]. \end{aligned} \tag{8.17}$$

The MDL criterion applied here, as it is defined by Saito in [22], will give us

$$\begin{aligned} MDL(q*,m*) &= \min_{\substack{2\le q\le n-1\\1\le m\le n-1}}\left[\frac{2^m q}{2}\log_2(n)+L(\hat{\sigma}_j|\hat{b}_p,\hat{\sigma}^2,\hat{\alpha},m,q)\right] \\ &= \min_{\substack{2\le q\le n-1\\1\le m\le n-1}}\left[2^{m-1}q\log_2(n)+\left(n-\frac{q}{2}\right)\log_2 e-(n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right]\right. \\ &\quad\left.+\frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q\left(1-\frac{1}{q}\right)\right)\right]\right]. \end{aligned} \tag{8.18}$$

Note the term $L(\hat{\sigma}_j|\hat{b}_p,\hat{\sigma}^2,\hat{\alpha},m,q)$ is independent of m, and the only term that depends on m in the computation of the MDL criterion is $2^{m-1}q\log_2(n)$. In this term, $\log_2(n)$ is constant. Due to this reason $2^{m-1}q\log_2(n)$ is a monotonically increasing function with respect to m and q, and the minimum is obtained for $m=1$.
Using this fact Equation (8.18) becomes

$$\begin{aligned} MDL(q*) &= \min_{2\le q\le n-1}\left[q\log_2(n)+\left(n-\frac{q}{2}\right)\log_2 e-(n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right]\right. \\ &\quad\left.+\frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q\left(1-\frac{1}{q}\right)\right)\right]\right]. \end{aligned} \tag{8.19}$$

The term $q\log_2(n)$ is introduced in the MDL in order to pick the model that possesses the smallest number of terms. In our case we are more interested in picking the best

fitting model. For this reason we can choose the model using only

$$MDL(q*) = \min_{2\leq q\leq n-1}\left[\left(n-\frac{q}{2}\right)\log_2 e-(n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right]\right.$$
$$\left.+\frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q\left(1-\frac{1}{q}\right)\right)\right]\right]. \tag{8.20}$$

The simulation results obtained using Equations (8.19) and (8.20) are shown on Section 8.5.

Note the final selection of the model order estimator has to be based on the results obtained by simulation instead of using only analytical reasons [32]. This is due to the lack of a consistent analytical method which gives us a criterion to choose one model order estimator from another.

8.4 Computational Complexity

In the computation of the MDL criterion the two terms with the highest computational complexity are $\sum_{j=q}^{n-1}\hat{\sigma}_j$ and ρ_q. For $\sum_{j=q}^{n-1}\hat{\sigma}_j$, we need $n-q-1$ additions. Note also that we need q multiplications and $q-1$ additions to compute ρ_q. Therefore, Equation (8.19) requires $n+3$ additions, $q+6$ multiplications, 4 divisions and 2 logarithms for every q. If we suppose that the computation of logarithms, additions, multiplications and division take one clock cycle each we can see the number of cycles needed to determine the MDL criterion is

$$\begin{aligned} c &= (n-2)\left[(n+3)+4+2\right]+\sum_{q=2}^{n-1}(q+6) \\ &= (n-2)\left[(n+3)+6+4+2\right]+\sum_{q=1}^{n-1}q-1 \\ &= (n-2)\left[(n+3)+6+4+2\right]+\frac{n(n-1)}{2}-1 \\ &= (n-2)(n+15)+\frac{n(n-1)}{2}-1. \end{aligned} \tag{8.21}$$

From Equation (8.21), we can infer that this implementation requires only $O(n^2)$ operations which is an improvement from the $O(n^3)$ operations used in the algorithm presented in [33].

8.5 Simulation Results

Suppose we have the system function

$$H(z) = \frac{B(z)}{A(z)} = \frac{\overline{B}(z)d(z)}{\overline{A}(z)d(z)}.$$

Now we add uniform pdf noise with a maximum noise value e, where the pdf is given by

$$p(t) = \begin{cases} \frac{1}{2e} & \text{If } t \in [-e, \ldots, e] \\ 0 & \text{otherwise} \end{cases}$$

to the polynomials $A(z)$ and $B(z)$ obtaining

$$\hat{A}(z) = A(z) + n_a(z)$$
$$\hat{B}(z) = B(z) + n_b(z),$$

where $n_a(z)$, $n_b(z)$ are the introduced additive noise terms. Then estimate the degree of the polynomial $\overline{A(z)}$ from $\hat{A}(z)$ and $\hat{B}(z)$. First, we obtain the matrix $\mathbf{R}$ from $\hat{A}(z)$ and $\hat{B}(z)$ using the theory given by Zarowski [30]. Afterwards, we compute the leading principal submatrices of matrix $\mathbf{R}$ using the ICE algorithm. The smallest singular values obtained using this procedure will be the input to the MDL rank estimation algorithms. The algorithms will estimate the degree of the polynomial $\overline{A(z)}$.
The problem will be recreated with different maximum values of noise e and it will be repeated 100 times for each different level of noise.

Example 8.5.1. Consider the following polynomials:

$$d(z) = \sum_{j=0}^{9} z^j, \quad \overline{A}(z) = z^{11} - 1, \overline{B}(z) = z^{10} - \frac{1}{2}.$$

Then the degree of the polynomial $\overline{A(z)}$ is $q = 11$. The smallest singular values obtained using ICE with $e = 0.01$ are shown in Figure 8.1. The results obtained by Zarowski using the polynomial MDL rank estimator are shown in Table 8.1. On the other hand, employing the same estimated singular values and maximum levels of noise and using Equation (8.20) for the estimator we obtained the results shown in Table 8.2. Figure 8.2

e	q						
	< 10	10	11	12	13	14	> 14
0.010	0	0	58	40	2	0	0
0.007	0	0	79	20	1	0	0
0.003	0	0	99	1	0	0	0
0.001	0	0	100	0	0	0	0

Table 8.1: Polynomial MDL rank estimator Example 8.5.1.

e	q						
	< 10	10	11	12	13	14	> 14
0.010	70	0	30	0	0	0	0
0.007	34	0	64	2	0	0	0
0.003	0	0	98	2	0	0	0
0.001	0	0	100	0	0	0	0

Table 8.2: Haar scaling function MDL rank estimator Example 8.5.1.

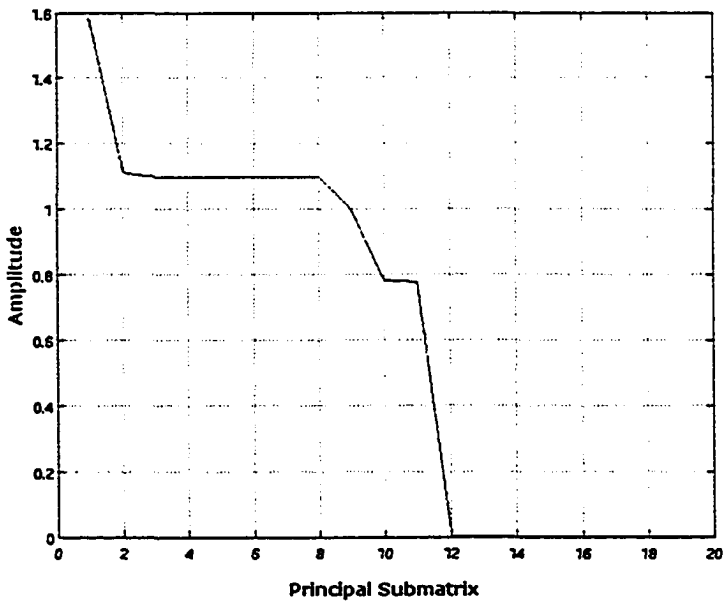

Figure 8.1: ICE-estimated smallest singular values of the leading principal submatrices of $\mathbf{R}$ (Example 8.5.1).

shows a plot (versus q) of

$$q\log_2(n) + \left(n - \frac{q}{2}\right)\log_2 e - (n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] + \frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q\left(1-\frac{1}{q}\right)\right)\right]$$

for various levels of noise, and Figure 8.3 shows a plot of

$$\left(n - \frac{q}{2}\right)\log_2 e - (n-q)\log_2\left[\frac{n-q}{\sum_{j=q}^{n-1}\hat{\sigma}_j}\right] + \frac{q}{2}\log_2\left[\frac{2\pi}{q}\left(\rho_q\left(1-\frac{1}{q}\right)\right)\right]$$

with the same levels of noise. From these results we note Equation (8.20) achieves a

