APPLICATIONS OF COMPRESSED SENSING IN COMPUTATIONAL PHYSICS

by

Andreas Våvang Solbrå

THESIS

for the degree of

MASTER OF SCIENCE

Faculty of Mathematics and Natural Sciences
University of Oslo

June 2014

Abstract

Conventional sampling theory is dictated by Shannon's celebrated sampling theorem: For a signal to be reconstructed from samples, it must be sampled with at least twice the maximum frequency found in the signal. This principle is key in all modern signal acquisition, from consumer electronics to medical imaging devices. Recently, a new theory of signal acquisition has emerged in the form of Compressed Sensing, which allows for complete conservation of the information in a signal using far fewer samples than Shannon's theorem dictates. This is achieved by noting that signals with information are usually structured, allowing them to be represented with very few coefficients in the proper basis, a property called sparsity.

In this thesis, we survey the existing theory of compressed sensing, with details on performance guarantees in terms of the Restricted Isometry Property. We then survey the state-of-the-art applications of the theory, including improved MRI using Total Variation sparsity and restoration of seismic data using curvelet and wave atom sparsity.

We apply Compressed Sensing to the problem of finding statistical properties of a signal based CS methods, by attempting to measure the Hurst exponent of rough surfaces by partial measurements.

We suggest an improvement on previous results in seismic data restoration, by applying a learned dictionary of signal patches for restoration.

Acknowledgements

There are many people, without the influence of which, this thesis would never have happened. Chronologically, the first would be my high school physics teacher, Sveinung Mjelde. He was the person who lit my passion for mathematics outside of simply doing what was required by the class curriculum. In helping me practice for the Norwegian Mathematical Olympiad, and encouraging me, he elevated my abilities in problem solving more than anyone had earlier, or has since.

Secondly, this thesis would most likely never have come about without the subtle influence of one of my best friends, Anders Hafreager. When finishing high school, I was unsure what path to follow to my higher education. I knew that I wanted to study physics in some capacity, but it was a dead race between the University of Oslo (UiO) and the Norwegian University of Science and Technology (NTNU). Anders broke the tie by inviting me to UiO and giving me a tour, as well as inviting me to live with him and two other friends in a shared apartment.

During my first year at UiO, I got in touch with one of my professors, Knut Mørken, about the structure of his course. We had several conversations which eventually led to me working three summer jobs on the ongoing project at UiO of bringing programming into the bachelor courses in the natural sciences, Computers in Science Education (CSE). During the second year of these summer jobs, Knut, along with Øyvind Ryan, wanted to make a bachelor course in the subject of wavelet theory in a linear algebra setting, an approach rarely found at universities. I was tasked with being a "test bunny" of sorts for this course. Again, without this early exposure to wavelets and basis representations, this thesis would be unlikely.

During the first year of my Master's studies, my to-be supervisor, Anders Malthe-Sørenssen mentioned to Anders Hafreager, this new theory called Compressed Sensing, which could restore signals with seemingly impossible quality, using some mixture of wavelets, Fourier transforms and "magic". Anders Hafreager made the connection to my previous experience with wavelets, and passed this concept on to me. I got in touch with Anders Malthe-Sørenssen, and we formulated a Master's project. I approached Øyvind about being my co-supervisor, which he accepted.

Both of my advisors have been invaluable to this thesis. Anders has provided a good opportunity for me to freely approach applications, giving useful insight and guidance when needed. Øyvind has provided with absolutely essential help on the technical and theoretical side; without his help this thesis would contain nothing of note, as I would have spent too much time getting the computational details of the basics in place, to be able to explore avenues such as the curvelet transform or dictionary learning.

While these are the people directly involved in the creation of this thesis, there are several people who significantly improves my quality of life, which deserve mention. My parents and my sister, as well as my grandmother, who I hope to be able to have dinner with more often in the future than I've had in the last six months. In addition, I would like to thank the two people I've shared office, as well as workouts with during my thesis work, Henrik Sveinsson and Fredrik Pettersen. You are both great people. This extends to the entire Computational Physics group.

Finally, I would like to thank my wonderful girlfriend, Mathilde Nygaard Kamperud, who brings me more happiness than I could reasonably convey within these pages.

Oslo, 8 June, 2014
Andreas Våvang Solbrå

Contents

Chapter 1

Introduction

Classic sampling theory dictates that a signal can only be exactly reconstructed if the sample frequency is at least double the maximum frequency in the signal, a principle which is called Shannon's sampling theorem. This underlying principle is found in any digital signal acquisition device, from the cameras on most new cell phones and music recorders, to medical MRIs and telescopes in orbit around the Earth. This poses a problem, as even very mundane signals must be recorded with a large amount of data in order to ensure that the signal features are preserved. This has led to a situation where we now collect far more data than we are able to store, even with the combined storage of all the worlds hard drives. In 2010, The Economist published a special report entitled "Data, data everywhere", where they noted that already in 2007, the total amount of information created exceeded the storage space worldwide, and that this trend was increasing (see figure 1.1). They referred to this problem as a "data deluge", where one might drown in the sheer amount of data, unable to find the interesting numbers.

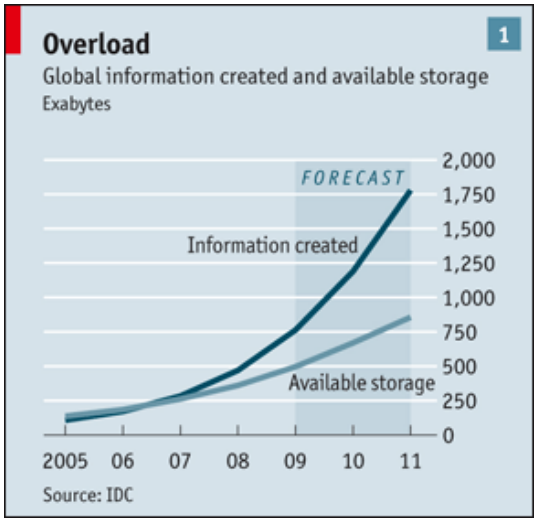


Figure 1.1: The total data created worldwide compared to the total storage space. The figure is from The Economist.

In order to overcome this massive influx of data, our solution is usually to immediately compress the data. In the case of sound, the Fourier spectrum of recorded sound will typically only have a few important components. We can approximate the signal well using only these

few components, and setting the rest to zero, regardless of their value. This concept is called *sparsity*, and a signal where almost all the coefficients are zero is called sparse. The MP3 audio standard uses this feature to compress the signal to only a fraction of the original size. Similar considerations are made in many modern signal compression algorithms. This process is illustrated in figure 1.2

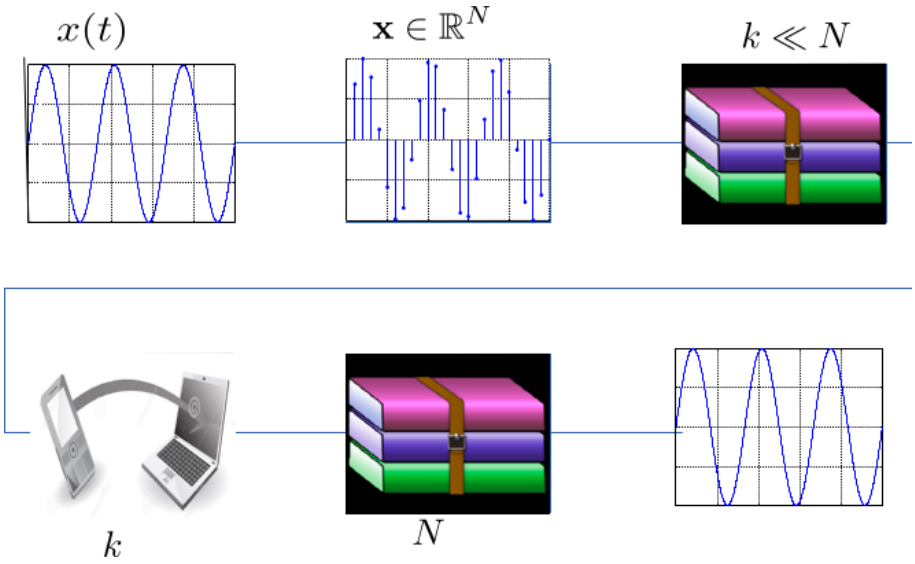


Figure 1.2: The classical process of signal acquisition. The signal is recorded at a very high frequency, then transformed and compressed, before it is stored. The signal can then later be decompressed and viewed.

This, however, seems wasteful. We collect a huge amount of data, only to throw away upwards of 90% afterwards, without a notable loss of quality. Clearly, Shannon's theorem does not tell the whole story, as we can represent the signal with far fewer samples than that required by the theorem, as long as we represent it in the proper basis. Ideally, we would be able to find these important coefficients right away, without the need for such through sampling. In the case of audio sampling this need is perhaps not obvious, however, in areas such as medical and seismic imaging, where the same principles apply, each sample is very costly, and being able to reduce the number of samples needed would be highly advantageous.

Compressed sensing is a rapidly emerging new field in signal processing, aiming to address this problem [1]. The key concept is that we might be able to select the few interesting coefficients of a signal in the proper basis if we make fewer measurements, but make them in a clever way. This can be illustrated well with the well known riddle often called the "12 coins" problem.

> There is a pile of twelve coins, all of equal size. Eleven are of equal weight, while one is counterfeit and is of a different weight. In three weighings on a balance scale, find the counterfeit coin and determine if it is heavier or lighter.

This problem has several solutions. Typically, any solution would start by comparing 4 coins on each side of the scale (one would perhaps guess that starting with 6 coins on each side is better, but as you do not known if the counterfeit coin is heavier or lighter than the other, this provides no information). One can even find a non-adaptive solution, a solution where the

result of one weighing does not dictate what coins to weigh next. The key to this solution is that counterfeit property is sparse in the "signal" of coins. Through a clever combination of weighings, we are then able to locate this sparse property of the signal. Compressed sensing formulates the mathematics of what makes weighings, or more generally measurements of a system, clever. The results are formidable; it turns out that the number of measurements needed to sample a sparse signal and give an exact reconstruction is mainly proportional to the number of non-zero samples, rather than the length of the signal itself. As we will signal $\mathbf{x}$ of length N only has k non-zero coefficients, then the number M of measurements needed to restore the signal is bounded by

$$M \geq Ck \log(N/k). \tag{1.1}$$

The measurements are linear, and are described by an $M \times N$ matrix A. The measurement are then stored in the vector $\mathbf{y}$ of length M as

$$\mathbf{y} = A\mathbf{x}. \tag{1.2}$$

Once the measurements are made, we want to restore the signal by finding the sparsest possible $\mathbf{x}^{\#}$, consistent with the measurements, i.e. such that $A\mathbf{x}^{\#} = \mathbf{y}$. The signal acquisition is a success if this restored signal is the same as the original, $\mathbf{x}^{\#} = \mathbf{x}$. As it turns out, the best way to ensure this is to minimize the ℓ_1-norm of $\mathbf{x}^{\#}$, $\|\mathbf{x}\|_1 = \sum_i |x_i|$.

While much theory has been developed for CS, there are still many unexplored applications of CS waiting to be utilized. The goals of this thesis will be the following:

1. Develop a solid understanding of the CS theory and apply it to simple one-dimensional time series.

2. Understand how to apply CS in a numerical setting to large 2-D datasets, and find suitable existing numerical packages for large-scale CS (developing code for numerically stable and efficient ℓ_1-minimization is unfeasible given the time scope of this thesis).

3. Investigate, understand and reproduce existing applications of CS in computational physics, such as MRI [2], molecular dynamics [3], [4], and restoration of seismic data [5].

4. Suggest new possible uses for CS in computational physics.

1.1 Structure of the thesis

This thesis contains a lot of theory! My hope is that someone with a physics background at a grad student or late undergrad level, wanting to develop an understanding for compressed sensing, will find what they need here to become fairly operative in CS. This requires quite a lot of theory to be developed, as most physics student have had only a minimal exposure to basis representations, which is at the heart of CS theory and applications.

The thesis is roughly structured by the following main points.

1. The second chapter contains a minimal introduction to digital sampling. Many reader may be familiar with the results of this chapter, but it serves as a nice backdrop for introducing some concepts and definitions which we will get back to later.

2. The third chapter deals with basis representations. First we consider Fourier series of continuous time signals, which again should be familiar to most physics students. We then quickly turn to the discrete Fourier transform, which is of more interest to us.

3. Next, we spend some time on the Wavelet transform. This will most likely be new to someone with a background in pure physics, as even someone who has met wavelets in physics previously, will find this presentation, rooted in signal processing, new to them. We also look at some new bases with interesting properties.

4. Chapter 4 begins with an introduction to linear inverse problems. Someone looking to get the gist of CS may want to read the first section of chapter 4 right away, and then follow up with chapter 5.

5. Next, several sections are spent on the theory of CS. This may be skipped on a first reading. Finally, section 4.12 will go through how to implement a CS problem in MATLAB.

6. The final chapter is divided in two parts; the first part, sections 5.1 through 5.4, highlights notable previous work made with CS in physics. The second part goes through original work made during work on this thesis.

1.2 My contributions

This thesis as a whole acts more like a review of the field of CS than it does a large body of original work. However, there are original ideas and results which I would like to highlight, which has not previously been explored in the literature. The terminology here might be unfamiliar for the reader, but will be explained in the thesis.

- **Random orthogonal bases for measuring 2-D signals**. We will see later that one of the best ways to measure a signal for CS is to make completely random measurements on the system. However for 2-D signals this is not feasible both in terms of memory usage and numerical complexity. Using direct products of random matrices produces poor results. My advisor, Øyvind Ryan, introduced the idea of using orthogonal random bases, and sensing using subsets of these. This turned out to work very well, but does not seem to be explored in the literature. In section 4.9.2 i have included some theoretical and experimental results for such bases. Notably, corollary 4.9.7 gives a better restoration guarantee for direct products of random measurements than what is previously known in the literature.

- **Restoration of Wavelet detail spaces using curvelet sparsity**. The detail wavelet spaces usually contain information to keep abrupt changes in a signal. These abrupt changes are governed by continuous curves, and should be well representable with curvelets. I have experimented with filling in missing detail spaces by finding the sparsest possible curvelet solution consistent with the low-resolution space. This can be considered a form of non-adaptive "super resolution", and produces better results than the traditional cubic spline method.

- **Estimation of Hurst exponents from partial measurements of rough surfaces**. We did some work in the restoration of rough surfaces based on partial samples. We were

interested in the ability to estimate the Hurst exponent based on only a small amount of measurements. We developed a method for estimating the Hurst exponent based on only the first measurements of a time series by generating a dictionary of rough surfaces.

- **Seismic data restoration based on measurements of varied resolution**. Some work has already been made in the area of restoration of partial seismic data. Here, we consider the case where we have partial measurements of different resolutions, which has not been explored earlier.
- **Seismic restoration based on learned dictionaries**. Finally, I approached the problem of seismic restoration using learned dictionaries, achieving better results than with methods that have previously been showcased in articles.

1.3 Code

The reference to code in this thesis in minimal. This has the advantage of making the thesis near independent of any particular programming language. There are some minimalist examples where the code will make the concept more clear. Relevant code producing the figures and result found in this thesis can, for the most part, be found at the GitHub reposity created for this project, which can be found at https://github.com/andreavs/Compressed-Sensing-code. If you want some code which for some reason is not available there, or you have questions about the code, please contact me at `a.v.solbra@fys.uio.no`.

Chapter 2

Classical sampling theory

To understand why Compressed Sensing (CS) is a powerful tool, we should first review traditional sampling theory. The first section in this chapter will quickly go through the classical results, and the rest of the sections will go in depth on the theory of CS.

Most signals encountered in physics are continuous functions either in space or time (or both). Newton's laws of physics deals with time as a continuous variable. When we want to simulate physical systems on a computer, we do not have infinite precision, and so we must find some approximation to this. The solution is most often to discretized the signal in time. Examples of this encountered in everyday life are plentiful; think of digitally recorded music, digital images, which are discretized in pixels, and videos, which are discretized in both time and space. Most often, the discretized signal is made by measuring the signal at with a fixed interval, and the measurement are called samples. Let us summarize this.

Fact 2.0.1. Continuous signals, when recorded digitally, are discretized. A signal $x(t)$ is recorded as a vector $\mathbf{x}$ with elements $x_i = x(t_i)$. Most often, the signal is recorded at with a fixed interval, i.e. $t_i = i\Delta t$. In this case, the number of samples per second is called the **sample rate**, which we will denote by f_s (this is the most common conversion, and comes from the term "sampling frequency", which is just another term for sample rate) can be calculated as

$$f_s = \frac{1 \text{ sample}}{\Delta t}. \tag{2.1}$$

This will is a good starting point for our discussion. In order to dive deeper into the concepts of digital signal processing (DSP), it will be useful to consider an example in some detail. In order to make this approachable for most people, I will here use sound signals as a common example. Even so, most of the results are easily transferable to any other type of physical signal.

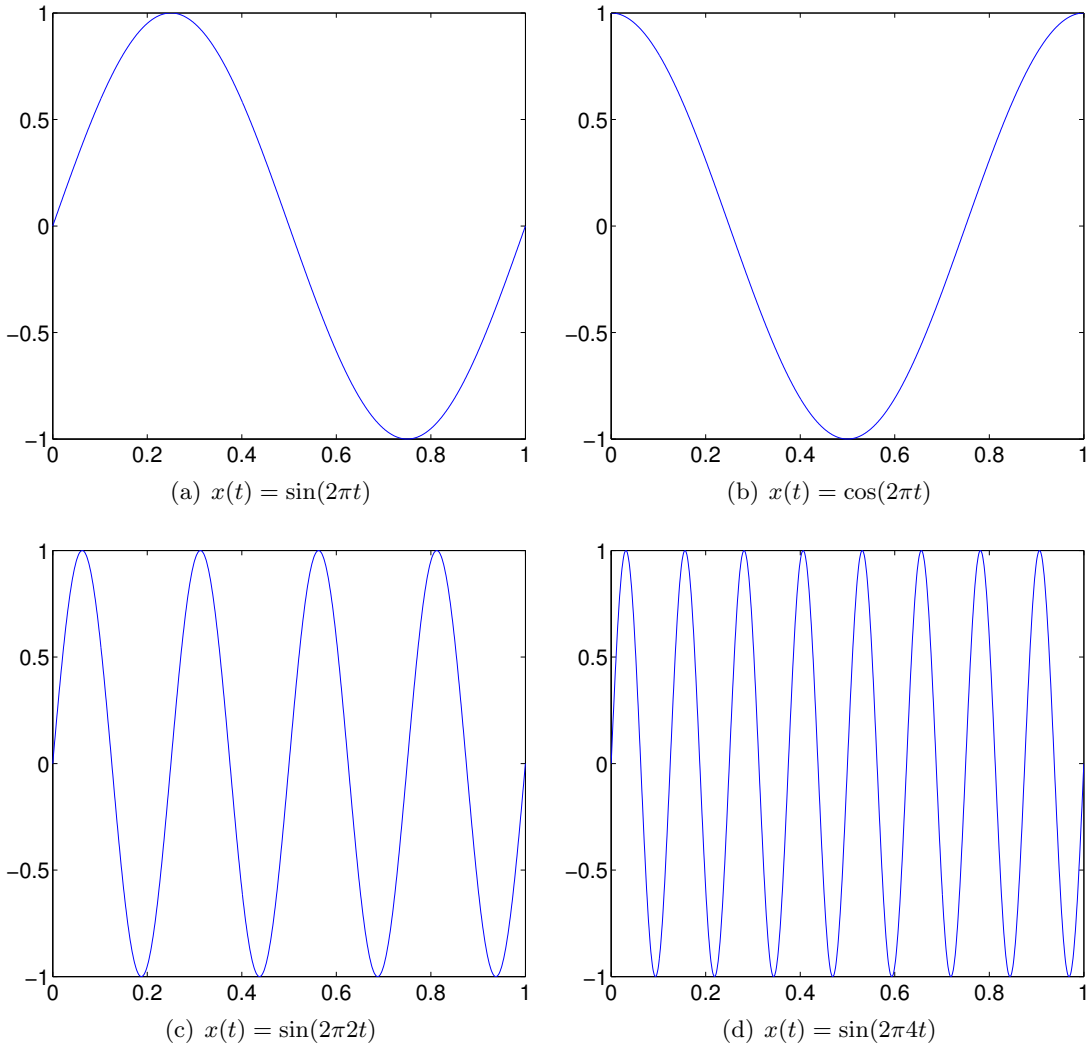
2.1 Sound

Physically, sound is variations in air pressure, which propagate through the air and reaches a listener. The variations cause muscles in the ear to resonate with the frequency of the air pressure, and this is how we perceive sound. The human ear can detect these vibrations as long as they are in the range 20-20 000 Hz. Before we go any further, it is useful to properly define some properties which will be useful in this section.

Definition 2.1.1 (Periodicity and frequency)**.** A function $x(t)$ is said to be periodic with period P if

$$x(t+P) = x(t). \tag{2.2}$$

for all t. The function $\sin(2\pi ft)$ is periodic with period $1/f$. In one unit of time the function repeats itself f times. We define f to be the frequency of $y(t)$. If f is an integer, a sine or cosine of this type is type is also called a pure tone.



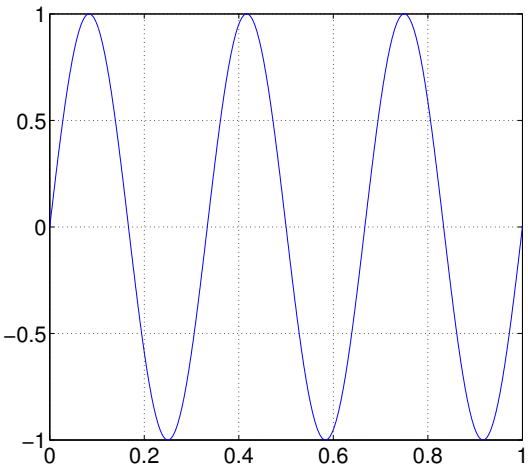
(a) $x(t) = \sin(2\pi t)$ (b) $x(t) = \cos(2\pi t)$

(c) $x(t) = \sin(2\pi 2t)$ (d) $x(t) = \sin(2\pi 4t)$

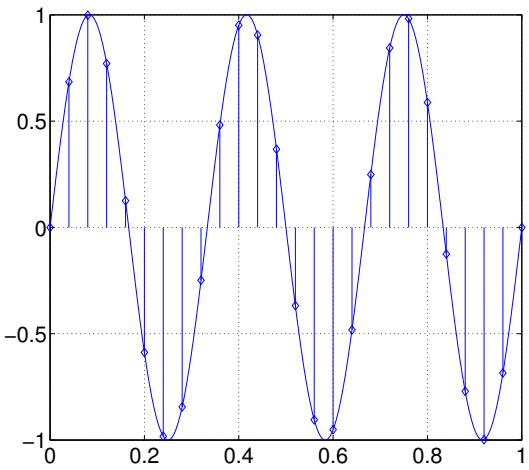
Figure 2.1: The figures show some examples of pure tones, as defined in Definition 2.1.1

Figure 2.1 shows a few pure tones. With these definitions in place, we can begin to consider digital sound. For now, we will consider the sound to be sampled at a constant rate. Sound can be stored digitally in several different ways, but the format will not be important in this text, so we will here use a minimalist definition.

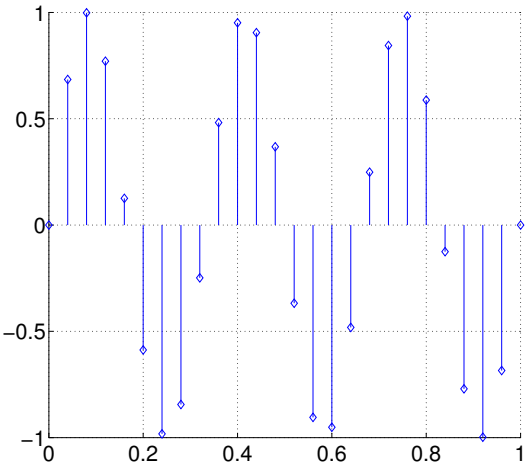
Definition 2.1.2. A digital sound signal $\mathbf{x}$ is an N-dimensional vector with sampled values from some sound signal $x(t)$, with elements $x_i = x(t_i)$. Normally, the sample rate is constant, i.e. $t_i = i\Delta t = 1/f_s$, where f_s is the sample rate.



(a) The initial signal, $x(t) = \sin(2\pi 3t)$



(b) The signal is sampled at a constant rate



(c) The samples are stored in a vector $\mathbf{x}$

Figure 2.2: The figures show the process of sampling as described in Definition 2.1.2

The process of sampling is illustrated in figure 2.2. It seems reasonable that if the signal is sampled at a high enough rate, we will be able to reproduce the original non-digital signal from the sample vector. It turns out that if we use a sample rate higher than two times the highest frequency in the signal, we can in fact reproduce the original signal exactly. This is called the Shannon-Nyquist sampling theorem, and we will get back to this in the next section. For now we will look at a brief example of the phenomenon called *aliasing*, which occurs when the sample rate is too low.

Example 2.1.3 (Aliasing)**.** We consider the signal $x(t) = \sin(2\pi 8t)$ for one second with a sampling rate $f_s = 10$. This is illustrated in figure 2.3(a). The samples will then be $x_i = \sin(2\pi 8i/10)$. We now note that from the trigonometric rules $\sin(-t) = -\sin(t)$ and $\sin(t) = \sin(2\pi + t)$, we get

$$\begin{aligned} x_i &= \sin(2\pi 8i/10) && (2.3)\\ &= -\sin(-2\pi 8i/10) && (2.4)\\ &= -\sin(2\pi i - 2\pi 8i/10) && (2.5)\\ &= -\sin(2\pi(10i - 8i)/10) && (2.6)\\ &= -\sin(2\pi 2i/10). && (2.7) \end{aligned}$$

And so, based on the collected samples these two signals are identical. The reconstruction process will then choose the lowest frequency solution, as shown i figure 2.3(c)-2.3(d). In general, for uniformly sampled signals. We have that for any frequency $f < f_s$, the samples will overlap with the samples of a signal with frequency $f_s - f$, i.e.

$$\sin(2\pi fi/N) = -\sin(2\pi(f_s - f)i/N). \tag{2.8}$$

♣

Now that we have solidified the basic concepts of discrete signals, we will turn to the topic of basis representation, which is needed to understand compressed sensing, which is closely tied to representing your signal in the proper basis.

2.2 Images

In much the same way that sound consists of air vibrations in a certain frequency range, which are interpreted by our ears, images are electromagnetic waves in the frequency range 430-790 THz. Images can also be stored digitally, but the story is a bit more complicated, as we cannot store only a single value per point in space. Many colors, such as pink or purple, can not be represented by a light with a single frequency, but require a mixture of colors. Colors containing only one wavelength are called *pure colors*. Typically, the image is stored with different intensities of 3 different colors, and these are typically chosen to be red, green and blue. We can however, store only the intensity of light at a given point, which gives black-and-white images.

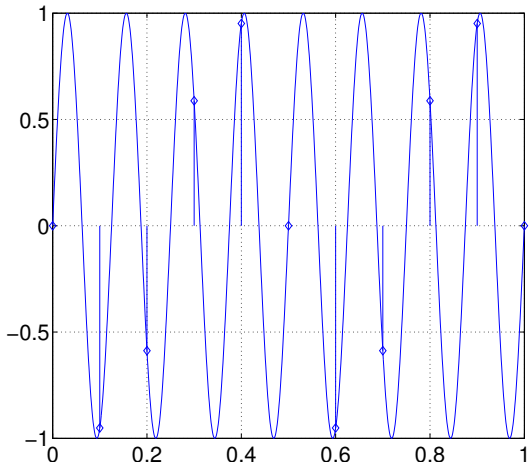
Definition 2.2.1 (Digital images)**.** A black-and-white digital image is a two-dimensional $N \times M$ array of numbers, corresponding to measurements of light intensity at different solid angles from the measurement device. The array may consists of integers in a given range or floating point numbers numbers, typically in the range $[0, 1)$.

A digital color image is usually a three dimensional $N \times M \times 3$ array, i.e. 3 two-dimensional $N \times M$ arrays. Each two dimensional arrays contains the measured light intensity in a specific frequency range.

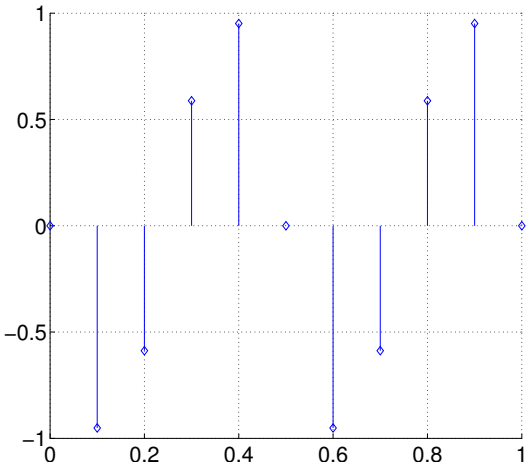
Images are widely using in DSP literature as test cases for proposed methods. Often the same few images are used, which makes it easier to compare different methods across articles.

Some of the most used images are shown in figure 2.4. A large library of common images used in articles can be found at the USC-SIPI image database [6]. We will use these images several times throughout this thesis.

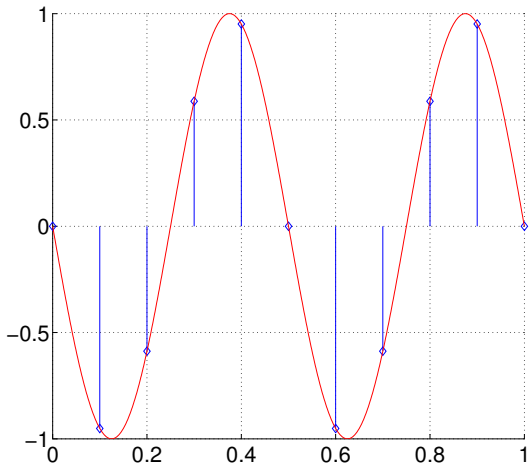
Usually black-and-white images are preferred to color images in DSP literature. Partly for printability of articles, and partly because a method for black-and-white images can be extended to color images simply by applying the method separately to the different color layers.



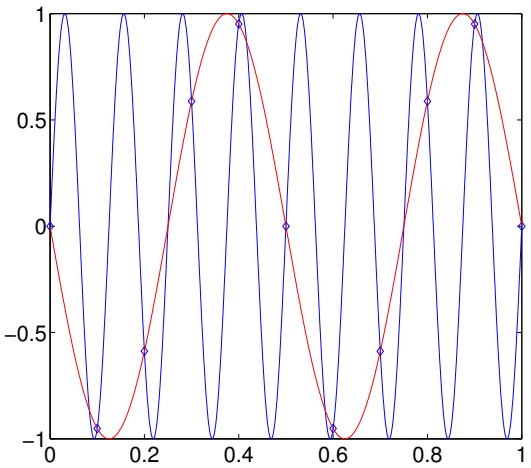
(a) The initial signal, $x(t) = \sin(2\pi 8t)$, sampled at $f_s = 10$.



(b) The sample vector.



(c) Reconstructed (wrong) signal.



(d) The two signals superimposed.

Figure 2.3: The figures illustrate how an undersampled signal will be erroneously reconstructed. See example 2.1.3 for details.

Figure 2.4: Some of the most common test images used in litterature. Left to right,tTop to bottom: “Lena”, “Peppers”, “Mandril”, “Cameraman”, “Boat”, “Straws” and “Modified Shepp-Logan Phantom.

Chapter 3

Basis Representations

In the introduction, the sound and image signals were represented in the time and position basis, respectively. The basis which represent the signal in the most expected manner, like the above, is often referred to as the *canonical basis*. Other basis representations may make important features of the signal more apparent, and are at the heart of DSP. Here we will introduce some of the most used basis representations apart from the canonical representation.

3.1 Representation of sound in terms of pure tones, Fourier transform

We begin our review with a representation which should be somewhat familiar to physicists, the Fourier Transform. We begin by introducing the transform for continuous signals, but for us the most important version will be the Fourier transform as applied to digital signals.

3.1.1 Fourier series

In our discussion of sound, we only considered pure tones. The following theorem will show that this is in fact sufficient, and is an extremely important result. We begin with a definition which will make make the theorem more clear.

Definition 3.1.1 (Fourier Series)**.** Let $y(t)$ be a function defined on the interval $[0, 1)$. We want to write $x(t)$ as a sum of pure complex tones, $1, e^{2\pi it}, e^{-2\pi it}, \ldots$.

$$x(t) = \sum_{n=-\infty}^{\infty} c_n e^{i2\pi n}. \tag{3.1}$$

This representation is called the Fourier series of $x(t)$.

One can show that the Fourier coefficients, $\{c_n\}$, are given by

$$c_n = \int_0^1 y(t) e^{-2\pi it}\, dt. \tag{3.2}$$

This formula will be explained in section 3.2. The following theorem will show that this series actually converges to $x(t)$. The formulation of the theorem is from [7].

Theorem 3.1.2 (theorem of Dirichlet)**.** If $x(t)$ is periodic with period 1, and if between 0 and 1 it has a finite number of discontinuities and if $\int_0^1 |x(t)|\,dt$ is finite, the the Fourier series converges to all points where $x(t)$ is continuous, and converges to the midpoint of the jump for all other points.

For an approachable proof of this theorem, see [8], and for an even more approachable discussion, see [7]. We turn to an example of why such a representation would be useful.

The energy of a signal and Parseval's Theorem

Consider an electric signal in the form of a sine function, $I(t) = A\sin(\omega t)$, where A is the amplitude of the signal, and ω is the frequency. The power over a resistor is then given by $P(t) = R|I(t)|^2$. This is the time-density of the energy output, in the sense that if we want to find total energy output of a period of time T, we integrate the power over time,

$$E = \int_0^T P(t)\,dt. \tag{3.3}$$

Next, we consider a more complicated signal, $I = I(t)$, but require the signal to be periodic with period 1. We know that the signal can be written by its Fourier series, $I(t) = \sum_k c_k e^{2\pi ik}$. We then try to find the energy of the signal,

$$\begin{aligned}
E &= \int_0^1 P(t)\,dt && (3.4)\\
&= \int_0^1 R\left(\sum_k c_k e^{2\pi ik}\right)\left(\sum_j c_j^* e^{-2\pi ij}\right)\,dt && (3.5)\\
&= R\int_0^1 \sum_k \sum_j c_k c_j^* e^{2\pi i(k-j)}\,dt && (3.6)\\
&= R\sum_k \sum_j c_k c_j^* \int_0^1 e^{2\pi i(k-j)}\,dt && (3.7)\\
&= R\sum_k \sum_j c_k c_j^* \delta_{jk} && (3.8)\\
&= R\sum_k |c_k|^2, && (3.9)
\end{aligned}$$

So we see that the squares of the fourier coefficients shows us what amount of power is in the signal as the different frequencies: it can be interpreted as the frequency density of the power. This result is a direct result from a much more general theorem called Parseval's theorem.

Theorem 3.1.3 (Parseval's Theorem)**.** Suppose two functions $A(t)$ and $B(t)$ have the Fourier series

$$A(t) = \sum_n a_n e^{2\pi i n t}, \tag{3.10}$$

and

$$B(t) = \sum_n b_n e^{2\pi i n t}, \tag{3.11}$$

respectively, then

$$\sum_n a_n b_n^* = \int_0^1 A(t)B(t)^*\, dt. \tag{3.12}$$

PROOF. This follows from the orthogonality of the exponential functions. □

If we set $A = B$ then we obtain our result directly from the theorem.

3.1.2 Discrete Fourier transform

This forms the background for approximating continuous signals with pure tones. However, once the signals we want to analyse have already been discretized, and so we will focus on methods for discrete signals. With that, let us now define a fourier transform for vectors. We want a transform which is similar to the continuous case, however, our signal only has values at a finite set of points $y(0), y(1/N), \ldots, y((N-1)/N)$. Because of this, we only need a finite set of pure tones to determine a vector that passes thorugh all these points. We choose these to be $1, e^{2\pi i t}, e^{2\pi i 2t}, \ldots, e^{2\pi i (N-1)t}\}$. Now we can define the Discrete Fourier basis.

Definition 3.1.4 (Discrete fourier basis)**.** The N-dimensional discrete Fourier basis for a sample vector $\mathbf{x}$, is the functions $\{1, e^{2\pi i t}, e^{2\pi i 2t}, \ldots, e^{2\pi i (N-1)t}\}$ sampled at the points $t_k = k/N$ for $k = 0, \ldots, N-1$, i.e. We also want the vectors to be normalized with respect to the normal inner product, which gives a prefactor of $1/\sqrt{N}$ (this will be explained more in section 3.2). This gives the vectors

$$\begin{aligned}
\mathbf{y}^{(0)} &= \frac{1}{\sqrt{N}}\left(1, \quad 1, \quad 1, \quad \ldots, \quad 1\right)^T, \\
\mathbf{y}^{(1)} &= \frac{1}{\sqrt{N}}\left(1, \quad e^{2\pi i/N}, \quad e^{2\pi i 2/N}, \quad \ldots, \quad e^{2\pi i (N-1)/N}\right)^T, \\
\mathbf{y}^{(2)} &= \frac{1}{\sqrt{N}}\left(1, \quad e^{2\pi i 2/N}, \quad e^{2\pi i 4/N}, \quad \ldots, \quad e^{2\pi i 2(N-1)/N}\right)^T, \\
&\vdots \\
\mathbf{y}^{(N-1)} &= \frac{1}{\sqrt{N}}\left(1, \quad e^{2\pi i (N-1)/N}, \quad e^{2\pi i (N-1)/N}, \quad \ldots, \quad e^{2\pi i (N-1)(N-1)/N}\right)^T,
\end{aligned} \tag{3.13}$$

We want to find coefficients the coefficients $\{y_i\}_{i=0}^{N-1}$ such that

$$x(t_i)x_i = y_0 \mathbf{y}_i^{(0)} + y_1 \mathbf{y}_i^{(1)} + y_2 \mathbf{y}_i^{(2)} + \cdots + y_{N-1}\mathbf{y}_i^{(N-1)}. \tag{3.14}$$

for all $i = 0, \ldots, N-1$. This gives us N equations for the N unknown coefficients. We can collect these into a matrix equation of the form

$$\mathbf{x} = G\mathbf{y}, \tag{3.15}$$

where $G_{jk} = \mathbf{y}_j^{(k)} = \frac{1}{\sqrt{N}} e^{2\pi ijk/N}$. This leads us directly to the next result

Theorem 3.1.5 (discrete fourier transform)**.** The Discrete Fourier transform (DFT) of a sample vector $\mathbf{x}$ is given by

$$\mathbf{y} = F_N \mathbf{x}, \tag{3.16}$$

where F_N is the N-dimensional Fourier matrix with elements $(F_N)_{jk} = e^{-2\pi ijk/N}$.

Proof. Note that the matrix G as defined in eq (3.15) is orthogonal (this is easy to check), so its inverse is simply $G^\dagger$, the hermittian conjugate. □

The inverse discrete Fourier transform (IDFT) is given by $F^\dagger$, $\mathbf{x} = F_N^\dagger \mathbf{y}$. At this point we now how to perform a DFT, but how should we interpret it? Consider a normalized vector in $\mathbb{R}^3$, represented with the usual cartesian coordinates. The size of each vector component tells you roughly how much your vector "looks like" that particular basis vector. The same is true for vectors represented in the fourier basis. The more the vector "looks like" a sine/cosine with frequency n the larger the imaginary/real parts of the n-th fourier coefficient will be. We can check this with the following simple example.

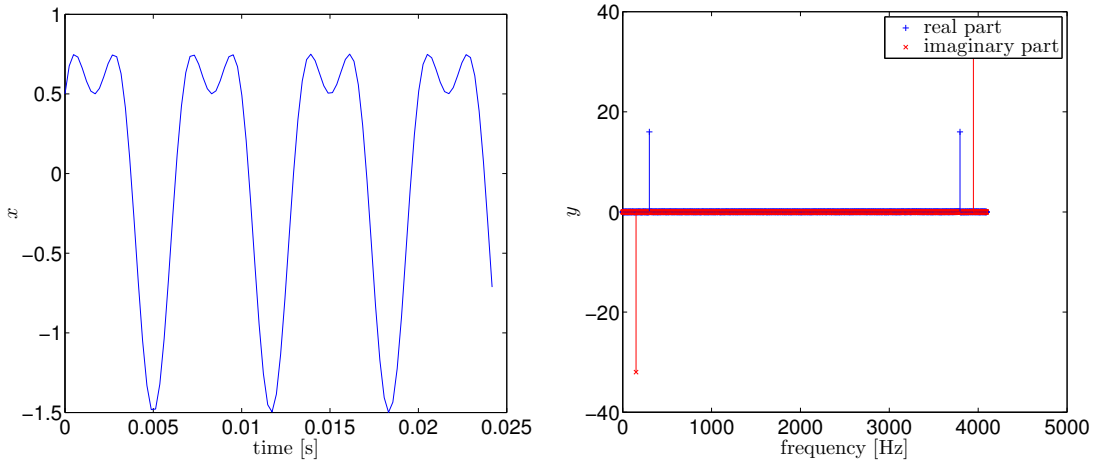
Example 3.1.6 (a simple example of DFT)**.** We want to study the fourier transform of

$$x(t) = \sin(2\pi 150t) + \frac{1}{2}\cos(2\pi 300t) \tag{3.17}$$

We store this signal as a vector with a sampling frequency of $f_s = 4096$. We expect the only fourier coefficients to p an imaginary part at $f = 150$ and a real part at $f = 300$. We also expect the real part to be about half the size of the imaginary part. The result is shown in figure 3.1(b), and agrees with our expectation. The figure also shows some other common representations of the fourier spectrum. 3.1(c) shows the absolute value of the fourier coefficient. This treats the frequencies equally independent of phase, which makes it well suited for investigating real sound signals. This is called the frequency spectrum. Figure 3.1(d) shows the square of the absolute values. This is a common way to represent physical signals, because this will tell us the frequency distribution of the power, as seen in section 3.1.1. For this reason, the final representation is often called the (frequency) power spectrum.

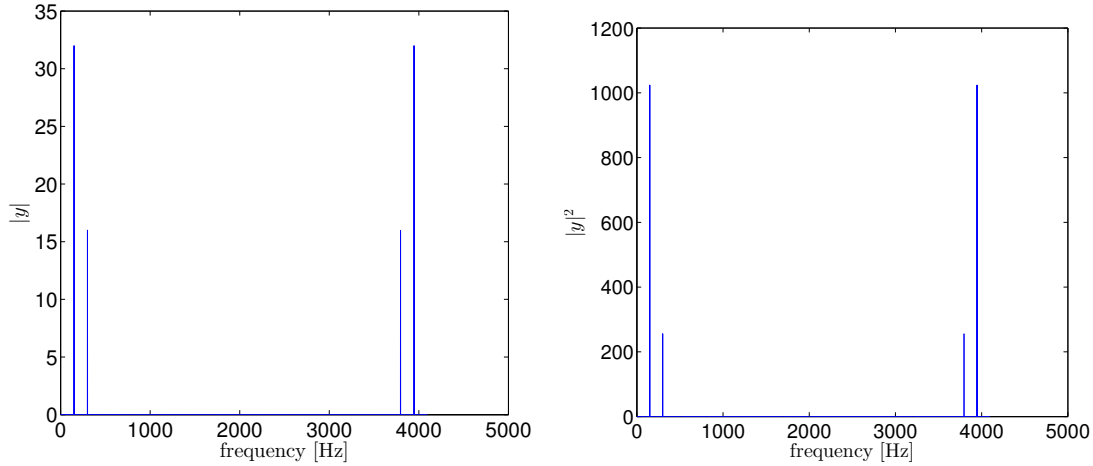
We see that in addition to the expected frequency spikes, there are similar spikes on the opposite end of the spectrum. This is an effect of aliasing, as mentioned in example 2.1.3. For this reason, only the first half of the Fourier coefficients are shown in many cases, but we include the entire Fourier spectrum here for clarity.

♣



(a) The time signal (zoomed in)

(b) Fourier components



(c) Frequency spectrum

(d) Frequency power spectrum

Figure 3.1: The figures 3.1(b)-3.1(d) show some different representations of the fourier spectrum of the time signal showed in figure 3.1(a)

3.1.3 The Fast Fourier Transform

Here we will introduce a very efficient algorithm for performing the DFT. In order to put this into the proper context, we should define the most common way to profile the efficiency of an algorithm, the *leading order*.

Definition 3.1.7 (Leading order)**.** Let A be an algorithm to be performed on some signal $\mathbf{x}$ of length N. The algorithm will will require some number K of operations, typically dependent on N, $K = K(N)$. We define the leading order of A is the fastest growing term in $K(N)$. If we denote functional form the fastest growing term by $f(N)$, i.e. the dominant term as $N \to \infty$, but normalized to remove any constant in front of it, then we denote the leading order of A as $\mathcal{O}(f(N))$. This is also sometimes referred to as the *complexity* of A.

As an example, solving a system of N linear equations by Gaussian elimination is known to require $N(N-1)/2$ divisions, $(2N^3+3N^2-5N)/6$ multiplications and $(2N^3+3N^2-5N)/6$ subtractions, for a total of $N(4N^2+9N-13)/6$ operations. The largest term in this expression is $2N^3/3$, which makes the leading order $\mathcal{O}(N^3)$. The multiplication of an $N \times N$ matrix with an N-length vector has order $\mathcal{O}(N^2)$, while the multiplication of two $N \times N$ matrices is of order $\mathcal{O}(N^3)$.

The implementation of the DFT we have mentioned so far is of order $\mathcal{O}(N^2)$, as we have presented it as a matrix-vector multiplication. The following theorem will be the key to greatly reducing this order, and its implementation is known as the Fast Fourier Transform (FFT). The FFT is an example of a *divide and conquer algorithm*. Such algorithms work by recursively breaking down a problem into two or more sub-problems of the same (or related) type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem.

Theorem 3.1.8 (FFT algorithm)**.** Let $\mathbf{x}$ be a vector of length N where N is even and let $\mathbf{y} = F_N$ be its DFT. If we denote the even entries of $\mathbf{x}$ by $\mathbf{x}^{(e)}$, i.e., $\mathbf{x}^{(e)} = (x_0, x_2, \ldots, x_{N-2})^T$, and similarly denote the odd elements of $\mathbf{x}$ by $\mathbf{x}^{(o)}$, then the elements of $\mathbf{y}$ are for any n in the interval $[0, N/2-1]$ given by

$$y_n = \frac{1}{\sqrt{2}}\big((F_{N/2}\mathbf{x}^{(e)})_n + \exp(-2\pi i n/N)(F_N\mathbf{x}^{(o)})_n\big), \tag{3.18}$$

$$y_{N/2+n} = \frac{1}{\sqrt{2}}\big((F_{N/2}\mathbf{x}^{(e)})_n - \exp(-2\pi i n/N)(F_N\mathbf{x}^{(o)})_n\big) \tag{3.19}$$

PROOF. This is proved using simple algebra. We split the expression of y_n into expressions involving $\mathbf{x}^{(e)}$ and $\mathbf{x}^{(o)}$:

$$y_n = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} x_k e^{-2\pi i n k/N} \tag{3.20}$$

$$= \frac{1}{\sqrt{N}} \sum_{k=0}^{N/2-1} x_{2k} e^{-2\pi i n 2k/N} + \frac{1}{\sqrt{N}} \sum_{k=0}^{N/2-1} x_{2k+1} e^{-2\pi i n (2k+1)/N} \tag{3.21}$$

$$= \frac{1}{\sqrt{2}} \left(\frac{1}{\sqrt{N/2}} \sum_{k=0}^{N/2-1} x_{2k} e^{-2\pi i n k/(N/2)} + e^{-2\pi i n/N} \frac{1}{\sqrt{N/2}} \sum_{k=0}^{N/2-1} x_{2k+1} e^{-2\pi i n (k)/(N/2)} \right) \tag{3.22}$$

$$= 1\sqrt{2} \left((F_{N/2} \mathbf{x}^{(e)})_n + e^{-2\pi i n/N} (F_{N/2})_n \right). \tag{3.23}$$

The expression for $y_{N/2+n}$ is derived similarly. □

If we use this expression to calculate $\mathbf{y}$, then rather than the N^2 operations needed for a regular matrix-vector multiplication, we need $2(N/2)^2 + 5N/2$ operations. This is still of order $\mathcal{O}(N^2)$, however, we can repeat this argument when we calculate $F_{N/2}\mathbf{x}^{(o)}$ and $F_{N/2}\mathbf{x}^{(e)}$, and continue to split the DFT calculations in this manner.

If we for simplicity consider only the multiplications in this expression, and also ignore the multiplication of $1/\sqrt{2}$, then it is clear that if we denote the number of operations needed to calculate the DFT by M_N, then

$$M_N = 2M_{N/2} + N/2. \tag{3.24}$$

In the case where $N = 2^n$, we can denote M_{2^n} by m_n then the expression takes the form

$$m_{n+1} = 2m_n + 2^n, \tag{3.25}$$

which is a difference equation with the general solution $m_n = (C+n)2^{n-1}$, which in turn implies that $M_N = N(C + \log_2(N))/2$. This shows that the leading order of the FFT algorithm is given by $\mathcal{O}(N \log_2(N))$. The FFT is possibly the most used algorithm in applied mathematics, and will usually find its way into any list of the most important algorithms created [9]. Several improvements can be made on the version proved here, but they all share the order of this simple version. A MATLAB implementation of the above theorem may look like the following:

```
function y = FFTImpl(x)
    N = length(x);
    if N == 1
         y = x(1);
    else
        xe = x(1:2:(N-1));
        xo = x(2:2:N);
        ye = FFTImpl(xe);
        yo = FFTImpl(xo);
        D=exp(-2*pi*1j*(0:N/2-1)'/N);
        y = [ ye + yo.*D; ye - yo.*D]/sqrt(2);
    end
end
```

One should note that MATLAB has a far more efficient and robust built-in FFT function called `fft(x)`. This is normalized different than the version presented here; if one wants the unitary transformations one should use the calls `fft(x)/sqrt(N)`, and `ifft(x)*sqrt(N)`.

3.2 General Basis representation

We have already developed some sense of Basis representations by considering the Fourier Basis. However, we will encounter some signals that are not very informative to look at in a Fourier basis, but rather in some other basis. We will therefore consider some general theorems for basis representation, which will make future work in new bases less tedious. This section will follow material from [10] closely, but the notation and discussions will be adjusted to our needs.

We begin by properly defining a basis representation.

Definition 3.2.1. Let $\mathcal{B} = \{\mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_{N-1}\}$ be a basis for an N-dimensional vector space V. Since $\mathcal{B}$ is a basis, a vector $\mathbf{x}$ can be written as a linear combination of the basis elements,

$$\mathbf{x} = c_0\mathbf{b}_0 + \cdots + c_{N-1}\mathbf{b}_{N-1}. \tag{3.26}$$

The coefficient vector $(c_0, \ldots, c_{N-1})^T$ is called the $\mathcal{B}$-coordinates of $\mathbf{x}$, and can be written $[\mathbf{x}]_\mathcal{B}$.

