DISCRETE WAVELETS ON $\mathbb{Z}_N$

Alberto Debernardi Pinos

Adviser: F. Javier Soria de Diego

Advanced Mathematics Master Thesis
University of Barcelona
Barcelona, June 2014

Contents

Acknowledgments

To begin I want to thank my advisor F. Javier Soria de Diego for all the help brought during the production of this thesis, and for bringing me the opportunity to develop my own (small) result, which was very gratifying (see Section 2.4). Also, for all the hours he spent on revising the former versions of this document (which were not few), and the support he brought me to solve some LaTeX and MATLAB issues I had, and last, but not least, for the working guidelines defined from the very beginning, that allowed me to successfully complete this work (relatively) fast and rigorously. I also appreciate the effort of all the professors I had during this year, who made me step up a lot in my mathematical knowledge.

Next, I want to thank all my Master's colleagues that made this year to be less tough (in terms of the hard work that this course requires), and specially Brendan, Víctor Á. and Dani G., with whom I have established a good friendship during this year. I also want to mention my closest friends Alberto and Eric, and my beloved Ainoa, who supported me every single moment during the course, and helped me with everything she could, as well as my family, who are taking care of me (in the economical sense) in these hard times.

Finally, I want to give credit to all the authors cited: all of them put their bit in the developing of this work, especially M. Frazier, whose book [7] is the cornerstone of this document. Most of the results were extracted from this book, and I do not claim to own any of these. Also, special mention to [3] and [4], which were crucial for the completion of my own result, since they are closely related to it.

Chapter 1

Introduction

This Master's thesis is devoted to develop the theory of discrete (finite-dimensional) wavelets. In this context, wavelets are very concrete basis of vectors with special properties that make them suitable for the purpose of data compressing or digital signal processing, among others. Thus, this will be our ultimate goal in the study of wavelets. Since we are considering finite-dimensional vectors, the whole theory is based in linear algebraic techniques. This theory can also be extended for vectors in $\mathbb{Z}$, and for functions in $\mathbb{R}$; actually, there will be several analogies between these different settings that will be remarked throughout the work. In fact, this theory was firstly studied in $\mathbb{R}$, being [10] and [11] its predecessors. After introducing the main results in the discrete setting, we also give several examples of data compression. Surprisingly, wavelets will allow us to reduce significantly the amount of information needed to reproduce a vector (or a matrix), which is very important for data storage and transmission, since the less information we need, the less resources we have to use.

Chapter 2 presents all the background needed: it begins with the definition of the space of vectors we are going to work with, $\ell^2(\mathbb{Z}_N)$ (Definition 2.1.1). As $\ell^2(\mathbb{Z}_N)$ is a finite-dimensional $\mathbb{C}$-vector space, we will be able to use several well-known results of linear algebra.

The main concept is the Fourier basis, the orthonormal basis for $\mathbb{C}^N$ whose vectors F_m are of the form

$$(F_m)_n = e^{2\pi inm/N},\ n = 0, \dots, N-1.$$

Such a basis leads to a linear transformation through the complex inner product, called the Discrete Fourier Transform (DFT) $\hat{z}$ of a vector z. As we will see, many results we are going to obtain throughout this work are expressed in terms of the DFT (and specially the ones related to wavelets). In fact, this is because such a transformation has very good properties. For instance, we have the formula for the convolution of two vectors

$$\widehat{z * w} = \hat{z}\hat{w} \qquad \text{(component-wise product)}$$

shown in Lemma 2.1.8. Another remarkable property is that we do not need all the N^2 products required to compute a linear transformation, since some cancellations

occur in the expression of the DFT. There is a fast algorithm which exploits this fact, named Fast Fourier Transform (FFT), and it is developed in Section 2.2. This algorithm can reduce the number of complex multiplications to $N \log N$, when N is a power of 2, which is a very good fact from the computational point of view.

Later on, in Section 2.3, we focus our interest in data compression, introducing the concept of localization (Definitions 2.3.1 and 2.3.3). Basis which are localized work properly for data compression, and since the DFT has very nice properties, we could ask ourselves if it is localized, but that is not the case. Contrarily, wavelet basis have the localization property, and that is why they will become our main object of study. Finally, we complete the chapter with Section 2.4, which is devoted to the uncertainty principle, a well-known (in all of its forms) principle that gives a lower bound for the number of nonzero elements of a vector and its DFT. More specifically, if we denote by $H(z), H(\hat{z})$ the number of nonzero elements of z and $\hat{z} \in \ell^2(\mathbb{Z}_N)$ respectively, then we have that

$$H(z)H(\hat{z}) \geq N.$$

Motivated by the uncertainty principle, a new question arises. Defining $L(N)$ as the maximum number of zero elements that z and $\hat{z}$ can have at the same time, for any z, we ask ourselves how big can $L(N)$ be. The development for the computation of these numbers is the most original part of this Master's thesis, and it concludes that we can compute the number $L(N)$, for every N, solving the inequalities given in Proposition 2.4.14. This can be done through the explicit formulas for the numbers H_{F_N} appearing in those inequalities. These formulas were obtained by S. Delvaux and M. Van Barel in [3] and [4], and they turned to be crucial in order to obtain the conclusion for every number N. Nevertheless, there are simple expressions of N for which we could already compute the number $L(N)$, being these $N = n^2$ or $N = n(n-1)$. In fact, we realize that $L(N)$ depends on the decomposition of N rather than on its magnitude. Theorems 2.4.7 and 2.4.9 give a lower bound for those numbers based on its factorization: for instance, the first one states that if $N = nm$, the bound increases as $|n-m|$ decreases. In fact, that is why the mentioned cases n^2 and $n(n-1)$ were easy to compute, since we already had an upper bound for $L(N)$ given by the uncertainty principle, and for those special decompositions, the lower and upper bounds coincide, yielding equality. For prime numbers (bigger than 2), it follows easily from Chebotarev theorem that $L(N) = [N/2]$, and for odd powers of 2 (Example 2.4.18), we have that $L(2^{2n+1}) = 2^{2n+1} - 2^{n+1}$, which coincides with the lower bound given in Theorem 2.4.7. At the end of the chapter, we show a table containing the values of $L(N)$, for the first 99 natural numbers.

Chapter 3 begins introducing two operations for vectors of $\ell^2(\mathbb{Z}_N)$: the conjugate reflection (Definition 3.1.1) and the circular translation (Definition 3.1.3), and their properties related to the DFT. An important result is Lemma 3.1.4, which allows us to compute inner products of a signal z and the translations $R_k w$ of a vector w through the convolution $z * w$. Thanks to the relation of the convolution of two vectors with the DFT mentioned earlier, we are allowed to compute inner products, and therefore, the coefficients $\langle z, w_j \rangle$ of a vector z in a specific basis $\mathcal{B} = \{w_j\}_{j=0}^{N-1}$

using the FFT. In principle, this may not seem so important, but by Lemma 3.1.4, we note that if $\mathcal{B}$ contains many translations of a single vector, then we will need to compute less convolutions in order to obtain the coefficients of z in the basis $\mathcal{B}$. Actually, a (first-stage) wavelet basis is defined as an orthonormal basis generated by the even translations of two vectors u and v (Definition 3.1.6), so that we only need two convolutions in order to compute the expansion of a vector in such a basis. Recall that we want to find basis of vectors which are localized in space and frequency; wavelet basis will have this property. Although, we first approach the problem of finding localized basis looking for basis consisting of translations of a single vector. This approach will not be successful, since, by Lemma 3.1.5, we have that a basis of this kind is not frequency localized (and the very first example of this fact is the Euclidean basis). The next step of this approach, i.e., allowing the basis to be formed by translations of two vectors instead of only one, will be the good choice. The rest of Section 3.1 contains some auxiliary results to set up the background for the proof of Theorem 3.1.14, the main result of the chapter. It gives a characterization of wavelet basis in a very simple way. Summarizing, this theorem asserts that

$$\{R_{2k}u\}_{k=0}^{N/2-1} \cup \{R_{2k}v\}_{k=0}^{N/2-1}$$

is a first-stage wavelet basis if and only if the equalities

$$|\hat{u}_n|^2 + |\hat{u}_{n+M}|^2 = 2, \qquad |\hat{v}_n|^2 + |\hat{v}_{n+M}|^2 = 2, \qquad \hat{u}_n\overline{\hat{v}_n} + \hat{u}_{n+M}\overline{\hat{v}_{n+M}} = 0$$

hold for every $n = 0, 1, \ldots, N/2 - 1$. To conclude, we show on Proposition 3.1.16 how to construct a wavelet basis using only a vector u such that $\{R_{2k}u\}_{k=0}^{N/2-1}$ is an orthonormal set with $N/2$ elements. Note that this set can potentially be a wavelet basis generator. In fact, we give an expression of v depending on u that makes u and v generate a wavelet basis for $\ell^2(\mathbb{Z}_N)$. We give examples of four well-known wavelet basis in Section 3.2. The first one is the Haar basis (Definition 3.2.1), which can be checked to be a wavelet basis directly from the definition. The linear transformation yielded by this basis can be seen as an increasing of resolution, as illustrated in Example 3.2.3, which shows the 3rd order Haar transform of a vector (that is, the iteration of the Haar first-stage wavelet basis as a $3rd$-stage wavelet basis, developed in Chapter 4. Next, we have the Shannon wavelet basis: one with complex vectors, and another one with real-valued vectors. We mainly focus in the real Shannon basis, since the signals we are interested in are real-valued (mainly audio signals and gray-scale pictures). Shannon basis are not defined explicitly, but through the DFTs $\hat{u}$ and $\hat{v}$ and the characterization of Theorem 3.1.14. Moreover, real Shannon basis turns out to be nicely localized, being this a good choice for the purpose of data compression. The last basis presented in the chapter is Daubechies D6 basis, whose generators are chosen to have only six nonzero entries, and therefore it is well localized, yielding good compression as well. Moreover, for any even number K, we can define Daubechies DK wavelet basis, where the generators are chosen to have K nonzero elements. Actually, the case $K = 2$ coincides with the Haar basis.

Chapter 4 is devoted to create orthonormal basis for $\ell^2(\mathbb{Z}_N)$ that will yield even better properties than the first-stage wavelets defined in the preceding chapter. Since

those basis will be obtained from a first-stage wavelet applying an iteration to one of its generators (namely, u). This process returns the so-called p-th stage wavelet basis. The iteration is described quite simply. Let p be a positive integer and suppose that 2^p divides N. Choose a wavelet basis (generated by even translations of two vectors, u and v) for $\ell^2(\mathbb{Z}_N)$ and a signal z in the same space. Now we apply the wavelet transform to z, obtaining $N/2$ coefficients that correspond to the inner products of z with the translations of u, and another $N/2$ obtained in the same way through v. Now we can regard the coefficients related to u as another vector y, belonging to $\ell^2(\mathbb{Z}_{N/2})$. Thus, if $N/2$ is divisible by 2, and we define u^2, v^2 as the generators of the same wavelet basis in $\ell^2(\mathbb{Z}_N)$, we can apply the transform to y and obtain $N/4$ coefficients related to u^2, and the same quantity related to v^2. As long as 2^k divides the initial length N, we can keep iterating this. Although the process is very simple, the improvement we have with respect to first-stage wavelet basis is huge. For instance, in Section 4.3 we show a couple of compression methods through the 3rd stage Shannon real wavelet and 4th stage Daubechies D6 wavelet. Those examples illustrate that higher stage wavelets bring much better compression that the first stage one, or equivalently, keeping the same amount of information for both stages, the loss of information is drastically reduced if we use higher stage wavelets (whenever we keep a reasonable amount of coefficients). Section 4.1 contains the background needed to develop the first iteration of a wavelet basis (i.e., the second stage), along with a helpful change of basis algorithm (illustrated by the filter bank of Figure 4.2) to implement the wavelet transforms (and their inverses). This algorithm decomposes a signal into its coefficients in a certain basis, and then performs a reconstruction to recover the signal expressed in the original basis, but in principle we do not know if the reconstruction will be perfect. The most important result of this part is Theorem 4.1.6, which provides necessary and sufficient conditions for a basis of $\ell^2(\mathbb{Z}_N)$ to yield a perfect reconstruction of the input signal by the filter bank of Figure 4.2, or equivalently, the input signal equals the output one. It turns out that first-stage wavelet basis return a perfect reconstruction of any input signal by this algorithm, as we note in Remark 4.1.7. In Section 4.2 we develop the iteration of the above algorithm through p-th stage filter banks (Figure 4.4 shows the last step of the decomposition, as the first step of the reconstruction). The vectors $u^1, v^1, u^2, v^2, \ldots, u^p, v^p$ mentioned above will be the ones yielding such a filter bank, and they will be defined as a p-th stage wavelet (Definition 4.2.2), although that set of vectors will not be a basis, since they do not even have the same lengths, but the definition is motivated by the fact that we can get an orthonormal basis for $\ell^2(\mathbb{Z}_N)$ using them, as we conclude in Theorem 4.2.18.

Chapter 5 contains several examples of data compression, in one and two dimensions. The purpose is to illustrate all the nice properties of wavelets that we have seen, as for instance, the localization property. In Section 5.1 we show the histograms of the magnitudes of the generators' coefficients for all the basis we have presented throughout the work, as well as the histograms of their DFTs. They are shown in Figures 5.3, 5.4, and 5.5: the one on the left corresponds to the generator, and the right, its DFT. An important thing to observe is that even though the co-

efficients of the DFTs are sparse, they are very small in magnitude: none of them exceeds 0.1. On the other hand, Figure 5.1 contains the histogram of the Euclidean basis. As expected, it is not frequency localized at all. In Section 5.2 there are examples of data compression in $\ell^2(\mathbb{Z}_N)$, although we are only considering real-valued signals, since an application of all this theory is audio digital analysis[1]. In each one of the examples, we compare the compression yielded by the four following basis of vectors: Euclidean, Fourier, Shannon (real), and Daubechies D6, and we also show a plot of the relative error for each one of the approximations. The compression of a signal z is done in the following way: first we choose the basis $\mathcal{B}$ we want to use. After that, we compute the coefficients of z in $\mathcal{B}$, say $[z]_{\mathcal{B}}$, and we make zero those which are not one of the $k\%$ highest, obtaining the compression $[z]'_{\mathcal{B}}$. It is worth to mention that the real compression is done through algorithms that can transform a vector with many zeros in a vector with less components (through an invertible process), so that we need less space to save the information of the original vector. Finally, the approximation of z, say w, is obtained by writing the coefficients of the vector $[z]'_{\mathcal{B}}$ in the original basis, and therefore, the relative error made in the approximation is given by the formula

$$\frac{\|z-w\|}{\|z\|}.$$

Note that once $\mathcal{B}$ and z are fixed, this expression is a function depending on k (for each percentage of non-zero values we keep, there is a different approximation). It is decreasing, and takes values in $[0,1]$. Thus, the faster its decay is, the better is the compression. For most of the signals z, wavelet basis have nicer relative error functions than the Fourier basis (and obviously, Euclidean basis as well). Example 5.2.3 shows a nice compression through Daubechies D6 wavelet. In this case, the relative error already vanishes when $k = 20$ (see Figure 5.8). In fact, this is because such a basis' generators have many components equal to zero, as stated previously. Therefore, this basis will work properly when compressing signals with a large amount of zeros, which is the case. But not every signal z is nicely compressed by wavelet basis: in Section 5.3 we observe that sinusoidal signals do not give the desired results, and this is because the elements of the Fourier basis are very similar to sinusoids. Hence, the number of vectors in the Fourier basis needed to represent such a signal will be low. Figure 5.11 shows that even though the relative error function of the Fourier is not so good, it is still better than wavelet basis'. To conclude the chapter, we apply the previous techniques to the 2-dimensional case in Section 5.4.

Finally, Chapter 6 shows the MATLAB code implementing the compression method for the Fourier, Haar, Shannon (real), and Daubechies D6 basis, as well as the construction of the relative error function. These compression programs work for signals in 1 and 2 dimensions. The programs relative to wavelet basis will have an extra input variable, namely the one that will define the chosen stage for the iteration. Moreover, the program related to the Haar transform will not have

[1]The pitch of a sound is represented mathematically with real-valued functions, containing its frequencies. It is known that human hearing range goes from 20 Hz to 20 000 Hz.

the option of choosing the number of coefficients we will keep, since as we noted, such transform can be seen as a change of resolution. Therefore, for this basis we will always keep half of the coefficients, which will be the ones obtained from the inner products with the first generator u. The last section of this chapter (6.4) corresponds to an algorithm that computes $L(N)$ joining the results obtained form Propositions 2.4.11, 2.4.14, and Theorem 2.4.17.

Chapter 2

The Discrete Fourier Transform (DFT)

2.1 Definitions and properties

In what follows we will be working with finite sequences of complex numbers, which we can consider just to be vectors. So, our workspace will always be the following:

Definition 2.1.1. 1. We define $\mathbb{Z}_N = \{0, 1, \ldots, N-1\}$ as the ring with N elements such that if $a, b \in \mathbb{Z}$, then $a \equiv b$ in $\mathbb{Z}_N$ if and only if $a \equiv b \mod N$.

2. $\ell^2(\mathbb{Z}_N) = \big\{z = (z(0), z(1), \ldots, z(N-1)) \,:\, z(k) \in \mathbb{C}, \text{ for all } k\big\}$.

Note that if $m \in \mathbb{Z}$, we can recover its corresponding index $k \in \mathbb{Z}_N$ from the following formula:

$$k \equiv m \mod N.$$

A way of interpreting the vectors $z \in \ell^2(\mathbb{Z}_N)$ is to think of them as functions from the set $\mathbb{Z}_N$ into the complex numbers. Here we can already find a big advantage with respect to analysis: given a measurable set Ω and a function $f : \Omega \to \mathbb{R}$, it is not always true that f is square-integrable, i.e., $f \in L^2(\Omega)$. In this context, $L^2(\Omega)$ turns into $\ell^2(\mathbb{Z}_N)$ and every vector of complex numbers of length N will belong to this space, since "square-integrable" turns into "square-summable", and a sum of a finite quantity of numbers will always be finite. This is one less thing we have to worry about, and moreover, we will be able to apply the whole analysis machinery into the development of this field. Furthermore, since $\ell^2(\mathbb{Z}_N)$ is an N-dimensional $\mathbb{C}$-vector space, linear algebra will also be very useful. This leads us to the next definition:

Definition 2.1.2. Let $z, w \in \ell^2(\mathbb{Z}_N)$. The complex inner product in this space is defined as

$$\langle z, w \rangle = \sum_{k=0}^{N-1} z(k)\overline{w(k)},$$

with the Euclidean norm

$$\|z\| = \left(\sum_{k=0}^{N-1} |z(k)|^2 \right)^{1/2}.$$

We will say that z, w are orthogonal if $\langle z, w \rangle = 0$, and denote it by $z \perp w$.

Notation: From now on, we will denote the components of a vector $z \in \ell^2(\mathbb{Z}_N)$ with subindices, i.e., $z_k := z(k)$.

Definition 2.1.3. In $\ell^2(\mathbb{Z}_N \times \mathbb{Z}_M)$, we define the (2-dimensional) Fourier basis $\{F_{n,m}\}_{n,m}$, where

$$(F_{n,m})_{j,k} = \frac{1}{\sqrt{NM}} e^{2\pi i \left(\frac{jn}{N} + \frac{km}{M}\right)}, \qquad 0 \leq n \leq N-1,\, 0 \leq m \leq M-1.$$

For the 1-dimensional case we consider $M = 1$.

Proposition 2.1.4. *The Fourier basis is an orthonormal basis.*

Proof. With the notation above, we will consider $M = 1$. Let $0 \leq k, l \leq N-1$. We want to prove that

$$\langle F_k, F_l \rangle = \delta_{kl}.$$

Suppose first that $k = l$. Then, trivially:

$$\langle F_k, F_l \rangle = \frac{1}{N} \sum_{j=0}^{N-1} e^{2\pi i j(k-l)/N} = \frac{1}{N} \cdot N = 1.$$

On the other hand, if $k \neq l$, the expression above turns to be a geometric sum:

$$\langle F_k, F_l \rangle = \frac{1}{N} \sum_{j=0}^{N-1} \left(e^{2\pi i (k-l)/N}\right)^j = \frac{1}{N} \cdot \frac{1 - e^{2\pi i (k-l)}}{1 - e^{2\pi i (k-l)/N}} = 0.$$

□

Definition 2.1.5. Let $A \in \ell^2(\mathbb{Z}_N \times \mathbb{Z}_M)$. We define the Discrete Fourier Transform (DFT) as follows:

$$\hat{A}_{n,m} = \sum_{j=0}^{N-1} \sum_{k=0}^{M-1} A_{j,k} e^{-2\pi i \left(\frac{jn}{N} + \frac{km}{M}\right)}. \tag{2.1}$$

Note that we are ignoring the coefficient $1/\sqrt{NM}$ appearing in the Fourier basis vectors. This definition is equivalent to say that

$$\hat{A}_{n,m} = \sqrt{N} \langle A, F_{n,m} \rangle. \tag{2.2}$$

Proposition 2.1.6. *1. The DFT is always invertible. Moreover, the Inverse Discrete Fourier Transform (IDFT) is given by*

$$\check{A}_{n,m} = \frac{1}{NM}\sum_{j=0}^{N-1}\sum_{k=0}^{M-1} A_{j,k}e^{2\pi i\left(\frac{jn}{N}+\frac{km}{M}\right)}. \tag{2.3}$$

2. Parseval's identity: for any $z, w \in \ell^2(\mathbb{Z}_N)$,

$$\langle z, w\rangle = \frac{1}{N}\langle \hat{z}, \hat{w}\rangle.$$

3. Plancherel's identity: for $z \in \ell^2(\mathbb{Z}_N)$,

$$\|z\| = \frac{1}{N}\|\hat{z}\|. \tag{2.4}$$

Proof. 1. Let $M = 1$ and $z \in \ell^2(\mathbb{Z}_N)$. Then, using equation (2.2) and the fact that the N Fourier vectors F_m form a basis, we have that

$$z_n = \sum_{k=0}^{N-1}\langle z, F_k\rangle (F_k)_n = \sum_{k=0}^{N-1}\frac{1}{\sqrt{N}}\hat{z}_k\frac{1}{\sqrt{N}}e^{2\pi ink/N} = \frac{1}{N}\sum_{k=0}^{N-1}\hat{z}_k\, e^{2\pi ink/N}.$$

2. For the second part, we use the linearity of the scalar product and equation (2.2):

$$\begin{aligned}\langle z, w\rangle &= \left\langle \sum_{k=0}^{N-1}\langle z, F_k\rangle F_k, w\right\rangle = \sum_{k=0}^{N-1}\langle z, F_k\rangle\langle F_k, w\rangle = \sum_{k=0}^{N-1}\langle z, F_k\rangle\overline{\langle w, F_k\rangle}\\ &= \frac{1}{N}\sum_{k=0}^{N-1}\hat{z}_k\overline{\hat{w}_k} = \frac{1}{N}\langle \hat{z}, \hat{w}\rangle.\end{aligned}$$

3. If we take $w = z$ in 2., we obtain Plancherel's identity.

□

In applied analysis, we will use the 1-dimensional transform for audio signals, where the component z_n is the value of the signal z at time n, and the 2-dimensional transform will be used for images, where the component $A_{n,m}$ will denote the gray-scale color intensity of an image at the pixel (n, m). For simplicity, we will always work the 1-dimensional case for theoretical purposes.

The first observation, is that for a signal $z \in \ell^2(\mathbb{Z}_N)$, the DFT requires N^2 products, but there is an algorithm called Fast Fourier Transform (FFT, Section 2.2) which computes the DFT and requires $N \log N$ products.

Before introducing the concept of convolution of two vectors, it is worth to make an observation which relates the DFT and its inverse: fix $n \in \mathbb{Z}_N$, and let $z \in \ell^2(\mathbb{Z}_N)$. Then, using equations (2.1) and (2.3):

$$\check{z}_{-n} = \frac{1}{N}\sum_{k=0}^{N-1} z_k e^{2\pi i(-nk)/N} = \frac{1}{N}\sum_{k=0}^{N-1} z_k e^{-2\pi ink/N} = \frac{1}{N}\hat{z}_n,$$

and since $\check{z}$ has period N, we can rewrite it as

$$\check{z}_{N-n} = \frac{1}{N}\hat{z}_n.$$

This observation implies that we can also use the FFT algorithm in order to compute the inverse of the DFT.

Definition 2.1.7. We define the convolution of two vectors $A, B \in \ell^2(\mathbb{Z}_N \times \mathbb{Z}_M)$ as

$$(A * B)_{n,m} = \sum_{j=0}^{N-1}\sum_{k=0}^{M-1} A_{j,k}B_{n-j,m-k}.$$

If we want to compute a convolution of two vectors $z, w \in \ell^2(\mathbb{Z}_N)$, we notice that it also requires a high number of multiplications (concretely N^2). As we are going to see, we can use the FFT algorithm in order to compute convolutions quickly:

Lemma 2.1.8. *For every $n \in \mathbb{Z}_N$:*

$$\widehat{(z * w)}_n = \hat{z}_n\hat{w}_n.$$

Proof. Using the definition of the DFT:

$$\begin{aligned}\widehat{(z * w)}_n &= \sum_{m=0}^{N-1}(z * w)_m\, e^{-2\pi inm/N} = \sum_{m=0}^{N-1}\sum_{k=0}^{N-1} z_{m-k}w_k e^{-2\pi in(m-k)/N}e^{-2\pi ink/N}\\ &= \sum_{k=0}^{N-1} w_k e^{-2\pi ink/N}\sum_{m=0}^{N-1} z_{m-k}e^{-2\pi in(m-k)/N}\\ &= \sum_{k=0}^{N-1} w_k e^{-2\pi ink/N}\sum_{j=-k}^{N-k-1} z_j\, e^{-2\pi inj/N}.\end{aligned}$$

In the last equation we made a change of index, namely $j = m - k$. It would only remain to prove that for any vector $x \in \ell^2(\mathbb{Z}_N)$, it holds that

$$\sum_{n=k}^{N+k-1} x_n = \sum_{n=0}^{N-1} x_n, \tag{2.5}$$

which is trivial. □

Hence, every time we need to compute a transformation which is given in terms of convolutions, we can use the DFT.

2.2 The Fast Fourier Transform

In Section 2.1 we noticed that computing the DFT of a vector $z \in \ell^2(\mathbb{Z}_N)$ requires N^2 complex products, and hence, four real products, which can be lowered to three by just adding sums. Indeed, consider two complex numbers $\omega_1 = a+bi$, $\omega_2 = c+di$. Then,

$$\begin{aligned}\omega_1\omega_2 &= ac - bd + i(ad + bc) = ac - bd + bc - bc + i(ad + bc + bd - bd)\\ &= c(a + b) - b(c + d) + i(b(c + d) + d(a - b)),\end{aligned}$$

so the multiplications needed are just $c(a + b)$, $b(c + d)$, $d(a - b)$. Nevertheless, we will reduce this number drastically if we use the FFT, which will be developed throughout this section. First of all, we will consider the simplest case when N is even, which will illustrate the basic idea behind the FFT.

Definition 2.2.1. Let $M \in \mathbb{N}$ and $N = 2M$. If $z \in \ell^2(\mathbb{Z}_N)$, define

$$\begin{aligned} u_k &= z_{2k}, && \text{for } k = 0, 1, \dots, M - 1,\\ v_k &= z_{2k+1}, && \text{for } k = 0, 1, \dots, M - 1.\end{aligned}$$

Moreover, we will denote $\hat{z}$ the DFT of z in $\ell^2(\mathbb{Z}_N)$, and $\hat{u}, \hat{v}$ the DFTs of u, v respectively, in $\ell^2(\mathbb{Z}_M)$.

Lemma 2.2.2. *In the context of Definition 2.2.1, for $m = 0, 1, \dots, M - 1$ it holds that*

$$\hat{z}_m = \hat{u}_m + e^{-2\pi im/N}\hat{v}_m.$$

Also, for $m = M, M+1, \dots, N-1$, let $l = m - M$ (and hence, $l = 0, 1, \dots, M-1$). Then:

$$\hat{z}_m = \hat{z}_{l+M} = \hat{u}_l - e^{-2\pi il/N}\hat{v}_l.$$

Proof. For any $m = 0, 1, \dots, N - 1$, by definition:

$$\begin{aligned}\hat{z}_m &= \sum_{n=0}^{N-1} z_n e^{-2\pi imn/N} = \sum_{k=0}^{M-1} z_{2k} e^{-2\pi i2km/N} + \sum_{k=0}^{M-1} z_{2k+1} e^{-2\pi i(2k+1)m/NM}\\ &= \sum_{k=0}^{M-1} u_k e^{-2\pi ikm/(N/2)} + e^{-2\pi im/N} \sum_{k=0}^{M-1} v_k e^{-2\pi ikm/(N/2)}\\ &= \sum_{k=0}^{M-1} u_k e^{-2\pi ikm/M} + e^{-2\pi im/N} \sum_{k=0}^{M-1} v_k e^{-2\pi ikm/M}.\end{aligned}$$

Observe that for $m = 0, 1, \dots, M - 1$, the last expression yields $\hat{z}_m = \hat{u}_m + e^{-2\pi im/N}\hat{v}_m$. On the other hand, if $m = M, M + 1, \dots, N - 1$, writing $m = l + M$:

$$\begin{aligned}\hat{z}_m &= \sum_{k=0}^{M-1} u_k e^{-2\pi ik(l+M)/M} + e^{-2\pi i(l+M)/N} \sum_{k=0}^{M-1} v_k e^{-2\pi ik(l+M)/M}\\ &= \sum_{k=0}^{M-1} u_k e^{-2\pi ikl/M} - e^{-2\pi il/N} \sum_{k=0}^{M-1} v_k e^{-2\pi ikl/M} = \hat{u}_l - e^{-2\pi il/N}\hat{v}_l.\end{aligned}$$

□

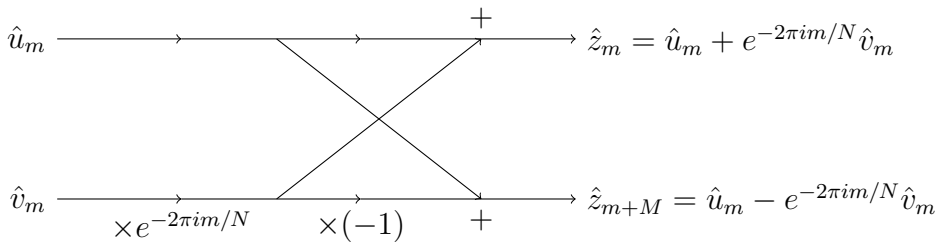

Figure 2.1: Butterfly diagram.

The main point of this procedure is to notice that for $m = 0, 1, \ldots, M-1$, the values z_m, z_{M+m} are computed through the values $\hat{u}_m, \hat{v}_m$, as it is shown in the diagram of Figure 2.1, which receives the name of "butterfly". Whenever we want to compute $\hat{z}$, we first compute $\hat{u}$ and $\hat{v}$. Since each one of these vectors has length $M = N/2$, computing their DFTs will require M^2 complex products for each one. Next, we need to compute the products

$$e^{-2\pi im/N}\hat{v}_m,$$

with $m = 0, 1, \ldots, M-1$, so it takes M additional complex multiplications. The rest of operations are just sums and subtractions, which are computed way faster than products. The total number of complex products needed are

$$2M^2 + M = 2\left(\frac{N}{2}\right)^2 + \frac{N}{2} = \frac{1}{2}(N^2 + N),$$

which is essentially $N^2/2$ when N is large. In conclusion, Lemma 2.2.2 reduces the computation time for $\hat{z}$ to almost half. However, if N is divisible by 4, we can make another step: in this case, M is even, and therefore, the computation of $\hat{u}, \hat{v}$ can be simplified using the same method, and so on, as long as $N/2^k$ is even. This leads to the next definition:

Definition 2.2.3. Let $N \geq 1$, and $z \in \ell^2(\mathbb{Z}_N)$. Define $\#_N$ as the least number of complex multiplications required to compute $\hat{z}$.

The first observation is that if N is even, Lemma 2.2.2 applies, and we obtain

$$\#_N \leq 2\#_M + M.$$

Furthermore, if N is a power of 2, we have the following result.

Proposition 2.2.4. *Let $N = 2^n$, for $n \in \mathbb{N}$. Then*

$$\#_N \leq \frac{1}{2} N \log_2 N. \tag{2.6}$$

Proof. The proof is by induction on n. Let $n = 1$. In this case, $z \in \ell^2(\mathbb{Z}_2)$ has only two components, namely $z = (a, b)$. By definition (see (2.1)),

$$\hat{z} = (a + b, a - b),$$

and we do not need any complex multiplication. Hence $0 < (2 \log_2 2)/2 = 1$, and the result holds. By induction, suppose that it is true for $n = k - 1$. Then, equation (2.6) and the induction hypothesis yield

$$\#_{2^k} \leq \#_{2^{k-1}} + 2^{k-1} \leq 2 \cdot \frac{1}{2} \cdot 2^{k-1}(k-1) + 2^{k-1} = k2^{k-1} = \frac{1}{2}k2^k = \frac{1}{2}N \log_2 N.$$

□

The first upcoming question after this result is what happens if N is not even. For N prime, we cannot apply the method of the FFT, but if $N = pq$, with $p, q \geq 2$, there is another way to describe this method (very similar to the first one), which yields the inequality

$$\#_{pq} = p\#_q + q\#_p + pq.$$

The construction of this algorithm with full detail can be found in [7], Lemma 2.40. However, we can restrict ourselves to the case of $N = 2^n$ whenever we need to do any computation. Indeed, if a vector z has length $N \neq 2^k$, for all k, we can just pad z with zeros to the closest power of 2 that is higher than N, and proceed as done before. Finally, note that we have reduced the number of computations to compute the FFT drastically (for $N = 2^k$): computing $\hat{z}$ directly takes $(N^2 + N)/2$ complex products, while using the FFT, just $(N \log_2 N)/2 = Nk/2$. The quadratic term disappears, yielding a large reduction of computation time. As an example, consider $N = 2^{15} = 32\,768$. The ratio of both computation times will be

$$\frac{2N^2}{N \log_2 N} = \frac{2^{31}}{15 \cdot 2^{15}} = \frac{2^{16}}{15} \approx 4\,369.$$

This ratio precisely tells us that we can compute in around 4 369 seconds (more than one hour, which has 3 600 seconds) the same operations that the FFT does in 1 second.