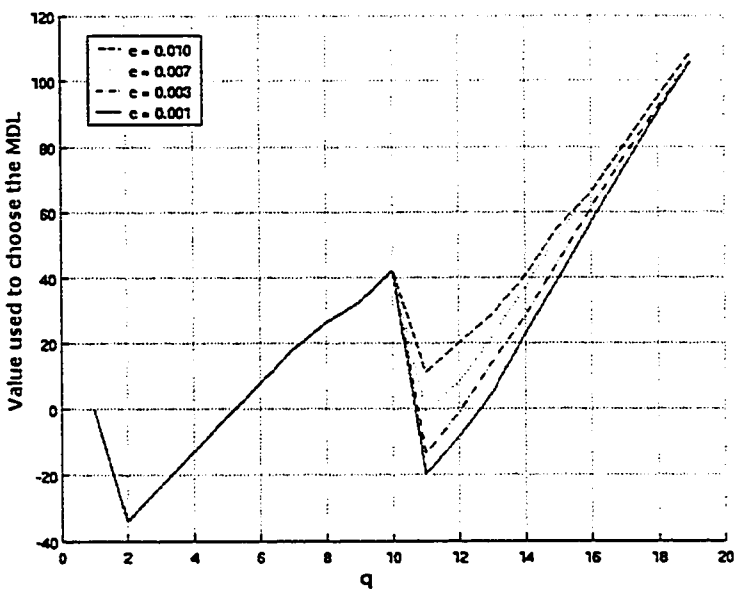

Figure 8.2: MDL versus q with the term $q\log_2(n)$ (Example 8.5.1).

better result compared to the results obtained using Equation (8.19). In addition we note the strong dominance of small values of q when we are computing the MDL criterion. This behavior is due to the dependence of the MDL criterion on the factor

$$\left(1 - \frac{1}{q}\right).$$

When q is close to one, $MDL \rightarrow -\infty$ and then the true MDL criterion value can be mistaken. It gives too much weight to small values of q.

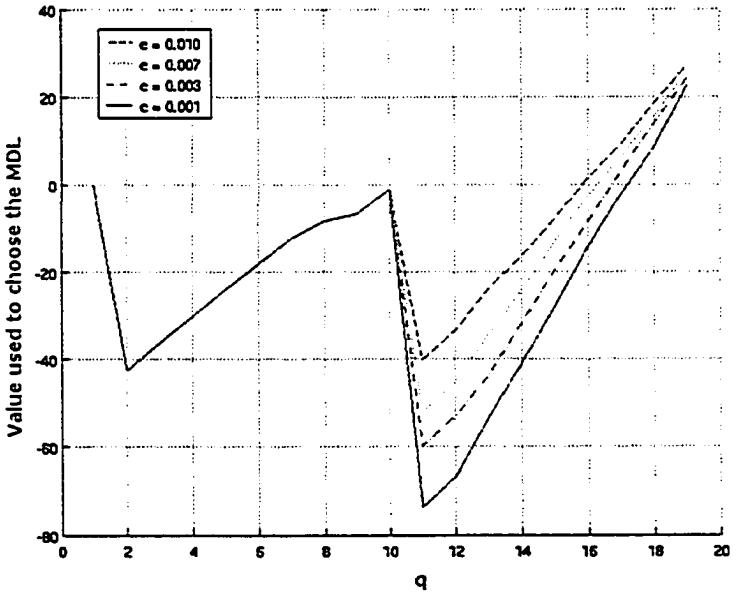

Figure 8.3: MDL versus q without the term $q\log_2(n)$ (Example 8.5.1).

e	q															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
10^{-4}	0	0	0	0	0	0	0	0	0	13	57	7	5	18	0	0
10^{-5}	0	0	0	0	0	0	0	0	0	51	47	1	1	1	0	0
10^{-6}	0	0	0	0	0	0	0	0	0	48	52	0	0	0	0	0
10^{-7}	0	0	0	0	0	0	0	0	0	46	49	3	1	1	0	0

Table 8.3: Polynomial MDL rank estimator Example 8.5.2.

Example 8.5.2. In this case we have the following polynomials:

$$\begin{aligned} d(z) &= z^5 - 0.6z^4 - 0.05z^3 - 0.05z^2 - 1.05z + 0.55 \\ \overline{A}(z) &= z^{10} - 1.6z^9 + 2.43z^8 - 1.148z^7 + 1.2248z^6 + 1.3875z^5 \\ &\quad - 0.9895z^4 + 0.9751z^3 - 0.7813z^2 - 0.623z + 0.0692 \\ \overline{B}(z) &= z^9 + 1.95z^8 + 0.0.6699z^7 + 0.1978z^6 + 0.2271z^5 \\ &\quad - 1.5652z^4 - 1.9118z^3 - 0.7413z^2 - 0.0801z + 0.0634. \end{aligned}$$

The predicted value in this case has to be $q = 10$.

The results obtained using the polynomial MDL rank estimator are shown in Table 8.3. Our results using Equation (8.20) are shown in Table 8.4. The singular values obtained

e	q															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
10^{-4}	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10^{-5}	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10^{-6}	0	98	0	0	0	0	0	0	0	2	0	0	0	0	0	0
10^{-7}	0	81	0	0	0	0	0	0	0	19	0	0	0	0	0	0

Table 8.4: Haar scaling function MDL rank estimator Example 8.5.2.

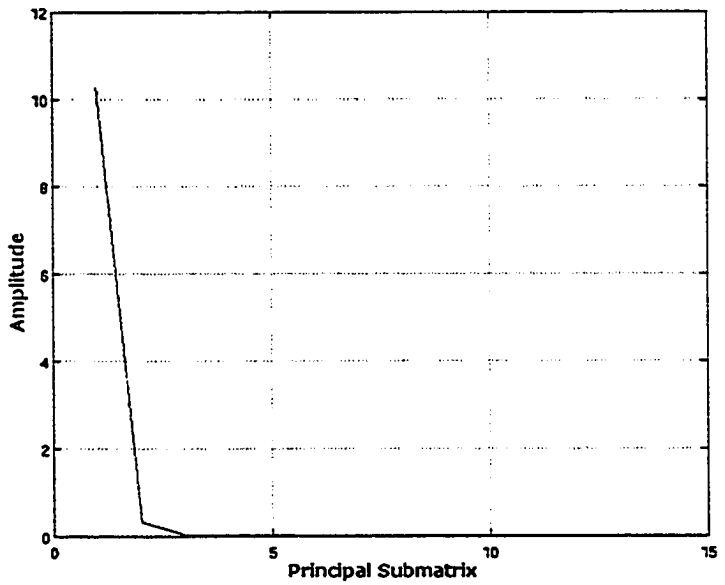

Figure 8.4: ICE-estimated smallest singular values of the leading principal submatrices of $\mathbf{R}$ (Example 8.5.2).

from ICE starting at the second submatrix are very close to zero. This can be seen in Figure 8.4. In this case our predictor take these values as zero. Hence the likelihood of obtaining a wrong predicted value is very high even for small values of noise.
Figure 8.5 plots Equation (8.19) which is used to compute the MDL for various levels of noise, and Figure 8.6 plots Equation (8.20) which is another alternative to compute the MDL with the same levels of noise employed when we used Equation (8.19). In this

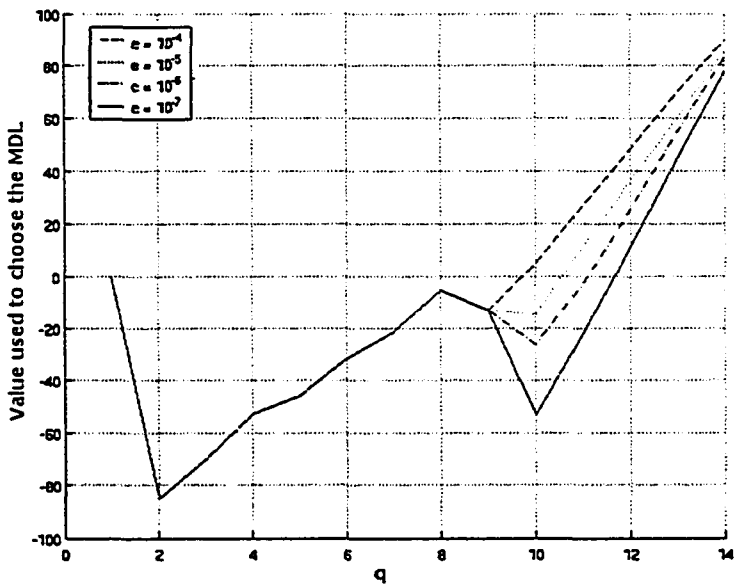

Figure 8.5: MDL versus q with the term $q\log_2(n)$ (Example 8.5.2).

case the algorithm's accuracy decreases substantially.