Just like for the Fourier basis, we can look at this as Matrix equation,

$$\mathbf{x} = P_\mathcal{B}[\mathbf{x}]_\mathcal{B}. \tag{3.27}$$

where $P_\mathcal{B}$ is given by the basis vectors,

$$P_\mathcal{B} = \begin{pmatrix} \mathbf{b}_0 & \mathbf{b}_1 & \ldots & \mathbf{b}_{N-1} \end{pmatrix}. \tag{3.28}$$

And because of this, we can find the basis representation of $\mathbf{x}$ as

$$[\mathbf{x}]_\mathcal{B} = P_\mathcal{B}^{-1}\mathbf{x}. \tag{3.29}$$

Note that if $P_\mathcal{B}$ is orthogonal, that is, the basis vectors are orthonormal, then inverse of $P_\mathcal{B}$ is simply its hermitian conjugate,

$$P_\mathcal{B}^{-1} = P_\mathcal{B}^\dagger. \tag{3.30}$$

Furthermore, the basis representation of a matrix A in the base $\mathcal{B}$ is

$$[A]_\mathcal{B} = P_\mathcal{B}^{-1}AP_\mathcal{B}. \tag{3.31}$$

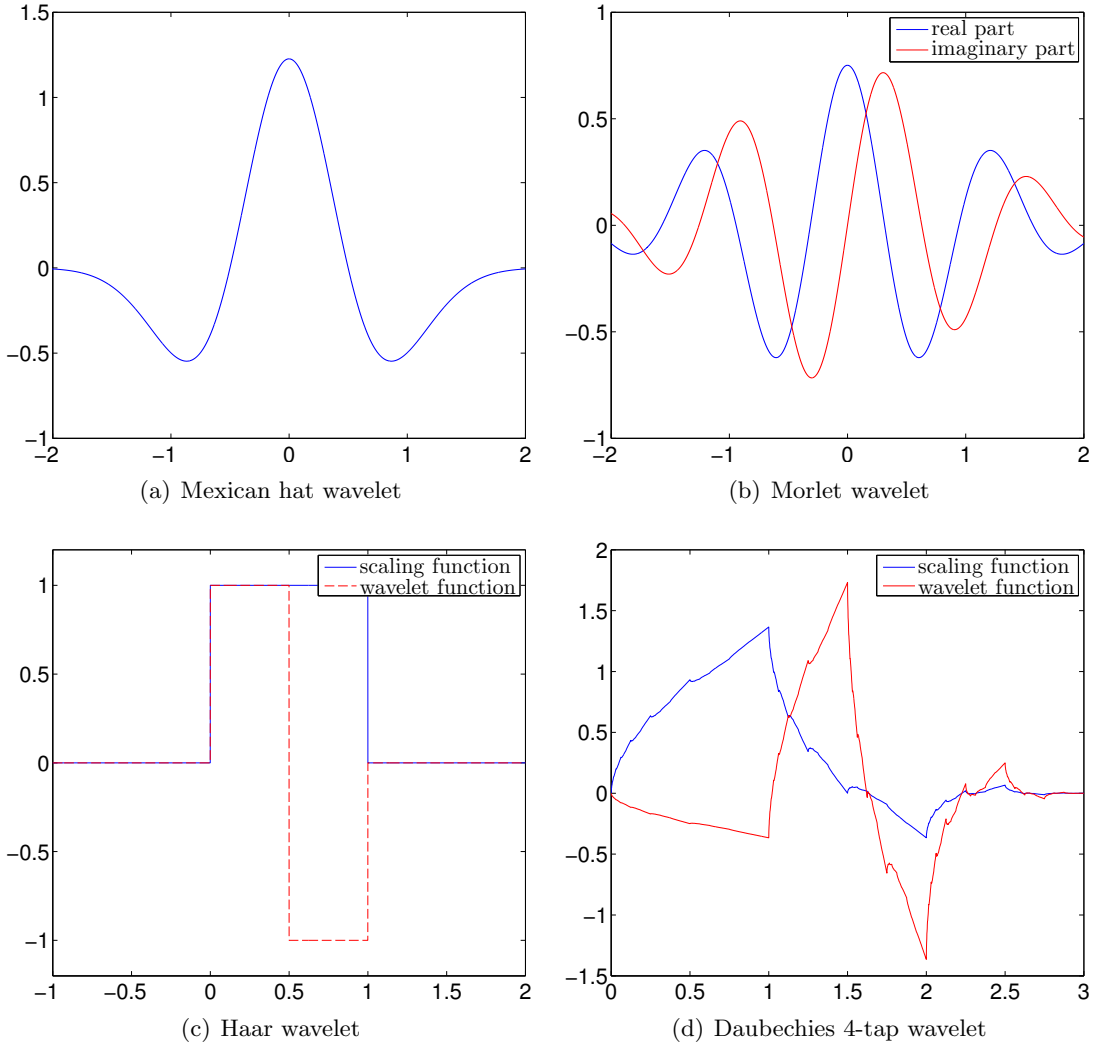
To check this, assume $\mathbf{y} = A\mathbf{x}$. Then

$$\begin{aligned} [\mathbf{y}]_\mathcal{B} &= [A]_\mathcal{B}[\mathbf{x}]_\mathcal{B} && (3.32)\\ &= P_\mathcal{B}^{-1}AP_\mathcal{B}P_\mathcal{B}^{-1}\mathbf{x} && (3.33)\\ &= P_\mathcal{B}^{-1}A\mathbf{x} && (3.34)\\ &= P_\mathcal{B}^{-1}\mathbf{y}, && (3.35) \end{aligned}$$

which means $\mathbf{y}$ gets its proper representation in the $\mathcal{B}$-basis.

3.3 Wavelet bases

In this section we will use the theory developed so far to study another set of bases which will be useful to us later, called *wavelets*. So far, we have seen the regular (Cartesian) basis, which is extremely local in time, but caries no frequency information, and the Fourier basis, which is very dilute in time, but carries maximum frequency information. Wavelets carry a mix of the two, some time information and some frequency information (the name wavelet means "small wave"). Figure 3.2 show some continuous wavelets commonly used in science. Like for the Fourier basis, the discrete wavelets are made by interpolating the continuous wavelets at a discrete set of points $\{1/N, 2/N, \ldots, (N-1)/N\}$. The functions are then turned into a basis by translating them in time.



(a) Mexican hat wavelet

(b) Morlet wavelet

(c) Haar wavelet

(d) Daubechies 4-tap wavelet

Figure 3.2: The figures show some common wavelets.

3.3.1 Continuous Wavelet Transformations

Rather than describing this process in detail, we will focus here on the Haar wavelet. In order to understand this wavelet, we need to get some basic terminology down.

Definition 3.3.1 (Resolution space)**.** Let N be a natural number. The resolution space V_0 is defined as the spaces of function which are constant on each interval $[n, n+1)$ for $n = 0, \ldots, N-1$.

With this in mind, we begin by defining the *scaling function* (sometimes called the *father wavelet*).

Lemma 3.3.2 (Scaling function)**.** We define the scaling function $\phi(t)$ by

$$\phi(t) = \begin{cases} 1 & \text{if } 0 \leq t < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.36}$$

We then let $\phi_n(t-n)$ for $n = 0, \ldots, N-1$. The functions $\{\phi_n\}$ then form an orthonormal basis for V_0 with respect to the inner product

$$\langle f, g \rangle = \int_0^N f(t)g(t)\,dt. \tag{3.37}$$

Now, it is clear that V_0 is not enough to provide a basis for all function, but we will see how we can extend this in a natural way. This will also clarify the name scaling function.

Definition 3.3.3 (Refined resolution spaces)**.** The space V_m is the space of piecewise constant functions on each interval $[n/2^m, (n+1)/2^m$ for $n = 0, \ldots, 2^m(N-1)$. We will construct a basis for this space by using the function

$$\phi_{m,n}(t) = 2^{m/2}\phi(2^m t - n) = \begin{cases} 2^{m/2} & \text{if } n/2^m \leq t < (n+1)/2^m, \\ 0 & \text{otherwise.} \end{cases} \tag{3.38}$$

The following now easily proved.

Lemma 3.3.4 (Properties of resolution spaces)**.** Resolution spaces are nested,

$$V_0 \subset V_1 \subset \cdots \subset V_m. \tag{3.39}$$

and the set $\{\phi_{m,n}\}_n$ converges to a basis of continuous functions when $m \to \infty$.

We can extend V_0 to be equal to V_1 by adding another space. Let us first clarify what this means.

Definition 3.3.5 (Direct sum)**.** The direct sum of two vector spaces $U, V \subseteq W$ is the space of all vectors on the form $\mathbf{u} + \mathbf{v}$ for any $\mathbf{u} \in U$ and any $\mathbf{v} \in V$. It is denoted as $U \oplus V$.

With this definition in mind, here is how we can extend V_0.

Definition 3.3.6 (detail space)**.** Let W_0 be the orthogonal complement of V_0 in V_1, i.e. the subspace of V_1 such that $W_0 \oplus V_0 = V_1$ and $\langle v, w \rangle = 0$ for any $v \in V_0$ and any $w \in W_0$. We call W_0 the *detail space* of V_0.

We then define the function $\psi(t)$ as

$$\psi(t) = \begin{cases} 1 & \text{if } 0 \leq t < 1/2, \\ -1 & \text{if } 1/2 \leq t < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.40}$$

It is clear that $\psi_n(t) = \psi(t - n)$ is a basis for W_0, and we call ψ the *mother wavelet*.

To end this section, we note that if a function f is in V_1, which means it can be written either as

$$f(t) = \sum_{n=0}^{N-1} c_{0,n}\phi_{0,n}(t) + w_{0,n}\psi_{0,n}(t). \tag{3.41}$$

or

$$f(t) = \sum_{n=0}^{2N-1} c_{1,n}\phi_{1,n}(t). \tag{3.42}$$

The relations between these coefficients are

$$\begin{pmatrix} c_{1,2i} \\ c_{1,2i+1} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} c_{0,i} \\ w_{0,i} \end{pmatrix}. \tag{3.43}$$

3.3.2 Discrete Wavelet Transform

The information in the last section was very dense. The good news is that we are now able to define the Discrete Wavelet Transform (DWT) in a meaningful way. First we make a small note about digital signals, which will tie the discrete transform to the continuous one.

If we let $f(t)$ be some signal in V_m, so that $f(t) = \sum_{n=0}^{2^m N} c_n \phi_{m,n}(t)$. Then for any $k \in \{0, 1, \ldots, 2^m N - 1\}$,

$$f(k) = \sum_{n=0}^{N-1} c_n \phi_{0,n}(k) = c_k, \tag{3.44}$$

which means that the basis representation in V_m of a signal $f \in V_m$ can be thought of as a vector of samples from f. Alternately, a sample vector can be though of as the basis representation of f in V_m. This will be our motivation for the following theorem.

Theorem 3.3.7 (Discrete Wavelet Transform)**.** We define the DWT to be the change of coordinates from V_m to $V_{m-1} \oplus W_{m-1}$. The change of coordinates, e.g. the DWT, is given by

$$c_{m-1,n} = (c_{m,2n} + c_{m,2n+1})/\sqrt{2}, \tag{3.45}$$
$$w_{m-1,n} = (c_{m,2n} - c_{m,2n+1})/\sqrt{2}. \tag{3.46}$$

The Inverse Discrete Wavelet Transform (IDWT) is given by

$$c_{m,2n} = (c_{m-1,n} + w_{m-1,n})/\sqrt{2}, \tag{3.47}$$
$$c_{m,2n+1} = (c_{m-1,n} - w_{m-1,n})/\sqrt{2}. \tag{3.48}$$

If we represent the vector in $V_0 \oplus W_0$ as $(c_{m-1,0}, c_{m-1,1}, \ldots, c_{m-1,N-1}, w_{m-1,0}, w_{m-1,1}, \ldots, w_{m-1,N-1})^T$. We can write the change of coordinates matrix as

$$P_{(V_{m-1}\oplus W_{m-1})\leftarrow V_m} = \frac{1}{\sqrt{2}} P \begin{pmatrix} 1 & 1 & 0 & 0 & \ldots & 0 \\ 1 & -1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 1 & & 0 \\ 0 & 0 & 1 & -1 & \ddots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & -1 \end{pmatrix}, \tag{3.49}$$

where P is a permutation matrix which sorts the vector from $(c_{m-1,0}, w_{m-1,0}, \ldots, c_{m-1,N-1}, w_{m-1,N-1})$ to $(c_{m-1,1}, \ldots, c_{m-1,N-1}, w_{m-1,1}, \ldots, w_{m-1,N-1})$. This matrix will take on the form

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 & 0 \\ \vdots & & & & \ldots & & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 1 & 0 \\ 0 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 1 & \ldots & 0 & 0 \\ \vdots & & & & \ldots & & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 0 & 1 \end{pmatrix}. \tag{3.50}$$

A matrix where blocks repeat along the diagonal as in (3.49) is called a *block Toeplitz* Matrix. The term Toeplitz matrix will be defined in the next section.

It may seem strange that we want to order the resulting vector as in (3.50), as it would be simpler to just do the transform. The reason for this ordering is that we can now perform a wavelet transform on the scaling part of the vector, i.e. we will represent the vector in the space $V_{m-2} \oplus W_{m-2} \oplus W_{m-1}$. This can then be repeated multiple times, giving a representation in the space

$$V_0 \oplus W_0 \oplus W_1 \oplus W_2 \oplus \cdots \oplus W_{m-1}. \tag{3.51}$$

This process is generally called a Multi-resolution Analysis (MRA). In this setting is also called an m-level wavelet transform.

The square of the non-P part of the change of coordinate matrix is the identity matrix, so the inverse change of basis matrix is on the form

$$P_{(V_{m-1}\oplus W_{m-1})\leftarrow V_m} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 1 & & 0 \\ 0 & 0 & 1 & -1 & \ddots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -1 \end{pmatrix} P', \tag{3.52}$$

where P' is a matrix which orders the vector back to $(c_{m-1,0}, w_{m-1,0}, \dots, c_{m-1,N-1}, w_{m-1,N-1})$. It has the form

$$P' = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ \vdots & & \dots & & \vdots \\ 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix}. \tag{3.53}$$

The outcome of such a transform is not as easy to interpret as a Fourier Transform, and it will depend on the type of wavelet used. It is useful at this point to look at an example.

Example 3.3.8. For this example we will signal with some structure to highlight the features of the DWT. The signal is the sound file `fiolin.wav`, which can be found at the git repository under `code/sounds`. We use the first 2^{20} samples of this signal, so we might say this is a signal in V_{20}. The signal along with various m-level transforms is shown in figure 3.3. Note that the first half of the signal in figure 3.3(b) is very similar to the signal itself. This is because the first half is the representation of the signal in V_{19}, which is just a coarser representation. We also see that the detail space W_{19} has small components, which means that little information is lost in this approximation of averaging every other samples. In the 2-level DWT, note how the last half of the signal is identical to the 1-level DWT. This is because we represent the signal by breaking it down in the space $V_{18} \oplus W_{18} \oplus W_{19}$, so the last part of the vector should remain untouched as we go to deeper levels of the DWT. Finally, figure 3.3(d) shows the complete 20-level DWT. At this point, only the first component represent the signal approximation, and the rest of the components are from the various detail spaces $W_0, W_1, \dots, W_{19}$. Now there is considerable information stored in the detail spaces. Still, the information is a lot more dense now, in the sense that many of the coefficients are small. ♣

3.3.3 Creating new DWTs

We went through some work on defining the discrete Haar wavelet transform from the continuous transform. In the end, we essentially ended up with two coefficient vectors, $(1,1)^T/\sqrt{2}$ and $(1, -1)^T/\sqrt{2}$. These two vectors are repeated along the diagonal of our transformation matrix. In wavelet and DSP literature, these vectors are called *filters*, and are denoted in a specific way. To appreciate their formulation, we should properly define what is meant by a filter in DSP.

Definition 3.3.9 (digital filter)**.** In DSP, a digital filter S is an operation performed on a sampled, discrete-time signal $\mathbf{x}$. The operation produces a new vector $\mathbf{z}$, element by element, by performing an operation centered around all the different elements of $\mathbf{x}$, one after the other.

An example of a filter may be the procedure

$$z_n = \frac{1}{4}(x_{n-1} + 2x_n + x_{n+1}), \quad \text{for } n = 0, \ldots, N-1. \tag{3.54}$$

This filter is called a smoothing filter, as it has the effect of spreading out sharp peaks in a signal. This filter brings up the point of what to do for the end points z_0 and z_{N-1}, as they are defined in terms of the "elements" x_{-1} and x_N, which do not exist. We solve this by assuming $\mathbf{x}$ to be periodic, which means $x_{-1} = x_{N-1}$ and $x_N = x_0$. We can then write the application of a filter S on a signal $\mathbf{x}$ as the matrix equation $\mathbf{z} = S\mathbf{x}$, where

$$S = \begin{pmatrix} 2 & 1 & 0 & \ldots & 0 & 1 \\ 1 & 2 & 1 & \ldots & 0 & 0 \\ 0 & 1 & 2 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 2 & 1 \\ 1 & 0 & 0 & \ldots & 1 & 2 \end{pmatrix}. \tag{3.55}$$

A matrix where the elements are repeated along each diagonal is called a *Toeplitz matrix*. If the elements also wrap around periodically such as above the matrix is called *circulant Toeplitz*. The application of a general filter S can be written as

$$z_n = \sum_k t_k x_{n-k}. \tag{3.56}$$

where t_k are the *filter coefficients* of S. For the smoothing filter introduced here, $t_{-1} = t_1 = 1/4$, and $t_0 = 1/2$. Using this, we can define a compact notation for filters

Definition 3.3.10 (Compact notation for filters)**.** We can define a compact notation for a digital filter in terms of its filter coefficients. We may write

$$S = \{t_{k_{\min}}, \ldots t_{-1}, \underline{t_0}, t_1, \ldots, t_{k_{\max}}\}, \tag{3.57}$$

where we underline the element along the diagonal of S, and we do not denote the zero-valued filter coefficients outside of the non-zero filter coefficients with the largest indices.

Given a filter matrix S, the compact notation can be found by reading of any column from top to bottom (while keeping the periodic wrap-around in mind). With this notation, the smoothing filter can be written as $S = (1/4)\{1, \underline{2}, 1\}$.

With this in mind, we can define the filters of a DWT

Definition 3.3.11 (filters of a DWT.)**.** We define the filters of a DWT, H_0 and H_1, to be defined by (uniquely) the filter coefficients along the first and second column of the DWT matrix. Conversely, we define the filters of an IDWT, G_0 and G_1, to be defined by the filter coefficients along the first and second column of the IDWT matrix.

With this definition, the filters of the Haar DWT are given by

$$\begin{aligned} H_0 &= \frac{1}{\sqrt{2}}\{1, \underline{1}\}, \\ H_1 &= \frac{1}{\sqrt{2}}\{\underline{-1}, 1\}, \end{aligned} \tag{3.58}$$

while the filters of the Haar IDWT are given by

$$\begin{aligned} G_0 &= \frac{1}{\sqrt{2}}\{\underline{1}, 1\}, \\ G_1 &= \frac{1}{\sqrt{2}}\{\underline{1}, -1\}, \end{aligned} \tag{3.59}$$

where we the underlined coefficient is the coefficient on the diagonal of the change of coordinates matrix.

Now we can define create new transforms just by defining two linearly independent filters (they do not have to be orthogonal). There is a lot of theory that goes into choosing "good" coefficients with right properties. The JPEG2000 image format uses the so-called Cohen-Daubechies-Feauveau (CDF) 9/7 wavelet, where the filters are given by

$$\begin{aligned} H_0 =& \{0.0378, -0.0238, -0.1106, 0.3774, \underline{0.8527}, 0.3774, -0.1106, -0.0238, 0.0378\} \\ H_1 =& \{0.0645, -0.0407, -0.4181, \underline{0.7885}, -0.4181, -0.0407, 0.0645\} \end{aligned} \tag{3.60}$$

These turn out to be very good for representing pictures, as we will see later.

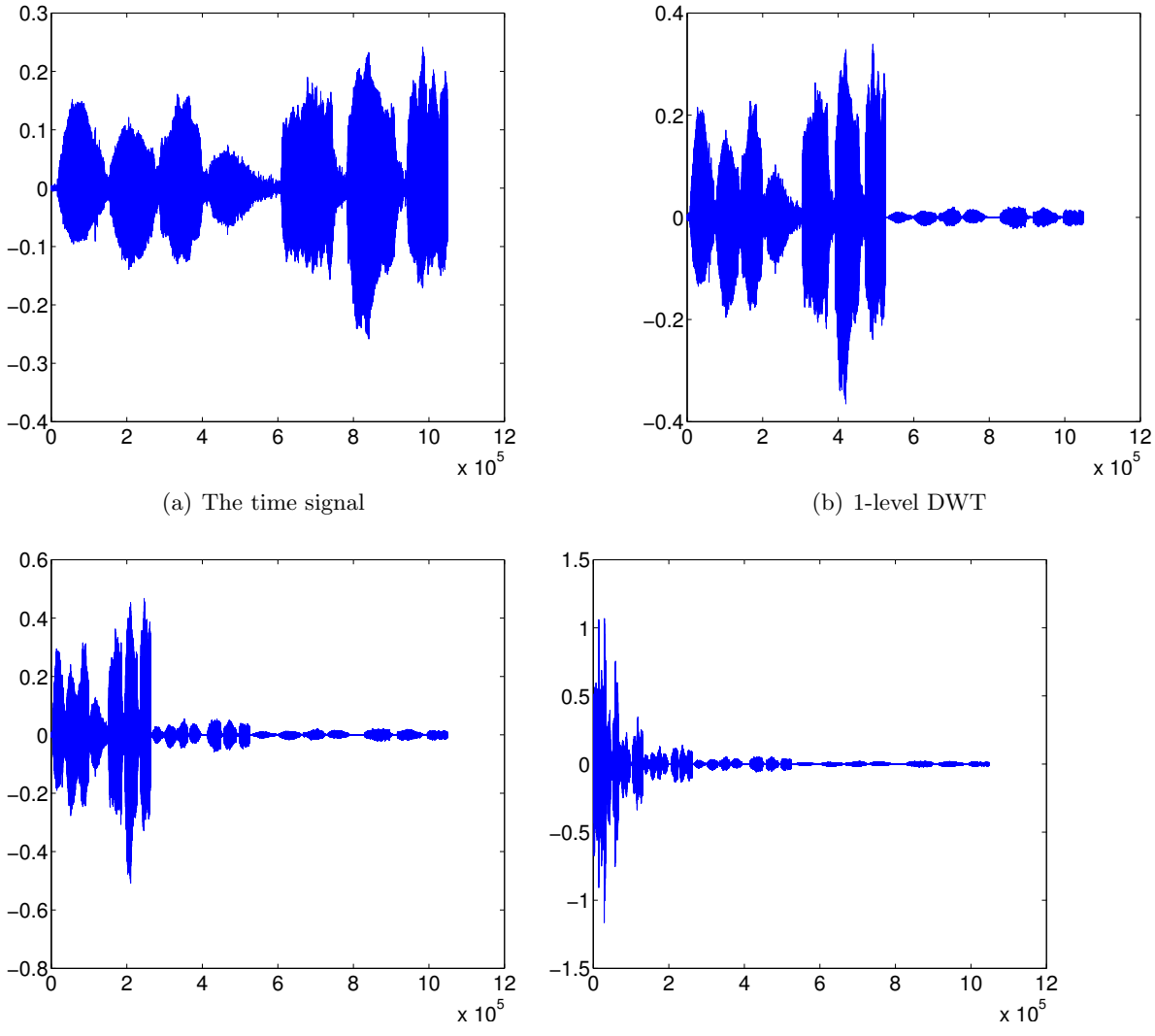
3.3.4 Efficient implementation of the DWT

Because the DWT matrices are typically very sparse, with only a few non-zero diagonals. Rather than setting up the entire DWT matrix, it is much more efficient to simply apply the filters H_0 and H_1 element-wise to a vector $\mathbf{x}$. A single level DWT will then require an order of $\mathcal{O}(N)$ operations, while for a signal of length $n = 2^m$ an m-level DWT will require an order of $\mathcal{O}(n \log_2(n))$ operations. A simple MATLAB implementation of the Haar DWT might look something like the following:

```
function xnew=DWTHaarImpl(x,m)
  xnew=x;
  for mres=m:(-1):1
    len=length(xnew)/2^(m-mres);
    c=(xnew(1:2:(len-1))+xnew(2:2:len))/sqrt(2);
    w=(xnew(1:2:(len-1))-xnew(2:2:len))/sqrt(2);
    xnew(1:len)=[c w];
  end
```

Once again, one should note that MATLAB has a highly efficient DWT function, called by `dwt(x,H0,H1)`, where `H0` and `H1` are the wavelet filters, or `dwt(x,'name')`, where `'name'`is the name of the wavelet used. Here we have decided on a periodic extension of our vectors to account for boundary effects. This is not the default setting in MATLAB. To change this, one may use the call `dwtmode('per')`.

The DWT can be implemented even more efficiently, by applying the so-called *lifting scheme* [11]. The details of this are not needed in our discussion. A lifting scheme implementations will cut the number of additions and multiplications needed in half.



(a) The time signal

(b) 1-level DWT

(c) 2-level DWT

(d) 20-level DWT

Figure 3.3: The figure shows several levels of the Haar DWT transform.

3.4 Basis representation in higher dimensions

So far we have only considered one-dimensional signals. We will need to consider signals in two and three dimensions as well for our applications. We will extend our work so far through the formalism of tensor products. Let us begin by defining what we mean by a tensor product in this setting.

Definition 3.4.1 (Tensor products of vectors)**.** Let V and W be two vector spaces. The Tensor product (also called direct product) of vectors v from V and w from W, denoted $v \otimes w$ is an operation from two vector spaces V and W to a vector space U with dimension $\dim(U) = \dim(V)\dim(W)$, such that

- $(v_1 + v_2) \otimes w = v_1 \otimes w + v_2 \otimes w$,
- $v \otimes (w_1 + w_2) = v \otimes w_1 + v \otimes w_2$,
- $cv \otimes w = v \otimes cw = c(v \otimes w)$.

The new vector space is called the tensor product vector space and is denoted $V \otimes W$.

This definition might seem very vague. The reason for this is that there are several ways to represent the tensor product of vectors. Here we present two different representations, each with their own advantages and drawbacks. First we have the Kronecker product representation, which we will denote by $\otimes^k$ in this section in order to avoid confusion. Later we will use the two formalisms interchangeably.

Definition 3.4.2 (Kronecker product representation of Tensor products)**.** In the Kronecker product representation of vectors, the Tensor product of two vectors $v = (v_1, v_2, \ldots, v_N)^T$ and $w = (w_1, w_2, \ldots, w_M)^T$ is given by $v \otimes^k w = (v_1 w_1, v_1 w_2, \ldots, v_1 w_M, v_2 w_1, v_2 w_2, \ldots, v_2 w_M, \ldots v_N w_1, v_N w_2, \ldots v_N w_M)^T$.

This is the representation of the Tensor product familiar from the way most textbooks deal with spin in quantum mechanics (see for instance [12]). The advantage of this method is be clear once we define how operators are applied on tensor products.

Definition 3.4.3 (Tensor products of operators)**.** We define the tensor product of two operators $A : V_1 \to V_2$ and $B : W_1 \to W_2$ to be the operator $A \otimes B : V_1 \otimes W_1 \to V_2 \otimes W_2$, defined by

$$(A \otimes B)(v \otimes w) = (Av) \otimes (Bw). \tag{3.61}$$

Note that this definition is independent on what representation we decide on.

One can show that using this definition and the Kronecker product representation of direct products, the tensor product is simply the Kronecker product of operators.

Lemma 3.4.4. Let A be an $m \times n$ matrix and B be a $p \times q$ matrix. The tensor product of operators $A \otimes B$ in the Kronecker representation is the $mp \times nq$ matrix

$$A \otimes^k B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix}. \tag{3.62}$$

This product is called the *Kronecker product of matrices.*

Proof. We consider only the special case where A is $n \times n$ and B is $m \times m$. We write our vector as $(v_1\mathbf{w}, v_2\mathbf{w}, \ldots, v_n\mathbf{w})^T$. Then

$$(A \otimes^k B)(v \otimes^k w) = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{n1}B & \cdots & a_{nn}B \end{pmatrix} \begin{pmatrix} v_1\mathbf{w}^T \\ \vdots \\ v_n\mathbf{w}^T \end{pmatrix} \tag{3.63}$$

$$= \begin{pmatrix} (a_{11}v_1 + a_{12}v_2 + \cdots + a_{1n}v_n)B\mathbf{w}^T \\ \vdots \\ (a_{n1}v_1 + a_{n2}v_2 + \cdots + a_{nn}v_n)B\mathbf{w}^T \end{pmatrix} \tag{3.64}$$

$$= (A\mathbf{v}) \otimes^k (B\mathbf{w}). \tag{3.65}$$

□

The advantage of this representation is that matrix-vector equations such as the DFT ($\mathbf{y} = F_N\mathbf{x}$) remain matrix-vector equations in 2D (and higher dimensions). This means that we can use code already developed for one-dimensional problems directly. The disadvantage is that the matrix will be of size $N^2 \times N^2 = N^4$, which offers challenges both in terms of memory usage and computation time. The computation of the matrix vector product will require an order of $\mathcal{O}(N^4)$ operations.

There is an alternate way to represent the tensor product, which we call the *outer product*, and simply denote by $\otimes$.

Definition 3.4.5 (outer product representation of tensor product)**.** In the outer product representation of the tensor product, the Tensor product of two vectors $v = (v_1, v_2, \ldots, v_N)^T$ and $w = (w_1, w_2, \ldots, w_M)^T$ is given by

$$(\mathbf{v} \otimes \mathbf{w}) = \mathbf{v}\mathbf{w}^T = \begin{pmatrix} v_1w_1 & v_1w_2 & \cdots & v_1w_M \\ v_2w_1 & v_2w_2 & \cdots & v_2w_M \\ \vdots & \vdots & \ddots & \vdots \\ v_Nw_1 & v_Nw_2 & \cdots & v_Nw_M \end{pmatrix}. \tag{3.66}$$

In this case we get a different formula for applying a tensor product of operators to a tensor product of vectors.

Lemma 3.4.6 (Tensor product of operators in the outer product representation)**.** Let A be an $m \times n$ matrix and B be a $p \times q$ matrix. The operation $(A \otimes B)(\mathbf{v} \otimes \mathbf{w})$ is given by

$$(A \otimes B)(\mathbf{v} \otimes \mathbf{w}) = A(\mathbf{v} \otimes \mathbf{w})B^T. \tag{3.67}$$

Proof. Note that

$$\begin{aligned} A(\mathbf{v} \otimes \mathbf{w})B^T &= A\mathbf{v}\mathbf{w}^T B^T && (3.68)\\ &= (A\mathbf{v})(B\mathbf{w})^T && (3.69)\\ &= (A\mathbf{v}) \otimes (B\mathbf{w}). && (3.70) \end{aligned}$$

□

This representation has several useful features, such as the fact that in this case we do not find the explicit representation of $A \otimes B$, as it is not necessary to calculate the operation. This saves a lot memory, as the operators will require no more memory than the signal itself. In addition, since we do two matrix multiplications of size N, the computation of this product is of order $\mathcal{O}(N^3)$. The disadvantage is that we do now longer have a matrix-vector product, which means that we need to adjust our algorithms for one-dimensional problems, and it is also less clear how we can extend this representation to higher dimensions.

If we denote $X = (\mathbf{v} \otimes \mathbf{w})$, then the operation can be written as

$$\begin{aligned} (A \otimes B)X &= AXB^T && (3.71)\\ &= (B(AX)^T)^T. && (3.72) \end{aligned}$$

This may seem convoluted, but it tells us that we can break the operation down in the following steps:

1. Apply A to every column in X, as the matrix matrix product AX.
2. Transpose the resulting matrix
3. Apply B to the resulting matrix from step 2.
4. Finally, transpose the result.

This will be useful in implementations, but we will not go into detail on this here.

3.4.1 Basis Representation in 2D

We have laid down most of the ground work at this point. In this section we will argue that if $\{\mathbf{v}^{(i)}\}_{i=0}^{N-1}$ is a set of vectors well suited for approximation or basis representation in the vector space V, and similarly, $\{\mathbf{w}^{(j)}\}_{j=0}^{M-1}$ for the vector space W, then $\{\mathbf{v}^{(i)} \otimes \mathbf{w}^{(j)}\}_{i,j=0,0}^{N-1,M-1}$ is a good representation in $V \otimes W$. To make this clear we first define the inner product in $V \otimes W$.

Definition 3.4.7. The inner product $\langle \cdot, \cdot \rangle$ in $V \otimes W$ is given by

$$\langle v_1 \otimes w_1, v_2 \otimes w_2 \rangle = \langle v_1, v_2 \rangle \langle w_1, w_2 \rangle. \tag{3.73}$$

With this in place, we can prove the following theorem

Theorem 3.4.8. If $\{\mathbf{v}^{(i)}\}_{i=0}^{N-1}$ is a basis for V and $\{\mathbf{w}^{(j)}\}_{j=0}^{M-1}$ is a basis for W, then $\{\mathbf{v}^{(i)} \otimes \mathbf{w}^{(j)}\}_{i,j=0,0}^{N-1,M-1}$ is a basis for $V \otimes W$.

PROOF. Assume that

$$\sum_{i,j=0,0}^{M-1,N-1} c_{i,j}\mathbf{v}^{(i)} \otimes \mathbf{w}^{(j)} = 0. \tag{3.74}$$

We will show that this implies $c_{i,j} = 0$, which means the vectors are linearly dependent. Let $\mathbf{u}^{(i)} = \sum_{j=0}^{M-1} c_{i,j}\mathbf{w}^{(j)}$. Then

$$\sum_{j=0}^{N-1} \mathbf{v}^{(i)} \otimes \mathbf{u}^{(i)} = \sum_{j=0}^{N-1} \mathbf{v}^{(i)}\mathbf{u}^{(i)T} = 0, \tag{3.75}$$

where we note that this is a matrix equation, which means that $\left(\sum_{j=0}^{N-1} \mathbf{u}^{(i)} \otimes \mathbf{w}^{(j)}\right)_{kl} = 0$ for any element (k, l). This in turns means that if we denote the k-th element of $\mathbf{u}^{(i)}$ by $u_k^{(i)}$, then $\sum_{i=0}^{N-1} u_k^{(i)}\mathbf{v}^{(i)} = 0$ for any k. From the linear independence of $\{\mathbf{v}^{(j)}\}_{j=0}^{M-1}$ we know that this implies $u_k^{(i)} = 0$ for all k. Since $u_k^{(i)} = \sum_{j=0}^{M-1} c_{i,j}v_k^{(j)}$, and $\{\mathbf{v}^{(i)}\}_{i=0}^{N-1}$ is linearly dependent, this tells us that $c_{i,j} = 0$ for all j. Since this argument hold for any i, we conclude that $c_{i,j} = 0$ for all i, j. □

It is hard to define in a quantitative way what a "good" basis is, but this theorem shows us that we can draw inspiration from one-dimensional bases when constructing bases in higher dimensions. At this point it will be useful to look at some examples to get a feel for how the two-dimensional transforms behave.

3.4.2 Some examples of Fourier and Haar transforms in 2D

In this section we will test the two-dimensional transforms. We will use two of the standard benchmarking images introduced in 2.4, Lena and Modified Shepp-Logan Phantom.

Note that we can think of a picture as a vector in $\mathbb{R}^n \otimes \mathbb{R}^m$, and as such, we may choose to order it as a long vector in the Kronecker representation. With this in mind we can start to look at the two-dimensional variants of the DFT and the DWT.

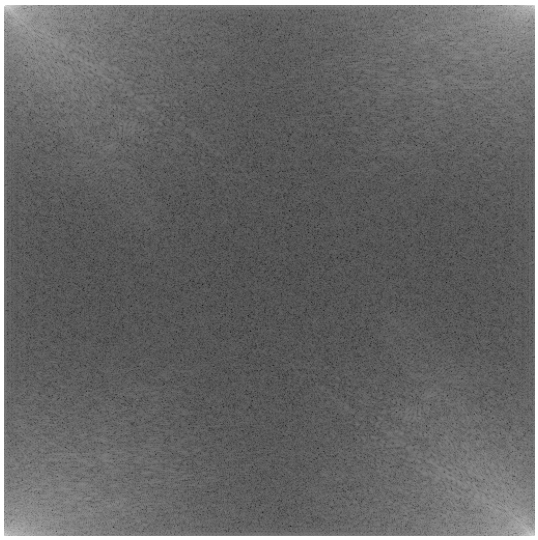
We begin by investigating the Fourier transform. To do this, we simply construct a 512×512 DFT matrix, as described section 3.1.2, and compute the change of coordinates

$$Y = F_N X F_N^T. \tag{3.76}$$

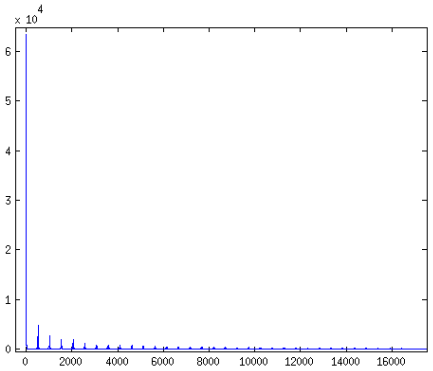
We could in principle have calculated the transform using the Kronecker formalism, but a $512^2 \times 512^2$ complex matrix with double precision would require $2 \times 512^4 \times 64$ bits $= 2^{25}$ bits $=$ 1024GB of memory, which at the time of writing is unreasonable on everything except large supercomputers. The result is shown in figure 3.4(b). It is difficult to extract any information from the signal Y itself, because it is so dominated by low frequencies. For this reason, it is common to take the logarithm of the frequency spectrum, as displayed in figure 3.4(b). We can also order the coefficients as a vector and display them in the Kronecker representation, as shown in figure 3.4(c). We repeat the process for the Shepp-Logan Phantom in figure 3.5.

We see the same folding effects as in the one-dimensional case, so that the spectrum is symmetric around both the x- and y-axis. We also see that the low frequencies dominate the spectrum. We also note that the DFT of the phantom image looks more ordered that the DFT of the lena image. Without going into detail, the ripples seen in the phantom DFT occurs from circular shapes in the image, while the vertical/horizontal lines in the middle occur from vertical/horizontal lines in the image.

(a) The original image, X



(b) $\log|Y|$



(c) the Kronecker representation of $|Y|$

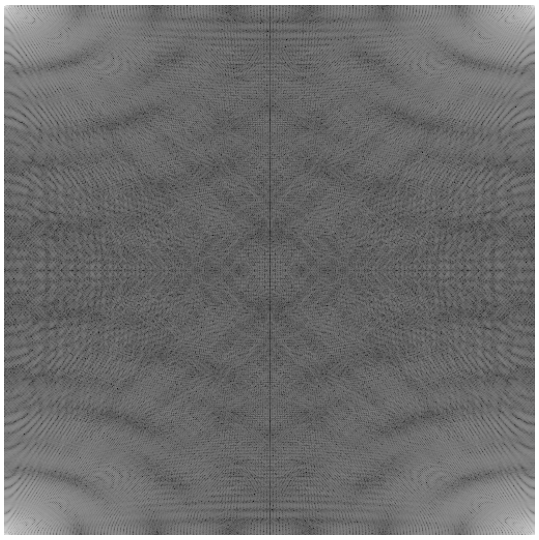
Figure 3.4: The figure shows different representations of the DFT of the lena image.

Next, we introduce the 2-D DWT. Note that the functions we use to generate the DWT will now be $\phi \otimes \phi$, $\phi \otimes \psi$, $\psi \otimes \phi$ and $\psi \otimes \psi$. We have not defined the tensor product of functions yet, however, it is simply $\psi \otimes \phi(x \otimes y) = \psi(x)\phi(y)$. These functions are illustrated in figure 3.6. this means that we split the information in the picture into 4 different groups,

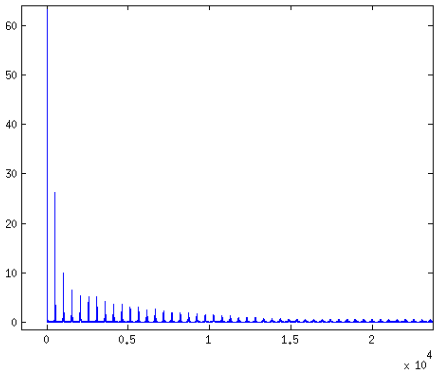
$$V_1 \to V_0 \oplus W_0^{(0,1)} \oplus W_0^{(1,0)} \oplus W_0^{(1,1)}, \tag{3.77}$$

where $W_0^{(0,1)}$ is the space generated by $\phi \otimes \psi$, $W_0^{(1,0)}$ by $\psi \otimes \phi$ and $W_0^{(1,1)}$ by $\psi \otimes \psi$. By convention

(a) The original image, X



(b) $\log|Y|$



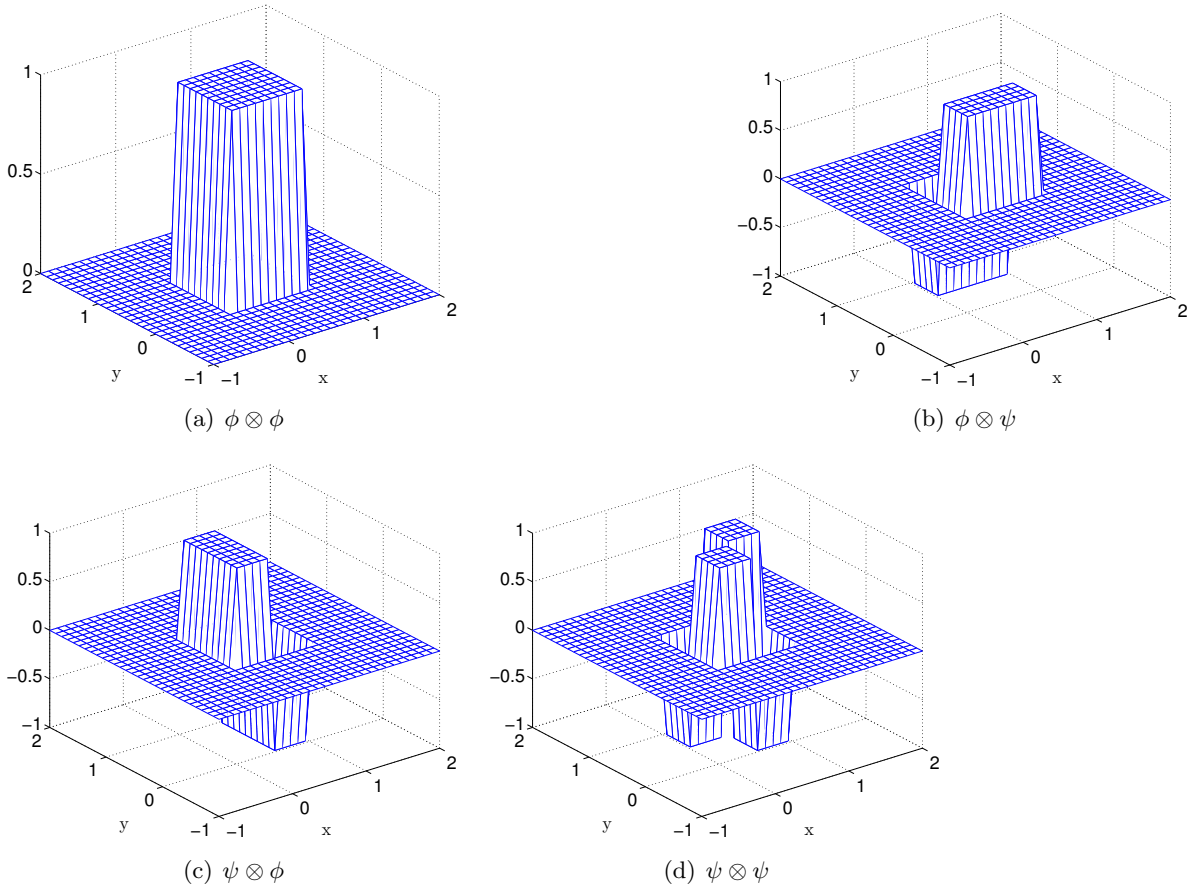
(c) the kronecker representation of $|Y|$

Figure 3.5: The figure shows different representations of the DFT of the modified Shepp-Logan image.

we sort the result according to

$$\begin{pmatrix} V_0 & W_0^{(0,1)} \\ W_0^{(1,0)} & W_0^{(1,1)} \end{pmatrix} \tag{3.78}$$

which means that the upper left corner of the transformed image will contain a low-resolution version of the original image. The other spaces are all referred to as detail spaces. When we perform an MRA, only the upper left quarter is used in the next iteration. An example of an MRA of the Lena image is shown in figure 3.7. Visually, the DWTs of images are much easier to interpret than the DFTs.



(a) $\phi \otimes \phi$ (b) $\phi \otimes \psi$

(c) $\psi \otimes \phi$ (d) $\psi \otimes \psi$

Figure 3.6: The figure shows the basis functions used to generate the 2-D Haar DWT.

(a) original image (b) $m = 1$

(c) $m = 2$ (d) $m = 3$

Figure 3.7: The figure shows the m-level DWT of the Lena image for various values of m.

3.5 Compressability of signals: sparsity

Most modern signal compression schemes are based on some clever representation of signals. MP3 uses a form of discrete cosine transform, while the JPEG2000 scheme uses the CDF-9/7 wavelet. These are usually followed by some form of numerical compression such as Huffman encoding or arithmetic encoding. The central idea in lossless compression is that if a lot of the coefficients in some basis are 0, we can use less bits to represent the 0-valued elements. We call this property the *sparsity* of the the signal.

Definition 3.5.1 (Sparse signals)**.** Informally, we refer to a signal $\mathbf{x}$ as *sparse* if most of the elements of $\mathbf{x}$ are 0. Formally, $\mathbf{x}$ is said to be *k-sparse* if $\mathbf{x}$ has at most k non-zero elements.

Sparsity can also be defined in terms of the *support* of $\mathbf{x}$. In order to do this, we should define some set properties, including the support.

Definition 3.5.2 (Subsets, complements, cardinality and support)**.** Let $\mathbb{N}_0^{N-1} = \{0, 1, \ldots N-1\}$ be the set of non-negative integers smaller than N.

- Let $\Lambda \subseteq \mathbb{N}_0^{N-1}$ be a *subset* of $\mathbb{N}_0^{N-1}$, i.e. some set of non-negative integers smaller than N. Λ may include all of the elements in $\mathbb{N}_0^{N-1}$, none of them, of anything in between.
- The set of all numbers in $\mathbb{N}_0^{N-1}$, but not in Λ is called the *complement* of Λ, and is denoted Λ^c. We write this selection as $\Lambda^c = \mathbb{N}_0^{N-1} \setminus \Lambda$.
- The number of elements in Λ is called the *cardinality* of Λ, and is denoted $|\Lambda|$.
- Let $\mathbf{x}$ be a signal of length N. The set of indices $\Lambda = \{i : x_i \neq 0\}$ is called the *support* of $\mathbf{x}$, and denoted $\text{supp}(\mathbf{x})$. In this sense $\mathbf{x}$ is k-sparse if its support has cardinality k.
- We define $\mathbf{x}_\Lambda$ as the vector made by setting all elements with indices not in Λ equal to zero:

$$(\mathbf{x}_\Lambda)_j = \begin{cases} x_j & \text{if } j \in \Lambda, \\ 0 & \text{otherwise.} \end{cases} \tag{3.79}$$

It may seem like a bit of an overkill to define all these properties to describe the sparsity of $\mathbf{x}$. However, we will find all these properties useful later, and this is as good a place as any to introduce them.

In most real cases, a signal will not have many elements which are exactly zero. However, if elements are sufficiently close to zero, setting them to zero will not result in a large error. By doing this, we can achieve an even higher degree of compression. Let us define the measure of this error.

Definition 3.5.3 (k-term approximation error)**.** Let $\mathbf{x}$ be a signal and let $\mathbf{x}_k$ be the approximation to $\mathbf{x}$ obtained by keeping only the k largest elements of $\mathbf{x}$, and setting the rest to zero. The k-term approximation error is defined as

$$\sigma_k(\mathbf{x})_p = \|\mathbf{x} - \mathbf{x}_k\|_p. \tag{3.80}$$

We should also define what we mean by a signal having "many elements close to zero".

Definition 3.5.4 (Compressible signals)**.** We say that a signal is *compressible* or *weakly sparse* with the parameter $s > 0$ if the coefficients x_i decay according to the power law

$$|x_i| \leq Ci^{-1/s}. \tag{3.81}$$

The closer s is to 0, the faster the signal decays. The following result for weakly sparse signals is easy to prove.

Lemma 3.5.5. If $\mathbf{x}$ is weakly sparse with parameter s, then the k-term approximation error decays as

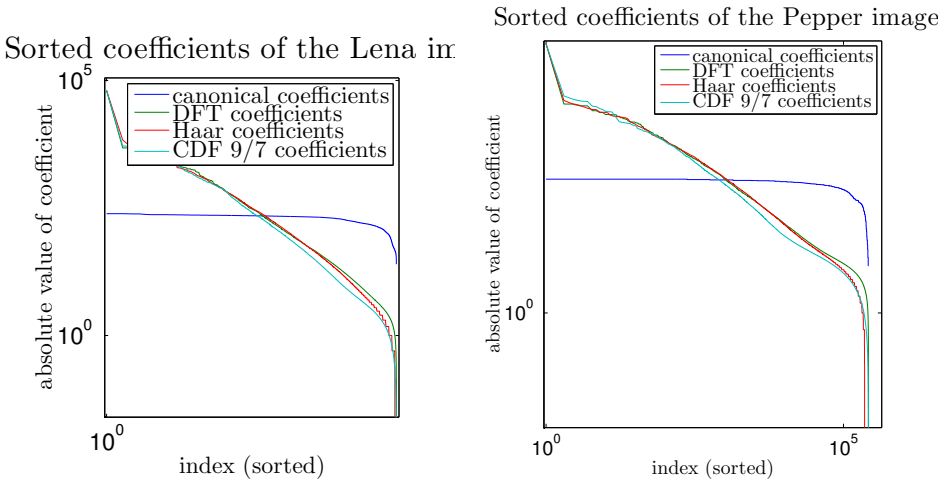
$$\sigma_k(\mathbf{x})_2 =\leq C(2/s-1)^{-1/2}k^{1/2-1/s}, \quad s < 2. \tag{3.82}$$

Proof. If we sort the elements of $\mathbf{x}$ and $\mathbf{x}_k$ from largest to smallest, then we have

$$\begin{aligned}
\sigma_k(\mathbf{x})_2^2 &= \sum_{i=1}^{N}(x_i - x_{k,i})^2 && (3.83)\\
&= \sum_{i=k+1}^{N}(x_i - 0)^2 && (3.84)\\
&\leq \sum_{i=k+1}^{N} C^2 i^{-2/s} && (3.85)\\
&\leq C^2 \int_{k}^{N-1} x^{-2/s}\,dx && (3.86)\\
&= C^2\left[(-2/s+1)^{-1}x^{-2/s+1}\right]_{k}^{N-1} && (3.87)\\
&= C^2(1-2/s)\left[(N-1)^{1-2/s} - k^{1-2/s}\right] && (3.88)\\
&\leq C^2(2/s-1)k^{1-2/s}, && (3.89)
\end{aligned}$$

for $s < 2$, where we have used the fact that the sum can be considered a lower Riemann sum for integral of $x^{-2/s}$. Taking the square root on each side gives the result. □

Most natural signals follow an approximate power law distribution. Figure 3.8 shows the distribution of coefficients in the Fourier, Haar and CDF 9/7 bases for the Lena and Pepper images. Note that distributions follow approximate power laws for both cases, and the distributions seem quite similar for all 6 signals.



(a) Lena coefficients (b) Peppers coefficients

Figure 3.8: The sorted coefficients of the Lena and Peppers image in different representations.

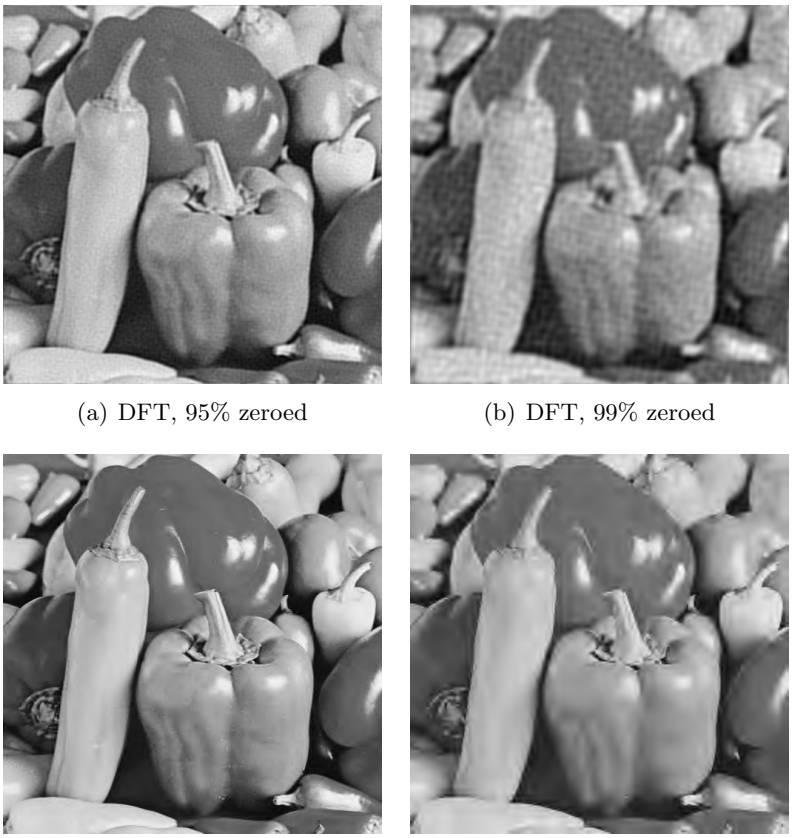
In DSP, signal compression is usually based on representing the signal is some appropriate basis. Then methods are split in *lossless compression*, where the signal is not altered, but encoded in a clever way, taking advantage of the coefficient distribution, and *lossy compression*, where only the largest coefficients of the signal is kept. The process of zeroing out the smallest elements of a vector has its own designation.

Definition 3.5.6 (Sparsing)**.** The act of zeroing out the smallest elements of a signal is called *sparsing*.

Usually, images can be sparsed greatly without noticeable loss in quality, in the proper basis. This is best shown through an example.

Example 3.5.7 (Sparsing of an image)**.** In this example we will look at one of the standard DSP benchmarking image from figure 2.4, Peppers. We will investigate the effect of sparsing the signal when representing it in the DWT and the CDF-9/7 wavelet basis. We will attempt to zero out 95% and 99% of the coefficients. The signal is then transformed back to the canonical basis, and finally rounded to real integers in the interval [0,255] (which is the same as the original image). The result of this process is shown in figure 3.9. We see that both generally perform well in the case where 5% of the coefficients are kept, while they display different artifacts in the case where 1% of the coefficients are kept. As is to be expected, the wavelet transform performs better for the more extreme compression. ♣

This example gives us some idea that we can preserve the quality of the image, but in order to quantize this notion, we should define what we mean by the error in the signal in a meaningful way. This will be related to how we measure the magnitude of a signal. Generally, the length of a vector is given by a function called the *vector norm*.