Summarizing, the key point to construct the FFT is to take profit of Lemma 2.2.2, which basically allow us to reduce the number of complex multiplications to half. Moreover, we can iterate this process as long as the dimension is even at each step; that is why computers actually fill the input vector with zeros, until the dimension is a power of 2.

2.3 Localized basis of vectors

Although the discrete Fourier transform is very useful to work with compactly supported signals, there are some applications in signal analysis for which the DFT fails, and this follows from the fact that the Fourier basis is not localized in space.

Definition 2.3.1. A vector $v \in \ell^2(\mathbb{Z}_N)$ is localized in space near n_0 if z_n is zero or relatively small (with respect to its highest coefficient) when n is far from n_0.

For example, the standard Euclidean basis $\{e_m\}_{m=0}^{N-1}$ is made of vectors which are localized, since every e_m has only one nonzero component. In contrast, we have the Fourier basis $\{F_m\}_{m=0}^{N-1}$, whose elements are $(F_m)_n = \frac{1}{\sqrt{N}} e^{2\pi imn/N}$, so that each component of F_m has the same magnitude, namely $1/\sqrt{N}$. This is completely the opposite concept of localized vector.

Consider a basis of $\ell^2(\mathbb{Z}_N)$, $\mathcal{B} = \{v_0, \ldots, v_{N-1}\}$, whose vectors are localized in space. If $z \in \ell^2(\mathbb{Z}_N)$, we write it as

$$z = \sum_{k=0}^{N-1} z_k v_k. \tag{2.7}$$

Now, if we want to focus in some portion of z near a fixed n_0, the localization in space of the vectors v_k will allow us to keep a reduced number of terms in (2.7). This fact plays an important role in the data compression. For instance, we can think of television video images. Similarly as in [7, p. 166], suppose that there is a static background with a bird flying in the horizon, so that every frame will be very close to the previous one (in terms of data). If we had a basis of localized vectors, in order to transmit the information we would only need to update a few coefficients, since those corresponding to the pixels which are far from the flying bird would be almost unaffected by the movement. This does not happen when $\mathcal{B}$ is the Fourier basis, since a single little change on any z_k may affect all the coefficients $\hat{z}_k$ significantly, in the sense of how are they distributed; the total norm will not vary so much (by Plancherel's identity (2.4)). An example with numbers is the following: if we want to reproduce a video with 60 FPS[1] and a resolution of 1920×1080, then we need to update the total number of

$$60 \times 1\,920 \times 1\,080 = 124\,416\,000$$

pixels every second, which is too much to do it without any optimization.

Example 2.3.2. This example shows the lack of localization of the Fourier basis: We will consider two 3×3 matrices A and B which will differ only on one coefficient by one unit, and check what happens with their respective DFTs. Consider the matrices $\hat{A}^*$ and $\hat{B}^*$ to contain the absolute values of the DFTs of A and B. Then, without normalization:

[1] Frames Per Second: The amount of images shown by a video device in one second. The higher FPS, the higher feeling of continuity we have.

$$A = \begin{pmatrix} 6 & 7 & 8 \\ 9 & 10 & 9 \\ 8 & 7 & 6 \end{pmatrix}, \qquad \hat{A}^* = \begin{pmatrix} 70 & 1 & 1 \\ 7 & 4 & 2 \\ 7 & 2 & 4 \end{pmatrix}$$

$$B = \begin{pmatrix} 6 & 7 & 8 \\ 9 & 9 & 9 \\ 8 & 7 & 6 \end{pmatrix}, \qquad \hat{B}^* = \begin{pmatrix} 69 & 0 & 0 \\ 6 & 3 & 3 \\ 6 & 3 & 3 \end{pmatrix}$$

As we can appreciate, the Fourier basis is not localized at all; one small change in the original matrix produced a large change on its DFT. This was a simple low-dimensional case, but the same happens for arbitrary $M \times N$ matrices.

Definition 2.3.3. We say that a vector basis $\{z_n\}_{n=0}^{N-1} \subset \ell^2(\mathbb{Z}_N)$ is frequency localized if the vectors $\hat{z}_m$ are localized in space near some n_m, for $m = 0, \ldots, N-1$.

We will see later that we are interested in finding basis of vectors which are both localized in space and frequency localized, which yields good properties (in fact, wavelets will provide spatial and frequency localization). However, it is not possible to have a basis of vectors which are completely localized and frequency localized, i.e., z and $\hat{z}$ cannot be highly concentrated, roughly speaking. This is motivated by the "uncertainty principle".

2.4 The Uncertainty Principle

This section is devoted to study how localized can a vector $x \in \ell^2(\mathbb{Z}_N)$ and its Fourier transform be at the same time. The results we are going to obtain are somehow similar to the other well-known uncertainty principles, such as Heisenberg's (which can be found in [14, p. 74]), or the continuous version of the uncertainty principle [8, Theorem 7.30], which gives a lower bound for the product of the measure of the supports of a function and its Fourier transform. Also, in [8, Section 7.8], we can find more inequalities related to the uncertainty principle, concretely, weighted inequalities. In [12, Proposition 2.4.12], we can find another uncertainty principle inequality in the unit disk which is not analogous to Heisenberg's.

For any $z \in \ell^2(\mathbb{Z}_N)$, define $H(z)$ and $H(\hat{z})$ as the number of nonzero elements of z and $\hat{z}$ respectively (this is called the hamming weight of a vector). We have the following result:

Theorem 2.4.1.

$$H(z) \cdot H(\hat{z}) \geq N. \tag{2.8}$$

Before we start with the proof of this theorem, we will need an auxiliary lemma:

Lemma 2.4.2. *Suppose that $z \in \ell^2(\mathbb{Z}_N)$ has $H(z)$ nonzero elements. Then $\hat{z}$ cannot have $H(z)$ consecutive zeros.*

Proof. Let $n_1, \ldots, n_{H(z)}$ denote the positions where z is different from zero and write $b_j = z_{n_j}$, where $j = 1, \ldots, H(z)$. For any $m \in \mathbb{Z}_N$, we will prove that there must be some nonzero element of $\hat{z}$ in the frequency interval $\{m+1, \ldots, m+H(z)\}$. Define $v_j = e^{-2\pi i n_j/N}$ and

$$y_k = \sum_{j=1}^{H(z)} b_j v_j^{m+k}, \text{ for } k = 1, \ldots, H(z),$$

and note that $\hat{z}_{m+k} = y_k$. Now we prove that $y_k \neq 0$, for some $k \in \{1, \ldots, H(z)\}$. Re-writing the expression of the terms y_k with matrix notation, we have:

$$y = Vb, \qquad V_{k,j} = v_j^{m+k}.$$

The conclusion of the lemma tells us that $y \neq 0$ and by construction, $b \neq 0$. Therefore, the lemma will be true if the system

$$0 = Vb$$

has no solution $b \neq 0$, i.e., V is nonsingular. Multiplying each column j of the matrix V by v_j^m, we obtain a re-scaled matrix V' which is nonsingular if and only if so is V (since $v_j \neq 0$ for all j). But

$$V' = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ v_1 & v_2 & \cdots & \cdots & v_{H(z)} \\ v_1^2 & v_2^2 & \cdots & \cdots & v_{H(z)}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_1^{H(z)-1} & \cdots & \cdots & \cdots & v_{H(z)}^{H(z)-1} \end{pmatrix}$$

is the $H(z) \times H(z)$ Vandermonde matrix, which is known to be nonsingular. A proof of this fact can be found in [9, Proposition 3.19]. □

Proof of Theorem 2.4.1. Suppose first that $H(z)$ divides N. Then, we take a partition of $\{0, \ldots, N-1\}$ into $N/H(z)$ intervals of length $H(z)$. By Lemma 2.4.2, $\hat{z}$ cannot vanish entirely in any interval, so there must be a nonzero element in each one of the intervals. Hence, $H(\hat{z}) \geq N/H(z)$. If $H(z)$ does not divide N, let $k = \lceil N/H(z) \rceil$. We cannot spread out fewer than k elements between N places without leaving spaces bigger than $H(z)$. Hence, again by the lemma, $H(\hat{z}) \geq k$, and therefore $H(\hat{z})H(z) > N$. □

Now we may ask ourselves when is (2.8) an equality. For the trivial case where $z = e_0$ is the first vector of the canonical basis of $\ell^2(\mathbb{Z}_N)$, it holds that $H(z) = 1$ and $H(\hat{z}) = N$. More generally, for any factorization $N = k \cdot l$, consider the sequence

$$x_j^k = \begin{cases} 1, & \text{if } j = k \cdot m,\ m = 0, \ldots k-1, \\ 0, & \text{otherwise.} \end{cases}$$

We have the following result:

Theorem 2.4.3. *It holds that $H(z)H(\hat{z}) = N$ only for the sequence $z = x^{H(z)}$ and sequences reducible to it by the following operations:*

1. *Scalar multiplication.*
2. *Cyclic permutation in the time domain.*
3. *Cyclic permutation in the frequency domain.*

Summarizing, equality $H(z)H(\hat{z}) = N$ holds if and only if

$$\hat{z}_{j-J} = \alpha\, x^{H(z)}_{k-K},$$

for some $\alpha \neq 0$ and some integers J, K.

For the proof of this theorem, we refer to [6, Appendix A]. Finally, we also have a lower bound for the total number of nonzero elements of $z, \hat{z} \in \ell^2(\mathbb{Z}_N)$:

Corollary 2.4.4. $H(z) + H(\hat{z}) \geq 2\sqrt{N}$.

Proof. Apply the arithmetic-geometric mean inequality with $H(z)$, $H(\hat{z})$ and combine it with (2.8). □

Motivated by this result, we are interested in computing the maximum number of zero components that both a vector $x \in \ell^2(\mathbb{Z}_N)$ and its DFT can have at the same time, depending on the dimension N.

Definition 2.4.5. For every $N \geq 1$, define

$$\begin{aligned} \ell^2_*(\mathbb{Z}_N) &:= \ell^2(\mathbb{Z}_N)\backslash\{0\}, \\ Z(x) &:= |\{j : x_j = 0\}|, \\ L(N) &:= \max_{x \in \ell^2_*(\mathbb{Z}_N)} \min\{Z(x), Z(\hat{x})\}. \end{aligned}$$

In order for this definition to make sense, we must exclude the zero vector from the definition of $L(N)$. Otherwise, $L(N) = N$ for all N.

Proposition 2.4.6. *For all $N \geq 1$,*

$$L(N) \leq N - \sqrt{N}.$$

Proof. First of all, note that for all $x \in \ell^2(\mathbb{Z}_N)$, it holds that $Z(x) = N - N(x)$. Using Corollary 2.4.4, we obtain that

$$N - \sqrt{N} \geq \frac{Z(x) + Z(\hat{x})}{2} \geq \min\{Z(x), Z(\hat{x})\}.$$

□

Theorem 2.4.7. *1. Let $N \geq 3$ be nonprime and suppose that $N = m \cdot n$, with $m \geq n$. Then,*

$$L(N) \geq \max\left\{K \,:\, \max\{m,n\}|K\,,\, K \leq N - \sqrt{N}\right\} = m(n-1). \tag{2.9}$$

2. Among all the possible decompositions $N = m \cdot n$, the greatest lower bound of $L(N)$ that can be obtained from equation (2.9) is given when $|m-n|$ is minimized.

Proof. 1. Suppose that $N = m \cdot n$, with m, n as in the hypothesis. Consider the sequence

$$x_j = x_j^n = \begin{cases} 1, & \text{if } j = k \cdot n, \text{ for } k = 0, \ldots, m-1, \\ 0, & \text{otherwise,} \end{cases}$$

and observe that $Z_x = N - m = m(n-1)$. Now let us compute $\hat{x}$. Suppose that $k \in \mathbb{Z}_N$ is such that $k = m \cdot l$, i.e., m divides k. Then:

$$\hat{x}_k = \sum_{j=0}^{N-1} x_j e^{-2\pi i j k/N} = \sum_{s=0}^{m-1} x_{s\cdot n} e^{-2\pi i s n k/(mn)} = \sum_{s=0}^{m-1} e^{-2\pi i s m l/m} = \sum_{s=0}^{m-1} 1 = m.$$

On the other hand, if m does not divide k, then $k = m \cdot q + d$, with $d \neq 0$, and

$$\begin{aligned} \hat{x}_k &= \sum_{j=0}^{N-1} x_j e^{-2\pi i j k/N} = \sum_{s=0}^{m-1} x_{s\cdot n} e^{-2\pi i s n (mq+d)/(nm)} = \sum_{s=0}^{m-1} e^{-2\pi i s n m q/(nm)} e^{-2\pi i s n d/(nm)} \\ &= \sum_{s=0}^{m-1} e^{-2\pi i s d/m} = \frac{1 - e^{-2\pi i d m/m}}{1 - e^{-2\pi i d/m}} = \frac{1-1}{1 - e^{-2\pi i d/m}} = 0. \end{aligned}$$

Observe that in the last expression the denominator can never vanish, since we have that $0 < d \leq m-1$. Also note that

$$\hat{x} = \widehat{x^n} = m \cdot x^m.$$

Therefore, $Z_{\hat{x}} = N - n = n(m-1)$. Now assume without loss of generality that $m \geq n$. Then, it can be easily checked that $m(n-1) \leq n(m-1)$, so we have found a vector $x = x^n \in \ell^2_*(\mathbb{Z}_N)$ such that $\min\{Z_x, Z_{\hat{x}}\} = m(n-1)$. This implies that $L(N) \geq m(n-1)$. The only thing we are left to prove is that

$$m(n-1) = \max\left\{K \,:\, m|K\,,\, K \leq N - \sqrt{N}\right\}.$$

Consider the set A_m of multiples of m that do not exceed N, that is:

$$A_m = \{0, m, 2m, \ldots, m(n-1), mn\},$$

and define $A_m^* = \{a \in A_m | a \leq mn - \sqrt{mn}\}$. Obviously nm does not belong to A_m^*, and on the other hand, if holds that $m(n-1) \leq mn - \sqrt{mn}$, since

$$m(n-1) \leq mn - \sqrt{mn} \Leftrightarrow mn - m \leq mn - \sqrt{mn} \Leftrightarrow \sqrt{m^2} \geq \sqrt{mn},$$

and the last expression is true because we are considering $m \geq n$. We conclude:

$$L(N) \geq \max A_m^* = m(n-1).$$

2. Suppose that $N = m \cdot n = m_0 \cdot n_0$, and without loss of generality, $m \geq n$, $m_0 \geq n_0$. Also, assume that $|m-n| \leq |m_0 - n_0|$. Under these conditions we have that $m_0 \geq m \geq n \geq n_0$. We want to prove that

$$m(n-1) \geq m_0(n_0 - 1).$$

But this happens if and only if $N - m \geq N - m_0$, which is obviously true, since $m_0 \geq m$. □

As an example to illustrate this result, we consider $N = 30 = 6 \cdot 5 = 2 \cdot 15$. Then:

$$Z(x^5) = 30 - 6 = 24,\ Z(\widehat{x^5}) = Z(x^6) = 30 - 5 = 25,$$
$$Z(x^2) = 30 - 15 = 15,\ Z(\widehat{x^2}) = Z(x^{15}) = 30 - 2 = 28.$$

We can observe that $\min\{Z(x^5), Z(x^6)\} = 24$ and $\min\{Z(x^2), Z(x^{15})\} = 15$. Hence, $L(30) \geq 24$. Moreover, by Proposition 2.4.6, it holds that

$$L(30) \leq \left[30 - \sqrt{30}\right] = 24,$$

and the conclusion is that $L(30) = 24$. As we are going to see, there are certain N for which we can determine $L(N)$ so far.

Corollary 2.4.8. *1. If $N = n^2$ for some n, then*

$$L(N) = N - \sqrt{N} = n^2 - n = n(n-1).$$

2. If $N = n(n-1)$ for some n, then

$$L(N) = \left[N - \sqrt{N}\right] = n(n-2).$$

Proof. 1. By Theorem 2.4.7, we have that $L(N) \geq N - \sqrt{N} = n(n-1)$. On the other hand, Proposition 2.4.6 tells us that $L(N) \leq N - \sqrt{N}$.

2. Again, by Theorem 2.4.7, $L(N) \geq n(n-2)$. Moreover, we have the following sequence of equivalences

$$\begin{aligned}
\left[n(n-1) - \sqrt{n(n-1)}\right] &= n(n-2) \\
&\Updownarrow \\
n(n-2) \leq n(n-1) - \sqrt{n(n-1)} &< n(n-2) + 1 \\
&\Updownarrow \\
-n \leq -\sqrt{n(n-1)} &< -n + 1 \\
&\Updownarrow \\
n - 1 < \sqrt{n(n-1)} &\leq n,
\end{aligned}$$

and the last expression is trivially true. Hence, $L(N) = \left[N - \sqrt{N}\right] = n(n-2)$. □

Before proceeding, we are going to improve the lower bound found in Theorem 2.4.7:

Theorem 2.4.9. *Let $N \geq 3$, $N = m \cdot n$ with $m \geq n$. Define the set*

$$A = \{m_0 \cdot n \,|\, 1 \leq m_0 < m,\, m_0 < N - m_0 n\}.$$

Then,

$$L(N) \geq \max\big(A \cup \{m(n-1)\}\big).$$

Proof. We already know from Theorem 2.4.7 that $L(N) \geq m(n-1)$. Thus, it only remains to prove that $L(N) \geq \max A$. Let $x \in \ell^2(\mathbb{Z}_N)$ be of the following form:

$$x = (a_0, \underbrace{0, \dots, 0}_{n-1}, a_1, \underbrace{0, \dots, 0}_{n-1}, a_2, 0, \dots\dots, 0, a_{m_0-1}, 0, \dots, 0), \text{ with } a_j \in \mathbb{C}.$$

To clarify, x only has nonzero components at indexes that are multiple of n (that is, $x_{jn} = a_j$), and obviously, we have the restriction $m_0 \leq m$. Let $\omega = e^{2\pi i/N}$, and consider $1 \leq m_0 < m$. Suppose that $m_0 < N - nm_0 =: k$, and write $c = m_0 - 1$. Then we can build the following system of equations

$$\begin{pmatrix} 0 \\ \vdots \\ \\ \vdots \\ \vdots \\ \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \hat{x}_2 \\ \vdots \\ \hat{x}_c \\ \hat{x}_m \\ \hat{x}_{m+1} \\ \vdots \\ \hat{x}_{m+c} \\ \vdots \\ \hat{x}_{n(m-1)+c} \end{pmatrix} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega^{n} & \cdots & \omega^{n(k-1)} \\ 1 & \omega^{2n} & \cdots & \omega^{2n(k-1)} \\ \vdots & & & \vdots \\ 1 & \omega^{nc} & \cdots & \omega^{cn(k-1)} \\ 1 & \omega^{nm} & \cdots & \omega^{nm(k-1)} \\ 1 & \omega^{n(m+1)} & \cdots & \omega^{n(m+1)(k-1)} \\ \vdots & & & \vdots \\ 1 & \omega^{n(m+c)} & \cdots & \omega^{n(m+c)(k-1)} \\ \vdots & & & \vdots \\ 1 & \omega^{n(m-1)+c} & \cdots & \omega^{n(n(m-1)+c)(k-1)} \end{pmatrix} \begin{pmatrix} x_0 \\ x_n \\ \vdots \\ x_{n(k-1)} \end{pmatrix}. \tag{2.10}$$

Now we are going to prove that this system of equations is compatible indeterminate. If we do that, then we can find a vector $x \in \ell^2(\mathbb{Z}_N)$ such that

$$\begin{aligned} Z(x) &= N - k = N - N + nm_0 = nm_0, \\ Z(\hat{x}) &\geq nm_0, \end{aligned}$$

so that $L(N) \geq nm_0$. We observe that the matrix of the system (2.10), say Ω, contains m identical blocks B_j (due to the exponential periodicity) of size $(c+1) \times k$ (or equivalently, $m_0 \times (N - nm_0)$), each one of them consisting on the rows indexed by the values $kn + j$ when $k \in \{0, 1, \dots m-1\}$ is fixed and $j = 0, 1, \dots c$. Then, the rank of the matrix Ω will be less than or equal to the rank of one B_k, reaching equality if the block has maximum rank, so that rank $\Omega \leq c + 1 = m_0$. Actually,

this rank will be maximum, since each one of the blocks consists on the rows of a Vandermonde matrix: for example, consider the first block B_1. Write $\beta = e^{2\pi i/m}$. Then:

$$B_1 = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega^n & \cdots & \omega^{n(k-1)} \\ 1 & \omega^{2n} & \cdots & \omega^{2n(k-1)} \\ \vdots & & & \vdots \\ 1 & \omega^{cn} & \cdots & \omega^{cn(k-1)} \end{pmatrix} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \beta & \cdots & \beta^{(k-1)} \\ 1 & \beta^2 & \cdots & \beta^{2(k-1)} \\ \vdots & & & \vdots \\ 1 & \beta^c & \cdots & \beta^{c(k-1)} \end{pmatrix}.$$

So, we conclude that rank $\Omega = c + 1 = m_0$. Thus, the system (2.10) will be compatible indeterminate whenever the number of columns of Ω (or the number of indeterminates, which is the same) is strictly greater than its rank, which we supposed with the hypotheses $m_0 < N - nm_0$. However, we do not know if we can find up to $N - nm_0$ multiples of n in the first N natural numbers: we can find them if and only if

$$N - nm_0 \leq m \Leftrightarrow N - m \leq nm_0 \Leftrightarrow m(n-1) \leq nm_0.$$

If this condition fails, then $L(N) \geq m(n-1)$, as we already knew. Otherwise, $L(N) \geq nm_0$. Since this procedure works for all m_0 satisfying the condition $m_0 < N - nm_0$, taking the maximum will lead to the desired result. □

Now we shall determine $L(N)$ for small N. Although there are values of N for which we cannot find yet $L(N)$, there are other methods that will allow us to determine it. For example, constructing one vector $x \in \ell^2(\mathbb{Z}_5)$ such that $Z(x) \geq 2$, $Z(\hat{x}) \geq 2$ would prove that $L(5) = 2$ (moreover, either x or $\hat{x}$ needs to have exactly two zero elements). More generally, if we have an upper bound M_N for $L(N)$ and we can find $x \in \ell^2(\mathbb{Z}_N)$ such that $Z(x) = M_N$ and $Z(\hat{x}) \geq M_N$, then necessarily $L(N) = M_N$. Thus, we are interested in lower bounds for $L(N)$.

The first observation we can do once we know the result yielded by Theorem 2.4.7 is that we are far from determining $L(N)$ when N is prime. Nevertheless, we can use Chebotarev's theorem about roots of unity to determine $L(N)$ (when N is prime), along with basic linear algebra results. The statement is the following:

Theorem 2.4.10 (Chebotarev)**.** *Let Ω be a matrix with entries $a_{jk} = \omega^{jk}$, $0 \leq j, k \leq N-1$, where $\omega = e^{2\pi i/N}$. Then, if N is prime, any minor of Ω is nonzero.*

There are several proofs for this theorem. We can find a proof similar to the original one made by Chebotarev in [13]. Dieudonné also gave an independent proof for this theorem, and it can be found in [5].

Proposition 2.4.11. *Let $N \geq 3$ be a prime number. Then, $L(N) = [N/2]$.*

Proof. First we prove that $L(N) \geq [N/2]$. After that, using Chebotarev's theorem, it is easy to see that $L(N) < [N/2] + 1$. Define $K = [N/2]$, and let

$x = (x_0, x_1, \ldots, x_K, 0, \ldots, 0) \in \ell^2(\mathbb{Z}_N)$. Consider the following system of equations:

$$\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{K-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-2\pi i/N} & \cdots & e^{-2\pi iK/N} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & e^{2\pi i(K-1)/N} & \ldots & e^{2\pi iK(K-1)/N} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_K \end{pmatrix}.$$

Its homogeneous system is

$$\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-2\pi i/N} & \cdots & e^{-2\pi iK/N} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & e^{2\pi i(K-1)/N} & \ldots & e^{2\pi iK(K-1)/N} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_K \end{pmatrix},$$

and it is clearly compatible indeterminate, since the matrix of the system is a Vandermonde matrix. Then, it has an infinite number of solutions, and moreover, we note that each one of the equations of the system is one component of the vector $\hat{x}$. Therefore, we have found that there is a vector $x \in \ell^2(\mathbb{Z}_N)$ with $Z_x = N - K = K + 1$ (since N is odd) and $Z_{\hat{x}} = K$. Now suppose that there exists a vector $x \in \ell^2_*(\mathbb{Z}_N)$ with $Z_x \geq K + 1$, $Z_{\hat{x}} \geq K + 1$. This is equivalent to say that the homogeneous system of equations

$$\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} e^{-2\pi is_1 r_1/N} & e^{-2\pi is_1 r_2/N} & \cdots & e^{-2\pi is_1 r_K/N} \\ e^{-2\pi is_2 r_1/N} & e^{-2\pi is_2 r_2/N} & \cdots & e^{-2\pi is_2 r_K/N} \\ \vdots & \vdots & \vdots & \vdots \\ e^{-2\pi is_{K+1} r_1/N} & e^{-2\pi is_{K+1} r_2/N} & \cdots & e^{-2\pi is_{K+1} r_K/N} \end{pmatrix} \begin{pmatrix} x_{r_1} \\ x_{r_2} \\ \vdots \\ x_{r_K} \end{pmatrix}$$

is compatible indeterminate, and this happens if and only if the matrix of the system has rank strictly less than K. By Theorem 2.4.10, we have that this rank is exactly K. This proves that $L(N) < K + 1$, and hence, $L(N) = K$. □

After this, we can already claim that $L(5) = 2$, $L(7) = 3$, and $L(11) = 5$. Moreover, motivated by the usage of the determinants of the Fourier matrices (i.e., matrices whose columns are the vectors of the Fourier basis), we can create an algorithm which will allow us to determine $L(N)$ exactly (with a high number of computations, though). The algorithm consists of the following:

1. Fix $N \in \mathbb{N}$. Suppose that we want to determine whether $L(N)$ is greater than or equal to a certain natural number $M < N$. If there exists a vector $x \in \ell^2_*(\mathbb{Z}_N)$ such that $Z(x) = M$ and $Z(\hat{x}) \geq M$, then it will be of the form

$$x = \sum_{j=1}^{N-M} x_j e_{k_j},$$

where $\{e_k\}_{k=0}^{N-1}$ is the Euclidean basis for $\ell^2(\mathbb{Z}_N)$, and $x_j \in \mathbb{C}\backslash\{0\}$, for all j, and there will exist $l_1, \ldots, l_M$ such that $\hat{x}_{l_j} = 0$, for $j = 1, \ldots, M$.

2. Writing the expression of $\hat{x}$ in matrix form, we consider the homogeneous system of equations with M equations and $N-M$ unknowns given by the rows $l_1, \ldots, l_M$ of the matrix which describes $\hat{x}$. That is, to consider the system

$$\begin{cases} 0 = & x_1 e^{-2\pi i k_1 l_1/N} + \ldots + x_{N-M} e^{-2\pi i k_{N-M} l_1/N} \\ & \vdots \\ & \vdots \\ 0 = & x_1 e^{-2\pi i k_1 l_M/N} + \ldots + x_{N-M} e^{-2\pi i k_{N-M} l_M/N}. \end{cases}$$

3. Now there are two possibilities: either the system is compatible determinate, or indeterminate. In the first case, we only have the trivial solution $x = 0$, so that a vector $x \in \ell^2(\mathbb{Z}_N)$ with M zeros and such that its DFT has more than M zeros must be the zero vector, and hence $L(N) < M$. In the second case, there are infinite nontrivial solutions for the system, and we can find the explicit expression of a vector x with $Z(x) = M$ and $Z(\hat{x}) \geq M$, which implies that $L(N) \geq M$.

4. If we can find any set of indexes $k_1, \ldots, k_{N-M}$, $l_1, \ldots, l_M$ such that the rank of A is strictly lower than the number of unknowns, (i.e, rank $A < N - M$), then there will be a compatible indeterminate system, whose solution will be a vector x with $Z(x) = M$, $Z(\hat{x}) \geq M$.

The main disadvantage of this algorithm comes when we want to check if we can get an indeterminate system or not. Let us call A to the matrix of the system of equations constructed in step 2. If we can find an indeterminate system, we can stop the algorithm and give the explicit expression of the vector x. On the other hand, if for every set of indexes $k_1, \ldots, k_{N-M}$, $l_1, \ldots, l_M$ the built system turns out to be determinate, we cannot stop the algorithm, and therefore we will need to compute the rank of A for every possible choice of indexes. Without loss of generality, we can always fix $k_1 = 0$. Then, the number of ranks that we will need to compute is

$$\binom{N-1}{N-M-1} \cdot \binom{N}{M},$$

which has factorial order. Moreover, as N and M increase, the matrix A also gets bigger, which makes the computation of its rank even slower. Adding the fact that we have to compute much more ranks, this algorithm will not be any good whenever N is not small. For example, we can apply this algorithm to prove that $L(8) = 4$ and $L(10) = 6$. For $N = 8$, we will consider $M_8 = 5$, and check that every system we can construct is determinate (this example is given in the Annex), but we cannot go much further. The number of required computations will be, in the last case:

$$\binom{7}{2} \cdot \binom{8}{5} = 1176.$$

If we compare the elapsed time during the computations for every one of those different values of N, we are going to see the its fast growth as N increases. If we

try to apply this algorithm for $N = 11$ (even though we already know the result in this case), we will find that it takes more than 4 seconds to be completed, while for $N = 8$, it does not take more than 0.1 seconds.

Although, if we define I to be the set of indexes where $\hat{x}$ is nonzero and J the set of indexes where x is nonzero, the idea of this algorithm is to find certain submatrices of the Fourier matrix $F := F_N$ of size $N \times N$, such that

$$F(N\backslash I, J)x_{|J} = 0 \tag{2.11}$$

and $F(N\backslash I, J)$ is rank-deficient (here we denoted $N = \{0, \dots, N-1\}$ and $x_{|J}$ is the vector x restricted to the set of indexes J, and therefore it has size $|J|$). Note that equation (2.11) describes a choice of a system of equations obtained from a submatrix of the Fourier matrix F, and if $F(N\backslash I, J)$ is rank deficient, we are in the situation of a compatible indeterminate system.

Definition 2.4.12. For a matrix $A \in \mathbb{C}^{m\times n}$, we say that A is rank-deficient if $\operatorname{rank} A < n$.

The papers [3] and [4] provide nice descriptions of maximal rank-deficient submatrices of Fourier matrices, in terms of the size of J. We will see that those descriptions will allow us to compute the value of $L(N)$ much easier than before, for any value of N.

Definition 2.4.13. ([3, Def. 2]) For a matrix $A \in \mathbb{C}^{N\times N}$ and an integer $d \in \{1, \dots, N\}$, we define the Hamming number $H_A(d)$ as the minimal cardinality of all index sets I for which $A(N\backslash I, J)$ is rank-deficient, for a suitable J with $|J| \leq d$. Here we denote $N = \{0, \dots, N-1\}$.

The first thing to observe is that we defined the Hamming numbers with respect to the complement of the set I. In this way we stay close to the formulation of the uncertainty principle. Indeed, joining Theorem 2.4.1 with Definition 2.4.13, we can re-write the uncertainty principle as

$$d \cdot H_{F_N}(d) \geq N.$$

Moreover, (as done in [3, Remark 4]) in Definition 2.4.13 we considered the sets J to be such that $|J| \leq d$, rather than $|J| = d$. We took the inequality because it can happen that the rank-deficiency of $A(N\backslash I, J)$ is caused by the rank-deficiency of $A(N\backslash I, \tilde{J})$ for some strict subset $\tilde{J} \subset J$. This guarantees that $H_A(d)$ is decreasing on d.

The key observation that will allow to compute the exact value of $L(N)$ is the following equality, which follows from the definitions:

$$\begin{aligned} N - H_{F_N}(d) &= \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), H(x) \leq d\} \\ &= \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), Z(x) \geq N - d\}. \end{aligned} \tag{2.12}$$

Then, we have the following result:

Proposition 2.4.14. *Let $N \in \mathbb{N}$ and $1 \leq k < N$. Then, $L(N) = k$ if and only if*

$$N - H_{F_N}(N-k) \geq k, \text{ and} \tag{2.13}$$
$$N - H_{F_N}(N-(k+1)) \leq k. \tag{2.14}$$

Before we prove this result, we give an interpretation of equations (2.13) and (2.14). The first one means that we can find a vector $z \in \ell^2(\mathbb{Z}_N)$ with at least k zero components such that $\hat{x}$ also has more than k zero components. On the other hand, the second inequality tells us that we can find no vector $z \in \ell^2(\mathbb{Z}_N)$ such that both z and $\hat{z}$ have more than k zero components, or equivalently, at least $k+1$ zero elements.

Proof. ($\Rightarrow$) Suppose that $L(N) = k$. By relation (2.12), we have that

$$\begin{aligned} N - H_{F_N}(N-k) &= \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), Z(x) \geq k\} \\ &\geq \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), Z(x) = k\} \geq k, \end{aligned}$$

where the last inequality is obtained by assumption. This proves (2.13). In order to prove (2.14), note that since we are assuming that $L(N) = k$, if $x \in \ell^2(\mathbb{Z}_N)$ is such that $Z(x) = k+1$, then necessarily $Z(\hat{x}) \leq k$. Joining this fact along with relation (2.12), it yields

$$N - H_{F_N}(N-(k+1)) = \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), Z(x) \geq k+1\} \leq k.$$

($\Leftarrow$) We prove this implication by contradiction. Suppose that $L(N) \neq k$. Then, either (i) $L(N) < k$, or (ii) $L(N) > k$. In the case of (i), for any vector $x \in \ell^2(\mathbb{Z}_N)$ such that $Z(x) \geq k$, then, necessarily $Z(\hat{x}) < k$ (otherwise, it would not be true that $L(N) < k$). Then, by relation (2.12):

$$N - H_{F_N}(N-k) = \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), Z(x) \geq k\} < k,$$

i.e., inequality (2.13) is false. Finally, if (ii) holds true, then there exists $N > \tilde{k} > k$ such that $L(N) = \tilde{k}$. Observe that since $H_{F_N}(d)$ is decreasing on the variable d, it follows that the expression

$$N - H_{F_N}(N-M)$$

is decreasing on the variable M. Now, since $k+1 \leq \tilde{k}$:

$$\begin{aligned} N - H_{F_N}(N-(k+1)) &\geq N - H_{F_N}(N-\tilde{k}) \\ &= \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), Z(x) \geq \tilde{k}\} \\ &\geq \max\{Z(\hat{x}) : x \in \ell^2(\mathbb{Z}_N), Z(x) = \tilde{k}\} \geq \tilde{k} > k, \end{aligned}$$

so that inequality (2.14) is false. □

After this, the only remaining thing to do is to conjecture the possible value for $L(N)$, say k, and check if inequalities (2.13) and (2.14) hold for k. Now we list the main results about Hamming numbers. The proofs can be found in each respective reference.