Example 8.5.3. Consider the following polynomials:

$$d(z) = (z+1)^3 = z^3 + 3z^2 + 3z + 1$$
$$\overline{A}(z) = z^3 + 2z^2 + 5z + 1$$
$$\overline{B}(z) = z + 1.01.$$

The predicted value in this case has to be $q = 3$.
The results obtained using the polynomial MDL rank estimator are shown in Table 8.5. Our results using Equation (8.20) are shown in Table 8.6. Figure 8.7 plots Equation (8.19) which is used to compute the MDL for various levels of noise, and Figure 8.8 plots Equation (8.20) which is another alternative to compute the MDL with the same

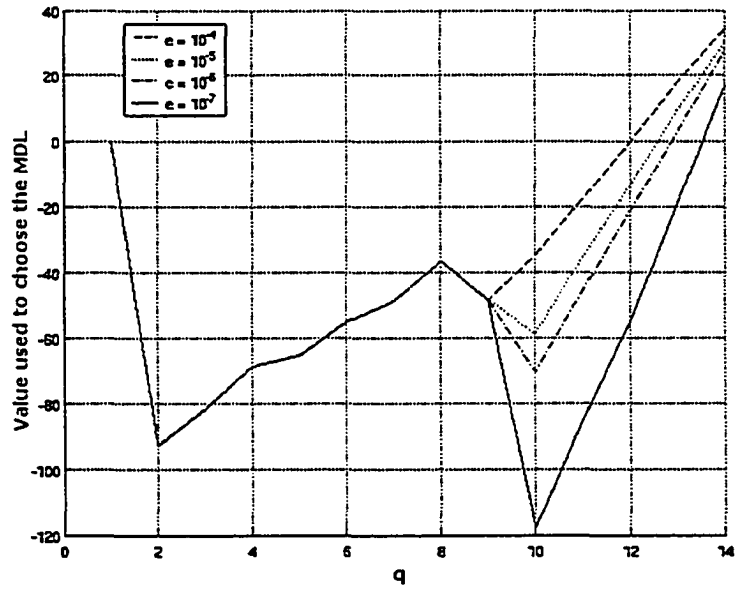

Figure 8.6: MDL versus q without the term $q \log_2(n)$ (Example 8.5.2).

e	q						
	1	2	3	4	5	6	7
10^{-3}	0	0	84	3	13	0	0
5×10^{-3}	0	0	89	4	7	0	0
10^{-2}	0	0	87	8	5	0	0
5×10^{-2}	3	0	71	17	9	0	0
10^{-1}	22	0	33	30	15	0	0

Table 8.5: Polynomial MDL rank estimator Example 8.5.3.

e	q						
	1	2	3	4	5	6	7
10^{-3}	0	2	98	0	0	0	0
5×10^{-3}	0	79	21	0	0	0	0
10^{-2}	0	95	5	0	0	0	0
5×10^{-2}	0	100	0	0	0	0	0
10^{-1}	0	97	3	0	0	0	0

Table 8.6: Haar scaling function MDL rank estimator Example 8.5.3.

levels of noise employed when we used Equation (8.19).

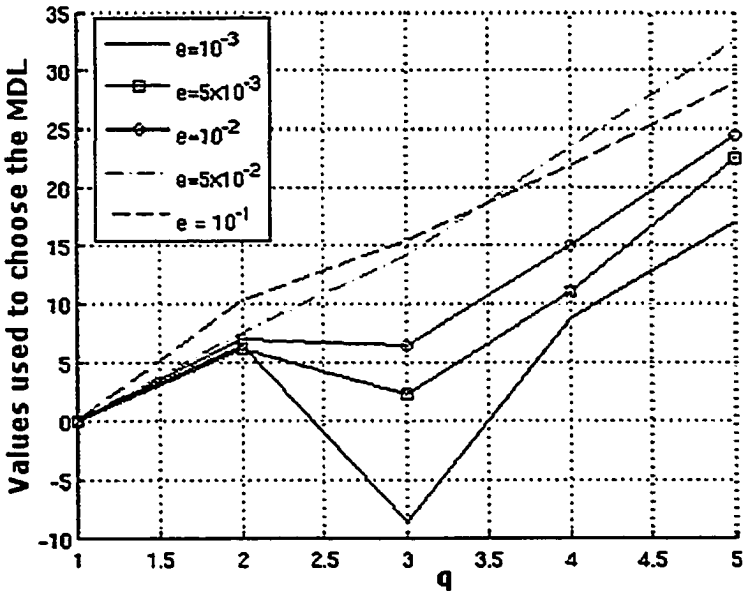

Figure 8.7: MDL versus q with the term $q\log_2(n)$ (Example 8.5.3).

Example 8.5.4. Consider the following polynomials:

$$\overline{A}(z)d(z) = z^5 + 5.503z^4 + 9.765z^3 + 7.647z^2 + 2.762z + 0.37725$$
$$\overline{B}(z)d(z) = z^4 + 2.993z^3 - 0.7745z^2 + 2.0070 + 0.7605.$$

In this case the common factor is:

$$d(z) = z^2 - 1.007z + 0.2534.$$

Hence, the predicted value in this case has to be $q = 3$.
Our results using Equation (8.20) are shown in Table 8.7. Figure 8.9 plots Equation (8.19) which is used to compute the MDL for various levels of noise, and Figure 8.10 plots Equation (8.20) which is another alternative to compute the MDL with the same levels of noise employed when we used Equation (8.19).

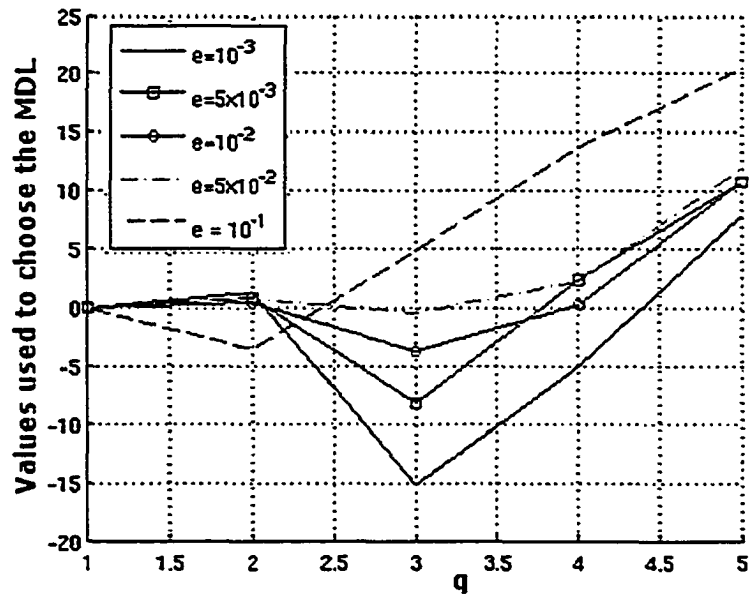

Figure 8.8: MDL versus q without the term $q \log_2(n)$ (Example 8.5.3).

e	q						
	1	2	3	4	5	6	7
10^{-3}	0	48	52	0	0	0	0
5×10^{-3}	0	93	7	0	0	0	0
10^{-2}	0	91	9	0	0	0	0
5×10^{-2}	0	100	0	0	0	0	0
10^{-1}	0	98	2	0	0	0	0

Table 8.7: Haar scaling function MDL rank estimator Example 8.5.4.

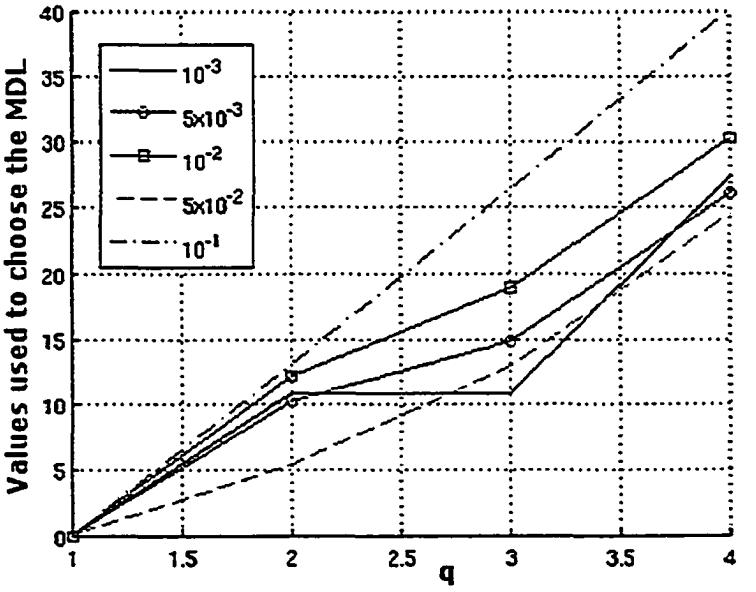

Figure 8.9: MDL versus q with the term $q \log_2(n)$ (Example 8.5.4).

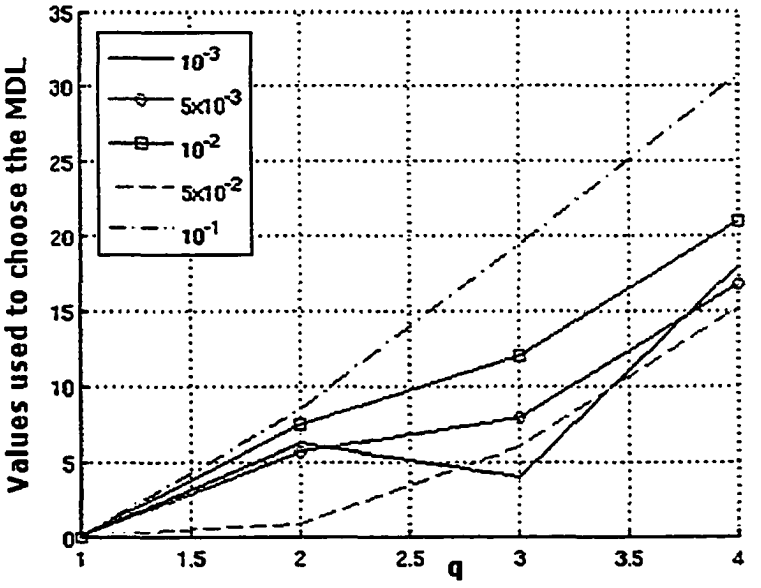

Figure 8.10: MDL versus q without the term $q \log_2(n)$ (Example 8.5.4).