(a) DFT, 95% zeroed (b) DFT, 99% zeroed

(c) DWT, 95% zeroed (d) DWT, 99% zeroed

Figure 3.9: Different compressed versions of the Pepper image.

Definition 3.5.8 (vector norm)**.** The vector norm is a function $\|\cdot\| : \mathbb{C}^N \to [0, \infty)$, which satisfies the following properties:

1. $\|c\mathbf{x}\| = |c|\|\mathbf{x}\|$, for any $c \in \mathbb{C}$,
2. $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ (the triangle inequality),
3. $\|\mathbf{x}\| = 0$ if and only if $\mathbf{x} = \mathbf{0}$.

There exists several different norm, each with their own field of use. A large group of norms, collectively called ℓ_p-norms, are defined as follows:

Definition 3.5.9 (ℓ_p-norms)**.** The ℓ_p-norm, $\|\mathbf{x}\|_p$, of a vector $\mathbf{x}$, is defined for any $p \in \mathbb{N}_0$ as

$$\|\mathbf{x}\|_p = \left(\sum_{i=0}^{N-1} |x_i|^p \right)^{\frac{1}{p}}. \tag{3.90}$$

The case $p = 2$ is the standard Euclidean norm. The case $p = 1$ is often referred to as the *taxicab* or *Manhattan* norm, as it is the distance one would have to walk between two points of a city where one could only make 90 degree turns. For the case $p = 0$ we need to define $0^0 = 0$ for the norm to give a meaningful result. In this case, the norm simply counts all the non-zero elements of $\mathbf{x}$, and is this way works as a measure of the sparsity of $\mathbf{x}$. This is often called the *discrete norm*. Finally, the *maximum* norm, which occurs when $p \to \infty$, is given by $\|\mathbf{x}\|_\infty = \max_i\{|x_i|\}$. The unit circle, the set of points $\mathbf{x}$ such that $\|\mathbf{x}\|_p = 1$ is shown in figure 3.10 for $p = 0, 1, 2, \infty$.

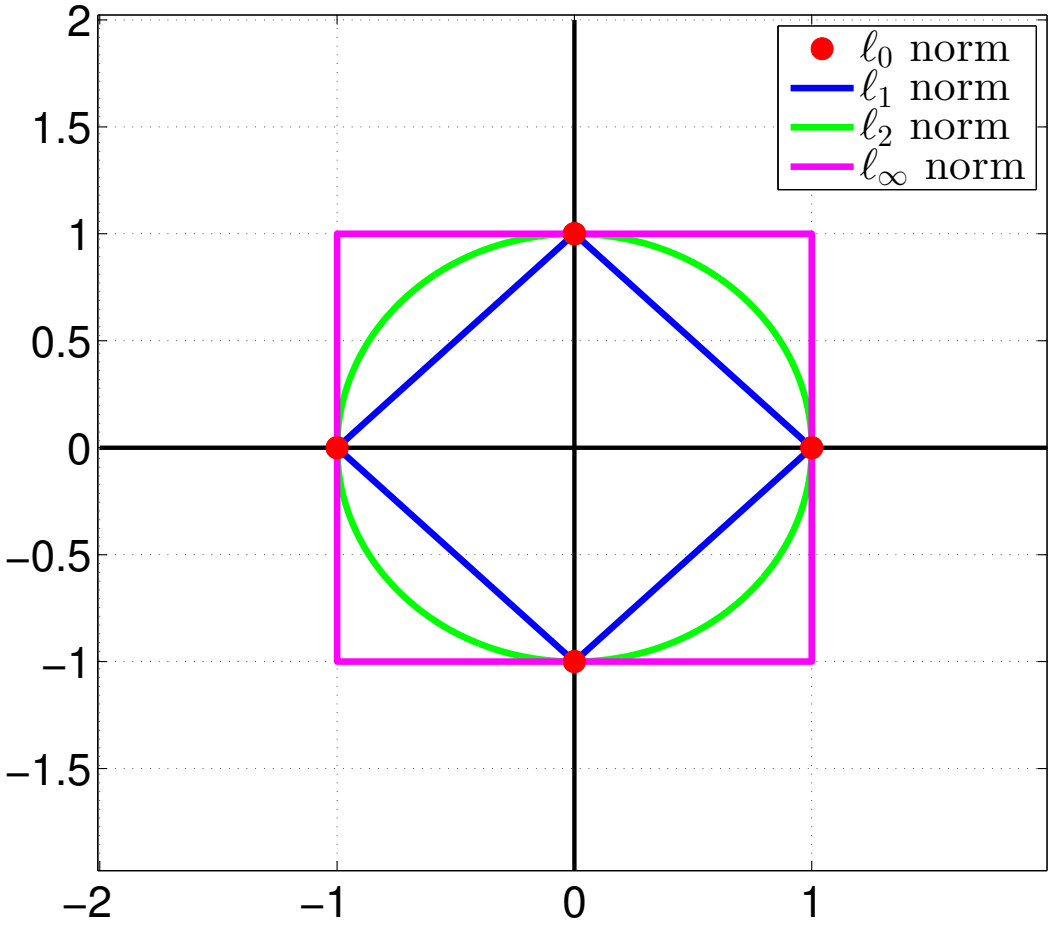


Figure 3.10: The figure shows the unit circles in the ℓ_0, ℓ_1, ℓ_2 and ℓ_∞ norms.

The $p = 0$, $p = 1$, $p = 2$ and $p = \infty$ norms are all reasonable ways to measure the error of an approximation to a signal. The signal processing community have generally chosen to use some functions related to the ℓ_2 norm to measure the error.

	DFT, 95%	DFT, 99%	DWT, 95%	DWT, 99%
ℓ_0	246839	252493	242758	249529
ℓ_1	1.1149×10^4	1.4741×10^4	3.458×10^3	6.145×10^3
ℓ_2	8.9922×10^2	1.65×10^3	2.8599×10^2	8.0550×10^2
ℓ_∞	1.03×10^2	1.47×10^2	3.6×10^1	1.03×10^2
SNR	24.34	19.76	27.73	22.75
MSE	3.0821	10.3855	0.3120	2.4751
$PSNR$	29.08	24.50	32.47	27.48

Table 3.1: The table lists the different error measurements from the sparsing in example 3.5.7.

Definition 3.5.10 (Signal-to-noise ratio and Mean Square Error)**.** The Signal-to-Noise Ratio (SNR) of a an approximation $\mathbf{x}^*$ to a signal $\mathbf{x} \in \mathbb{R}^N$ is defined as

$$SNR(\mathbf{x}^*, \mathbf{x}) = 10 \log_{10} \left(\frac{\|\mathbf{x}\|_2^2}{\|\mathbf{x} - \mathbf{x}^*\|_2^2} \right) = 20 \log_{10} \left(\frac{\|\mathbf{x}\|_2}{\|\mathbf{x} - \mathbf{x}^*\|_2} \right). \tag{3.91}$$

The Mean Square Error (MSE) is defined as

$$MSE(\mathbf{x}^*, \mathbf{x}) = \frac{1}{N} \|\mathbf{x}^* - \mathbf{x}\|_2^2. \tag{3.92}$$

The Peak Signal-to-Noise Ratio (PSNR) is defined as

$$PSNR(\mathbf{x}^*, \mathbf{x}) = 10 \log_{10} \left(\frac{MAX^2}{MSE(\mathbf{x}^*, \mathbf{x})} \right) = 20 \log_{10} \left(\frac{MAX}{\sqrt{MSE(\mathbf{x}^*, \mathbf{x})}} \right), \tag{3.93}$$

where MAX is the maximum possible signal value. For an image stored as integers with 8 bits per pixel, this is 255.

Note that for the SNR and the PSNR, a higher value corresponds to better quality. The different errors for the approximations in example 3.5.7 are shown in table 3.1. Note that the ℓ_0 error in all cases is very close to the maximum value of $512^2 = 262144$. If not for the rounding operation at the end of the reconstruction, the error would be the maximum value for all the reconstruction attempts, highlighting why the ℓ_0 error is too strict for most uses.

We note one more result, which relates the error on one basis to that of another. First, we should define the norm of a matrix.

Definition 3.5.11 (Matrix norm)**.** The norm of a matrix A, $\|A\|_p$ is given by

$$\|A\|_p = \max \{ \|A\mathbf{x}\|_p : \|\mathbf{x}\|_p = 1 \}. \tag{3.94}$$

Note in particular that for any unitary matrix U, such as the DFT or the Haar DWT, $\|U\|_2 = \|U^{-1}\|_2 = 1$, as for any vector $\mathbf{x}$,

$$\|U\mathbf{x}\|_2 = (U\mathbf{x})^\dagger (U\mathbf{x}) = \mathbf{x}^\dagger U^\dagger U \mathbf{x} = \mathbf{x}^\dagger \mathbf{x} = \|\mathbf{x}\|_2. \tag{3.95}$$

With this in mind, we have the following result

Lemma 3.5.12. Let $\mathbf{x}$ be some signal, and let $\mathbf{y} = A\mathbf{x}$ be its representation in some basis. Let $\mathbf{y}^*$ be an approximation to $\mathbf{y}$, with error $\|\mathbf{y} - \mathbf{y}^*\|_p \leq \epsilon$. Then the restored signal in the canonical basis, $\mathbf{x}^* = A^{-1}\mathbf{y}^*$ has the error

$$\|\mathbf{x} - \mathbf{x}^*\|_p \leq \epsilon \|A^{-1}\|_p. \tag{3.96}$$

Proof. Note that

$$\begin{aligned} \|\mathbf{x} - \mathbf{x}^*\|_p &= \|A^{-1}\mathbf{y} - A^{-1}\mathbf{y}^*\|_p && (3.97)\\ &= \|A^{-1}(\mathbf{y} - \mathbf{y}^*)\|_p && (3.98)\\ &\leq \|A^{-1}\|_p \|\mathbf{y} - \mathbf{y}^*\|_p && (3.99)\\ &\leq \|A^{-1}\|_p \epsilon. && (3.100) \end{aligned}$$

□

In particular, this means that for unitary transforms, the ℓ_2-error is the same in the canonical basis as in the transformed basis. This is important in order to guarantee quality of a restored signal after sparsing.

We will get back to compressibility of signals in the next chapter, as it is at the heart of Compressed Sensing methods. For now we turn our discussion to some other basis with interesting properties.

3.6 A brief look at exotic bases: Curvelets

Traditional wavelets in 1-D are efficient for representing signals with singularities. This in turn makes the 2-D extension of wavelets efficient for representing signals with point singularities. However, many natural images exhibit line-like edges (so-called line or curve singularities). Recently, a large group of transforms, including curvelets [13, 14, 15], steerable wavelets [16], [17], Gabor wavelets [18], wedgelets [19], beamlets [20], bandlets [21], [22], contourlets [23], shearlets [24], [25], wave atoms [26], platelets [27] and surfacelets [28] have been proposed independently to restore geometric features. Here we will briefly look into the construction of the curvelet Transform, as a sample of the properties of these new bases. We will introduce several other transforms which are used to explain features of the curvelet transform, but we will intentionally be brief on the details of these transforms, as we will not come back to them later. Computational details can be found in their respective literature.

The success of wavelets is mainly due to good performance in for piecewise smooth functions in one dimension. In 1D, wavelets are good for picking up point singularities with a high compression rate. In 2D the construction of wavelets using direct products ensures that point singularities are also represented well. However, line singularities, which does not follow perfect vertical or horizontal lines, will need (comparatively) a high amount of coefficients. These relatively new transforms belong to a group of 2D-transforms seeking to amend the problems of standard wavelets.

3.6.1 The Radon Transform

We begin with the Radon transform, which is much older than the other transforms we will visit, but serves as a prerequisite for the other transforms. It was initially introduced in 1917 by Johan Radon, and is the function you get from making projections along straight lines through space. In 2D, the projections are taken along the curves

$$(x(t), y(t)) = (t\sin\alpha + s\cos\alpha), (-t\cos\alpha + s\sin\alpha)) \,. \tag{3.101}$$

We can think of the lines in the following way: Let s be the radius of a circle in 2D. Then, α is the angle of the circle at which the projection lines intersects the circle, as illustrated in figure 3.11.

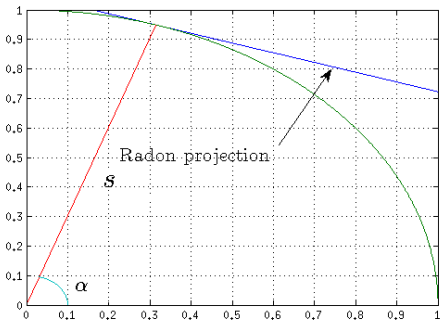


Figure 3.11: A Radon projection line

We can now define the Radon transform.

Theorem 3.6.1. The Radon transform of $f(x, y)$, denoted by $Rf(\alpha, s)$ is given by

$$Rf(\alpha, s) = \int_{-\infty}^{\infty} f(x(t), y(t))\, dt \tag{3.102}$$

$$= \int_{-\infty}^{\infty} f(t\sin\alpha + s\cos\alpha, -t\cos\alpha + s\sin\alpha)\, dt. \tag{3.103}$$

The Radon transform is closely related to the Fourier transform. If we define the 2-D $\text{CTFT}_{2\pi}$, by which we will mean a CTFT, the continuous Fourier transform to be defined in section 5.4, where there is a factor 2π in the complex exponential, of $f(\mathbf{x})$ as

$$F(\mathbf{w}) = \int\int f(\mathbf{x}) \exp(-2\pi i \mathbf{x}\cdot\mathbf{w})\, d\mathbf{x}.$$

We then have the following result

Theorem 3.6.2 (The Fourier slice theorem)**.** Let $FR_\alpha[f](\omega)$ be the $\text{CTFT}_{2\pi}$ of $Rf(\alpha, s)$ with respect to s. Then

$$FR_\alpha[f](\omega) = F(\omega\mathbf{n}(\alpha)), \tag{3.104}$$

where $\mathbf{n}(\alpha) = (\cos\alpha, \sin\alpha)$.

PROOF.

$$FR_\alpha[f](\omega) = \int Rf(\alpha, s)\exp(-2\pi i s\omega)\,ds \tag{3.105}$$

$$= \int Rf(\alpha, s)\exp(-2\pi i s\omega)\,ds \tag{3.106}$$

$$= \int \left(\int_{-\infty}^{\infty} f(t\sin\alpha + s\cos\alpha, -t\cos\alpha + s\sin\alpha)\,dt\right)\exp(-2\pi i s\omega)\,ds \tag{3.107}$$

$$= \int \left(\int_{-\infty}^{\infty} f(t\sin\alpha + s\cos\alpha, -t\cos\alpha + s\sin\alpha)\exp(-2\pi i s\omega)\,dt\right)\,ds, \tag{3.108}$$

while

$$F(\omega\mathbf{n}(\alpha)) = \int f(x, y)\exp(2\pi i\omega(x\cos\alpha + y\sin\alpha))\,dxdy. \tag{3.109}$$

We use the change of variables $x = t\sin\alpha + s\cos\alpha$ and $y = -t\cos\alpha + s\sin\alpha$ The Jacobi determinant for this change of variables is

$$J = \begin{vmatrix} \frac{\partial x}{\partial t} & \frac{\partial x}{\partial s} \\ \frac{\partial y}{\partial t} & \frac{\partial y}{\partial s} \end{vmatrix} = \begin{vmatrix} \sin\alpha & \cos\alpha \\ -\cos\alpha & \sin\alpha \end{vmatrix} = 1. \tag{3.110}$$

Notice also that

$$x\cos\alpha + y\sin\alpha = (t\sin\alpha + s\cos\alpha)\cos\alpha + (-t\cos\alpha + s\sin\alpha)\sin\alpha \tag{3.111}$$

$$= t(\sin\alpha\cos\alpha - \sin\alpha\cos\alpha) + s(\cos^2\alpha + \sin^2\alpha) \tag{3.112}$$

$$= s, \tag{3.113}$$

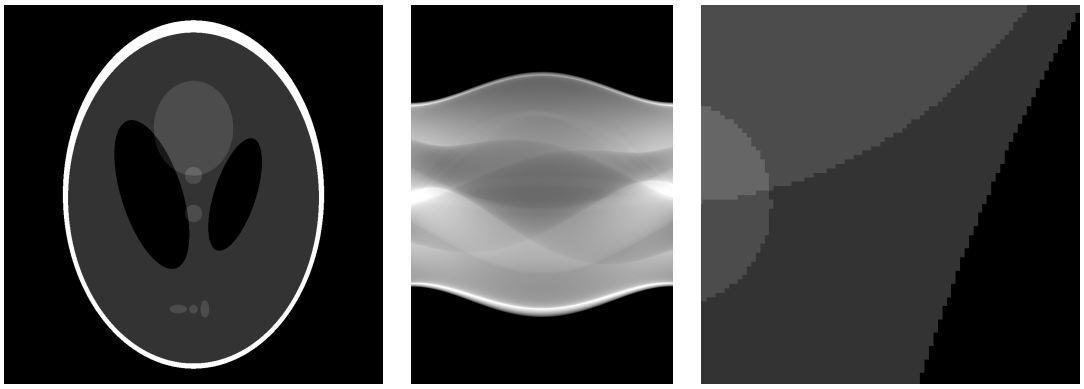
which implies that

$$F(\omega\mathbf{n}(\alpha)) = \int \left(\int_{-\infty}^{\infty} f(t\sin\alpha + s\cos\alpha, -t\cos\alpha + s\sin\alpha)\exp(-2\pi i s\omega)\,dt\right)\,ds. \tag{3.114}$$

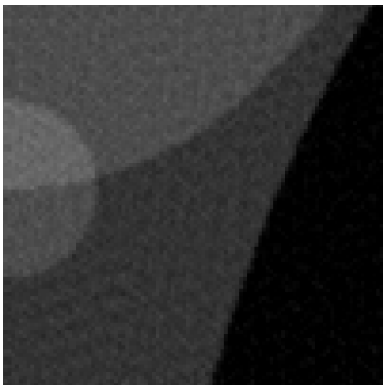
which proves the theorem. □

This theorem shows that the Radon transform can be related to making partial Fourier transforms along radial lines through the origin at different angles (in the frequency space). This will not be important presently, but is important to understand applications where partial reconstruction from radial lines in the Fourier space is used, while partial measurements in the Radon space are experimentally more likely to be available. It also opens the path for an efficient Radon transform implementation, based of the FFT.

The *Discrete Radon Transform* (DRAT) is made by choosing a finite subset of angles $\{\alpha_i\}$ for which to makes projections along. A common implementation (and the one used in MATLAB) simply takes the angles as input and performs the projection numerically. Because of this, the discrete Radon transform is not perfectly invertible, nor is it a transform of the type $\mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$. Rather, the output will be roughly $N_\alpha \times \sqrt{m^2 + n^2}$, where N_α is the number of angles used. The factor $\sqrt{m^2 + n^2}$ is due to the fact that the projections will go out all the way to the edges of the image, and is only approximate. Usually it will be somewhat larger. Because of this, a lot of the projection lines will be 0, because they are projections along lines going



(a) Phantom data (b) Radon transform (c) Phantom excerpt



(d) Inverse Radon excerpt

Figure 3.12: A Radon transform of the modified Shepp-Logan Phantom. Note that the restored signal contains some artifacts.

outside the image completely. An example of a Radon transform and a subsequent approximate inversion is shown in figure 3.12.

Another implementation allows the projections to loop periodically over the edges of the signal. A version of this implementation is described in [29], where the DRAT is defined as follows:

Definition 3.6.3. Discrete Radon transform The discrete Radon transform R_X of an $N \times N$ signal X is given by

$$(R_X)_{k,l} = \frac{1}{\sqrt{N}} \sum_{i,j \in L_{k,l}} X_{i,j}, \tag{3.115}$$

where $L_{k,l}$ is the set of points that make up a line in X,

$$L_{k,l} = \{(i,j) : j = ki + l \pmod p, i \in \{1,2,\ldots,N-1\}\}, \quad 1 \leq k < N, \tag{3.116}$$

$$L_{N,l} = \{(l,j) : j \in \{1,2,\ldots,N\}. \tag{3.117}$$

This transform takes us from a $p \times p$ to a $p \times (p+1)$ signal. There is some redundancy in

this representation. The redundancy can be sorted out by noting that

$$\sum_{l=0}^{p-1}(R_X)_{k,l} = \frac{1}{\sqrt{p}}\sum_{i,j} X_{i,j}, \tag{3.118}$$

which means that for each k we can omit one value for l if we desire (however, it is usually practical to retain all coefficients). Figure 3.13 shows the different elements of a 7×7 DRAT.

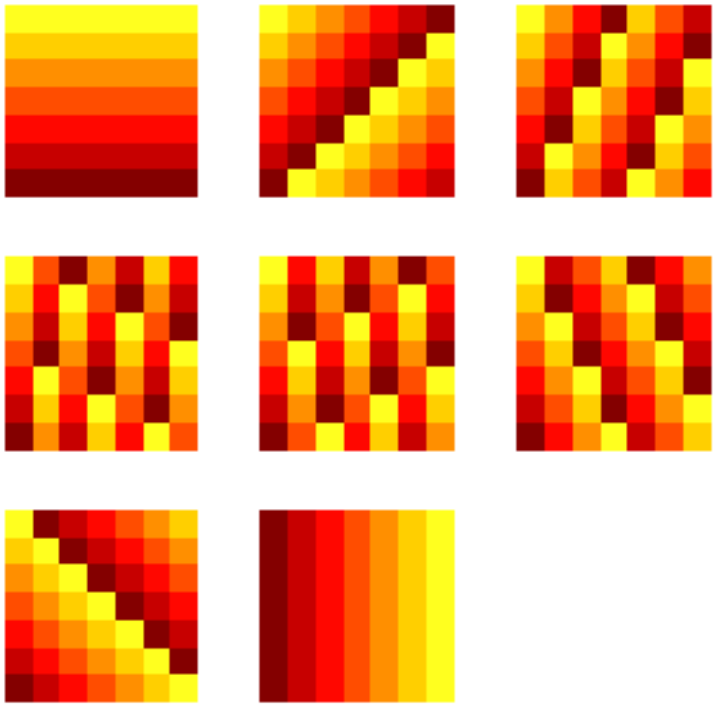


Figure 3.13: The figure shows the 56 elements of the 7×7 DRAT. Each basis element is denoted by its own color.

The advantage with this version is that it is exactly invertible for zero-mean signals, in cases where p is prime. The inverse operation is called the Finite Back-projection (FBP), and is defined as the sum of all Radon coefficients of lines that go through a given point. That is,

$$FBP_{R_X} = \frac{1}{\sqrt{p}}\sum_{(k,l)=P_{i,j}} r_k[l], \quad 0 \le i,j < N, \tag{3.119}$$

where $P_{i,j}$ is the set of indices of the lines that passes through the point (i,j). We can write this more explicit as

$$P_{i,j} = \{(k,l) : l = j - ki \pmod p, 0 \le k < N\} \cup \{(p,i). \tag{3.120}$$

To see that this is indeed the inverse, note that every set of two points $(i,j),(i',j')$ in the signal lie on exactly one line $L_{k,l}$. Note also that every single point (i',j') lie in exactly one line in the set $P_{i,j}$, except the point (i,j) itself which lies on all $p+1$ lines.

Thus, FBP applied to the DRAT of an image X can be written as

$$(FBR_{R_X}(R_X))_{i,j} = \frac{1}{p} \sum_{(k,l)\in P_{i,j}} \sum_{(i'.j')\in L_{k,l}} X_{i',j'} \tag{3.121}$$

$$= \frac{1}{p} \left(\sum_{(i',j')\in[0,p-1]^2} X_{i',j'} + pX[i,j] \right) \tag{3.122}$$

$$= X_{i,j}. \tag{3.123}$$

Finally, we note that it can be shown that the matrices for the FRAT and the FBP are transposed of each other.

The Radon transform introduces the idea of transforms using ideas different from those of the DFT and the DWT. However, the there is no reason to believe that the Radon transform should be any more sparse than the original Cartesian representation. With this in mind we can look into some other transforms.

3.6.2 The Ridgelet Transform

This following is based on the Ph.D. Thesis of Candës [30], as well as [29]. We begin by introducing the notation $\psi_{a,\mathbf{u},b}(\mathbf{x}) = a^{-1/2}\psi(\frac{\mathbf{u}\cdot\mathbf{x}-b}{a})$. We can think of a as a scaling of ψ, $\mathbf{u}$ as a direction and b as a translation. In this section, we again let $\Psi(\omega)$ be the CTFT of $\psi(t)$. With this in mind, we present the following definition.

Definition 3.6.4. Let $\psi : \mathbb{R} \to \mathbb{R}$ satisfy the condition

$$K_\psi = \int \frac{|\Psi(\omega)|^2}{|\omega|^d} < \infty, \tag{3.124}$$

where d is the dimensionality. Then ψ is called an *admissible neural activation function.*

Ridgelets emerged in a setting of modeling neural networks, which motivates this name. Here, we will simply refer to such an ψ as *admissible.*

Theorem 3.6.5. Suppose $f(\mathbf{x})$ and $F(\omega)$ are L^1-integrable. If ψ is admissible, then

$$f(\mathbf{x}) = c_\psi \int \langle f, \psi_{a,\mathbf{u},b}\rangle \psi_{a,\mathbf{u},b} \frac{\sigma_d \, da \, d\mathbf{u} \, db}{a^{d+1}}, \tag{3.125}$$

where $c_\psi = \pi(2\pi)^{-d}K_\psi^{-1}$ and σ_d is the surface area of the unit sphere in $d-1$ dimensions.

The proof of this theorem is fairly straight forward, but involves a bit of theory which is not suited for this setting. The proof can be found in [30]. The important part is that this theorem says that the function f can be reconstructed from the coefficients $\langle f, \psi_{a,\mathbf{u},b}\rangle$.

Since we are mainly interested ridgelets in 2-D, we can replace $\mathbf{u}$ by a single parameter, θ, giving the orientation of the ridgelet. Let us now properly define the Continuous Ridglet Transform (CRT)

Definition 3.6.6 (The continuous ridgelet transform)**.** The CRT in 2D is defined by

$$CFT_f(a,b,\theta) = \int_{\mathbb{R}^2} \psi_{a,b,\theta}(\mathbf{x}) f(\mathbf{x})\, dx. \tag{3.126}$$

where the ridgelets are defined from a wavelet-type function $\psi(x)$ as

$$\psi_{a,b,\theta}(\mathbf{x}) = a^{-1/2}\psi((x_1\cos\theta + x_2\cos\theta - b)/a). \tag{3.127}$$

While both 2D wavelets and ridgelets are based on 1D wavelets, we no longer construct our basis by using tensor products. A key difference is that while in $\psi_{m_1,n_1} \otimes \psi_{m_2,n_2}$ the parameters n_1, n_2 describe points in $\mathbb{R}^2$, the parameter pair b, θ describes lines.

We can relate this transform to the Radon transform. Note that we can write the Radon transform similarly to the CRT as

$$Rf(\alpha, s) = \int_{\mathbb{R}^2} f(\mathbf{x})\delta(x_1\cos\theta + x_2\sin\theta - s)\, d\mathbf{x}, \tag{3.128}$$

where $\delta(t)$ is the Dirac delta-function. This gives us the following result

Lemma 3.6.7. The CRT of f can be expressed in terms of the radon transform of f as

$$CRT_f(a,b,\theta) = \int_{-\infty}^{\infty} \psi_{a,b}(t) Rf(\theta,t)\, dt, \tag{3.129}$$

where $\psi_{a,b}(t) = \psi((t-b)/a)$.

Proof. First of all, we have that

$$\int_{-\infty}^{\infty} \psi_{a,b}(t) Rf(\theta,t)\, dt = \int_{-\infty}^{\infty} \psi_{a,b}(t) \int_{\mathbb{R}^2} f(\mathbf{x})\delta(x_1\cos\theta + x_2\sin\theta - t)\, d\mathbf{x}\, dt. \tag{3.130}$$

Now, if we carry out the integral over t, the delta function simply picks out the value $t = x_1\cos\theta + x_2\sin\theta$.

$$\int_{-\infty}^{\infty} \psi_{a,b}(t) Rf(\theta,t)\, dt = \int_{-\infty}^{\infty} \psi_{a,b}(x_1\cos\theta + x_2\sin\theta) f(\mathbf{x})\, d\mathbf{x} \tag{3.131}$$

$$= \int_{-\infty}^{\infty} \psi_{a,b,\theta}(\mathbf{x}) f(\mathbf{x})\, d\mathbf{x}. \tag{3.132}$$

□

This shows us that the CRT is just the 1-D CWT of the Radon transform. This will allow us to easily define the ridgelet transform.

3.6.3 Discrete Ridgelet Transform

In the previous section we defined the DRAT which is a version of the discrete Radon transform which, when used on a $p \times p$ image, produces a $p \times (p+1)$. The Discrete Ridgelet Transform (DRIT) is obtained by simply applying a 1-D DWT to each column in this matrix.

In the same way that a regular 1-D DWT provides a sparse representation for signals with large periods of small deviations from a mean value, the DRIT will provide a sparse representation for signals which are roughly constant for large stretches of straight lines with different angles. This is best shown through an example.

Example 3.6.8. Let us examine the compressive properties of the DRIT. Once again we will use a standard test image. An image which will highlight the properties of the DRIT is called "straws", this image is shown in figure 3.14, and can be found in [6]. Figure 3.15 shows the relative size of the coefficients (ordered from largest to smallest), computed as

$$\log\left(\frac{x_i}{\|\mathbf{x}\|_2}\right).$$

We see that the coefficients for the DRIT are centered on a few large coefficients, and the remaining coefficients are much smaller than for the regular DWT. We also show here that the DRAT coefficients are not sparse.

Finally, figure 3.16 show images where 80%, 90% and 95% of the coefficients are zeroed out. This image has quite a lot of features, compared to for instance "Peppers" or "Lena", so we are not able to obtain very good results when compressing the image. However, many of the details of the image are more apparent with the DRIT compression.

♣

Figure 3.14: A standard test image, "Straws".

We end this section by noting that this is only one formulation of the DRIT. In a wider setting, this is referred to as the Orthonormal Discrete Ridgelet Transform, due to the fact that it is orthonormal for signals of length n where $2n+1$ is prime (we will no go into detail on why this is). Other ridgelet transforms are based on other forms of Radon transforms, such as the Fast Slant Stack transform ([31]) and the A review of different ridgelets transforms can be found in [32]. Notably, other transforms tend to give more intuitive ridgelet functions, however, many are transformations of the form $n \times n \to 2n \times 2n$, which makes them less suited for compression, which has been our focus so far. They may prove suitable for CS, however. Figure 3.17 shows the Cartesian representations of single ridgelet coefficients using different Radon transforms.

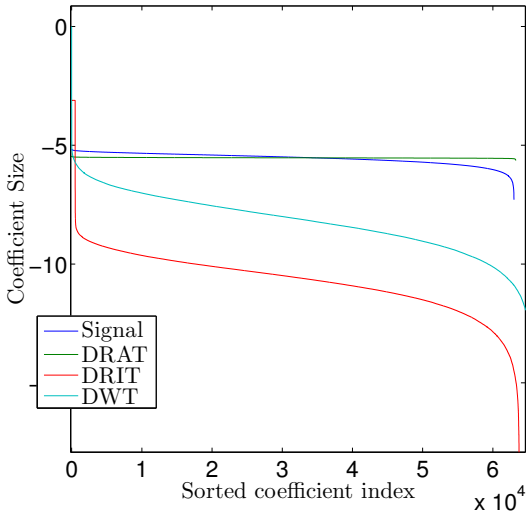
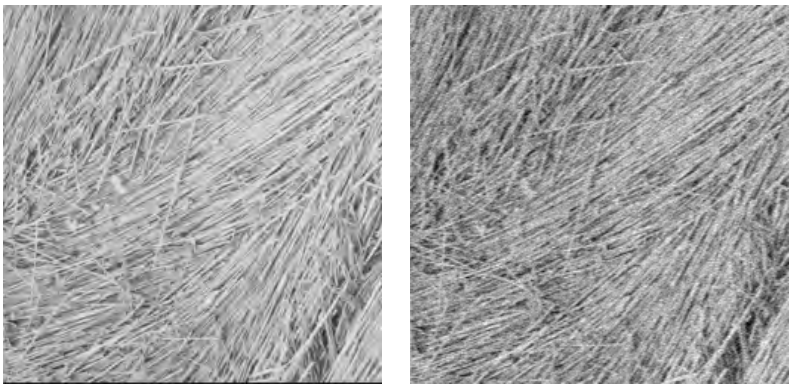


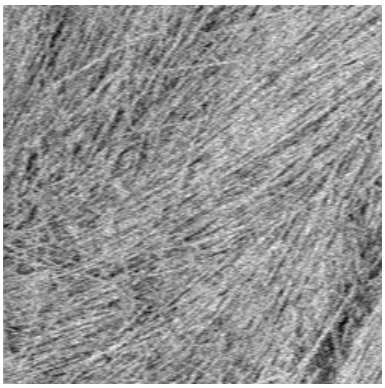
Figure 3.15: Relative mass distribution among the coefficients of the "Straws" image. See text for details.



(a) DWT 80% zeroed out

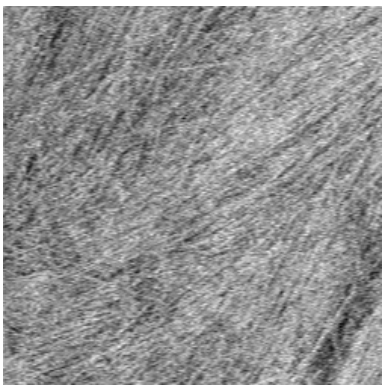
(b) DRIT 80% zeroed out

(c) DWT 90% zeroed out



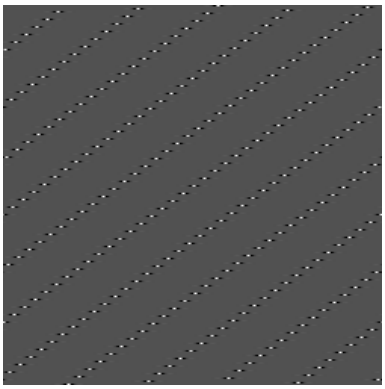
(d) DRIT 90% zeroed out

(e) DWT 95% zeroed out



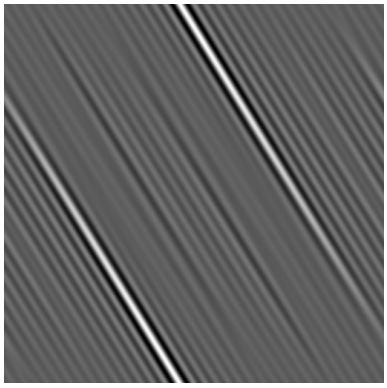
(f) DRIT 95% zeroed out

Figure 3.16: The figures show the result of zeroing out most coefficients in the "Straws" images using different representations.



(a) DRAT as explained in the text

(b) Slant-Stack



(c) Recto-polar

Figure 3.17: The figures single ridgelet coefficients in their Cartesian representation, for different ridgelet formalisms.

3.6.4 1st generation Curvelet Transform

The DRIT provides a good representation for signals where the "ridges" stretch along the entirety of the signal. For realistic signals, this will at best be an approximation. More realistically, the signals will consist of continuous curves. However, on short scales, curves will essentially look like straight lines. By this motivation we can introduce local ridgelets, transforms obtained by splitting the image into small blocks and performing a ridgelet transform on each block. This will usually provide a better result, however, we can take this concept even further. Remember from section 3.4.2 that the detail spaces of the 2-D DWT is usually dominated by mostly small coefficients and some lines of larger coefficients (see figure 3.18 for an illustration).

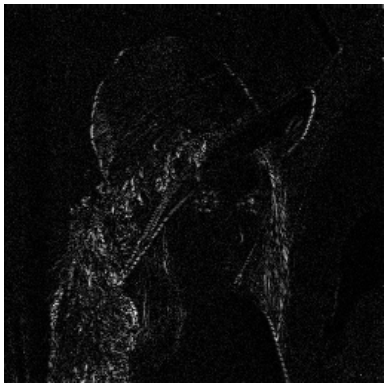


Figure 3.18: A detail space of the lena image

The "First Generation" Discrete Curvelet Transform (DCUT1) is defined by first doing an m_1-level DWT of the signal, then performing a local m_2-level ridgelet transform on each detail space. See figure 3.19 for a schematic overview of this process.

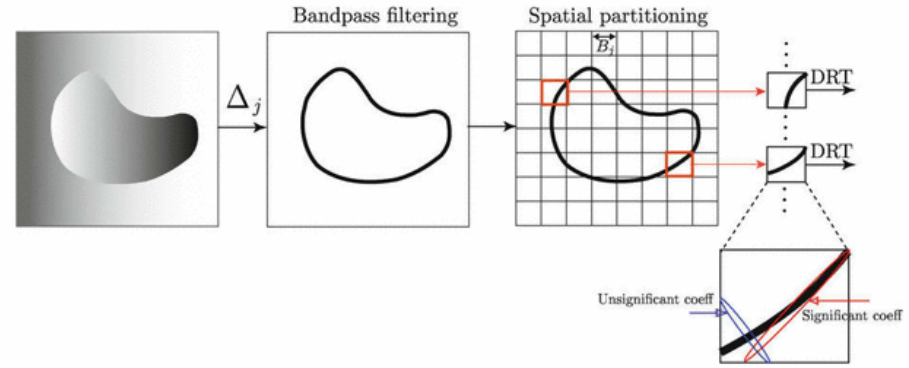


Figure 3.19: An illustration of the steps in the first generation curvelet transform.

However, this process is somewhat convoluted, as the resulting curvelet coefficients would be naturally represented by no less than 7 indices (2 detailing the 2-D DWT space, 2 detailing which block in the block-DRIT transform, 1 detailing the DRAT line and 1 detailing the 1-D DWT space and 1 detailing the position in the DWT space). For this reason, as well as others (see [15] for details), a 2nd generation curvelet has been developed using an entirely different structure, introduced in [15]. To better understand this most recent formulation of the curvelet,

we need to extend our notion of bases.

3.6.5 Frame of a vector space

Frames can be thought of as generalization of bases, to include vector sets which are linearly dependent. For a given vector space V, we have a sequence of vectors $\{\mathbf{e}_k\}$, $\mathbf{e}_k \in V$, and we want to express an arbitrary element $\mathbf{v} \in V$ as a linear combination of the vectors $\{\mathbf{e}_k\}$:

$$\mathbf{v} = \sum_k c_k \mathbf{e}_k. \tag{3.133}$$

In order to do this, we must determine the coefficients c_k. If the set $\{\mathbf{e}_k\}$ does not span V, then no such set exist. If $\{\mathbf{e}_k\}$ spans V, and the elements are also linearly independent, then this set forms a basis of V. In this case, the coefficients c_k are uniquely determined by $\mathbf{v}$. However, if $\{\mathbf{e}_k\}$ spans V but are not linearly dependent, the question of how to determine the coefficients is less clear-cut.

The following is from [33].

Given that $\{\mathbf{e}_k\}$ spans V and is linearly dependent, a strategy is to remove vectors from the set until it becomes linearly independent and forms a basis. There are some problems with this plan:

1. *By removing vectors arbitrarily from the set, it may lose its possibility to span V before it becomes linearly dependent.*
2. *Even if it is possible to devise a specific way to remove vectors from the set until it becomes a basis, this approach may not work in practice for very large or infinite sets.*
3. *In some applications, it may be an advantage to use more vectors than necessary to represent $\boldsymbol{v}$, This means that we want to find the coefficients c_k without removing elements in $\{\boldsymbol{e}_k\}$. The coefficients will then no longer be uniquely determined.*

In 1952, Duffin and Schaeffer gave a solution to this problem, by describing the conditions of a set $\{\boldsymbol{e}\}$ that makes it possible to compute the coefficients c_k in a simple way [34]. More precisely, a frame is a set $\{\boldsymbol{e}_k\}$ of elements which saitisfies the so-called frame condition:

Definition 3.6.9 (Frame Condition)**.** The frame condition states that for a set of vectors $\{\mathbf{e}_k\}$ in a vector space V, there exist two real numbers A and B such that $0 < A \leq B\infty$ and

$$A\|\mathbf{v}\|_2^2 \leq \sum |\langle \mathbf{v}, \mathbf{e}_k \rangle|^2 \leq B\|\mathbf{v}\|_2^2 \tag{3.134}$$

for all $\mathbf{v} \in V$. This means that the constants A and B can be chosen independently of $\mathbf{v}$, they only depend on the set $\{\mathbf{e}_k\}$. A vector set which obeys the frame condition is called a frame of V.

The numbers A and B are called lower and upper frame bounds, respectively. We say that a frame is *overcomplete* or *redundant*. It can be shown that the frame condition entails the existence of a set of *dual frame vectors* $\{\widetilde{\mathbf{e}}_k\}$ with the property that

$$\mathbf{v} = \sum_k \langle \mathbf{v}, \widetilde{\mathbf{e}}_k \rangle \mathbf{e}_k = \sum_k \langle \mathbf{v}, \mathbf{e}_k \rangle \widetilde{\mathbf{e}}_k \tag{3.135}$$

A frame is *tight* if the frame bounds A and B can be set equal. In this case the fame obeys a generalized Parseval's Identity. For example, if $\{\mathbf{e}_k\}$ is the union of 2 orthonormal bases, then the set is a tight vector frame with $A = B = 2$.

The second generation curvelet transform forms a tight frame, as we will see.

3.6.6 The second generation curvelet transform

The continuous 1-D wavelet transforms considers a family of dilated and translated functions $\psi_{m,n}(x) = 2^{m/2}\psi(2^m x - n)$, generated by one mother wavelet ψ. Each function $f(x) \in V_N$ can be written as $\sum_{m,n} c_{m,n}\psi_{m,n}$, where $c_{m,n} = \langle f, \psi_{m,n}\rangle$. Note that the CTFT of the basis functions have the form

$$\hat{\psi}_{m,n}(\xi) = 2^{-m/2} e^{i2^{-j}\xi n}\hat{\psi}(2^{-m}\xi), \tag{3.136}$$

which means that a dilation by 2^j in the space domain corresponds to a dilation by 2^{-j} in the frequency domain, and the translation corresponds to a phase shift.

For a good frequency localization of the wavelet basis, the main idea is to construct a wavelet basis that provides a partition of the frequency axis into almost disjoint frequency bands. In such a construction, each wavelet sub-band gives information about f in different frequency ranges. Such a partition can be ensured if the Fourier transform of the dyadic wavelet $\hat{\psi}$ has a localized or even compact support and satisfies the admissibility condition

$$\sum_m |\hat{\psi}(2^{-m}\xi)|^2 = 1, \tag{3.137}$$

for all ξ. This admissibility condition also ensures the typical wavelet property $\hat{\psi}(0) = \int_{-\infty}^{\infty} \psi(x)\,dx = 0$.

A particularly good frequency location is obtains if $\hat{\psi}$ is compactly supported in $[-2, -1/2] \cup [1/2, 2]$. Such a construction has been used for the so-called Meyer wavelets, which is defined in the frequency domain as

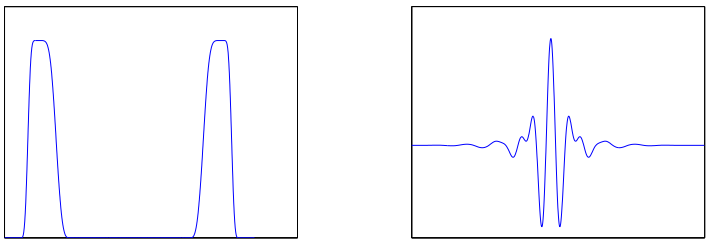
$$\hat{\psi}(\omega) := \begin{cases} \frac{1}{\sqrt{2\pi}} \sin\left(\frac{\pi}{2}\nu\left(\frac{3|\omega|}{2\pi} - 1\right)\right) e^{j\omega/2} & \text{if } 2\pi/3 < |\omega| < 4\pi/3, \\ \frac{1}{\sqrt{2\pi}} \cos\left(\frac{\pi}{2}\nu\left(\frac{3|\omega|}{4\pi} - 1\right)\right) e^{j\omega/2} & \text{if } 4\pi/3 < |\omega| < 8\pi/3, \\ 0 & \text{otherwise,} \end{cases} \tag{3.138}$$

where $\nu(x)$ is a smooth function satisfying

$$\nu(x) := \begin{cases} 0 & \text{if } x = 0, \\ \nu(1-x) & \text{if } 0 < x < 1, \\ 1 & \text{if } x = 1, \end{cases} \tag{3.139}$$

and shown in figure 3.20. In turn, the dilated Meyer wavelets $\hat{\psi}(2^{-m}\xi)$ generate a tiling of the frequency axis into frequency bands in the intervals $[-2^{m+1}, -2^{m-1}] \cup [2^{j-1}, 2^{j+1}]$. In this case, for a fixed ξ, at most two wavelet functions overlap in the Fourier space. This condition also ensures that the family of functions $\{\psi_{m,n} : m, n \in \mathbb{Z}\}$ forms a tight frame of square-integrable functions on $\mathbb{R}$.

We want to transfer this notion to the construction of a 2-D transform which also incorporates a rotation invariance. So, we wish to construct a frame, generated by one basic element,



(a) Frequency representation (b) Time representation

Figure 3.20: The Meyer wavelet in 3.20(a)) the frequency domain, and 3.20(b)) the time representation.

ϕ this time using translations, dilations and rotations of ϕ. Following the considerations of the 1-D case, we also want the elements of the curvelet family to provide a tiling of the 2-D frequency space. Therefore the curvelet transform is now based on the following two main ideas:

1. Consider polar coordinates in the frequency domain.
2. Construct curvelet elements being locally supported near wedges according to figure 3.21, where the number of wedges is $N_j = 4 \times 2^{\lceil j/2 \rceil}$ at the scale 2^{-j} (it doubles in ever second ring).

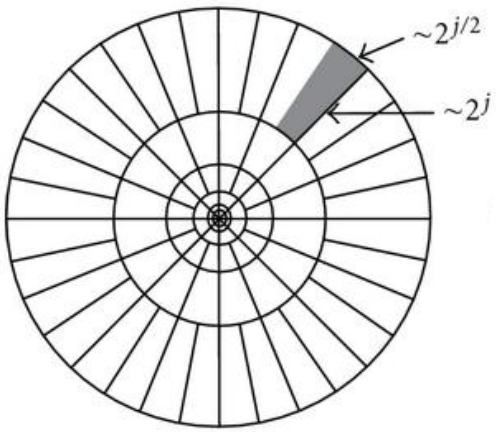


Figure 3.21: Tiling of the frequency domain. A particular curvelet function will be compactly supported on a tile.

Let $\boldsymbol{\xi} = (\xi_1, \xi_2)^T$ be the variable in frequency domain, and $r = \sqrt{\xi_1^2 + \xi_2^2}, \omega = \arctan \frac{\xi_1}{\xi_2}$ be the polar coordinates in the frequency domain. We use the ansatz for the dilated basic curvelets in polar coordinates:

$$\hat{\phi}_{j,0,0}(r,\omega) = 2^{\frac{-3j}{4}} W(2^{-j}r)\widetilde{V}_{N_j}(\omega), \quad r \geq 0, \quad \omega \in [0, 2\pi), \quad j \in N_0, \tag{3.140}$$

To construct a basic curvelet with compact support near a "basic wedge", the two windows W and $\widetilde{V}_{N_j}$. Here, we can simply take $W(r)$ to cover $(0, \infty)$ with dilated curvelets, and $\widetilde{V}_{N_j}$

such that each circular ring is covered by the translations of $\widetilde{V}_{N_j}$. Here, we have $r \in [0, \infty)$, therefore we cannot take the complete Meyer wavelet to determine W, but only the part that is supported in $[1/2, 2]$. Then the admissibility condition yields

$$\sum_{j=-\infty}^{\infty} |W(2^{-j}r)|^2 = 1, \quad r \in (0, \infty). \tag{3.141}$$

We will get back to a more explicit construction of $W(r)$ later.

Further, for the tiling of a circular ring into N wedges, where N is an arbitrary positive integer, we need a 2π-periodic nonnegative window $\widetilde{V_N}$ with support inside $[-2\pi/N, 2\pi/N]$ such that

$$\sum_{l=0}^{N-1} \widetilde{V}_N^2 \left(\omega - \frac{2\pi l}{N}\right) = 1, \quad \text{for all } \omega \in [0, 2\pi), \tag{3.142}$$

is satisfied. Then only two "neighbored" translates of $\widetilde{V}_N^2$ in the sum overlap. Such windows $\widetilde{V}_N$ can be simply constructed as 2π-periodizations of a scaled window $V(N\omega/2\pi)$, where V will be given in the next section.

In this way we approach the goal to get a set of curvelet function with compact support in frequency domain in wedges, where in the circular ring that corresponds to the scale 2^{-j} the sum of the squared rotated curvelet functions depend only on $W(2^{-j}r)$, i.e., it follows that

$$\sum_{l=0}^{N_j-1} \left| 2^{3/4} \hat{\phi}_{j,0,0} \left(r, \omega - \frac{2\pi l}{N_j}\right)\right|^2 = |W(2^{-j}r)|^2 \sum_{l=0}^{N_j-1} \widetilde{V}_{N_j}^2 \left(\omega - \frac{2\pi l}{N_j}\right) \tag{3.143}$$

$$= |W(2^{-j}r)|^2. \tag{3.144}$$

Because of this separability, together with the admissibility condition for W gives an admissibility condition of the basic curvelets $\hat{\phi}_{j,0,0}$ similar to that of the 1-D wavelets. Note that the translates of $\phi_{j,0,0}$ correspond to phase shifts in $\hat{\phi}_{j,0,0}$, so they have no impact on the frequency support.

We should also attend to the "hole" that arises in the origin of the frequency plane, since the rotations of the dilated basic curvelets work only in the scales 2^{-j} for $j = 0, 1, 2, \ldots$. Taking now all scaled and rotated curvelet elements together with $N_j = 4 \times 2^{\lceil j/2 \rceil}$ we find for the scales 2^{-j}, $j = 0, 1, \ldots$ with the admissibility condition

$$\sum_{j=0}^{\infty} \sum_{l=0}^{N_j-1} \left| 2^{3/4} \hat{\phi}_{j,0,0} \right|^2 = \sum_{j=0}^{\infty} |W(2^{-j}r)|^2, \tag{3.145}$$

which is equal to 1 only for $r > 1$ (notice the different summation limits). For a complete covering of the frequency plane, we therefore need to define a low-pass element

$$\hat{\phi}_{-1}(\boldsymbol{\xi}) = W_0(|\boldsymbol{\xi}|) \quad \text{with} \quad W_0^2(r)^2 = 1 - \sum_{j=0}^{\infty} W(2^{-j}r)^2, \tag{3.146}$$

which is supported on the unit circle and where we do not consider any rotation.

Window Functions

For constructing the curvelet functions we use the following window functions. Let us consider the following scaled scaled Meyer windows

$$V(\omega) = \begin{cases} 1 & |\omega| \leq 1/3, \\ \cos[\frac{\pi}{2}\nu(3|\omega| - 1)] & 1/3 \leq |\omega| \leq 2/3, \\ 0 & \text{else}, \end{cases} \tag{3.147}$$

and

$$W(r) = \begin{cases} \cos[\frac{\pi}{2}\nu(5 - 6r)] & 2/3 \leq r \leq 5/6, \\ 1 & 5/6 \leq r \leq 4/3, \\ \cos[\frac{\pi}{2}\nu(3r - 4)] & 4/3 \leq r \leq 5/3, \\ 0 & \text{else}. \end{cases} \tag{3.148}$$

These two functions satisfies the conditions

$$\sum_{l=-\infty}^{\infty} V^2(\omega - l) = 1, \quad \omega \in \mathbb{R}, \tag{3.149}$$

$$\sum_{j=-\infty}^{\infty} W^2(2^l r) = 1. \tag{3.150}$$

The 2π-periodic window functions $\widetilde{V}_N(\omega)$ needed for curvelet construction, can now be obtained as a 2π-periodization of $V(N\omega/2\pi)$.

How many wedges should be taken in one circular ring

As we have seen in figure 3.21, for the curvelet construction there are $N_j = 4 \times 2^{\lceil j/2 \rceil}$ angles (or wedges) chosen in the circular ring (with radius $2^{j-1/2} \leq r \leq 2^{j+1/2}$) corresponding to the 2^{-j}-th scale. But looking at the above idea to ensure the admissibility condition for a tight frame, one is almost free to choose the number of wedges/angles in each scale. Principally, the construction works for all ratios of angles and scales. In fact this is an important point, where the curvelets differ from other constructions mentioned earlier.

1. If we take the number of wedges in a fixed way, independent of the scale, we essentially obtain steerable wavelets.

2. If the number of wedges increases like 1/scale (i.e., like 2^j) then we obtain tight frames of ridgelets.

3. If the number of wedges increases like $\sqrt{1/\text{scale}}$ (like $2^{j/2}$), the curvelet frame is obtained. This special anisotropic scaling law yields the typical curvelet elements whose properties are considered next.

This special anisotropic scaling law yields the typical curvelet elements whose properties are considered next.

What properties do the curvelet elements have?