Theorem 2.4.15. *[4, Theorem 9] Let p^m be a power of a prime number. Let $d \in \{1, 2, \dots, p^m\}$ be such that*

$$cp^k \leq d < (c+1)p^k$$

for certain $c \in \{1, \dots, p-1\}, k \in \{0, \dots, m-1\}$. Then,

$$H_{F_{p^m}}(d) = (p - c + 1)p^{m-k-1}.$$

Theorem 2.4.16. *[3, Corollary 23] For each divisor d of N, we have that $H_{F_N}(d) = N/d$, i.e., equality in the uncertainty principle is reached.*

Theorem 2.4.17. *[3, equation (4)] Let $1 \leq t < N$. Then,*

$$H_{F_N}(t) = \min\left\{(p - c + 1)\frac{N}{pd} : pd \text{ divides } n,\ p \text{ prime}, c \in \{1, \dots, p\},\ cd \leq t\right\}.$$

Now we are going to see a few examples of the computation of $L(N)$. First of all, let us consider the case when $N = n^2$, with $n > 1$. We already know that $L(N) = n(n-1)$, but we are going to prove it using Proposition 2.4.14. First of all, let us check that (2.13) holds whenever $k = n(n-1)$. We have that the inequality

$$n^2 - H_{F_N}(N - n(n-1)) \geq n(n-1)$$

is true if and only if $H_{F_N}(N - n(n-1)) \geq n$. But $N - n(n-1) = n^2 - n(n-1) = n$. By Theorem 2.4.16,

$$H_{F_N}(n) = n^2/n = n \geq n,$$

so that the first inequality holds. On the other hand, we do not need to prove inequality (2.14). Indeed, if it was false, it is easy to see that there would be contradiction with the uncertainty principle. So, we conclude

$$L(n^2) = n(n-1).$$

Example 2.4.18. Consider $N = 2^{2n+1}$, with $n \in \mathbb{N}$. We claim that $L(N) = 2^{n+1}(2^n - 1) = 2^{2n+1} - 2^{n+1}$. Note that we are only considering odd powers of 2, since any even power is covered by the case $N = k^2$. First we prove the inequality (2.13):

$$N - H_{F_N}(N - (2^{2n+1} - 2^{n+1})) = 2^{2n+1} - H_{F_N}(2^{n+1}).$$

By Theorem 2.4.16, we have that $H_{F_N}(2^{n+1}) = 2^n$. Therefore,

$$2^{2n+1} - 2^n \geq 2^{2n+1} - 2^{n+1}.$$

Now we prove inequality (2.14). Note that $N - (2^{2n+1} - 2^{n+1} + 1) = 2^{n+1} - 1$. Using Theorem 2.4.15 to compute $H_{F_N}(2^{n+1} - 1)$, we observe that $k = n$ and $c = 1$. Hence,

$$H_{F_N}(2^{n+1} - 1) = (2 - 1 + 1) \cdot 2^{2n+1-n-1} = 2 \cdot 2^n = 2^{n+1}.$$

Finally, since

$$N - 2^{n+1} = 2^{2n+1} - 2^{n+1} \leq 2^{2n+1} - 2^{n+1},$$

we conclude that $L(2^{2n+1}) = 2^{2n+1} - 2^{n+1}$. Moreover, observe that if we consider $2^{2n+1} = m_1 \cdot m_2$ with $m_1 \geq m_2$ and such that the quantity $m_1 - m_2$ is minimized, then $m_1 = 2^{n+1}$ and $m_2 = 2^n$. In this case:

$$L(2^{2n+1}) = 2^{2n+1} - 2^{n+1} = 2^{n+1}(2^n - 1) = m_1(m_2 - 1),$$

which is the lower bound of $L(2^{2n+1})$ given in Theorem 2.4.7.

Finally, we show the table with the values of $L(N)$ for the 99 first natural numbers:

	0	1	2	3	4	5	6	7	8	9
0		0	0	1	2	2	3	3	4	6
10	6	5	8	6	8	10	12	8	12	9
20	15	15	14	11	18	20	16	21	21	14
30	24	15	24	24	22	28	30	18	24	28
40	32	20	35	21	34	36	30	23	40	42
50	40	37	40	26	45	45	48	42	38	29
60	50	30	40	54	56	53	55	33	53	51
70	60	35	63	36	48	65	60	66	66	39
80	70	72	54	41	72	70	56	64	77	44
90	80	78	72	69	62	78	84	48	84	88

We observe that for the first 10 natural numbers, $L(N)$ seems to be monotone, but $L(11) = 5 < L(10)$. This is because, as we have seen, $L(N)$ does not depend on how big N is, but on the decomposition $N = n \cdot m$. Indeed, the key point is that if we have such decomposition, we can find a vector z with n nonzero elements such that $\hat{z}$ has m nonzero elements. For instance, since we can decompose 10 as $2 \cdot 5$, while 11 is a prime number, we can expect a different behavior for $L(10)$ and $L(11)$. Also, note that if we compute the lower bound given in Theorem 2.4.9 for all these numbers, we find that it is close to get the ultimate result; the numbers for which the lower bound given is not equal to $L(N)$ are the following:

$$27,\ 39,\ 44,\ 51,\ 65,\ 68,\ 75,\ 87,\ 95.$$

Chapter 3

Wavelets on $\mathbb{Z}_N$

3.1 Construction of a first-stage wavelet

Before introducing the definition of a wavelet, we are going give some auxiliary notations and their main properties. This will allow us to characterize the wavelets in an easy way.

Definition 3.1.1. For every $z \in \ell^2(\mathbb{Z}_N)$, define its conjugate reflection $\tilde{z} \in \ell^2(\mathbb{Z}_N)$ by

$$\tilde{z}_n = \overline{z_{-n}} = \overline{z_{N-n}}, \qquad \text{for all } n.$$

Proposition 3.1.2. *For any $z \in \ell^2(\mathbb{Z}_N)$, $\widehat{(\tilde{z})}_n = \overline{\hat{z}_n}$.*

Proof. By definition:

$$\widehat{(\tilde{z})}_n = \sum_{j=0}^{N-1} \tilde{z}_j e^{-2\pi i n j/N} = \sum_{j=0}^{N-1} \overline{z_{N-j}} e^{-2\pi i n j/N} = \sum_{j=0}^{N-1} \overline{z_{-j}} \overline{e^{-2\pi i n(-j)/N}} = \overline{\hat{z}_n},$$

where in the last step we use equation (2.5). □

Definition 3.1.3. For any $z \in \ell^2(\mathbb{Z}_N)$, define the circular translation of z as

$$(R_k z)_n := z_{n-k}.$$

Lemma 3.1.4. *Let $z, w \in \ell^2(\mathbb{Z}_N)$. Then, for every $k \in \mathbb{Z}$:*

$$(z * \tilde{w})_k = \langle z, R_k w \rangle,$$
$$(z * w)_k = \langle z, R_k \tilde{w} \rangle.$$

Proof. By definition of scalar product:

$$\langle z, R_k w \rangle = \sum_{n=0}^{N-1} z_n \overline{R_k w_n} = \sum_{k=0}^{N-1} z_n \overline{w_{n-k}} = \sum_{n=0}^{N-1} z_n \tilde{w}_{k-n} = (\tilde{w} * z)_k = (z * \tilde{w})_k.$$

In the last step we used the commutativity of the convolution. To prove the second equality, we only need to replace w by $\tilde{w}$ in the first one, and the second will follow, since $\tilde{\tilde{w}} = w$. □

Now observe that if $w \in \ell^2(\mathbb{Z}_N)$ is such that $\mathcal{B} = \{R_k w\}_{n=0}^{N-1}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$, the coefficients of the expansion of a vector z in terms of $\mathcal{B}$ are the inner products $\langle z, R_k w\rangle$. Then, the previous lemma tells us that

$$[z]_{\mathcal{B}} = z * \tilde{w}.$$

As every convolution can be computed through the DFT (this will allow us to compute it quickly with the FFT), then, for every orthonormal basis $\mathcal{B}$ generated by translations of a vector w, the change of basis matrix from the Euclidean basis to $\mathcal{B}$ can be computed rapidly.

The first example of a basis of the form $\{R_k w\}_{k=0}^{N-1}$ is the Euclidean basis. Now we give a characterization of such basis in terms of the DFT of the vector w.

Lemma 3.1.5. *If $w \in \ell^2(\mathbb{Z}_N)$, then $\{R_k w\}_{k=0}^{N-1}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$ if and only if $|\hat{w}_n| = 1$ for all $n \in \mathbb{Z}_N$.*

Proof. Consider the Dirac delta function

$$\delta_k = \begin{cases} 1, & \text{if } k = 0 \\ 0, & \text{if } k = 1, \dots, N-1. \end{cases}$$

It is easy to see that $\{R_k w\}_{k=0}^{N-1}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$ if and only if $\langle w, R_k w\rangle = \delta_k$. By Lemma 3.1.4, this is equivalent to

$$w * \tilde{w} = \delta.$$

Now we apply the DFT on both sides of the equation. In one hand, for any $n \in \mathbb{Z}_N$:

$$\hat{\delta}_n = \sum_{k=0}^{N-1} \delta_k e^{-2\pi i n k/N} = e^{-2\pi i n \cdot 0/N} = 1.$$

On the other hand, by Lemma 2.1.8 and Proposition 3.1.2:

$$\widehat{(w * \tilde{w})}_n = \hat{w}_n \widehat{(\tilde{w})}_n = \hat{w}_n \overline{\hat{w}}_n = |\hat{w}_n|^2.$$

□

The condition for an orthonormal basis $\mathcal{B}$ to be of the form $\{R_k w\}_{k=0}^{N-1}$ is quite simple, but there is a problem: Lemma 3.1.5 tells us that we cannot obtain a basis of this form with the property that it is frequency localized. Nevertheless, we can slightly modify the way of constructing the orthonormal basis and it will lead to key results.

Definition 3.1.6. Let $M \in \mathbb{Z}$ and $N = 2M$. An orthonormal basis for $\ell^2(\mathbb{Z}_N)$ of the form

$$\mathcal{B} = \{R_{2k} u\}_{k=0}^{M-1} \cup \{R_{2k} v\}_{k=0}^{M-1}$$

for some $u, v \in \ell^2(\mathbb{Z}_N)$, is called a first-stage wavelet basis for $\ell^2(\mathbb{Z}_N)$.

Similarly as we have just done, we are going to find characterizations of the basis of this form in terms of the DFTs of the vectors u, v.

Definition 3.1.7. Let $M \in \mathbb{N}$, $N = 2M$ and $z \in \ell^2(\mathbb{Z}_N)$. Define $z^* \in \ell^2(\mathbb{Z}_N)$ by

$$z^*_n = (-1)^n z_n,$$

for every n.

Lemma 3.1.8. *For all n,*

$$\widehat{z^*}_n = \hat{z}_{n+M}.$$

Proof. Using the definition:

$$\begin{aligned}\widehat{z^*}_n &= \sum_{k=0}^{N-1} z^*_k e^{-2\pi ikn/N} = \sum_{k=0}^{N-1} (-1)^k z_k e^{-2\pi ikn/N} = \sum_{k=0}^{N-1} z_k e^{-i\pi k} e^{-2\pi ikn/N} \\ &= \sum_{k=0}^{N-1} z_k e^{-2\pi ik(n+M)/N} = \hat{z}_{n+M}.\end{aligned}$$

□

Remark 3.1.9. Note that for any $z \in \ell^2(\mathbb{Z}_N)$, with N even,

$$(z + z^*)_n = z_n(1 + (-1)^n) = \begin{cases} 2z_n, & \text{if } n \text{ is even,} \\ 0, & \text{if } n \text{ is odd.} \end{cases}$$

This equality will be useful in the proof of the following lemma:

Lemma 3.1.10. *Let $M \in \mathbb{N}$, $N = 2M$, and $w \in \ell^2(\mathbb{Z}_N)$. Then $\{R_{2k}w\}_{k=0}^{M-1}$ is an orthonormal set (with M elements) if and only if*

$$|\hat{w}_n|^2 + |\hat{w}_{n+M}|^2 = 2,$$

for $n = 0, 1, \ldots, M-1$.

Proof. We have already seen (Lemma 3.1.5) that $\{R_{2k}w\}_{k=0}^{M-1}$ is an orthonormal set with M elements if and only if

$$(w * \tilde{w})_{2k} = \langle w, R_{2k}w \rangle = \begin{cases} 1, & \text{if } k = 0, \\ 0, & \text{if } k = 1, \ldots, M-1. \end{cases}$$

Now, as we have observed in Remark 3.1.9:

$$(w * \tilde{w} + (w * \tilde{w})^*)_n = \begin{cases} 2(w * \tilde{w})_n, & \text{if } n \text{ is even,} \\ 0, & \text{if } n \text{ is odd.} \end{cases}$$

Hence, the result will hold true if and only if

$$w * \tilde{w} + (w * \tilde{w})^* = 2\delta.$$

Using that $\hat{\delta}_n = 1$ for all n, and applying the Fourier inversion (Proposition 2.1.6) to both sides of the equality, we obtain that the result will be true if and only if

$$\widehat{(w * \tilde{w})}_n + \quad \widehat{(w * \tilde{w})}^*_n = 2, \text{ for } n = 0, 1, \ldots, N-1. \tag{3.1}$$

By Proposition 3.1.2 and Lemma 2.1.8:

$$\widehat{(w * \tilde{w})}_n = \hat{w}_n \widehat{(\tilde{w})}_n = \hat{w}_n \overline{\hat{w}_n} = |\hat{w}_n|^2,$$

and combining this equation with Lemma 3.1.8, we obtain

$$\widehat{(w * \tilde{w})}^*_n = \widehat{(w * \tilde{w})}_{n+M} = |\hat{w}_{n+M}|^2.$$

Substituting the last two identities in the equation (3.1), we finally obtain

$$|\hat{w}_n|^2 + |\hat{w}_{n+M}|^2 = 2,$$

for $n = 0, \ldots, M-1$, which is equivalent to the orthonormality of $\{R_{2k} w\}_{k=0}^{M-1}$. □

Definition 3.1.11. Let $M \in \mathbb{N}$, $N = 2M$, and $u, v \in \ell^2(\mathbb{Z}_N)$. For any $n \in \mathbb{Z}$, define the system matrix of u and v, $A(n)$, by

$$A(n) = \frac{1}{\sqrt{2}} \begin{pmatrix} \hat{u}_n & \hat{v}_n \\ \hat{u}_{n+M} & \hat{v}_{n+M} \end{pmatrix}.$$

Definition 3.1.12. Let $A \in M_{n\times n}(\mathbb{C})$. We say that A is unitary if and only if A is invertible and $A^{-1} = \overline{A^t}$, where $\overline{A^t}$ denotes the conjugate transpose matrix of A.

Lemma 3.1.13. *Let $A \in M_{n\times n}(\mathbb{C})$. Then A is unitary if and only if the columns of A form an orthonormal basis for $\mathbb{C}^n$.*

Finally, we are ready to state and prove the theorem which characterizes first-stage wavelets:

Theorem 3.1.14. *Let $M \in \mathbb{N}$, $N = 2M$, and $u, v \in \ell^2(\mathbb{Z}_N)$. Then,*

$$\mathcal{B} = \{R_{2k} v\}_{k=0}^{M-1} \cup \{R_{2k} u\}_{k=0}^{M-1}$$

is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$ if and only if the system matrix $A(n)$ of u and v is unitary for every $n = 0, \ldots, M-1$. Equivalently, $\mathcal{B}$ is a first-stage wavelet basis for $\ell^2(\mathbb{Z}_N)$ if and only if

$$\begin{aligned} |\hat{u}_n|^2 + |\hat{u}_{n+M}|^2 &= 2, \\ |\hat{v}_n|^2 + |\hat{v}_{n+M}|^2 &= 2, \\ \hat{u}_n \overline{\hat{v}_n} + \hat{u}_{n+M} \overline{\hat{v}_{n+M}} &= 0, \end{aligned}$$

for all $n = 0, \ldots, M-1$.

Proof. We know (Lemma 3.1.13) that a 2×2 matrix is unitary if and only if its columns form an orthonormal basis for $\mathbb{C}^2$. By Lemma 3.1.10, we know that $\{R_{2k}u\}_{k=0}^{M-1}$ is an orthonormal set if and only if

$$|\hat{u}_n|^2 + |\hat{u}_{n+M}|^2 = 2,$$

for all $n = 0, \dots, M-1$, and the same happens with $\{R_{2k}v\}_{k=0}^{M-1}$. But this is the same as saying that the two columns of $A(n)$ have norm equal to 1 for every $n = 0, \dots, M-1$. Next, we claim:

$$\langle R_{2k}u, R_{2j}v \rangle = 0, \text{ for } j, k = 0, \dots, M-1 \tag{3.2}$$

if and only if

$$\hat{u}_n \overline{\hat{v}_n} + \hat{u}_{n+M} \overline{\hat{v}_{n+M}} = 0, \text{ for } n = 0, \dots, M-1. \tag{3.3}$$

If we are able to prove this equivalence, then we will be done, since equation (3.3) says that the columns of $A(n)$ are orthonormal. First of all, observe that equation (3.2) is equivalent to

$$(u * \tilde{v})_{2k} = \langle u, R_{2k}v \rangle = 0, \text{ for } k = 0, \dots M-1,$$

by Lemma 3.1.4 and the fact that equation (3.2) holds if and only if $\langle u, R_{2k}v \rangle$ for all $k = 0, \dots M-1$ (which is easy to prove). Now, using Remark 3.1.9 with $z = u * \tilde{v}$, the last equation is equivalent to

$$u * \tilde{v} + (u * \tilde{v})^* = 0.$$

But then,

$$\widehat{(u * \tilde{v})} + \widehat{(u * \tilde{v})^*} = 0.$$

By Lemma 2.1.8 and Proposition 3.1.2, we have that

$$\widehat{(u * \tilde{v})}_n = \hat{u}_n \overline{\hat{v}_n},$$

and by Lemma 3.1.8,

$$\widehat{(u * \tilde{v})}^*_n = \hat{u}_{n+M} \overline{\tilde{v}_{n+M}}.$$

Finally, note that the left hand side of equation (3.3) is periodic with period M, so it will vanish for all n if and only if it vanishes for $n = 0, \dots, M-1$, and therefore, equation (3.3) is equivalent to (3.2). □

Before proceeding, let us compare the hypothesis given in Lemma 3.1.5 and Theorem 3.1.14. In the first one, we force $|\hat{w}_n|^2 = 1$ for all n, while in the last one, we only restrict that the average of $|\hat{u}_n|^2$ and $|\hat{u}_{n+M}|^2$ equals 1. For instance, it can happen that $|\hat{u}_n|^2 = 2$ and $|\hat{u}_{n+M}|^2 = 0$, for some n. If this is the case, by Theorem 3.1.14, necessarily, $|\hat{v}_n|^2 = 0$ and $|\hat{v}_{n+M}|^2 = 2$. Recall that the Fourier basis vectors for $\ell^2(\mathbb{Z}_N)$ are given by

$$(F_n)_m = \frac{1}{N} e^{2\pi i n m/N}, \text{ for } n = 0, \dots N-1.$$

By Proposition 2.1.6, we have that

$$v = \sum_{k=0}^{N-1} \hat{v}_k F_k,$$

and since $|\hat{v}_n|^2 = 0$, we conclude that v has no component in the direction F_n. We can use this fact in the construction of a wavelet basis in the following way: choose u to contain the low frequencies (the low-pass filter), and v to contain the high frequencies (the high-pass filter, see 3.2.6).

Example 3.1.15. Consider $N = 4$, $\hat{u} = \left(\sqrt{2}, 1, 0, 1\right)$, $\hat{v} = \left(0, 1, \sqrt{2}, -1\right)$. Then:

$$A(0) = \frac{1}{\sqrt{2}} \begin{pmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{pmatrix} = Id,$$

$$A(1) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Clearly, $A(0)$ and $A(1)$ are unitary matrices. Then, by Theorem 3.1.14, it holds that $\mathcal{A} = \{u, R_2 u, v, R_2 v\}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_4)$. Indeed, we can compute u, v using the IDFT, and obtain:

$$u = \frac{1}{4}\left(2 + \sqrt{2}, \sqrt{2}, -2 + \sqrt{2}, \sqrt{2}\right),$$
$$v = \frac{1}{4}\left(\sqrt{2}, -\sqrt{2} + 2i, \sqrt{2}, -\sqrt{2} - 2i\right),$$

and now we can check directly that $\mathcal{A}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_4)$.

To conclude this section, we are going to give a result that will allow us to construct a wavelet basis from only one potential wavelet basis generator $u \in \ell^2(\mathbb{Z}_N)$, that is,

$$|\hat{u}_n|^2 + |\hat{u}_{n+N/2}|^2 = 2, \text{ for all } n.$$

Later on, we will see that Daubechies' wavelet basis are constructed using this result.

Proposition 3.1.16. *Let $M \in \mathbb{N}$, $N = 2M$, and $u \in \ell^2(\mathbb{Z}_N)$ such that $\{R_{2k}u\}_{k=0}^{M-1}$ is an orthonormal set with M elements. Define $v \in \ell^2(\mathbb{Z}_N)$ in the following way:*

$$v_k = (-1)^{k-1}\overline{u_{1-k}},$$

for all k. Then, $\{R_{2k}v\}_{k=0}^{M-1} \cup \{R_{2k}u\}_{k=0}^{M-1}$ is a first-stage wavelet basis for $\ell^2(\mathbb{Z}_N)$.

Proof. Write $k = 1 - n$, and compute

$$\begin{aligned}
\hat{v}_m &= \sum_{n=0}^{N-1} v_n e^{-2\pi i m n/N} = \sum_{n=0}^{N-1} (-1)^{n-1}\overline{u_{1-n}} e^{-2\pi i m n/N} = \sum_{k=0}^{N-1} \overline{u_k}(-1)^{-k} e^{-2\pi i m(1-k)/N} \\
&= e^{-2\pi i m/N} \sum_{k=0}^{N-1} \overline{u_k}(e^{-i\pi})^{-k} e^{2\pi i m k/N} = e^{-2\pi i m/N} \overline{\sum_{k=0}^{N-1} u_k e^{-2\pi i (m+M)k/N}} \\
&= e^{-2\pi i m/N}\overline{\hat{u}_{m+M}}.
\end{aligned}$$

Hence,

$$\hat{v}_{m+M} = e^{-2\pi i(m+M)/N}\overline{\hat{u}_{m+2M}} = e^{-2\pi iM/N}e^{-2\pi im/N}\overline{\hat{u}_m} = -e^{-2\pi im/N}\overline{\hat{u}_m},$$

and using Lemma 3.1.10, we conclude that

$$|\hat{v}_m|^2 + |\hat{v}_{m+M}|^2 = |\hat{u}_{m+M}|^2 + |\hat{u}_m|^2 = 2,$$

for $m = 0, 1, \ldots, M-1$. Finally:

$$\hat{u}_m\overline{\hat{v}_m} + \hat{u}_{m+M}\overline{\hat{v}_{m+M}} = \hat{u}_m e^{2\pi im/N}\hat{u}_{m+M} - \hat{u}_{m+M}e^{2\pi im/N}\hat{u}_m = 0.$$

Applying Theorem 3.1.14, we obtain that the vectors u, v generate a first-stage wavelet basis. □

3.2 Examples of first-stage wavelets

From now on, we will consider $N = 2M$, $M \in \mathbb{N}$, unless we specify otherwise. The purpose of the first-stage wavelets (wavelets) is to compress data. That is, for any vector $x \in \ell^2(\mathbb{Z}_N)$, we want to keep the least possible information in such a way that after the decompression is done, the loss of quality is minimal. For a wavelet basis $\{R_{2k}u\}_{k=0}^{M-1} \cup \{R_{2k}v\}_{k=0}^{M-1} \subset \ell^2(\mathbb{Z}_N)$, we can construct the matrix M containing the vectors of such a basis, obtaining a linear transformation, namely A. Applying A to the vector x, we have

$$A(x)_j = \begin{cases} \langle R_{2j}u, x\rangle, & \text{if } j = 0, \ldots, M-1, \\ \langle R_{2j}v, x\rangle, & \text{if } j = M, \ldots, 2M-1. \end{cases}$$

The particularity of wavelet basis is that the vector $A(x)$ will have a high number of components equal to zero (or at least relatively small). This will allow us to discard those elements without losing much quality, obtaining a vector y with a high number of components exactly equal to zero. After we decompress y, or equivalently, after computing the inverse transform $x' = A^{-1}(y)$, x' will not differ too much from x. Typically, the compression method will consist of the following steps:

1. For $x \in \ell^2(\mathbb{Z}_N)$, we apply the transformation, $x \mapsto A(x)$.

2. We apply a filter F to the components of $A(x)$. That is, we make zero all the components which norm is below $\varepsilon > 0$. We note that as ε increases, the information we lose increases as well, since we are forcing more components to be zero. After this, we obtain a vector $y = F(A(x))$, with more zeros than $A(x)$, and we apply a compression algorithm.

3. We can recover the filtered vector through the decompression algorithm, obtaining $x' = A^{-1}(y) = A^{-1}(F(A(x)))$. Then, x' will not be equal to the original vector x (for reasonable values of ε), but the two vectors will be close to each other, meaning that the norm $\|x - x'\|$ will be small.

The fundamental observation is that since we are applying linear transformations, if we take small values of ε, the difference between x and x' will also be small (in norm), and therefore we will not lose much quality in the process. We shall now see examples of wavelets illustrating this compression.

3.2.1 The Haar transform

Definition 3.2.1. Let

$$u = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, 0, \ldots, 0\right),$$
$$v = \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 0, 0, \ldots, 0\right).$$

Then, we define the first-stage Haar basis for $\ell^2(\mathbb{Z}_N)$ as

$$\mathcal{H} = \{R_{2k}u\}_{k=0}^{M-1} \cup \{R_{2k}v\}_{k=0}^{M-1}.$$

We can check trivially that the Haar first-stage wavelet basis is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$. Define the Haar transform matrix as

$$H_N = \big(h_{jk}\big) = \begin{cases} (R_{2j}u)_k, & \text{if } j = 0, \ldots, M-1, \\ (R_{2j}v)_k, & \text{if } j = M, \ldots, N-1. \end{cases}$$

Then, for any $z \in \ell^2(\mathbb{Z}_N)$ we have that

$$z = \sum_{k=0}^{M-1} \Big(\langle z, R_{2k}u\rangle R_{2k}u + \langle z, R_{2k}v\rangle R_{2k}v \Big).$$

If we write

$$P(z) = \sum_{k=0}^{M-1} \langle z, R_{2k}u\rangle R_{2k}u, \qquad Q(z) = \sum_{k=0}^{M-1} \langle z, R_{2k}v\rangle R_{2k}v,$$

we can notice that

$$P(z)_{2j} = \sum_{k=0}^{M-1} \langle z, R_{2k}u\rangle (R_{2k}u)_{2j} = \sum_{k=0}^{M-1} \langle z, R_{2k}u\rangle (R_{2k}u)_{2j+1} = P(z)_{2j+1}, \tag{3.4}$$

since it holds that $(R_{2k}u)_{2j} = (R_{2k}u)_{2j+1}$, for every $0 \le j, k \le M-1$. Moreover:

$$P(z)_{2j} = \frac{1}{\sqrt{2}} \langle z, R_{2j}u\rangle = \frac{z_{2j} + z_{2j+1}}{2}.$$

This means that we obtain the vector $P(z)$ by replacing the values of the signal z at $2m$ and $2m+1$ by their average. Then, we can interpret the vector $P(z)$ as z

seen at resolution of 2. This vector will be the tendency of the signal, containing most of its information. On the other hand, $Q(z)$ will be the vector containing the details which are needed to pass from a resolution of 2 to a resolution of 1, and hence, if we want to apply a compression, we will discard $Q(z)$, keeping at most $N/2$ different coefficients which define $P(z)$. Similarly as in (3.4), we can check that $Q(z)_{2j} = -Q(z)_{2j+1}$, for all $0 \leq j \leq M-1$. This means that we will keep at most $N/2$ different coefficients to reconstruct the vector $Q(z)$. Then we define:

$$\begin{aligned}(P_1'(z))_j =&(P(z))_{2j}, \text{ for } j = 0, \ldots, M-1,\\ (Q_1'(z))_j =&(Q(z))_{2j}, \text{ for } j = 0, \ldots, M-1,\end{aligned} \tag{3.5}$$

so that $P_1'(z), Q_1'(z) \in \ell^2(\mathbb{Z}_M)$. Now suppose that 4 divides N (and hence, 2 divides M). Then we can apply the Haar transform for $\ell^2(\mathbb{Z}_{N/2})$ to the vector $P_1'(z)$, containing the tendency of the first iteration of the Haar transform. We obtain:

$$P_2(z) = \sum_{k=0}^{M/2-1} \langle P_1'(z), R_{2k}u'\rangle R_{2k}u', \qquad Q_2(z) = \sum_{k=0}^{M/2-1} \langle P_1'(z), R_{2k}v'\rangle R_{2k}v',$$

where

$$u' = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \underbrace{0, \ldots, 0}_{M-2}\right), \; v' = \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, \underbrace{0, \ldots, 0}_{M-2}\right).$$

The vectors $P_2(z)$ and $Q_2(z)$ are now the tendency and the details of $P_1'(z)$, respectively. Also observe that the vectors $P_2'(z), Q_2'(z) \in \ell^2(\mathbb{Z}_{M/2})$, defined in the same way as in (3.5), contain all the information about $P_2(z)$ and $Q_2(z)$ keeping only half of the coefficients. The sequence of vectors that we need to fully recover the signal z is now:

$$H^2(z) = (P_2'(z), Q_2'(z), Q_1'(z)).$$

More generally, if 2^n divides N, then we can iterate the Haar transform until we obtain the n-th tendency of the signal, $P_n'(z)$, in such a way that the sequence of vectors needed to recover the original signal z will be

$$H^n(z) = (P_n'(z), Q_n'(z), Q_{n-1}'(z), \ldots, Q_1'(z)).$$

We will refer to H^n as the n-th order Haar transform. In this case, the vector $P_n'(z)$ has size $N/2^n$, and therefore, we will have a vector of size $N - N/2^n$ containing the details obtained in all the iterations of the transformation.

Remark 3.2.2. For the 2-dimensional Haar transform, our signal z will be a complex matrix, and the 2D Haar transform is defined just by iteration. That is, first we apply the transformation to the rows of z, obtaining a new matrix y, and then apply the transformation to the columns of y.

Example 3.2.3. Consider the function $f(x) = \sin(13x) - \cos(7x)$. We will use the function `HaarTrans.m` given in the Annex to find the Haar transform of order 3 of

a finite number of values of f. Let $X = \{0, \ldots, 399\}$. The vector of images will be defined by $Y_n = f(X_n)$. Figure 3.1 contains the graphics of the stages of the 3rd order Haar transform. Note that at the n-th step, the vector containing the tendency will be the one corresponding to the first $N/2^n$ coefficients (in this case 200, 100, and 50).

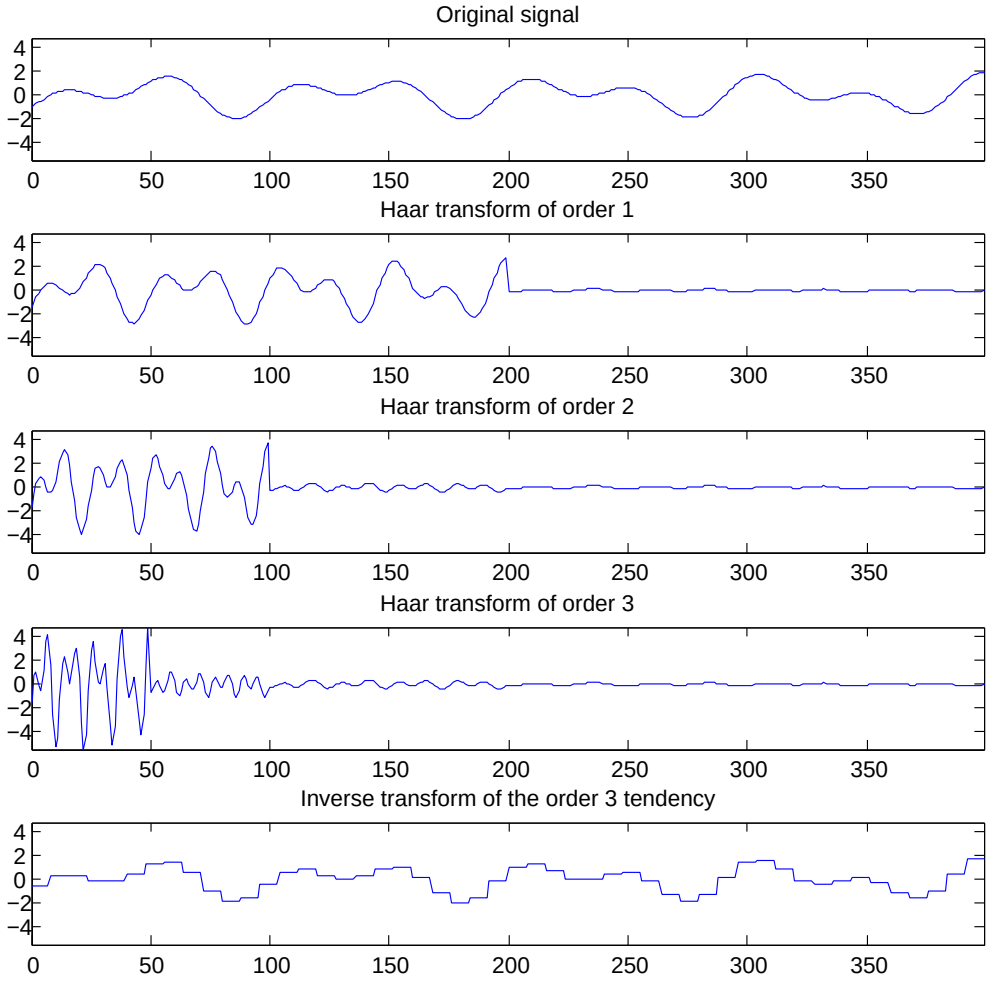

Figure 3.1: Different stages of the 3rd order Haar transform.