Example 8.5.5. Finally, consider the polynomials:

$$\overline{A}(z)d(z) = z^4 + z + 1$$
$$\overline{B}(z)d(z) = z^3 + 0.01z.$$

In this case the common factor is:

$$d(z) = 1.$$

Hence, the predicted value in this case has to be $q = 4$.
In this case our prediction algorithm predicts $q = 2$ instead of the actual value. This behavior may be produced by the rapid decrease of the ICE value as can be seen in Figure 8.11. In contrast the polynomial MDL rank estimator predicts the correct value. Therefore, in this case it is better to sacrifice the speed increment achieved by our algorithm and use the polynomial MDL estimator instead.

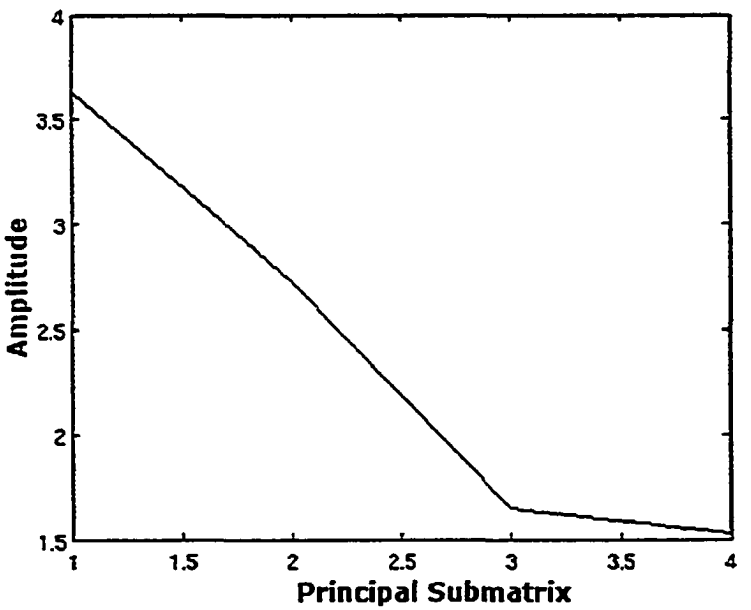

Figure 8.11: Estimated ICE value with $e = 1$ (Example 8.5.5).

8.6 Conclusion

In this Chapter, a rank estimator for a matrix using its principal submatrices smallest singular values has been presented. The singular values are real positive-valued signals.

Due to this reason, a projection for this type of signals has been used. The basis function used to project the signals is the Haar wavelet. The results obtained have been compared with the ones obtained from a rank estimator using an orthogonal polynomial projection [33]. It has been shown that the rank estimator developed in this thesis has a better computational efficiency. However, it has also been shown that the increase of speed diminishes the accuracy when the noise level is high.

Chapter 9

Conclusion and Further Work

9.1 Conclusion

In the first four Chapters, an introduction has been given to the basic theory in orthonormal series projections, wavelet theory and the minimum description length criterion. The information given in these Chapters has been used to develop the theory introduced in the main part of this thesis.

In Chapter 6, the analysis of the output convolution matrix from a chain of decimators with white Gaussian noise as an input has been presented. It has been shown that this matrix possesses a diagonal like structure consisting of three submatrices. A recursive method has been developed to compute the size of these submatrices. Since the matrix inverse of the correlation matrix plays an important role in signal detection and signal estimation problems, the existence of this inverse has also been proven. However, as it has also been shown the inversion problem for this matrix is ill-conditioned.

In Chapter 7, a series projection with uniform convergence in compact subsets has been developed. This projection can be used when the signal to be processed is a positive-valued real signal with compact support. Due to its uniform convergence, this projection avoids the Gibbs' phenomenon. Two methods to compute the coefficients of the projection have been developed. The first one consists of a series truncation. In this case, a bound for the truncation error magnitude has been obtained. The second method obtained to compute the projection coefficients is a recursive method. In addition, a series projection to process positive-valued digital signals has been given. An equation to obtain the projection coefficients for this projection is also given. This projection also avoids the Gibbs' phenomenon.

Finally, in Chapter 8, a rank estimator using a projection for positive-valued signals has

been developed, where the basis used to project the signals is the Haar wavelet. The results obtained have been compared with the ones obtained from a rank estimator using orthogonal polynomials [33]. It has been found that the rank estimator developed in this thesis has a better computational efficiency. However, it has also been found that it is less accurate than the orthogonal polynomial rank estimator when the noise level is high.

9.2 Further Work

The existence of the output correlation matrix inverse for a chain of decimators with white Gaussian noise as an input has been shown in this thesis. However, an algorithm to compute this inverse still needs to be developed. In addition, the analysis of the output correlation matrix for a chain of interpolators when the input signal is white Gaussian noise is also needed. These two problems can be the subject of new research work which may lead to better methods for signal estimation and signal description using wavelet transforms.

The projection of one dimensional positive-valued signals has been analyzed in this thesis. There are multiple applications where this theory can be used (i.e. fiber optics light intensity functions, histograms). However, it may also be of interest to extend the theory so that it can be applied to multiple dimension positive-valued signals (i.e. digital images).

Bibliography

[1] G. G. Walter and X. Shen, *Wavelets and Other Orthogonal Systems*, 2nd ed., ser. Studies in Advanced Mathematics, S. G. Krantz, Ed. Chapman & Hall/CRC, 2000.

[2] I. Daubechies, *Ten Lectures on Wavelets*, ser. CBMS-NSF Series in Appl. Math. SIAM Publ., 1992.

[3] S. Akkarakaran and P. Vaidyanathan, "Bifrequency and bispectrum maps: A new look at multirate systems with stochastic inputs," *IEEE Tran. on Signal Proc.*, vol. 48, no. 3, pp. 723–736, March 2000.

[4] D. Leporini and J. Pesquet, "High-order wavelet packets and cumulant field analysis," *IEEE Trans. on Inf. Theory*, vol. 45, no. 3, pp. 863–876, April 1999.

[5] V. P. Sathe and P. Vaidyanathan, "Effects of multirate systems on the statistical properties of random signals," *IEEE Trans. on Signal Proc.*, vol. 41, no. 1, pp. 131–145, January 1993.

[6] R. Coifman and D. Donoho, "De-noising by soft-thresholding," Standford University, Tech. Rep., 1992.

[7] ——, "Translation invariant de-noising," Yale University and Standford University, Tech. Rep., 1995.

[8] E. Kreyzig, *Introductory Functional Analysis with Applications.* John Wiley and Sons, 1978.

[9] A. Pinkus and S. Zafrany, *Fourier Series and Integral Transforms.* Cambridge University Press, 1997.

[10] A. A. Michelson, "Letter to the editor," *Nature*, no. 58, pp. 544–545, 1898.

[11] J. Gibbs, "Letter to the editor," *Nature*, no. 59, p. 606, 1899.

[12] S. G. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $L^2(\mathbb{R})$," *Trans. Amer. Math. Soc.*, vol. 315, pp. 69–87, Sept. 1989.

[13] C. J. Zarowski, "Notes on orthogonal wavelets and wavelet packets," Queen's University, Department of Electrical and Computer Engineering, Internal Report 1-95, September 1996.

[14] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, M. Wellesley, Ed. Wellesley-Cambridge Press, 1996.

[15] C. J. Zarowski, "Gaussian random vectors as input to decimators," 2002, set of notes on the decimation and interpolation problem.

[16] S. E. Kelly, "Gibbs phenomenon for wavelets," *Applied and Computational Harmonic Analysis*, vol. vol. 3, no. Issue 1, pp. pp. 72–81, January 1996.

[17] D. Hirshberg and N. Merhav, "Robust methods for model order estimation," *IEEE Transactions on Signal Processing*, vol. 44, no. 3, pp. 620–628, March 1996.

[18] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *Ann. Statist.*, vol. 11, no. 2, pp. 416–431, 1983.

[19] ——, "Universal coding, information, prediction, and estimation," *IEEE Trans. Inform. Theory*, vol. 30, no. 4, pp. 629–636, 1984.

[20] P. Grünwald, "A tutorial introduction to the minimum description length principle," 2004, to appear in "Advances in Minimum Description Length: Theory and Applications".

[21] S. M. Kay, *Statistical Signal Processing: Estimation Theory*, ser. Signal Processing, A. Oppenheim, Ed. Prentice Hall, 1993, vol. I.