To obtain the complete family of curvelet functions, we need to consider rotations and the translations of the dilated basic curvelets $\phi_{j,0,0}$. We choose

- an equidistant sequence of rotation angles $\theta_{j,l}$,

$$\theta_{j,l} = \frac{\pi l 2^{\lceil j/2 \rceil}}{2}, \quad \text{with } l = 0, 1, \ldots, N_j - 1. \tag{3.151}$$

- the positions $\mathbf{b}_{\mathbf{k}}^{j,l} = \mathbf{b}_{k_1,k_2}^{j,l} = R_{\theta j,l}^{-1}(k_1/2^j, k_2/2^j)^T$, where $k_1, k_2 \in \mathbb{Z}$ and R_θ denotes the rotation matrix with angle θ.

The family of curvelet functions is then given by

$$\phi_{j,\mathbf{k},l}(\mathbf{x}) = \phi_{j,0,0}(R_{\theta_{j,l}}(\mathbf{x} - \mathbf{b}_{\mathbf{k}}^{j,l})), \tag{3.152}$$

with indices $j = 0, 1, \ldots$, and k_1, k_2, l as above.

The curvelet elements have the following properties:

- Support in the frequency domain. In the frequency domain, the curvelet function $\hat{\phi}_{j,\mathbf{k},l}$ is supported inside the polar wedge with radius $2^{j-1} \leq r \leq 2^{j+1}$ and angle $2^{-\lceil j/2 \rceil}\pi(-1 - l)/2 < \omega < 2^{-\lceil j/2 \rceil}\pi(1 - l)/2$. The support does not depend on the $\mathbf{b}_{\mathbf{k}}^{j,l}$, which only corresponds to a phase shift in the frequency domain. The support of some curvelets is shown in figure 3.22.

- Support in the time domain and oscillatory features. In the time domain, things are more involved. Since $\hat{\phi}_{j,\mathbf{k},l}$ cannot have compact support in the time domain. From Fourier analysis, one knows that the decay of $\phi_{j,\mathbf{k},l}(\mathbf{x})$ for large $|\mathbf{x}|$ depends on the smoothness of $\hat{\phi}_{j,\mathbf{k},l}$ in the frequency domain. The smoother the $\hat{\phi}_{j,\mathbf{k},l}$, the smoother the decay.

  By definition, $\hat{\phi_{j,0,0}}(\boldsymbol{\xi}), j \in \mathbb{N}_0$, is supported away from the vertical axis $\xi_1 = 0$, but near the axis $\xi_2 = 0$. Hence for large j the function $\phi_{j,0,0}(\mathbf{x})$ is less oscillatory in the x_2-direction, and very oscillatory in the x_2-direction

- Tight frame property. The system of curvelets

$$\{\phi_{-1,\mathbf{k},0} : \mathbf{k} \in \mathbb{Z}\} \bigcup \{\phi_{j,\mathbf{k},l} : j \in \mathbb{N}_0, l = 0, \ldots, 4 \times 2^{\lceil j/2 \rceil} - 1, \mathbf{k} = (k_1, k_2)^T \in \mathbb{Z}^2\} \tag{3.153}$$

  satisfies a tight frame property. This means that every square integrable function f can be written as

$$f(\mathbf{x}) = \sum_{j,\mathbf{k},l} \langle f, \phi_{j,\mathbf{k},l} \rangle \phi_{j,\mathbf{k},l} \tag{3.154}$$

  and the Parseval identity

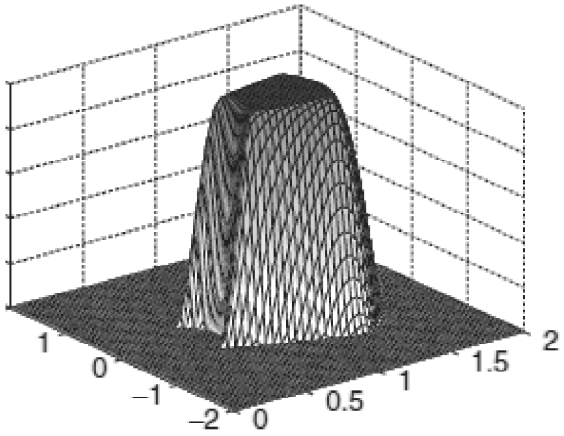
$$\sum_{j,\mathbf{k},l} |\langle f, \phi_{j,\mathbf{k},l} \rangle|^2 = \|f\|_{L^2(\mathbb{R}^2)} \tag{3.155}$$

holds. The terms $c_{j,\mathbf{k},l}(f) = \langle f, \phi_{j,\mathbf{k},l} \rangle$ are called curvelet coefficients. By Plancherel's Theorem, we have

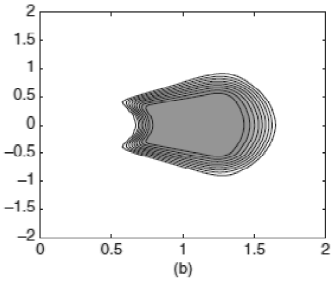
$$c_{j,\mathbf{k},l}(f) = \int_{\mathbb{R}^2} f(\mathbf{x}) \overline{\phi_{j,\mathbf{k},l}(\mathbf{x})}\, d\mathbf{x} \tag{3.156}$$

$$= \int_{\mathbb{R}^2} \hat{f}(\boldsymbol{\xi}) \overline{\hat{\phi}_{j,\mathbf{k},l}(\boldsymbol{\xi})}\, d\boldsymbol{\xi} \tag{3.157}$$

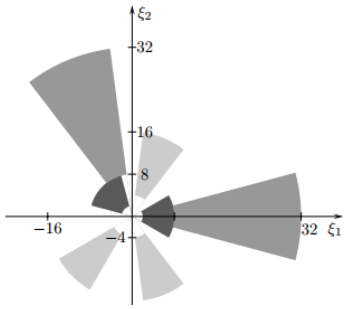
$$= \int_{\mathbb{R}^2} \hat{f}(\boldsymbol{\xi}) \hat{\phi}_{j,0,0}(R_{\theta_{j,l}}\boldsymbol{\xi}) \exp(i\mathbf{b}_{\mathbf{k}}^{j,l} \cdot \boldsymbol{\xi})\, d\boldsymbol{\xi}. \tag{3.158}$$



(a) $\hat{\phi}_{0,0,0}$



(b) support of $\hat{\phi}_{0,0,0}$



(c) support of some curvelet functions

Figure 3.22: Some illustrations of curvelets in the frequency domain. 3.22(c) shows the maximal support of $\hat{\phi}_{2,\mathbf{k},0}$ and $\hat{\phi}_{2,\mathbf{k},5}$ (dark gray), of $\hat{\phi}_{3,\mathbf{k},3}$, $\hat{\phi}_{3,\mathbf{k},6}$ and $\hat{\phi}_{3,\mathbf{k},13}$ (light gray), and of $\hat{\phi}_{4,\mathbf{k},0}$ and $\hat{\phi}_{4,\mathbf{k},11}$ (gray). The figures are from [35].

3.6.7 The Discrete 2nd Generation curvelet Transform

In practical implementations, one would like to have Cartesian arrays instead of the polar tiling of the frequency plane. Cartesian tilings based on squares and shears. Therefore, a construction based on window functions on trapezoids instead of polar wedges are preferable.

We now use the following new ansatz for the discrete 2nd generation curvelet transform (DCUT2):

$$\hat{\tilde{\phi}}_{j,0,0}(\boldsymbol{\xi}) = 2^{-3/4} W(2^{-j}\xi_1) V\left(\frac{2^{\lfloor j/2 \rfloor}\xi_2}{\xi_1}\right), \tag{3.159}$$

with W the same as earlier, and V with compact support in $[-2/3, 2/3]$. The modified curvelet is shown in figure 3.23(a) and its support in figure 3.23(b). The support of $V_j(\boldsymbol{\xi}) = V(2^{\lfloor j/2 \rfloor}\xi_2/\xi_1)$ is now inside the cone $K_1 = \{(\xi_1, \xi_2) : \epsilon_1 > 0, \xi_2 \in [-2\xi_1/3, 2\xi_1/3]\}$. The basic curvelet has support in the trapezoid

$$\left\{(\xi_1, \xi_2) : 2^{j-1} \le \xi_1 \le 2^{j+1}, -2^{-\lfloor j/2 \rfloor}\frac{2}{3} \le \frac{\xi_2}{\xi_1} \le 2^{-\lfloor j/2 \rfloor}\frac{2}{3}\right\}. \tag{3.160}$$

The rotation is now replaced by shearing of the basic wavelet in each hemisphere. For the eastern hemisphere, we define a set of equispaced slopes

$$\tan\theta_{j,l} = l2^{-\lfloor j/2 \rfloor}, \quad l = -2^{\lfloor j/2 \rfloor} + 1, \ldots, 2^{\lfloor j/2 \rfloor} - 1. \tag{3.161}$$

The curvelets for the other hemispheres are constructed by a rotation of $\pi/2$ and a reflection. Observe that the angles are not equispaced, but the slopes are.

The modified curvelets are now given by

$$\tilde{\phi}_{j,\mathbf{k},l}(\mathbf{x}) = \tilde{\phi}_{j,0,0}(S^T_{\theta_{j,l}}(\mathbf{x} - \tilde{\mathbf{b}}^{j,l}_{\mathbf{k}})), \tag{3.162}$$

where

$$S_\theta = \begin{pmatrix} 1 & 0 \\ -\tan\theta & 0 \end{pmatrix}, \tag{3.163}$$

and where $\tilde{b}^{j,l}_{\mathbf{k}} = S^{-T}_{\theta_{j,l}}(k_1 2^{-j}, k_2 2^{-\lfloor j/2 \rfloor}) = S^{-T}_{\theta_{j,l}}\mathbf{k}_j$ denotes the position in the space domain.

The Cartesian curvelet coefficients are given by

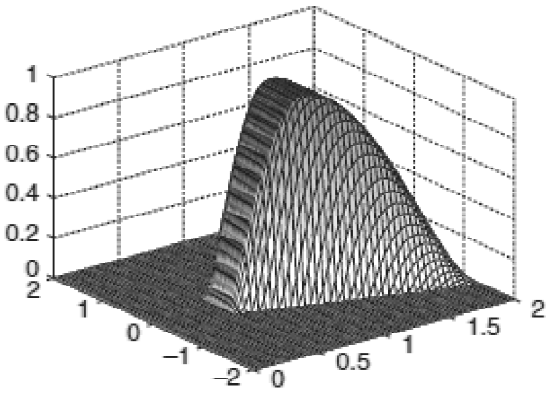
$$\tilde{c}_{j,\mathbf{k},l} = \langle f\hat{\tilde{\phi}}_{j,\mathbf{k},l}\rangle = \int_{\mathbb{R}^2} \hat{f}(\boldsymbol{\xi})\hat{\tilde{\phi}}_{j,0,0}S^{-1}_{\theta_{j,l}\boldsymbol{\xi}} e^{i\langle \widetilde{\mathbf{b}^{j,l}_{\mathbf{k}}}, \boldsymbol{\xi}\rangle}\, d\boldsymbol{\xi} \tag{3.164}$$

$$= \int_{\mathbb{R}^2} \hat{f}(S_{\theta_{j,l}}\boldsymbol{\xi})\hat{\phi}_{j,0,0}\boldsymbol{\xi} e^{i\langle \mathbf{k}_j, \boldsymbol{\xi}\rangle}\, d\boldsymbol{\xi}. \tag{3.165}$$

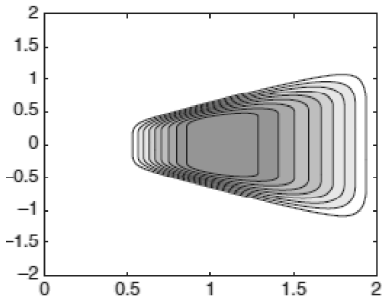
For an implementation for discrete data, we calculate $\hat{f}(S_{\theta_{j,l}}\boldsymbol{\xi})$ by interpolation. Implementations of numerical order$\mathcal{O}(N^2 \log_2 N)$ for an $N \times N$ image are available at www.curvelet.org. Figure 3.24 shows an example of a discrete curvelet in the position space.

3.6.8 Wave Atoms

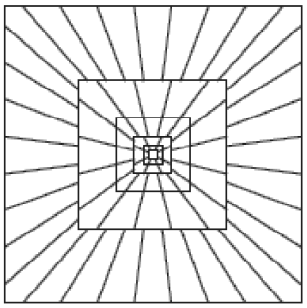
Finally, we briefly mention a construction similar to the curvelets, called Wave Atoms. The wave atoms have the same scaling features in the frequency space, but in addition to the ridges featured in the curvelet coefficients, the wave atom coefficients contain oscillatory features in the direction perpendicular to the ridge. Such signals naturally occur many places in nature, from seismic data to fingerprints. Figure 3.25 shows an example of a wave atom in the position space.



(a) $\hat{\tilde{\phi}}_{0,0,0}$



(b) support of $\hat{\tilde{\phi}}_{0,0,0}$



(c) Cartesian polar tiling.

Figure 3.23: Some illustrations of curvelets in the Cartesian-tiled frequency domain. The figures are from [35].

3.7 Basis representation in a Signal Processing setting

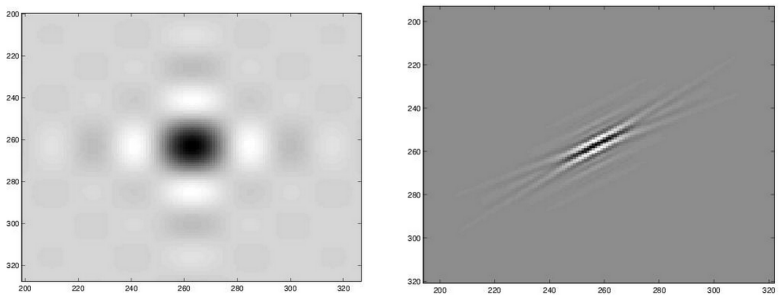
To end this chapter, we note that while we have used the standard terminology found in linear algebra literature, this differs somewhat from the terminology used when discussing the same topics in signal processing literature. Here we will go through some of the most common terminology used in signal processing.

3.7.1 Atom

An atom refers to a single function in a basis (for example a single complex exponential in the DFT, a single square wave in the Haar DWT, or a single curvelet function in the curvelet transform).

3.7.2 Dictionary

A dictionary $\mathbf{\Phi}$ is an indexed collection of atoms $\{\phi_i\}_{i\in\Gamma}$, where Γ may be a finite or countable infinite set, with cardinality $|\Gamma| = M$ In discrete-time finite-length signal processing, $\mathbf{\Phi}$ is an



(a) A low-resolution DWT used in the DCUT2

(b) A curvelet coefficient used in the DCUT2

Figure 3.24: Some illustrations of curvelets in the position space.

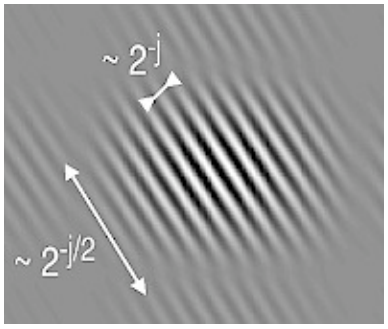


Figure 3.25: Example of a wave atom in the position space.

$M \times N$ matrix. The dictionary may be complete (like a basis), overcomplete or *redundant*, like a frame. In this case the equation $\mathbf{x} = \mathbf{\Phi y}$ is an underdetermined system of linear equations. A dictionary may also be incomplete, which is the case where the dictionary matrix cannot be reduced to a basis.

3.7.3 Analysis and Synthesis

Given a dictionary, the *analysis* operation corresponds to what we normally would call a forward transform in linear algebra, $\mathbf{y} = \mathbf{\Phi}^\dagger \mathbf{x}$. *Synthesis* is the inverse transform, $\mathbf{x} = \mathbf{\Phi y}$. In the overcomplete case, $\mathbf{\Phi}$ is not invertible, and the reconstruction is not unique. The only application for transforms we have looked into so far is signal compression, for which overcomplete representations are not very useful. For several other applications, such as signal analysis, reconstruction and denoising, overcomplete representations will prove extremely useful.

Chapter 4

Theory of Compressed Sensing

In this chapter our goal will be to present the results of recently emerged theory of compressed sensing. In order to do this in a meaningful way, we need to put the theory into the proper context. CS is far from the only method for attempting to insert data based on prior information of the signal. CS belongs to a larger class of problems called Linear Inverse Problems (LIPs), which we will now introduce.

4.1 Linear Inverse Problems

Many problems in signal processing (and more generally, in computational science), can be cast as inverting the linear system of equations

$$\mathbf{b} = A\mathbf{x} + \boldsymbol{\epsilon}, \tag{4.1}$$

where $\mathbf{x} \in \mathbb{C}^N$ is the data we want to recover $\mathbf{b} \in \mathbb{C}^M$ is some set of observations which are made made on the signal $\mathbf{x}$, that have been polluted by noise. The error $\boldsymbol{\epsilon}$ can be either stochastic measurement noise or systematic measurement error (as in the form of a constant perturbation), but is generally not known. $A : \mathbb{C}^N \to \mathbb{C}^M$ is a linear operator which describes the measurements on the signal. A typically represents an underdetermined set of equations. Such problems are said to be *ill-posed*, as they do not have a unique solution.

If we at first assume the signal is noiseless, then one particular solution to this system is given by

$$\mathbf{x}^* = A^\dagger (AA^\dagger)^{-1}\mathbf{b}. \tag{4.2}$$

(it can be shown that if A is an $M \times N$ matrix with $M < N$, then $(AA^\dagger)$ is invertible). The matrix $A^\dagger(AA^\dagger)^{-1}$ is called the *pseudoinverse* of A. This solution solution has a very specific property.

Theorem 4.1.1. Given an underdetermined system

$$A\mathbf{x} = \mathbf{b}, \tag{4.3}$$

the solution

$$\mathbf{x}^* = A^\dagger (AA^\dagger)^{-1}\mathbf{b}, \tag{4.4}$$

has the least ℓ_2 (Euclidean) norm.

PROOF. Let x be a solution, so that $A(x - x^*) = 0$. Then

$$(x - x^*)^\dagger x^* = (x - x^*)^\dagger A^\dagger (AA^\dagger)^{-1}\mathbf{b} \tag{4.5}$$
$$= (A(x - x^*))^\dagger (AA^\dagger)^{-1}\mathbf{b} \tag{4.6}$$
$$= 0. \tag{4.7}$$

This implies that $\langle x - x^*, x\rangle = 0$, which means that we can use the Pythagorean theorem to show that

$$\|x\|^2 = \|x^* + x - x^*\|^2 = \|x^*\|^2 + \|x - x^*\|^2 \geq \|x^*\|^2. \tag{4.8}$$

□

This is sometimes referred to as the "Best solution theorem". The problem with that statement is that for many systems, this solution is completely wrong. Let us illustrate this in an example.

Example 4.1.2 (Partial measurements of a pure tone)**.** Let us consider a simple sine tone of 440 Hz, sampled uniformly for 1 second with a sampling frequency of $f_s = N$, with the aim of finding its Fourier spectrum:

$$x_i = \sin(2\pi 440 t_i), \quad t_i = i/N, \quad i = 0, \ldots, N. \tag{4.9}$$

Assume that for some reason or another, some of the samples are lost, and we are left with a set of M samples indexed by $\Lambda = \lambda_1, \lambda_2, \ldots, \lambda_M$, where $\lambda_j \in \{0, 1, \ldots, N\}$ and $\lambda_i = \lambda_j$ if and only if $i = j$. Let $\mathbf{x}_\Lambda$ be the vector of length M consisting of only the elements indexed by Λ, and similarly let $F_\Lambda^\dagger$ be the $N \times M$ matrix constructed from keeping only the rows indexed by Λ in an $N \times N$ inverse Fourier matrix. We are then interested in solving the problem

$$\mathbf{x}_\Lambda = F_\Lambda^\dagger \mathbf{y}. \tag{4.10}$$

The "best solution" would then be

$$\mathbf{y} = F_\Lambda (F_\Lambda^\dagger F_\Lambda)^{-1}\mathbf{x}_\Lambda = F_\Lambda \mathbf{x}_\Lambda, \tag{4.11}$$

where we have used that the columns in F_Λ are orthonormal. The result of this for the case $N = 1000$, $M = 100$ and Λ is created by measuring random indices is shown in figure 4.1. While the spikes at $f = 440$ are still apparent, there is a lot of noise in the restored spectrum. The ℓ_2 norm minimizing solution will have the property of corresponding to the signal with the least energy, as we have seen in section 3.1.1. ♣

4.1.1 Finding sparse solutions of LIPs

There is no reason a priori to assume that the ℓ_2-minimizing solution of a LIP is the optimal solution. For many cases, such as the above, we may know beforehand that the restored signal should be very sparse. If we know we are looking for a k-sparse solution, we should tailor our methods to this. Is it possible to accurately reconstruct such a sparse solution? For this to be possible, we must require that there are no other k-sparse vectors $\mathbf{x}$ satisfying $A\mathbf{x} = \mathbf{b}$. The following lemma shows us that it is possible to obtain this uniqueness property.

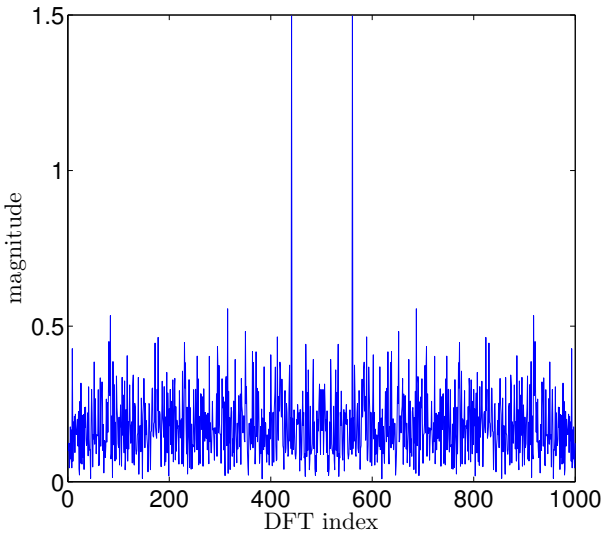


Figure 4.1: A reconstructed Fourier spectrum based on partial time measurements and ℓ_2 minimization.

Lemma 4.1.3. Suppose that A is an $m \times n$ matrix such that every set of $2k$ columns of A are linearly independent. Then a k-sparse vector $\mathbf{x}^k \in \mathbb{C}^n$ can be reconstructed uniquely from $A\mathbf{x}^k = \mathbf{b}$.

PROOF. Assume that $A\mathbf{x} = \mathbf{b}$ and $A\mathbf{y} = \mathbf{b}$ for $\mathbf{x} \neq \mathbf{y}$, where $\mathbf{x}$ and $\mathbf{y}$ are both k-sparse. This implies that $A(\mathbf{x} - \mathbf{y}) = 0$. But $\mathbf{x} - \mathbf{y}$ is $2k$-sparse, so this in turn implies that there are $2k$ linearly dependent columns of A, a contradiction. □

This lemma proves the needed uniqueness, and also suggests a way to solve the problem. Since any $k - j$-sparse vector, $j = 1, \ldots, k$, is also k-sparse, we should look for the sparsest possible solution. As long as A obeys the restriction outlined by the lemma, this approach will give the unique solution $\mathbf{x}^k$. Our aim is then formally

$$\begin{aligned} &\text{minimize} && \|\mathbf{x}\|_0, \\ &\text{subject to} && A\mathbf{x} = \mathbf{b}. \end{aligned} \tag{P0}$$

The problem is that this problem is very hard to solve[1]. If we want to solve (P0) directly, we need to need to try out all possible combinations of placements for the 0s. For a k-sparse vector of size N, without knowledge of exactly what k is, we need to try up to $\sum_{j=0}^{k} \binom{N}{j}$ combinations. So on one hand we hand the ℓ_2 minimization which is easy to do, but gives the wrong result, and on the other we have ℓ_0 minimization, which gives the right result, but is impossible to do in practice.

It turns out that we can avoid the problem of minimizing the ℓ_0 norm. If we instead replace the ℓ_0 norm with the ℓ_1 norm, we will actually in a lot of cases get the same result. ℓ_1 minimization is a well-understood problem, and there exists efficient algorithms for it (not as efficient as the ℓ_2 minimization, but computationally feasible for large systems, unlike the ℓ_0 minimization). The problem we want to solve now becomes

[1]For the mathematically inclined, it is NP-hard.

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{x}\|_1, \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b}. \end{aligned} \tag{P1}$$

The fact that we can often choose to solve (P1) rather than (P0) is a very important point, and as such, we should take some time to explain why it is reasonable.

Figure 3.10 shows the unit circle for the ℓ_0, ℓ_1 and ℓ_2 norms, i.e. the set of points $\{\mathbf{x}\}$ such that $\|\mathbf{x}\|_p = 1$. Note that the ℓ_1 norm is "pointy"; it has edges. these edges coincide with the points on the ℓ_0 norm. This is the key point for why we can use the ℓ_1 norm instead of the ℓ_0 norm. Let us illustrate this with an example.

Example 4.1.4. Consider the 2-dimensional problem

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{x}\|_1, \\ \text{subject to} \quad & x_1 - nx_2 = 1. \end{aligned} \tag{4.12}$$

The solutions to this problem lie along the line $x_2 = (x_1 - 1)/n$. The ℓ_1 and ℓ_2 minimizing solutions for the case $n = 2$ are shown in figure 4.2. Note that the ℓ_1 solution corresponds to the sparest possible solution, and this would be true for any slope. For linear constraints, the same effect happens in higher dimensions as well. Meanwhile, the ℓ_2 minimizing solution will not be the sparsest solution unless $n = 0$, in which case the equation just reads $x_1 = 1$. ♣

Let us return to the problem of restoring the sparse Fourier spectrum from the signal given by (4.9). We can now attempt to restore the signal by ℓ_1-minimization. We will get back to exactly how to do this in section 4.12.2, but here we will just assume some pre-made program can do this for us. The result of such a minimization is shown in figure 4.3. This restores exact signal within numerical errors in this case. We should note that we have performed no analysis of the set of equations defined by $F_\Lambda^\dagger$ to guarantee such a strong performance, so this might be considered a lucky case, for now. We will see later why we get such a good results for the Fourier measurements.

4.1.2 Signals with noise: different variations of the sparsity-seeking LIP

When dealing with exactly sparse signals, lemma 4.1.3 provides a complete characterization of when an sparse recovery is possible. However, real measured signals will usually contain some noise, and thus will not be exactly sparse in a any reasonable basis, even though we might expect the noiseless signal to be exactly. The task is then to solve the equation $A\mathbf{x} = \mathbf{b} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ represents the noise in the measurements. In this case, (P1) can intuitively be altered as

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{x}\|_1, \\ \text{subject to} \quad & \|A\mathbf{x} - \mathbf{b}\|_2 \leq \sigma. \end{aligned} \tag{P1-σ}$$

To use the ℓ_2 norm in the inequality here might seem ambiguous, and it is, but this ensures that "small" deviations from equality spread across all coefficients are acceptable, while large deviations on only a select few coefficients are heavily punished, which seems very reasonable when dealing with noise. The exact ℓ_1-reconstruction is often referred to as Basis Pursuit (BP), while the case for $\sigma > 0$ is called Basis Pursuit De-Noising (BPDN).

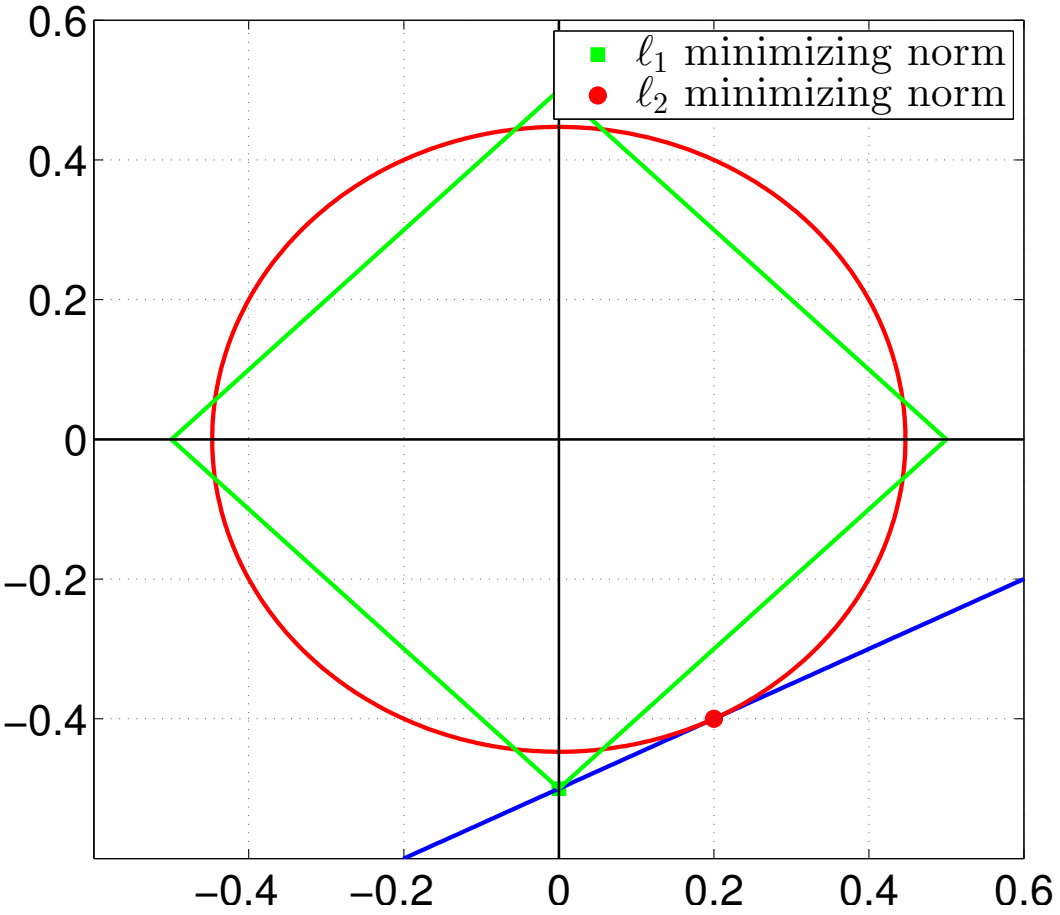


Figure 4.2: The figure shows the norm minimizing solutions of an under-determined system of equations. The solutions lie along the blue line.

Another version, which is often called the Lasso problem and which was explored by [36] is posed as

$$\begin{aligned} \text{minimize} \quad & \|A\mathbf{x} - \mathbf{b}\|_2, \\ \text{subject to} \quad & \|\mathbf{x}\|_1 \leq \tau. \end{aligned} \tag{P1-Lasso}$$

This problem can be (and often is) posed in its a way that is related to its Lagrangian form, which was studied in [37],

$$\text{minimize} \quad \|A\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1. \tag{P1-LL}$$

One should note that while the names of the methods presented here are the historically correct ones, one may find that this last method is referred to as BP, BPDN, the Lagrangian of the Lasso or sometimes just the Lasso in literature. Generally, the BP and BPDN monikers are wildly used for any ℓ_1-minimization scheme.

It can be shown that for appropriate values of σ, τ and λ, these three variations yield the same solution. However, unless A is orthogonal, we can not know a priori what these values are. Let us investigate this in an example.

Example 4.1.5 (Different ℓ_1-minimizing strategies)**.** In this example we will try to restore the vector $\mathbf{x} = (0, 1)^T$, from under-sampled coefficients in some basis, and check that we are able

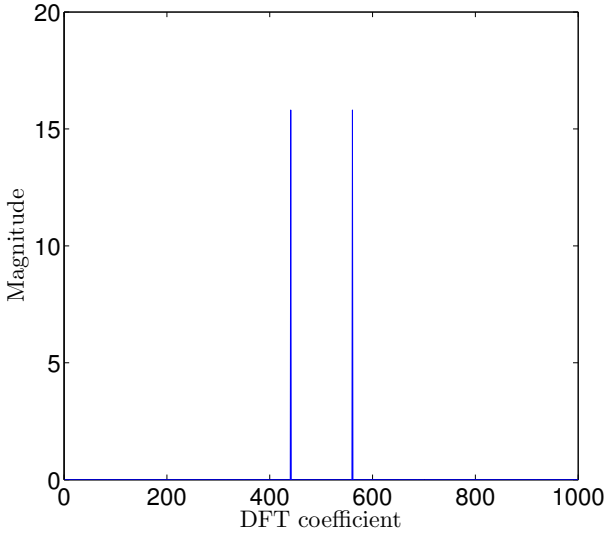


Figure 4.3: A reconstructed Fourier spectrum based on partial time measurements and ℓ_1 minimization.

to find the same result for all methods if we can choose the parameters freely. In this case, we choose the matrix A to be

$$A = \frac{1}{\sqrt{5}} \begin{pmatrix} 1 & 2 \end{pmatrix} \tag{4.13}$$

Which means that $\mathbf{b} = A\mathbf{x} = \frac{2}{\sqrt{5}}$. Let us assume that we only have access to the first element of the matrix. We will investigate this problem using each of the different methods.

1. Let us first find the relaxed ℓ_1-minimization solution, where we choose $\sigma = 0.01$. We want to solve

$$\begin{aligned} &\text{minimize} \quad \|\mathbf{x}\|_1 \\ &\text{subject to} \left\| \frac{1}{\sqrt{5}} \begin{pmatrix} 1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \left(\tfrac{2}{\sqrt{5}} \right) \right\|_2 \leq 0.01. \end{aligned} \tag{4.14}$$

   The minimization constraint can be written as $|x_1 + 2x_2 - 2| \leq \sqrt{5}/100$. It is easy to check that the solution of this problem with the minimal ℓ_1 norm is the point $(0, 1 - \sqrt{5}/200)^T$.

2. Next we look at the Lasso formulation. If we choose $\tau = 1 - \sqrt{5}/200$, We want to solve

$$\begin{aligned} &\text{minimize} \quad \|x_1 + 2x_2 - 2\|_2, \\ &\text{subject to} \quad |x_1| + |x_2| \leq 1 - \sqrt{5}/200. \end{aligned} \tag{4.15}$$

   and it should be clear from figure 4.4 that in this case the solution is once again $(0, 1 - \sqrt{5}/200)^T$.

3. Finally, we solve the Lagrangian form of the Lasso,

$$\text{minimize} \quad (x_1 + 2x_2 - 2)^2/5 + \lambda(|x_1| + |x_2|). \tag{4.16}$$

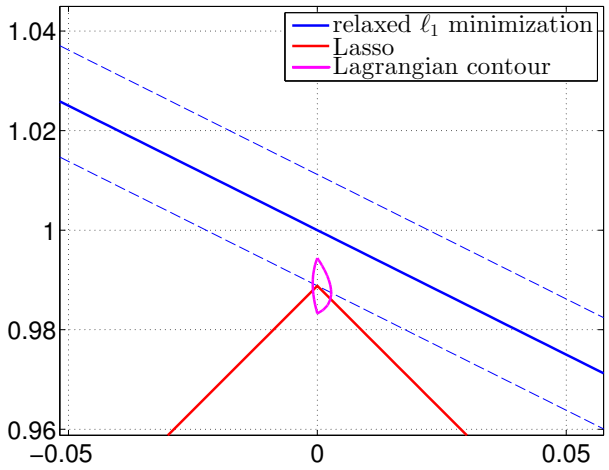


Figure 4.4: Different ℓ_1-minimizing schemes solutions. The blue lines illustrate the relaxed ℓ_1 minimization. The red line illustrates the LASSO solution, where we search for solutions within the requirements $\|\mathbf{x}\|_1 \leq \tau$. The purple line is a contour line for the Lagrangian version of the LASSO, for a value slightly larges than the value we find if we insert $\mathbf{x} = (0, 1 - \sqrt{5}/200)^T$ and $\lambda = 4/(100\sqrt{5})$, as found in the example text. This is meant to illustrate that the Lagrangian version of the LASSO gives the same result.

In this case it is not quite as obvious what value we should assign to λ. However, numerical investigations will show that for $\lambda > 0$, the optimal value for x_1 will always be 0. This leaves us with the expression $(2x_2 - 2)^2/5 + \lambda|x_2|$. If $x_2 > 0$, the optimal value for x_2 is $x_2 = 1 - 5\lambda/8$. If we choose $\lambda = 4/(100\sqrt{5})$, we find the optimal solution once again to be $\mathbf{x}^* = (0, 1 - \sqrt{5}/200)^T$. ♣

Total Variation minimization

For a digital signal $\mathbf{y}$, we can define the Total Variation (TV) as

$$V(\mathbf{y}) = \sum_i |y_{i+1} - y_i|, \tag{4.17}$$

or for a 2-D signal,

$$V(Y) = \sum_{i,j} \sqrt{|y_{i+1,j} - y_{i,j}|^2 + |y_{i,j+1} - y_{i,j}|^2}. \tag{4.18}$$

Minimizing the TV is a popular alternative alternative to minimizing the ℓ_1 norm, for cases where the signal has a sparse gradient. Most notably, this has produced great results for reconstructing images such as the Shepp-Logan phantom, in cases where the Fourier coefficients are sampled along radial lines at different angles (see [38] for a demonstration). The restoration is often done using the exact or relaxed equality constraint with the ℓ_1 minimization replaced with a TV-minimization, or the Lagrangian of the Lasso, with the ℓ_1 norm again replaced. Additionally, another approach, usually called the Dantzig TV minimization, aims to solve the problem

$$\begin{aligned} \text{minimize} \quad & TV(\mathbf{x}), \\ \text{subject to} \quad & \|A^\dagger(A\mathbf{x} - \mathbf{b})\|_\infty \leq \gamma. \end{aligned} \tag{TV-Dantzig}$$

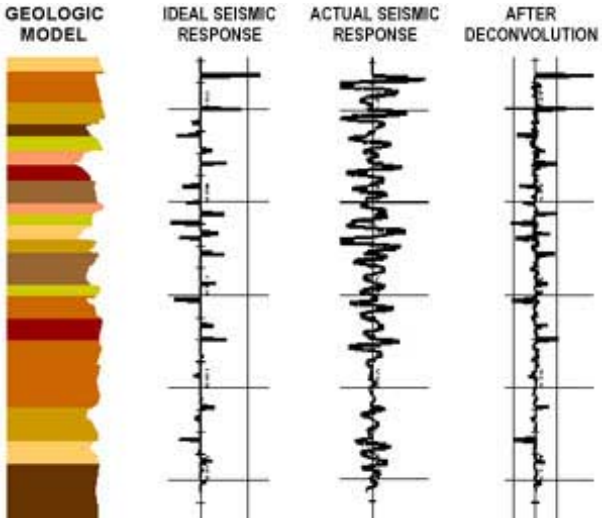


Figure 4.5: A simple model of seismic data acquisition. From [39].

We will get back to the details of solving (P1) and its variations in section 4.12.2. For now we will simply assume some black box is able to solve this problem for us. The rest of this section will be devoted to introducing some common cases of LIPs, and briefly explore how they relate to sparse solutions.

4.1.3 Deconvolution

Deconvolution of sparse spikes is one of the oldest inverse linear problems, and has its origins in recovery in seismic imaging. The ground is modeled as a 1-D profile $\mathbf{x}$, mostly zeros with a few spikes accounting for interfaces between layers in the ground. In order to measure this signal, typically a sound wave is sent down in the material, and for each layer transition, the signal will be partly reflected because the new layer will have a somewhat different propagation speed, as follows from elementary wave mechanics.

The problem is that because the wave used is not perfectly sharp, and because the transition is not perfectly sharp, the actual measurement will be considerably blurred. The task at hand is then to restore the sharp layers. An overview of this problem is shown in figure 4.5.

It is easy to make a "toy model" of this process, here we present the model used in [40]. Let $\mathbf{x}$ be the sparse vector. Then the measured signal can be modeled as

$$\mathbf{y} = D\mathbf{x} + \boldsymbol{\epsilon}, \tag{4.19}$$

where D is some filter representing the spread of the wave, and $\boldsymbol{\epsilon}$ represents measurement noise. Here we choose D to be the second derivative of a Gaussian function,

$$D(t) \propto \left(1 - \frac{t^2}{\sigma^2}\right) \exp\left(\frac{t^2}{2\sigma^2}\right). \tag{4.20}$$

We have briefly seen this filter this earlier, in section 3.3, as the Mexican hat wavelet. In addition, we let D be of size $2N \times N$. This simply reflects the fact that we make a lot of measurements of the signal, and will help ensure a stable recovery. If we let the noise be i.i.d.

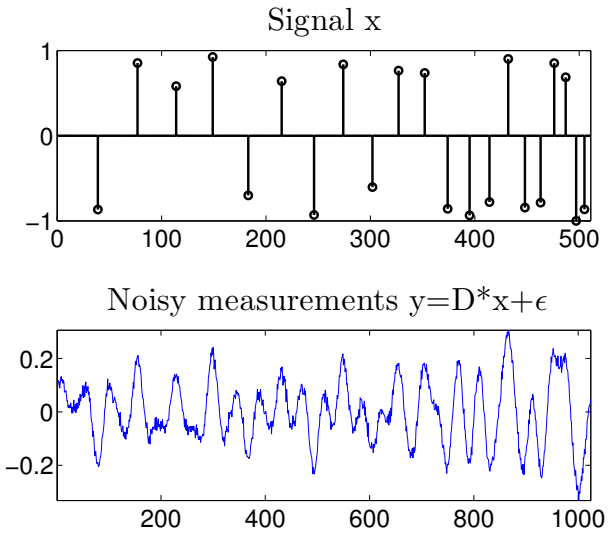


Figure 4.6: A toy seismic signal and its measured counterpart.

Gaussian variables, then the resulting signal may look like in figure 4.6. The task is then to restore the original sparse vector $\mathbf{x}$ from only $\mathbf{y}$.

It is reasonable to attempt to solve this as the optimization problem (P1-σ). The result of such an optimization is shown in figure 4.7. The result, while not exact, is certainly good enough to find the edges of the real signal.

4.1.4 Denoising

Image denoising is one of the most studied fields in DSP. The problem is easy to state. Some image $\mathbf{x}$ has been exposed to some form of additive noise $\boldsymbol{\epsilon}$, resulting in the polluted version of the image $\mathbf{y}$,

$$\mathbf{y} = \mathbf{x} + \boldsymbol{\epsilon}. \tag{4.21}$$

There are many ways to approach the problem of restoring $\mathbf{x}$. Here we will focus on representation based sparsity-promoting methods, often referred to as Thresholding methods.

Consider the Lena image, and its polluted counterpart, as shown in figure 4.8. Figure 4.8 also shows the logarithm of the DWT CDF 9/7 wavelet coefficients, as well as the logarithm of the DWT of the noisy signal. We see here that the noise hits the first detail spaces the hardest, while in the second, third and fourth detail space the features of the image are still apparent. We know that the image will still be of high quality if we set the first detail space to zero. It seems like the DWT is somehow able to sort the noise from the coherent signal.

We will now attempt to remove the noise from the image by simply compressing it; we set all coefficients less than some threshold equal to zero. This approach is called *hard thresholding*, and can be likened to solving the relaxed (P0) problem. The result is shown in figure 4.9. The result is fairly good. We could have introduced a new wavelet which might perform better, but this would detract from the discussion at this point.

It is interesting to repeat this experiment with the curvelets introduced in section 3.6. This will highlight the advantage of overcomplete dictionaries, which was somewhat unmotivated when we had only discussed the application of compression. The result for the DCUT2-based

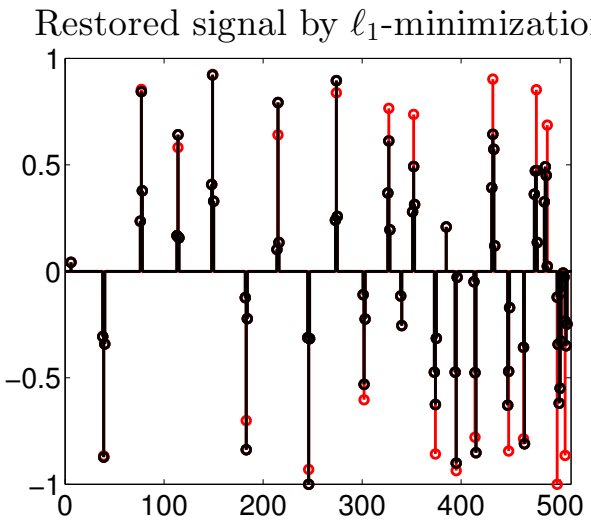


Figure 4.7: A restoration of the toy signal based on (P1-σ). The red points is the original data, while the black points is the restored signal.

	PSNR, $\sigma = 0.1$
Noisy Image	68.1693
DWT-97	75.6077
DCUT2	73.3452

Table 4.1: The results for a simple denoising experiment. The image was normalized to [0,1], and Gaussian noise was added with a standard deviation, σ, as specified.

denoising is shown in figure 4.10. One should note that for this result we have used a different threshold for the coarsest scale, which in the `CurveLab`-implementation is computed as a regular wavelet transform. This is reasonable, as these coefficients are not as strictly related to the rest of the wavelet coefficients as is the case for the DWT. The performance is somewhat poorer that the DWT in terms of error, but one should note that the error is less coherent for the DCUT2, in the sense that the error along edges is larger for the DWT than the DCUT2. One should also note that these test are very basic, and do not provide definitive answers as to which method generally performs better. Such an analysis would, once again, detract from the discussion at hand. The interested reader might look to [41] for a more in-depth analysis.

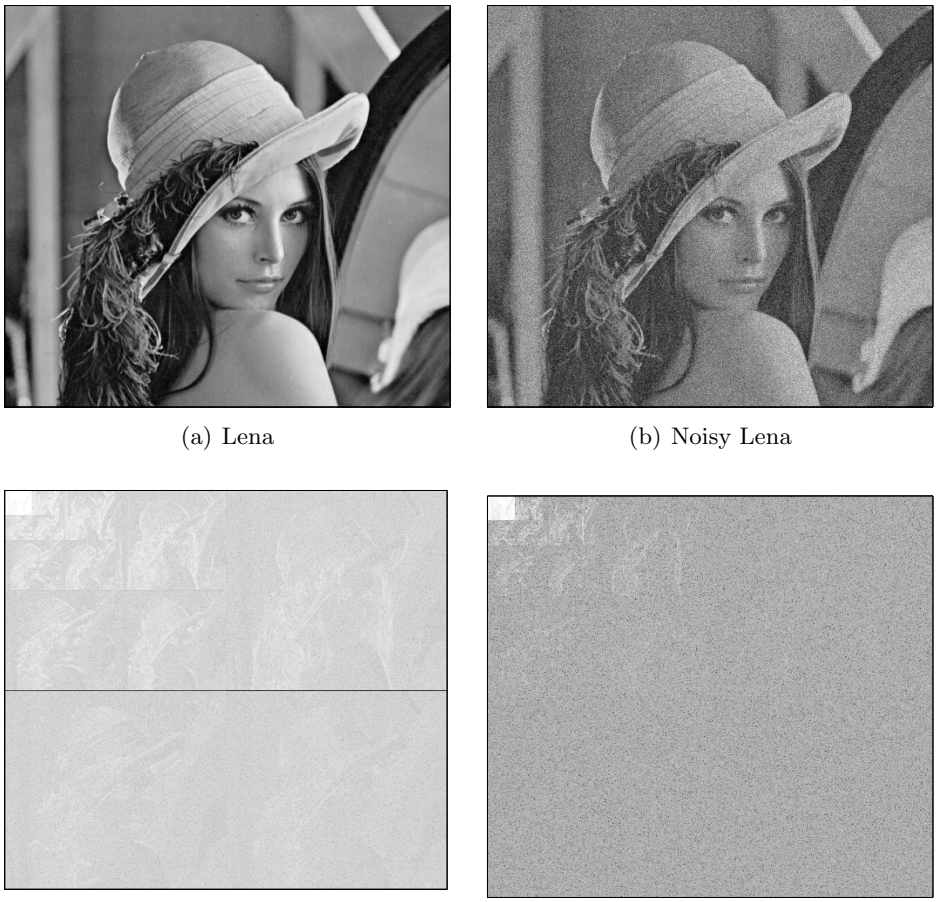
The results are gathered in table 4.1. Finally, we mention that we might try to solve this problem as (P1-σ), i.e. with ℓ_1 minimization. This approach is called *Soft Thresholding*.

4.1.5 Inpainting

What about cases where some information is perfectly kept, but some parts of the signal is missing completely? The process of filling in missing data in a signal is called signal *inpainting*. Like with denoising, there are many ways to approach this problem, some involving sparse representations, and some not. For a more complete review of methods, see [42].

A simple example of an inpainting process in shown in figure 4.11. Here we consider the cage to be "corrupted" data, and we would like to paint in the missing parts of the signal.

In a way, we already considered this problem for the most part in our introductory Fourier



(a) Lena (b) Noisy Lena

(c) Lena DWT coefficients (d) Noisy DWT coefficients

Figure 4.8: The Lena image and the noise polluted version.

example. The idea is that if we can restore the signal in the Fourier basis, then the IDFT will provide the reconstructed of *inpainted* signal. Formally, we let $\mathbf{x} \in \mathbb{R}^N$ be some signal, and let $\mathbf{x}_{us} \in \mathbb{R}^M$ be the subsampled version,

$$\mathbf{x}_{us} = D\mathbf{x}, \tag{4.22}$$

where D is the a downsampling operator, an $M \times N$ matrix created from removing rows from the $N \times N$ identity matrix.

The inpainting problem can can be formulated as the task of finding a "good" $\mathbf{y} \in \mathbb{R}^L$ such that

$$D\mathbf{\Psi}\mathbf{y} = \mathbf{x}_{us}, \tag{4.23}$$

where $\mathbf{\Psi}$ is an $N \times L$ matrix such that the rows of $\mathbf{\Psi}$ are atoms in a dictionary (possibly basis elements in a basis). In the example introduced earlier, $\mathbf{\Psi}$ was the IDFT matrix, $F_N^\dagger$, which means the rows of $\mathbf{\Psi}$ are the complex exponentials $\{e^{2\pi ikl/N}\}$, while D was a random subsampling operator. The term "good" here means a $\mathbf{y}$ such that $\mathbf{\Psi}\mathbf{y}$ is as close to $\mathbf{x}$ as possible. If we know that the signal should be sparse in the $\mathbf{\Psi}$ basis, then we may approach this problem as (P1) or one of its variants, with the matrix $A = D\mathbf{\Psi}$.

(a) Lena restored (b) Restoration error

Figure 4.9: The denoised image and the error

(a) Lena restored (b) Restoration error

Figure 4.10: The denoised image and the error for the DCUT2-based denoising.

MCALab - Reproducible Research in Signal and Image Decomposition and Inpainting

A notably user friendly MATLAB toolkit for solving such inpainting problems has been developed by M.J. Fadili et al, called MCALab [43]. MCALab has support for DCUT2 representations as well as a structure that is easily modified to add additional transforms.

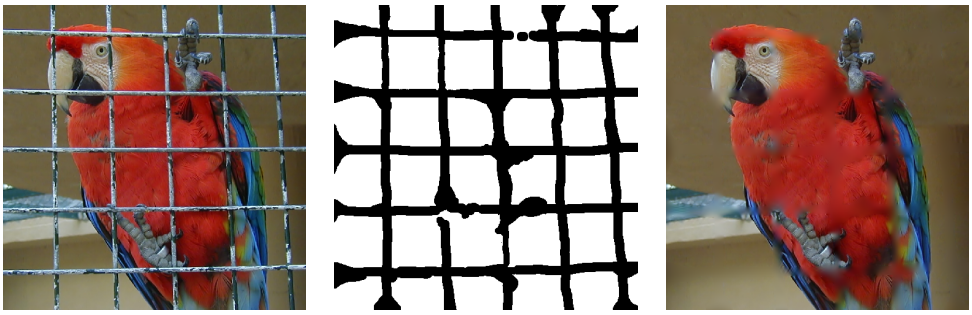
It is worth noting that MCALab does not solve (P1), but instead two related problems. The first one is

$$\text{minimize}_{y_1,\ldots,y_L,\sigma} \quad \lambda f(\mathbf{y}) + \|\mathbf{x}_{us} - D\mathbf{\Psi}\mathbf{y}\|_2^2/(2\sigma^2), \tag{4.24}$$

which is solved with the function with the function `EM_Inpaint(...)`. Here $f(\mathbf{y})$ can be any (convex) function. Setting $f(\mathbf{y}) = \|\mathbf{y}\|_1$ gives a problem formulation equivalent to (P1-LL).

The second function, `MCA2_Br(...)`, solves the problem

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{y}\|_1, \\ \text{subject to} \quad & \|D\mathbf{\Psi}\mathbf{y} - \mathbf{x}_{us}\|_2 \leq \sigma, \end{aligned} \tag{4.25}$$



(a) Caged parrot (b) Subsampling operator (c) Freed parrot

Figure 4.11: Inpainting of a cage in a parrot image.

which is simply (P1-σ). The names of these functions are derived from the algorithms used to solve the minimization problems; the problem we have so far chosen to overlook.

MCALab is available for download on their website: https://fadili.users.greyc.fr/demos/WaveRestore/downloads/mcalab/Home.html.