3.2.2 The Shannon transform

Definition 3.2.4. Let N be divisible by 4. Define $\hat{u}, \hat{v} \in \ell^2(\mathbb{Z}_N)$ by

$$\begin{aligned} \hat{u}_n &= \begin{cases} \sqrt{2}, & \text{if } n = 0, 1, \dots, N/4 - 1 \text{ or } n = 3N/4, 3N/4 + 1, \dots, N - 1, \\ 0, & \text{otherwise,} \end{cases} \\ \hat{v}_n &= \begin{cases} 0, & \text{if } n = 0, 1, \dots, N/4 - 1 \text{ or } n = 3N/4, 3N/4 + 1, \dots, N - 1, \\ \sqrt{2}, & \text{otherwise.} \end{cases} \end{aligned} \tag{3.6}$$

Then, we define the Shannon wavelet basis for $\ell^2(\mathbb{Z}_N)$:

$$\mathcal{S} = \{R_{2k}u\}_{k=0}^{N/2-1} \cup \{R_{2k}v\}_{k=0}^{N/2-1}.$$

Proposition 3.2.5. *$\mathcal{S}$ is a wavelet basis for $\ell^2(\mathbb{Z}_N)$.*

Proof. This result follows easily from the characterization of the wavelet basis given in Theorem 3.1.14. Indeed, let $n \in \{0, 1, \dots N/2 - 1\}$. It always holds that if $\hat{u}_n = \sqrt{2}$, then $\hat{u}_{n+N/2} = 0$, and vice-versa. The same happens with the coefficients of $\hat{v}$. Moreover, for every $m \in \mathbb{Z}_N$, either $\hat{u}_m$ or $\hat{v}_m$ vanish. Therefore,

$$\hat{u}_n\overline{\hat{v}_n} + \hat{u}_{n+N/2}\overline{\hat{v}_{n+N/2}} = 0.$$

□

We observe that the coefficients of the vector $\hat{u}$ vanish in the $N/2$ different positions which are closer to $N/2$ (in $\mathbb{Z}_N$). In the same way, the components of $\hat{v}$ vanish in the $N/2$ indexes which are closer to 0. This leads to the following definition:

Definition 3.2.6. Let $x \in \ell^2(\mathbb{Z}_N)$. The indexes of $\hat{x}$ are called the frequency scale. The high frequencies are the ones closer to $N/2$, while the low frequencies are the ones closer to 0, with the arithmetic of $\mathbb{Z}_N$. Thus, if x is frequency localized near 0 ($N/2$ respectively), we will say that x is a low-pass filter (high-pass filter respectively).

In the way that $\hat{u}$ and $\hat{v}$ have been defined in (3.6), it turns out that u and v will be the low and high pass filters, respectively. Since we have obtained the wavelet basis through the DFTs of the vectors u, v, if we want to find their explicit expression, we need to compute the IDFTs. We can easily check that if $n = 1, 2, \dots, N - 1$, then

$$u_n = \frac{\sqrt{2}}{N} e^{-in\pi/N} \frac{\sin(\pi n/2)}{\sin(\pi n/N)}, \qquad v_n = (-1)^n u_n,$$

and $u_0 = v_0 = 1/\sqrt{2}$.

We observe that the vectors u and v are not real-valued. This is something we do not want to happen, since we are going to deal with real-valued signals z (as for example, audio or video signals). If we have a basis $\mathcal{B}$ of real-valued vectors, then the coefficients in the expansion of z in terms of $\mathcal{B}$ will also be real, because they are the inner products of z with the basis elements. In this case, the computations are simpler and moreover, we will save computer memory, since complex vectors are stored as pairs of real vectors. Nevertheless, we can slightly modify the definition of $\hat{u}$ and $\hat{v}$ in order to get an orthonormal basis made of real-valued vectors.

3.2.3 The real Shannon transform

Definition 3.2.7. Let $\hat{u}$ and $\hat{v}$ be as in (3.6), except for the following specified coefficients:

$$\hat{u}_{N/4} = -i, \qquad \hat{u}_{3N/4} = i, \qquad \hat{v}_{N/4} = \hat{v}_{3N/4} = 1.$$

We define the real Shannon basis as

$$\mathcal{S}_{\mathbb{R}} = \{R_{2k}u\}_{k=0}^{N/2-1} \cup \{R_{2k}v\}_{k=0}^{N/2-1}.$$

Proposition 3.2.8. *$\mathcal{S}_{\mathbb{R}}$ is a first-stage wavelet basis of real-valued vectors.*

In order to prove this proposition, we need an auxiliary lemma about real-valued vectors:

Lemma 3.2.9. *Let $z \in \ell^2(\mathbb{Z}_N)$. Then z is real-valued if and only if $\hat{z}_m = \overline{\hat{z}_{N-m}}$, for all m.*

Proof. Note that z is real-valued if and only if $\overline{z} = z$, and this holds if and only if $\hat{z} = \widehat{(\overline{z})}$. But by the properties of complex conjugation:

$$\widehat{\overline{z}}_m = \sum_{n=0}^{N-1} \overline{z}_n e^{-2\pi imn/N} = \overline{\sum_{n=0}^{N-1} z_n e^{2\pi imn/N}} = \overline{\hat{z}_{-m}} = \overline{\hat{z}_{N-m}},$$

and hence, we conclude that $\hat{z} = \widehat{(\overline{z})}$ if and only if $\hat{z}_m = \overline{\hat{z}_{N-m}}$, for all m. □

Proof of Proposition 3.2.8. First we prove that u and v are real-valued vectors. Note that $\hat{u}_{N/4} = i = \overline{-i} = \overline{\hat{u}_{3N/4}}$. On the other hand, $\hat{v}_{N/4} = \hat{v}_{3N/4} = 1$. For the rest of indexes it is clear that the symmetry condition is satisfied. In order to prove that $\mathcal{S}_{\mathbb{R}}$ is a first-stage wavelet basis we only need to prove that the system matrix $A(N/4)$ is unitary. We already know that $A(n)$ is unitary for all $n \neq N/4$ (it follows from the fact that $\mathcal{S}$ is a wavelet basis and Theorem 3.1.14). We have that

$$A(N/4) = \frac{1}{\sqrt{2}} \begin{pmatrix} i & 1 \\ -i & 1 \end{pmatrix}$$

is clearly unitary, and hence, the result follows. □

Thus, we have improved the Shannon wavelet basis into a basis consisting of real-valued vectors, and it preserves the property that u is a low-pass filter and v is a high-pass filter. This will be a suitable basis to work with in signal analysis. Again, note that we defined $\mathcal{S}_{\mathbb{R}}$ in terms of the DFTs of u and v, so we need to compute the IDFTs in order to obtain the explicit expressions of u, v. Figure 3.2 shows how u, v (or either, its translation) look like, for $N = 512$.

For the 2D real Shannon transform, we iterate in the same way as we did with the 2D Haar transform (Remark 3.2.2).

We shall now see a couple of examples of the compression that can be done through the real Shannon basis. In order to get a compression, we will discard the lower coefficients of the Shannon transform (in absolute value) and keep the higher ones (i.e., the most relevant). Depending on how much coefficients we drop, the compression will improve, as the recovered signal quality will decrease.

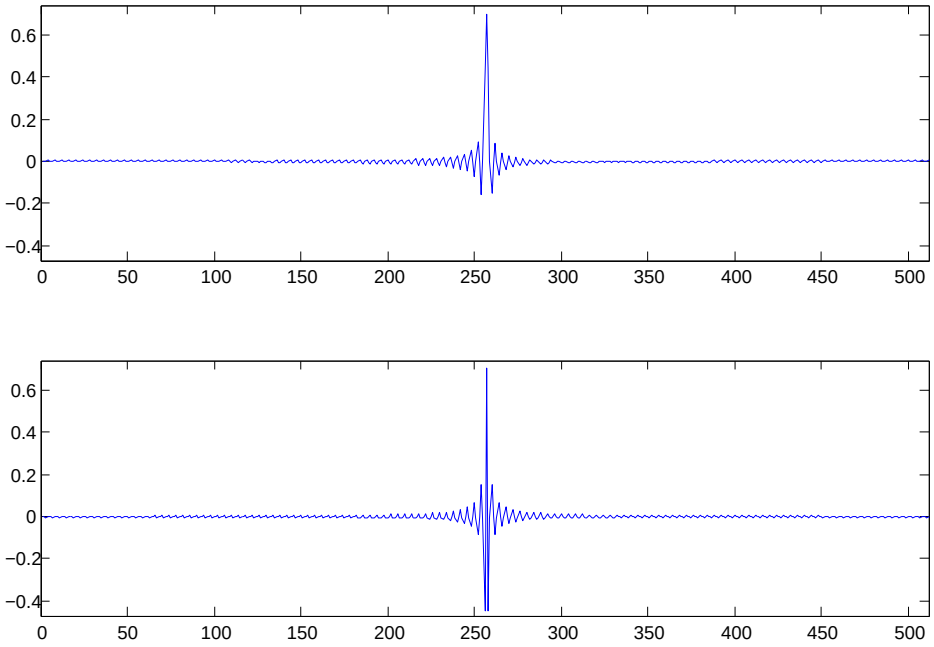

Figure 3.2: Real Shannon wavelets: with $N = 512$, $R_{256}u$ and $R_{256}v$, respectively.

Example 3.2.10. Consider $x_j = j$, for $j = 0, \ldots, 511$, and $y_j = f(x_j)$, where $f(x) = \sin(x/175) - 3\cos(x/100)$. Figure 3.3 presents three reconstructions of the signal, keeping $70\%, 50\%$, and 30% of the coefficients.

We can observe that the recovered signal is very different from the original one when keeping only 30% of the coefficients. This is because we have lost all the information about the coefficients involving the vector v, since those were (mostly) the lowest ones. In contrast, we can see that the recovered signal with 50% of the coefficients is quite good compared with the original. This is because there are also low coefficients coming from the scalar product $\langle z, R_k\tilde{u}\rangle$ (Lemma 3.1.4), so there are relatively high coefficients involving v which did not vanish when compressing, and the reconstruction is better. Figure 3.4 shows such coefficients.

3.2.4 Daubechies' D6 wavelet

Daubechies' D6 wavelets were developed by Ingrid Daubechies in 1988, being these the most recent wavelets we are going to study. They were firstly constructed in the contexts of $\mathbb{R}$ and $\mathbb{Z}$, (see [2], Chapters 2 and 3, respectively) but here we adapt it to the case of $\mathbb{Z}_N$.

We begin assuming that $N > 6$ is an even natural number, and $M = N/2$. We are going to construct a vector $u \in \ell^2(\mathbb{Z}_N)$ with only 6 nonzero elements, and

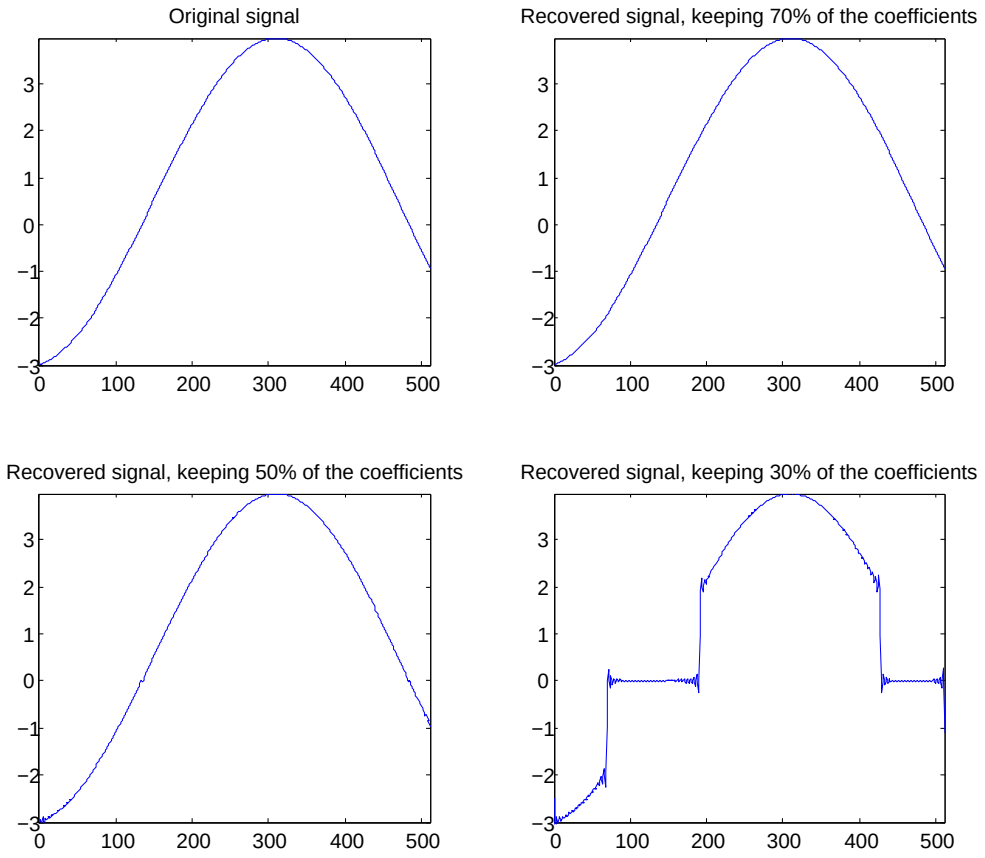

Figure 3.3: Plots corresponding to Example 3.2.10. The relative errors done in the compression are $< 10^{-3}$, 0.0044, and 0.2638, respectively.

eventually obtain a first-stage wavelet basis using Proposition 3.1.16, and it will have good localization in space, rather than good frequency localization. We start with the identity

$$\left(\cos^2\left(\frac{\pi n}{N}\right) + \sin^2\left(\frac{\pi n}{N}\right)\right)^5 = 1.$$

Expanding it, we obtain:

$$\begin{aligned}&\cos^{10}\left(\frac{\pi n}{N}\right) + 5\cos^8\left(\frac{\pi n}{N}\right)\sin^2\left(\frac{\pi n}{N}\right) + 10\cos^6\left(\frac{\pi n}{N}\right)\sin^4\left(\frac{\pi n}{N}\right)\\ &+ 10\cos^4\left(\frac{\pi n}{N}\right)\sin^6\left(\frac{\pi n}{N}\right) + 5\cos^2\left(\frac{\pi n}{N}\right)\sin^8\left(\frac{\pi n}{N}\right) + \sin^{10}\left(\frac{\pi n}{N}\right) = 1. \qquad (3.7)\end{aligned}$$

Now consider the identities

$$\begin{aligned}\cos\left(\frac{\pi(n+M)}{N}\right) &= \cos\left(\frac{\pi n}{N} + \frac{\pi}{2}\right) = -\sin\left(\frac{\pi n}{N}\right),\\ \sin\left(\frac{\pi(n+M)}{N}\right) &= \cos\left(\frac{\pi n}{N}\right). \qquad (3.8)\end{aligned}$$

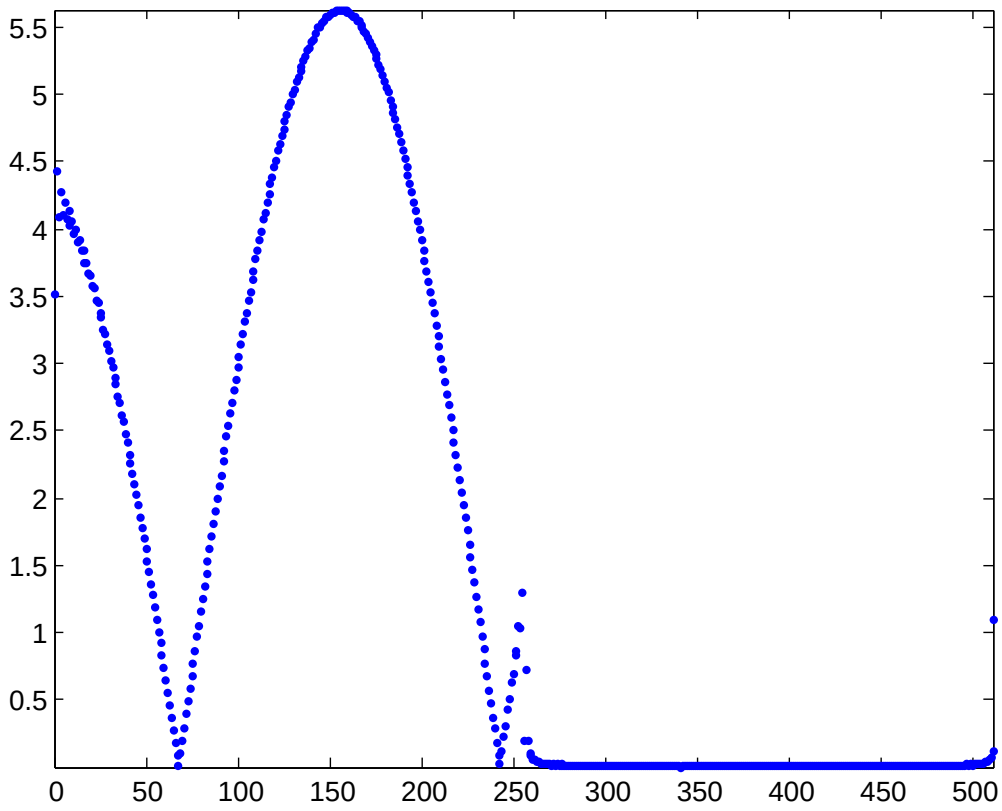

Figure 3.4: Modulus of the coefficients in the real Shannon basis of the signal from Example 3.2.10.

If we define

$$b_n = \cos^{10}\left(\frac{\pi n}{N}\right) + 5\cos^8\left(\frac{\pi n}{N}\right)\sin^2\left(\frac{\pi n}{N}\right) + 10\cos^6\left(\frac{\pi n}{N}\right)\sin^4\left(\frac{\pi n}{N}\right),$$

and then use the identities shown in (3.8), it turns out that

$$b_{n+M} = 10\cos^4\left(\frac{\pi n}{N}\right)\sin^6\left(\frac{\pi n}{N}\right) + 5\cos^2\left(\frac{\pi n}{N}\right)\sin^8\left(\frac{\pi n}{N}\right) + \sin^{10}\left(\frac{\pi n}{N}\right).$$

Thus, by equation (3.7), we have that

$$b_n + b_{n+M} = 1, \text{ for all } n.$$

Finally, we choose $u \in \ell^2(\mathbb{Z}_N)$ such that $|\hat{u}_n|^2 = 2b_n$, so that

$$|\hat{u}_n|^2 + |\hat{u}_{n+M}|^2 = 2, \text{ for } n = 0, 1, \ldots, M-1.$$

Our constructed vector u is a potential wavelet generator, though we still need to find its expression from the conditions we have so far. We can re-write b_n in the following way:

$$\begin{aligned} b_n &= \cos^6\left(\frac{\pi n}{N}\right)\Big[\cos^4\left(\frac{\pi n}{N}\right) + 5\cos^2\left(\frac{\pi n}{N}\right)\sin^2\left(\frac{\pi n}{N}\right) + 10\sin^4\left(\frac{\pi n}{N}\right)\Big] \\ &= \cos^6\left(\frac{\pi n}{N}\right)\Big[\Big(\cos^2\left(\frac{\pi n}{N}\right) - \sqrt{10}\cos^2\left(\frac{\pi n}{N}\right)\Big)^2 \\ &\quad + \big(5+2\sqrt{10}\big)\cos^2\left(\frac{\pi n}{N}\right)\sin^2\left(\frac{\pi n}{N}\right)\Big]. \end{aligned}$$

We define $\hat{u} \in \ell^2(\mathbb{Z}_N)$ as

$$\begin{aligned} \hat{u}_n =& \sqrt{2}e^{-5\pi i n/N}\cos^3\left(\frac{\pi n}{N}\right)\Bigg[\cos^2\left(\frac{\pi n}{N}\right) - \sqrt{10}\sin^2\left(\frac{\pi n}{N}\right) \\ &+ i\sqrt{5+2\sqrt{10}}\cos\left(\frac{\pi n}{N}\right)\sin\left(\frac{\pi n}{N}\right)\Bigg]. \end{aligned}$$

It can be easily checked that under this definition, $|\hat{u}_n|^2 = 2b_n$. Using the fact that $e^{ix} = \cos x + i\sin x$ (Euler's formula) for every $x \in \mathbb{R}$ and the double angle identities, we can write:

$$\begin{aligned} \hat{u}_n =& \sqrt{2}e^{-2\pi i 4n/N}e^{3\pi i n/N}\left(\frac{e^{\pi i n/N} + e^{-\pi i n/N}}{2}\right)^3\Bigg[\frac{1}{2}\left(1+\cos\left(\frac{2\pi n}{N}\right)\right) \\ &- \frac{\sqrt{10}}{2}\left(1-\cos\left(\frac{2\pi n}{N}\right)\right) + i\frac{\sqrt{5+2\sqrt{10}}}{2}\sin\left(\frac{2\pi n}{N}\right)\Bigg]. \end{aligned}$$

In order to simplify the notation, we will write

$$a = 1-\sqrt{10},\ b = 1+\sqrt{10},\ c = \sqrt{5+2\sqrt{10}},$$

and using Euler's formula again, we re-arrange the terms to obtain

$$\hat{u}_n = \frac{\sqrt{2}}{8}e^{-2\pi i 4n/N}\big(e^{2\pi i n/N}+1\big)^3\left(\frac{a}{2} + \frac{b}{4}\big(e^{2\pi i n/N}+e^{-2\pi i n/N}\big) + \frac{c}{4}\big(e^{2\pi i n/N}-e^{-2\pi i n/N}\big)\right).$$

In order to have the best localization possible, we can force u to be of the form

$$u = (u_0, u_1, \ldots, u_5, \ldots, 0),$$

which implies that

$$\hat{u}_n = \sum_{k=0}^{5} u_k e^{-2\pi i k n/N}.$$

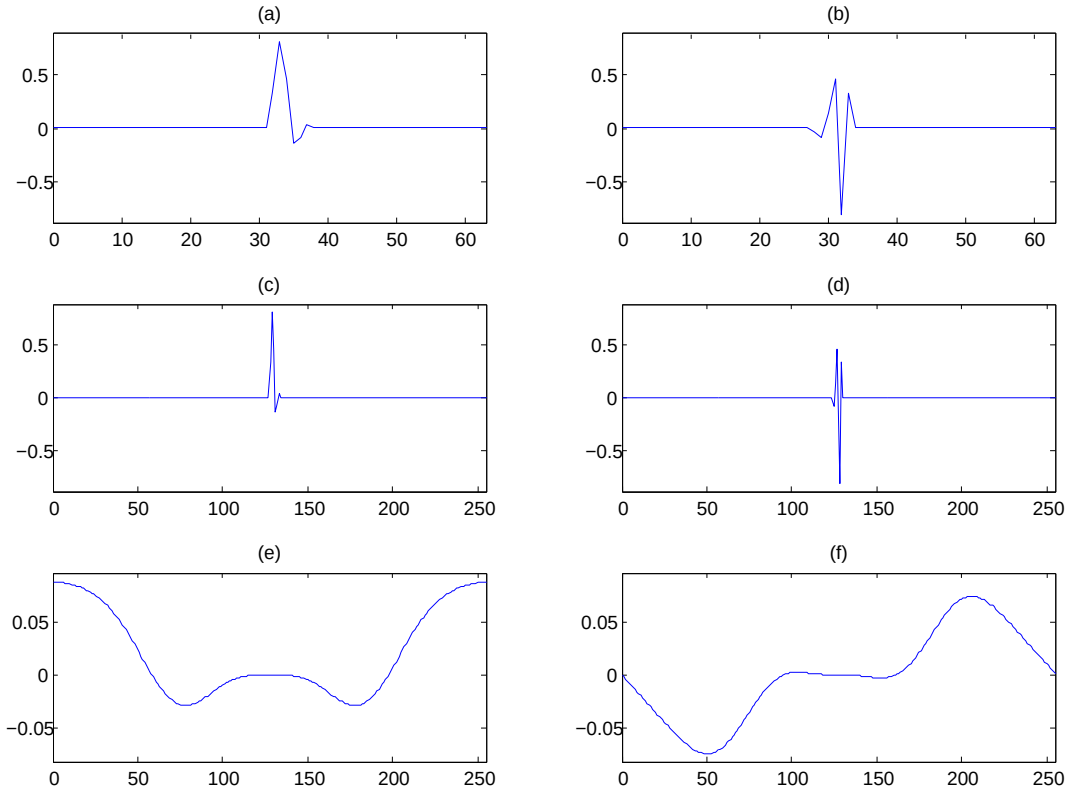

Figure 3.5: Items (a) and (b): Daubechies' $D6$ wavelets, for $N = 64$. Plots (c) and (d) refer to the case $N = 256$, and (e), (f) show Re $\hat{u}$, Im $\hat{u}$ for $N = 256$, respectively.

At this point, after some algebraic manipulations, we can already check out that

$$u = \frac{\sqrt{2}}{32}(b + c, 2a + 3b + 3c, 6a + 4b + 2c, 6a + 4b - 2c, 2a + 3b - 3c, b - c, 0, \ldots, 0).$$

Finally, the expression of v is computed through Proposition 3.1.16:

$$v = (-u_1, u_0, 0, \ldots, 0, -u_5, u_4, -u_3, u_2).$$

Note that v has 6 nonzero components as well.

Before going further, it is worth to comment that the terminology "$D6$" specifies the number of nonzero entries in the vectors u, v. In fact, Daubechies constructed a similar basis whose vectors have $2k$ nonzero entries, for all $k > 1$. Furthermore, if we consider $D2$ wavelet basis, which is constructed in the same way as above, and start with the identity

$$\cos^2\left(\frac{2\pi n}{N}\right) + \sin^2\left(\frac{2\pi n}{N}\right) = 1,$$

we can check that the resulting basis is the Haar wavelet, with the exception that v is multiplied by -1.

We can see the good localization of the Daubechies' $D6$ wavelet basis in Figure 3.5, with two different dimensions, $N = 64$ and $N = 256$. Items (a) and (b) show $R_{32}u$ and $R_{32}v$, respectively. The rest of pictures refer to the case $N = 256$: (c) and (d) are $R_{128}u$, $R_{128}v$, respectively, and the last two pictures, (e) and (f), show the real and imaginary parts of u when $N = 256$. We can appreciate that the $D6$ wavelets have better space than frequency localization, as we stated earlier on. However, this basis provides much more frequency localization than the Euclidean one (which does not have frequency localization at all), just by extending the number of nonzero elements of u, v to 6, instead of 1. This is why Daubechies $D6$ wavelet basis works much better than the standard basis in matter of signal processing.

Chapter 4

The iteration step: p-th stage wavelets

So far we have constructed and characterized (Theorem 3.1.14) orthonormal bases for $\ell^2(\mathbb{Z}_N)$, N even, of the form

$$\{R_{2k}v\}_{k=0}^{N/2-1} \cup \{R_{2k}u\}_{k=0}^{N/2-1},$$

named first-stage wavelets. We have also concentrated the high frequencies in the terms involving v, and the low frequencies in those involving u, in such a way that we get some degree of frequency localization. This also provided us spatial localization, as we can be observe in Figure 3.2. In fact, for instance, the real Shannon transform divided the frequency scale in 2 halves. Nevertheless, if we want to apply discrete wavelet theory to musical signals, it is more natural to consider frequencies on a logarithmic scale, in octaves. Motivated by this observation, we can split the low frequency half scale into two equal parts while leaving the high frequency untouched. Then, we can subdivide the lowest frequency quarter, and so on. Doing this, the basis decomposition will give us a more refined frequency analysis of a signal.

This kind of iteration can already be appreciated in the n-th order Haar transform. Recall that in the first iteration, we split a signal z into two vectors, $P'(z)$ and $Q'(z)$ (of length $N/2$), corresponding to the terms coming from the translates of u and v, respectively. We also noticed that $P'(z)$ is obtained by averaging each pair of values $2k, 2k+1$, for $k = 0, \ldots, N/2-1$, and hence, it can be seen as the original signal with a scale of 2 instead of 1. Perhaps we have enough information with a large-scale behavior of z, so we can ignore the coefficients of Q', or maybe even a larger scale is enough for the purpose we may be interested in, so there is no need to keep the full detail of a signal. With this procedure, we will probably save time and energy. To illustrate how important this kind of compression is, consider, for example, a NASA non-manned scouting mission in Mars. A lot of pictures of the environment are sent, meaning that we cannot expect to have full detail of each one of the pictures[1]. But we can send the pictures with a larger scale that will allow

[1]The Earth's orbit radius is around 150 million km, while Mars' is around 227 million km (in

us to decide whether there is something of interest in that image or not. If there is, we can add details gradually until we can identify it.

This chapter will be devoted to the study of p-th stage wavelets: definition, characterization, applications and advantages with respect to the first-stage wavelets. But before, we need to introduce various concepts and prove some results that will be used. Since we will still be working with wavelet basis, we will only consider even dimensions, i.e., $N = 2M$, for some $M \in \mathbb{N}$.

4.1 Preliminaries: Operators and auxiliary results

Lemma 4.1.1. *Let $A \in M_N(\mathbb{C})$. Then, the following are equivalent:*

1. *A is unitary.*
2. *The columns of A form an orthonormal basis for $\mathbb{C}^N$.*
3. *The rows of A form an orthonormal basis for $\mathbb{C}^N$.*

Lemma 4.1.2. *Let $\mathcal{E} = \{e_0, e_1, \dots, e_{N-1}\}$ be the Euclidean basis for $\mathbb{C}^N$, and let $\mathcal{U} = \{u_0, u_1, \dots, u_{N-1}\}$ be an orthonormal basis for $\mathbb{C}^N$. Define U to be the matrix whose j-th column is the vector u_j. Then:*

1. *U is the $\mathcal{U}$-to-$\mathcal{E}$ change of basis matrix, and hence, $\overline{U^t}$ is the $\mathcal{E}$-to-$\mathcal{U}$ change of basis.*
2. *Let $T : \mathbb{C}^N \to \mathbb{C}^N$ be a linear transformation represented by the matrix $A_{\mathcal{E}}$ in the standard basis. Then, the matrix which represents T in the basis $\mathcal{U}$ is*

$$A_{\mathcal{U}} = \overline{U^t} A_{\mathcal{E}} U.$$

Now suppose that we have a wavelet basis $\mathcal{B} = \{R_{2k}v\}_{k=0}^{M-1} \cup \{R_{2k}u\}_{k=0}^{M-1}$. By Lemma 4.1.2, the $\mathcal{B}$-to-$\mathcal{E}$ change of basis matrix, say B, is formed by the vectors $v, R_2v, \dots R_{M-1}v, u, R_2u, \dots, R_{M-1}v$, in this order. By Lemma 4.1.1, since $\mathcal{B}$ is orthonormal, then B is unitary, so that the $\mathcal{E}$-to-$\mathcal{B}$ change of basis matrix is $B^{-1} = \overline{B^t}$. However, if we want to compute $[z]_{\mathcal{B}}$ by multiplying $\overline{B^t}z$ directly, we need to compute N^2 multiplications. Nevertheless, we can make this process faster: the coefficient of $R_{2k}v$ in the expansion of z is $\langle z, R_{2k}v\rangle = (z * \tilde{v})_{2k}$, by Lemma 3.1.4, and the same happens with the coefficients involving u. This means that we can reduce the number of multiplications using the FFT in order to compute the convolutions $z * \tilde{v}$, $z * \tilde{u}$, as stated in Lemma 2.1.8. The expression of $[z]_{\mathcal{B}}$ will be:

$$[z]_{\mathcal{B}} = ((z * \tilde{v})_0, (z * \tilde{v})_2, \dots, (z * \tilde{v})_{N-2}, (z * \tilde{u})_0, (z * \tilde{u})_2, \dots, (z * \tilde{u})_{N-2}). \quad (4.1)$$

Now we are going to define an operator which will allow us to re-write the expression (4.1) in an easy way.

mean). The minimum distance between the Earth and Mars is around 59 million km. This means that the information sent from the probes must travel a huge distance.

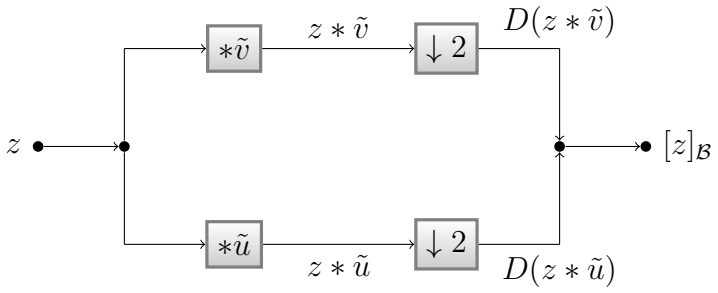

Figure 4.1: Diagram of the $\mathcal{E}$-to-$\mathcal{B}$ change of basis of a vector z, as noted in (4.1).

Definition 4.1.3. Let $N = 2M$ with $M \in \mathbb{N}$. We define $D : \ell^2(\mathbb{Z}_N) \to \ell^2(\mathbb{Z}_M)$ as

$$D(z)_n = z_{2n}, \text{ for } n = 0, 1, \ldots, M-1.$$

D is called the downsampling operator, and it will denoted by $\downarrow 2$ in diagrams.