[22] N. Saito, "Simultaneous noise supression and signal compresion using a library of orthonormal bases and the minimum description length," *in Wavelets in Geophysics*, pp. 299–324, 1994.

[23] S. M. Kay, *Statistical Signal Processing Detection Theory*, ser. Signal Processing, A. Oppenheim, Ed. Prentice Hall, 1993.

[24] R. A. Horn and C. R. Johnson, *Matrix Analysis.* Cambridge University Press, 1990.

[25] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 3rd ed., W. Stephen, Ed. McGraw-Hill, 1991.

[26] H.-T. Shim and H. Volkmer, "On the gibbs phenomenon for wavelet expansions," *Journal of Approximation Theory*, vol. 84, no. Issue 1, pp. 74–95, January 1996.

[27] G. G. Walter and X. Shen, "Positive estimation with wavelets," *Contemporary Mathematics*, vol. 216, pp. 63–79, 1998.

[28] C. J. Zarowski and J. Pulido, "Positive estimation with wavelets: A computational method," in *Int. Signal Proc. Conf. (ISPC), and Global DSP Expo, Dallas, Texas*, 31 March to 3 April 2003.

[29] G. G. Walter and X. Shen, "Positive sampling in wavelet subspaces," *Applied and Computational Harmonic Analysis*, vol. 12, pp. 150–165, 2002.

[30] C. J. Zarowski, X. Ma, and F. W. Fairman, "QR-factorization method for computing the greatest common divisor of polynomials with inexact coefficients," *IEEE Trans. in signal Proc.*, vol. vol. 48, pp. 3042–3051, Nov. 2000.

[31] C. H. Bischof, "Incremental condition estimation," *SIAM J.Matrix Anal.*, vol. 11, pp. 312–322, April 1990.

[32] P. Stoica and Y. Selén, "Model-order selection," *IEEE Signal Processing Magazine*, vol. 21, no. 4, pp. 36–47, July 2004.

[33] C. J. Zarowski, "The MDL criterion for rank determination via effective singular values," *IEEE Transactions on Signal Processing*, vol. Vol. 46, no. No. 6, pp. 1741–1744, June 1998.

Appendix A

Matlab Correlation Matrices

A.1 M=8 N=8 Results

$$\begin{pmatrix}
1.1\cdot10^{-3} & -6.4\cdot10^{-3} & 2.1\cdot10^{-2} & -2.4\cdot10^{-10} & 0 & 0 & 0 \\
-6.4\cdot10^{-3} & 3.7\cdot10^{-2} & -1.2\cdot10^{-1} & 6.7\cdot10^{-10} & -2.4\cdot10^{-10} & 0 & 0 \\
2.1\cdot10^{-2} & -1.2\cdot10^{-1} & 4.3\cdot10^{-1} & -1.9\cdot10^{-9} & 6.7\cdot10^{-10} & -2.4\cdot10^{-10} & 0 \\
-2.4\cdot10^{-10} & 6.7\cdot10^{-10} & -1.9\cdot10^{-9} & 1.0\cdot10^{0} & -1.9\cdot10^{-9} & 6.7\cdot10^{-10} & -2.4\cdot10^{-10} \\
0 & -2.4\cdot10^{-10} & 6.7\cdot10^{-10} & -1.9\cdot10^{-9} & 9.9\cdot10^{-1} & 6.4\cdot10^{-3} & -2.1\cdot10^{-2} \\
0 & 0 & -2.4\cdot10^{-10} & 6.7\cdot10^{-10} & 6.4\cdot10^{-3} & 9.6\cdot10^{-1} & 1.2\cdot10^{-1} \\
0 & 0 & 0 & -2.4\cdot10^{-10} & -2.1\cdot10^{-2} & 1.2\cdot10^{-1} & 5.6\cdot10^{-1}
\end{pmatrix}$$

A.2 M=4 N=8 Results

$$\begin{pmatrix}
6.6\cdot10^{-2} & 5.0\cdot10^{-17} & 0. & 0. & 0. \\
5.0\cdot10^{-17} & 1.0\cdot10^{0} & 5.0\cdot10^{-17} & 0. & 0. \\
0. & 5.0\cdot10^{-17} & 1.0\cdot10^{0} & 5.0\cdot10^{-17} & 0. \\
0. & 0. & 5.0\cdot10^{-17} & 1.0\cdot10^{0} & 5.0\cdot10^{-17} \\
0. & 0. & 0. & 5.0\cdot10^{-17} & 9.3\cdot10^{-1}
\end{pmatrix}$$

Appendix B

Matlab Code

B.1 Code Function makephi

Listing B.1: makephi

```
%
%                          makephi.m
%
% Given the two-scale sequence which specifies a scaling
% function construct the matrix whose eigenvectors give the
% scaling function at the integer knots (this is matrix M).
%
% This program then finds the scaling function at the
% integer knots and normalizes.
%
% The interpolatory graphical display algorithm (IGDA) is
% used to compute the scaling function in between the
% integer knots.
%
% This function requires the inputs:
%
%  p = two-scale sequence corresponding to the lowpass QMF
%  filter J = we want to interpolate the scaling function at
%  the dyadic points 2^(-J)*Z
%
% The scaling function phi may be plotted by the user at the
```

```
% points in vector tphi.
%
% NOTE: Execute M-file psequence.m before running this
% M-file.  This will create vector p.
%
%       This function requires upsample.m.
%

function [tphi,phi] = makephi(p,J)

N = length(p) - 1;

% Compute matrix M

for i = 1:N-1
  for j = 1:N-1
    k = 2*i - j + 1;
    if (k > 0) & (k < N+2)
      M(i,j) = p(2*i - j + 1);
    else
      M(i,j) = 0;
    end
  end;
end;

% Find the eigenvector of M corresponding to
% the eigenvalue of unity and normalize it to
% give the scaling function at integer knots

[V,D] = eig(M);
sum = 0;
for i = 1:N-1
  sum = sum + V(i,1);
end;
phii = (1/sum)*V(:,1);
phii = [ 0 ; phii ; 0 ];
```

```
% Use the IGDA to compute scaling function phi
% at the dyadic points

b = [ 1 ];
for j = 1:J
  bt = upsample(b,2);
  b = conv(bt,p);
end;
phi = conv(phii,b);
stepphi = 2^J;
for k = 0:stepphi*N
  tphi(k+1) = k/stepphi;
end;
```

B.2 Code Function upsample

Listing B.2: upsample

```
%
%                          upsample.m
%
% This routine upsamples an input sequence x by factor I.
%

function y = upsample(x,I)

N = length(x);
if I == 1
  y = x;
else
  y = zeros(1,I*N-I-1);
  for k = 1:N
    y(I*k - I + 1) = x(k);
    end;
  end;
```

B.3 Code Function psequence

Listing B.3: psequence

```
%
% psequence.m
%
% This routine creates the desired two-scale sequence p
% which is associated with the lowpass QMF filter. The
% parameters for the Daubechies wavelets are from Table 6.1
% (p. 195) of I. Daubechies, Ten Lectures on Wavelets.
%

function p = psequence(order)

switch order
% Haar wavelet (Daubechies 2-tap wavelet)
   case 2
      p = [ 1 1 ];
% Daubechies 4-tap wavelet (N = 2 in Daubechies notation)
   case 4
      p = (1/4)*[ 1+sqrt(3) 3+sqrt(3) 3-sqrt(3) ...
            1-sqrt(3) ];
% Daubechies 6-tap wavelet (N = 3 in Daubechies notation)
   case 6
      p = sqrt(2)*[ .3326706 .8068915 .4598775 ...
               -.13501102 -.085441274 .03522629 ];
% Daubechies 8-tap wavelet (N = 4 in Daubechies notation)
   case 8
      p = sqrt(2)*[ .23037781 .71484657 .630880768 ...
               -.027983769 -.187034812 .030841382 ...
               .0328830117 -.010597402 ];
% Daubechies 10-tap wavelet (N = 5 in Daubechies notation)
   case 10
      p = sqrt(2)*[ .1601023980 .6038292698 .7243085284 ...
               .1384281459 -.2422948871 -.0322448696 ...
               .0775714938 -.0062414902 -.0125807520 ...
```

```
                .0033357253 ];
% Daubechies 12-tap wavelet (N = 6 in Daubechies notation)
   case 12
      p = sqrt(2)*[ .11154074335 .494623890398 ...
                    .751133908021  .315250351709 ...
                   -.226264693965 -.129766867567 ...
                   .097501605587 .027522865530 ...
                   -.031582039317 .0005538422011 ...
                   .0047772575109 -.0010773010853 ];
end
```

B.4 Code Function phifunc

Listing B.4: phifunc

```
%
%                          phifunc.m
%
% This function returns phi(k/2^J) for any integer k.  It
% needs the output from makephi.m.  Note that N = length(p)
% - 1, where p is the p-sequence vector from psequence.m.
%

function y = phifunc(k,phi,J,N)

if k <= 0
  y = 0;
elseif k >= (2^J)*N
  y = 0;
else
  y = phi(k+1);
end;
```