Numerical experiments with MCALab

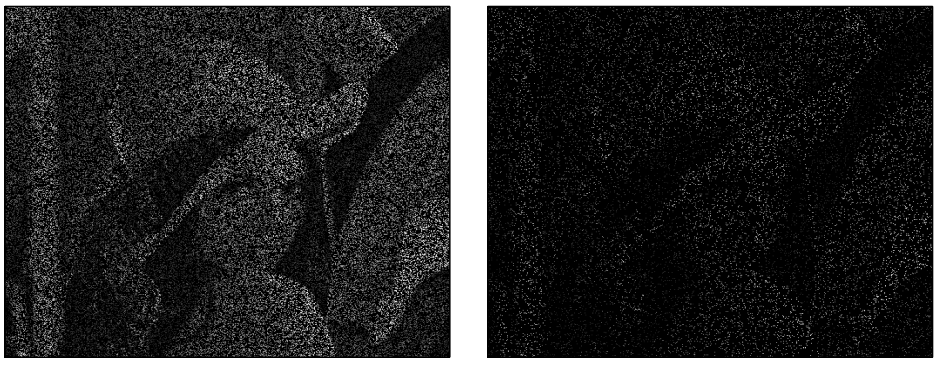
In the introductory example, we tried to restore the Fourier spectrum of a signal based on partial samples. We have also noted that if we are able to find the correct Fourier spectrum, we can simply apply an IDFT to obtain the restored signal. Here we will attempt this task using the various transforms we have introduced so far, the DFT, the Haar DWT, the CDF 9/7 DWT and the DCUT2. In each case, we restore the signal in the relevant basis, and then apply the inverse transform to restore the image in the spatial basis.

We will once again use the Lena image for our tests. We will look in particular at the cases where we subsample the image randomly by factors of 1/4 and 1/16 (as exemplified in Figure 4.12), and then restore the image using MCALab. The result of these simulations are shown in Table 4.2, as well as Figures 4.13 and 4.14.

There are several interesting things to note about these results. Even though the Lena image is, as we saw in section 3.5, more approximately sparse in the CDF 9/7 basis than the DFT basis (in the sense that that it is compressible with a smaller s, as defined in Definition 3.5.4), the restoration is far worse for the CDF 9/7 transform, both qualitatively and in terms of the PSNR. Clearly, sparsity is not the only feature which determines the quality of the restored signal. We also note that the DCUT2 seems to come out as the winner amongst these methods. Finally, even though the Haar DWT performs the worst for $M = N/4$, it catches up to, and passes, the CDF 9/7 transform for $M = N/16$. We will not attempt to explain these phenomena presently, but we note them as we move on to the next section, where we will begin to study the theory necessary to understand these results.

Number of samples	Transform	PSNR
$N/4$	zero-filling	6.56
	DFT	27.27
	Haar	22.91
	CDF 9/7	24.27
	DCUT2	31.37
$N/16$	zero-filling	5.97
	DFT	22.27
	Haar	19.19
	CDF 9/7	18.82
	DCUT2	25.31

Table 4.2: The table shows the results of inpainting the Lena image, solved as a LIP with (P1-σ) with different representations.



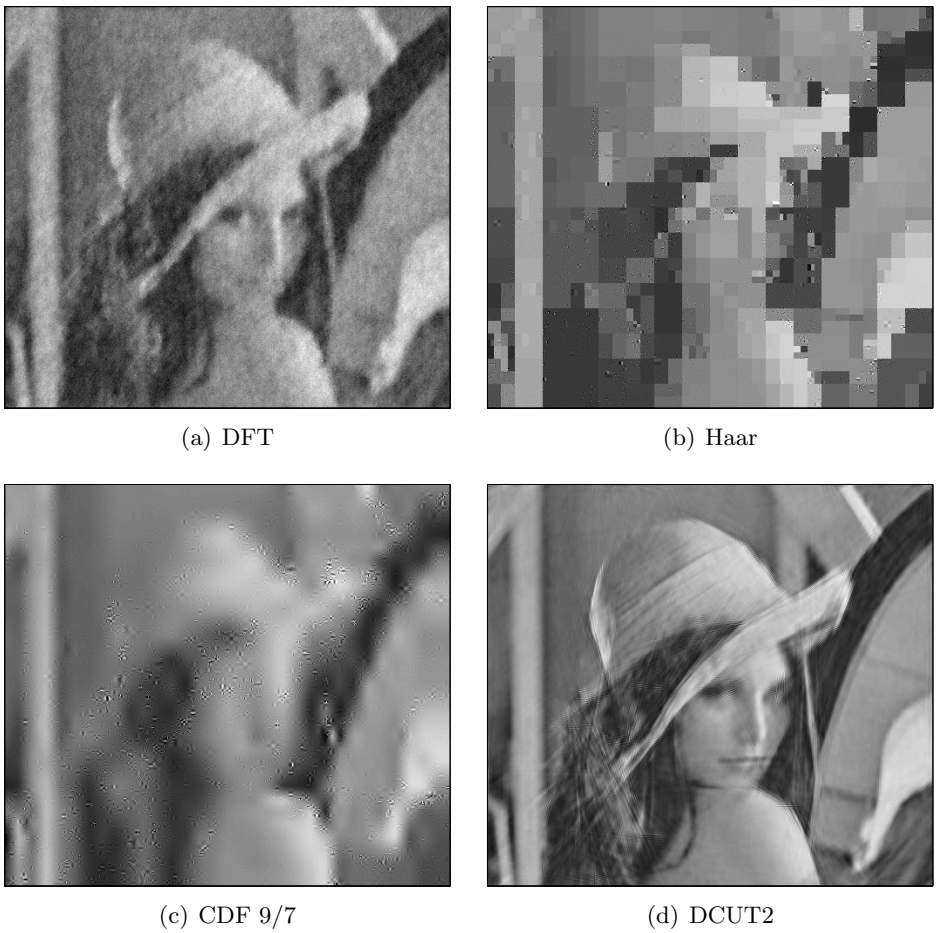
(a) $M = N/4$ (b) $M = N/16$

Figure 4.12: Subsamples of a $N = 512 \times 512$ Lena image, with $M = N/4$ and $M = N/16$ randomly selected pixels kept.

(a) DFT (b) Haar

(c) CDF 9/7 (d) DCUT2

Figure 4.13: Inpainting of the Lena image from $M = N/4$ randomly selected pixels.



(a) DFT (b) Haar

(c) CDF 9/7 (d) DCUT2

Figure 4.14: Inpainting of the Lena image from $M = N/16$ randomly selected pixels.

4.1.6 Atomic Decomposition

So far we have considered signal restoration methods. Here we will briefly introduce a signal analysis method based on overcomplete representations. Consider a signal which consists of a single sine as well as a few large singularity. To make this more concrete, consider the signal $\mathbf{x}$ defined by

$$x_i = \sin\left(\frac{2\pi 10 \times i}{1000}\right) + \delta_{i,100} + \delta_{i,300} + \delta_{i,470}, \quad i = 0, 1, \ldots, 999, \tag{4.26}$$

where $\delta_{i,j}$ is the Kronecker-delta function, defined by

$$\delta_{i,j} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{4.27}$$

This signal in shown in figure 4.15(a). The singularities creates a lot of noise in the DFT spectrum. In order to analyze the signal it would be better if we had some frame which includes both pure tones and singularities. This is the idea behind atomic decomposition: Let $\mathbf{\Psi}_1$ and $\mathbf{\Psi}_2$ be two bases. Now let $\mathbf{\Psi} = \mathbf{\Psi}_1 \cup \mathbf{\Psi}_2$ be the dictionary created by using all basis elements in $\mathbf{\Psi}_1$ and $\mathbf{\Psi}_2$. Even though $\mathbf{x}$ might not be sparse in the basis of delta functions or in the Fourier basis, it might have a sparse representation in the dictionary defined by combining them. This approach is called it atomic decomposition, as we separate the information in $\mathbf{x}$ into two separate domains. The result of such a decomposition into Diracs and Fourier functions is shown in figure 4.15(b). A more involved 2-D image separation performed in MCALab is shown in figure 4.16. This test was done by J.-L. Starck and can be viewed at http://jstarck.free.fr/mca.html.

We note the following result, shown in [44], which tells us that we can predictably find such sparse representations by ℓ_1-minimization.

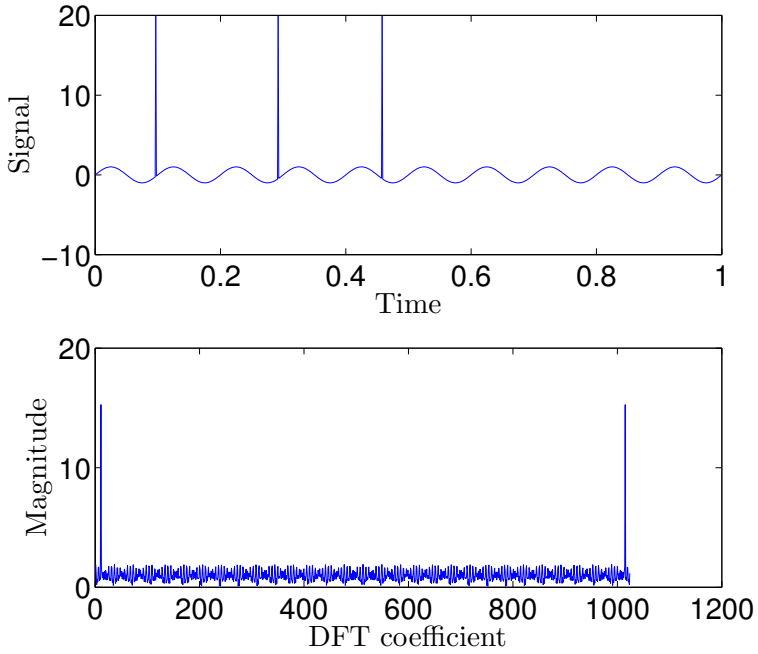
Theorem 4.1.6. Let $\mathbf{x}$ be a signal of length N. Let $\mathbf{\Psi}_1$ be a Fourier basis, let $\mathbf{\Psi}_2$ be a Dirac basis and let $\mathbf{\Psi} = \mathbf{\Psi}_1 \cup \mathbf{\Psi}_2$ be the combined dictionary, with elements $(\mathbf{\Psi})_n$. Assume that there is a $\mathbf{y}$ of length $2N$ such that

$$\mathbf{x} = \mathbf{\Psi}\mathbf{y}, \tag{4.28}$$

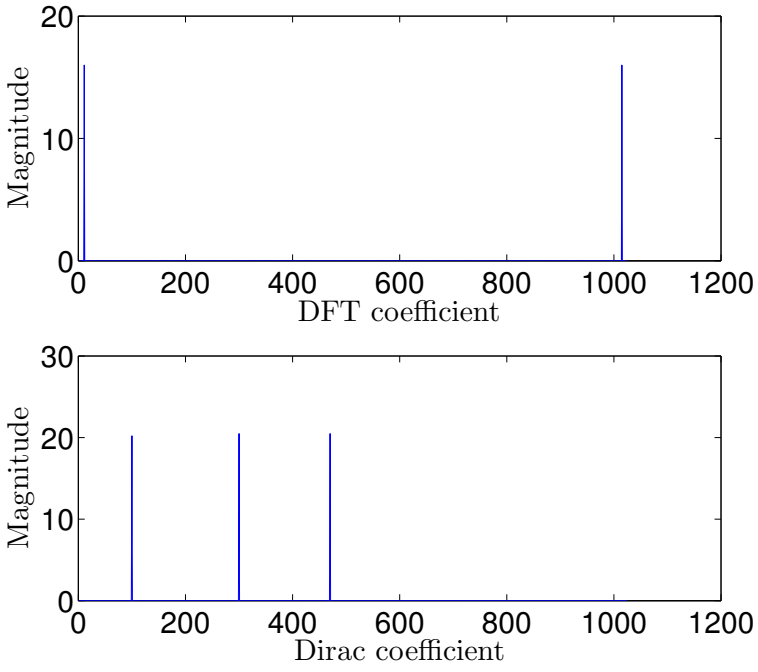
and $\mathbf{y}$ is $\sqrt{N}/2$-sparse. Then the problem

$$\begin{aligned} &\text{minimize} && \|\mathbf{z}\|_1, \\ &\text{subject to} && \mathbf{\Psi}\mathbf{z} = \mathbf{x}, \end{aligned} \tag{4.29}$$

has a unique solution, and it is given by $\mathbf{z} = \mathbf{y}$.

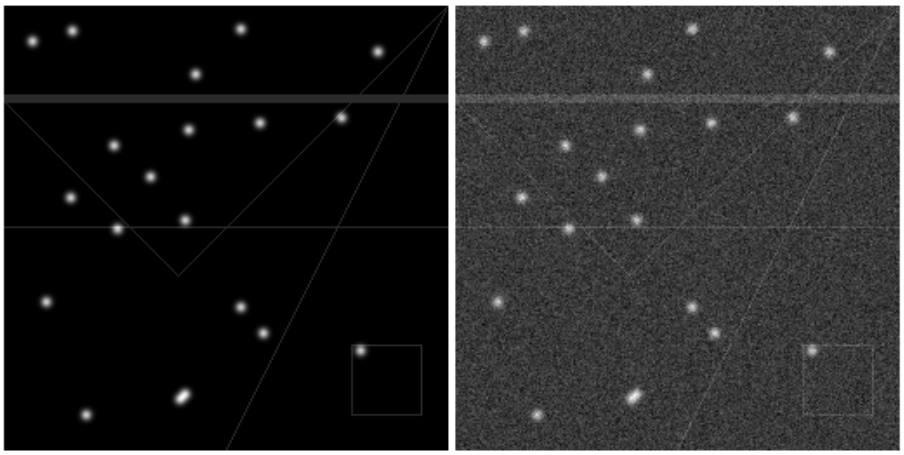


(a)

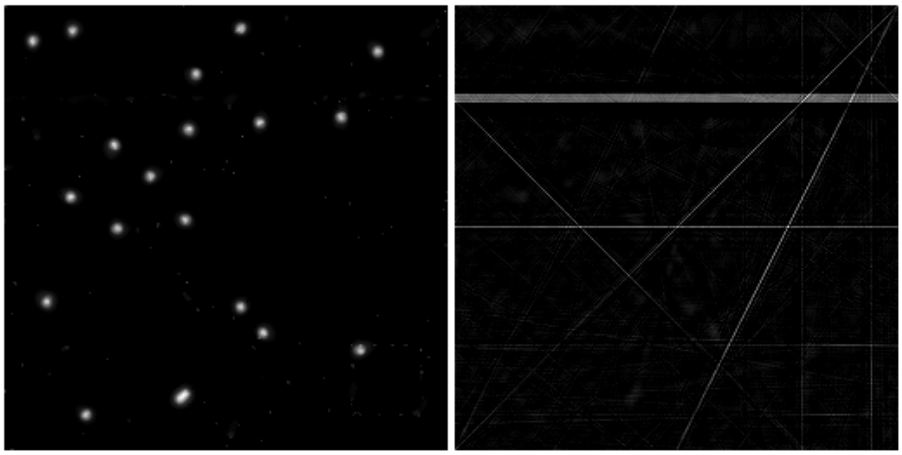


(b)

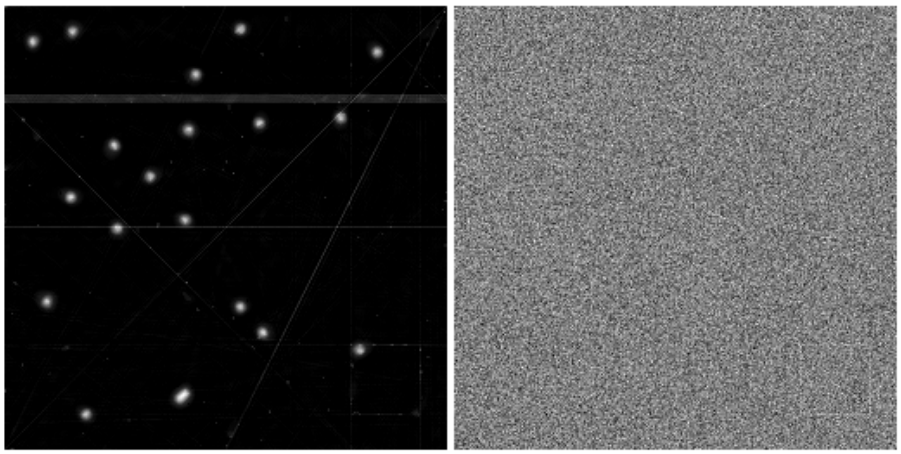
Figure 4.15: The figures show 4.15(a) a signal with a sine and some singularities, along with its DFT, and 4.15(b) its sparse decomposition into the Fourier and Dirac bases.



(a) An image of lines and Gaussians and a polluted version of the same image



(b) Decomposition by curvelets and wavelets of the noisy image



(c) The noise-free sum of the decompositions and the residual compared to the noisy image

Figure 4.16: The figures illustrate the image separation and denoising of a signal with lines and Gaussian functions.

4.1.7 Compressed Sensing

In the inpainting problem, we were limited to M samples of an N-length signal $\mathbf{x}$. These samples were selected by some sub-sampling operator D. Consider now a similar situation. You can still make M measurements on the system, but you can make these measurements in any way you want. Formally, we obtain a vector of measurements $\mathbf{z} \in \mathbb{R}^M$ as

$$\mathbf{z} = \mathbf{\Phi}\mathbf{x}, \tag{4.30}$$

Then, in a similar vein as the inpainting problem, we assume that $\mathbf{x}$ has a sparse representation in some dictionary $\mathbf{\Psi}$, i.e.

$$\mathbf{x} = \mathbf{\Psi}\mathbf{y}, \tag{4.31}$$

where $\mathbf{y}$ is a sparse vector. There are two possible ways to approach to this problem. The first is called the *Synthesis solution*, and involves simply substituting $\mathbf{x} = \mathbf{\Psi}\mathbf{y}$ into the minimization problem. In this case, we solve the problem

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{y}\|_1, \\ \text{subject to} \quad & \mathbf{\Phi}\mathbf{\Psi}\mathbf{y} = \mathbf{z}. \end{aligned} \tag{4.32}$$

The other approach is called the *Analysis solution*. In this case, we solve the problem

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{\Psi}^\dagger\mathbf{x}\|_1, \\ \text{subject to} \quad & \mathbf{\Phi}\mathbf{x} = \mathbf{z}. \end{aligned} \tag{4.33}$$

Note that in the case where $\mathbf{\Psi}$ is an orthobasis, these problems are equivalent, but this is not true for general dictionaries. Some work has been made in comparing the two approaches [45], but one does not appear to be universally superior to the other. Throughout this thesis, we will use the synthesis restoration approach, due to the fact that the ℓ_1-minimizing software used here (see section 4.12.2) in general only handles this approach. Note however that in the case where $\mathbf{\Psi}$ is invertible, but not orthogonal, we may rewrite the analysis problem by defining

$$\mathbf{y} = \mathbf{\Psi}^\dagger\mathbf{x}. \tag{4.34}$$

We can then solve the problem as

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{y}\|_1 \\ \text{subject to} \quad & \mathbf{\Phi}(\mathbf{\Psi}^\dagger)^{-1}\mathbf{y} = \mathbf{z} \end{aligned} \tag{4.35}$$

This result is useful for non-orthogonal bases, such as the CDF 9/7 wavelet, but it is not valid for the frames we have investigated, such as the DCUT2 or the Wave Atom, as they do not have explicit inverses.

This is the formal description of the CS problem, and is the problem we will focus on for the rest of this chapter. As such, we should properly introduce its players.

Definition 4.1.7 (Sensing matrix)**.** The *sensing matrix* or *measurement matrix* $\mathbf{\Phi}$ describes the measurements performed on a system. These measurements are all some linear combination of the N-length signal vector $\mathbf{x}$. $\mathbf{\Phi}$ is an $M \times N$ matrix, where $M < N$, such that $\mathbf{z} = \mathbf{\Phi}\mathbf{x}$ is an M-length *measurement vector*. We denote the individual measurement (the rows of $\mathbf{\Phi}$) by ϕ_j, $j = 1, \dots M$.

From these measurement in $\mathbf{\Phi}$, we now want to try to restore the signal. To do this, we choose a *representation dictionary*.

Definition 4.1.8 (Representation dictionary and matrix)**.** The representation dictionary $\{\psi_k\}_{k=1}^L$ is the dictionary in which we will try to restore the signal. Ideally, the signal should be sparse in this dictionary. $\mathbf{\Psi}$ refers to the $N \times L$ "change of basis matrix" from the position basis to the representation basis, in the sense that

$$\mathbf{x} = \mathbf{\Psi}\mathbf{y}. \tag{4.36}$$

Note however that the atoms in the dictionary do not necessarily constitute a basis. It might be a frame. It may also have rank lower than N, but some care should be taken in this case, as then $\mathbf{x}$ might be impossible to restore.

The CS problem in the synthesis form is really just another formulation of (P1). If we collect the product $\mathbf{\Phi\Psi}$ into a single $L \times M$ matrix, we recover the familiar formulation. As such, we may consider the matrix $\mathbf{\Phi\Psi}$ to be the sensing matrix, and solve this problem in the $\mathbf{\Psi}$-basis.

Definition 4.1.9. We define

$$\mathbf{\Phi}_{tot} = \mathbf{\Phi\Psi}, \tag{4.37}$$

to be the *total sensing matrix*.

We can then simplify the problem by trying to restore the sparsest possible $\mathbf{y}$ such that $\mathbf{\Phi}_{tot}\mathbf{y} = \mathbf{z}$. This view will be useful now, when we will investigate some theoretical aspects of CS.

4.2 Overview of the rest of this chapter

Let us quickly recap what we know so far. We have seen that (P0) has a unique solution in some circumstances (Lemma 4.1.3). We have also argued that solving (P1) will "in a lot of cases" provide the same solution as (P0). In the rest of this chapter, we will want to investigate the following.

1. What properties of $\mathbf{\Phi}$ guarantee that (P1) will exactly reproduce the measured signal $\mathbf{x}$?
2. Are we able to construct such matrices in a predictable way?
3. What kind of guarantees can be established for noisy signals which are not exactly sparse? What about compressible signals?
4. What considerations must be made when extending our results from 1-D to 2-D?

The known theory of CS is spread across many articles, and no known dogma-defining textbook has emerged as of yet. By keeping this path in mind, the goal is to present an as clear and consistent version of the CS-theory as possible. The theory here is collected from [46], [47] and [48], along with several articles which will be cited when relevant.

4.3 Warm up: Mutually incoherent bases

In this section we present one of the first results for recovery guarantees with compressed sensing. This result somewhat skips to the ending of our theoretical investigations, however, it is useful to present it early. It will be put in a wider setting in section 4.7.3.

We saw earlier that a reconstruction of a picture using a DWT basis gave surprisingly poor results. In order to explain this we will introduce the concept of incoherence. Coherence in the context of CS can mean a few different things. First we will use the version suggested in first in [44] and studied in [49].

Definition 4.3.1 (Coherence of elements)**.** Let U be an $N \times N$ matrix where the rows have unit ℓ_2 norm. The *coherence of elements* of U, $M(U)$, is defined as

$$M(U) = \sqrt{N} \max_{k,j} |U_{kj}|. \tag{4.38}$$

The parameter $M(U)$ can be interpreted as a "worst case" measure of how concentrated the rows of U are. Since each row of U has ℓ_2 norm equal to 1, $M(U)$ will take a value between 1 and $\sqrt{N}$. If U is simply single measurements in position space $\mu(U) = \sqrt{N}$, while if U consists of Fourier measurements, $M(U) = 1$. For any m-level Haar DWT, the coherence is $\sqrt{N/2}$.

We can relate the coherence of a total measurement matrix to the relationship between the sensing matrix and the representation matrix.

Definition 4.3.2 (mutual coherence property)**.** Let $\mathbf{\Phi}$ be an $N \times N$ orthogonal sensing matrix, let ϕ_k be the N-length measurement vectors of $\mathbf{\Phi}$. Let $\mathbf{\Psi}$ be an $N \times N$ orthogonal representation matrix, and let ψ_k be the rows making up $\mathbf{\Psi}$. Then we define the *mutual coherence* of $\mathbf{\Phi}$ and $\mathbf{\Psi}$, $M(\mathbf{\Phi}, \mathbf{\Psi})$ as

$$M(\mathbf{\Phi}, \mathbf{\Psi}) = \sqrt{N} \max_{1 \leq k,j \leq N} |\langle \psi_k, \phi_j \rangle| \tag{4.39}$$

Loosely speaking, if a pair of bases $\mathbf{\Phi}$ and $\mathbf{\Psi}$ have a low (i.e. not dependent on N) coherence are said to have the Mutual Incoherence Property (MIP).

Note that if $\mathbf{\Phi}_{tot} = \mathbf{\Phi}\mathbf{\Psi}$, then $M(\mathbf{\Phi}, \mathbf{\Psi}) = M(\mathbf{\Phi}_{tot})$. The mutual coherence of $\mathbf{\Phi}$ and $\mathbf{\Psi}$ is directly related to the performance quality of CS, as the following theorem shows.

Theorem 4.3.3 (Incoherence of bases)**.** Let $\mathbf{\Phi}^+$ be an $N \times N$ sensing matrix, and let $\mathbf{\Phi}$ be the $M \times N$ matrix constructed from drawing M rows from $\mathbf{\Phi}^+$ uniformly at random. We want to restore the signal $\mathbf{x}$ from the measurements $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$. If $\mathbf{x}$ is k-sparse when represented in a basis $\mathbf{\Psi}$, then if

$$M \geq C M^2(\mathbf{\Phi}^+, \mathbf{\Psi}) k \log(N) \log(\epsilon^{-1}), \tag{4.40}$$

the probability of achieving an exact reconstruction of $\mathbf{x}$ exceeds $1 - \delta$.

This is the main result of [49], and one of the most important early results of CS. We are now able to understand the somewhat disappointing results we obtained when attempting to

restore the Lena image with a DWT basis. Because the position basis and the DWT basis are quite coherent, we would need a much larger number of measurements to obtain a good reconstruction.

The formulation of this result is somewhat strange. We begin by creating an $N \times N$ sensing matrix, only to reduce it to an $M \times N$ matrix for the actual sensing. The result is also probabilistic in nature, which is not optimal. Let us briefly look at why we cannot provide an absolute guarantee by this random subsampling, and why it is still preferred.

Subsampling schemes and aliasing-effects

We have mandated a random subsampling of signals, rather than a uniform one. Let us now address this point. We know that if we subsample a sound signal uniformly, we will experience problems with aliasing. Let us consider this idea in some more detail. Assume we have a signal $\mathbf{x}$, which is sampled with a frequency of $N_f = 1200$ Hz. The signal is a pure sine signal of 440 Hz. The elements of $\mathbf{x}$ are then

$$x_j = \sin(2\pi 440(j/1200)), \quad j = 0, \ldots, 1199. \tag{4.41}$$

This vector can we written as $x_j = (e^{2\pi 440 i(j/1200)} - e^{2\pi i 760(j/1200}})/(2i)$. Which means the DFT of $\mathbf{x}$ is the vector $\mathbf{y}$ where $y_{440} = -y_{560} = \frac{1}{2i\sqrt{1200}}$ and the rest of $\mathbf{y}$ is zero.

Now, if we first imagine every second sample is stored in a vector $\mathbf{z}$, where

$$z_i = x_{2i} = \sin(2\pi 440(2i/1200)), \quad i = 1, \ldots 599. \tag{4.42}$$

If we perform a CS restoration in a Fourier basis, the same vector $\mathbf{y}$ would still be a solution to the ℓ_0 norm minimization. However, as we saw in example 2.1.3, the vector $\mathbf{x}^*$ with elements $x_i^* = -\sin(2\pi 160(i/1200)), \quad i = 0, \ldots, 599$ would yield the exact same $\mathbf{z}$. The full length (1200 point) DFT of $\mathbf{x}^*$, however, would be $\mathbf{y}^*$, where $\frac{y^*_{160}=-y^*_{1040}=1}{2i\sqrt{1200}}$, and the rest is zero. Which means that the ℓ_0-norm would be just as minimized for this solution. To make matters worse, since we only minimize the ℓ_1 norm, any solution of the form

$$\mathbf{y}_{rec} = a\mathbf{y} + (1-a)\mathbf{y}^* \tag{4.43}$$

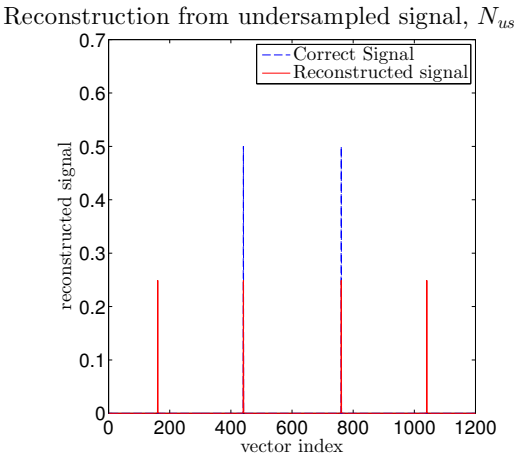
where $0 \leq a \leq 1$ would give the exact same ℓ_1 norm, and thus none of these would be preferred as the CS solution above the others. The result of an attempted restoration is shown in figure 4.17(a), where the minimizing Matlab routine has chosen the solution corresponding to $a = 1/2$.

If we kept every third sample, then we can in a similar fashion work out that while an acceptable result is $\mathbf{y}$, another just as acceptable result is the DFT of the signal $\mathbf{x}^{**}$ where $x_i^{**} = -\sin(2\pi(-40)(i/1200)) = \sin(2\pi(40)(i/1200)), \quad i = 0, \ldots, 399$. The DFT of this vector we will denote $\mathbf{y}^{**}$. Another possible solution in this case is $\mathbf{x}^{***}$, where $x_i^{***} = -\sin(2\pi 360(i/1000))$, with a DFT denoted $\mathbf{y}^{***}$ A CS restoration will then return

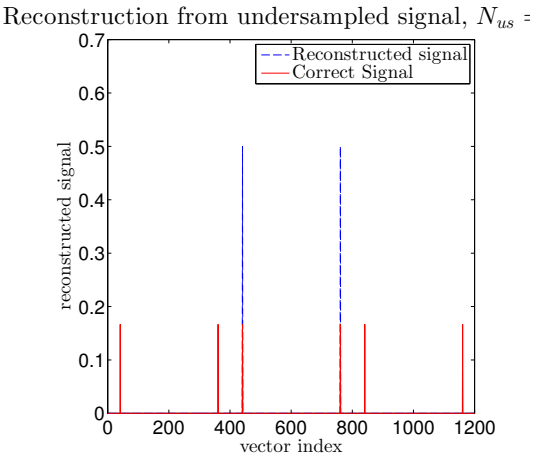
$$\mathbf{y}_{rec} = b\mathbf{y} + c\mathbf{y}^{**} + (1-b-c)\mathbf{y}^{***} \tag{4.44}$$

with suitable constraints for b and c. This problem will get progressively worse as we try to increase the sub-sampling distance, with more and more pure tones fitting the signal. The result of an attempted restoration is shown in figure 4.17(b).

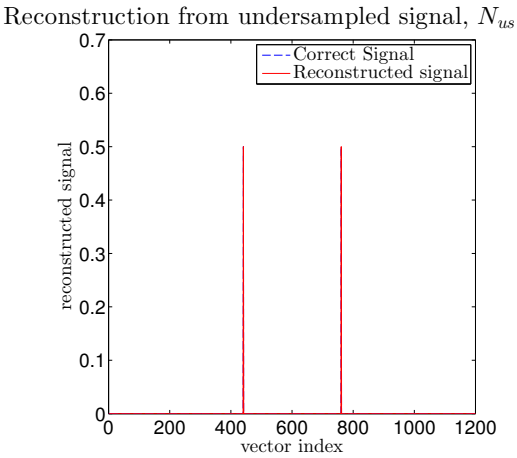
Now let us consider what happens if we alternate sampling every 2nd and third sample. We let $\mathbf{z} = (x_0, x_2, x_5, x_7, x_{10}, \ldots, x_{997})^T$. Now we might expect the situation to improve somewhat.



(a) Every 2nd sample kept



(b) Every 3rd sample kept



(c) 2nd and 3rd sample alternating

Figure 4.17: Attempted CS restoration of a sparse Fourier spectrum.

Now neither $\mathbf{y}^*, \mathbf{y}^{**}$ or $\mathbf{y}^{***}$ are possible solutions. The result of this restoration is shown in figure 4.17(c)

The trend here is that if the sampling is equidistant, the aliasing effects that may occur between each subsampled point will build itself up, which will lead to poor performance. We will follow the terminology of [2] which first discussed this in some detail, and refer to this as *coherence of undersampling artifacts*. Similar effects will occur for other transforms as well, but we will not go into detail on this here. For good performance of CS, we will require that the undersampling artifacts must be incoherent (noise-like) in the sparse domain (the representation basis). The best way to ensure that these artifacts are incoherent is to sample the signal randomly.

4.3.1 Known MIP-pairs

We have already noted that the Fourier basis and the position basis are maximally incoherent. Here we will consider another pair of maximally inchoherent bases, Haar Wavelets and *noiselets* [50].

Definition 4.3.4 (Noiselets)**.** Noiselets are functions designed to be completely incompressible under the Haar transform. The family of noiselets are constructed on the interval $[0, 1)$ as follows:

$$f_1(x) = \chi_{[0,1)}(x) \tag{4.45}$$
$$f_{2n}(x) = (1-i)f_n(2x) + (1+i)f_n(2x-1) \tag{4.46}$$
$$f_{2n+1} = (1+i)f_n(2x) + (1-i)f_n(2x-1) \tag{4.47}$$

where $\chi_{[0,1)}(x) = 1$ for $x \in [0, 1)$ and 0 otherwise.

It can be shown that the set of functions $\{f_n\}_{n=0}^{2^N-1}$ is an ortonormal basis for the vector space V_{2^N}. We can then discretize the noiselets the same way we did wavelets. Figure 4.18 show some noiselet transform matrices.

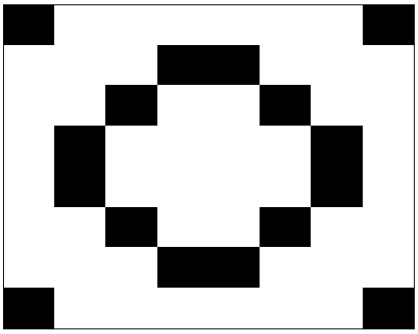
It can be shown that the Noiselets are maximally incoherent with any m-level Haar DWT [51]. Furthermore, the noiselets have fast (of order $\mathcal{O}(N\log(N))$) implementations, which makes them preferable to random measurements.

4.4 Null Space Property

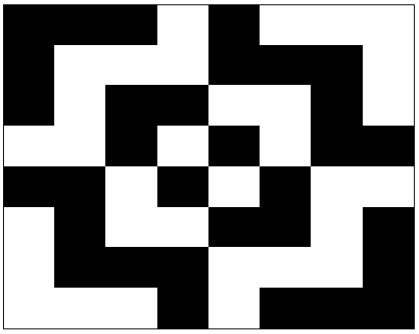
In this and the next few sections, we will develop some of the "modern" theory for when an exact reconstruction is possible (modern here is of course relative, as the MIP-result only dates back to 2006 in the form presented here). We will consider properties of the total sensing matrix $\mathbf{\Phi}_{tot}$. For convenience we will dub this matrix A in most results. We first look back to Lemma 4.1.3. Let us define the property that "A is an $m \times n$ matrix such that every set of $2K$ columns of A are linearly independent".

Definition 4.4.1 (Spark)**.** The *spark* of a matrix A is the smallest number of columns if A that are linearly dependent.

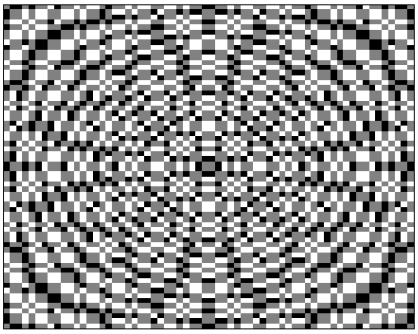
With this definition, Lemma 4.1.3 states that we can achieve exact reconstruction if $\text{spark}(A) > 2k$. This lemma gives a necessary condition for $\mathbf{x}$ to be recoverable by (P1). It is however not sufficient; there is no guarantee that the relaxed optimization will find the correct solution. We will call this property the Exact Recovery Property (ERP). Before we go further, one should refresh on the terminology and notation given in Definition 3.5.2, which will be used thoroughly in the remainder of the section. We also remind readers that the *null space* or *kernel* of A, $\ker(A)$ is the space of all vectors $\mathbf{x}$ such that $A\mathbf{x} = 0$.



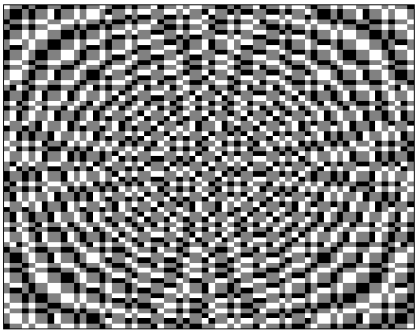
(a) Real part of 8×8 noiselet matrix



(b) Imaginary part of 8×8 noiselet matrix



(c) Real part of 64×64 noiselet matrix



(d) Real part of 64×64 noiselet matrix

Figure 4.18: Noislet bases.

Definition 4.4.2 (Exact Recovery Property)**.** Let $\Lambda \subseteq \{0, 1, \ldots, N-1\}$ be of cardinality $|\Lambda| = k$. An $M \times N$ total sensing matrix A is said to satisfy the ERP for Λ if for any $\mathbf{x}$ such that $\text{supp}(\mathbf{x}) = \Lambda$, $\mathbf{x}$ is the unique solution to (P1), i.e. $\mathbf{x}$ can be restored exactly from the measurements $A\mathbf{x} = \mathbf{y}$, by ℓ_1-minimization. If A satisfies the ERP for any Λ such that $|\Lambda| \leq k$, A is said to be k-ERP.

So how do we determine if A is k-ERP? One possible sufficient condition is the Null Space Property (NSP).

Definition 4.4.3 (Null Space Property)**.** Let $\Lambda \subseteq \{0, 1, \ldots, N-1\}$ be of cardinality $|\Lambda| = k$. An $M \times N$ sensing matrix A is said to satisfy the NSP for Λ if for every $\mathbf{x} \in \ker(A) \setminus \{0\}$ satisfies $\|\mathbf{x}_\Lambda\|_1 < \|\mathbf{x}_{\Lambda^c}\|_1$. If A satisfies the NSP for any Λ such that $|\Lambda| \leq k$, A is said to be k-NSP.

The null space property ensures that the elements of $\ker(A)$ are not too concentrated on a few indices. If A is k-NSP, and $\mathbf{x}$ is k-sparse, then there is a Λ of cardinality k such that

$\|\mathbf{x}_{\Lambda^c}\|_1 = 0$. The NSP condition then ensures that $\|\mathbf{x}_\Lambda\|_1 = 0$ as well, which means that if A is k-NSP, then the only k-sparse vector in $\ker(A)$ is the zero-vector. This is naturally important for a reconstruction to be possible, as if the measurement of k-sparse vector were to yield the zero vector, a reconstruction would be impossible. We should also note the following reformulations of the NSP:

- If we add $\|\mathbf{x}_\Lambda\|_1$ to both sides, then (as $\|\mathbf{x}\|_1 = \|\mathbf{x}_\Lambda\|_1 + \|\mathbf{x}\|_{\Lambda^c}$) the NSP reads
$$2\|\mathbf{x}_\Lambda\|_1 \leq \|\mathbf{x}\|_1. \tag{4.48}$$
- Let A be k-NSP. Letting Λ be the index set of the k largest elements of some vector $\mathbf{x}$ and adding $\|\mathbf{x}_{\Lambda^c}\|_1$ shows that
$$\|\mathbf{x}\|_1 \leq 2\sigma_k(\mathbf{x})_1, \tag{4.49}$$
where $\sigma_k(\mathbf{x})_1$ is the k-term approximation error introduced in Definition 3.5.3.

The following result shows the NSP is both necessary and sufficient for the ERP.

Theorem 4.4.4. *A satisfies the NSP for Λ, if and only if A satisfies the ERP for Λ.*

Proof. This proof is from [52].

We use contrapositive arguments. First, assume that A does not satisfy the the NSP for Λ. Then there exists $\mathbf{z} \in \ker(A) \setminus \{0\}$ such that $\|\mathbf{z}_\Lambda\|_1 > \|\mathbf{z}_{\Lambda^c}\|_1$. If we set $\mathbf{x}_1 = \mathbf{z}_\Lambda$ and $\mathbf{x}_2 = -\mathbf{z}_{\Lambda^c}$, then $A\mathbf{x}_1 - A\mathbf{x}_2 = A\mathbf{z} = 0$, which implies $A\mathbf{x}_1 = A\mathbf{x}_2$. Since $\|\mathbf{x}_2\|_1 < \|\mathbf{x}_1\|_1$, ℓ_1 minimization will not be able to restore $\mathbf{x}_1$ (such an approach will at the very least prefer $\mathbf{x}_2$ as the solution), and thus A is not ERP for Λ.

Now assume that A is not ERP for Λ. Then there is at least one one $\mathbf{x}^1$ with support on Λ which is not recovered by (P1). This in turns means that there is a $\mathbf{x}^2$ such that $A\mathbf{x}^1 = A\mathbf{x}^2$ and $\|\mathbf{x}^1\|_1 > \|\mathbf{x}^2\|_1$. If we set $\mathbf{z} = \mathbf{x}^1 - \mathbf{x}^2$, then $A\mathbf{z} = 0$. As such, $\mathbf{z} \in \ker(A) \setminus \{0\}$. From the triangle inequality, it follows that

$$\begin{aligned}
\|\mathbf{z}_{\Lambda^c}\|_1 &= \|(\mathbf{x}^2)_{\Lambda^c}\|_1 && (4.50)\\
&= \|\mathbf{x}^2_{\Lambda^c}\|_1 + \|\mathbf{x}^2_\Lambda - \mathbf{x}^2_\Lambda + \mathbf{x}\|_1 + \|\mathbf{x}\|_1 && (4.51)\\
&\leq \|\mathbf{x}^2_{\Lambda^c}\|_1 + \|\mathbf{x}^2_\Lambda - \mathbf{x}^2_\Lambda\|_1 + \|\mathbf{x}\|_1 + \|\mathbf{x}\|_1 && (4.52)\\
&= \|\mathbf{x}^2\|_1 + \|\mathbf{z}_\Lambda\|_1\|\mathbf{x}^1\|_1 && (4.53)\\
&\leq \|\mathbf{z}_\Lambda\|_1, && (4.54)
\end{aligned}$$

which shows that A is not NSP for Λ □

It follows directly from this that A is k-ERP if and only if A is k-NSP.

We can note a few more properties of the NSP. The NSP is a necessary and sufficient requirement on A to exactly recover any $\mathbf{x}$ from $\mathbf{y} = A\mathbf{x}$ through (P1). It is also necessary and sufficient to restore $\mathbf{x}$ through (P0). Let $\mathbf{z}$ be the solution of (P0) for some k-NSP A. Then $\|\mathbf{z}\|_0 \leq \|\mathbf{x}\|_0$, i.e. $\mathbf{z}$ is at least k-sparse. However, since every k-sparse vector is uniquely restored by A, it follows that $\mathbf{z} = \mathbf{x}$.

If the measurements are stretched, rotated or shifted, the NSP is still satisfied, as

$$\ker(A) = \ker(BA), \tag{4.55}$$

where B is an invertible $M \times M$ matrix. Furthermore, if we add more measurements, i.e. let B be an $M' \times N$ matrix such that $M' > M$ and the first M rows of B are equal to A, then $\ker(B) \subset \ker(A)$.

4.4.1 An alternate definition of the null space property

As we noted at the beginning of this chapter, the theory of CS is still in its adolescent stage. As such, there are some competing formulations of results. Here we highlight one of these inconsistencies, by noting an alternative definition of the NSP. We shall denote this as the NSP-2 for clarity, but in the literature it is simply referred to as the NSP.

Definition 4.4.5 (NSP-2)**.** A is said to be k-NSP-2 if there is a constant $C > 0$ such that

$$\|\mathbf{x}_\Lambda\|_2 \leq C \frac{\|\mathbf{x}_{\Lambda^c}\|_1}{\sqrt{k}}, \tag{4.56}$$

for all $\mathbf{x} \in \ker(A)$ and all Λ such that $|\Lambda| \leq k$.

The connection between the NSP and the NSP-2 can be made with the help of the following small result:

Lemma 4.4.6. For any k-sparse vector $\mathbf{x} \in \mathbb{C}^N$,

$$\|\mathbf{x}\|_1 \leq \sqrt{k}\|\mathbf{x}\|_2 \leq k\|\mathbf{x}\|_\infty. \tag{4.57}$$

Proof. For a vector $\mathbf{x}$, let $\mathbf{y} = \operatorname{sign}(\mathbf{x})$ be the vector defined by

$$y_i = \begin{cases} 1 & \text{if } x > 0, \\ -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0. \end{cases} \tag{4.58}$$

With this definition, we have that $\|\mathbf{x}\|_1 = |\langle \mathbf{x}, \operatorname{sign}(\mathbf{x})\rangle|$, and for a k-sparse $\mathbf{x}$, $\|\operatorname{sign}(\mathbf{x})\|_2 = \sqrt{k}$. From the Cauchy-Schwartz inequality, it then follows that $\|\mathbf{x}\|_1 \leq \sqrt{k}\|\mathbf{x}\|_2$. The upper bound is obtained by noting that

$$\|\mathbf{x}\|_2 = \left(\sum_{i=0}^{N-1} x_i^2 \right)^{1/2}, \tag{4.59}$$

and if we replace all nonzero x_i with $\max_k\{x_k\} = \|\mathbf{x}\|_\infty$ we obtain

$$\|\mathbf{x}\|_2 \leq \left(\sum_{i=0}^{k-1} \|\mathbf{x}\|_\infty^2 \right)^{1/2} = \sqrt{k}\|\mathbf{x}\|_\infty. \tag{4.60}$$

□

We can now show the following.

Lemma 4.4.7. If A is k-NSP-2 with constant $C < 1$, then it is also k-NSP.

PROOF. Let Λ be an index set with $|\Lambda| = k$, and let $\mathbf{x} \in (A)$. Then $\mathbf{x}_\Lambda$ is k-sparse. We know that $\|\mathbf{x}_\Lambda\|_1 \leq \|\mathbf{x}_\Lambda\|_2\sqrt{k}$, and because A is NSP-2 with $C < 1$, we also have $\|\mathbf{x}_\Lambda\|_2 < \|\mathbf{x}_{\Lambda^c}\|_1/\sqrt{k}$, which in total gives us

$$\|\mathbf{x}_\Lambda\|_1 \leq \sqrt{k}\|\mathbf{x}_\Lambda\|_2 < \|\mathbf{x}_{\Lambda^c}\|_1. \tag{4.61}$$

□

We can also show a similar implication in the other direction.

Lemma 4.4.8. If A is k-NSP, then it is also k-NSP-2 with constant $C = \sqrt{k}$.

PROOF. Note that $\|\mathbf{x}\|_1 \geq \|\mathbf{x}\|_2$, as

$$\|\mathbf{x}\|_1^2 = \left(\sum_i |x_i|\right)^2 = \sum_i |x_i|^2 + \sum_{i \neq j} |x_i||x_j| \geq \sum_i |x_i|^2 = \|\mathbf{x}\|_2^2. \tag{4.62}$$

If A is k-NSP, then for any $\mathbf{x} \in \ker(A) \setminus \{\mathbf{0}\}$, then for any set Λ with $|\Lambda| = k$, we have

$$\|\mathbf{x}_\Lambda\|_2 \leq \|\mathbf{x}_\Lambda\|_1 < \|\mathbf{x}_{\Lambda^c}\|_1 = C\frac{\|\mathbf{x}_{\Lambda^c}\|_1}{\sqrt{k}}, \tag{4.63}$$

if $C = \sqrt{k}$. □

The NSP-2 is somewhat more general than the the NSP. Guarantees from the NSP-2 are usually of the form "Let $\Delta : \mathbb{C}^M \to \mathbb{C}^N$ be some specific recovery method. Then if A is k-NSP-2,

$$\|\Delta(A\mathbf{x}) - \mathbf{x}\|_2 \leq \frac{C\sigma_k(\mathbf{x})_1}{\sqrt{k}}, \tag{4.64}$$

for any $\mathbf{x}$".

4.5 Coherence of measurements

The NSP is both necessary and sufficient for perfect reconstruction. However, checking for which k a sensing matrix is NSP is difficult. Here we introduce a useful tool for considering the quality of a sensing matrix, the *coherence*.

Definition 4.5.1 (Coherence)**.** Let A be an $M \times N$ total sensing matrix, with ℓ_2-normalized columns $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_N$. The coherence of A, $\mu(A)$ is defined as

$$\mu = \max_{i \leq i \neq j \leq N} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle|. \tag{4.65}$$

When there is risk of confusion with the coherence of elements introduced in section 4.3, we will refer to this as the *coherence of measurements*.

It is possible to show that the coherence of measurements will always fall in the range $\left[\sqrt{\frac{N-M}{M(N-1)}}, 1\right]$ (we will do this in the proof of theorem 4.7.3). This is known as the Welch bound [53].

Consider the case where two columns $\mathbf{a}_j$ and $\mathbf{a}_k$ of A are exactly equal. In this scenario, there will be no information in the measurements to distinguish the values of the signal components x_j and x_k. This will also cause the maximal possible coherence of $\mu(A)$. We might be able to restore what the value of some linear combination of x_j and x_k is, but we will not be able to find x_j and x_k, and exact reconstruction is impossible. We can make the importance of the coherence more clear by the following theorems. First we introduce the notion of the *Gramian* of a matrix A.

Definition 4.5.2 (Gramian of a matrix)**.** Let A be an $M \times N$ matrix. The *Gram matrix* or Gramian G, of A is defined as the $N \times N$ matrix

$$G = A^\dagger A. \tag{4.66}$$

The coherence of measurements can be related to the Gramian of A. If we let G be the Gramian of A, then the following properties hold:

- $G_{ii} = 1$.
- $G_{ij} < \mu(A)$ for $i \neq j$.
- $\mu(A) = \max_{j \neq k} |G_{jk}|$.

This will be useful when in proving some theorems later. We also need one more result, which gives an easy to calculate bound on the eigenvalues of an $N \times N$ matrix A.

Theorem 4.5.3 (Gershgorin circle theorem [54])**.** For each eigenvalue λ of a $k \times k$ matrix A, there is an index $i \in \{1, 2, \ldots, k\}$ such that

$$|\lambda - A_{ii}| \leq \sum_{j \neq i} |A_{ij}|. \tag{4.67}$$

This can be interpreted geometrically as saying that the more diagonally dominant this matrix is, the closer the eigenvalues will lie to the diagonal values. This result will be useful later as well. Now we are ready to prove the following:

Theorem 4.5.4 (Incoherence of measurements and reconstruction)**.** For any $M \times N$ measurement matrix A,

$$\text{spark}(A) \geq 1 + \frac{1}{\mu(A)}. \tag{4.68}$$

Additionally, if

$$k < \frac{1}{2}\left(1 + \frac{1}{\mu(A)}\right), \tag{4.69}$$

then (P0) has a unique solution, and if

$$k < \left(1 + \frac{1}{3\mu(A)}\right), \tag{4.70}$$

then A is k-ERP.

PROOF. This proof is a slightly expanded version of the one found in [46] p. 24-25. Let $\Lambda \subseteq \mathbb{N}_0^{N-1}$ with $|\Lambda| < p$ be an index set and let A_Λ be the $M \times p$ reduced measurement matrix constructed from selecting the p columns in A with indices found in Λ, and let $G_\Lambda = A_\Lambda^\dagger A_\Lambda$ be the Gramian of A_Λ. As we have noted $G_{ii} = 1$, which implies that $(G_\Lambda)_{ii} = 1$ as well.

Assume now that $p \leq \text{spark}(A)$. This implies that the columns of A_Λ are linearly independent. This in turn implies that G_Λ is positive definite, which means that

$$(G_\Lambda)_{ii} \leq \sum_{j \neq i} (G_\Lambda)_{ij}. \tag{4.71}$$

Now, since $|(G_\Lambda)_{ij}| \leq \mu(A)$, this inequality takes the form

$$1 \leq (p-1)\mu(A), \tag{4.72}$$

or $1 + 1/\mu(A) \leq p$. Bringing this all together, we have shown that

$$1 + \frac{1}{\mu(A)} \leq p \leq \text{spark}(A), \tag{4.73}$$

which proves the first part of the theorem.

The second part now follows directly from lemma 4.1.3, as $2k \leq 1 + 1/\mu(A)$ implies that $2k \leq \text{spark}(A)$, which is the condition presented in the lemma. The proof of the last part will come in the next section. □

This result deserves some discussion. If we combine this result with the Welch bound, we find some the theoretical "best case" performance under these requirements. We pose the question: For a k-sparse signal, how many measurements are at the very least needed in order to restore the signal? Let us assume A obeys the Welch bound, such that

$$\mu(A) = \sqrt{\frac{M-N}{N(M-1)}}. \tag{4.74}$$

To ensure that A is k-ERP, we must require that

$$k < 1 + \frac{1}{3}\sqrt{\frac{M(N-1)}{N-M}}, \tag{4.75}$$

which can be solved for M as

$$M > \frac{9(k-1)^2 N}{N + 9(k-1)^2 - 1}, \tag{4.76}$$

which motivates the following result regarding the number of measurements needed to guarantee a restoration of a signal according to the coherence.