What this operator does is, actually, discarding the odd-indexed entries of a vector $z \in \ell^2(\mathbb{Z}_{2M})$. We have already used this operator without defining it, in order to work with the n-th order Haar transform. Recall that, for instance, at first step, we obtained two vectors, corresponding to the 1st order tendency and details respectively, say $P(z)$ and $Q(z)$. We noted that those two vectors could be determined by taking only half of their coefficients, which were the even-indexed ones. Hence, equation (3.5) is equivalent to $P_1'(z) = D(P(z))$, $Q_1'(z) = D(Q(z))$.

Definition 4.1.4. For any sequence of convolutions and other operations applied to a vector (and its operation's outputs) such as upsample or downsample, sum, or concatenate, we refer to it as a filter bank.

The study of filter banks is a whole subject in engineering, called multirate signal analysis. The filter banks we are going to use will be simple, and we will often represent them with diagrams. In diagrams, we will have boxes and arrows. Boxes will represent operators, and every arrow will be associated to a vector; for a box C, the input for the corresponding operator will be the vector whose arrow ends at C, while the output will be a vector whose arrow starts at C. The first example of filter bank is Figure 4.1, which shows the $\mathcal{E}$-to-$\mathcal{B}$ change of basis process of a vector $z \in \ell^2(\mathbb{Z}_N)$.

The next question we may approach is whether the inverse transformation, the $\mathcal{B}$-to-$\mathcal{E}$ change of basis, can be done quickly. We can always multiply a vector z by the matrix U, but it requires N^2 products, as we noted before. Using filter banks, we will be able to quicken this process. So, the first thing to do, is to define a kind of inverse operator for D:

Definition 4.1.5. Let $N = 2M$ with $M \in \mathbb{N}$. We define $U : \ell^2(\mathbb{Z}_M) \to \ell^2(\mathbb{Z}_N)$ as

$$U(z)_n = \begin{cases} z_{n/2}, & \text{if } n \text{ is even,} \\ 0, & \text{if } n \text{ is odd.} \end{cases}$$

U is called the upsampling operator, and it will be denoted by $\uparrow 2$ in diagrams.

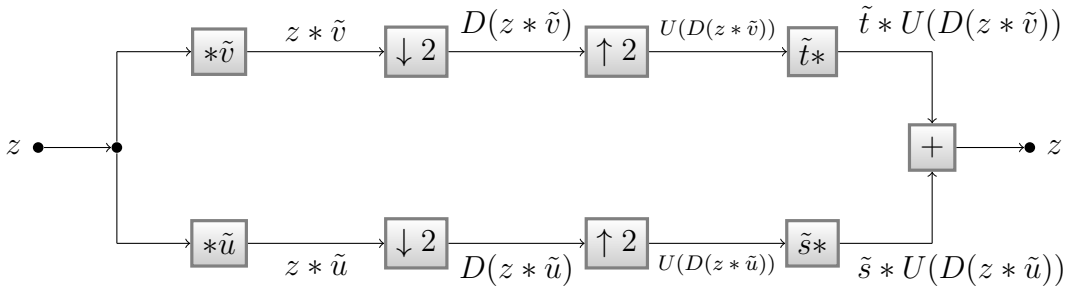

Figure 4.2: Reconstruction of z from the output of the left filter bank of Figure 4.1.

The operator U applied on a vector z does nothing else than doubling its size by putting a zero between two adjacent values. The first thing to note is that

$$D \circ U = Id,$$

but this is not true if we change the order of the operators: in fact, consider the example $z = (1, 1, -1, 1)$. Then:

$$D(z) = (1, -1), \qquad U(D(z)) = (1, 0, -1, 0) \neq z.$$

Therefore, it is not true that $U \circ D = Id$, so that D and U are only one-sided inverse operators. However, if we compare the operator $U \circ D$ with Remark 3.1.9, we observe that

$$U \circ D(z) = \frac{1}{2}(z + z^*). \tag{4.2}$$

In this case, suppose that we want to get back z as output of a filter bank whose left part is the same as in Figure 4.1. We can follow this filter bank with a right part built as shown in Figure 4.2: in this diagram, $s, t \in \ell^2(\mathbb{Z}_N)$ are unknown, but we are going to give necessary and sufficient conditions on u, v, s, t that will ensure us the perfect reconstruction of z.

Theorem 4.1.6. *Let $N = 2M$, with $M \in \mathbb{N}$, and $u, v, s, t \in \ell^2(\mathbb{Z}_N)$. Let $A(n)$ be the system matrix for u and v, for $n = 0, \ldots, N-1$. Then, we can reconstruct z as in Figure 4.2, i.e.*

$$\tilde{t} * U(D(z * \tilde{v})) + \tilde{s} * U(D(z * \tilde{u}))) = z \tag{4.3}$$

for all $z \in \ell^2(\mathbb{Z}_N)$ if and only if

$$A(n) \begin{pmatrix} \hat{s}_n \\ \hat{t}_n \end{pmatrix} = \begin{pmatrix} \sqrt{2} \\ 0 \end{pmatrix},$$

for all $n = 0, \ldots, N-1$. Moreover, if $A(n)$ is unitary, then $\hat{t}_n = \overline{\hat{v}_n}$ and $\hat{s}_n = \overline{\hat{u}_n}$.

Proof. By (4.2), we know that

$$U(D(z * \tilde{v})) = \frac{1}{2}(z * \tilde{v} + (z * \tilde{v})^*),$$

and the same happens if we replace v by u. Now, by the linearity of the DFT, Lemma 2.1.8, Proposition 3.1.2, and Lemma 3.1.8, we have that

$$[U(D(z * \tilde{v}))\hat{}\,]_n = \frac{1}{2}\Big(\hat{z}_n\overline{\hat{v}_n} + \hat{z}_{n+M}\overline{\hat{v}_{n+M}}\Big),$$

for every n, and similarly with v replaced by u. Hence, using again Lemma 2.1.8 and Proposition 3.1.2, we obtain:

$$\begin{aligned}
&\widehat{\Big[\tilde{t} * U(D(z * \tilde{v})) + \tilde{s} * [U(D(z * \tilde{u}))\Big]_n} \\
&= \frac{1}{2}\Big(\overline{\hat{t}_n}\Big(\hat{z}_n\overline{\hat{v}_n} + \hat{z}_{n+M}\overline{\hat{v}_{n+M}}\Big) + \overline{\hat{s}_n}\Big(\hat{z}_n\overline{\hat{u}_n} + \hat{z}_{n+M}\overline{\hat{u}_{n+M}}\Big)\Big) \\
&= \frac{1}{2}\Big(\hat{z}_n\Big(\overline{\hat{t}_n\hat{v}_n} + \overline{\hat{s}_n\hat{u}_n}\Big) + \hat{z}_{n+M}\Big(\overline{\hat{t}_n\hat{v}_{n+M}} + \overline{\hat{s}_n\hat{u}_{n+M}}\Big)\Big). \qquad (4.4)
\end{aligned}$$

By Fourier inversion (Theorem 2.1.6), there will be perfect reconstruction of the original signal z if and only if the expression in equation (4.4) agrees with $\hat{z}_n$, for all n. We claim that this happens if and only if

$$\hat{s}_n\hat{u}_n + \hat{t}_n\hat{v}_n = 2, \qquad (4.5)$$

$$\hat{s}_n\hat{u}_{n+M} + \hat{t}_n\hat{v}_{n+M} = 0. \qquad (4.6)$$

Indeed, replacing (4.5) and (4.6) in equation (4.4), we easily check that this is a sufficient condition to get perfect reconstruction. On the other hand, if we suppose that we have perfect reconstruction of z for all $z \in \ell^2(\mathbb{Z}_N)$, fix n and choose z such that $\hat{z}_n = 1$ and $\hat{z}_{n+M} = 0$. Then, equation (4.5) holds. On the other hand, choosing a different z such that $\hat{z}_n = 0$ and $\hat{z}_{n+M} = 1$, we deduce that equation (4.6) holds as well. If we now divide equations (4.5) and (4.6) by $\sqrt{2}$ and put them in matrix notation, we obtain (4.3).

Finally, suppose that $A(n)$ is unitary. Then, it is invertible and moreover, $A(n)^{-1} = \overline{A(n)^t}$. Solving equation (4.3), it turns out that $\hat{s}_n = \overline{\hat{u}_n}$ and $\hat{t}_n = \overline{\hat{v}_n}$. □

Remark 4.1.7. In Theorem 4.1.6 we saw that if $A(n)$ is unitary for some n, then $s_n = \overline{\hat{u}_n}$ and $\hat{t}_n = \overline{\hat{v}_n}$. This means that if $A(n)$ is unitary, then s, t are completely determined by u, v: indeed, by Proposition 3.1.2 and Proposition 2.1.6, we have:

$$s = \tilde{u}, \qquad t = \tilde{v}.$$

But we know, from Theorem 3.1.14, that $A(n)$ is unitary for all n if and only if $\{R_{2k}u\}_{k=0}^{M-1} \cup \{R_{2k}v\}_{k=0}^{M-1}$ is an orthonormal basis for $\ell^2(\mathbb{Z}_N)$, i.e., it is a first-stage wavelet basis. We conclude that we always have perfect reconstruction of a signal z for the filter bank shown in Figure 4.2 whenever u, v are first-stage wavelet basis generators.

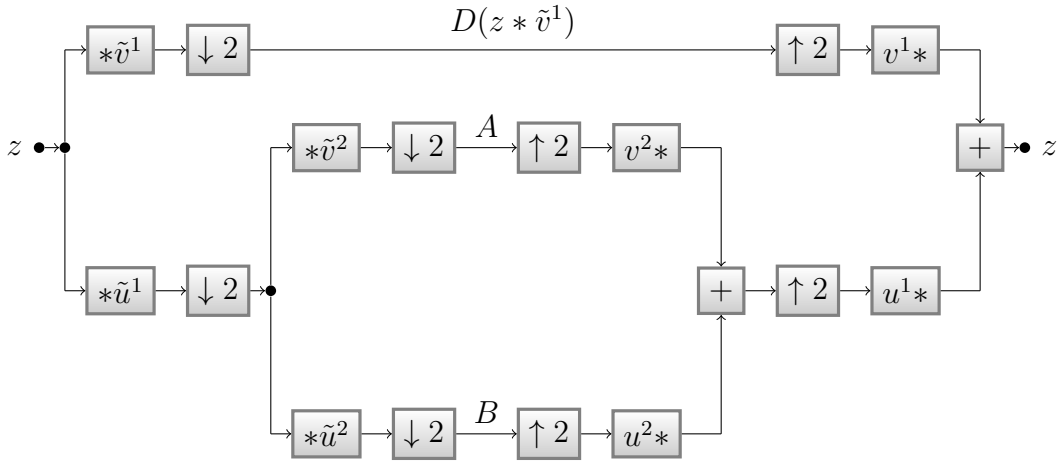

Figure 4.3: Second stage iteration of the Figure 4.2. Note that $A = D(D(z*\tilde{u}^1)*\tilde{v}^2)$, $B = D(D(z * \tilde{u}^1) * \tilde{u}^2) \in \ell^2(\mathbb{Z}_{N/4})$.

4.2 The iteration step

The filter bank shown in Figure 4.2 suggests a possibility for iteration in the following way: in the left part of this filter bank (which is the same as the one shown in Figure 4.1), we can apply the same procedure again. That is, we can pass the output of the top and bottom branches through two more filters, and then downsample again in each one of the new branches. In the right side, we can pass the signal from each one of the new branches through an upsampler, and finally a new filter, say s_2, t_2 to keep the previous notation. If the filters at this second stage are compatible, then we will have perfect reconstruction, and we can iterate this procedure as long as we are allowed to. We can apply the iteration at each one of the branches coming from the last step, but typically we will be interested in the branch coming from the convolution with a low-pass filter u, for the reasons stated at the beginning of this chapter. So, now we are going to study this iteration procedure.

Consider the filter bank from Figure 4.2, and write $u^1 = u$, $v^1 = v$. We shall assume that u^1, v^1 generate a first-stage wavelet basis, so that the system matrix $A(n)$ of u^1 and v^1 is unitary for all n. Then, Theorem 4.1.6 tells us that for perfect reconstruction, the filters in the fight half of Figure 4.2 must be u and v. Now suppose that we have an input signal $z \in \ell^2(\mathbb{Z}_N)$ for this diagram, and also assume that N is divisible by 4. At the first step, we get a pair of vectors, $D(z * \tilde{v}^1)$, $D(z * \tilde{u}^1) \in \ell^2(\mathbb{Z}_{N/2})$. We can assume that v^1 corresponds to the high-frequency part (although it is not necessary), and then leave $D(z*\tilde{v}^1)$ untouched. Now choose $u^2, v^2 \in \ell^2(\mathbb{Z}_{N/2})$ whose system matrix is unitary for all n. Now we can use these two new vectors to apply the filter shown in Figure 4.1, but this time using the vector $D(z * \tilde{u}^1)$ as input. The resultant filter bank of the whole process is shown in Figure 4.3.

Example 4.2.1. To illustrate this process, we can consider an easy example: the n-th order Haar transform (3.2.1). Suppose that $N > 3$ is divisible by 2^n, for some $n \in \mathbb{N}$. The Haar basis was defined as $\{R_{2k}u^1\}_{k=0}^{N/2-1} \cup \{R_{2k}v^1\}_{k=0}^{N/2-1}$, where

$$u^1 = \Big(1/\sqrt{2}, 1/\sqrt{2}, \underbrace{0, \ldots, 0}_{N-2}\Big), \qquad v^1 = \Big(1/\sqrt{2}, -1/\sqrt{2}, \underbrace{0, \ldots, 0}_{N-2}\Big).$$

For a signal $z \in \ell^2(\mathbb{Z}_N)$, recall (3.5) that we defined the tendency and the details of z by the Haar transform as $P_1'(z) = D(z * \tilde{u}^1)$ and $Q_1'(z) = D(z * \tilde{v}^1)$, respectively, and those vectors have length $N/2$. Now we can consider a new wavelet basis of the form $\{R_{2k}u^2\}_{k=0}^{N/4-1} \cup \{R_{2k}v^2\}_{k=0}^{N/4-1}$, where

$$u^2 = \Big(1/\sqrt{2}, 1/\sqrt{2}, \underbrace{0, \ldots, 0}_{N/2-2}\Big), \qquad v^2 = \Big(1/\sqrt{2}, -1/\sqrt{2}, \underbrace{0, \ldots, 0}_{N/2-2}\Big).$$

Next, we can apply a new wavelet transform to the vectors $P_1'(z)$, $Q_1'(z)$ (but in the iterations of the Haar transform, if we are looking for compression, we will only apply the transformation to $P_1'(z)$). We can repeat this process until the n-th step as long as N is divisible for 2^n.

Actually, p-th stage wavelets can be defined in terms of wavelet basis for different dimensions:

Definition 4.2.2. Let $N \in \mathbb{N}$ be divisible by 2^p, for $p > 1$. A p-th stage wavelet is a sequence of vectors $u^1, v^1, u^2, v^2, \ldots, u^p, v^p$ such that for every $m = 1, 2, \ldots, p$,

$$\{R_{2k}u^m\}_{k=0}^{N/2^m-1} \cup \{R_{2k}v^m\}_{k=0}^{N/2^m-1}$$

is a wavelet basis, or equivalently, the matrix

$$A_m(n) = \frac{1}{\sqrt{2}} \begin{pmatrix} \hat{u}_n^m & \hat{v}_n^m \\ \hat{u}_{n+N/2^m}^m & \hat{v}_{n+N/2^m}^m \end{pmatrix}$$

is unitary for all $n = 0, 1, \ldots, N/2^m - 1$.

Definition 4.2.3. In the same conditions as in Definition 4.2.2, define

$$x^1 = D(z * \tilde{v^1}) \in \ell^2(\mathbb{Z}_N),$$
$$y^1 = D(z * \tilde{u}^1) \in \ell^2(\mathbb{Z}_N),$$

and inductively, for $m = 2, \ldots, p$:

$$x^m = D(y_{m-1} * \hat{v}^m) \in \ell^2(Z_{N/2^m}),$$
$$y^m = D(y_{m-1} * \hat{u}^m) \in \ell^2(Z_{N/2^m}). \tag{4.7}$$

We say that the set of vectors $\{x_1, x_2, \ldots x_p, y_p\}$ is the output of the p-th stage wavelet filter bank.

The first thing to note is that by Theorem 4.1.6, we have perfect reconstruction of a signal $z \in \ell^2(\mathbb{Z}_N)$ from the output of a p-th stage wavelet filter bank. As we could expect, the total number of coefficients that define the output is

$$\frac{N}{2} + \frac{N}{4} + \cdots + \frac{N}{2^{p-1}} + \frac{N}{2^p} + \frac{N}{2^p} = N. \tag{4.8}$$

Next, we notice that we need no y_k, for $k = 1, \ldots, p-1$ in order to get a perfect reconstruction of z. Indeed, y_k, x_k are defined inductively in (4.7). Consider the first reconstruction step of a p-th stage wavelet filter bank. By the perfect reconstruction property, we have that

$$(U(y^p)) * u^p + (U(x^p)) * v^p = y_{p-1}.$$

We can repeat this process inductively: since we know y^{p-1} and x^{p-1} we can now obtain y^{p-2}, and so on. Eventually:

$$z = (U(y^1)) * u^1 + (U(x^1)) * v^1.$$

Definition 4.2.4. Consider a p-th stage wavelet filter bank F. The analysis phase of F will consist of those blocks whose last operation is a convolution followed by a downsampling. Contrarily, the reconstruction (or synthesis) phase will be formed by the blocks whose last operation is an upsampling followed by a convolution.

Precisely, the reconstruction phase of a signal z begins when z is totally decomposed by the filter bank, as shown in Figure 4.4.

Notice that we have defined the filter bank procedures recursively in Definition 4.2.3. Since this way of defining a p-stage wavelet does not show our original intent of constructing orthonormal bases for $\ell^2(\mathbb{Z}_N)$, working a bit, we are going to find an equivalent definition that leads to orthonormal bases.

Lemma 4.2.5. *Let $N = 2M, M \in \mathbb{N}$, $z \in \ell^2(\mathbb{Z}_N)$ and $x, y, w \in \ell^2(\mathbb{Z}_{N/2})$. Then:*

$$(D(z)) * w = D(z * U(w)), \tag{4.9}$$

$$U(x) * U(y) = U(x * y). \tag{4.10}$$

Proof. First, we prove equation (4.9). Note that, for every m, $w_m = U(w)_{2m}$. Therefore:

$$\begin{aligned}(D(z) * w)_n &= \sum_{m=0}^{N/2-1} D(z)_{n-m} w_m = \sum_{m=0}^{N/2-1} z_{2n-2m} U(w)_{2m} \\ &= \sum_{k=0}^{N-1} z_{2n-k} U(w)_k = (z * U(w))_{2n} = D(z * U(w))_n.\end{aligned}$$

In the last sum, we have added the odd-indexed values, since for k odd, $U(w)_k = 0$, so it does not affect the result. To prove equation (4.10), recall that $U(y)_m = 0$ if m is odd, and $U(y)_m = y_{m/2}$ if m is even. Then, we have that

$$(U(x) * U(y))_n = \sum_{m=0}^{N-1} U(x)_{n-m} U(y)_m = \sum_{k=0}^{N/2-1} U(x)_{n-2k}\, y_k. \tag{4.11}$$

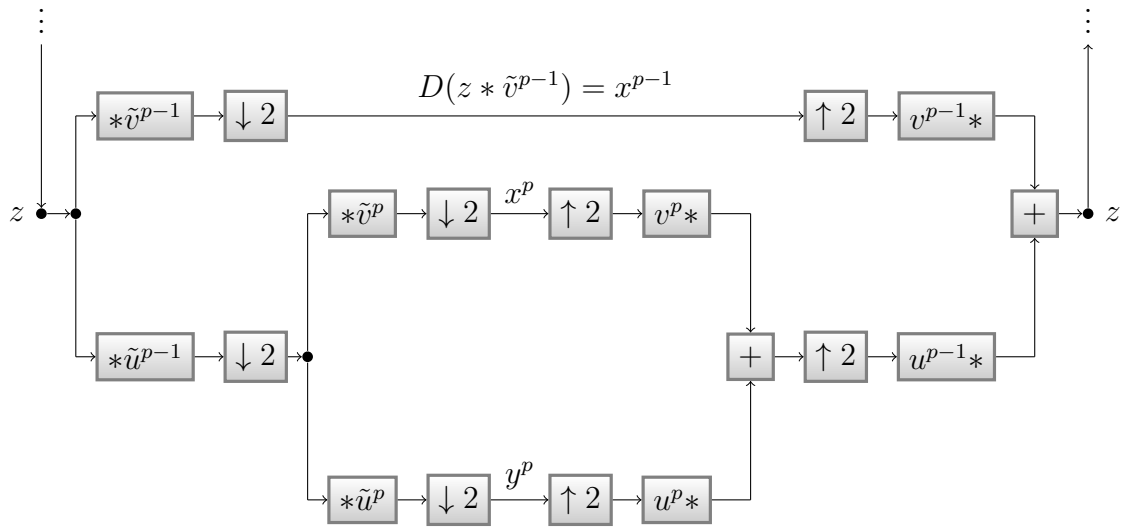

Figure 4.4: Ending of the analysis phase (left half) and beginning of the reconstruction phase (right half).

Now suppose that n is odd. Then $n - 2k$ is odd as well, and hence, $U(x)_{n-2k} = 0$ for all k. This yields

$$(U(x) * U(y))_n = 0 = U(x * y)_n.$$

On the other hand, suppose that n is even, i.e., $n = 2t$. In this case, $U(x)_{n-2k} = U(x)_{2t-2k} = x_{t-k}$, and using relation (4.11), we obtain:

$$(U(x) * (y))_n = \sum_{k=0}^{N/2-1} x_{t-k} y_k = (x * y)_t = U(x * y)_{2t} = U(x * y)_n.$$

□

Definition 4.2.6. Let $N \in \mathbb{N}$ be such that 2^k divides N. Then, for any $z \in \ell^2(\mathbb{Z}_N)$, we define $D^k(z)$ as the composition of D with itself k times, applied to z. Formally, we write $D = D^1$, and by induction, $D^k(z) = \big(D \circ D^{k-1}\big)(z)$, for $k > 1$. Also, $U^k(z)$ is defined in the same way.

Remark 4.2.7. Note that $D^k : \ell^2(\mathbb{Z}_N) \to \ell^2(\mathbb{Z}_{N/2^k})$ is given by

$$D^k(z)_n = z_{2^k n},$$

while $U^k : \ell^2(\mathbb{Z}_{N/2^k}) \to \ell^2(\mathbb{Z}_N)$ has the expression

$$U^k(z)_n = \begin{cases} w_{n/2^k}, & \text{if } 2^k \text{ divides } n, \\ 0, & \text{otherwise.} \end{cases}$$

We are now going to generalize Lemma 4.2.5 for D^k and U^k:

Proposition 4.2.8. *Suppose that $N \in \mathbb{N}$ is divisible by 2^k, $x, y, w \in \ell^2(\mathbb{Z}_{N/2^l})$, and $z \in \ell^2(\mathbb{Z}_N)$. Then*

$$D^k(z) * w = D^k\big(z * U^k(w)\big),$$
$$U^k(x * y) = U^k(x) * U^k(y).$$

Proof. The proof is done by induction on k. For $k = 1$, this result is just Lemma 4.2.5. Suppose that the statement is true for $k = l$. We need to prove it for $k = l + 1$. So, for $n \in \mathbb{Z}_{N/2^{l+1}}$, we have that

$$\begin{aligned}\big(D^{l+1} * w\big)_n &= \big(D\big(D^l(z)\big) * w\big)_n = \sum_{m=0}^{N/2^{l+1}-1} D\big(D^l(z)\big)_{n-m} w_m \\ &= \sum_{m=0}^{N/2^{l+1}-1} D^l(z)_{2n-2m} U(w)_{2m} = \sum_{t=0}^{N/2^l-1} D^l(z)_{2n-t} U(w)_t \\ &= \big(D^l(z) * U(w)\big)_{2n} = D\big(D^l(z) * U(w)\big)_n.\end{aligned}$$

By induction hypothesis, replacing w with $U(w)$, together with the last equality yield

$$D\big(D^l(z) * U(w)\big) = D\big(D^l(z * U^{l+1}(w))\big) = D^{l+1}\big(z * U^{l+1}(w)\big).$$

The second equality is proved in the same way, imitating the proof of Lemma 4.2.5. □

Finally, we are able to write a non-recursive definition of p-stage wavelet, equivalent to Definition 4.2.2.

Definition 4.2.9. Let $N \in \mathbb{N}$ be divisible by 2^p. Let $u^1, v^1, u^2, v^2, \ldots, u^p, v^p$ be vectors such that, for $k = 1, 2, \ldots, p$,

$$u^k, v^k \in \ell^2(\mathbb{Z}_{N/2^{k-1}}).$$

Define $f_1 = v_1$ and $g_1 = u_1$, and for $k = 2, 3, \ldots, p$ define inductively

$$f^k = g^{k-1} * U^{k-1}(v^k) \in \ell^2(\mathbb{Z}_N),$$
$$g^k = g^{k-1} * U^{k-1}(u^k) \in \ell^2(\mathbb{Z}_N).$$

We can observe that the general form of f^k, g^k is

$$f^k = u^1 * U(u^2) * U(u^3) * \cdots * U^{k-1}(v^k),$$
$$g^k = u^1 * U(u^2) * U(u^3) * \cdots * U^{k-1}(u^k).$$

Hence, all the convolutions involved in Definition 4.2.9 are computed using the filters u^k, except for the last convolution of g^k, which involves v^k. We will use the following three properties, which will not be proved since they easily follow from a few algebraic manipulations:

Proposition 4.2.10. *For $z, w \in \ell^2(\mathbb{Z}_N)$,*

1. $\widetilde{(z * w)} = \tilde{z} * \tilde{w}$,
2. $\widetilde{D(z)} = D(\tilde{z})$,
3. $\widetilde{U(z)} = U(\tilde{z})$.

The following expressions will be useful to express further results:

$$\tilde{f}^k = \tilde{g}^{k-1} k * U^{k-1}(\tilde{v}^k),$$
$$\tilde{g}^k = \tilde{g}^{k-1} * U^{k-1}(\tilde{u}^k).$$

We are now going to give a result that describes the output of the analysis phase of a p-th stage (recursive) wavelet filter bank as a set of (non-recursive) convolutions. The reconstruction phase will be described in a similar way.

Lemma 4.2.11. *Let $N \in \mathbb{N}$ be divisible by 2^p, $z \in \ell^2(\mathbb{Z}_N)$, and let $u^1, v^1, \ldots, u^p, v^p$ be such that*

$$u^k, v^k \in \ell^2(\mathbb{Z}_{N/2^{k-1}}),$$

for $k = 1, \ldots, p$. For the same values of k, Let x^k, y^k be as in Definition 4.2.3, and f^k, g^k as in Definition 4.2.9. Then

$$x^k = D^k\big(z * \tilde{f}^k\big),$$
$$y^k = D^k\big(z * \tilde{g}^k\big).$$

Proof. We will prove this result by induction on k, once again. For $k = 1$, it is trivial by just applying the definitions. Now suppose that the statement holds for $k = l$. Then, by the definition of x^{l+1} and Proposition 4.2.8,

$$x^{l+1} = D\big(y^l * \tilde{v}^{l+1}\big) = D\big(D^l\big(z * \tilde{g}^l\big) * \tilde{v}^{l+1}\big) = D \circ D^l\big(z * \tilde{g}^l * U^l\big(\tilde{v}^{l+1}\big)\big).$$

By the properties shown in Proposition 4.2.10, we conclude that

$$D^{l+1}\big(z * \tilde{g}^l * U^l\big(\tilde{v}^{l+1}\big)\big) = D^{l+1}\big(z * \tilde{f}^{l+1}\big).$$

The second equality is proved similarly as the first one, yielding

$$\begin{aligned} y^{l+1} &= D\big(y^l * \tilde{u}^{l+1}\big) = D\big(D^l\big(z * \tilde{g}^l\big) * \tilde{u}^{l+1}\big) = D \circ D^l\big(z * \tilde{g}^l * U^l\big(\tilde{u}^{l+1}\big)\big) \\ &= D^{l+1}\big(z * \tilde{g}^l * U^l\big(\tilde{u}^{l+1}\big)\big) = D^{l+1}\big(z * \tilde{g}^{l+1}\big). \end{aligned}$$

□

At this point we already have an expression for the vectors x_k, y_k, used to construct the output of the analysis phase of a p-th stage filter bank, and the important remark here is that we have obtained those expressions inductively. Such output is illustrated in the left half of Figure 4.5. The other half shows the reconstruction phase and it is going to be described in the next statement.

The output of the reconstruction phase will be described in what follows, but it will not be proved. The proof is a simple exercise which is done in a very similar way as the proof of Lemma 4.2.11.

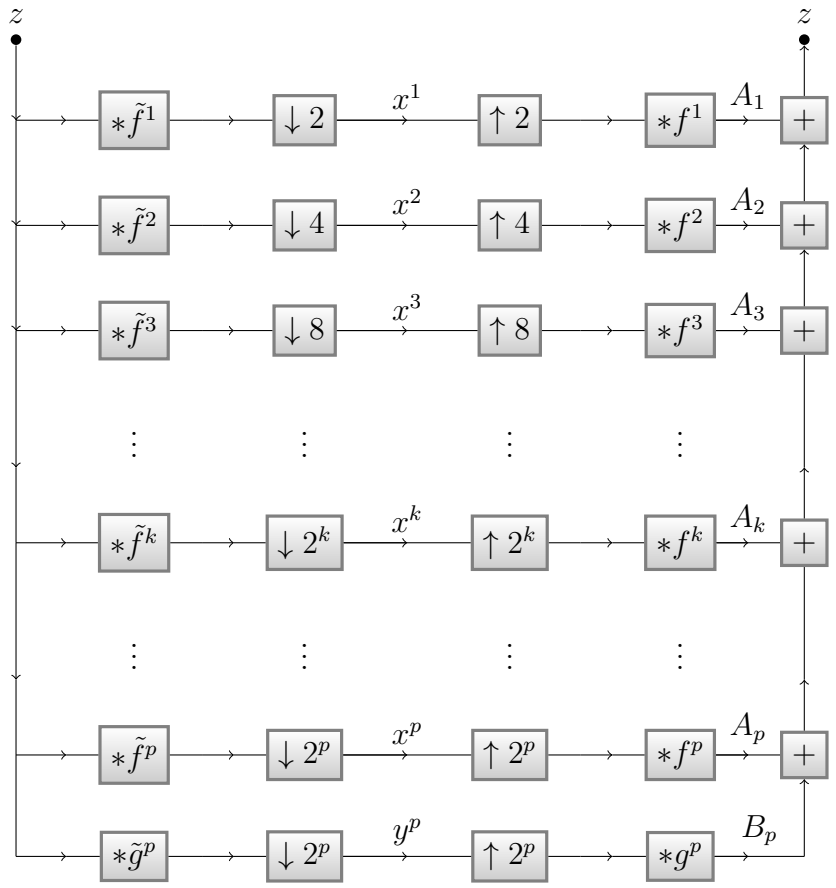

Figure 4.5: Diagram of a p-th stage wavelet filter bank (inductive). Note that for $l = 1, \ldots, p$, we have $A_l = f^l * U^l(x^l)$, and $B_p = g^p * U^p(y^p)$, by Lemma 4.2.12.

Lemma 4.2.12. *Let $N \in \mathbb{N}$ be divisible by 2^p. Consider a p-th stage filter bank sequence $u^1, v^1, \ldots, u^p, v^p$, and $f^1, \ldots, f^p, g^p$ to be as in Definition 4.2.9. If the input of the k-th branch ($1 \leq k \leq p$) of the reconstruction phase is x^k, then the output of such a branch is*

$$f^k * U^k(x^k).$$

Moreover, the output of the final branch of the reconstruction phase is

$$g^p * U^p(y^p).$$

Recall (Definition 4.2.3) that the output of the analysis phase of our filter bank is $x^1, x^2, \ldots, x^p, y^p$. By Lemma 4.2.11, we have that if $1 \leq k \leq p$, then

$$x_n^k = D^k\big(z * \tilde{f}^k\big) = \big(z * \tilde{f}^k\big)_{2^k n} = \langle z, R_{2^k n} f^k \rangle,$$

for all $n = 0, 1, \ldots, N/2^k - 1$. Similarly, when $k = p$,

$$y_n^p = D^p\big(z * \tilde{g}^p\big) = \big(z * \tilde{g}^p\big)_{2^p n} = \langle z, R_{2^p n} f^p \rangle.$$

Also, by equation (4.8), the total number of components of the analysis phase output is N. We may wonder if this output is actually formed by the coefficients of z with respect to an orthonormal basis. We will see that certainly, it is, and the proof of this fact is the main result of this section. To begin, let us see a new definition and a couple of lemmas that will help us throughout the developing of the main theorem.

Definition 4.2.13. Let $N \in \mathbb{N}$ be divisible by 2^p, and let $\mathcal{B}$ be a set of the form

$$\{R_{2k}f^1\}_{k=0}^{N/2-1} \cup \{R_{4k}f^2\}_{k=0}^{N/4-1} \cup \cdots \cup \{R_{2^pk}f^p\}_{k=0}^{N/2^p-1} \cup \{R_{2^pk}g^p\}_{k=0}^{N/2^p-1}, \tag{4.12}$$

with $f^1, f^2, \ldots, f^p, g^p \in \ell^2(\mathbb{Z}_N)$. If $\mathcal{B}$ forms an orthonormal basis for $\ell^2(\mathbb{Z}_N)$, we refer to it as a p-th stage wavelet basis.

Thus, we now need to prove that the vectors $f^1, f^2, \ldots, f^p, g^p$ obtained by Definition 4.2.9 generate a p-th stage wavelet basis.