B.5 Code Function dechain

Listing B.5: dechain

```
%                         dechain.m
```

```
% This archive introduces the output of a decimator into the
% input of another one "nt" times.
%
% An identity matrix of size "N" is introduced to the first
% decimator. This input represents white Gaussian noise with
% variance equal to one.
%

nt = 8;
N = 320;

Rin = eye(N);

em = zeros(1,nt);
eM = zeros(1,nt);
em(1) = 1;
eM(1) = 1;
cndi(1) = 1;

for i=1:(nt-1)
  [Rout,eM,em,cndi] = decimator(Rin,eM,em,i);
  Rin = Rout;
end

figure(1);
plot(em(2:nt));
title('Minimum Eigenvalue Bound');
xlabel('Number of interconnected decimators');
ylabel('Eigenvalue Bound');

figure(2);
plot(eM(2:nt));
title('Maximum eigenvalue bound');
xlabel('Number of interconnected decimators');
ylabel('Bound');
```

```
figure(3);
plot(cndi);
title('Condition Number');
xlabel('Number of interconnected decimators');
ylabel('Condition Number');
```

B.6 Code Function decimator

Listing B.6: decimator

```
%                          decimator.m
%
% This routine computes the output covariance matrix of a
% decimator given a correlation input matrix with size NxN
%
% The maximum and minimum eigenvalues of the output
% covariance are also calculated, and the accumulated
% maximum and minimum bounds are computed again and added to
% the vectors eigMaxAcc,eigMinAcc respectively.
%
% The covariance matrix is plotted in a gray scale
%
% This routine needs psequence.m and shift.m
%


function [Ry,eigMaxAcc,eigMinAcc,cnd] = ...
         decimator(Rx,eigMaxAcc,eigMinAcc,numb)

N = length(Rx);
p = psequence(8);     % obtain the wavelet coefficients

h = (1/sqrt(2))*p;
ht = fliplr(h);
M = length(h);

c = [ ht zeros([1,N-1]) ];
```

```
c = c.';

H = [ c ];

for n = 1:N-1
  H = [ H shift(c,n) ];
end;

L = floor((N+M-1)/2);

% Compute Output Covariance to Input Rx
R = H*Rx*H';
r = diag(R);

% Compute Ry = DHRxH'D', where D is the down-sampling
% matrix

for i=1:L
  for j=1:L
    Ry(i,j) = R(2*i,2*j);
  end;
end;

% Compute Output's Covariance max and min eigenvalues and
% condition number
d = eig(Ry);
Meig = max(d);
meig = min(d);
cnd(numb+1) = cond(Rhy);

% Calculate the accumulate max and min eigenvalues
eigMaxAcc(numb+1) = eigMaxAcc(numb) * Meig;
eigMinAcc(numb+1) = eigMinAcc(numb) * meig;

variances = diag(Ry);
```

```
clf

y = abs(Ry);

bigy = eigMaxAcc(numb+1);

for i=1:L
  for j=1:L
   y(i,j) = bigy - y(i,j);
  end
end

% Plot the values of the matrix as a shade of gray colors
% square surface.

y = y/bigy;
y = 256 * y;

minval = - ceil((L-1)/2) + 0.5;

%Axis = minval:minval+L-1;
Axis = 0:L-1;

figure(4);
image(Axis,Axis,y);
colormap(gray(256));

title('Gray scale correlation values for chain decimator')
```

B.7 Code Function shift

Listing B.7: shift

```
%
% shift.m
%
% This routine down-shifts a vector x by
```

```
% n adding n zeros to the vector.
%

function y = shift(x,n)

N = length(x);

if n==0
  y = x;
else
  x = x.';
  y = [ zeros([1,n]) x(1:N-n) ];
  y = y.';
end;
```

B.8 Code Function unitapproxr

Listing B.8: unitapproxr

```
%
%                              unitapproxr.m
%
% This routine computes the positive wavelet projection
% 'f_m^r' for the unit square pulse f(t) = 1 for 0 <= t <= 1
% in a Recursive and non Recursive manner.
%

function [interval,f] = unitapproxr(m)

J = 0;
r = .54;

p = psequence(4);
N = length(p)-1;
[tphi,phi] = makephi(p,J);

offset = 2^m;
```

```
scale = 2^J;
N1 = -N;
N2 = 2*offset -1;

% b1 will store the non recursive projection coefficients
% and b2 will store the recursive projection coefficients.
[b1,b2] = UnitSqCoeff(m,N2,r,p);

t1 = -2^(m+J);
t2 = 2*2^(m+J);
for k = t1:t2
  f(k - t1 + 1) = 0;
  f1(k - t1 + 1) = 0;
  for n = N1:N2
    pf = phifunc(k-n*scale,phi,J,N);
    un(k - t1 + 1) = 0;
    f(k - t1 + 1) = f(k - t1 + 1) + b1(n - N1 + 1)*pf;
    f1(k - t1 + 1) = f1(k - t1 + 1) + b2(n - N1 + 1)*pf;

  end;
end;
-t1:-t1+2^(m+J)
un(1,2^(m+J)+1:2*2^(m+J)+1)=1;
f = (2^(m/2))*f;

f1 = (2^(m/2))*f1;

interval = [t1:t2]/(offset*scale);

clf

% Plot the non recursive projection and the unit square
% pulse.

hold on;
plot(interval,f,'-'), grid
```

```
plot(interval,un)
xlabel(' t ')
ylabel(' f_{ m}^{ r} (t) ')
hold off;

% Plot the recursive projection and the unit square pulse.

figure;
hold on;
plot(interval,f1,'-'), grid
plot(interval,un)
xlabel(' t ')
ylabel(' f_{ m}^{ r} (t) ')
hold off;
```

B.9 Code Function UnitSqCoeff

Listing B.9: UnitSqCoeff

```
%
%                          UnitSqCoeff.m
%
%
% This routine computes the g-sequence in a non recursively
% manner. In addition it computes the projection
% coefficients b-sequence using series and recursive
% approaches. This function can be only used to compute the
% coefficients of a square pulse signal.
%
% This function needs h0func.m.
%
% User inputs:
%      m = index on V_m and m >= 0 is assumed
%      N2 = number of points used to compute the projection
%      r = a number between 0 and 1 chosen to control
%          convergence behavior of the series according to
%          the theory in Walter and Shen (Positive
```

```
%          Estimation with Wavelets).
%      p = required p-sequence for the scaling function used
%          in the series (obtained using psequence.m)
%
% The routine returns b1 (b-sequence computed using a series
% expansion), and b2 (b-sequence computed recursively).
%

function [b1,b2] = UnitSqCoeff(m,N2,r,p)

N = length(p) - 1;
M = 2^m;

% Set up v-axis; t(v) has support on [-2^m + 1,N-1]
v = [-M:1:N];

% Compute matrix Mx

for i = 1:N-1
  for j = 1:N-1
    k = 2*i - j + 1;
    if (k > 0) & (k < N+2)
      Mx(i,j) = p(2*i - j + 1);
    else
      Mx(i,j) = 0;
    end
  end;
end;

% Compute the integral h^{(0)}(x) of the scaling function on
% the dyadic points over the interval [0,N]

I = toeplitz([ 1 zeros(size([1:N-2]))]);
A = I - .5*Mx;

% Compute the integral h0(x) for x = 1,2,...,N-1
```

```
for k = 1:N-1
  if (k >= 1) & (k <= (N-1)/2)
    c(k) = 0;
  else
    c(k) = 0;
    for j = N:2*k
      c(k) = c(k) + .5*p(2*k - j + 1);
    end;
  end;
end;

h0 = inv(A)*c.';      % integral for x = 1,2,...,N-1
h0 = [ 0 ; h0 ; 1 ]; % integral for x = 0,1,2,...,N-1,N


% Compute the coefficients f_m,k (projection coordinates of
% f(t) onto space V_m;)

for k = 1:length(v)
  f(k) = h0func(h0,M + v(k)) - h0func(h0,v(k));
end;
f = f/sqrt(M);

% Compute the g-sequence using series expansion which
% gives the vector g1

for k = -N:N2
  g1(k+N+1) = 0;
  for j = 0:(N2 -p)
    indx = -j - k;
    if (indx >= -M & indx <= N)
      g1(k+N+1) = g1(k+N+1) + r^(j)*f(M+indx + 1);
    end;
  end;
end;
```

```
% Compute the b-sequence using series expansion which gives
% the vector b1

beta = (1-r).^2;

for n = -N:N2
  b1(n + N + 1) = 0;
  for k = 0:M
    g = 0;
    for j = 0:(M+k -n)
      indx = k - j - n;
      if (indx >= -M & indx <= N)
      p = n-k;
      g = g + r^(j)*f(M+indx + 1);
      end;
    end;
    b1(n + N + 1) = b1(n + N + 1) + r^(k)*g;
  end;
end;

b1 = beta*b1;

% Compute the b-sequence using the recursive form.

c1 = r;

b2(1) = b1(1);

for n = -N:N2-1
  n+N+2;
  b2(n+N+2) = 0;

  indx = n+1;
  b2(n+N+2) = b2(n+N+2) + beta*g1(N +indx + 1);
```

```
  b2(n+N+2) =  b2(n+N+2)+ c1*b2(n+N+1);
end;
```