Lemma 4.5.5. For a k-sparse signal we can ensure perfect reconstruction with

$$M \geq Ck^2 \tag{4.77}$$

measurements, where C is some constant.

Next we consider a more general ℓ_1 coherence function.

Definition 4.5.6 (Generalized ℓ_1 coherence function)**.** Let A be a total sensing matrix, with ℓ_2-normalized columns $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_N$. The ℓ_1 coherence function μ_1 of A, $\mu_1(s)$ is defined as

$$\mu_1(k) = \max_{i=0}^{N-1} \max_{\Lambda} \left\{ \sum_{j \in \Lambda} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle| : |\Lambda| = k, i \notin \Lambda \right\}. \tag{4.78}$$

For $k = 1$ this simplifies to the regular coherence. It is straightforward to check that for $1 \leq s \leq N-1$,

$$\mu \leq \mu_1(k) \leq k\mu. \tag{4.79}$$

We end this section with a result that may seem unimportant and and of little use at first sight, but will be useful in the next section.

Theorem 4.5.7. Let A be an $M \times N$ matrix with ℓ_2-normalized columns and let $k \in \mathbb{N}_0^{N-1}$. For all k-sparse vectors $\mathbf{x} \in \mathbb{C}^N$,

$$(1 - \mu_1(k-1))\|\mathbf{x}\|_2^2 \leq \|A\mathbf{x}\|_2^2 \leq (1 + \mu_1(k-1))\|\mathbf{x}\|_2^2. \tag{4.80}$$

We will delay the proof of this result to the next section.

4.6 Restricted Isometry Property

The coherence is a simple and useful measure of the quality of a measurement matrix. The performance guarantee given by the coherence is however quite poor, which leads to ensured recovery only for very small sparsity levels. Here we introduce a finer requirement, called the *Restricted Isometry Property* (RIP), also known as the *uniform uncertainty principle*. This property is considered to be the state of the art when it comes to ensuring success of sparse recovery.

Definition 4.6.1 (Restricted Isometry Property)**.** An $M \times N$ matrix A is said to obey the RIP of order k with constant δ_k (or be (k, δ_k)-RIP) if δ_k is the smallest positive constant such that

$$(1 - \delta_k)\|\mathbf{x}\|_2^2 \leq \|A\mathbf{x}\|_2^2 \leq (1 + \delta)\|\mathbf{x}\|_2^2, \tag{4.81}$$

for all k-sparse vectors.

If we let A satisfy the RIP or order $2k$ with constant $\delta_k < 1$ and $\mathbf{x}, \mathbf{x}'$ be k-sparse vectors. Then the RIP essentially says that the distance between vectors $\mathbf{x} - \mathbf{x}'$ is (roughly) conserved when transformed by A. Note that the factors on the RIP bounds do not really need to be symmetric about 1, we could also allow arbitrary bounds

$$\alpha\|\mathbf{x}\|_2^2 \leq \|A\mathbf{x}\|_2^2 \leq \beta\|\mathbf{x}\|_2^2, \tag{4.82}$$

for any positive and finite α and β.

For any matrix A which obeys the RIP of order k with parameter δ_k, if we select an index set Λ with cardinality $|\Lambda| \leq k$ and store only the columns of A with indices in Λ as the $M \times k$ matrix A_Λ, then the eigenvalues of the $k \times k$ matrix $G_\Lambda = A_\Lambda^T A_\Lambda$ will always lie in the interval $(1-\delta_k, 1+\delta_k)$ [55]. To see this, let $\mathbf{x}$ be a k-sparse vector of unit length, and let Λ be the set $\{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ of indices where $\mathbf{x}$ is nonzero. Now let A_Λ be the $M \times k$ matrix constructed from keeping only the columns of A belonging to Λ. Then $A_\Lambda \mathbf{x}_\Lambda = A\mathbf{x}$, and

$$\|A_\Lambda \mathbf{x}_\Lambda\|_2^2 = (A_\Lambda \mathbf{x}_\Lambda)^\dagger (A_\Lambda \mathbf{x}_\Lambda) \tag{4.83}$$

$$= \mathbf{x}_\Lambda^\dagger A_\Lambda^\dagger A_\Lambda \mathbf{x}_\Lambda \tag{4.84}$$

$$= \mathbf{x}_\Lambda^\dagger G_\Lambda \mathbf{x}, \tag{4.85}$$

where G_Λ is the Gramian of A_Λ. Since $\|A\mathbf{x}\|_2^2 \in (1-\delta_k, 1+\delta_k)$, so is $\mathbf{x}_\Lambda^\dagger G_\Lambda \mathbf{x}$. Since we can choose Λ any way we want, and similarly choose $\mathbf{x}$ any way we want (as long as it is k-sparse), the result holds for any k-sparse vector which implies that all the eigenvalues of G_Λ lie in the interval $(1-\delta_k, 1+\delta_k)$ for any Λ with cardinality k. These steps can be repeated in the opposite direction, which shows that this statement is equivalent to the RIP. We can also phrase this as that for any index set Λ the eigenvalues of $(G_\Lambda - I_k)$, where I_k is the $k \times k$ identity matrix, have a magnitude less than or equal to δ_k, because for any matrix A, if λ is an eigenvalue of A, then $\lambda - 1$ is an eigenvalue of $(A - I_k)$.

With this equivalence shown, we are ready to prove Theorem 4.5.7.

PROOF.[Proof of theorem 4.5.7] Let Λ be an index set with $|\Lambda| \leq k$. We will show that if the eigenvalues of $G_\Lambda = A_\Lambda^\dagger A_\Lambda$ lie in the interval $(1-\mu_1(k-1), 1+\mu_1(k-1))$, which is equivalent to the statement in the theorem, for the same reasons as above. From the Gershgorin circle theorem,

$$|\lambda_i - 1| \leq \sum_{j \neq i} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle|, \tag{4.86}$$

where λ_i are the eigenvalues of G_Λ, from the definition of $\mu_1(k)$ we also have

$$\sum_{j \neq i} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle| \leq \max_{i \in \mathbb{N}_0^{N-1}} \max_{\Lambda} \left\{ \sum_{j \in \Lambda} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle| : |\Lambda| = k-1, i \notin \Lambda \right\} = \mu_1(k-1), \tag{4.87}$$

which implies that $|\lambda_i - 1| \leq \mu_1(k-1)$ for all i. □

If A satisfies the RIP of order k with bound δ_k, the it also satisfies the RIP of order $k' < k$ with a bound $\delta_{k'} < \delta_k$. Generally we have

$$\delta_1 \leq \delta_2 \leq \ldots, \leq \delta_N. \tag{4.88}$$

We also have the following result.

Theorem 4.6.2. If the $M \times N$ matrix A has ℓ_2-normalized columns, then

$$\delta_1 = 0, \tag{4.89}$$

$$\delta_2 = \mu, \tag{4.90}$$

$$\delta_k \leq \mu_1(k-1) \leq (k-1)\mu, \quad k \geq 2. \tag{4.91}$$

PROOF. This proof is from [47], p. 135.

The ℓ_2-normalization means that $\|A\mathbf{e}_j\|_2^2 = \|\mathbf{e}_j\|_2^2$ for all j. This implies that $\delta_1 = 0$.

For the second point, remember that $\mu(A) = \max_{i \neq j} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle|$. For an index set Λ with $\Lambda = 2$, i.e. $\Lambda = \{i, j\}$, $i \neq j$, we have

$$G_\Lambda = \begin{pmatrix} 1 & \langle \mathbf{a}_i, \mathbf{a}_j \rangle \\ \langle \mathbf{a}_j, \mathbf{a}_i \rangle & 1 \end{pmatrix}, \tag{4.92}$$

which can be shown to have the eigenvalues $\lambda = 1 \pm |\langle \mathbf{a}_i, \mathbf{a}_j \rangle|$. This means that

$$|1 - \lambda| = |\langle \mathbf{a}_i, \mathbf{a}_j \rangle| \leq \mu(A), \tag{4.93}$$

with equality for the optimal choice of i and j.

The last set of inequalities follow directly from Theorem 4.5.7 and equation (4.79). □

What is perhaps more surprising is that we can sometimes also guarantee that A satisfies the RIP of order $k' > k$. The following is a result first published in [56].

Lemma 4.6.3. Suppose that A satisfies the RIP of order k with constant δ_k. Let γ be a positive integer. Then A satisfies the RIP of order $k' = \gamma \lfloor \frac{k}{2} \rfloor$ with constant $\delta_{k'} < \gamma \delta_k$, where $\lfloor \cdot \rfloor$ denotes the floor operator.

Note that this will allow us to extend the RIP to higher orders if $\gamma \delta_k$ is sufficiently small.

4.6.1 The ERP and the RIP

We will now show a relation between the ERP and the RIP. We first need a small result.

Lemma 4.6.4. Given $q > p > 0$, if $\mathbf{u} \in \mathbb{C}^S$ and $\mathbf{v} \in \mathbb{C}^T$ satisfy

$$\|\mathbf{u}\|_\infty \leq \min_{j \in \mathbb{N}_0^{T-1}} |v_j|, \tag{4.94}$$

where we remember that $\|\mathbf{u}\|_\infty = \max_{i \in \mathbb{N}_0^{S-1}} |u_i|$, then

$$\|\mathbf{u}\|_q \leq \frac{S^{1/q}}{T^{1/p}} \|\mathbf{v}\|_p. \tag{4.95}$$

As a special case, for $p = 1$, $q = 2$ and $T = S$, we have

$$\|\mathbf{u}\|_2 \leq \frac{1}{\sqrt{S}} \|\mathbf{v}\|_1. \tag{4.96}$$

PROOF. We have that

$$\frac{\|\mathbf{u}\|_q}{S^{1/q}} = \left(\frac{1}{S} \sum_{i=0}^{S-1} |u_i|^q \right)^{1/q} \leq \|\mathbf{u}\|_\infty, \tag{4.97}$$

$$\frac{\|\mathbf{v}\|_q}{T^{1/q}} = \left(\frac{1}{T} \sum_{j=0}^{T-1} |u_j|^q \right)^{1/q} \geq \min_{j \in \mathbb{N}_0^{T-1}} |v_j|, \tag{4.98}$$

which implies that

$$\frac{\|\mathbf{u}\|_q}{S^{1/q}} \leq \|\mathbf{u}\|_\infty \leq \min_{j \in \mathbb{N}_0^{T-1}} |v_j| \leq \frac{\|\mathbf{v}\|_p}{T^{1/p}}. \tag{4.99}$$

□

We need one further result.

Lemma 4.6.5. Let $\mathbf{u}, \mathbf{v} \in \mathbb{C}^N$ be k-sparse vectors and let $A \in \mathbb{C}^{M \times N}$ be $(2k, \delta_{2k})$-RIP. If $\text{supp}(\mathbf{u}) \cap \text{supp}(\mathbf{v}) = \emptyset$. Then

$$|\langle A\mathbf{u}, A\mathbf{v} \rangle| \leq \delta_{2k} \|\mathbf{u}\|_2 \|\mathbf{v}\|_2. \tag{4.100}$$

Proof. This proof is from [47], p. 142.

Let $\Lambda = \text{supp}(\mathbf{u}) \cup \text{supp}(\mathbf{v})$, and let $\mathbf{u}_\Lambda, \mathbf{v}_\Lambda \in \mathbb{C}^{2k}$ be the vectors constructed from keeping only the elements with indices in Λ. Note that $\langle \mathbf{u}_\Lambda, \mathbf{v}_\Lambda \rangle = 0$ because $\text{supp}(\mathbf{u}) \cap \text{supp}(\mathbf{v}) = \emptyset$. We have

$$\begin{aligned}
|\langle A\mathbf{u}, A\mathbf{v} \rangle| &= |\langle A_\Lambda \mathbf{u}_\Lambda, A_\Lambda \mathbf{v}_\Lambda \rangle - \langle \mathbf{u}_\Lambda, \mathbf{v}_\Lambda \rangle \,| && (4.101)\\
&= |\langle (A_\Lambda^\dagger A_\Lambda - I_{2k}) \mathbf{u}_\Lambda, \mathbf{v}_\Lambda \rangle| && (4.102)\\
&\leq \|(A_\Lambda^\dagger A_\Lambda - I_{2k}) \mathbf{u}_\Lambda\|_2 \|\mathbf{v}_\Lambda\|_2 && (4.103)\\
&\leq \|A_\Lambda^\dagger A_\Lambda - I_{2k}\|_2 \|\mathbf{u}_\Lambda\|_2 \|\mathbf{v}_\Lambda\|_2 && (4.104)\\
&= \delta_{2k} \|\mathbf{u}\|_2 \|\mathbf{v}\|_2, && (4.105)
\end{aligned}$$

where we have used that the ℓ_2 matrix norm as defined in Definition 3.5.11 is bounded by the largest eigenvalue of the matrix. □

We can now show the following, which is a very good example of a performance guarantee with the RIP.

Theorem 4.6.6 (The RIP and the ERP)**.** Suppose A satisfies the RIP of order $2k$ with constant

$$\delta_{2k} < \frac{1}{3}. \tag{4.106}$$

Then A also satisfies the ERP of order k.

Proof. This proof is from [47], p. 142-143. We will show that A satisfies the NSP, which is equivalent to the ERP. We aim to the version from equation (4.48),

$$\|\mathbf{x}_\Lambda\|_1 < \frac{1}{2} \|\mathbf{x}\|_1. \tag{4.107}$$

If this hold for any $\mathbf{x} \in \ker A \setminus \{\mathbf{0}\}$, and all index sets Λ such that $|\Lambda| = k$, then A is k-NSP. This will follow from showing that

$$\|\mathbf{x}_\Lambda\|_2 \leq \frac{\rho}{2\sqrt{k}} \|\mathbf{x}\|_1, \tag{4.108}$$

where

$$\rho = \frac{2\delta_{2k}}{1 - \delta_{2k}}. \tag{4.109}$$

which satisfies $\rho < 1$ whenever $\delta_{2k} < 1/3$. Given $\mathbf{x} \in \ker A$, we notice that it is enough to consider an index set $\Lambda = \Lambda_0$ of the indices k entries with the largest magnitude of the vector $\mathbf{x}$. We partition the complement Λ^c of Λ in $\mathbb{N}_0^{N-1}$ as $\Lambda^c = \Lambda_1 \cup \Lambda_2 \cup \ldots, \cup \Lambda_J$, where

$$\Lambda_1 \ : \text{ index set of } k \text{ entries with the largest magnitude in } \Lambda^c \tag{4.110}$$

$$\Lambda_2 \ : \text{ index set of } k \text{ entries with the largest magnitude in } (\Lambda \cup \Lambda_1)^c \tag{4.111}$$

and so on. We have sorted the elements of $\mathbf{x}$ into $J = \lceil N/k \rceil - 1$ bins according to magnitude. Since $\mathbf{x} \in \ker A$, we have $A\mathbf{x}_{\Lambda_0} = A(-\mathbf{x}_{\Lambda_1} - \mathbf{x}_{\Lambda_2} - \ldots \mathbf{x}_{\Lambda_J})$, which means

$$\|\mathbf{x}_{\Lambda_0}\|_2^2 \leq \frac{1}{1-\delta_{2k}} \|A\mathbf{x}_{\Lambda_0}\|_2^2 \tag{4.112}$$

$$= \frac{1}{1-\delta_{2k}} \langle A\mathbf{x}_{\Lambda_0}, A(-\mathbf{x}_{\Lambda_1} - \mathbf{x}_{\Lambda_2} - \ldots - \mathbf{x}_{\Lambda_J}) \rangle \tag{4.113}$$

$$= \frac{1}{1-\delta_{2k}} \sum_{j=1}^{J} \langle A\mathbf{x}_{\Lambda_0}, A\mathbf{x}_{\Lambda_j} \rangle. \tag{4.114}$$

From lemma 4.6.5 we also have

$$\langle A\mathbf{x}_{\Lambda_0}, A\mathbf{x}_{\Lambda_j} \rangle \leq \delta_{2k} \|\mathbf{x}_{\Lambda_0}\|_2 \|\mathbf{x}_{\Lambda_j}\|_2. \tag{4.115}$$

These inequalities can be combined to show

$$\|\mathbf{x}_{\Lambda_0}\|_2 \leq \frac{\delta_{2k}}{1-\delta_{2k}} \sum_{j=1}^{J} \|\mathbf{x}_{\Lambda_j}\|_2 = \frac{\rho}{2} \sum_{j=1}^{J} \|\mathbf{x}_{\Lambda_j}\|_2. \tag{4.116}$$

note that the largest element of $\mathbf{x}_{\Lambda_{j+1}}$ is smaller than the smallest element of $\mathbf{x}_{\Lambda_j}$, so we have

$$\|\mathbf{x}_{\Lambda_k}\|_2 \leq \frac{1}{\sqrt{k}} \|\mathbf{x}_{\Lambda_{k-1}}\|_1. \tag{4.117}$$

This gives

$$\|\mathbf{x}_{\Lambda_0}\|_2 \leq \frac{\rho}{2} \sum_{j=1}^{J} \|\mathbf{x}_{\Lambda_j}\|_2 \leq \frac{\rho}{2\sqrt{k}} \sum_{k=1}^{J} \|\mathbf{x}_{\Lambda_{k-1}}\|_1 \leq \frac{\rho}{2\sqrt{k}} \|\mathbf{x}\|_1, \tag{4.118}$$

which is what we aimed to show. □

Note that in (4.112), we used that $\mathbf{x}_{\Lambda_0}$ was $2k$-sparse. This is true, but it is in fact k-sparse, which implies $\|\mathbf{x}_\Lambda\|_2^2 \leq \|A\mathbf{x}_{\Lambda_0}\|_2^2/(1-\delta_k)$. Using this would give the sufficient requirement

$$\delta_k + \delta_{2k} \leq 1, \tag{4.119}$$

which is easier to satisfy than (4.106). We can now show a weaker version of the last part of theorem 4.5.4 (the ERP-requirement for the coherence). We have shown that

$$\delta_{2k} \leq (2k-1)\mu, \tag{4.120}$$

which means that A is k-ERP if $(2k-1)\mu < 1/3$, or $k < 1/2 + 1/(6\mu)$. A better bound on the δ_{2k} needed for k-ERP will give a better bound for the coherence result.

Finally we return to the question on the number of measurements needed to ensure restoration of a k-sparse signal $\mathbf{x}$. The following result is from [57].

Theorem 4.6.7. Let A be an $M \times N$ matrix that satisfies the RIP of order $k \leq N/2$ with constant $\delta_k \in (0, 1)$. Then

$$M \geq C_{\delta_k} k \log(N/k), \tag{4.121}$$

where $C_{\delta_k} < 1$ is a constant depending only on δ_k.

Because the log-function so greatly dampens the dependency on N, this is usually a much smaller number of measurements needed that with the coherence, which is proportional to k^2. This is the reason why matrix constructions based on the RIP have been more popular than constructions based on the coherence.

4.6.2 Better bounds on the RIP constant

Here we have shown $\delta_{2k} < 1/3$ to be a sufficient requirement. We used many approximations, and it is possible through a more elegant approach to bring the requirement up substantially. In this section we briefly highlight the improvements and bounds made on the RIP constant.

The first results involved both δ_{2k} and δ_{3k} or similar combinations.

Theorem 4.6.8 (Candès and Tao, 2005 [58])**.** If $\delta_{2k} + \delta_{3k} < 1$, A is k-ERP.

Theorem 4.6.9 (Candès and Tao, 2006 [59])**.** If $\delta_{3k} + \delta_{4k} < 2$, A is k-ERP.

The first notable result involving only δ_{2k} came in 2008.

Theorem 4.6.10 (Candès, 2008 [60])**.** If $\delta_{2k} < 1$, the problem (P0) has a unique solution. Furthermore, if $\delta_{2k} < 1 - \sqrt{2}$, A is k-ERP.

This result was improved on several times, notably:

Theorem 4.6.11 (Foucart, 2009 [61])**.** If $\delta_{2k} < 0.4531$, A is k-ERP.

Theorem 4.6.12 (Foucart, 2010 [62])**.** If $\delta_{2k} < 0.4652$, A is k-ERP.

Theorem 4.6.13 (Cai, Wang and Xu, 2010 [63])**.** If $\delta_{2k} < 0.472$, A is k-ERP.

This paper also introduced the shifting inequality, which has been a key tool in producing better bounds. The same authors also introduced the *square root lifting inequality* in [64] which further helped improve bounds.

Theorem 4.6.14 (The shifting inequality and the square root lifting inequality)**.** Given $a_1 \geq a_2 \geq \ldots \geq a_{k+l} \geq 0$,

$$\sqrt{a_{l+1}^2 + \ldots + a_{l+k}^2} \leq c_{k,l}(a_1 + \ldots + a_k), \quad \text{where } c_{k,l} = \max\{\frac{1}{\sqrt{k}}, \frac{1}{\sqrt{4l}}\}. \tag{4.122}$$

This is known as the shifting inequality. Furthermore, for $a_1 \geq a_2 \geq \ldots \geq a_k \geq 0$,

$$\sqrt{a_1^2 + \ldots + a_k^2} \leq \frac{a_1 + \ldots + a_k}{\sqrt{k}} + \frac{\sqrt{k}}{4}(a_1 - a_k), \tag{4.123}$$

which is known as the square root lifting inequality.

Using these, a new bound was shown in 2011.

Theorem 4.6.15 (Mo and Li, 2011 [65])**.** If $\delta_{2k} < 0.4931$, A is k-ERP.

The newest bound, available on arXiv, but has yet to be published in a peer-reviewed journal.

Theorem 4.6.16 (Andersson and Strömberg, 2013 [66])**.** If $\delta_{2k} < 4/\sqrt{41} \approx 0.62$, A is k-ERP.

The following was shown in 2009, giving an upper bound on the RIP constant.

Theorem 4.6.17 (Davies and Gribonval, 2009 [67])**.** There are matrices A_n with RIP constants arbitrarily close to $1/\sqrt{2} \approx 0.7071$ such that A_n is **not** k-ERP.

Some bounds have also been made on δ_k directly.

Theorem 4.6.18 (Cai, Wang and Xu, 2010 [64])**.** If $\delta_k < 0.307$, A is k-ERP.

This was improved on recently.

Theorem 4.6.19 (Cai and Zhang, 2013 [68])**.** If $\delta_k < 1/3$, A is k-ERP.

This is the result we have used in the last part of theorem 4.5.4.

4.7 Known good sensing matrix constructions

With the theory developed, we are now ready to investigate some of the known popular matrix constructions used in CS. The most popular constructions are based on the RIP, but we can obtain decent constructions based on the coherence as well. Let us first study this approach.

4.7.1 Matrices with low coherence

We begin this section by defining some properties relevant to matrices with minimal coherence, before we will give some specific examples of such matrices. The columns of a coherence-optimal matrix form an *equiangular tight frame* (ETF). Equiangularity and tightness are two properties which we will define separately.

Definition 4.7.1 (Equiangular dictionary)**.** A system of ℓ_2-normalized vectors $(\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{N-1})$ is called equiangular if there is a constant c such that

$$|\langle \mathbf{a}_i, \mathbf{a}_j \rangle| = c, \quad i \neq j. \tag{4.124}$$

We have already given the definition of a tight frame in the context of curvelets, but here we include a somewhat more complete definition.

Definition 4.7.2 (Tight frame)**.** A system of vectors $(\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{N-1}) \in \mathbb{C}^M$, is called a tight frame if there exists a constant λ such that one of the following equivalent statements hold:

1. $\|\mathbf{x}\|_2^2 = \lambda \sum_{j=0}^{N-1} |\langle \mathbf{x}, \mathbf{a}_j \rangle|^2 \quad$ for all $\mathbf{x} \in \mathbb{C}^M$,
2. $\mathbf{x} = \lambda \sum_{j=0}^{N-1} \langle \mathbf{x}, \mathbf{a}_j \rangle \mathbf{a}_j \quad$ for all $\mathbf{x} \in \mathbb{C}^M$,
3. $AA^\dagger = \frac{1}{\lambda} I_M$, where A is the matrix with columns $\mathbf{a}_0, \ldots, \mathbf{a}_{N-1}$.

We now have the following result.

Theorem 4.7.3. Let A be an $M \times N$ matrix with ℓ_2-normalized columns. The coherence of A satisfies the Welch bound if and only if A is an ETF.

Proof. This proof is from [47], p. 114.

Let $G = A^\dagger A$ be the Gramian of A, and let $H = AA^\dagger$. We have that

$$\operatorname{tr}(G) = \sum_{i=0}^{N-1} \|\mathbf{a}_i\|_2^2 = N, \tag{4.125}$$

where $\operatorname{tr}(G)$ is the trace of G (the sum of the diagonal elements). In general, for any matrices X and Y,

$$\operatorname{tr}(X^\dagger Y) = \operatorname{tr}(XY^\dagger), \tag{4.126}$$

which means $\operatorname{tr}(H) = N$ as well.

We now introduce the Frobenius norm for matrices,

$$\|A\|_F = \left(\sum_{i,j} |A_{ij}|^2 \right)^{1/2} = \sqrt{\operatorname{tr}(A^\dagger A)}. \tag{4.127}$$

The inner product which induces this norm is given by

$$\langle A, B\rangle_F = \operatorname{tr} AB^\dagger. \tag{4.128}$$

The Cauchy-Schwartz inequality yields

$$\operatorname{tr}(H) = \langle H, I\rangle_F \leq \|H\|_F \|I\|_F = \sqrt{M}\sqrt{\operatorname{tr}(HH^\dagger)}. \tag{4.129}$$

We now observe that

$$\operatorname{tr}(HH^\dagger) = \operatorname{tr}(AA^\dagger AA^\dagger) = \operatorname{tr}(A^\dagger AA^\dagger A) = \operatorname{tr}(GG^\dagger) = \sum_{i,j} |\langle \mathbf{a}_j, \mathbf{a}_j\rangle|^2 \tag{4.130}$$

$$= N + \sum_{i\neq j} |\langle \mathbf{a}_j, \mathbf{a}_j\rangle|^2. \tag{4.131}$$

Putting this together, and squaring everything, we have

$$N^2 \leq M\left(N + \sum_{i\neq j} |\langle \mathbf{a}_j, \mathbf{a}_j\rangle|^2\right). \tag{4.132}$$

If we now let $\mu = \mu(A)$, then

$$|\langle \mathbf{a}_i, \mathbf{a}_j\rangle| \leq \mu, \tag{4.133}$$

which we can use to obtain

$$N^2 \leq M(N + (N^2 - N)\mu^2), \tag{4.134}$$

which can be rearranged to show the Welch bound. Furthermore, this holds with equality only if and only if (4.129) and (4.133) hold with equality, i.e. if and only if $|\langle \mathbf{a}_i, \mathbf{a}_j\rangle| = \mu$ (equiangularity) and $\langle H, I\rangle_F = \|H\|_F\|I\|_F$, which holds if and only if $H = \frac{1}{\lambda}I$ (tightness version 3). □

In the setting of CS, we would like to construct matrices which are maximally incoherent matrices $M \times N$ matrices, where N is as large as possible compared to M (we want as few measurements as possible). The following result give a useful upper bound to how large N can possibly be for an ETF.

Theorem 4.7.4. Let $A \in \mathbb{K}^{M\times N}$ where $\mathbb{K}$ is either $\mathbb{R}$ or $\mathbb{C}$, with ℓ_2-normalized columns $(\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{N-1})$ such that the columns of A form an equiangular system. Then

$$N \leq M(M+1)/2 \qquad \text{when } \mathbb{K} = \mathbb{R}, \tag{4.135}$$

$$N \leq M^2 \qquad \text{when } \mathbb{K} = \mathbb{C}. \tag{4.136}$$

If equality is achieved, then the system $(\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{N-1})$ is also a tight frame.

The proof for this theorem is perfectly understandable, but quite lengthy, so we omit it here. It can be found in [47], page 117-119.

There is no known guarantee that systems of this maximum size exist. In fact, for the case $\mathbb{K} = \mathbb{R}$, it can be shown that there are some M where no such system exists. For $\mathbb{K} = \mathbb{C}$, numerical experiments have found maximally sized ETFs for $M < 45$ [69], but it remains at open question whether they exist for any M.

The theory of creating sensing matrices based on coherence is usually centered around creating ETFs. Let us now turn to some examples of these.

Examples of matrix constructions with low coherence

There are many ways to construct ETFs. Here we will only highlight a few. The first, which serves as an example without much use in CS, will be defined by the vertices of a *regular simplex*.

Definition 4.7.5 (Simplex)**.** Let $\mathbf{e}_1 = (1, 0, \ldots, 0)^T$, $\mathbf{e}_2 = (0, 1, \ldots, 0)^T$ and so on up to $\mathbf{e}_N$. the standard simplex in N dimensions is constructed by considering all linear combinations

$$\mathbf{x} = \sum_i c_i \mathbf{e}_i, \tag{4.137}$$

such that $c_i \geq 0$ and $\sum_i c_i = 1$. This is a point in 1-D, a line in 2-D, a triangle in 3-D and so on.

The regular simplex is constructed by considering a collection of $N + 1$ points in $\mathbb{R}^N$ such that,

1. There is a point $\mathbf{b}$ such that all $N + 1$ points $\mathbf{e}_i$ are equally far from $\mathbf{b}$. $\mathbf{b}$ is called the center of the simplex.
2. the angle between any two points and the center is $\arccos(-1/n)$

This is a line in 1-d, an equiangular triangle in 2-D, a regular tetrahedron in 3-D and so on.

The final point can be interpreted as saying that if we choose a coordinate system with $\mathbf{b}$ as its origin, the the dot product of any two points $\mathbf{e}_i$, $\mathbf{e}_j$ is $-1/N$.

Theorem 4.7.6. The $N+1$ vertices of a regular simplex in $\mathbb{R}^N$ centered around the origin form an equiangular tight frame.

Proof. We have already noted that $|\langle \mathbf{e}_i, \mathbf{e}_j \rangle| = 1/N$. The Welch bound for the case of an $N \times (N + 1)$ matrix is given by

$$\sqrt{\frac{N + 1 - N}{N(N + 1 - 1)}} = \frac{1}{N}, \tag{4.138}$$

which is sufficient to prove that the vertices form an ETF by theorem 4.7.3. □

This example shows that we are able to create ETFs, but in the context of CS it is not very useful as we usually want the number of measurements to be far smaller than N. The next example will be more relevant. We will now show that under certain conditions, the partial Fourier measurements we have already studied will form equiangular tight frames.

The following equivalence was proved in two parts in [70] and [71]:

Theorem 4.7.7. Construct $\mathbf{\Phi}$ by collecting rows from the $N \times N$ DFT which are indexed by Λ. Then $\mathbf{\Phi}$ is an ETF if and only if Λ is a *difference set* for $\mathbb{N}_0^{N-1}$. That is, there exists a positive integer λ such that every nonzero element in $\{0, 1, \ldots, N - 1\}$ can be expressed as the difference of members of Λ in exactly λ different ways.

N	M	DFT row indices	μ	Largest allowable k
7	3	$\{1, 2, 4\}$	0.4714	1
7	4	$\{0, 3, 5, 6\}$	0.3536	1
13	4	$\{0, 1, 3, 9\}$	0.4339	1
11	5	$\{1, 3, 4, 5, 9\}$	0.3464	1
21	5	$\{3, 6, 7, 12, 14\}$	0.4	1
11	6	$\{0, 2, 6, 7, 8, 10\}$	0.2886	2
31	6	$\{1, 5, 11, 24, 25, 27\}$	0.3727	1
15	7	$\{0, 1, 2, 4, 5, 8, 10\}$	0.2857	2
15	8	$\{3, 6, 7, 9, 11, 12, 13, 14\}$	0.25	2
57	8	$\{1, 6, 7, 9, 19, 38, 42, 49\}$	0.3307	2
13	9	$\{2, 4, 5, 6, 7, 8, 10, 11, 12\}$	0.1925	2
37	9	$\{1, 7, 9, 10, 12, 16, 26, 33, 34\}$	0.2940	2
73	9	$\{1, 2, 4, 8, 16, 32, 37, 55, 64\}$	0.3143	2
40	13	$\{0, 1, 3, 5, 9, 15, 22, 26, 27, 34, 35, 38\}$	0.2308	2

Table 4.3: Some known difference sets, along with μ and the largest k for which reconstruction is possible according to Theorem 4.5.4.

Finding difference sets is no easy task, and as we will see have seen, we can get better performance guarantees with a RIP-analysis, so this approach is rarely used. It does however give deterministic restoration guarantees. Table 4.3 is taken from [71] and lists some known difference sets, along with the highest possible k for which the k-ERP is obeyed according to theorem 4.5.4.

We now leave the coherence based constructions behind, and focus on constructions based on the RIP. These allow for the best known performance guarantees.

4.7.2 Random sensing matrices

We have noted that the RIP will allow for larger M than the coherence, as the best-case RIP is bounded by

$$M \geq C_{\delta_k} k \log(N/k) = \mathcal{O}(k), \tag{4.139}$$

while the coherence gives bounds by

$$M \geq Ck^2 = \mathcal{O}(k^2). \tag{4.140}$$

By matrices with good RIP properties, we generally mean matrices which improves on the coherence bound. This has proven to be a difficult task, and for this reason the coherence bound is often referred to as the "quadratic bottleneck" (or "square root bottleneck" if you consider it a bound on k). As of writing, there are no known deterministic ways of constructing matrices with bounds of order $\mathcal{O}(k)$.

There are, however, a few constructions which has been shown to yield good RIP constants with high probability. In this section we will highlight the existence of such matrices. We will forgo most proofs, as they would require theory which is beyond the scope of this text. Let us first define the matrices we will be studying.

Definition 4.7.8 (Random matrices)**.** Let A be an $M \times N$ matrix.

1. If the elements of A are independent Bernoulli random variables, taking values +1 or -1 with equal probability, then A is a Bernoulli random matrix.
2. If the elements of A are independent Gaussian random variables, then A is a Gaussian random matrix.
3. A random variable x is called *subgaussian* if there are parameters β, κ such that

$$\mathbb{P}(|x| \geq t) \leq \beta e^{-\kappa t^2}, \tag{4.141}$$

where $\mathbb{P}(|x| \geq t)$ is the probability that $|x| \geq t$.

If the elements of A are independent zero-mean subgaussian random variables then A is a subgaussian random matrix.

It is easy to check that Gaussian and Bernoulli matrices are both examples of subgaussian matrices. The following is the main result on the RIP for subgaussian random matrices.

Theorem 4.7.9. Let A be an $M \times N$ subgaussian random matrix. Then there exists a constant $C > 0$, depending only on κ and β, such that the restricted isometry constant of $\frac{1}{\sqrt{M}}A$ satisfies $\delta_k \leq \delta$ with probability at least $1 - \epsilon$, provided

$$M \geq C\delta^{-2}(k \ln(eN/k) + \ln(2\epsilon^{-1})). \tag{4.142}$$

Setting $\epsilon = 2\exp(-\delta^2 M/(2C))$ yields the condition

$$M \geq 2C\delta^{-2}k \ln(eN/k), \tag{4.143}$$

with probability at least $1 - 2\exp(-\delta^2 M/(2C))$, which is a result often quoted in literature. This result was shown independently by Mendelson et al. [72] and Baraniuk et al. [73]. This result holds for Gaussian and Bernoulli matrices as special cases. For Gaussian matrices, the factor 2 in (4.143) can be removed. This is currently the best known matrix construction for sensing.

Another advantage of a random sensing scheme is in the cases where $\mathbf{x}$ is not sparse in the canonical basis, but rather in some basis $\mathbf{\Psi}$, it is really the total sensing matrix $A\mathbf{\Psi}$ which should obey the RIP. In [73] Barniuk et al. showed that if the RIP holds for a subgaussian random matrix A, it also holds for $A\mathbf{\Psi}$ with high probability, where $\mathbf{\Psi}$ is any orthonormal basis.

Random sensing is very efficient in terms of the number of measurements needed for restoration, but in real settings often not very viable. In the next section we will look at a construction which gives better results than the coherence, and is more plausible in practice.

4.7.3 Structured random sensing matrices

In the opening section we highlighted random Fourier measurements as giving very good recovery properties (optimal recovery properties, as we have seen now). This will turn out to be a special

case of a much wider class of sensing matrices, constructed from Bounded Orthonormal Systems (BOS). The following is a somewhat edited/simplified presentation of the result collected in [74].

Let $\mathcal{D} \subset \mathbb{R}^d$ be endowed with some probability measure ν. Formally a probability measure is a real valued function defined on a set of events in probability space that satisfies measure properties such as countable additivity. We will not pay much attention to this property here, but it is necessary to define what follows. Further, let $\mathbf{\Phi} = \{\phi_1, \ldots, \phi_N\}$ be an orthonormal system of real-valued functions on $\mathcal{D}$, that is, for $j, k \in \mathbb{N}_1^N$,

$$\int_{\mathcal{D}} \phi_j(\mathbf{t})\overline{\phi_k(\mathbf{t})}\, d\nu(\mathbf{t}) = \delta_{j,k}. \tag{4.144}$$

Definition 4.7.10. We say that $\mathbf{\Phi}$ is a BOS with constant K if it satisfies (4.144) and if

$$\|\phi_j\|_\infty = \sup_{\mathbf{t}\in\mathcal{D}} |\phi_j(\mathbf{t})| \leq K \quad \text{for all } j \in \mathbb{N}_1^N. \tag{4.145}$$

The smallest value that the constant K can take is $K = 1$. In this case, we necessarily have $|\phi_j(\mathbf{t})| = 1$ for ν-almost all $\mathbf{t} \in \mathcal{D}$ (ν-almost means the probability for $|\phi_j(\mathbf{t})| \neq 1$ is 0 with the given probability measure).

We consider functions of the form

$$f(\mathbf{t}) = \sum_{j=1}^{N} x_k \phi_k(\mathbf{t}). \tag{4.146}$$

Let $\mathbf{t}_1, \ldots, \mathbf{t}_M$ be some sampling points and suppose we are given the sample values

$$y_l = f(t_l)) \sum_{j=1}^{M} x_j \phi_j(\mathbf{t}_l). \tag{4.147}$$

Introducing the sample matrix $A \in \mathbb{C}^{M\times N}$ with entries

$$A_{lk} = \phi_k(\mathbf{t}_l), \tag{4.148}$$

the vector $\mathbf{y}$ of sampled data can be written as

$$\mathbf{y} = A\mathbf{x}. \tag{4.149}$$

We next highlight some examples of BOSs.

1. **Trigonometric polynomials**. Let $\mathcal{D} = [0, 1]$ and for $j \in \mathbb{Z}$ set

$$\phi_j(\mathbf{t}) = e^{2\pi i j \mathbf{t}}. \tag{4.150}$$

The probability measure is the regular integral measure on $[0, 1]$. Then

$$\int_0^1 \phi_j \phi_l \, d\mathbf{t} = \delta_{j,l}, \tag{4.151}$$

and the constant K is $K = 1$. We then consider trigonometric polynomials on the form

$$f(\mathbf{t}) = \sum_{j\in\Lambda} x_j \phi_k(\mathbf{t}) = \sum_{j\in\Lambda} x_j e^{2\pi i j \mathbf{t}}, \tag{4.152}$$

where a common choice for Λ is $\Lambda = \{-q, -q+1, \ldots, q-1, q\}$ which yields the normal Fourier series for continuous functions. Note however that we can chose Λ in other ways. The sampling points $\mathbf{t}_1, \ldots, \mathbf{t}_M$ will be chosen independently and uniformly at random from $[0, 1]$. The sensing matrix is then

$$A_{l,j} = e^{2\pi i j t_l}. \tag{4.153}$$

This is sometimes called a non-equispaced Fourier matrix, as the sampling times are not chosen equidistantly.

2. **Real Trigonometric Polynomials**. Rather than using complex exponentials, we can use sums of sines and cosines such as

$$\phi_{2j}(\mathbf{t}) = \sqrt{2}\cos(2\pi j \mathbf{t}), \quad j \in \mathbb{N}_0^{N-1}, \tag{4.154}$$

$$\phi_{2j+1}(\mathbf{t}) = \sqrt{2}\sin(2\pi j t), \quad j \in \mathbb{N}_0^{N-1}. \tag{4.155}$$

3. **Discrete Orthonormal Systems**. Let $U \in \mathbb{C}^{N\times N}$ be a unitary matrix. The normalized columns $\sqrt{N}\mathbf{u}_k$ form an orthonormal system with respect to the discrete uniform probability measure on $\mathbb{N}_1^N$. This means than

$$\frac{1}{N}\sum_{t=1}^{N} \sqrt{N}\mathbf{u}_k(\mathbf{t})\overline{\sqrt{N}\mathbf{u}_l(\mathbf{t})} = \langle \mathbf{u}_j, \mathbf{u}_l\rangle = \delta_{j,l}. \tag{4.156}$$

The boundedness requires that the normalized columns are bounded, i.e.,

$$M(U) = \sqrt{N}\max_{j,k\in\mathbb{N}_1^N} |U_{jk}| = \max_{j\in\mathbb{N}_1^N} |\sqrt{N}\mathbf{u}_j(\mathbf{t})| \le K. \tag{4.157}$$

where $M(U)$ is the coherence of elements introduced in section 4.3.

Choosing the points $\mathbf{t}_1, \ldots, \mathbf{t}_M$ uniformly at random corresponds to creating the random matrix A by selecting its rows independently and uniformly at random from the rows of $\sqrt{N}U$, that is,

$$A = \sqrt{N}R_T U, \tag{4.158}$$

where R_T is the random subsampling operator.

4. **Incoherent Bases**. We have already noted this one in section 4.3. Let $\mathbf{\Psi}$, $\mathbf{\Phi} \in \mathbb{C}^{N\times N}$ be two unitary matrices with columns $(\psi_1, \ldots, \psi_N)$ and $(\phi_1, \ldots, \phi_N)$, respectively. Assume that a vector $\mathbf{z} \in \mathbb{C}^N$ is sparse with respect to the basis $\{\psi_j\}$ rather than the canonical basis, i.e. $\mathbf{z} = \mathbf{\Psi}\mathbf{x}$ for some sparse $\mathbf{x}$. Further, assume that $\mathbf{z}$ us sampled with respect to the basis $\{\phi_l\}$, i.e. we obtain measurements

$$y_l = \langle \mathbf{z}, \phi_{t_k}, \quad k = 1, \ldots, M, \tag{4.159}$$

with $\{t_1, \ldots, t_M\} \in \mathbb{N}_1^N$. The matrix form of this equation is then

$$\mathbf{y} = R_T \mathbf{\Phi}^\dagger \mathbf{z} = R_T \mathbf{\Phi}^\dagger \mathbf{\Psi} \mathbf{x}. \tag{4.160}$$

The boundedness condition now reads

$$M(\mathbf{\Phi}, \mathbf{\Psi}) = \sqrt{N} \max_{j,l} |\langle \phi_j, \psi_l \rangle| \leq K. \tag{4.161}$$

Examples of highly incoherent bases are, as we have seen, the canonical and the Fourier bases, and the Haar and Noiselet bases.

5. **Legendre Polynomials**. The Legendre polynomials P_j are a system of orthogonal polynomials, where P_j is a polynomial of precise degree j, and orthogonality is with respect to the integral inner product measure on $[-1, 1]$. Their supremum norm is given by $\|P_j\|_\infty = \sqrt{2j+1}$, so considering the polynomials $P_1, \ldots, P_N$ gives the constant $K = \sqrt{2N-1}$. Unfortunately, K grows rather quickly with N. This problem can be avoided with a small trick. One takes sampling points with respect to the "Chebyshev" measure $d\nu(x) = \pi^{-1}(1-x^2)^{-1/2}\,dx$ and uses a preconditioned measurement matrix. The reader is invited to investigate [75] for details.

Nonuniform vs uniform recovery

If we look back to theorem 4.3.3, we note that the formulation of this theorem is slightly different than the performance guarantees given in section 4.4. The theorem states that "for a given sparse vector $\mathbf{x}$...", while the other theorems give statements of the form "for any sparse vector $\mathbf{x}$...". This marks the difference between *nonuniform* and *uniform* recovery. For nonuniform recovery, we look at the probability that a randomly chosen sensing matrix can restore a specific k-sparse vector, while for uniform recovery, we look at the probability that a randomly chosen matrix can restore any k-sparse vector. We have the following results for nonuniform and uniform recovery.

Theorem 4.7.11 (Nonuniform recovery)**.** A fixed k-sparse vector can be reconstructed via ℓ_1-minimization with probability higher that $1-\epsilon$ if measurements are made using M uniformly random samples from a BOS with constant $K \geq 1$ provided that

$$M \geq CK^2 k \ln(N) \ln(\epsilon^{-1}), \tag{4.162}$$

where C is a universal constant.

The proof of this first given by Candès and Plan in 2011 [76]. Notably, in the case of random partial Fourier measurements, this can be improved to

$$M \geq CK^2 k \ln(N/\epsilon), \tag{4.163}$$

which is better. However, the proof of this relies heavily on the structure of the Fourier transform. It is presently not known whether the bound for general BOSs can be improved to reach the Fourier version.

Theorem 4.7.12 (Uniform recovery)**.** Let $A \in \mathbb{C}^{M\times N}$ be a random sampling matrix associated to a BOS with constant $K \geq 1$. If, for $\delta \in (0,1)$,

$$M \geq CK^2\delta^{-2}k\ln^3(N), \tag{4.164}$$

then with probability at least $1 - N^{\ln^3(N)}$ the restricted isometry constant δ_k of $\frac{1}{\sqrt{M}}A$ satisfies $\delta_k < \delta$. C is a universal constant.

The first bound was found in the early papers on CS by Candès and Tao as $M \geq Ck\ln^5(N)\ln(\epsilon^{-1})$ in 2006 [77]. This was improved by Rudelson and Vershynin in [78] later that same year. The bound as presented above was published by Cheraghchi et al. in 2013. The proof can be found in [79].

Other structured random sensing matrices

- **random circulant Toeplitz matrices** It turns out we can use the circulant Toeplitz matrices introduced in section 3.3.2 as a basis for sensing matrices. The following result was noted in [80].

  Theorem 4.7.13. Let $\mathbf{b} \in \mathbb{R}^N$ be a vector with elements randomly set to +1 or -1 with equal probability, and let $\mathbf{b}$ be the filter for a circulant Toeplitz matrix $T^{\mathbf{b}}$. Further, let $\Lambda \subseteq \mathbb{N}_0^{N-1}$ be a set of M random chosen indices, and let $T^{\mathbf{b}}_\Lambda$ be the $M \times N$ matrix constructed from keeping only the rows indexed by Λ. Now let $\mathbf{x}$ be a k-sparse vector and let $\mathbf{y} = T^{\mathbf{b}}_\Lambda$. Then if

  $$M \geq Ck\log^3(N/\epsilon), \tag{4.165}$$

  $\mathbf{x}$ can be restored with ℓ_1-minimization with probability at least $1-\epsilon$.

- **Spherical Harmonics**. It has also been noted that one can find signals with a sparse representation in spherical harmonics quite efficiently. In [81] it was noted that if we let $f(\phi,\theta)$ be some function defined on N points on the unit sphere $\{(\phi_1,\theta_1),(\phi_2,\theta_2),\ldots,(\phi_N,\theta_N)\}$, and also let f have a k-sparse , and one draws M points uniformly at random from this set, then the function f could be recovered with high probability given

  $$M \geq Ck\log^3(k)N^{1/4}\log(N). \tag{4.166}$$

4.8 Signals with noise

So far we have considered signals which were exactly sparse. In a realistic setting, there will always be some noise in the measurements, which means that $\mathbf{y}$ is only an approximation to $A\mathbf{x}$, with

$$\|A\mathbf{x} - \mathbf{y}\| \leq \eta. \tag{4.167}$$

where the norm used is typically the ℓ_2 norm (though any norm can be used). In this case we want to solve (P1-σ), and hopefully make guarantees of the form $\|\mathbf{x}-\mathbf{x}^\#\| \leq f(\sigma_k(\mathbf{x}),\eta)$, where $\mathbf{x}^\#$ is the solution found by the minimization.

Guarantees can be made in this relaxed case. First we define a more robust null space property.

Definition 4.8.1. The matrix $A \in \mathbb{C}^{M\times N}$ is said to satisfy the *robust null space property*(RNSP) with respect to a given norm $\|\cdot\|$, with constants $0 < \rho < 1$ and $\tau > 0$ relative to a set Λ (or to be Λ-RNSP) if

$$\|\mathbf{v}_\Lambda\|_1 \leq \rho\|\mathbf{v}_{\Lambda^c}\|_1 + \tau\|A\mathbf{v}\|_2, \tag{4.168}$$

for all $\mathbf{v} \in \mathbb{C}^N$.

Note that in this case, we do not require $\mathbf{v} \in \ker(A)$. If this holds, then the $A\mathbf{v}$ term disappears, and we are left with something similar to the regular NSP. A is said to be k-RNSP if A is Λ-RNSP for any set Λ such that $|\Lambda| \leq k$

The following result is just what we want in terms of performance guarantees.

Theorem 4.8.2. Suppose that $A \in \mathbb{C}^{M\times N}$ is k-RNSP with constants ρ and τ. Then for any $\mathbf{x} \in \mathbb{C}^N$, a solution $\mathbf{x}^\#$ of (P1-σ) with tolerance η, and with $\mathbf{y} = A\mathbf{x} + \mathbf{e}$ and $\|\mathbf{e}\| \leq \eta$ approximates the vector $\mathbf{x}$ with ℓ_1-error

$$\|\mathbf{x} - \mathbf{x}^\#\|_1 \leq \frac{2(1+\rho)}{(1-\rho)}\sigma_k(\mathbf{x})_1 + \frac{4\tau}{1-\rho}\eta. \tag{4.169}$$

We will prove this by proving a stricter property for any index set Λ.

Theorem 4.8.3. The matrix $A \in \mathbb{C}^{M\times N}$ is Λ-RNSP if and only if

$$\|\mathbf{z} - \mathbf{x}\|_1 \leq \frac{1+\rho}{1-\rho}(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1) + \frac{2\tau}{1-\rho}\|A(\mathbf{z}-\mathbf{x})\|_2, \tag{4.170}$$

for all vectors $\mathbf{x}, \mathbf{z} \in \mathbb{C}^N$.

Note that if this property holds, then the previous theorem follows simply by setting $\mathbf{z} = \mathbf{x}^\#$ and noting first that from the triangle inequality,

$$\|A(\mathbf{x}^\# - \mathbf{x})\| \leq \|A\mathbf{x}^\# - \mathbf{y}\| + \|\mathbf{y} - A\mathbf{x}\| \leq 2\eta, \tag{4.171}$$

and secondly that if we chose Λ to be the index set of the k largest elements of $\mathbf{x}$, then $\|\mathbf{x}_{\Lambda^c}\|_1 = \sigma_k(\mathbf{x})$, and since $\mathbf{x}^\#$ is the result of (P1-σ), $\|\mathbf{x}^\#\|_1 \leq \|\mathbf{x}\|_1$, so

$$\|\mathbf{x}^\#\|_1 - \|\mathbf{x}\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1 \leq 2\|\mathbf{x}_{\Lambda^c}\|_1 = 2\sigma(\mathbf{x})_1. \tag{4.172}$$

We now continue with the proof.

Proof. This proof is from [47] p. 86-87, but some details have been added to make it more digestible. Assume first that (4.170) holds for all vectors $\mathbf{x}, \mathbf{z} \in \mathbb{C}^N$. Then let $\mathbf{x} = -\mathbf{v}_\Lambda$ and $\mathbf{z} = \mathbf{v}_{\Lambda^c}$. Then $\mathbf{x}_{\Lambda^c} = (-\mathbf{v}_\Lambda)_{\Lambda^c} = 0$, so inserting this into (4.170) we get

$$(1-\rho)\|\mathbf{v}_\Lambda + \mathbf{v}_{\Lambda^c}\|_1 \leq (1+\rho)(\|\mathbf{v}_{\Lambda^c}\|_1 - \|\mathbf{v}_\Lambda\|_1) + 2\tau\|A\mathbf{v}\|_2. \tag{4.173}$$

This can be rearranged to

$$\|\mathbf{v}_\Lambda\|_1 \leq \rho\|\mathbf{v}_{\Lambda^c}\|_1 + \tau\|A\mathbf{v}\|_2, \tag{4.174}$$

which shows that A is k-RNSP.