Lemma 4.2.14. *Let $N \in \mathbb{N}$ be divisible by 2^t. Let $g^{t-1} \in \ell^2(\mathbb{Z}_N)$, and suppose that the set*

$$\{R_{2^{t-1}k}g^{t-1}\}_{k=0}^{N/2^{t-1}-1}$$

is an orthonormal set with $N/2^{t-1}$ elements. Suppose that $u^t, v^t \in \ell^2(\mathbb{Z}_{N/2^{t-1}})$, and the system matrix $A_t(n)$ of u^t, v^t is unitary, for all $n = 0, 1, \ldots, N/2^t - 1$. If we define

$$f^t = g^{t-1} * U^{t-1}(v^t), \qquad g^t = g^{t-1} * U^{t-1}(u^t), \tag{4.13}$$

then the set $\mathcal{B}_t = \{R_{2^tk}f^t\}_{k=0}^{N/2^t-1} \cup \{R_{2^tk}g^t\}_{k=0}^{N/2^t-1}$ is an orthonormal set with $N/2^{t-1}$ elements.

Example 4.2.15. Before we prove this theorem, we shall see a simple example illustrating this fact. Consider $N = 8$ and the Haar wavelet, with generators

$$u = u^1 = \left(1/\sqrt{2}, 1/\sqrt{2}, 0, 0, 0, 0, 0, 0\right),$$
$$v = v^1 = \left(1/\sqrt{2}, -1/\sqrt{2}, 0, 0, 0, 0, 0, 0\right).$$

Since $8 = 2^3$, following the notation of Lemma 4.2.14, we consider $t = 2$ and define $g^1 = u^1$. Clearly, $\{R_{2k}g^1\}_{k=0}^3$ is an orthonormal basis with 4 elements, since it is the subspace spanned by the translations of the vector u^1 (Lemma 4.2.17), which is a generator of the Haar wavelet. We will consider u^2 and v^2 to be the generators of the Haar wavelet, when $N = 4$. This ensures that the system matrix $A_2(n)$ of u^2, v^2 is unitary, for all n. Finally, a simple computation shows that

$$f^2 = g^1 * U(v^2) = \frac{1}{2}(1, 1, -1, -1, 0, 0, 0, 0),$$
$$g^2 = g^1 * U(u^2) = \frac{1}{2}(1, 1, 1, 1, 0, 0, 0, 0).$$

Trivially, the set $\{R_{4k}f^2\}_{k=0,1} \cup \{R_{4k}g^2\}_{k=0,1}$ is an orthonormal set with 4 elements, as stated in Lemma 4.2.14.

Proof of Lemma 4.2.14. By Lemma 3.1.4 and the orthogonality of the set (4.13), we have that

$$\left(g^{t-1} * \tilde{g}^{t-1}\right)_{2^{t-1}k} = \langle g^{t-1}, R_{2^{t-1}k} g^{t-1}\rangle = \begin{cases} 1, & \text{if } k = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{4.14}$$

Now, because $A_t(n)$ is unitary for all n, we can apply Theorem 3.1.14 and obtain that the set

$$\{R_{2k} v^t\}_{k=0}^{N/2^t-1} \cup \{R_{2k} u^t\}_{k=0}^{N/2^t-1}$$

is an orthonormal basis for $\ell^2(\mathbb{Z}_{N/2^{t-1}})$. Hence, by Lemma 3.1.4,

$$(v^t * \tilde{v}^t)_{2k} = \langle v^t, R_{2k} v^t\rangle = \begin{cases} 1, & \text{if } k = 0, \\ 0, & \text{otherwise,} \end{cases}$$

and the same expression is true if we replace v^t by u^t. Moreover, $(v^t * \tilde{u}^t)_{2k} = \langle v^t, R_{2k} u^t\rangle = 0$, for every k. Again, by Lemma 3.1.4, along with the definition of the f^k's, the commutativity and the associativity of the convolution, and Proposition 4.2.8, we can write the inner products in the following way:

$$\begin{aligned} \langle f^t, R_{2^t k} f^t\rangle &= \left(\left(f^t * \tilde{f}^t\right)_{2^t k} = g^{t-1} * U^{t-1}(v^t) * \tilde{g}^{t-1} * U^{t-1}(\tilde{v}^t)\right)_{2^t k} \\ &= \left(\left(g^{t-1} * \tilde{g}^{t-1}\right) * \left(U^{t-1}(v^t * \tilde{v}^t)\right)\right)_{2^t k} \\ &= \sum_{m=0}^{N-1} \left(g^{t-1} * \tilde{g}^{t-1}\right)_{2^t k - m} \left(U^{t-1}(v^t * \tilde{v}^t)\right)_m. \end{aligned}$$

We observe that $U^{t-1}(v^t * \tilde{v}^t)_m = (v^t * \tilde{v}^t)_j$ whenever $m = 2^{t-1} j$, and 0 otherwise. Thus,

$$\langle f^t, R_{2^t k} f^t\rangle = \sum_{m=0}^{N/2^{t-1}-1} \left(g^{t-1} * \tilde{g}^{t-1}\right)_{2^t k - 2^{t-1} m} (v^t * \tilde{v}^t)_m.$$

And by equation (4.14), we have

$$\left(g^{t-1} * \tilde{g}^{t-1}\right)_{2^t k - 2^{t-1} m} = \left(g^{t-1} * \tilde{g}^{t-1}\right)_{2^{t-1}(2k-m)} = \begin{cases} 1, & \text{if } m = 2k, \\ 0, & \text{if } m \neq 2k, m \in \mathbb{Z}_{N/2^{t-1}}. \end{cases}$$

Hence,

$$\langle f^t, R_{2^t} k f^t\rangle = (v^t * \tilde{v}^t)_{2k} = \begin{cases} 1, & \text{if } k = 0, \\ 0, & \text{if } k = 1, 2, \ldots, N/2^t - 1. \end{cases}$$

Therefore, the set $\{R_{2^t k} f^t\}_{k=0}^{N/2^t-1}$ is orthonormal. By the same procedure, but taking g^t in place of f^t, we obtain that the set $\{R_{2^t k} f^t\}_{k=0}^{N/2^t-1}$ is also orthonormal. Moreover, in the process we obtain the equality

$$\langle f^t, R_{2^t k} g^t\rangle = (v^t * \tilde{u}^t)_{2k} = 0,$$

for all k. This equality, along with the fact that $\langle f^t, R_{2^t k} f^t\rangle = 0$ for all k, we obtain (after a few easy manipulations):

$$\langle R_{2^t k} f^t, R_{2^t j} g^t\rangle = 0, \text{ for all } k, j.$$

Hence, the set $\mathcal{B}_t$ defined in the statement is orthonormal. □

Actually, the splitting method shown in Lemma 4.2.14 is deeper in the sense that we get decompositions of an orthonormal basis in the direct sum of two orthonormal subspaces, each with the same amount of elements.

Definition 4.2.16. Let X be an inner product space, and let U, V be subspaces of X. Suppose that $U \perp V$. Define the direct sum of U and V as

$$U \oplus V = \{u + v : u \in U, v \in V\}.$$

In particular, if $X = U \oplus V$, every element $x \in X$ can be written as $x = u + v$, with $u \in U$ and $v \in V$.

Lemma 4.2.17. *With the same assumptions as in Lemma 4.2.14, if we define the spaces*

$$\begin{aligned} V_{-t+1} &= \operatorname{span}\{R_{2^{t-1}k} g^{t-1}\}_{k=0}^{N/2^{t-1}-1}, \\ W_{-t} &= \operatorname{span}\{R_{2^t k} f^t\}_{k=0}^{N/2^t-1}, \\ V_{-t} &= \operatorname{span}\{R_{2^t k} g^t\}_{k=0}^{N/2^t-1}, \end{aligned}$$

then $V_{-t+1} = V_{-t} \oplus W_{-t}$.

Proof. By Lemma 4.2.14, every basis element of V_{-t} is orthogonal to every basis element of W_{-t}. By linearity, it follows that every element of V_{-t} is orthogonal to every element of W_{-t}. This proves the orthonormality of those spaces. Next, we need to prove that they actually are subspaces. Note that, for $k = 0, 1, \ldots, N/2^t - 1$,

$$\begin{aligned} (R_{2^t k} g^t)_n = g^t_{n-2^t k} &= g^{t-1} * (U^{t-1}(u^t))_{n-2^t k} \\ &= \sum_{m=0}^{N-1} g^{t-1}_{n-2^t k-m} (U^{t-1}(u^t))_m. \end{aligned}$$

Since $(U^{t-1}(u^t))_m = u^t_{m/2^{t-1}}$ if $2^{t-1} | m$, and 0 otherwise, the sum over m gets reduced by only keeping the indexes of the form $2^{t-1} j$:

$$(R_{2^t k} g^t)_n = \sum_{j=0}^{N/2^{t-1}-1} g^{t-1}_{n-2^t k-2^{t-1} j} u^t_j = \sum_{j=0}^{N/2^{t-1}-1} u^t_j (R_{2^{t-1}(j+2k)} g^{t-1})_n.$$

This is true for all n. Hence, $R_{2^t k} g^t = \sum_{j=0}^{N/2^{t-1}-1} u^t_j (R_{2^{t-1}(j+2k)} g^{t-1})$. By the same procedure, we obtain that $R_{2^t k} f^t = \sum_{j=0}^{N/2^{t-1}-1} u^t_j (R_{2^{t-1}(j+2k)} g^{t-1})$. Thus, the vectors $R_{2^t k} g^t$ and $R_{2^t k} f^t$ belong to V_{-t+1}, since they are linear combinations of translates of g^{t-1} by integer multiples of 2^{t-1}. Therefore, all the basis elements from V_{-t}, W_{-t} belong to V_{-t+1}, and so are the elements of their spans. Since V_{-t} and W_{-t} have dimension $N/2^t$, then $V_{-t} \oplus U_{-t}$ has dimension $N/2^{t-1}$, which is the same as the dimension of V_{-t+1}. We conclude $V_{-t} \oplus U_{-t} = V_{-t+1}$. □

We first observe that the notation of the spaces is given by negative indexes. This way we get the spaces increasing with the index (i.e., $V_{-t} \subset V_{-t+1}$). Moreover, as we could observe in Example 4.2.15, the spaces obtained by splitting V_{-t+1} into two pieces are actually subspaces. We shall now see that Lemma 4.2.17 is the main step to prove that the output of a p-th stage wavelet filter bank yields the coefficients of z with respect to a p-th stage wavelet basis.

Theorem 4.2.18. *In the context of Definition 4.2.9, we have that (4.12) is a p-th stage wavelet basis.*

Proof. Since $f^1 = v^1$ and $g^1 = u^1$, by Theorem 3.1.14 and the hypotheses, we have that

$$\{R_{2k} f^1\}_{k=0}^{N/2-1} \cup \{R_{2k} g^1\}_{k=0}^{N/2-1}$$

is orthonormal. By induction, using Lemma 4.2.14, we obtain that the set of vectors $\{R_{2^l k} f^l\}_{k=0}^{N/2^l-1}$ is orthonormal for each $l = 1, 2, \ldots, p$, and $\{R_{2^p k} g^p\}_{k=0}^{N/2^p-1}$ is orthonormal as well. Thus, to prove the orthonormality of (4.12), we only need to check the orthogonality of elements in different sets. Consider some $R_{2^l k} f^l$ and $R_{2^m j} f^m$, and assume, without loss of generality, that $m < l$. Then, by Lemma 4.2.17, we have that

$$R_{2^l k} f^l \in W_{-l} \subset V_{-l+1} \subset \cdots \subset V_{-m},$$

and $R_{2^m j} f^m \in W_{-m}$. Again, by Lemma 4.2.17, $V_{-m} \perp W_{-m}$, so the vectors $R_{2^l k} f^l$ and $R_{2^m j} f^m$ are orthogonal. Similarly, for every $m \leq p$, $R_{2^p k} g^p$ belongs to the space $V_{-p} \subset V_{-m}$, and hence, it is orthogonal to any vector $R_{2^l k} f^l \in W_{-l}$. □

Summarizing, if we join all the results from this section, we will be able to implement any p-th stage wavelet transform (when possible), iterating a first-stage wavelet basis: if such a basis is generated by two vectors $u = u^1$, $v = v^1$, we consider the same wavelet basis for $\ell^2(\mathbb{Z}_{N/2})$, if it is possible. This basis will be generated by two vectors u^2, v^2, of length $N/2$. As long as 2^k divides N, we can find the vectors u^k, v^k. With this process we obtain the vectors $u^1, v^1, u^2, v^2, \ldots, u^p, v^p$ appearing in Definition 4.2.9 and hence, construct a new orthonormal basis for $\ell^2(\mathbb{Z}_N)$, opening the possibility to obtain even better properties than a first-stage wavelet. In the next section we shall compare a couple of different stages of the Shannon real wavelet and Daubechies D6 wavelet.

4.3 Examples comparing first-stage and p-th stage wavelet basis

As we mentioned earlier on, p-th stage wavelet basis provide better analysis and compression than the original first-stage wavelets. We will see this fact in terms of the coefficients. Concretely, the iteration step will yield less relevant coefficients, which is a good thing if we think about compression.

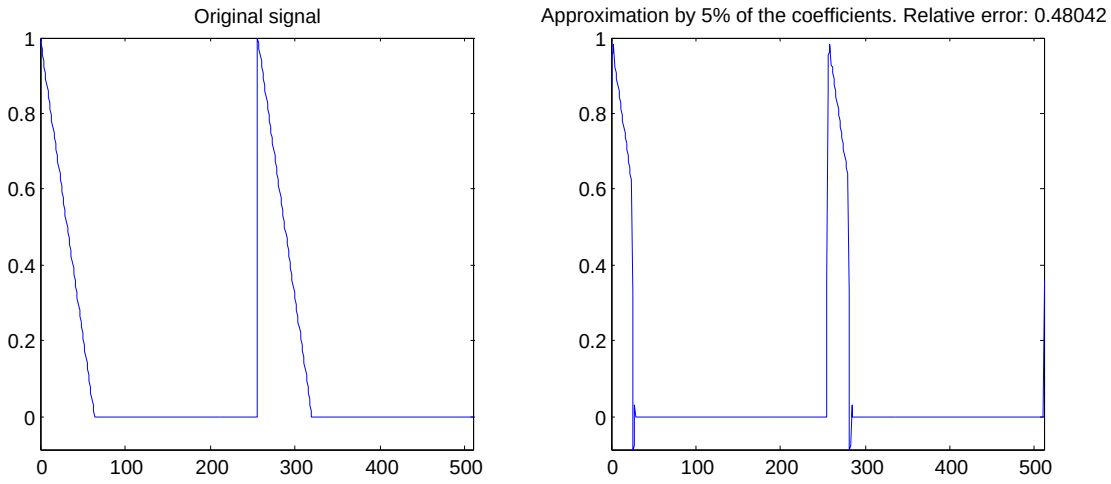

Figure 4.6: Signal from Example 4.3.1 and its first-stage approximation in Daubechies D6 basis.

Example 4.3.1. The first example will be using the signal from Example 5.2.3. We are going to keep the 5% of the coefficients every time (see Section 5.2), and observe the huge improvement whenever we use the p-th stage wavelet. Figure 4.6 shows the signal and its approximation by the first-stage Daubechies D6 wavelet. We observe that the relative error is very high, and hence, the approximation is very poor. In fact, if we take a look at the coefficients of the first-stage Daubechies D6 basis, shown in Figure 4.7, we find many of them which are significantly big. Moreover, as we iterate this wavelet basis, the most significative coefficients are just a few. We have compressed the signal keeping only 5% of the coefficients, i.e., around 25 coefficients. Therefore, we could already predict the bad compression using the first-stage basis just by looking at the correspondent coefficients (Figure 4.7, (a)). Indeed, as there are many high coefficients, keeping only the highest 25 will lead to an important loss of information. Finally, we have the approximation by the 4-th stage Daubechies D6 basis in Figure 4.8. In this case, the relative error is lower than 0.06, yielding a good compression: the improvement is evident.

Example 4.3.2. Consider, for $N = 512$, the signal $x_n = \sin(x/175) - 3\cos(x/100)$, used as an example of the compression yielded by the Shannon real basis (Example 3.2.10). We noticed that keeping 30% of the coefficients caused the loss of those related to the generator v. Although, when considering higher stages of this basis, the number of relevant coefficients decreases, in the same way as in the previous example. As we can see in Figure 4.9, the percentage of relevant coefficients does not exceed 25%. Indeed, in this case, the relative error is very small, around 0.003. But moreover, this compression is even better than that one made by the Fourier basis. In Example 5.2.1, we discuss the characteristics of its Fourier compression.

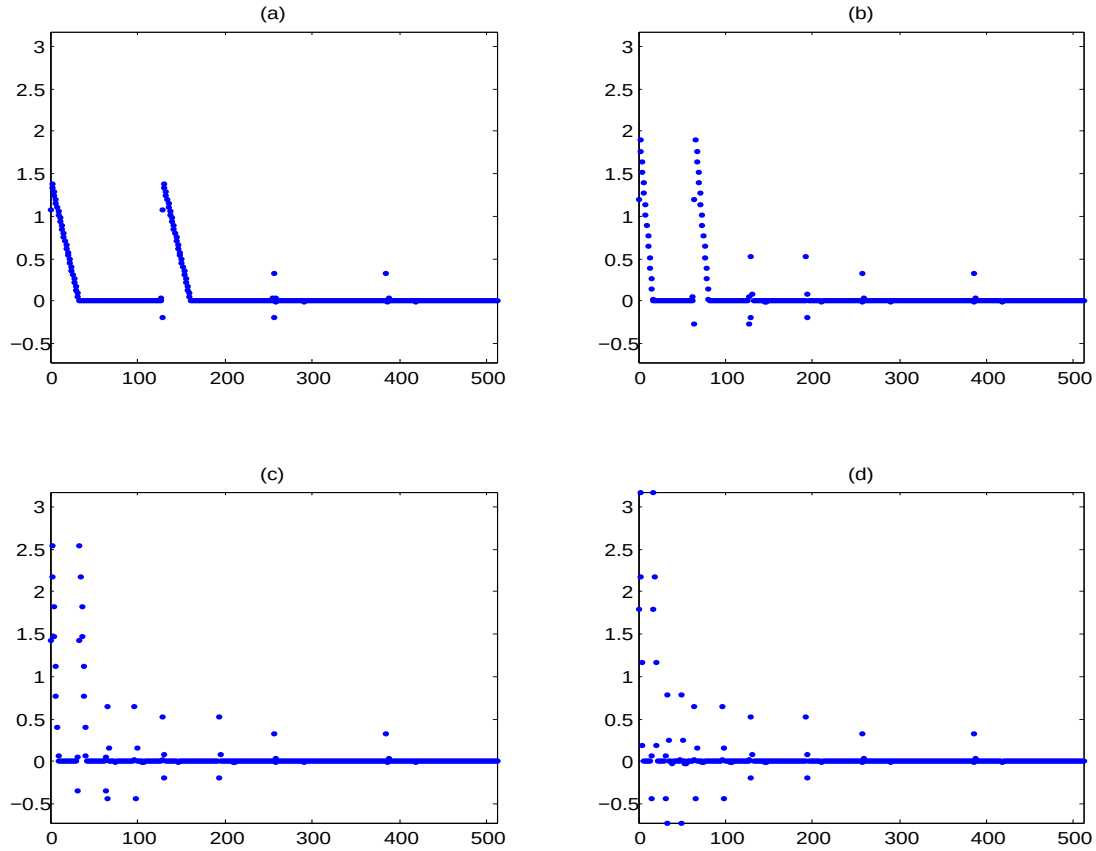

Figure 4.7: Coefficients in the p-th stage Daubechies D6 basis of the signal from Example 4.6, for $p = 1, \ldots, 4$, respectively.

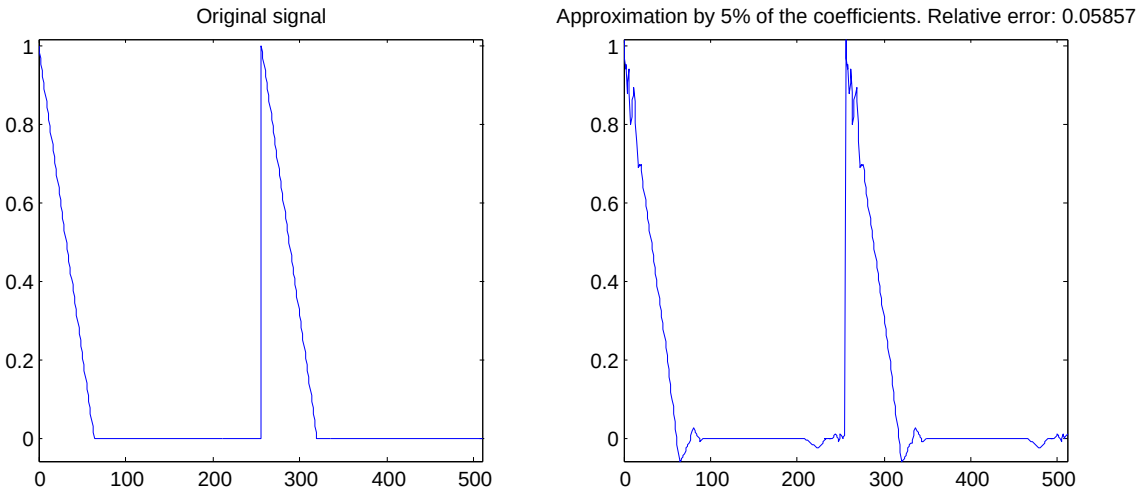

Figure 4.8: Signal from Example 4.3.1 and its 4-th stage approximation in Daubechies D6 basis.

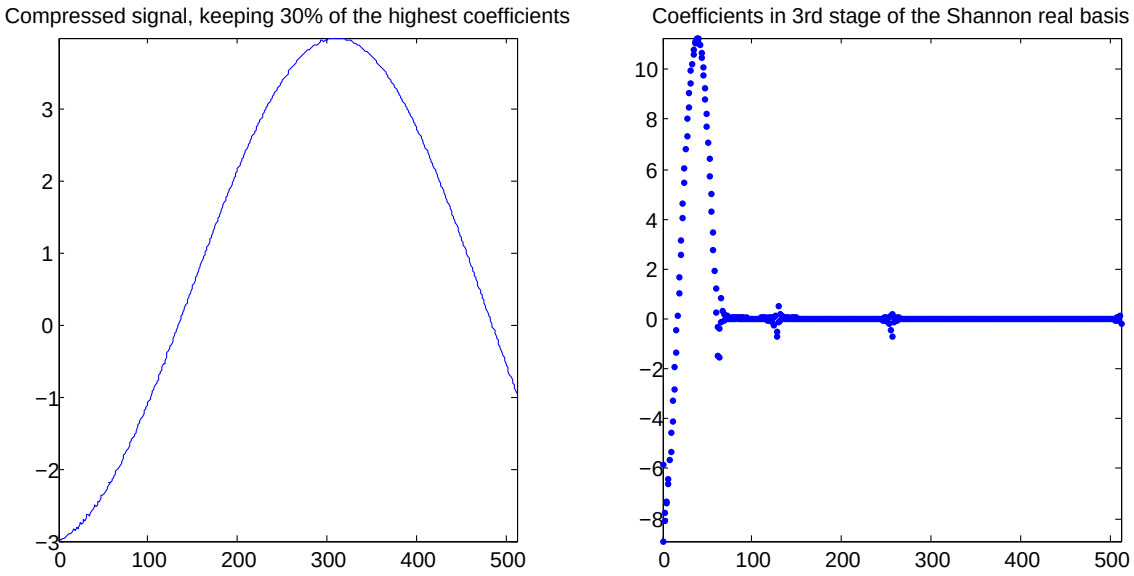

Figure 4.9: Compression of the signal from Example 4.3.2 and the coefficients of its expansion in the 3rd iteration of the Shannon basis.

Chapter 5

Applications and comparisons

This whole chapter will be devoted to analyze the different compressions yielded by the wavelet basis we have presented. As we may expect, there is no wavelet that works well for absolutely all the signals. In fact, the relative error function will show better efficiency for a certain wavelet basis $\mathcal{B}$ if the signal can be constructed by a linear combination of few vectors of $\mathcal{B}$. In fact, the Fourier transform does quite well at compressing sinusoidal signals.

5.1 Wavelet localization

First of all, we are going to focus in the wavelet basis we have presented and study their characteristics. Because all our wavelets are different, they may also have different properties (and actually, they will). We are also going to consider the DFT not as a wavelet, but just as a linear transformation, which can also be used for our purpose of compressing data. The main points we are going to compare between different linear transformations are space and frequency localization. After doing that, suppose we want to compress a signal z. The comparison we are going to do will allow us to choose the best linear transformation to be applied to z in such a way that the compression rate is high and the quality is good enough. To do this, the ideal procedure is to draw the histograms containing the magnitudes (or complex modulus) of the vectors $u, v, \hat{u}, \hat{v}$ (in the case of a wavelet basis), for different basis, and compare them. The first example is, of course, the Euclidean basis, which can be done at the same time as the DFT, since a simple computation yields that

$$\hat{\hat{z}}_n = z_{-n},$$

for all n. We are only interested in the magnitudes of the basis vectors, so changing their order will not make any difference. We can already expect those two histograms to be "degenerated", if we take Lemma 3.1.5 into account. Indeed, Figure 5.1 shows that both histograms are as expected, when $N = 256$.

Let us do one step further: instead of considering perfectly localized vectors (with only one nonzero component), we allow our basis to have two nonzero components.

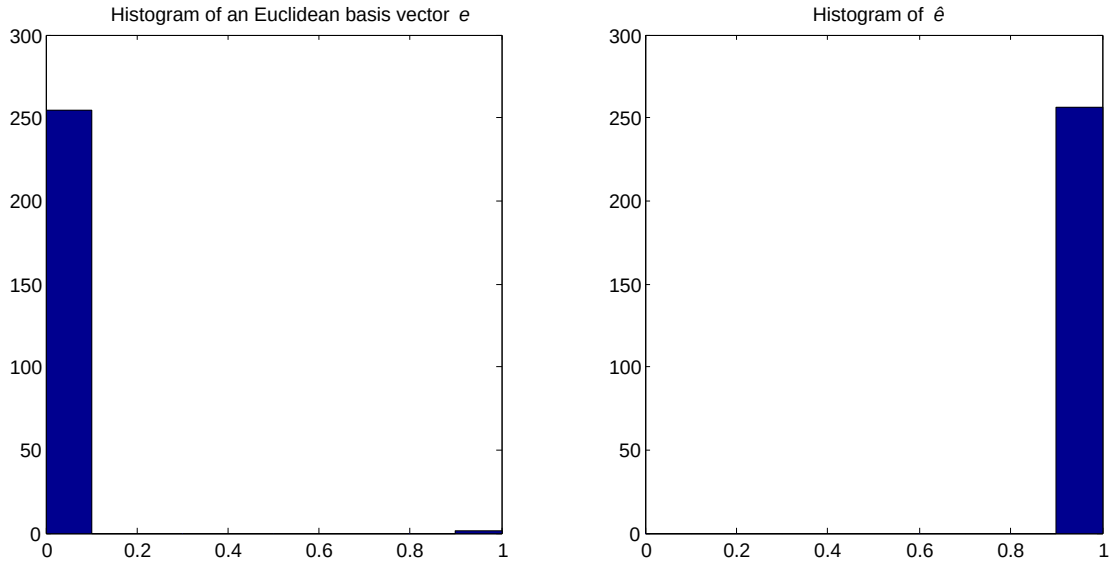

Figure 5.1: Both histograms show that having the best spatial localization possible makes a vector be non-frequency localized at all.

A wavelet basis with this property is the Haar basis (or also, Daubechies' D2). Recall that in the Haar basis,

$$u = (1/\sqrt{2}, 1/\sqrt{2}, 0, \ldots, 0),$$
$$v = (1/\sqrt{2}, -1/\sqrt{2}, 0, \ldots, 0).$$

The decomposition of $\hat{u}$ in real and imaginary part is shown in Figure 5.2 (with normalization). We can already see an improvement by just adding one nonzero component to the basis generators. Specifically, putting an extra nonzero component has caused the vectors $\hat{u}$, $\hat{v}$ to have non-constant and small magnitudes (all of them components are lower than $1/10$ in modulus, as we can appreciate in the histogram of Figure 5.3), and moreover, there are relatively small[1] coefficients near a certain point n_0, in this case the center of the x axis. Note that we need not show the histogram of v, since the magnitudes of its coefficients will be the same as the ones shown in Figure 5.3.

The next basis to deal with is the Shannon real wavelet. Recall that the generators u and v were defined in terms of $\hat{u}$ and $\hat{v}$ (Definition 3.2.7). As we observed in Figure 3.2, this basis is nicely localized in space (most of its components are small), and several components of its DFT vanish, by definition. Figure 5.4 shows the histograms for v and $\hat{v}$, which illustrates how nice this basis is.

Finally, we consider the vector u to be a generator of the Daubechies D6 wavelet basis. By definition, we took u to have only 6 nonzero entries, so the spatial localization property was covered. This can be viewed as an extension of the Haar

[1] That is, significantly small with respect to the highest coefficient.

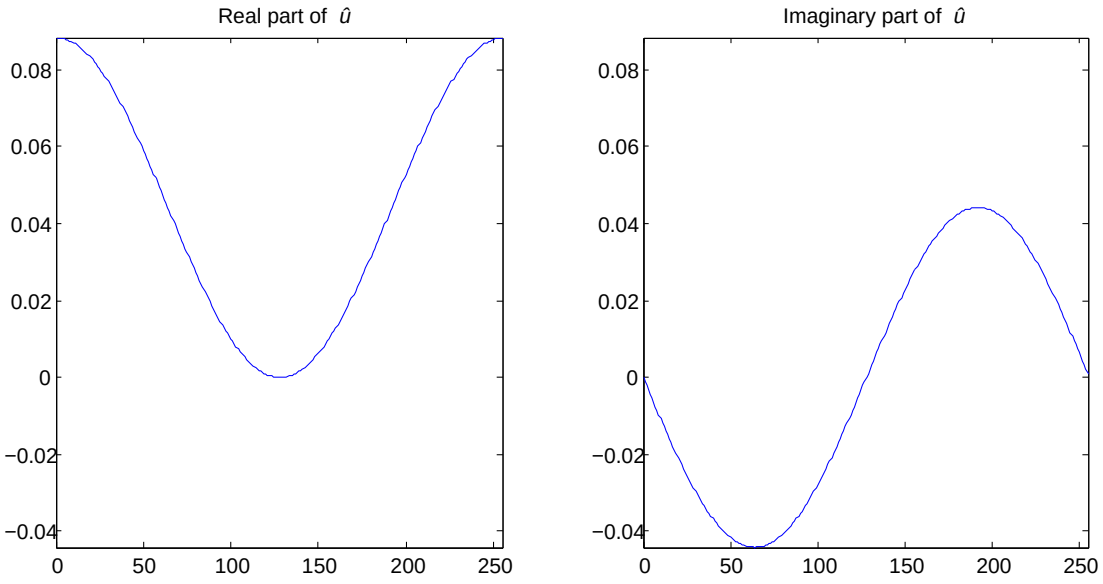

Figure 5.2: Decomposition of u in real and imaginary parts, for u the first generator of the Haar basis.

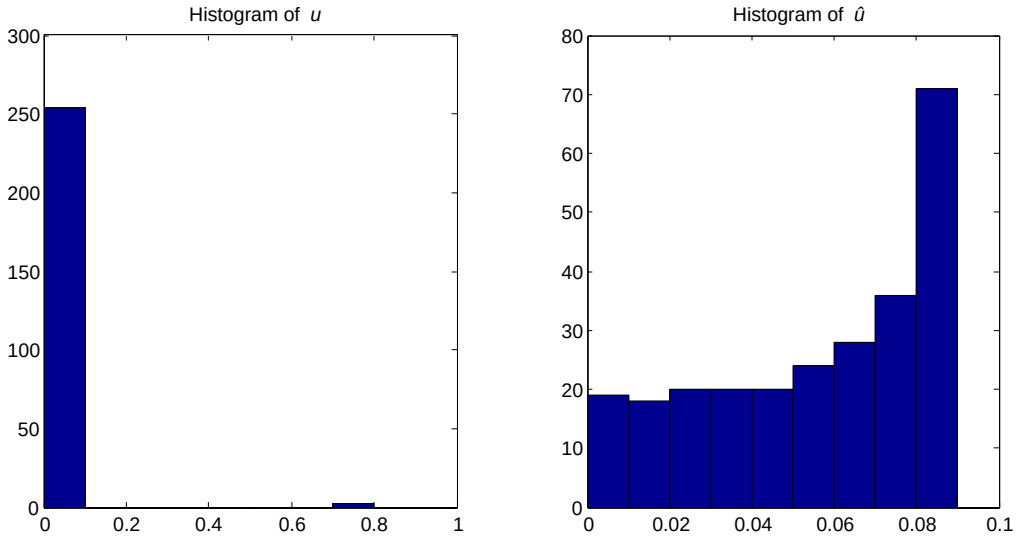

Figure 5.3: Histogram of the Haar basis generators.

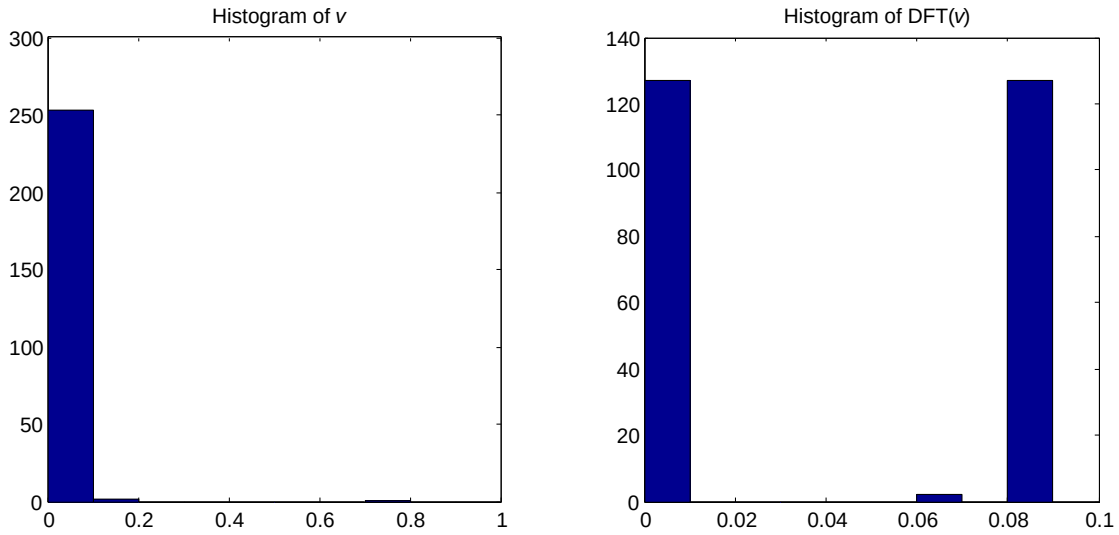

Figure 5.4: Histogram of the Shannon real basis generators.

wavelet in the sense that we allow the basis generators to have some more nonzero components. The histogram of u and $\hat{u}$ appears in Figure 5.5.