B.10 Code Function triangapproxr

Listing B.10: triangapproxr

```
%
%                              triangapproxr.m
%
% This routine computes the positive wavelet projection
% 'f_m^r' for the triangular pulse f(t) = t for 0 <= t <= 1
% in a Recursive and non Recursive manner.
%

function [interval,f] = triangapproxr(m)

J = 0;
r = .50;

p = psequence(4);
N = length(p)-1;
[tphi,phi] = makephi(p,J);

offset = 2^m;
scale = 2^J;
N1 = -N;
N2 = 2*offset -1;

% Compute the projection coefficients b1 has the non
% recursive coefficients and b2 has the recursive components

[b1,b2] = TriangCoeff(m,N2,r,p);

t1 = -2^(m);
t2 = 2*2^(m);
```

```
% Compute the projections f is the non recursive projection
% and f1 is the recursive projection.

for k = t1:t2
  f(k - t1 + 1) = 0;
  f1(k - t1 + 1) = 0;
  for n = N1:N2
    pf = phifunc(k-n*scale,phi,J,N);
    f(k - t1 + 1) = f(k - t1 + 1) + b1(n - N1 + 1)*pf;
    f1(k - t1 + 1) = f1(k - t1 + 1) + b2(n - N1 + 1)*pf;
  end;
end;

f = (2^(m/2))*f;

f1 = (2^(m/2))*f1;

interval = [t1:t2]/(offset*scale);

clf

% Plot the non recursive projection

plot(interval,f,'-'), grid
xlabel(' t ')
ylabel(' f_{ m}^{ r} (t) ')

% Plot the recursive projection

figure;
plot(interval,f1,'-'), grid
xlabel(' t ')
ylabel(' f_{ m}^{ r} (t) ')
```

B.11 Code Function TriangCoeff

Listing B.11: TriangCoeff

```
%
%                          TriangCoeff.m
%
%
% This routine computes the g-sequence and
% the b-sequence both using series and
% recursive approaches.  Obviously, the two methods ought to
% agree with each other.  This is done on the assumption
% that the input signal is a triangular function.
%
% This function needs h0func.m and h1func.m.
%
% User inputs:
%      m = index on V_m and m >= 0 is assumed
%      N2 = number of points used to compute the projection
%      r = a number between 0 and 1 chosen to control
%          convergence behavior of the series according to
%          the theory in Walter and Shen (Positive
%          Estimation with Wavelets).
%      p = required p-sequence for the scaling function used
%          in the series (obtained using psequence.m)
%
% The routine returns b1 (b-sequence computed using a series
% expansion), and b2 (b-sequence computed recursively).
%

function [b1,b2] = TriangCoeff(m,N2,r,p)

N = length(p) - 1;
M = 2^m;

% Set up v-axis; t(v) has support on [-2^m + 1,N-1]
v = [-M:1:N];

% Compute matrix Mx
```

```
for i = 1:N-1
  for j = 1:N-1
    k = 2*i - j + 1;
    if (k > 0) & (k < N+2)
      Mx(i,j) = p(2*i - j + 1);
    else
      Mx(i,j) = 0;
    end
  end;
end;

% Compute the integral h^{(0)}(x) of the scaling function on
% the dyadic points over the interval [0,N]

I = toeplitz([ 1 zeros(size([1:N-2]))]);
A = I - .5*Mx;
A1 = I - .25*Mx;

% Compute the integrals h0(x) and h1(x) for x = 1,2,...,N-1

for k = 1:N-1
  if (k >= 1) & (k <= (N-1)/2)
    c(k) = 0;
  else
    c(k) = 0;
    for j = N:2*k
      c(k) = c(k) + .5*p(2*k - j + 1);
    end;
  end;
end;

% integral for x = 1,2,...,N-1
h0 = inv(A)*c.';

m1 = 0;
```

```
for k = 0:N
  m1 = m1 + k*p(k+1);
end;
m1 = .5*m1;   % scaling function first moment

c1 = .5*m1*c;

for k = 1:N-1
  eta1(k) = 0;
  for n = (2*k - N):(2*k-1)
    if ( n >= 1 ) & ( n <= (N-1) )
      eta1(k) = eta1(k) + .25*p(2*k - n + 1)*h0(n);
    elseif ( n >= N )
      eta1(k) = eta1(k) + .25*p(2*k - n + 1);
    end;
  end;
end;
h1 = inv(A1)*(c1 + eta1).'; % integral for x = 1,2,...,N-1

h0 = [ 0 ; h0 ; 1 ];  % integral for x = 0,1,2,...,N-1,N
h1 = [ 0 ; h1 ; m1 ]; % integral for x = 0,1,2,...,N-1,N


% Compute the coefficients f_m,k (projection coordinates
% of f(t) onto space V_m; see (4.1))

for k = 1:length(v)
  f(k) = h1func(h1,m1,M + v(k)) - h1func(h1,m1,v(k));
  f(k) = f(k) - v(k)*( h0func(h0,M + v(k)) - h0func(h0,v(k)) );
end;
f = f/(M.^(3/2));

% Compute the g-sequence using series expansion (4.8) which
% gives the vector g1

for p = -N:N2
```

```
  g1(p+N+1) = 0;
  for j = 0:(N2 -p)
    indx = -j - p;
    if (indx >= -M & indx <= N)
      g1(p+N+1) = g1(p+N+1) + r^(j)*f(M+indx + 1);
    end;
  end;
end;

c1 = -1/r;
c2 = 1/r;

% Compute the b-sequence using series expansion.

beta = (1-r).^2;

for n = -N:N2
  b1(n + N + 1) = 0;
  for k = 0:M
    g = 0;
    for j = 0:(M+k -n)
      indx = k - j - n;
      if (indx >= -M & indx <= N)
      p = n-k;
      g = g + r^(j)*f(M+indx + 1);
      end;
    end;
    b1(n + N + 1) = b1(n + N + 1) + r^(k)*g;
  end;
end;

b1 = beta*b1;

% Compute the b-sequence using the recursive method.

c1 = r;
```

```
b2(1) = b1(1);

for n = -N:N2-1
  n+N+2;
  b2(n+N+2) = 0;

  indx = n+1;
  b2(n+N+2) = b2(n+N+2) + beta*g1(N +indx + 1);

  b2(n+N+2) =  b2(n+N+2)+ c1*b2(n+N+1);
end;
```

B.12 Code Function h0func

Listing B.12: h0func

```
%
%                    h0func.m
% This function computes some values needed to project a
% square and a triangular signal into a wavelet space.
%

function x = h0func(h0,k)

N = length(h0)-1;
if k < 0
  x = 0;
elseif k > N
  x = 1;
else
  x = h0(k+1);
end;
```

B.13 Code Function h1func

Listing B.13: h1func

```
%
```

```
%                        h1func.m
% This function computes some values needed to project a
% triangular signal into a wavelet space.
%

function x = h1func(h1,m1,k)

N = length(h1)-1;
if k < 0
  x = 0;
elseif k > N
  x = m1;
else
  x = h1(k+1);
end;
```

B.14 Code Function DiscRecProj

Listing B.14: DiscRecProj

```
%         DiscRecProj.m
%
% This function is designed to probe the discrete positive
% sampling proposed by G. Walter.
% The test signal will be constructed inside the function
% and it will be a train of two unit square pulses.

function DiscRecProj(m,r)

% m equals the order of the representation
% The series will we computed with an interval of 2^(m)

offset = 2^m;

% Definition of impulse train time length = n

tf = ones(1,10);
```

```
tf(1,5:7) = 0;
l_n_tf = length(tf);

f = zeros(1,(l_n_tf)*offset);

for i = 0 : l_n_tf-1
   f(1,i*offset+1:(i+1)*offset) = tf(i+1);
end

l_n = length(f);
interval = [0:l_n-1]/offset;

% Plot function to be projected
figure(1);
plot(interval,f);
grid
xlabel(' t ')
ylabel(' f (t) ')

% psequence will return the values for building Daubechies
% wavelet
p = psequence(4);
N = length(p)-1;

% phi will have the scaling function values from zero in
% the integer points.
[tphi,phi] = makephi(p,0);

N1 = -N+1;
N2 = offset*(l_n_tf) - 1;

% Compute the discrete projection coefficients
b = DiscPrCoeff(m,N1,N2,f,r);

% Compute the projection of the function f using the
% discrete coefficients
```

```
t1 = 0;
t2 = offset*(l_n_tf);
for k = t1:t2
  f_p(k+1) = 0;
  for n = N1:N2
    pf = phifunc(k-n,phi,0,N);
    f_p(k+1) = f_p(k+1) + b(n - N1 + 1)*pf;
  end;
end;
f_p = (2^(m/2))*f_p;

% Plot the projected function

interval = [t1:t2]/offset;

figure(2);
clf

plot(interval,f_p,'-')
grid
xlabel(' t ')
ylabel(' f_{m}^{ r } (t) ')
```