Now assume that A is k-RNSP. First note that

$$\|\mathbf{x}_{\Lambda^c} - \mathbf{z}_{\Lambda^c}\|_1 \leq \|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + \|\mathbf{x}_\Lambda - \mathbf{z}_\Lambda\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1. \tag{4.175}$$

This can be shown by adding the the two inequalities

$$\|\mathbf{x}\|_1 = \|\mathbf{x}_{\Lambda^c}\|_1 + \|\mathbf{x}_\Lambda\|_1 \leq \|\mathbf{x}_{\Lambda^c}\|_1 + \|\mathbf{x}_\Lambda - \mathbf{z}_\Lambda\|_1 + \|\mathbf{z}_\Lambda\|_1 \tag{4.176}$$

$$\|\mathbf{x}_{\Lambda^c} - \mathbf{z}_{\Lambda^c}\|_1 \leq \|\mathbf{x}_{\Lambda^c}\|_1 + \|\mathbf{z}_{\Lambda^c}\|_1. \tag{4.177}$$

Now set $\mathbf{v} = \mathbf{z} - \mathbf{x}$. Then,

$$\|\mathbf{x}_\Lambda - \mathbf{z}_\Lambda\|_1 \leq \rho\|\mathbf{x}_{\Lambda^c} - \mathbf{z}_{\Lambda^c}\|_1 + \tau\|A(\mathbf{z}-\mathbf{x})\|_2 \tag{4.178}$$

$$\leq \rho\big(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + \|\mathbf{x}_\Lambda - \mathbf{z}_\Lambda\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1\big) + \tau\|A(\mathbf{z}-\mathbf{x})\|_2, \tag{4.179}$$

which can be rearranged to

$$(1-\rho)\|\mathbf{x}_\Lambda - \mathbf{z}_\Lambda\|_1 \leq \rho\big(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1\big) + \tau\|A(\mathbf{z}-\mathbf{x})\|_2. \tag{4.180}$$

Using the RNSP again we have

$$\|\mathbf{v}\|_1 = \|\mathbf{v}_\Lambda\|_1 + \|\mathbf{v}_{\Lambda^c}\|_1 \tag{4.181}$$

$$\leq (1+\rho)\|\mathbf{v}_{\Lambda^c}\|_1 + \tau\|A\mathbf{v}\|_2. \tag{4.182}$$

If we insert $\mathbf{v}_{\Lambda^c} = \mathbf{z}_{\Lambda^c} - \mathbf{x}_{\Lambda^c}$ into this expression we get

$$\|\mathbf{v}\|_1 \leq (1+\rho)\|\mathbf{z}_{\Lambda^c} - \mathbf{x}_{\Lambda^c}\|_1 + \tau\|A\mathbf{v}\|_2. \tag{4.183}$$

Inserting (4.175) gives

$$\|\mathbf{v}\|_1 \leq (1+\rho)\left(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + \|\mathbf{x}_\Lambda - \mathbf{z}_\Lambda\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1\right) + \tau\|A\mathbf{v}\|_2 \tag{4.184}$$

Finally, we insert (4.180) to get

$$\|\mathbf{v}\|_1 \leq (1+\rho)\left(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + \frac{1}{1-\rho}\Big(\rho\big(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1\big) + \tau\|A\mathbf{v}\|_2\Big) + 2\|\mathbf{x}_{\Lambda^c}\|_1\right) + \tau\|A\mathbf{v}\|_2 \tag{4.185}$$

$$= (1+\rho)\Big(\frac{1}{1-\rho}\left(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1\right)\Big) + \left(\frac{1+\rho}{1-\rho} + 1\right)\tau\|A\mathbf{v}\|_2 \tag{4.186}$$

$$= \frac{1+\rho}{1-\rho}(\|\mathbf{z}\|_1 - \|\mathbf{x}\|_1 + 2\|\mathbf{x}_{\Lambda^c}\|_1) + \frac{2\tau}{1-\rho}\|A(\mathbf{z}-\mathbf{x})\|_2, \tag{4.187}$$

which is what we aimed to show. □

The k-RNSP can be further generalized to arbitrary ℓ_q-norms.

Definition 4.8.4. A matrix A is said to be ℓ_q-k-RNSP if for any set Λ with $|\Lambda| \leq k$, there exists some constants $0 < \rho < 1$ and $\tau > 0$ such that

$$\|\mathbf{v}_\Lambda\|_q \leq \rho \|\mathbf{v}_{\Lambda^c}\|_1 + \tau \|A\mathbf{v}\|_2, \tag{4.188}$$

for all $\mathbf{v} \in \mathbb{C}^N$.

In this case we have the following result.

Theorem 4.8.5. Suppose A is ℓ_2-k-RNSP with respect to the ℓ_2 norm. Then, for any $\mathbf{x} \in \mathbb{C}^N$, the solution $\mathbf{x}^\#$ of (P1-σ) with tolerance η, $\mathbf{y} = A\mathbf{x}+\mathbf{e}$ and $\|\mathbf{e}\|_2 \leq \nu$ approximates the vector $\mathbf{x}$ with ℓ_p-error

$$\|\mathbf{x} - \mathbf{x}^\#\|_p \leq \frac{C}{k^{1-1/p}} \sigma(\mathbf{x})_1 + D k^{1/p-1/2} \eta, \tag{4.189}$$

for some constants C and D depending only on ρ and η.

The proof of this is very similar to that for the k-RNSP property. Note in particular that for $p = 1$ and $p = 2$ this gives

$$\|\mathbf{x} - \mathbf{x}^\#\|_1 \leq C\sigma(\mathbf{x})_1 + D\sqrt{k}\eta, \tag{4.190}$$

$$\|\mathbf{x} - \mathbf{x}^\#\|_2 \leq \frac{C}{\sqrt{k}} \sigma(\mathbf{x})_1 + D\eta. \tag{4.191}$$

We can also give performance guarantees for recovery of noisy vectors in terms of the RIP. These are related to the guarantee from the RNSP, as the following theorem shows.

Theorem 4.8.6. If A is $(2k, \delta_{2k})$-RIP, with

$$\delta_{2k} < \frac{4}{\sqrt{41}} \approx 0.6246, \tag{4.192}$$

then A is ℓ_2-k-RNSP with constants ρ and τ depending only on δ_{2k}

The proof of this theorem is fairly long, and somewhat similar to the proof of the exactly sparse RIP-recovery, so we refer to [47], p. 144-147 for the details. The constants are shown to be

$$\rho = \frac{\delta_{2k}}{\sqrt{1 - \delta_{2k}^2} - \delta_{2k}/4}, \tag{4.193}$$

$$\tau = \frac{\sqrt{1 + \delta_{2k}^2}}{\sqrt{1 - \delta_{2k}} - \delta_{2k}}. \tag{4.194}$$

The requirement $\rho < 1$ gives the particular bound on δ_{2k}. We should note that like the bound on δ_{2k} to ensure k-NSP, this bound can be improved significantly.

4.8.1 Compressible signals

A nice feature of the theory we have developed in this section is that the theoretical results place no restrictions on the measured signal $\mathbf{x}$, and for this reason the results are also valid for compressible signals. Remember that compressible signals are signals which, when ordered, decay by some power law,

$$x_j = Cj^{-1/s}. \tag{4.195}$$

In section 3.5 we showed that this gives a bound on $\sigma_k(\mathbf{x})_2$. The bounds we have developed here are mostly in terms of $\sigma_k(\mathbf{x})_1$. We therefore include the following result.

Lemma 4.8.7. If a signal $\mathbf{x}$ is compressible with parameter s, then its k-term approximation error is bounded by

$$\sigma_k(\mathbf{x})_1 \leq \frac{Cs}{1-s}k^{1-1/s}, \quad s < 1. \tag{4.196}$$

PROOF. The proof is similar to that of $\sigma_k(\mathbf{x})_2$. The main difference is that in this case $\sigma(\mathbf{x})_1$ is bounded by

$$\sigma_k(\mathbf{x})_1 \leq \sum_{j=k+1}^{N} Cj^{-1/s}. \tag{4.197}$$

We then make the same integral approximation and arrive at the above result. □
For compressible signals, we can enter this into the bounds we have found, should we want to investigate the error in a recovery.

4.9 Sensing matrices in 2-D

In this Chapter, after the introductory examples on LIPs, we have essentially treated the recovery as a problem for a 1-D signal. This is fine, as we have noted that a 2-D matrix signal can be transformed to a long 1-D signal by choosing the appropriate direct product representation. There are however some caveats we should investigate regarding CS of 2-D signals. for instance, we now that random lines from a Fourier matrix will create a good sensing matrix, but what about random lines from a 2-D Fourier matrix? How is the RIP of a Kronecker product related to the RIP of its constituents? Here we will try to answer some of these questions.

4.9.1 CS properties of Kronecker products

This section is based on results published in [82], [83] and [84]. We begin with a simple result for the mutual coherence.

Lemma 4.9.1. Let $\mathbf{\Psi}_d, \mathbf{\Phi}_d \in \mathbb{R}^{N_d}$ be orthonormal bases for $d = 1, \ldots, D$. Then

$$M(\mathbf{\Phi}_1 \otimes \mathbf{\Phi}_2 \otimes \ldots \otimes \mathbf{\Phi}_D, \mathbf{\Psi}_1 \otimes \mathbf{\Psi}_2 \otimes \ldots \otimes \mathbf{\Psi}_D) = \prod_{d=1}^{D} M(\mathbf{\Phi}_d, \mathbf{\Psi}_d). \tag{4.198}$$

PROOF. The coherence can be written as

$$M(\mathbf{\Phi}, \mathbf{\Psi}) = \|\mathbf{\Phi}^T \mathbf{\Psi}\|_\infty, \tag{4.199}$$

where $\|\cdot\|_\infty$ is the maximum norm, equal to the largest magnitude of any element in $\mathbf{\Phi}^T\mathbf{\Psi}$. The following two properties are easy to prove, and sufficient for the theorem to follow.

1. For any set of matrices $A_1, \ldots, A_D$ and $B_1, \ldots, B_D$,

$$(A_1 \otimes A_2 \otimes \ldots \otimes A_D)^T (B_1 \otimes B_2 \otimes \ldots \otimes B_D) = A_1 B_1 \otimes \ldots \otimes A_D B_D. \tag{4.200}$$

2. For any matrices $\|A \otimes B\|_\infty = \|A\|_\infty \|B\|_\infty$.

□

This is good news, as this would imply that our results from using maximally incoherent basis, such as the Fourier basis, or the Noiselet/Haar combination will carry over directly, as they have the minimal mutual coherence of $M(\mathbf{\Phi}, \mathbf{\Psi}) = 1$ in 1-D, and therefore have the same in 2-D, as well as in higher dimensions.

Next, we consider the coherence of measurements. We have the following theorem.

Theorem 4.9.2. Consider the $M_1 \times N_1$ matrix A and the $M_2 \times N_1$ matrix B, with normalized columns and coherence $\mu_1 = \mu(A)$ and $\mu_2 = \mu(B)$. The coherence of $A \otimes B$ is given by

$$\mu(A \otimes B) = \max\{\mu_1, \mu_2\}. \tag{4.201}$$

PROOF. We denote the columns of A and B by $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_{N_1}$ and $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{N_2}$. Let C be the $M \times N$ matrix $C = A \otimes B$, where $M = M_1 M_2$ and $N = N_1 N_2$, and denote the columns of C by $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N$. Note that the columns $\mathbf{c}_i$ and $\mathbf{c}_j$ are made up by $\mathbf{c}_i = \mathbf{a}_p \otimes \mathbf{b}_q$ and $\mathbf{c}_j = \mathbf{a}_r \otimes \mathbf{b}_s$. for some p, q, r, s, which implies that

$$\langle \mathbf{c}_i, \mathbf{c}_j \rangle = \langle \mathbf{a}_p \otimes \mathbf{b}_q, a_r \otimes \mathbf{b}_s \rangle = \langle \mathbf{a}_p, \mathbf{a}_r \rangle \cdot \langle \mathbf{b}_q, \mathbf{b}_s \rangle. \tag{4.202}$$

The coherence of C is then

$$\mu(C) = \max_{\substack{p,q,r,s \\ (p,q)\neq(r,s)}} \{|\langle \mathbf{a}_p, \mathbf{a}_r \rangle \langle \mathbf{b}_q, \mathbf{b}_s \rangle|\} \tag{4.203}$$

$$= \max_{\substack{p,q,r,s \\ p\neq r, q\neq s}} \{|\langle \mathbf{a}_p, \mathbf{a}_r \rangle \langle \mathbf{b}_q, \mathbf{b}_s \rangle|, |\langle \mathbf{a}_p, \mathbf{a}_r \rangle|, |\langle \mathbf{b}_q, \mathbf{b}_s \rangle|\}. \tag{4.204}$$

Because of the column normalization, $|\langle \mathbf{a}_p, \mathbf{a}_r \rangle| \leq 1$ and similarly $|\langle \mathbf{b}_q, \mathbf{b}_s \rangle| \leq 1$, which means

$$\mu(C) = \max_{\substack{p,q,r,s \\ p\neq r, q\neq s}} \{|\langle \mathbf{a}_p, \mathbf{a}_r \rangle|, |\langle \mathbf{b}_q, \mathbf{b}_s \rangle|\} \tag{4.205}$$

$$= \big\{ \max_{p\neq r} |\langle \mathbf{a}_p, \mathbf{a}_r \rangle|, \max_{q\neq s} |\langle \mathbf{b}_q, \mathbf{b}_s \rangle| \big\} \tag{4.206}$$

$$= \max\{\mu_1, \mu_2\}. \tag{4.207}$$

□

This can immediately be generalized to any number of dimensions, in the sense that if $A = A_1 \otimes \ldots, \otimes A_N$, then

$$\mu(A) = \max_{i=1,\ldots,N} \{\mu(A_i)\}. \tag{4.208}$$

This is a far more unfortunate result. Recall that for large N, the Welch bound for an $M \times N$ matrix A scales as

$$\mu(A) \approx \frac{1}{\sqrt{M}}, \tag{4.209}$$

while if A is the Kronecker product of two $\sqrt{M} \times \sqrt{N}$ matrices (so A is still $M \times N$), then the best case coherence of A scales as $1/M^{1/4}$, which is a much weaker result.

We have a similar result for the restricted isometry constant δ_k.

Theorem 4.9.3. Let $A \in \mathbb{R}^{p,q}$ and $B \in \mathbb{R}^{r,s}$ have normalized columns and restricted isometry constants δ_k^A, δ_k^B, respectively. Then,

$$\delta_k^{A\otimes B} = \delta_k^{B\otimes A} \geq \max\{\delta_k^A, \delta_k^B\}. \tag{4.210}$$

Proof. We first note that as $B \otimes A$ is simply a permutation of the columns in $A \otimes B$, it follows from

$$\delta_k^A = \max_{|\Lambda|\leq k} \text{eig}(A_\Lambda^\dagger A_\Lambda - I), \tag{4.211}$$

that a permutation of the columns in A does not change δ_k. It is therefore sufficient to show that $\delta_k^{A\otimes B} \geq \delta_k^A$.

Now assume there is some B such that $\delta_k^{A\otimes B} < \delta_k^A$. We know that δ_k^A is the smallest constant such that for all k-sparse $\mathbf{x}$, we have

$$(1 - \delta_k^A)\|\mathbf{x}\|_2^2 \leq \|A\mathbf{x}\|_2^2 \leq (1 + \delta_k^A)\|\mathbf{x}\|_2^2. \tag{4.212}$$

Now let $\mathbf{x}$ be some k-sparse vector, and let $\mathbf{y} = (1, 0, 0, \ldots, 0)^T$. The ℓ_2-norm of a Kronecker product can be evaluated as

$$\|\mathbf{x} \otimes \mathbf{y}\|_2^2 = \sum_i \sum_j x_i^2 y_j^2 = \sum_i x_i^2 \|\mathbf{y}\|_2^2 = \sum_j y_j^2 \|\mathbf{x}\|_2^2, \tag{4.213}$$

which in our case gives $\mathbf{x} \otimes \mathbf{y} = \sum_j y_j^2 \|\mathbf{x}\|_2^2 = \|\mathbf{x}\|_2^2$. Furthermore,

$$\|(A \otimes B)(\mathbf{x} \otimes \mathbf{y})\|_2^2 = \|(A\mathbf{x}) \otimes (B\mathbf{y})\|_2^2 = \|(A\mathbf{x}) \otimes \mathbf{b}_1\|_2^2 \tag{4.214}$$

where $\mathbf{b}_1$ is the first column in B. This gives

$$\|(A \otimes B)(\mathbf{x} \otimes \mathbf{y})\|_2^2 = \sum_i (\mathbf{b}_1)_i^2 \|A\mathbf{x}\|_2^2 = \|A\mathbf{x}\|, \tag{4.215}$$

as the columns of B are normalized. Since this works for any $\mathbf{x}$, we have a contradiction on the assumption $\delta_k^{A\otimes B} < \delta_k^A$. □

We also have an upper bound on $\delta_k^{A\otimes B}$.

Theorem 4.9.4. Let A and B be matrices with normalized columns and restricted isometry constants δ_k^A and δ_k^B respectively. Then

$$\delta_k^{A\otimes B} \leq (1+\delta_k^A)(1+\delta_k^B) - 1. \tag{4.216}$$

PROOF. Let A_Λ as usual denote the $M \times k$ submatrix created from selecting k columns in some way, with indices found in the set Λ. Then all the eigenvalues of $A_\Lambda^\dagger A_\Lambda$ lie in the interval $(1-\delta_k^A, 1+\delta_k^A)$, and similarly for B, all the eigenvalues of $B_\Lambda^\dagger B_\Lambda$ lie in the interval $(1-\delta_k^B, 1+\delta_k^B)$. Now let the eigenvectors of A and B are given as $\mathbf{v}_1, \ldots, \mathbf{v}_{N_A}$ and $\mathbf{w}_1, \ldots, \mathbf{w}_{N_B}$, with eigenvalues $\lambda_1^A, \ldots, \lambda_{N_A}^A$ and $\lambda_1^B, \ldots, \lambda_{N_B}^B$, respectively. Then the eigenvectors of $A \otimes B$ are given by $\mathbf{v}_i \otimes \mathbf{w}_j$ with eigenvalues $\lambda_i^A \lambda_j^B$.

Let now Λ be an index set with $|\Lambda| \leq$. The submatrix $(A \times B)_\Lambda$ can not be simply constructed from a direct product of submatrices of A and B. However, it is easy to check that we can build index sets Λ_1, Λ_2 with $|\Lambda_i| \leq k$ such that $(A \times B)_\Lambda$ is a submatrix of $A_{\Lambda_1} \otimes B_{\Lambda_2}$. The Gramian of this matrix will have eigenvalues in the range $\big((1-\delta_k^A)(1-\delta_k^B), (1+\delta_k^A)(1+\delta_k^B)\big)$. Furthermore, it is possible to show [85] that the eigenvalues of the Gramian of a submatrix are nested within the range of the eigenvalues of the Gramian of the full matrix. This shows that the eigenvalues of the Gramian of $(A \times B)_\Lambda$ lie in the range $\big((1-\delta_k^A)(1-\delta_k^B), (1+\delta_k^A)(1+\delta_k^B)\big)$, which is what we wanted to prove. □

This bound is once again not very strong. If A and B both have restricted isometry constants $\delta_k^A = \delta_k^B = 1/2$, then the bound only guarantees that the restricted isometry property of $A \otimes B$ is bounded by $5/4$, not enough for any reconstruction guarantee. To understand why these guarantees are so weak, we will consider the random sensing approach in 2-D.

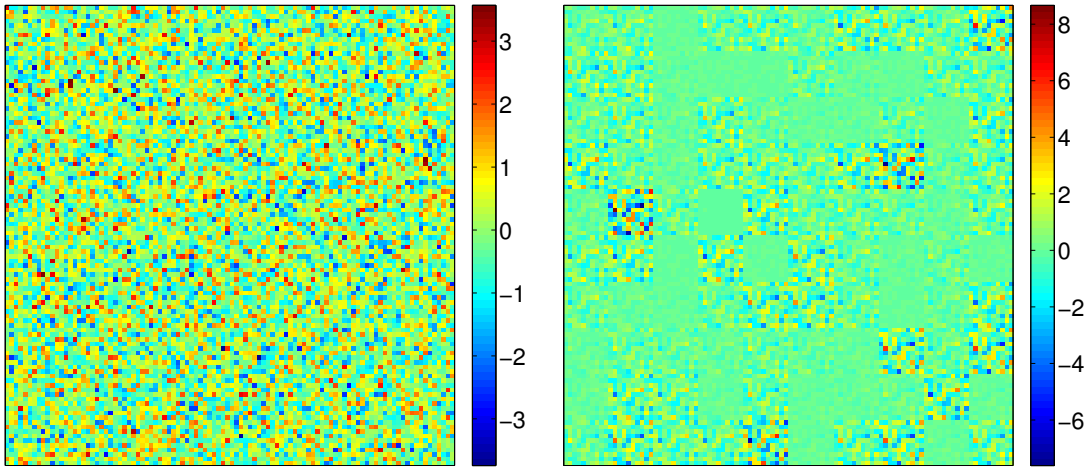
4.9.2 Random Sensing in 2-D

One of the best sources for generating good sensing matrices turned out to be Gaussian independently random elements. Figure 4.19 shows a 100×100 Gaussian random matrix, as well as the Kronecker product of two 10×10 Gaussian random matrices. While the last one still has random elements, they are clearly not independent, and in fact they are not even Gaussian. This all shows us that some care must be taken when using Kronecker products to measure a signal. We could of course avoid this problem by creating M Gaussian random matrices and measure the signal for each of them, but for large datasets, this is not feasible in practice.

Not much theory exist on to what degree results from 1-D are transferable to 2-D. Notably, [86] gives a bound for restoration using the Kronecker product of two sub-Gaussian matrices, however, this bound does not beat the quadratic bottleneck (it requires $M \geq \mathcal{O}(k^2)$), which leaves it somewhat unimpressive.

We have seen that incoherent discrete orthonormal systems works fairly well for constructing sensing matrices. We have also seen that the mutual coherence framework was highly transferable to 2-D. This might motivate the following idea: If we can create orthonormal random matrices which have low coherence with the canonical basis, they may have low coherence with other bases as well, and they may be suitable for reconstruction.

The following theorem, from [44] helps us get started.



(a) Gaussian random 100×100 matrix

(b) Kronecker product of two 10×10 Gaussian random matrices

Figure 4.19: The figure shows a Gaussian random matrix and the Kronecker product of two Gaussian random matrices.

Theorem 4.9.5. Let A denote a random real orthogonal matrix, uniformly distributed on the unit sphere. Then the exceedance probability,

$$P_{\epsilon,N} = \mathbb{P}\left(\max_{ij} |U_{ij}| > 2\sqrt{\log(N)/N}(1+\epsilon)\right), \tag{4.217}$$

obeys $P_{\epsilon,N} \to 0$ as $N \to \infty$.

In other words, for large N, the mutual coherence of A with the identity matrix I_N, will with large probability not be much larger than $2\sqrt{\log N}$. The cited articles also argues that this result can be extended to any orthogonal basis (rather than just I_N).

When we compare the to theorem 4.7.11, which gives a bound on M for nonuniform recovery for a general BOS, and we note that in this case, the boundedness constant with large probability can be set to $K = \sqrt{2\log N}$, we immediately get the following result.

Theorem 4.9.6. Let A be a matrix constructed by orthonormalizing M vectors selected from a uniform distribution on the N-dimensional unit sphere. If N is large, then a fixed k-sparse vector can be reconstructed via ℓ_1-minimization with probability higher than $1-\epsilon$ provided that

$$M \geq 4Ck\ln^3(N)\ln(\epsilon^{-1}), \tag{4.218}$$

where C is the same constant as in theorem 4.7.11.

The good news on this result is that it also extends to 2-D. Let $C = A \otimes B$ be the Kronecker product of two random orthogonal $n \times n$ matrices, where $n^2 = N$. The largest elements of the direct product of two $n \times n$ random orthogonal matrices will be the product of the largest

elements in each of the two smaller matrices. If we now assume that n is large, then the largest magnitude of $A \otimes B$ will high probability be bounded by $4\log(n)/n = 2\log(N)/\sqrt{N}$. This gives us the following corollary.

Corollary 4.9.7. Let $A = R_T(B \otimes C)$, where B and C are $n \times n$ matrices constructed by orthnormalizing n vectors selected from a uniform distribution on the n-dimensional unit sphere, and R_T is a random subsampling matrix selecting M rows uniformly at random from the $N \times N$ matrix $(B \otimes C)$. If n is large, then a fixed k-sparse vector can be reconstructed via ℓ_1-minimization from the measurements $\mathbf{y} = A\mathbf{x}$ with probability higher than $1 - \epsilon$ provided that

$$M \geq 4Ck \ln^5(N) \ln(\epsilon^{-1}), \tag{4.219}$$

where C is the same constant as in theorem 4.7.11.

This result beats the quadratic bottleneck, which makes it noteworthy. We could also recreate the results for uniform recovery for these matrix constructions. Note that the result does not really require B and C to be distinct, so we can let $B = C$ in order to conserve memory. This result is not found elsewhere in literature, so we will explore it here numerically.

Figure 4.20 shows several examples of a common numerical analysis technique called *phase plots*. The idea is to numerically investigate the recovery properties of a matrix structure by creating several test vectors for various scenarios (i.e. values of M and k), and recording the result of reconstructions as either successes or failures.

The following four experiments were carried out. The length of the signal in each experiment was set to $N = 400$.

1. First, I simply let A consist of Gaussian i.i.d. elements. The result of this is shown in figure 4.20(a), and are good, as is to be expected.

2. Then, I let $A = Q_1 \otimes Q_2$, where Q_1 and Q_2 were matrices with Gaussian i.i.d. elements. The result of this is shown in figure 4.20(b) and is very poor. Again, this is to be expected, based on our discussion in this section.

3. Then, I let $A = R_T(\text{orth}(Q))$, where Q consists of N uniformly random vectors on the unit sphere, orth refers to the operation of orhogonalizing Q, and R_T is a random sub-selection operator, selecting M rows at random. In order to create Q, I used the method noted in [87], which consist of letting the elements of Q be Gaussian i.i.d. elements, and normalizing each row in Q. The rows were then orthogonalized using the Gram-Schmidt method. The result of this simulation is shown in figure 4.20(c).

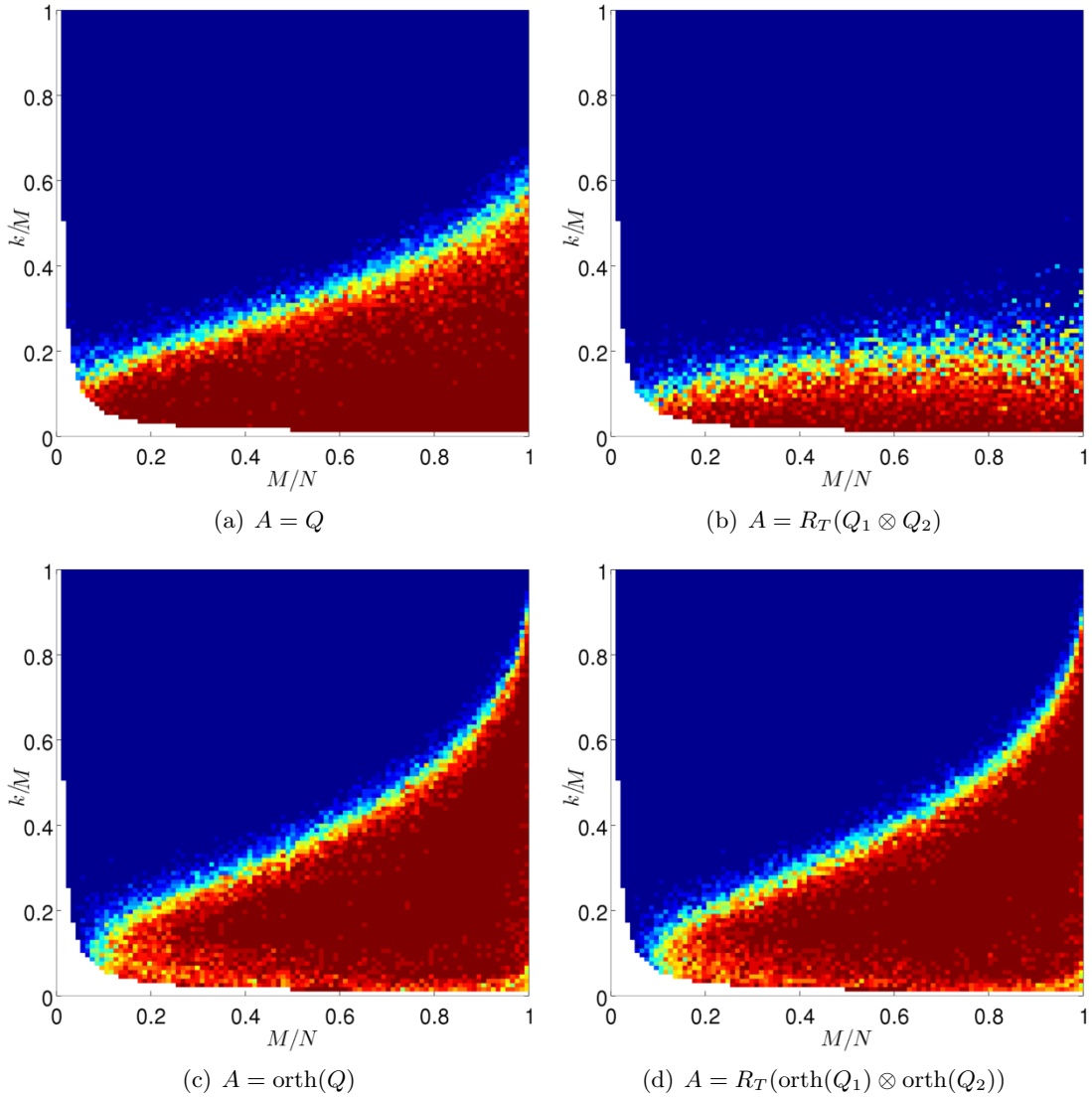
4. Finally, I let $A = R_T(\text{orth}\, Q_1 \otimes \text{orth}\, Q_2)$, where Q_1 and Q_2 where $n \times n$ matrices, with $n = \sqrt{N}$, created as in the previous case. The result of this method is shown in 4.20(d), and almost identical to the third experiment, which can be expected, given that they share their dependence on k.

The whited out areas of each plot corresponds to areas where k was set to 0, because of our limit on N. These will naturally give a perfect reconstruction, as it is very simple to restore the zero vector.

Surprisingly, the third and fourth methods give better results than the first experiment, even though the theoretical result for the first case is better. This might be because the numerical

ℓ_1-minimizer used (spgl1) prefers orthogonal measurements, but I have not looked into finding a definite answer for this effect. The area which this method outperforms the first methods are in reality not very interesting, as they correspond to cases where the measurement matrix is almost of full rank.

One tendency that is worth noting, is that for we are not able to achieve 100% restoration for small values of k/M for the final two methods. This effect is highly surprising, and at the time of writing, I do not have a good explanation for them.



(a) $A = Q$ (b) $A = R_T(Q_1 \otimes Q_2)$

(c) $A = \text{orth}(Q)$ (d) $A = R_T(\text{orth}(Q_1) \otimes \text{orth}(Q_2))$

Figure 4.20: The figures show phase plots for various random matrices when used in restoration. Dark red corresponds to 100% success, while dark blue corresponds to 100% failures. The white areas of the plots indicates areas where the experiment parameters made $\mathbf{x}$ 0-sparse, i.e. $\mathbf{x} = 0$. The length of the signal was set to $N = 400$.

Number of samples	Transform	PSNR
$N/4$	zero-filling	8.20
	Haar	31.81
	CDF 9/7	35.1
$N/16$	zero-filling	6.25
	Haar	23.46
	CDF 9/7	26.85

Table 4.4: The table shows the results of a CS restoration of missing information using partial random noiselet measurements.

4.10 Examples on the Lena image

In this section we will once again turn to the Lena image to explore some of the results we have found. First, let us see if we can improve the on the inpainting results of the Haar and CDF 9/7 wavelets by sampling the image in the proper basis.

4.10.1 Wavelet reconstruction based on partial noiselet samples

Because the wavelet bases and the canonical basis are highly coherent, the inpainting results for wavelets in section 4.1.5 were poor. Since then, we have introduced the noiselet basis, designed to be maximally incoherent with the Haar wavelet basis. While noiselets are not maximally incoherent with the CDF 9/7 wavelets, they are still fairly incoherent, and we include the results also for these wavelets. The results are shown in figures 4.21 and 4.22, and the errors are summarized in table 4.4. It is worth noting that the CDF 9/7 wavelets give better results here that both the DFT and the DCUT2 did in the section on inpainting.

4.10.2 Image reconstruction based on random sensing

Here we use the orthogonal random sensing matrices discussed in section 4.9.2. We will attempt to restore the image in the Fourier, Haar, CDF 9/7, DCUT2 and Wave Atom basis. In this case, the sampling should be equally useful for any of the bases, which makes this a good indicator as to which basis is the best (sparsest) one to represent the Lena image. We should keep in mind that only two experiments are performed, so we can not make any definitive claims from our results.

Throughout, an image X of size $N = n \times n$ where $n = 512$ pixels were used. We will denote the vectorization of X by $\mathbf{x}$. Two orthogonal matrices Q_1 and Q_2 were created. The direct product of these was then applied to the image X as $\mathbf{b} = Q\mathbf{x} = \text{vec}(Q_1 X Q_2^T)$, where $\text{vec}(A)$ is the vectorization of a matrix A. M samples were then collected at random from $\mathbf{b}$ by a random sampling operator R, $\mathbf{y} = R\mathbf{b}$. We then aimed to find the signal $\mathbf{x}$ such that

$$\mathbf{y} = RQ\mathbf{x}, \tag{4.220}$$

and $\mathbf{\Psi x}$ is as sparse as possible, where $\mathbf{\Psi}$ is the matrix corresponding to the elements of the different bases. Alternatively, if $\mathbf{z} = \mathbf{\Psi x}$ such that $\mathbf{x} = \mathbf{\Psi}^{-1}\mathbf{z}$, we can formulate this as

$$\begin{aligned} &\text{minimize} && \|\mathbf{z}\|_0, \\ &\text{subject to} && RQ\mathbf{\Psi}^{-1}\mathbf{z} = \mathbf{y}, \end{aligned} \tag{4.221}$$

Number of measurements	Transform	PSNR
$N/4$	zero-filling	6.95
	DFT	28.01
	Haar	27.86
	CDF 9/7	31.30
	DCUT2	31.68
	Wave Atom	30.95
$N/16$	zero-filling	5.96
	DFT	22.61
	Haar	21.54
	CDF 9/7	24.03
	DCUT2	25.04
	Wave Atom	23.66

Table 4.5: The table shows the result of CS restorations on random orthogonal samples of the Lena image.

which is really the standard ℓ_1-minimization problem we have explored in great detail. The results for our experiment is shown in 4.5, and the restoration for $M = N/16$ is shown for the various transforms in figure 4.23. It is interesting to note the different artifacts from the various transforms.

4.11 A note on performance guarantees

While the RIP is considered a "holy grail" in terms of recovery guarantees, one should always keep in mind that the RIP gives a sufficient, but not necessary condition for reconstruction. We have already noted that for partial Fourier measurements, we are able to show stronger results for nonuniform recovery than for uniform. In section 5.1 we will see that one of the most prominent used of CS, efficient MRI imaging, gives impressive result despite offering no guarantees in terms of the RIP.

A recent paper by Adcock et al. [88] note this gap between theory and practice in CS. In this paper, they note that the recovery property of signals is highly dependent on the the structure of the signal, not just the sparsity. In this paper, they suggest a new version of sparsity, called *sparsity in levels*, which also puts bounds on what level of an MRA the non-zero coefficients of a signal $\mathbf{x}$ may be in. Here, we simply note that even though we may not be able to guarantee reconstruction in terms of the RIP or even the k-NSP, we should not be deterred from trying to restore signals we are interested in.

4.12 Implementation

So far we have discussed to great length when we are able to restore a signal by ℓ_1-minimization, but we have not said anything about how we should perform this minimization. Now we finally add some details on how to perform this minimization. The details and examples will be MATLAB-specific, but they are easily transferable to other languages.

4.12.1 Setting up a numerical experiment

Let us first note how to set up a test case for compressed sensing in MATLAB. We should define the signal length N as well as the sparsity level k, or alternatively the compression coefficient s. After this is done, we randomly spread the sparse or compressible coefficients randomly throughout the signal `x`. For this, two useful MATLAB functions are `randsample(N,k)`, which draws k numbers at random without replacement from the set $\mathbb{N}_1^N$, and `randperm(N)`, which performs a random permutation of the set $\mathbb{N}_1^N$.

The following code will generate an exactly sparse vector,

```
function [x, idx_nonzero] = generate_sparse_vector(N,k)
    x = zeros(N,1);
    idx_nonzero = randsample(N,k);
    x(idx_nonzero) = rand(k,1)
end
```

while the next one will generate a compressible signal.

```
function x = generate_compressible_vector(N,s)
    x = (1:N).^(-1/s);
    x = x(randperm(N));
end
```

Note that these signals will be strictly positive, but it is a simple procedure to switch some of the signs in the signal if we do not want this feature.

Once this signal has been generated we generate some sensing matrix. Generating a gaussian random matrix is very simple, we simply use the MATLAB function `randn(M,N)` which generates an $M \times N$ matrix with i.i.d. random Gaussian variables with standard deviation 1. We can also use the function `rand(M,N)` to help us construct a Bernoulli matrix. Finally, we can use the function `dftmtx(N)` which generates the $N \times N$ Fourier matrix, to generate a partial Fourier matrix. The following code generates the different matrices.

```
% Gaussian matrix:
A = randn(M,N);

% Bernoulli Matrix:
B = 2*(rand(M,N) < 0.5)-1

% Fourier Matrix:
C = dftmtx(N);
idx_us = randsample(N,M);
C = C(idx_us,:);
```

Once this matrix has been created, we simply measure the signal as `y=A*x`. Now, the task is to restore the signal.

4.12.2 ℓ_1-minimization

Here we will describe in very simple terms how to construct an ℓ_1-minimizer with equality and inequality constraints. To start with, we dismiss these constraints in order to simplify our discussion.

Unconstrained minimization

We want to minimize some function $f(\mathbf{x}) : \mathbb{R}^N \to \mathbb{R}$. We first introduce some useful concepts.

Definition 4.12.1. Let $f : \mathbb{R}^N \to \mathbb{R}$ be a continuous function. We denote the *gradient* of f by

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_N} \right)^T. \tag{4.222}$$

Furthermore, we denote the *Hessian* of f by

$$H_f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N^2} \end{pmatrix}. \tag{4.223}$$

We now have the following, which should be known, but is repeated for convenience.

Theorem 4.12.2. Assume that $f : \mathbb{R}^N \to \mathbb{R}$ has continuous partial derivatives, and that $\mathbf{x}^*$ is a local minimum of f. Then

$$\nabla f(\mathbf{x}^*) = 0. \tag{4.224}$$

Furthermore, if f has continuous partial derivatives, then $H_f(\mathbf{x}^*)$ is positive semidefinite.

The proof of this is straight forward, but widely available in textbooks, so it is omitted here. Furthermore, note that the only function we want to minimize, $f(\mathbf{x}) = \|\mathbf{x}\|_1$, does *not* have continuous derivatives. They are discontinuous at $x_i = 0$. Still, these are terms which should be a part of any discussion on function optimization. One redeeming quality of $\|\mathbf{x}\|_1$ is that it is convex. Let us define this property.

Definition 4.12.3. A function $f(\mathbf{x})$ is said to be convex if for any two points $\mathbf{x}_1$ and $\mathbf{x}_2$, and any $\lambda \in [0, 1]$

$$f(\lambda \mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2) \leq \lambda f(\mathbf{x}_1) + (1 - \lambda) f(\mathbf{x}_2), \tag{4.225}$$

If $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}$, this can be interpreted as saying that the line through the points $(x_1, f(x_1))$ and $(x_2, f(x_2))$ will lie above the function.

This makes the problem much more approachable, as it can be shown that if f is convex, then a local minimum of f is a global minimum.

Any numerical minimization algorithm will begin with some first guess $\mathbf{x}_0$ for the minimum. From here there are two main categories of methods.

1. *Line search methods*: Here we chose a direction $\mathbf{d}_k$ from the current point $\mathbf{x}_k$, and move according to
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k, \tag{4.226}$$
for some choice of α_k.

2. *Trust region methods*: Here one chooses an approximation $\hat{f}$ to f in some region around $\mathbf{x}_k$, and then minimizes $\hat{f}$ in this area. The point of minimization is then the next point of the iteration $\mathbf{x}_{k+1}$.

Here we will focus on line search methods. A simple example of such a method is the *steepest descent method*, where one moves in the direction where the function falls the fastest accoring to the gradient. This corresponds to setting
$$\mathbf{d}_k = -\nabla f(\mathbf{x}_k), \tag{4.227}$$
for each k. This is a very stable method, but it is not very fast, so other methods have been developed.

Notably, *Newton's method* offers faster convergence. Here, one moves according to
$$\mathbf{d}_k = -\alpha_k H_f(\mathbf{x}_k)^{-1} \nabla f(\mathbf{x}_k). \tag{4.228}$$
While the steepest descent can be thought of as approximating f with a linear function, Newton's method can be thought of as approximating f with a quadratic function. If we set $\alpha_k = 1$ we arrive at the classical Newton's method, which will find $\mathbf{x}_{k+1}$ as the minimum for the quadratic approximation around $\mathbf{x}_k$.

The steepest descent method has a linear convergence, while Newton's method has a quadratic convergence. Roughly speaking, this means that while we may expect each iteration of the steepest descent method to give us one more correct leading digit to the minimum, we can for Newton's method expect the number of correct leading digits to double each iteration. One should note however that the proof of the quadratic convergence of Newton's method relies on f having continuous partial derivatives, so these convergence properties may not hold for $f(\mathbf{x}) = \|\mathbf{x}\|_1$.

Equality constraints

We now look to solve the problem
$$\begin{aligned} &\text{minimize} && f(\mathbf{x}), \\ &\text{subject to} && h_i(\mathbf{x}) = 0, \quad (i = 1, \dots, M). \end{aligned} \tag{4.229}$$
In order to solve this we introduce the Lagrangian function, $L : \mathbb{R}^N \times \mathbb{R}^M$ defined by
$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i=1}^{M} \lambda_i h_i(\mathbf{x}) = f(\mathbf{x}) + \boldsymbol{\lambda}^T \mathbf{h}(\mathbf{x}), \tag{4.230}$$
where $\mathbf{h}(\mathbf{x}) = (h_1(\mathbf{x}), \dots, h_M(\mathbf{x}))$. One can show that the minima of f obeying $h_i = 0$ satisfy
$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = 0, \tag{4.231}$$
$$\nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = 0, \tag{4.232}$$

where $\nabla_{\mathbf{x}}$ is the normal gradient, and $\nabla_{\boldsymbol{\lambda}}$ is the gradient with regards to the variables λ_i. Again we refer to any standard textbook on multi variable calculus for the proof. The second equation here simply states that $\mathbf{h}(\mathbf{x}^*) = 0$.

Now we consider the problem of optimizing $f(\mathbf{x})$ under the constraint $A\mathbf{x} = \mathbf{b}$. We can use Newton's method for this problem as well, with two small modifications. First, we need to make sure that the initial point is feasible, i.e. $A\mathbf{x}_0 = \mathbf{b}$. We then need to make sure that the next point is feasible as well. This holds if $A\mathbf{d}_k = 0$, i.e. $\mathbf{d}_k \in \ker(A)$.

The Lagrangian for this constrained optimization, with $\mathbf{x} = \mathbf{x}_k + \mathbf{d}_k$ is

$$L(\mathbf{x}, \boldsymbol{\lambda}) = \nabla f(\mathbf{x}_k) + H_f(\mathbf{x}_k)\mathbf{d}_k + A^T\boldsymbol{\lambda}. \tag{4.233}$$

All in all, we need to solve the linear system

$$\begin{pmatrix} H_f(\mathbf{x}_k) & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} \mathbf{d}_k \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} -\nabla f(\mathbf{x}_k) \\ 0 \end{pmatrix}. \tag{4.234}$$

Inequality constraints

When we would rather solve (P1-σ), the relaxed optimization problem, then we are are looking at a problem of the form

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}), \\ &\text{subject to} && g_i(\mathbf{x}) \leq 0, \quad (i = 1, \ldots, R). \end{aligned} \tag{4.235}$$

In our case, the only inequality would be $\|A\mathbf{x} - \mathbf{b}\|_2 - \sigma \leq 0$.

A simple way to solve this is by the *interior-point barrier* method. We replace the original problem by the problem

$$\text{minimize} \quad f(\mathbf{x}) + \mu\phi(\mathbf{x}), \tag{4.236}$$

where

$$\phi(\mathbf{x}) = -\sum_{j=1}^{R} \ln(-g_j(\mathbf{x})), \tag{4.237}$$

and $\mu > 0$ is some parameter. The function ϕ is called the logarithmic barrier function, as it ensures that $f(\mathbf{x}) + \mu\phi(\mathbf{x}) \to \infty$ as $g_j(\mathbf{x}) \to 0$, effectively creating a barrier for the feasible solutions. Furthermore, one can show that if $\mathbf{x}^*$ is the solution to (4.235), and $\mathbf{x}(\mu)$ is the solution to (4.236), then $\mathbf{x}(\mu) \to \mathbf{x}^*$ as $\mu \to 0$. Typically, we modify our program by adding an external loop where we first choose some μ and solve (4.236), then decrease μ and use the previous solution as the starting point for the new iteration.

Minimization software

We have described in a minimalistic way. There are far more sophisticated ℓ_1-minimizers available. Here we will describe the ones used in this thesis.

1. **ℓ_1-MAGIC**. This was one of the first packages made specifically for solving compressed sensing problems. It uses a Newton method with an interior-point logarithmic barrier method, much like what we have described here. It can solve most of the problems described in section 4.1.2. Drawbacks of this package are somewhat poor performance compared to other tools on large systems and fast implementations of operators (see 4.12.3 for details). Details on this package can be found in [89]

2. **SPGL1**. This more sophisticated solver is based on the paper [90], where they show that the there is a curve parameterized by a single parameter that traces the optimal trade-off between the ℓ_2 error $\|A\mathbf{x} - \mathbf{b}\|_2$ and the ℓ_1-norm of $\mathbf{x}$, which they refer to as the Pareto Frontier. They further show that this curve is convex and continuously differentiable. The package attempts to find the best solution along this curve. Details on the package can be found in [91]. This package works far better for large systems than ℓ_1-MAGIC, and it also works well with fast implementations of operators.

3. **Sparco**. Sparco is not a minimization software, rather it is a wrapper which makes it easy to implement several transforms and useful tools for any ℓ_1-minimizer. It includes several operators, such as the curvelet, Fourier and Haar transforms, as well as a framework which allows user to easily add more operators.

4. **MCALab/SplittingSolvers** This package is more of a complete software for doing CS experiments than simply an ℓ_1-solver. MCALab was mentioned as a tool for inpainting in section 4.1.5. SplittingSolvers expands on this package to allow for sampling in bases other that the canonical one. The package is fast, and notably has built-in support for many exotic operators such as the curvelet and the Wave Atom bases. The package can be downloaded from http://www.multiresolutions.com/sparsesignalrecipes/software.html. For documentation of the package, refer to the documentation of MCALab [43].

All the tests performed in this thesis are performed using either MCALab or spgl1.

4.12.3 Efficient implementation of fast operators

When describing how to set up a numerical example in section 4.12.1, we set up a partial Fourier transform by first constructing the whole Fourier matrix and then removing rows from it, leaving us with an $M \times N$ sensing matrix A. Calculating $A\mathbf{x}$ and $A^\dagger\mathbf{b}$ will be required at each step in the iteration. calculating $A\mathbf{x}$ as explained so far, will require $\mathcal{O}(MN)$ operations. For most applications, it would be preferable if we were able to use some efficient implementation such as the FFT, which requires $\mathcal{O}(N \log_2 N)$ operations. However, the partial Fourier matrix cannot be split in the same way that the full Fourier matrix can, as the frequencies are not equally spaced. Here we show another approach which allows us to implement these fast operators.

The idea is that rather than selecting the M relevant rows from F_N as $A = R_T F_N$, where F_N is the DFT matrix and R_T is the row selection matrix, we consider a scheme where we instead compute $F_N\mathbf{x}$, and then select the appropriate elements of $\mathbf{x}$ (the elements with the same indices as the kept rows in A). This will clearly produce the same results, as it can be interpreted as making N measurements on the signal, and then trowing away $N-M$ of them. If we use a regular matrix, then this is inefficient, as if requires $\mathcal{O}(N^2)$ operations, however, since we now use a complete Fourier matrix, we can replace this with an FFT function, effectively reducing the order to $\mathcal{O}(N \log_2 N)$. When computing $A^T\mathbf{b}$, we can rather place $\mathbf{b}$ into a larger vector $\mathbf{b}^N$ of length N such that if $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_M\}$ is the index set of the rows of the DFT kept in A, then $b_i^N = b_{\lambda_i}$. If we then compute $F_N^\dagger \mathbf{b}^N$, this gives the same result as $F_N^\dagger \mathbf{b}$. Once again, this allows us to use fast operators such as the IFFT.

As an example, we may write the partial IFFT function as

```
function [ y ] = partialFFTImpl( idx, N, x, mode )
    if mode==1
        y = fft(x)*sqrt(N);
        y = y(idx);
    else
        y = zeros(N,1);
        y(idx) = x;
        y = ifft(y)/sqrt(N);
    end
end
```

which uses the built-in MATLAB FFT and IFFT functions (note that these are normalized differently from the version presented here, which is the reason for the factors $\sqrt{N}$). Here, `idx` contains the indices in Λ, `N` is the size of the signal, `mode` defined whether to compute A or A^T, and `x` is either $\mathbf{x}$ or $\mathbf{b}$, depending on `mode` (it is the vector we would like to multiply with either A or A^T).

Going even further, since $\mathbf{x}$ will typically be sparse, there are much work presently going on to find even more efficient Fourier transforms for sparse data, often referred to as Sparse Fourier Transforms. Notably, Hassanieh et al. recently published an algorithm computing the Fourier transform of an exactly k-sparse vector in $\mathcal{O}(k \log_2 N)$ iterations [92].

Implementation of 2-D signals

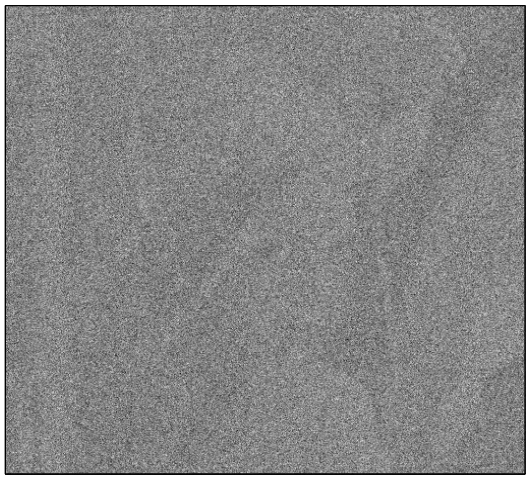
Most packages will require $\mathbf{x}$ to be a vector. We have seen that an image X can be represented as a long vector $\mathbf{x}$, but computing the operation $(A \otimes^k B)\mathbf{x}$ will computationally be much more taxing than computing $(A \otimes B)X = AXB^T$. We can solve this problem in the same way we did for the FFT. For an image X represented by a vector $\mathbf{x}$, we want to compute $A\mathbf{x}$, where A is $M \times N$ matrix constructed from selecting M rows from $A \otimes^k B$ with indices in the set Λ. We can do this by first turning $\mathbf{x}$ into a matrix X, then calculating $Y = AXB^T$, then turn Y into a vector $\mathbf{y}$, and finally remove all the elements with indices not in Λ. We can naturally apply fast operators such as the 2-D FFT in the same way.

(a) Zero filling

(b) Haar basis

(c) CDF 9/7 basis

Figure 4.21: The figure shows reconstructions based on partial noiselet samples. The reconstructions are made in the Haar and CDF 9/7 bases with $M = N/4$ measurements.

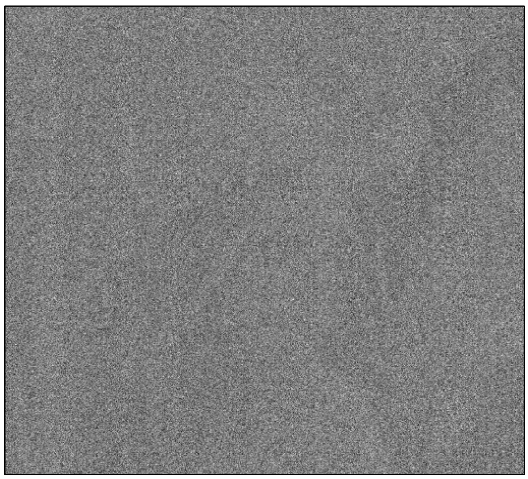


(a) Zero filling

(b) Haar basis

(c) CDF 9/7 basis

Figure 4.22: The figure shows reconstructions based on partial noiselet samples. The reconstructions are made in the Haar and CDF 9/7 bases with $M = N/16$ measurements.



(a) Zero filling

(b) Haar basis

(c) DFT basis

(d) CDF 9/7 basis

(e) Curvelet basis

(f) Wave Atom basis

Figure 4.23: The figure shows reconstructions based on partial random samples. The reconstructions are made with $M = N/16$ measurements.