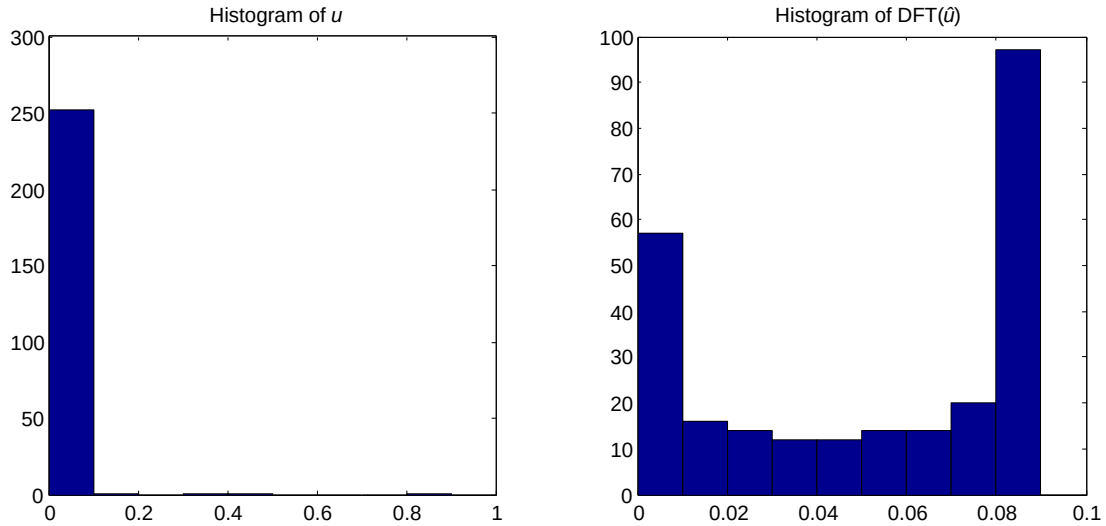

Figure 5.5: Histogram of the Daubechies D6 wavelet generator u.

5.2 Data compression in 1 dimension

In this section we will consider several signals, and compress them with the different linear transformations presented in Section 5.1. The compression will be done as follows: for a vector $z \in \ell^2(\mathbb{Z}_N)$, and a basis $\mathcal{B} = \{v_j\}_{j=0}^{N-1}$ of $\ell^2(\mathbb{Z}_N)$, the expansion

of z in terms of $\mathcal{B}$ will be

$$[z]_{\mathcal{B}} = \sum_{j=0}^{N-1} \langle z, v_j \rangle v_j.$$

Next, we choose $0 < K < N$, which will be the number of coefficients to be used to approximate z. More precisely, we sort the coefficients $\langle z, v_j \rangle$ in order of magnitude, and we keep the K highest, while the rest are set to zero. This ensures that we are keeping the most relevant coefficients to obtain a reconstruction of z: indeed, since wavelet basis are orthonormal, the norm is preserved by those transformations (and by their inverses, of course). Hence, if we approximate the vector z by w, we want the norm $\|z - w\|$ to be small. Dropping small coefficients of $[z]_{\mathcal{B}}$ will make the approximation be accurate (in norm). Although, instead of taking the K highest coefficients, where $K < N$, we will be working with the $k\%$ largest coefficients, with $0 \leq k \leq 100$.

We will also be interested in the energy ratio of each one of the basis we work with. To be precise, we define the relative error made in approximating z by w as the ratio $\|z - w\|/\|z\|$. Thus, we can consider the relative error as function of k (the percentage of coefficients we keep). Note that the only element depending of k in the relative error is the vector w. Having such functions plotted will show which wavelet is the best (for each case), and as we are going to see, there is no wavelet better than the others, in the sense that the characteristics of the signal to analyze will determine the most suitable wavelet to use (in matter of compression).

Example 5.2.1. Consider the signal x from Example 4.3.2. This signal has a jump at $n = 512$ (recall that for $n > 512$ we consider the signal extended periodically), which makes the Fourier basis not to yield a good compression whenever we keep a certain number of coefficients. This is because such a basis comes from smooth functions, so there may be trouble if our signal contains big jumps: the analogy with the continuous case is the jump of a function which makes it not to be continuous, and hence, not smooth. Figure 5.6 shows the approximation of x by 30% of the coefficients, and we observe an interesting phenomenon around this jump: the signal has not been reconstructed properly. In the continuous case, this is known as Gibbs phenomenon, (see [1, Section 4.8]) and it essentially asserts that we cannot have convergence of the Fourier transform of a function f around the points where f has a jump. Therefore, we have found a new advantage of wavelet basis with respect to the Fourier basis: they behave better at the points where a signal has a jump (of course, in order to have a good reconstruction we need to keep a reasonable amount of coefficients).

Example 5.2.2. Let $N = 511$, and consider the signal

$$z_n = \begin{cases} \sin\left(\frac{|n-128|^{1.7}}{128}\right), & \text{if } 128 \leq n \leq 255, \\ \sin\left(\frac{|n-128|^2}{128}\right), & \text{if } 384 \leq n \leq 447, \\ 0, & \text{otherwise.} \end{cases}$$

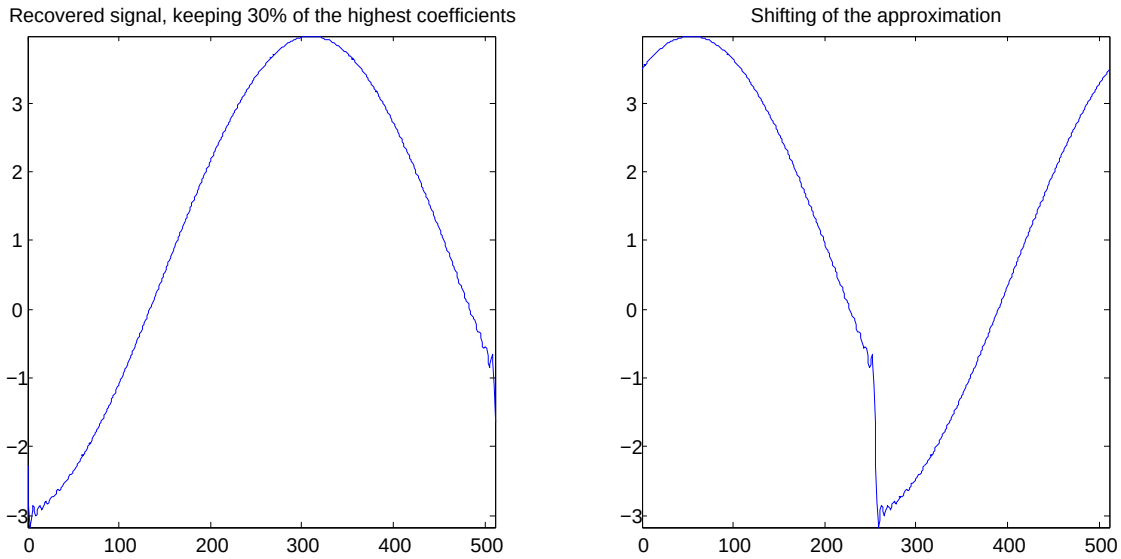

Figure 5.6: Compression of the signal from Example 4.3.2 by the Fourier basis, and its shifting, which allows us to observe the Gibbs phenomenon clearly.

We are going to compress this signal keeping only 10% of the coefficients. Figure 5.7 contains the compressed signal through the different basis; for the wavelet basis, we choose the 3rd stage.

We can observe how badly the Fourier basis works to compress this signal, but we could already expect that. In this case, the compression yielded is comparable to the Euclidean when keeping a low number of coefficients, as illustrated in the energy ratio plot of Figure 5.7. Moreover, when keeping more coefficients the Euclidean basis turns to be better. Since the Fourier basis consists of sinusoidals, if we want to compress a signal with constant pieces (like in this case) with a very low amount of coefficients, the cancellation will not be enough and the constant pieces will be very distorted, as we can observe. On the other hand, wavelet basis show a good compression. Indeed, taking a look at the energy ratio plot we can observe that if we keep 20% of the highest coefficients instead of 10%, the reconstruction will be nearly perfect using the Daubechies D6 3rd stage wavelet. Shannon real basis (on the same stage) does not yield such a good result, but nevertheless, it is much better than the one obtained through the Fourier basis.

Example 5.2.3. Let $N = 512$ and consider the signal

$$z_n = \begin{cases} 1 - n/64, & \text{if } 0 \leq n \leq 63, \\ 5 - n/64, & \text{if } 256 \leq n \leq 319, \\ 0, & \text{otherwise.} \end{cases}$$

In Example 4.3.1 we already saw that Daubechies D6 4th stage wavelet worked properly to compress z, but now we are going keep 7% of the coefficients, instead of the 5% from Example 4.3.1. Figure 5.8 shows the compression, and there we can

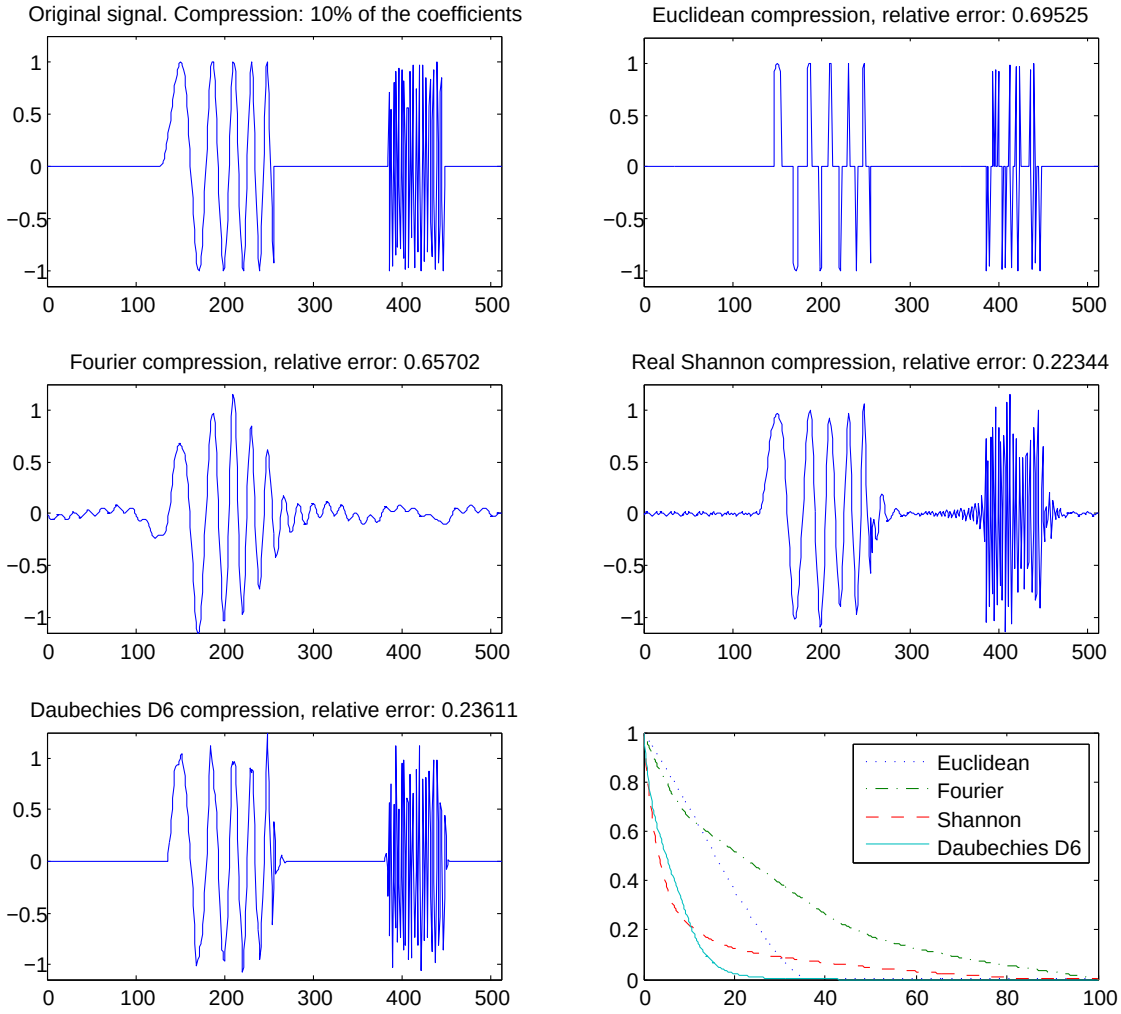

Figure 5.7: Compression from the signal shown in Example 5.2.2.

observe the nice approximation that Daubechies D6 basis yields (we chose the 4th stage for both of our wavelets), but the Shannon real basis does not work that well, even though it is still better than the Fourier basis. This is because z has so many zeros, concretely, the 75% of its length, and we saw in Figure 3.2 that the generators of the Shannon real basis had many small coefficients that were different from zero. Finally, we observe in the energy ratio plot that Daubechies D6 wavelet does not need many coefficients in order to provide a perfect reconstruction of our signal.

5.3 Extremal examples

Here we will present some signals which are closely related with vectors of the different basis we have seen throughout the document. The contrast will be very clear when looking at the relative error curves yielded by each example.

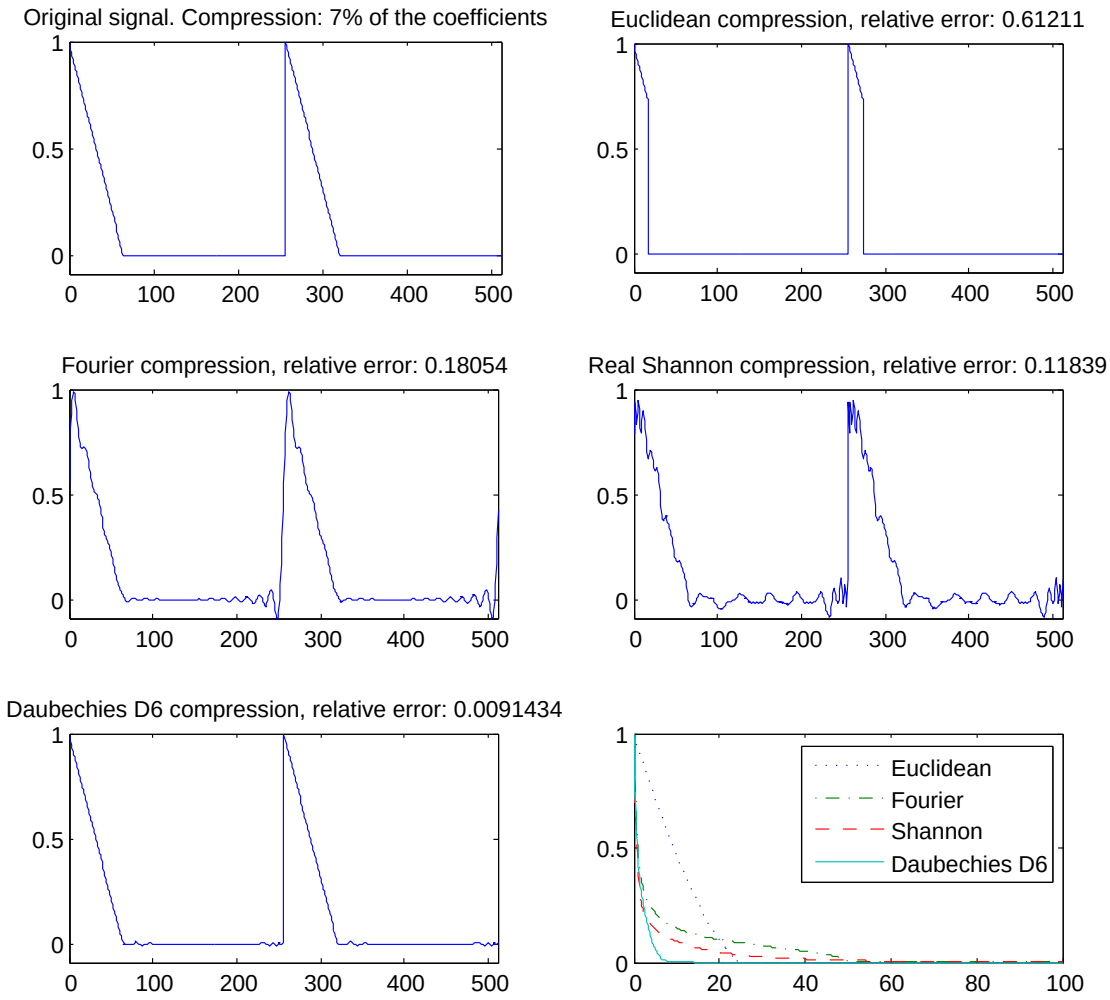

Figure 5.8: Compression of the signal from Example 5.2.3.

Example 5.3.1. Consider the signal $z_n = \cos(40n), n = 0, \dots 599$. Figure 5.9 shows the signal and its spectrum: the complex modulus of the coefficients of z in the Fourier basis. More generally, if a signal w is the sum of different sinusoids with various frequencies (relatively spaced between them), those will determine the number of "spikes" in the spectrum of the signal. Figure 5.10 shows this fact when $w_n = \sin(400n) + \cos(300n) - \cos(20n)$, for $n = 0 \dots 599$: in this case there are 3 "spikes".

Example 5.3.2. Consider the signal w defined in Example 5.3.1. Figure 5.11 shows the compression of the signal, with 15% of the coefficients, using the 3rd stage Shannon and Daubechies D6 basis. We observe in the relative error curves that the Fourier basis yields a high quality compression with a very low amount of coefficients. Moreover, Daubechies D6 approximation for w turns out to be very bad. This is because the signal is not localized at all (it is composed by 3 sinusoidals, two of them with high frequency). As Daubechies D6 basis generators were localized in space, keeping a low amount of coefficients will yield us "holes" in the signal, i.e.,

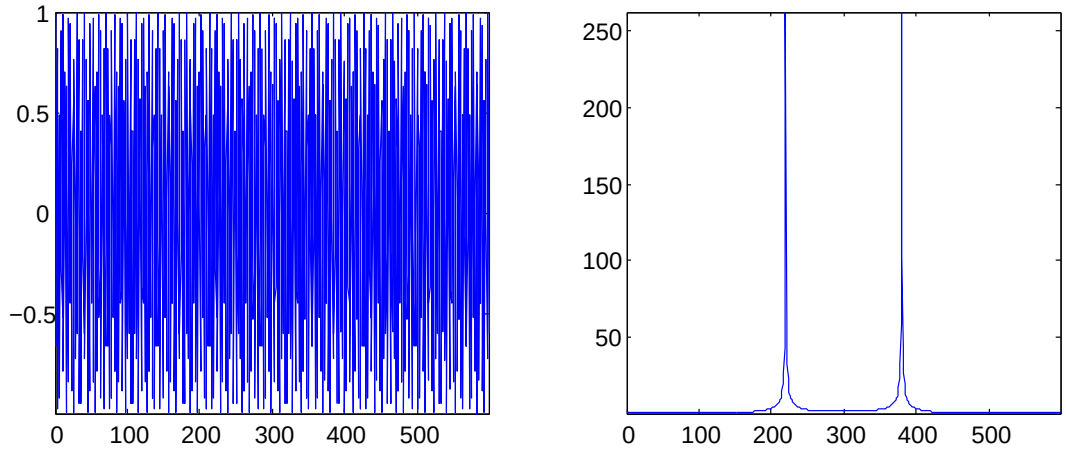

Figure 5.9: Signals $z_n = \sin(40n)$ and $\hat{z}$, respectively.

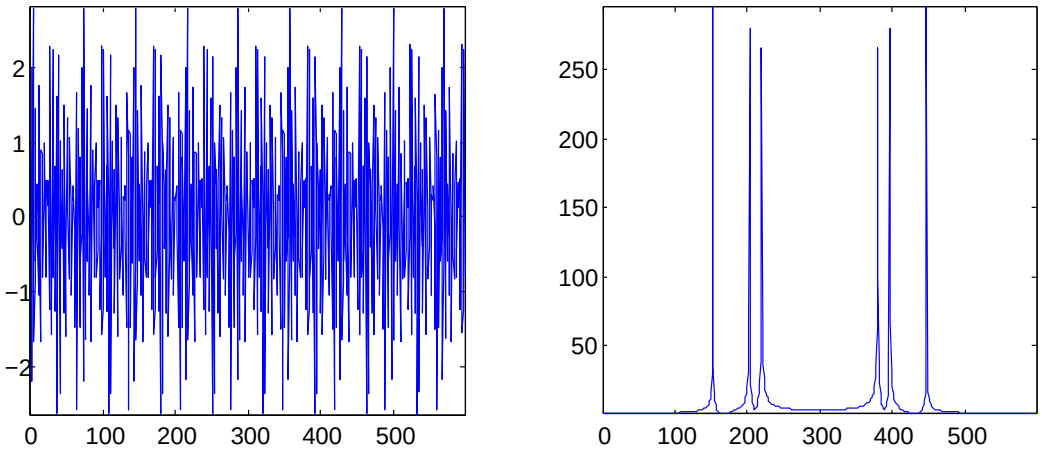

Figure 5.10: Signals w and $\hat{w}$ from Example 5.3.1, respectively.

indexes for which the approximation is exactly zero. Indeed, recall that each vector of such basis had only 6 nonzero entries, which makes impossible to cover the whole approximation (the 512 components) with nonzeros whenever we use a low amount of generators. We conclude that Daubechies D6 wavelet is bad when the signal is not localized in space. For the completeness of the example, in Figure 5.12 we show the coefficients of the 3rd stage wavelets used in the example: we can see the sparsity of the coefficients that causes the bad compression by wavelet basis.

Example 5.3.3. Before comparing further, let us take a look back to the Haar transform (Subsection 3.2.1). Since this transformation is, in some sense, an increase of resolution, can it work properly for a piecewise constant signal? The answer is affirmative, and to check it we just need to note that if the signal z is good enough, then $z * \tilde{v}$ is the zero vector. Recall (Section 3.2.1) that the coefficients of the first-stage Haar transform of a signal z are the following:

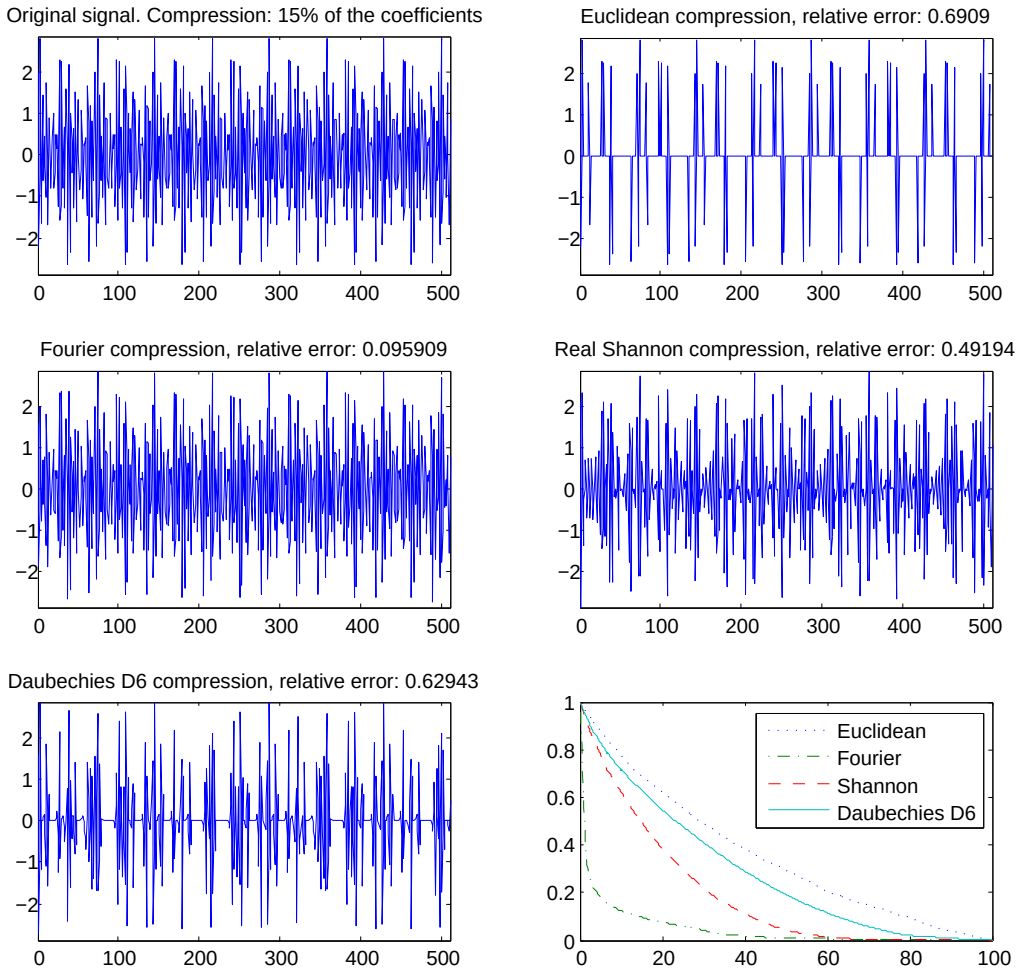

Figure 5.11: Pictures of the signal $w_n = \sin(400n) + \cos(300n) - \cos(20n)$ and its different compression methods (taking 3rd stage wavelets).

$$P_1'(z)_j = \sum_{k=0}^{N/2-1} \langle z, R_{2k}u\rangle (R_{2k}u)_{2j} = \frac{z_{2j} + z_{2j+1}}{2},$$
$$Q_1'(z)_j = \sum_{k=0}^{N/2-1} \langle z, R_{2k}v\rangle (R_{2k}v)_{2j} = \frac{z_{2j} - z_{2j+1}}{2}.$$

Now, if the vector z is piecewise constant, let $I_k, k \in \mathbb{N}$ be the sets of indexes corresponding to each constant piece of z. For example, if

$$z = (0, 1, 1, 2, 2, 1, 1, 0),$$

then $I_1 = \{0, 7\}$, $I_2 = \{1, 2\}$, $I_3 = \{3, 4\}$, $I_4 = \{5, 6\}$. Note that if, for every k, $|I_k|$ divides 2, then, up to a translation of z, we have that

$$P_1'(z)_j = \frac{z_{2j} + z_{2j+1}}{2} = z_{2j}, \qquad Q_1'(z)_j = 0,$$

and the exact reconstruction of z is given by at most $N/2$ coefficients. Moreover, if, for every k, $|I_k|$ divides 4, then we can carry this argument one step further, applying it to the vector $P_1'(z)$, since its corresponding I_k' will have cardinality divisible by 2. More generally, if we define, for a vector z,

$$n = \max\{n : |I_k| \text{ divides } 2^n, \text{ for all } k\},$$

then the Haar transform provides a full reconstruction of z with at most $N/2^n$ coefficients.

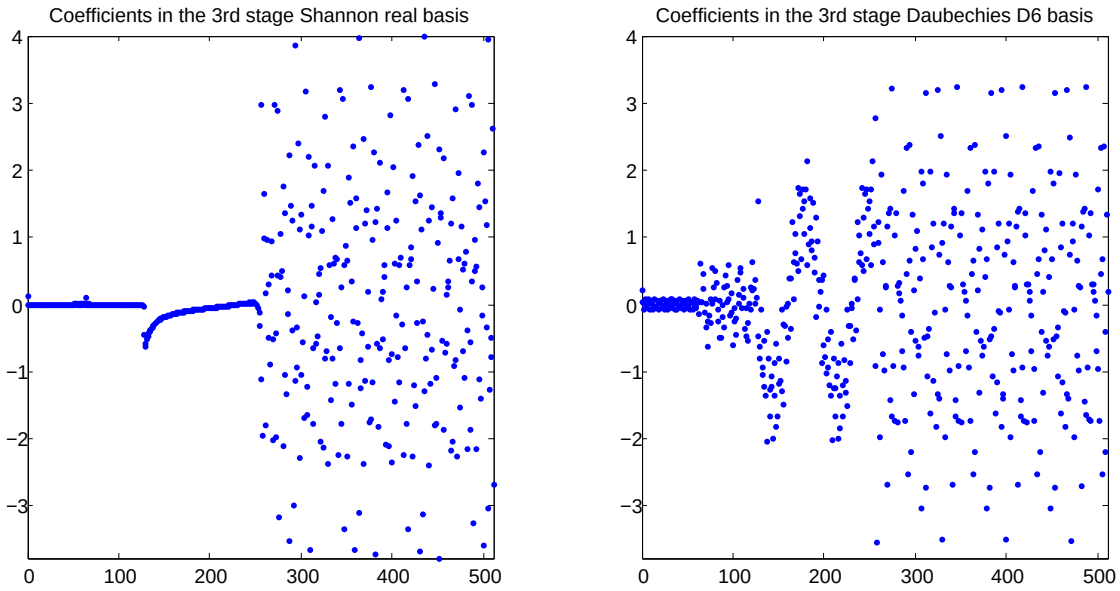

Figure 5.12: Coefficients of the signal $w_n = \sin(400n) + \cos(300n) - \cos(20n)$ in the 3rd stage wavelet basis.

5.4 Data compression in 2 dimensions

As we have already stated when we introduced the wavelet basis in Section 3.2, the 2-dimensional wavelet transforms are defined just as an iteration: for an $N \times M$ complex matrix y (or equivalently, a signal in $\ell^2(\mathbb{Z}_N \times \mathbb{Z}_M)$), we apply the transform T to all the rows of y, obtaining a new matrix z. Then, we apply again the transform T to the columns of z, yielding the desired matrix, say $T(y)$.

This 2-dimensional application of wavelet basis is mainly used for image processing. The images we are going to treat will be gray-scaled, that is, every component of our signals will contain the gray-scale intensity: an integer between 0 (black) and 255 (white), often also represented by a real number between 0 and 1 (the re-scaling of the integer scale). An important fact that we need to consider is that, even though we lose certain information whenever we perform a compression, this loss can sometimes lead to an indiscernible error (for the human eye) in the resulting image.

Example 5.4.1. We are going to compress a signal using the Haar transform. Recall that, for the 1-dimensional Haar transform, the obtained vector of coefficients corresponded to the "tendency" (the main information of the signal), and the "details', each one of them consisting on half of the total number of coefficients. For the 2-dimensional case, we will have 3 kinds of different details: indeed, if $z \in \ell^2(\mathbb{Z}_N \times \mathbb{Z}_M)$, the transform is defined as the iteration of one-dimensional transforms, first by rows, and then by columns. Thus, if we transform the rows of z, we obtain a matrix $P \in \ell^2(\mathbb{Z}_N \times \mathbb{Z}_M)$. Then, we apply the transform to the columns of P, obtaining a new matrix, say Q. Note that the matrix Q will be subdivided in 4 different blocks,

$$Q = \left(\begin{array}{c|c} A & H \\ \hline V & D \end{array}\right).$$

Those blocks correspond to the following: A contains the tendency of the signal, or equivalently, the approximation. Blocks H and V contain the horizontal and vertical details, respectively, and D the diagonal details. Thus, for the 2-dimensional case, the compression done by the Haar basis is keeping 25% of the coefficients at each stage (since the block A belongs to $\ell^2(\mathbb{Z}_{N/2} \times Z_{M/2})$. In Figure 5.13 we can see a first-stage approximation, which seems quite good. Also, we show the coefficients for the first-stage transform in Figure 5.14. Finally, the coefficients for the second-stage transform are contained in Figure 5.15. Note that with respect to the first-stage approximation, we only modified the tendency.

Figure 5.13: First-stage approximation by Haar basis (Example 5.4.1).