B.15 Code Function DiscPrCoeff

Listing B.15: DiscPrCoeff

```
%
%                              DiscPrCoeff.m
%
% This function computes the Discrete projection
% coefficients of a function f.
%
% User inputs:
%      m = index on V_m and m >= 0 is assumed
%      N1, N2 = define range of coefficients b_n
%               (i.e., n = N1,...,N2)
```

```
%       r = a number between 0 and 1 chosen to control
%           convergence behavior of the series according to
%           the theory in Walter and Shen (Positive
%           Estimation with Wavelets).
%

function b = DiscPrCoeff(m,N1,N2,f,r)

offset = 2^m;
l_n = length(f);

n_terms = l_n;

for n = N1:N2
  b(n - N1 + 1) = 0;

   g = 0;
   for l = n - (l_n -1):n
      if (n-l) >= 0 & (n-l)<n_terms
         g = g + r^(abs(l))*f(n-l+1);
      end
   end;
   b1(n - N1 + 1) = g;

end;

b1 = (2^(-m/2))*(1-r)/(1+r)*b1;
b = b1;
```

B.16 Code Function QRMdl

Listing B.16: QRMdl

```
%                    QRMdl.m
%
%
%  This function computes the GCD of a system with the form
```

```
% H    = A1/B1=A*d/B*d when the matrices have noise.
%
% The noisy model equation is assumed to be
% Hn= (A1 + n1)/(B1 + n2) where noise sources n1 and n2 are
% uniform noise defined over the interval [-e,e].  The
% algorithm is based on "MDL criterion for Rank
% Determination Via Effective Singular Values" where the MDL
% rank estimator is changed to a positive wavelet
% decomposition criteria.
%
% A and B are the coefficients of the system and d are the
% coefficients of the common factor. Vector e has the levels
% of noise that will test the system.
%

function ma=QRMdl(A,B,d,e)

% Definition of the terms extracted from the model given in
% Zarowski MDL paper

A1 = conv(A,d);
B1 = conv(B,d);
%A1 = A;
%B1 = B;

lA1 = length(A1);
lB1 = length(B1);

% Define the maximum deviation. We will use uniform noise.

ma = zeros(length(e),lA1);

figure(1);
clf;
figure(2);
clf;
```

```
% Reproduce the test 100 times for every maximum level of
% noise given on vector e

for i = 1:length(e)
   for j = 1:100
      n1 = e(i)*(rand(size(A1)) - 0.5);
      n2 = e(i)*(rand(size(B1)) - 0.5);

      An = A1 + n1;
      Bn = B1 + n2;

      % Generate the four matrices given in Zarowski MDL paper.

      S1 = zeros(lA1-1);
      S2 = zeros(lA1-1);
      S3 = zeros(lB1);
      S4 = zeros(lB1);

      TempRow1 = [An(1:end-1)];
      TempRow2 = [An(2:end)];

      TempRow3 = [0 Bn(1:end-1)];
      TempRow4 = Bn;


      for k = 1:lA1-1
         S1(k,:) = TempRow1;
         TempRow1 = [0,TempRow1([1:end-1])];
         S2(lA1-k,:) = TempRow2;
         TempRow2 = [TempRow2([2:end]),0];
         S3(k,:) = TempRow3;
         TempRow3 = [0,TempRow3([1:end-1])];
         S4(lA1-k,:) = TempRow4;
         TempRow4 = [TempRow4([2:end]),0];
      end
```

```
        S1Inv = inv(S1);

        K = -S3*S1Inv;
        % Obtain the desired QR decomposition
        [Q,R] = qr(K'*S2'+S4');

        % Use the ICE algorithm on matrix R
        alp = ICE(R,j);

        [q,m] = MDL2(alp,j);

        [q,m] = MDL1(alp,j);

        ma(i,q) = ma(i,q) + 1;
    end
end
```

B.17 Code Function ICE

Listing B.17: ICE

```
%                              ICE.m
% Function to obtain the Incremental Condition Estimation
% for all the principal submatrices of the upper triangular
% matrix R.
%

function di=ICE(R,ite)

% Transpose matrix R to obtain a lower triangular matrix

R = R';

x(1) = 1/R(1,1);
di(1) = 1/norm(x) ;
```

```
for i=2:size(R,1)
   v = R(i,1:i-1);
   alp = v*x;
   beta = R(i,i)^2*x'*x + alp^2 - 1;
   n = beta/(2*alp);
   u = n + sign(alp)*sqrt(n^2+1);
   lmax = alp*u +1;
   ny = sqrt(lmax)/abs(R(i,i));
   di(i) = 1/ny;
   if (alp ~= 0)
      fact = 1/sqrt(u^2+1);
      s = fact*u;
      c = fact*(-1);
   else
      s = 0;
      c = 1;
      if abs(R(i,i))*norm(x) > 1
         s = 1;
         c = 0;
      end
   end

   x = [s*x ; (c-s*alp)/R(i,i)];
end
di = di';

% Plot the estimated singular values for the first
% iteration. (Used when this function is called multiple
% times inside another routine).
if (ite == 1)
   figure(3);
   plot(di);
end
```

B.18 Code Function MDL1

Listing B.18: MDL1

```
%                              MDL1.m
%
% This function computes the MDL for for a positive sampling
% series without the q/2 coefficient
%

function [q_m,m_m] = MDL1(alp,iter)

% Total number of given  singular values.

n = length(alp);
N = 1;

m = 1;
m_m =1;
% Calculate first time set the minimum to this value

% We will obtain the first term on the MDL supposing we only
% have one singular value different than zero and m=0
% (space V0).

q = 2;

% Calculate the value of rho_q
rho_q = sum(alp(1:q).^2);

% mdl_min will store the minimum value obtain for the MDL.
% Due to the fact that this is the first iteration we will
% store the MDL straight into mdl_min. The MDL is computes
% according to the formula obtained. (See research notes for
% more information).


mdl_min = ((n - q/2)*log2(exp(1)) ...
          - (n-q)*log2((n-q)/sum(alp(q+1,end)))...
```

```
            + q/2*log2(2*pi/q*rho_q*(1-1/q)));

% q_m will store the number of terms "q" of the MDL and "m"
% will store the order of the projection space used to
% obtain the MDL.
q_m = 2;

% Compute the value for the MDL for the remaining values of
% "q" and "m". At the end of this recursion the MDL will be
% stored in mdl_min and the corresponding "q" and "m" values
% will be stored in "q_m" and "m_m" respectively.

for (q=2:n-1)
   rho_q = sum(alp(1:q).^2);
   rem = sum(alp(q+1,end));
   mdl(q) = ((n - q/2)*log2(exp(1)) ...
            - (n-q)*log2((n-q)/rem)...
            + q/2*log2(2*pi/q*rho_q*(1-1/q)));

   if (mdl(q) < mdl_min)
      mdl_min=mdl(q);
      q_m = q;
   end
end

% Plot the result obtained in the first iteration when this
% function is called multiple times inside another routine

if (iter == 1)
   figure(1);
   grid on;
   hold all;
   plot(mdl);
   hold off;
end
```

B.19 Code Function MDL2

Listing B.19: MDL2

```
%                                   MDL2.m
%
% This function computes the MDL for for a positive sampling
% series without the q/2 coefficient
%


function [q_m,m_m] = MDL2(alp,iter)

% Total number of given  singular values.

n = length(alp);
N = 1;

m = 1;
m_m =1;
% Calculate first time set the minimum to this value

% We will obtain the first term on the MDL supposing we only
% have one singular value different than zero and m=0
% (space V0).

q = 2;

% Calculate the value of rho_q
rho_q = sum(alp(1:q).^2);

% mdl_min will store the minimum value obtain for the MDL.
% Due to the fact that this is the first iteration we will
% store the MDL straight into mdl_min. The MDL is computes
% according to the formula obtained. (See research notes for
% more information).
```

```
mdl_min = (q*log2(n)+ (n - q/2)*log2(exp(1)) ...
          - (n-q)*log2((n-q)/sum(alp(q+1,end)))...
          + q/2*log2(2*pi/q*rho_q*(1-1/q)));

% q_m will store the number of terms "q" of the MDL and "m"
% will store the order of the projection space used to
% obtain the MDL.
q_m = 3;

% Compute the value for the MDL for the remaining values of
% "q" and "m". At the end of this recursion the MDL will be
% stored in mdl_min and the corresponding "q" and "m" values
% will be stored in "q_m" and "m_m" respectively.

for (q=2:n-1)
   rho_q = sum(alp(1:q).^2);
   rem = sum(alp(q+1,end));
   mdl(q) = (q*log2(n)+(n - q/2)*log2(exp(1)) ...
             - (n-q)*log2((n-q)/rem)...
            + q/2*log2(2*pi/q*rho_q*(1-1/q)));

   if (mdl(q) < mdl_min)
      mdl_min=mdl(q);
      q_m = q;
   end
end

% Plot the result obtained in the first iteration when this
% function is called multiple times inside another routine

if (iter == 1)
   figure(2);
   grid on;
   hold all;
   plot(mdl);
```

```
    hold off;
end
```