Example 5.4.2. Now we are going to compress the picture of Example 5.4.1 with the rest of basis we have. In principle, we would expect better compression from the other wavelet basis, since the Haar transform is the simplest basis we have treated, and as we have already seen, we are keeping 50% of the coefficients at each stage, but this is still a high number. Moreover, for high stages, the compression will not

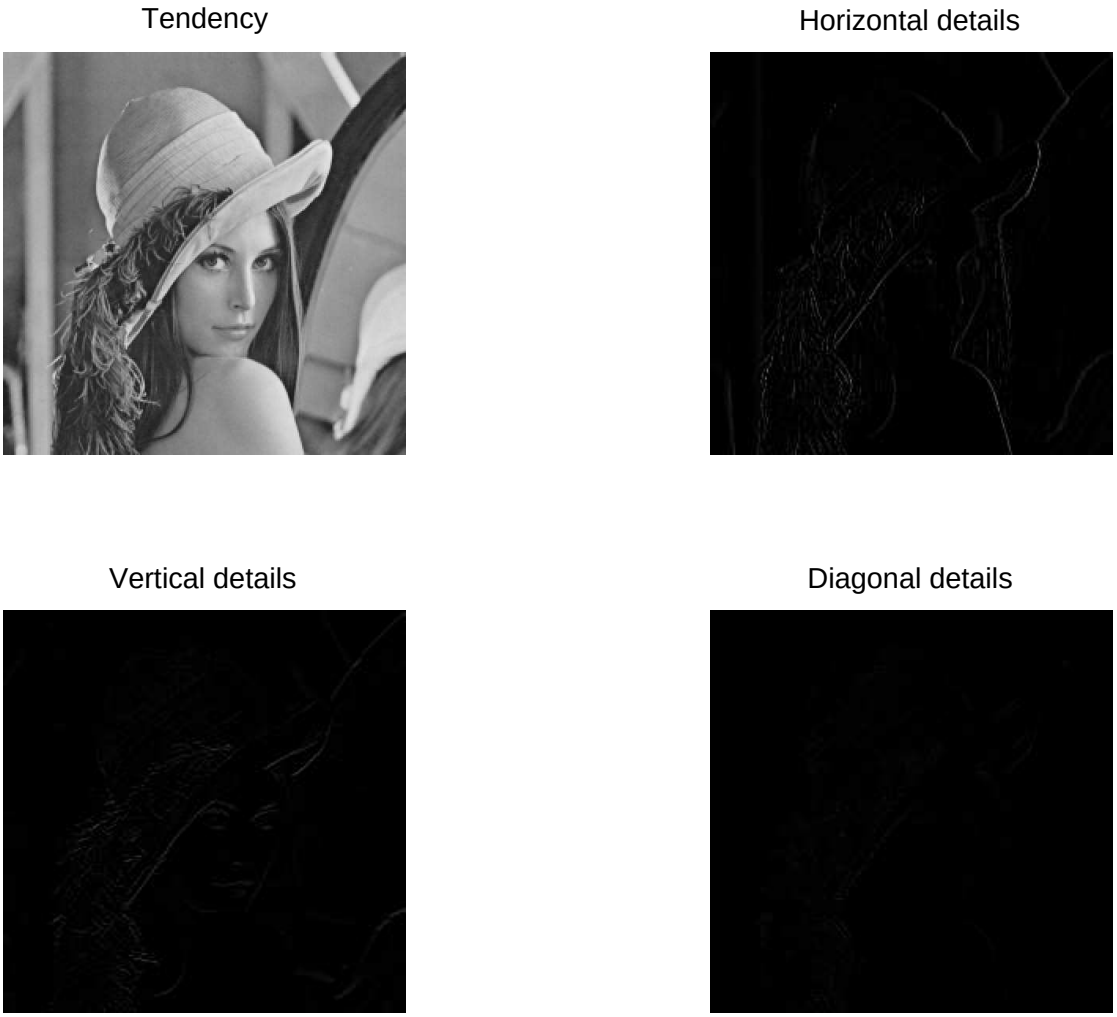

Figure 5.14: First-stage coefficients in the Haar basis (Example 5.4.1).

be so good. Figure 5.16 shows how badly Haar basis works at high stages. If we compare it with the compression given by the Shannon and Daubechies wavelets, the improvement is evident, taking the 4-th stage for those wavelet basis as well. Figure 5.17 shows the compression done by the other basis, taking the 4-th stage for Shannon and Daubechies D6 wavelets. As we can see, Fourier compression adds some noise to the image. Wavelets yield a very good compression considering that we are only keeping 5% of the coefficients. The results are obviously much better than those obtained by the 4-th stage Haar wavelet (Figure 5.16): more compression, and less error.

Figure 5.15: Second-stage coefficients in the Haar basis. (Example 5.4.1)

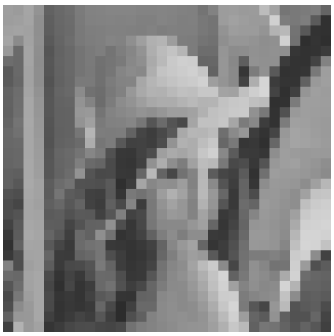

Figure 5.16: Approximation by fourth-stage Haar basis of the picture shown in Figure 5.13 (keeping 6.25% of the coefficients).

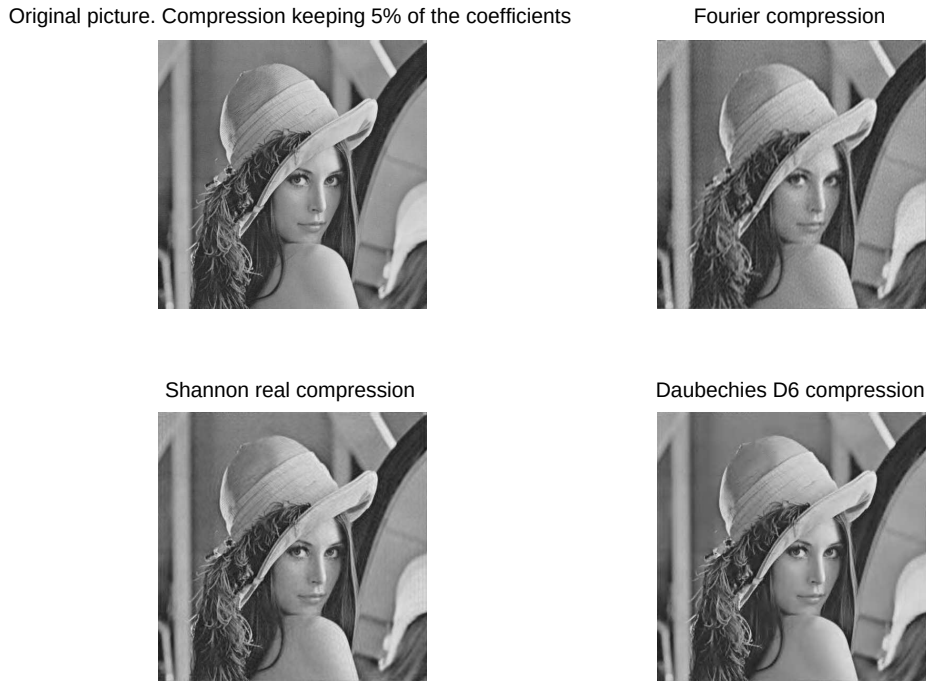

Figure 5.17: Approximation by different basis of the picture shown in Figure 5.13 (wavelets: 4-th stage).

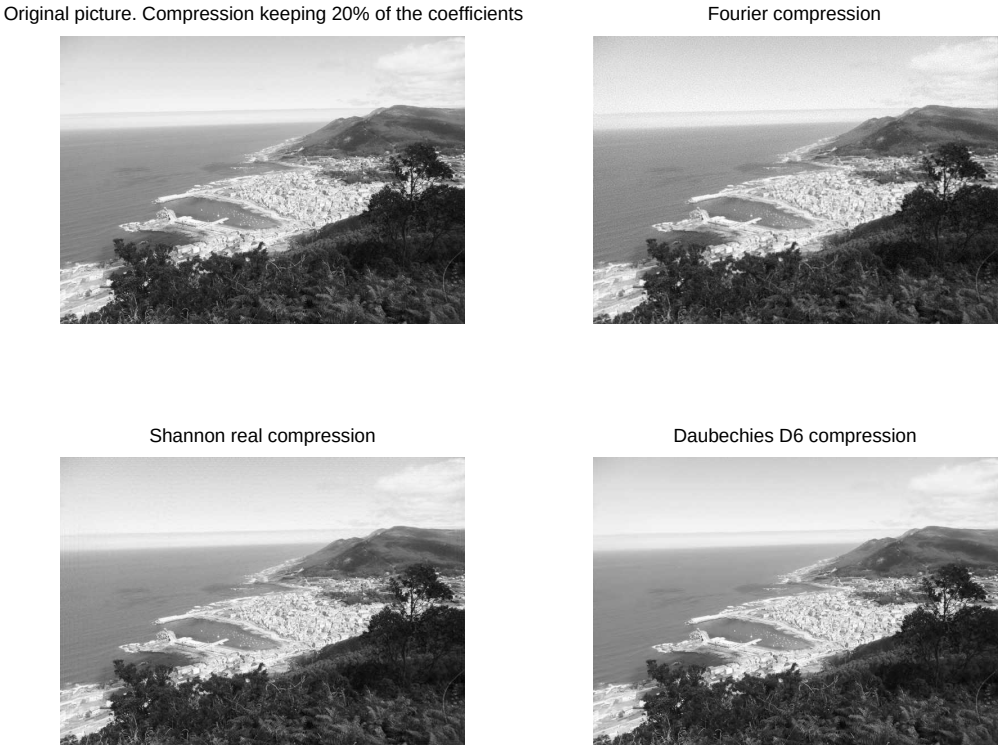

Figure 5.18: Compression corresponding to Example 5.4.3 (wavelets: 4-th stage).

Example 5.4.3. In this example we consider a picture with very small details, and compress it keeping a higher number of coefficients than before, namely 20%. Figure 5.18 shows such compression. As we have been observing, wavelet basis work better (in this case, the relative error in the approximation by Shannon and Daubechies D6 wavelets is lower than 0.005, and 0.002, respectively). Once again, the approximation made by Fourier basis shows the picture with some noise.

So, we conclude that whenever we want to compress pictures, the Fourier basis turns to be very bad for the compression: in the last example (5.4.3), it was bad even though we were keeping 20% of the coefficients, much more than we have been keeping throughout the other examples. Contrarily, wavelet basis are very good for this purpose: the approximations do not need a high percentage of coefficients to be accurate (see Example 5.4.2), and whenever it increases a bit, the error in the approximation decays very fast.

Chapter 6

Annex – compression and comparison code in MATLAB

In this chapter several programs (or just scripts) which have been used in what precedes will be shown. All these programs are written in MATLAB code. We need to remark that concerning vectors, MATLAB indexing begins at 1, while in this document we have always taken the indexes to begin at 0.

The routines shown here are the ones mostly used to construct the examples of Chapter 5. There will be 4 different transformations: Fourier, Haar, Shannon (real), and Daubechies D6. The first input variable for all these transformations will be a signal `y`. The rest of inputs will change depending on which program we are using, (being the Haar transform the most exceptional case). For wavelet transforms, the second variable `order` corresponds to the stage of the wavelet that we want to use. For instance, if we only want to compress a signal using a first-stage wavelet basis, we will choose `order=1`. The rest of input variables is varying depending on the transform we are choosing. Except for the Haar transform routine, the last and penultimate input variables will correspond to the truncation coefficient, and the truncation method, respectively. That is, if we want to keep an absolute number of coefficients, we will choose `'amount'` as the truncation method. Contrarily, if we want to keep relative number of them (i.e., a percentage), we will write `'percentage'`. Obviously, in the first case, the truncation coefficient will be between 1 and the length of the input signal, and in the second case, between 0 and 100. In all the examples we have discussed, the truncation method was chosen as `'percentage'`.

On the other hand, all the functions return a vector containing the following variables concatenated: the compressed signal, `T`, and the coefficients of the signal in the chosen basis. For instance, the Fourier transform routine returns the coefficients through the variable `FTcoef`, the Shannon program does the same through `STcoef`, etc.

We also remark that the user should be conscious of what are the conditions for those programs to be able to run (described in each section of Chapter 3) , as for example, a signal of even length for wavelet transforms. For bad input data, the

programs will not work.

6.1 Fourier transform

As there is no such thing as a p-th stage Fourier basis (since it is not a wavelet basis), in this routine we omit the input variable `'order'`.

```
function [output] = FourierTrans(y,truncation,k)
a=size(y); c=length(size(a));
if(min(a)==1 || c~=2)
    a=length(y);
    type='1d';
else
    type='2d';
end

switch type

    case '1d'

    FT=fft(y);
    FTcoef=FT;
    [~,I]=sort(abs(FT),'descend');
    if(strcmp(truncation,'percentage')==1)
        if(k<0 || k>100)
            display('Wrong percentage');
        end
        perc=round(k*a/100);
        aux=zeros(1,a);
        m=1;
        while m<=perc
            aux(I(m))=FT(I(m));
            m=m+1;
        end
        FT=aux;
    end
    if(strcmp(truncation,'amount')==1)
        if(k<1 || k>a)
            display('Wrong amount');
        end
        aux=zeros(1,a);
        m=1;
        while m<=perc
            aux(I(m))=FT(I(m));
```

```
            m=m+1;
        end
        FT=aux;
    end

    case '2d'

    FT=fft2(y);
    FTcoef=FT;
    aux=reshape(FT,1,a(1)*a(2));
    aux2=zeros(1,a(1)*a(2));
    [~,I]=sort(abs(aux),'descend');
    if(strcmp(truncation,'percentage')==1)
        if(k<0 || k>100)
            display('Wrong percentage');
            return;
        end
        perc=round(k*a(1)*a(2)/100);
        m=1;
        while m<=perc
            aux2(I(m))=aux(I(m));
            m=m+1;
        end
        FT=reshape(aux2,a(1),a(2));
    end
    if(strcmp(truncation,'amount')==1)
        if(k<1 || k>a)
            display('Wrong amount');
            return;
        end
        m=1;
        while m<=k
            aux2(I(m))=aux(I(m));
            m=m+1;
        end
        FT=reshape(aux2,a(1),a(2));
    end
    T=ifft2(FT);
    output=[T;FTcoef];
end
```

6.2 Wavelet transforms

6.2.1 Haar transform

As we can note, this function does not allow us to specify how many coefficients we will keep during the compression. Instead, it will always keep half of them for each stage: as we said earlier in this work (Section 3.2.1), the Haar transform can be seen as change of resolution, where the coefficients relative to u contain the important information about the signal, and the ones relative to v, the details to pass from a resolution of 2 to a resolution of 1. Therefore, since we are only keeping the coefficients obtained by the inner products with the translations of u, the compression will be of the 50% in each stage, so that, if the signal has length N and we choose the p-th iteration of this basis, the number of needed coefficients to obtain the compressed signal will be $N/2^p$.

```
function [output] = HaarTrans(y,order)
a=size(y); c=length(size(a));
if(min(a)==1 || c~=2)
    a=length(y);
    if(rem(a,2^order)~=0)
        display('Invalid dimensions of the signal');
        return;
    end
    type='1d';
else
    if(rem(a(1),2^order)~=0 || rem(a(1),2^order)~=0)
        display('Invalid dimensions of the signal');
        return;
    end
    type='2d';
end

switch type

    case '1d'

    n=1; HT=y;
    while n<=order
        HT(1:a/2^(n-1))=transform(HT(1:a/2^(n-1)));
        n=n+1;
    end
    HTcoef=HT;
    T=HT(1:a/2^order);
    l=1;
    while l<=order
```

```
        T(1:a/2^(order-l))=inverse(T);
        l=l+1;
    end
    output=[T;HTcoef];

    case '2d'

    n=1; HT=y;
    while n<=order
        m=1;
        while m<=a(1)/2^(n-1)
            HT(m,1:a(2)/2^(n-1))=transform(HT(m,1:a(2)/2^(n-1)));
            m=m+1;
        end
        m=1;
        while m<=a(2)/2^(n-1)
            HT(1:a(1)/2^(n-1),m)=...
                (transform((HT(1:a(1)/2^(n-1),m))'))';
            m=m+1;
        end
        n=n+1;
    end
    HTcoef=HT;
    n=1; T=HT(1:a(1)/2^order,1:a(2)/2^order);
    while n<=order
        m=1;
        while m<=a(2)/2^(order-n+1)
            T(1:a(1)/2^(order-n),m)=...
                (inverse((T(1:a(1)/2^(order-n+1),m))'))';
            m=m+1;
        end
        m=1;
        while m<=a(1)/2^(order-n)
            T(m,1:a(2)/2^(order-n))=...
                inverse(T(m,1:a(2)/2^(order-n+1)));
            m=m+1;
        end
        n=n+1;
    end
    output=[T;HTcoef];
end

    function [HT] = transform(y)
        b=length(y);
```

```
        HT=zeros(1,b);
        u=zeros(1,b); v=zeros(1,b);
        u(1)=1/sqrt(2); u(b)=u(1); v(b)=u(1); v(1)=-u(1);
        i=1;
        yu=ifft(fft(y).*conj(fft(u)));
        yv=ifft(fft(y).*conj(fft(v)));
        while i<=b/2
            HT(i)=yu(2*i);
            HT(i+b/2)=yv(2*i);
            i=i+1;
        end
    end

    function [RecSig] = inverse(y)
        b=length(y);
        aux=zeros(1,2*b);
        u=zeros(1,2*b);     u(1)=1/sqrt(2); u(2*b)=u(1);
        j=1;
        aux(2*b)=y(b);
        while j<=b-1
            aux(2*j)=y(j);
            j=j+1;
        end
        RecSig=ifft(fft(u).*fft(aux));
    end
end
```

6.2.2 Shannon real transform

```
function [output] = ShannonTrans(y,order,truncation,k)
    a=size(y); c=length(size(a));
    if(min(a)==1 || c~=2)
        a=length(y);
        if(rem(a,2^(order+1))~=0)
            display('Wrong dimensions');
            return;
        end
        type='1d';
    else
        if(rem(a(1),2^(order+1))~=0 || rem(a(1),2^(order+1))~=0)
            display('Wrong dimensions');
            return;
        end
        type='2d';
```

```
    end

switch type

    case '1d'

    ST=y; n=1;
    while n<=order
        ST(1:a/2^(n-1))=transform(ST(1:a/2^(n-1)));
        n=n+1;
    end
    STcoef=ST;
    [~,I]=sort(abs(ST),'descend');
    if(strcmp(truncation,'percentage')==1)
        if(k<0 || k>100)
            display('Wrong percentage');
        end
        perc=round(k*a/100);
        aux=zeros(1,a);
        m=1;
        while m<=perc
            aux(I(m))=ST(I(m));
            m=m+1;
        end
        ST=aux;
    end
    if(strcmp(truncation,'amount')==1)
        if(k<1 || k>a)
            display('Wrong amount');
        end
        aux=zeros(1,a);
        m=1;
        while m<=k
            aux(I(m))=ST(I(m));
            m=m+1;
        end
        ST=aux;
    end
    T=ST; n=1;
    while n<=order
        T(1:a/2^(order-n))=inverse(T(1:a/2^(order-n)));
        n=n+1;
    end
    output=[T;STcoef];
```

```
case '2d'

ST=y; n=1;
while n<=order
    m=1;
    while m<=a(1)/2^(n-1)
        ST(m,1:a(2)/2^(n-1))=transform(ST(m,1:a(2)/2^(n-1)));
        m=m+1;
    end
    m=1;
    while m<=a(2)/2^(n-1)
        ST(1:a(1)/2^(n-1),m)=...
            (transform((ST(1:a(1)/2^(n-1),m))'))';
        m=m+1;
    end
    n=n+1;
end
STcoef=ST;
aux=reshape(ST,1,a(1)*a(2));
aux2=zeros(1,a(1)*a(2));
[~,I]=sort(abs(aux),'descend');
if(strcmp(truncation,'percentage')==1)
    if(k<0 || k>100)
        display('Wrong percentage');
        return;
    end
    perc=round(k*a(1)*a(2)/100);
    m=1;
    while m<=perc
        aux2(I(m))=aux(I(m));
        m=m+1;
    end
    ST=reshape(aux2,a(1),a(2));
end

if(strcmp(truncation,'amount')==1)
    if(k<1 || k>a)
        display('Wrong amount');
        return;
    end
    m=1;
    while m<=k
        aux2(I(m))=aux(I(m));
```

```
            m=m+1;
        end
        ST=reshape(aux2,a(1),a(2));
    end
    n=1;
    T=ST;
    while n<=order
        m=1;
        while m<=a(2)/2^(order-n)
            T(1:a(1)/2^(order-n),m)=...
                (inverse((T(1:a(1)/2^(order-n),m))'))';
            m=m+1;
        end
        m=1;
        while m<=a(1)/2^(order-n)
            T(m,1:a(2)/2^(order-n))=inverse(T(m,1:a(2)/2^(order-n)));
            m=m+1;
        end
        n=n+1;
    end
    output=[T;STcoef];
end

    function [ST]= transform(y)
        b=length(y);
        hatu=zeros(1,b); hatv=zeros(1,b);
        j=1;
        while j<=b/4
            hatu(j)=sqrt(2);
            hatv(b/4+1+j)=sqrt(2);
            hatv(b/2+j)=sqrt(2);
            hatu(b-j+1)=sqrt(2);
            j=j+1;
        end
        hatu(b/4+1)=1i; hatu(3*b/4+1)=-1i;
        hatv(b/4+1)=1; hatv(3*b/4+1)=1;
        z1=circshift(ifft(fft(y).*conj(hatu)),[1,1]);
        z2=circshift(ifft(fft(y).*conj(hatv)),[1,1]);
        ST(b/2)=z1(b); ST(b)=z2(b); j=1;
        while j<=b/2-1
            ST(j)=z1(2*j);
            ST(b/2+j)=z2(2*j);
            j=j+1;
        end
```

```
    end

    function [signal] =inverse(ST)
        b=length(ST);
        hatu=zeros(1,b); hatv=zeros(1,b);
        j=1;
        while j<=b/4
            hatu(j)=sqrt(2);
            hatv(b/4+1+j)=sqrt(2);
            hatv(b/2+j)=sqrt(2);
            hatu(b-j+1)=sqrt(2);
            j=j+1;
        end
        hatu(b/4+1)=1i; hatu(3*b/4+1)=-1i;
        hatv(b/4+1)=1; hatv(3*b/4+1)=1;
        r=zeros(1,b); s=zeros(1,b);
        r(b)=ST(b/2); s(b)=ST(b); j=1;
        while j<=b/2-1
            r(2*j)=ST(j);
            s(2*j)=ST(b/2+j);
            j=j+1;
        end
        q=circshift(ifft(hatu.*fft(r)),[-1,-1]);
        w=circshift(ifft(hatv.*fft(s)),[-1,-1]);
        signal=real(q+w);
    end
end
```

6.2.3 Daubechies D6 transform

```
function [output] = D6Trans(y,order,truncation,k)

a=size(y); c=length(size(a));
if(min(a)==1 || c~=2)
    a=length(y);
    if(rem(a,2^order)~=0 || a/2^order<8)
        display('Invalid dimensions of the signal');
        return;
    end
    type='1d';
else
    if(rem(a(1),2^order)~=0 || a(1)/2^order<8 ||...
            rem(a(1),2^order)~=0 ||a(2)/2^order<8)
        display('Invalid dimensions of the signal');
```

```
        return;
    end
    type='2d';
end
A=1-sqrt(10); B=1+sqrt(10); C=sqrt(5+2*sqrt(10));

switch type

    case '1d'

    DT=y; n=1;
    while n<=order
        DT(1:a/2^(n-1))=transform(DT(1:a/2^(n-1)));
        n=n+1;
    end
    DTcoef=DT;
    [~,I]=sort(abs(DT),'descend');
    if(strcmp(truncation,'percentage')==1)
        if(k<0 || k>100)
            display('Wrong percentage');
            return;
        end
        perc=round(k*a/100);
        aux=zeros(1,a);
        m=1;
        while m<=perc
            aux(I(m))=DT(I(m));
            m=m+1;
        end
        DT=aux;
    end

    if(strcmp(truncation,'amount')==1)
        if(k<1 || k>a)
            display('Wrong amount');
            return;
        end
        aux=zeros(1,a);
        m=1;
        while m<=k
            aux(I(m))=DT(I(m));
            m=m+1;
        end
        DT=aux;
```

```
end
n=1;
T=DT;
while n<=order
    T(1:a/2^(order-n))=inverse(T(1:a/2^(order-n)));
    n=n+1;
end
output=[T;DTcoef];

case '2d'

DT=y; n=1;
while n<=order
    m=1;
    while m<=a(1)/2^(n-1)
        DT(m,1:a(2)/2^(n-1))=transform(DT(m,1:a(2)/2^(n-1)));
        m=m+1;
    end
    m=1;
    while m<=a(2)/2^(n-1)
        DT(1:a(1)/2^(n-1),m)=...
            (transform((DT(1:a(1)/2^(n-1),m))'))';
        m=m+1;
    end
    n=n+1;
end
DTcoef=DT;
aux=reshape(DT,1,a(1)*a(2));
aux2=zeros(1,a(1)*a(2));
[~,I]=sort(abs(aux),'descend');
if(strcmp(truncation,'percentage')==1)
    if(k<0 || k>100)
        display('Wrong percentage');
        return;
    end
    perc=round(k*a(1)*a(2)/100);
    m=1;
    while m<=perc
        aux2(I(m))=aux(I(m));
        m=m+1;
    end
    DT=reshape(aux2,a(1),a(2));
end
```

```
    if(strcmp(truncation,'amount')==1)
        if(k<1 || k>a)
            display('Wrong amount');
            return;
        end
        m=1;
        while m<=k
            aux2(I(m))=aux(I(m));
            m=m+1;
        end
        DT=reshape(aux2,a(1),a(2));
    end
    n=1;
    T=DT;
    while n<=order
        m=1;
        while m<=a(2)/2^(order-n)
            T(1:a(1)/2^(order-n),m)=...
                (inverse((T(1:a(1)/2^(order-n),m))'))';
            m=m+1;
        end
        m=1;
        while m<=a(1)/2^(order-n)
            T(m,1:a(2)/2^(order-n))=inverse(T(m,1:a(2)/2^(order-n)));
            m=m+1;
        end
        n=n+1;
    end
    output=[T;DTcoef];
end

    function [DT]= transform(y)
        b=length(y);
        u=zeros(1,b); v=zeros(1,b);
        u(b)=B+C; u(1)=2*A+3*B+3*C; u(2)=6*A+4*B+2*C;
        u(3)=6*A+4*B-2*C; u(4)=2*A+3*B-3*C; u(5)=B-C;
        u=sqrt(2)/32*u; v(1)=u(b); v(b)=-u(1); v(b-1)=u(2);
        v(b-2)=-u(3); v(b-3)=u(4); v(b-4)=-u(5);
        z1=circshift(ifft(fft(y).*conj(fft(u))),[1,1]);
        z2=circshift(ifft(fft(y).*conj(fft(v))),[1,1]);
        DT(b/2)=z1(b); DT(b)=z2(b);
        j=1;
        while j<=b/2-1
            DT(j)=z1(2*j);
```

```
            DT(b/2+j)=z2(2*j);
            j=j+1;
        end
    end

    function [signal] =inverse(DT)
        b=length(DT);
        u=zeros(1,b); v=zeros(1,b);
        u(b)=B+C; u(1)=2*A+3*B+3*C; u(2)=6*A+4*B+2*C; u(3)=6*A+4*B-2*C;
        u(4)=2*A+3*B-3*C; u(5)=B-C; u=sqrt(2)/32*u;
        v(1)=u(b); v(b)=-u(1); v(b-1)=u(2); v(b-2)=-u(3); v(b-3)=u(4);
        v(b-4)=-u(5);
        r=zeros(1,b); s=zeros(1,b);
        j=1;
        r(b)=DT(b/2); s(b)=DT(b);
        while j<=b/2-1
            r(2*j)=DT(j);
            s(2*j)=DT(b/2+j);
            j=j+1;
        end
        q=circshift(ifft(fft(u).*fft(r)),[-1,-1]);
        w=circshift(ifft(fft(v).*fft(s)),[-1,-1]);
        signal=q+w;
        end
end
```

6.3 Relative error computation

This code will compute the relative error done when compressing a one-dimensional signal z by w, as a function of the percentage of values which we keep for w. This means that the Y axis will contain the value $\|z-w\|/\|z\|$, while the X axis will be the percentages, from 0 to 100. The input for this function will be a signal `y`, followed by a percentage coefficient, `percentage`. The last two input parameters are the stages for the Shannon real basis, and Daubechies D6, respectively. It will compute and draw the four different compressions (Euclidean, Fourier, real Shannon, and Daubechies D6) of z (through the different routines presented previously in this section, except Haar transform), as well as the plot of the original signal. Moreover, it will also construct the relative error curves for all the compression methods we are considering, so we can decide with ease which transform is better for the input signal just by looking at those curves. This code generates most of the plots shown in Section 5.2.

```
function [] = Error(y,percentage,ShanStage,D6Stage)
    if(percentage<0 || percentage >100)
```

```
    display('Error, wrong percentage');
    return;
end
a=length(y); ynorm=norm(y);
[Y,I]=sort(abs(y),'descend');
k=1; aux=zeros(1,a);
while k<=round(percentage*a/100)
    aux(I(k))=y(I(k));
    k=k+1;
end
compId=aux;
SFT=FourierTrans(y,'percentage',percentage); compFT=SFT(1,:);
FTcoef=1/sqrt(a)*SFT(2,:); [~,IFT]=sort(abs(FTcoef),'descend');
SST=ShannonTrans(y,ShanStage,'percentage',percentage);
compST=SST(1,:); STcoef=SST(2,:);
[~,IST]=sort(abs(STcoef),'descend');
SDT=D6Trans(y,D6Stage,'percentage',percentage); compDT=SDT(1,:);
DTcoef=SDT(2,:); [~,IDT]=sort(abs(DTcoef),'descend');
yId=zeros(1,a); yFT=zeros(1,a); yST=zeros(1,a); yDT=zeros(1,a);
yId(1)=norm(y)^2; yFT(1)=norm(y)^2; yST(1)=norm(y)^2;
yDT(1)=norm(y)^2;
j=2; x=linspace(0,100,a);
while j<=a
    N=round(x(j)*a/100); n=round(x(j-1)*a/100)+1;
    if N>=n
        yId(j)=yId(j-1)-sum(y(I(n:N)).^2);
        yFT(j)=yFT(j-1)-sum(abs(FTcoef(IFT(n:N))).^2);
        yST(j)=yST(j-1)-sum(abs(STcoef(IST(n:N))).^2);
        yDT(j)=yDT(j-1)-sum(abs(DTcoef(IDT(n:N))).^2);
    else
        yId(j)=yId(j-1); yFT(j)=yFT(j-1);
        yST(j)=yST(j-1); yDT(j)=yDT(j-1);
    end
    j=j+1;
end
X=[0:a-1];
m=min([min(y), min(compId), min(compFT), min(compST),...
    min(compDT)]);
M=max([max(y), max(compId), max(compFT), max(compST),...
    max(compDT)]);
subplot(3,2,1); plot(X,y), title(['Original signal.'...
    ' Compression: ',num2str(percentage),...
    '\% of the coefficients']);
axis([0 a-1 m M]);
```

```
    subplot(3,2,2); plot(X,compId), title(['Euclidean'...
        ' compression, relative error: ',...
        num2str(norm(y-compId)/ynorm)]);
    axis([0 a-1 m M]);
    subplot(3,2,3); plot(X,compFT), title(['Fourier compression,'...
        ' relative error: ',num2str(norm(y-compFT)/ynorm)]);
    axis([0 a-1 m M]);
    subplot(3,2,4); plot(X,compST), title(['Real Shannon'...
        ' compression, relative error: ',...
        num2str(norm(y-compST)/ynorm)]);
    axis([0 a-1 m M]);
    subplot(3,2,5); plot(X,compDT), title(['Daubechies D6'...
        ' compression, relative error: ',...
        num2str(norm(y-compDT)/ynorm)]);
    axis([0 a-1 m M]);
    subplot(3,2,6); plot(x,sqrt(yId)/ynorm,':',x,sqrt(yFT)/ynorm,...
        '-.',x,sqrt(yST)/ynorm,'--',x,sqrt(yDT)/ynorm,'-');
    axis([0 100 0 1]);
    legend('Euclidean','Fourier','Shannon','Daubechies D6');
end
```

6.4 Computation of $L(N)$

The code presented in this section is devoted to compute the exact value of $L(N)$ for any N. It is based in the results from Propositions 2.4.11 and 2.4.14, and Theorem 2.4.17.

```
function [LN] = L(N)
if N==2
    LN=0;
    return;
elseif isprime(N)==1
    LN=floor(N/2);
    return;
end
j=2; index=1;
divisors=zeros(floor(sqrt(N)),1);
while j<=floor(N/2)
    if(rem(N,j)==0)
        divisors(index)=j;
        index=index+1;
    end
    j=j+1;
end
```

```
a=index-1;
j=floor(N-sqrt(N));
flag=0;
while(flag==0)
    if(N-hamming(N-j)>=j && N-hamming(N-j-1)<=j)
     LN=j; flag=1;
    end
    j=j-1;
end

    function[M]=hamming(d)

        l=1; M=N;
        while l<=a
            T=divisors(l);
            c=0; m=2;
            while m<=N/T
                if(rem(N/T,m)==0)
                    p=m; break;
                end
                m=m+1;
            end
            k=1;
            while k<p
                if(k*T<=d && (k+1)*T>d)
                    c=k; break;
                end
                k=k+1;
            end
            if(c~=0 && (p+1-c)*N/(p*T)<M)
                M=(p+1-c)*N/(p*T);
            end
            l=l+1;
        end
    end
end
```

Bibliography

[1] G. Bachman, L. Narici, and E. Beckenstein, *Fourier and Wavelet Analysis*, Universitext, Springer-Verlag, New York, 2000.

[2] I. Daubechies, *Ten Lectures on Wavelets*, CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, 1992.

[3] S. Delvaux and M. Van Barel, *Rank-deficient submatrices of Kronecker products of Fourier matrices*, Linear Algebra Appl. **426** (2007), 349–367.

[4] S. Delvaux and M. Van Barel, *Rank-deficient submatrices of Fourier matrices*, Linear Algebra Appl. **429** (2008), 1587–1605.

[5] J. Dieudonné, *Une Propriété des Racines de l'Unité*, Rev. Un. Mat. Argentina **25** (1970/71), 1–3.

[6] D. L. Donoho and P. B. Stark, *Uncertainty principles and signal recovery*, SIAM J. Appl. Math. **49** (1989), 906–931.

[7] M. W. Frazier, *An Introduction to Wavelets Through Linear Algebra*, Undergraduate Texts in Mathematics, Springer-Verlag, New York, 1999.

[8] M. W. Frazier and R. Torres, *The sampling theorem, ϕ-transform, and Shannon wavelets for $\mathbb{R}$, $\mathbb{Z}$, $\mathbb{T}$ and $\mathbb{Z}_N$. Wavelets: mathematics and applications*, Stud. Adv. Math., CRC, 1994, 221–245.

[9] P. A. Fuhrmann, *A Polynomial Approach to Linear Algebra*, Universitext, Springer, New York, 2012.

[10] E. Hernández and G. Weiss, *A First Course on Wavelets*, with a foreword by Yves Meyer. Studies in Advanced Mathematics, CRC Press, Boca Raton, FL, 1996.

[11] Y. Meyer, *Wavelets and Operators*, translated from the 1990 French original by D. H. Salinger. Cambridge Studies in Advanced Mathematics, 37. Cambridge University Press, Cambridge, 1992.

[12] M. A. Pinsky, *Introduction to Fourier Analysis and Wavelets*, Brooks/Cole Series in Advanced Mathematics, Brooks/Cole, 2002.

[13] P. Stevenhagen and H. W. Lenstra Jr., *Chebotarëv and his density theorem*, Math. Intelligencer **18** (1996), 26–37.

[14] L. A. Takhtajan, *Quantum Mechanics for Mathematicians*, Graduate Studies in Mathematics, American Mathematical Society, Rhode Island, 2008.

Index