Chapter 5

Applications of Compressed Sensing in Computational Physics

Now that we have developed the theory we need to be operative in CS, it is finally time to highlight its uses in computational physics. This chapter will be split in two parts; Sections 5.1, 5.2, 5.3 and 5.4 highlight notable work already made in Physics with CS, explaining the work and its significance. Sections 5.5, 5.6, 5.7 and 5.8 represent original work.

5.1 Compressed sensing MRI

Compressed sensing saw its first applications in medical imaging, so it is natural to start our discussion on applications here. We here detail the basics of Magnetic Resonance Imaging (MRI), and explain how CS can improve the MRI quality.

The MRI signal is generated by protons in the body, mostly those in water molecules. A strong static field B_0 polarizes the protons, yielding a net magnetic moment oriented in the direction of the static field. A radio frequency applied to the protons will "flip" the magnetic moment to the opposite direction. This flip happens at a characteristic frequency

$$f_0 = \frac{\gamma}{2\pi} B_0 \tag{5.1}$$

where $\gamma/2\pi$ is a constant. The transverse component of the precessing magnetization produces a signal detectable by the receiver coil. The transverse magnetization at a position r is represented by the complex quality $m(r) = |m(r)|e^{-i\phi(t)}$ where $|m(r)|$ is the magnitude of the transverse magnetization, and $\phi(t)$ is its phase. The signal of interest is $m(r)$, which may represent many different physical properties of the tissue. A common property is the proton density.

In order to find $m(r)$, we need the spatial resolution of the image. In order to get this, we apply a gradient field of the form $B(x) = B_0 + G_x x$, where G_x is some constant. This gives a characteristic frequency

$$f(x) = \frac{\gamma}{2\pi}(B_0 + G_x x), \tag{5.2}$$

This gives different frequencies for different points in space, giving us the spatial resolution. More generally, in 2-D or 3-D, the additional frequency contributed by gradient fields can be written as

$$f(r) = \frac{\gamma}{2\pi}\mathbf{G}(t) \cdot \mathbf{r}, \tag{5.3}$$

where $\mathbf{G}(t)$ is a vector of the gradient field amplitudes. The phase of the magnetization is the integral of frequency starting from time zero (immediately following the RF excitation):

$$\phi(\mathbf{r}, t) = 2\pi \int_0^t \frac{\gamma}{2\pi} \mathbf{G}(s) \cdot \mathbf{r}\, ds \tag{5.4}$$

$$= 2\pi \mathbf{r} \cdot \mathbf{k}(t), \tag{5.5}$$

where

$$\mathbf{k}(t) = \frac{\gamma}{2\pi} \int_0^t \mathbf{G}(s)\, ds. \tag{5.6}$$

The receiver coil integrates over the entire volume, producing a signal

$$s(t) = \int_R m(\mathbf{r}) e^{-2\pi i \mathbf{k}(t) \cdot \mathbf{r}}\, d\mathbf{r}. \tag{5.7}$$

This is the *signal equation for MRI*. In other words, the received signal at time t if the Fourier transform of the object $m(\mathbf{r})$ sampled at the spatial frequency $k(t)$.

Magnetization decays exponentially with time. This limits the useful acquisition time window. Also, the gradient system performance and physiological constraints limit the speed at which k-space can be traversed. These two effects combine to limit the total number of samples per acquisition. As a result, most MRI imaging methods use a sequence of acquisitions; each one samples part of k-space. The data from this sequence of acquisitions is then used to reconstruct an image.

Let us look at a toy model for how this sampling works in practice. Figure 5.1(a) shows the Shepp-Logan phantom, a simple model for MRI data. Figure 5.1(b) shows a typical sampling pattern in the frequency space. Note that this representation has the low frequencies near the center of the image, and the high frequencies near the edge. Figure 5.1(c) shows the zero-filling reconstruction, i.e. the simple IDFT with zeros padded for the missing samples. This is the traditional reconstruction method, and for this reason, a high number of radial lines are needed for a reconstruction.

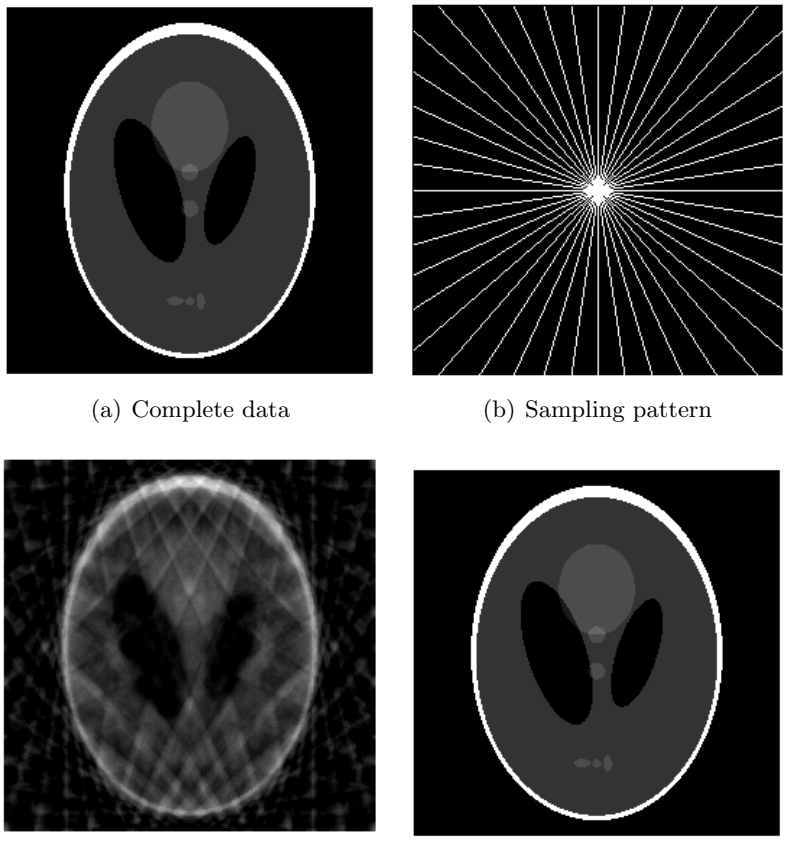
Note however, that the image has an extremely sparse gradient. This inspires us to attempt to restore the image using TV-minimization. This restoration is shown in figure 5.1(d), and is in fact exact.

Note that we have not made any guarantees in terms of the RIP or NSP in this case. Still, the quality of the result speaks volumes of the potential of this method. We are also free to try other bases such as the DCUT2 or a wavelet basis. See [2] for a complete investigation of CS-based MRI.

In 2009, CS MRI was implemented at Lucile Packard Childrens Hospital as part of a two-year test program. Reports concluded that they were able to speed up the MRI process by a factor of 6, allowing faster treatment times and increased capacity [93].

5.2 Compressed Sensing in Astronomy

The Herschel/Photodetector Array Camera and Spectrometer mission of the European Space Agency is faced with a strenuous compression dilemma: it needs a compression rate equal to $= 1/P$ with $P = 6$. A first approach has been proposed, which consists of averaging $P = 6$ consecutive images of a raster scan and transmitting the final average image. Nevertheless,



(a) Complete data (b) Sampling pattern

(c) Zero-filling reconstruction (d) TV reconstruction (exact)

Figure 5.1: Reconstruction of the Shepp-Logan phantom based on partial Fourier samples. The figure is from a CS demonstration available at http://www.cs.tut.fi/~comsens/

doing so with high-speed raster scanning leads to a dramatic loss in resolution. We emphasize the redundancy of raster scan data: 2 consecutive images are almost the same images up to a small shift $\mathbf{t} = (t_1, t_2)$ (t_1 and t_2 are the translations along each direction). Then, jointly compressing/decompressing consecutive images of the same raster scan has been proposed to alleviate the Herschel/PACS compression dilemma. The problem consists of recovering a single image $\mathbf{x}$ from P compressed and shifted noisy versions of it:

$$\mathbf{x}_j = S_{\mathbf{t}_j}\mathbf{x} + \eta_j, \quad j = 1, \ldots, P, \tag{5.8}$$

where $S_{\mathbf{t}_j}$ is an operator that shifts the original image $\mathbf{x}$ with a shift $\mathbf{t}_j$. The term η_j is instrumental noise or model imperfections. According to the compressed sensing paradigm, we observe

$$\mathbf{y}_l = H_l\mathbf{x}_l, \tag{5.9}$$

where the sampling matrices are such that their union spans $\mathbb{R}^N$. In this application, each sensing matrix H_l takes $\lfloor N/P \rfloor$ measurements such that the measurement subset extracted by H_l is disjoint from the other subsets taken by $H_{j\neq l}$. Obviously, when there is no shift

between consecutive images, this condition on the measurement subset ensures that the system of linear equations in (5.9) is determined, and hence $\mathbf{x}$ can be reconstructed uniquely from the measurements $\mathbf{y}_l$. When there is a time shift, there is no such guarantee.

Figure 5.2 shows the result of a CS restoration, compared to the averaging approach. The CS approach appears to give superior results.

This application was highlighted along with several other applications of CS in astronomy by Bobin et al. [94]. We refer to their article for more details.

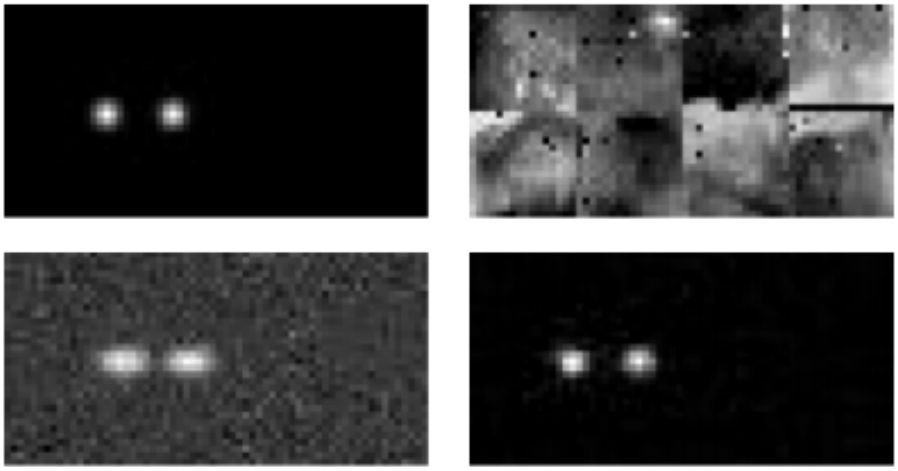


Figure 5.2: (top left) Original image. (top right) One of the 6 noisy images observed by the instrument. (Bottom left) Average of 6 noisy images. (Bottom right) Noiselet-based CS reconstruction. The figure is from [94].

5.3 CS for seismic data and surface metrology

The following problem introduction is from an example problem at [95], which explains how the problem of seismic data inpainting arises better than I could, given that this is not my area of expertise.

> In reflection seismology, sound waves are sent into the ground using an energy source such as dynamite or vibrator trucks. These waves are reflected off of, and transmitted through, different rock layers. The reflections are recorded at the surface via receivers called geophones that are planted in the ground. The recorded data set is an ensemble of seismic traces (time series from the different receivers). These help create an image of the subsurface, which is then interpreted for oil and gas exploration.
>
> These days, seismic data is collected as massive data volumes with up to five dimensions (one for time, two for the receiver positions, two for the source positions). This data shows a 3D volume of the Earth. In our example, we work with 2D seismic data, i.e., data showing a single slice of the Earth. It is set up as a three-dimensional matrix - one coordinate for time, one for the receivers, and one for the sources. Furthermore, we will work with only one shot gather (the data from a single source), which is a function of receiver index and time sample.

Seismic data are often spatially undersampled due to physical and budget constraints, resulting in missing traces (receiver positions) in the acquired data. A solution to this problem is trace interpolation, which estimates the missing traces from an undersampled dataset.

The data used in this example is fully sampled, so we will first simulate the effect of missing traces by removing the data from random receiver indices. We will then interpolate to try to fill in the gaps. From our earlier discussion on LIPs, this would be classified as an inpainting problem, but the lines between compressed sensing and inpainting are not very strict.

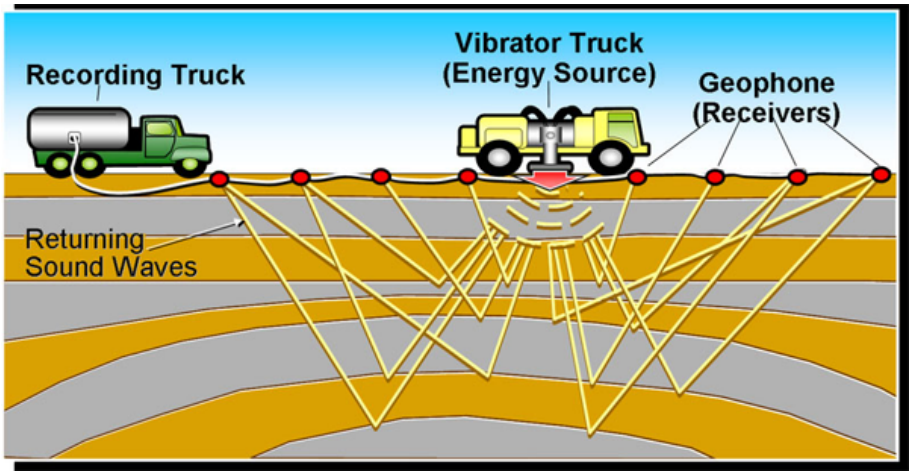


Figure 5.3: Overview of seismic data acquisition. Image from http://lingo.cast.uark.edu/LINGOPUBLIC/natgas/search/index.htm

An example of seismic data is shown in figure 5.4. This example data is included in the distributed Wave Atom toolkit for MATLAB, available at http://www.waveatom.org/. This data can be shown to be sparse in the curvelet, Wave Atom and Fourier basis. As a simple test, we remove 25% of the transceivers, which corresponds to removing 25% of the samples along vertical lines in the data set, as shown in figure 5.5.

We now try to fill in the missing samples. In this test we used the spgl1 solver to find the ℓ_1-minimizing in the respective bases. The results are shown in figure 5.6, and summarized in table 5.1. All the reconstructions are quite good, giving only notable errors at the edges of the signal. The curvelet and Wave Atoms transforms give roughly the same quality, while they both outperform the DFT.

The result of a larger test is given in figure 5.7 shows that for extremely small fractions of gatherers used, the curvelet performs somewhat poorly, while for a decent number of gatherers, the DFT falls behind the curvelet and the Wave Atom.

This problem is studied in more detail in [5], [96], [97] and [98], where they note that performance can be improved somewhat by using a jittered subsampling, which ensures there are no large areas going completely unsampled.

Surface metrology is the science of measuring small-scale features on surfaces. In, [99], J. Ma introduces CS on a number of such problems. Engineering surfaces are composed of multiscale topographies, such as roughness, waviness, form errors, random ridges and valleys and peaks/pits. These features directly impact performance and physical properties of the system, such

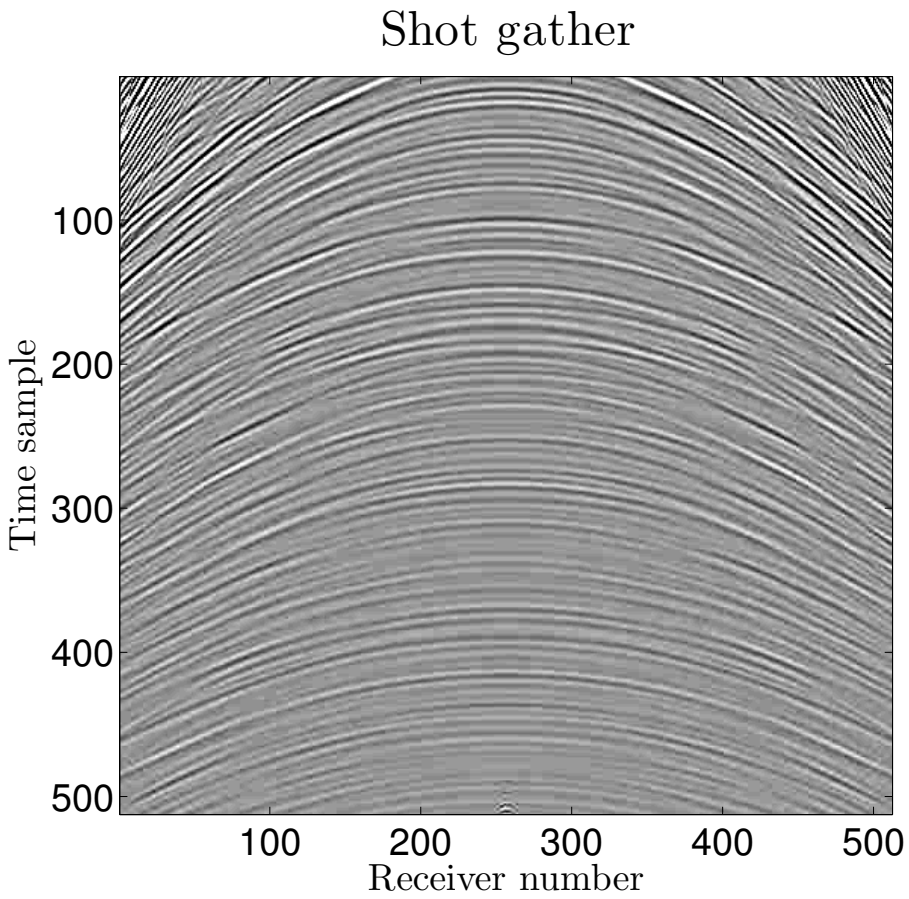
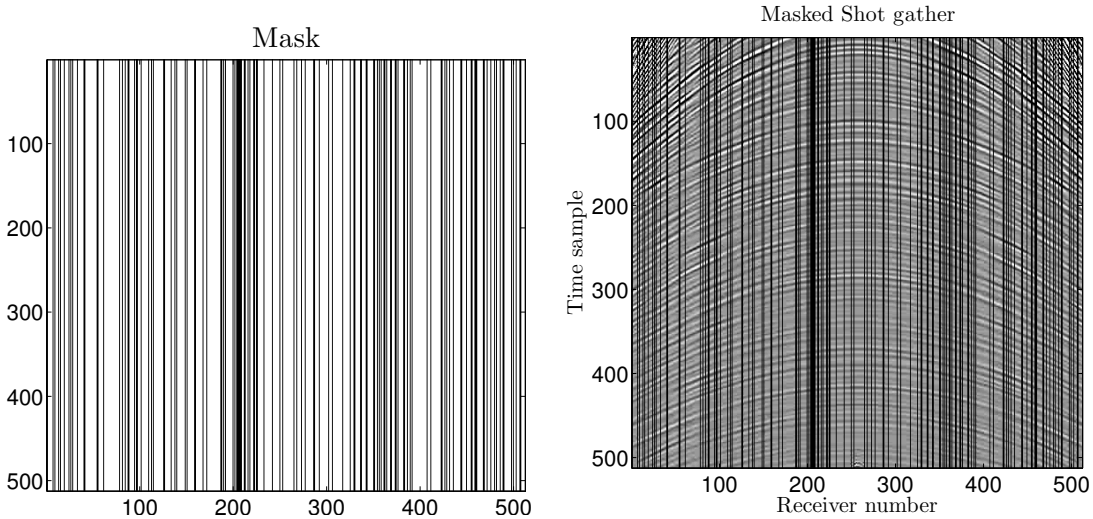


Figure 5.4: An example of seismic data.

as wear, friction, lubrication, corrosion, fatigue, coating, and paintability in many disciplines, including tribology, fluid mechanics, optics, semiconductors, microelectronics, manufacturing, biology, and medicine. For instance, during the functional operation of interacting surfaces, peaks and ridges act as sites of high contact stress and abrasion, whereas the pits, valleys, and scratches (i.e., polish line or line-like wear) affect the lubrication and fluid retention properties. Surfaces also play a vital role in biology and medicine with most biological reactions occurring on surfaces and interfaces, in vivo. Multiple methods were implemented to measure different characteristics, including engineering surfaces. Methods including scan, ultrasonics, and scatter have also been developed. Methods include scanning electron microscope, scanning tunneling microscope, magnetic force microscopy, computer tomography and MRI, which we have already seen in the context of medical imaging.

Figure 5.8(a) displays a raw surface topography from a worn metallic joint head with morphological structures consisting of roughness and deep scratches. We then simulate partial measurements by measuring the surface at random points in the Fourier space. If we fill in the missing samples in the Fourier space with zeros, we get figure 5.8(b). Two CS approaches are shown: In figure 5.8(c) a TV-minimization is performed, while in figure 5.8(d) a curvelet-based approach is shown. These figures are from [99], where several more illustrated examples are given.

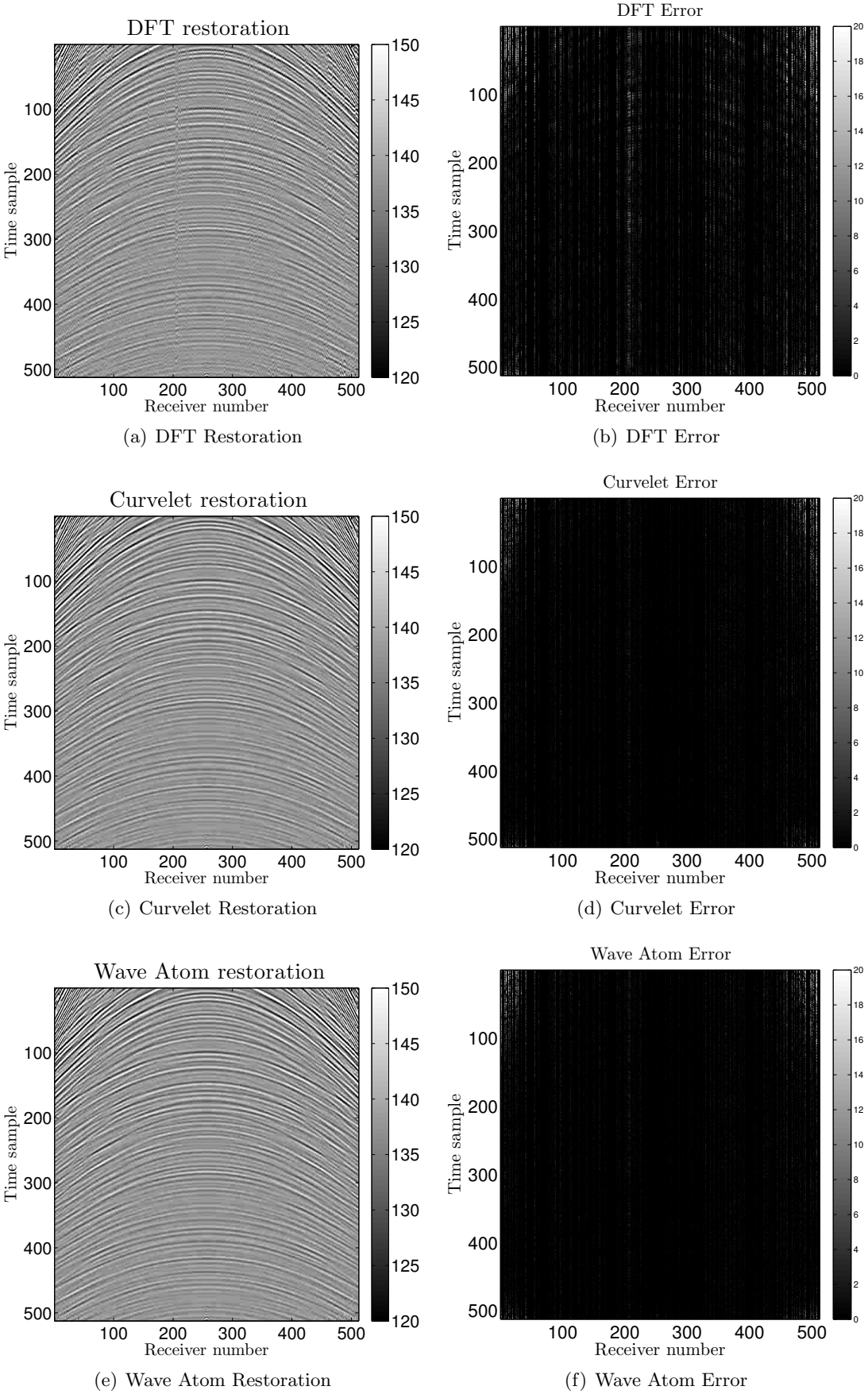


(a) Mask (b) Subsampled data

Figure 5.5: Masking of seismic data.

Number of traces	Transform	PSNR
75%	zero-filling	11.41
	DFT	40.90
	DCUT2	46.02
	Wave Atom	45.90

Table 5.1: The table shows the result of inpainting of seismic data with 25% missing traces.



(a) DFT Restoration

(b) DFT Error

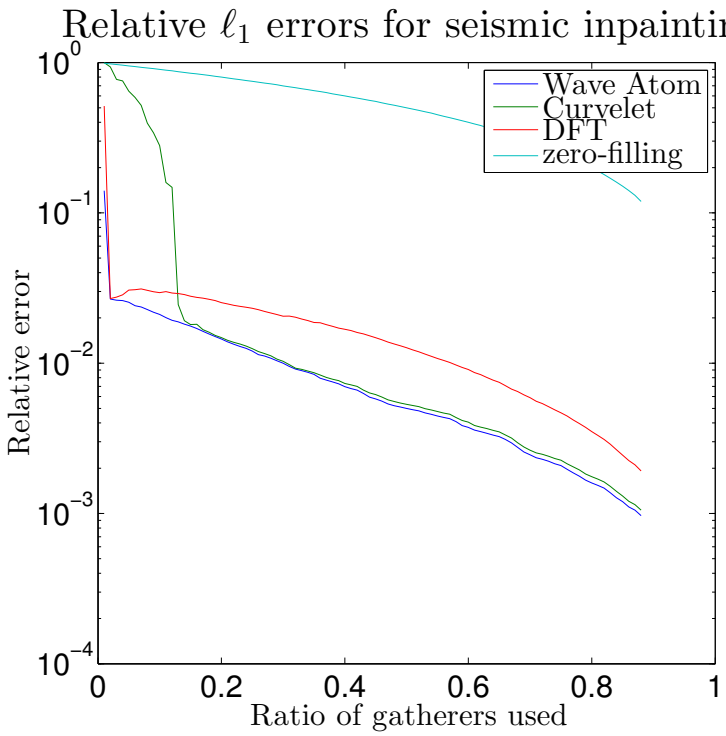
(c) Curvelet Restoration

(d) Curvelet Error

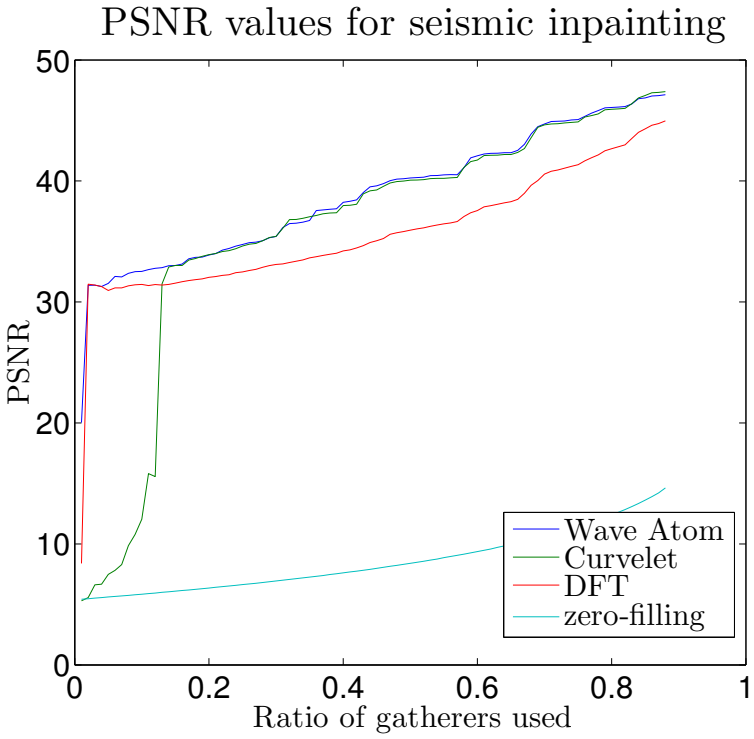
(e) Wave Atom Restoration

(f) Wave Atom Error

Figure 5.6: Inpainting of seismic data using different bases.

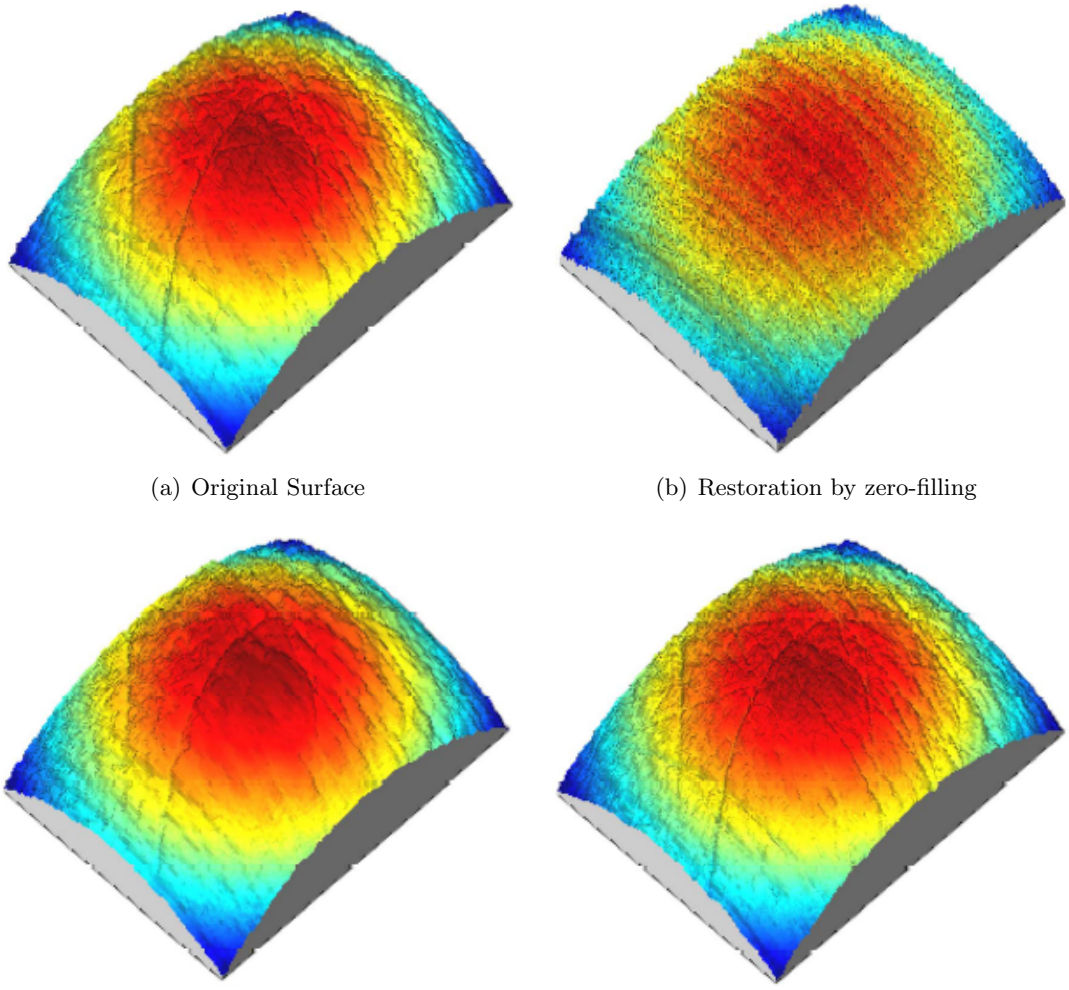


(a) ℓ_1-error



(b) PSNR

Figure 5.7: The result of a large inpainting experiment, varying the number of gatherers used in the restoration.



(a) Original Surface

(b) Restoration by zero-filling

(c) TV-minimization

(d) Curvelet restoration

Figure 5.8: Restoration of a raw joint surface based on partial Fourier Measurements. The figure is from [99].

5.4 Properties of molecular dynamics

This section aims to reproduce the results of two recent articles by Andrade and Sanders et al. ([3] and [4]) and put the ideas presented there in a more general setting. In [3] they approach the problem of measuring the Continuous Time Fourier Transform (CTFT), defined by

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}\,dt, \tag{5.10}$$

in a numerical setting. To be precise, the article studies the Continuous Time Cosine Transform (CTCT), given by

$$F_c(\omega) = \int_{-\infty}^{\infty} f(t)\cos(\omega t)\,dt, \tag{5.11}$$

which is just the real part of the CTFT. The approach is to evaluate the integral numerically at a discrete set of frequencies $\{\omega_j\}_{j=0}^{M-1}$, over a finite time $t \in [0, T)$. The frequency and time samples are uniform, so that $\omega_k = \omega_0 + k\Delta\omega$ and $t_j = j\Delta t$. The integral is evaluated numerically, which may give the crude formula

$$F_c(\omega_k) = \delta t \sum_{j=0}^{N-1} f(t_j)\cos(\omega_k t_j)\,dt. \tag{5.12}$$

If we denote the sample vectors $F_c(\omega_k)$ by $\mathbf{y}$ and $f(t_j)$ p $\mathbf{x}$, this set of equations can be written as the matrix equation

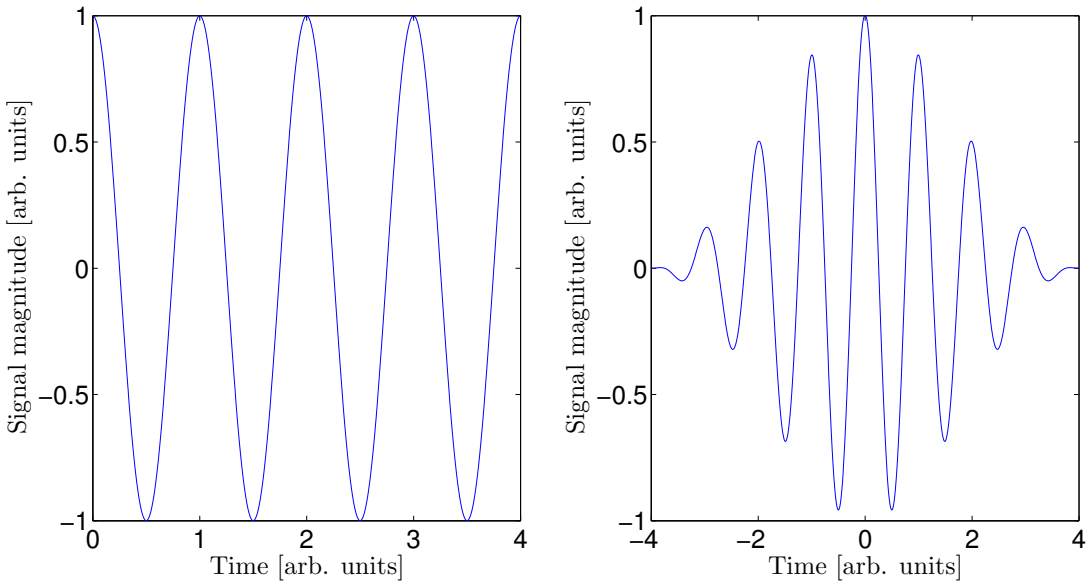
$$\mathbf{y} = F\mathbf{x}, \tag{5.13}$$

where F is an $M \times N$ (where $M > N$) matrix with elements $F_{kj} = \Delta t\cos(\omega_k t_j)$. This looks a lot like our definition of the Fourier Transform. However, it is quite different. First of all, we have made no restrictions on lower or upper bounds of frequencies, or the spacing between them. This means that there is no guarantee that the columns in F are orthogonal. There is also no requirement to have the same amount of points in time and frequency. To a point, we would like to have as many points in frequency and time as possible. Typically, we can use any number of frequencies that we like, but experimental considerations may often not allow large T and/or small Δt. Let us begin by investigating the effects of only measuring the signal for a short time.

We begin by creating a signal $x_j = \cos(ft_j)$. Here we will set $f = 2\pi$, and $\Delta t = 0.01$. We will use frequencies from $\omega_0 = \pi$ to $\omega_{M-1} = 6\pi$, with $M = 1000$. A final time of $T = 1$ will ensure one complete oscillation. Let us see what happens if we use final times of $T = 0.5$, 2 and 10. The resulting transforms are shown in figure 5.10. One standard way to improve the result of the CTCT is to multiply the signal with some damping function, and then add a symmetric extension of the signal [100]. We use the damping polynomial

$$p(t) = 1 - 3\left(\frac{t}{T}\right)^2 + 2\left(\frac{t}{T}\right)^3, \tag{5.14}$$

which has the properties $p(0) = 1$, $p(T) = p'(T) = p'(0) = 0$. This causes the function to vanish smoothly at T. This is illustrated in figure 5.9. The resulting CTCT is also shown in figure 5.10



(a) Original time series (b) Dampened and extended time series

Figure 5.9: The figures show the effect of applying a damping polynomial to a signal and symmetrically extending it.

Now we will attempt to approach the problem in a different way. The inverse continuous time cosine transform (ICTCT) is given by

$$f(t) = \frac{2}{\pi} \int_{-\infty}^{\infty} F_c(w) \cos(\omega t)\, d\omega, \tag{5.15}$$

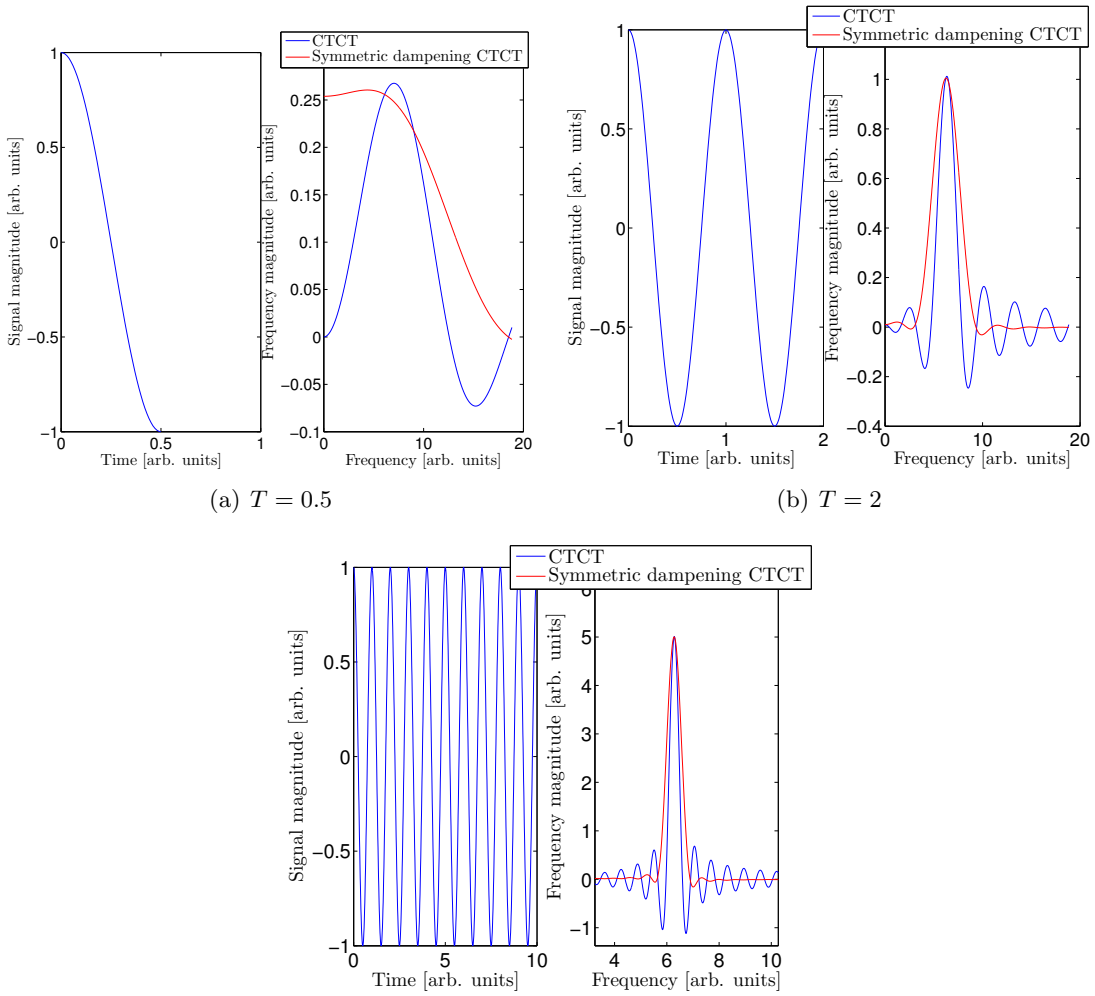
and using the same discretization, we can formulate this as the matrix equation

$$G\mathbf{y} = \mathbf{x}, \tag{5.16}$$

where G is an $M \times N$ matrix (where $M > N$) with elements $G_{jk} = \Delta\omega \cos(\omega_j t_k)$. Now, since $M > N$, the system is underdetermined, and we can choose any solution from the solution space. We also know that the CTCT should show only a sharp peak at the selected frequency. This motivates us to look for the sparsest possible solution, which is why CS might be well suited for this problem. Using the same parameters as previously, figure 5.11 includes the result of ℓ_1 as well as ℓ_2 minimization of the inverse problem. We note that even for time spans of less than one full oscillation, the CS method gives very reasonable results.

We should also check if the frequency found by the ℓ_1 minimization is correct. We follow the analysis of [3]. We consider the CS result to have no standard deviation, and use a Gaussian fit to model the result for the symmetrically extended CTCT result. The results are shown in figure 5.12, and are at least as good as the results for the traditional approach of symmetrically extending the signal.

This is of course highly related to the sparse spikes deconvolution problem, but the idea presented here is somewhat different.



(a) $T = 0.5$

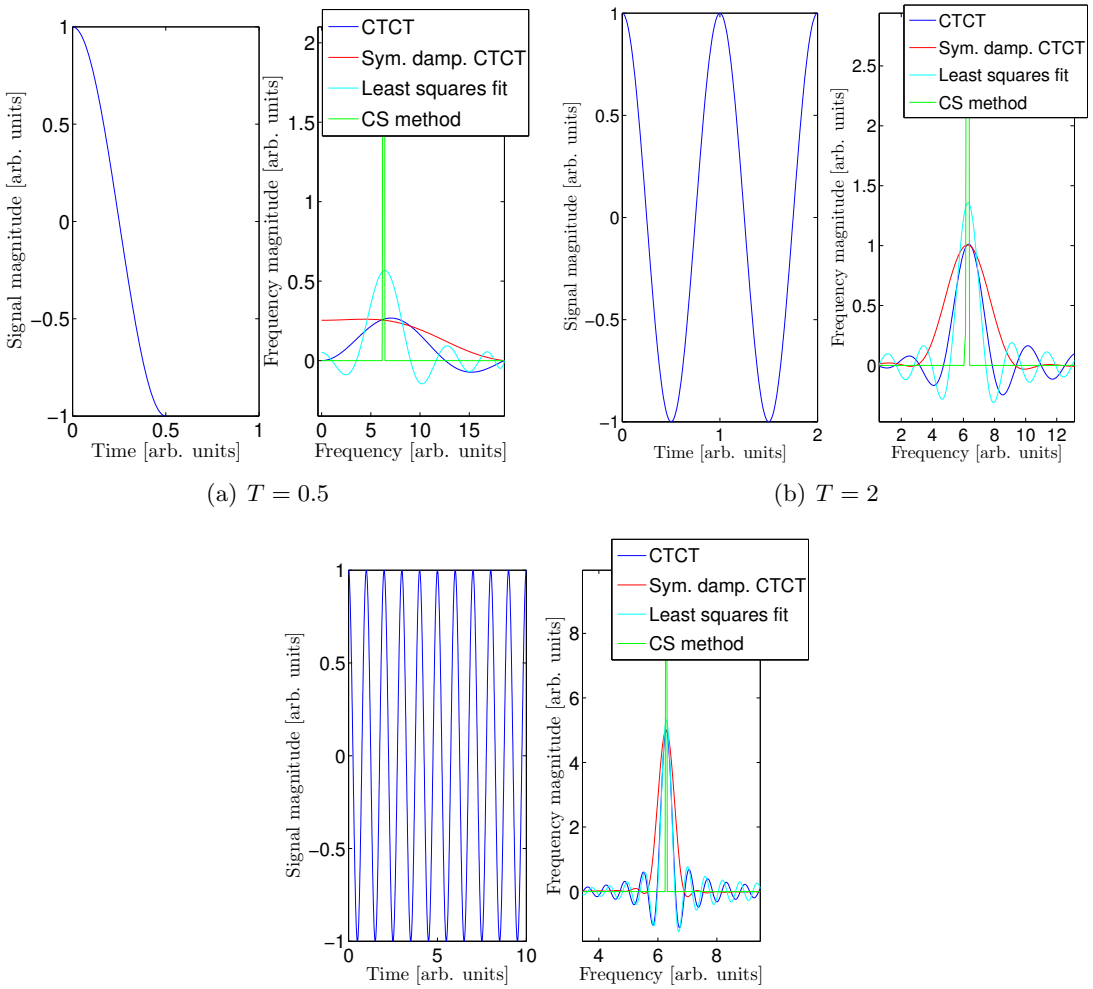
(b) $T = 2$

(c) $T = 10$

Figure 5.10: The figures show the numerical CTCT results for various measurement times.

Idea 5.4.1. If some property, such as a transform, when sampled for a sufficiently long time, converges to a sparse signal, we may be able to find a sparse solution of the inverse problem after a much shorter time period.

This feature hold for many properties in molecular dynamics, such as vibrational spectrum, optical absorption spectrum, spectroscopy data and more, as highlighted by Andrade and Sanders et al..



(a) $T = 0.5$

(b) $T = 2$

(c) $T = 10$

Figure 5.11: The figures show the numerical CTCT and the ICTCT restoration results for various measurement times. The peaks of the CS results have been cropped in order to show more detail of the other curves.

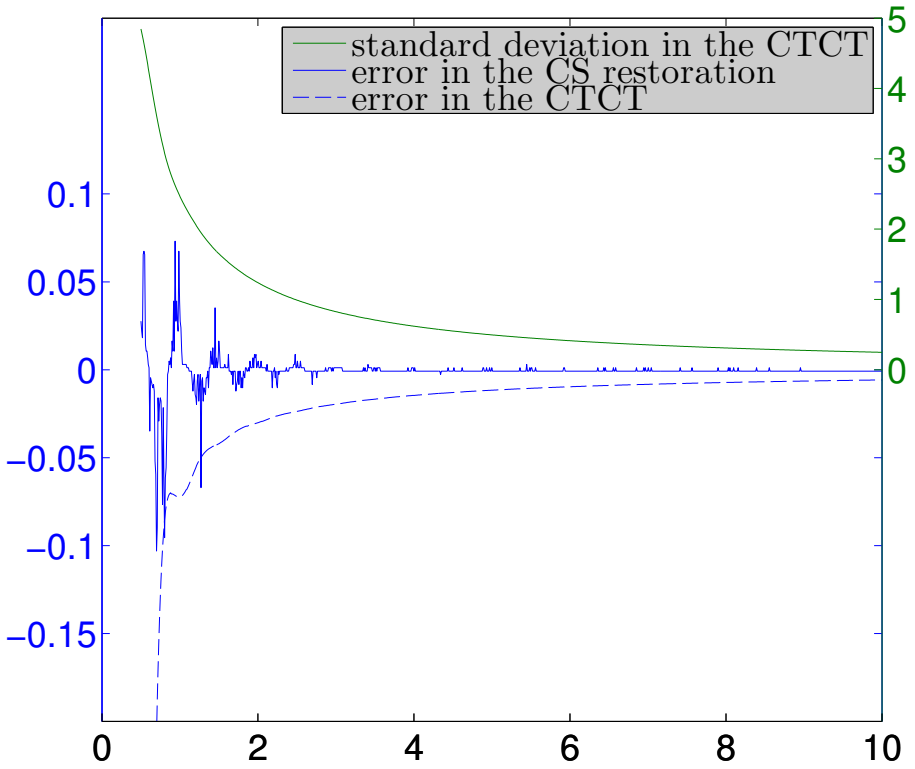


Figure 5.12: The figure shows the frequencies found by the CTCT and the CS methods, as well as the standard deviation of the CTCT frequency.

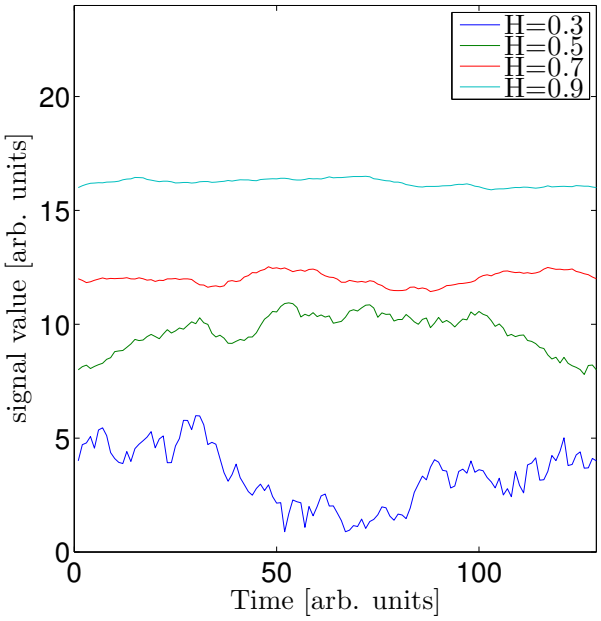


Figure 5.13: The figure shows some curves with different Hurst exponents created by the midpoint displacement method.

5.5 Characterization of rough surfaces

In fractal geometry, the *Hurst exponent* H is a measure of how wild the randomness of a time series is. It is directly related to the fractal dimension of a series. A value of $0 < H < 0.5$ will mean that a high value will typically be followed by a low value. A value of $H = 0.5$ will correspond to no correlation between adjacent values, a Brownian motion is an example of this. A value of $0.5 < H < H$ means a high value will typically be followed by another high value. Figure 5.13 shows some time series with different Hurst exponents. Let us properly define the Hurst exponent.

Definition 5.5.1 (Hurst exponent)**.** The Hurst exponent H is defined in terms of the asymptotic behavior of the rescaled range as a function of the time span of a time series as

$$\left\langle \frac{R(n)}{S(n)} \right\rangle \propto n^H \text{ as } n \to \infty, \tag{5.17}$$

where $R(n)$ is the range of the first n values and $S(n)$ is the standard deviation

Bassingthwaighte and Raymond [101], they analyze the problem of determining the Hurst exponent of a short time series. Here, we will investigate whether or we are able to determine the Hurst exponent using only a small amount of samples with reasonable accuracy. In order to estimate the Hurst exponent, we will use the same approach as in [101]. The reader may investigate the theoretical underpinning of this method in the cited article. The method consists of the following steps:

1. Calculate the successive differences, $\mathbf{y}$, of the signal $\mathbf{x}$,

$$y_i = x_{i+1} - x_i, \quad i = 0, \ldots, N-1. \tag{5.18}$$

2. Calculate the standard deviation of the set of observations:

$$\sigma_1 = \frac{1}{N}\sqrt{N\sum_i y_i^2 - \left(\sum_i y_i\right)^2}. \tag{5.19}$$

3. Collect the averages in consecutive groups of two elements in a new signal $\mathbf{y}^{(2)}$, and calculate the standard deviation of this new group σ_2. We refer to these groups as *bins* with *bin size* m. The bins should have bin size $m_2 = 2$ at this stage.

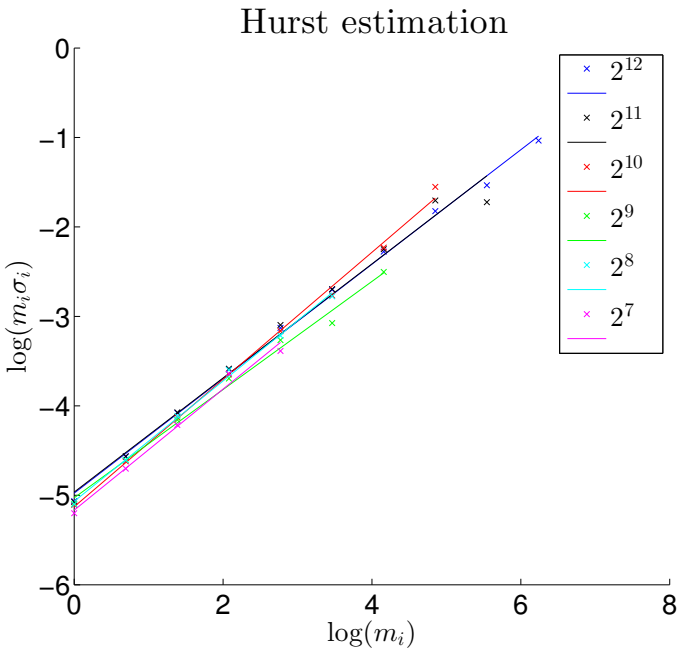
4. Repeat the previous step until you are left with $N_i \leq 4$ elements (4 here is arbitrary, in the sense that we could have stopped at 8 or 2 as well).

5. Plot $\log(\sigma_i)$ vs. $\log(m_i)$. The slope of this curve is related to the estimated Hurst exponent $\hat{H}$ as $\hat{H} = 1 + \text{slope}$.

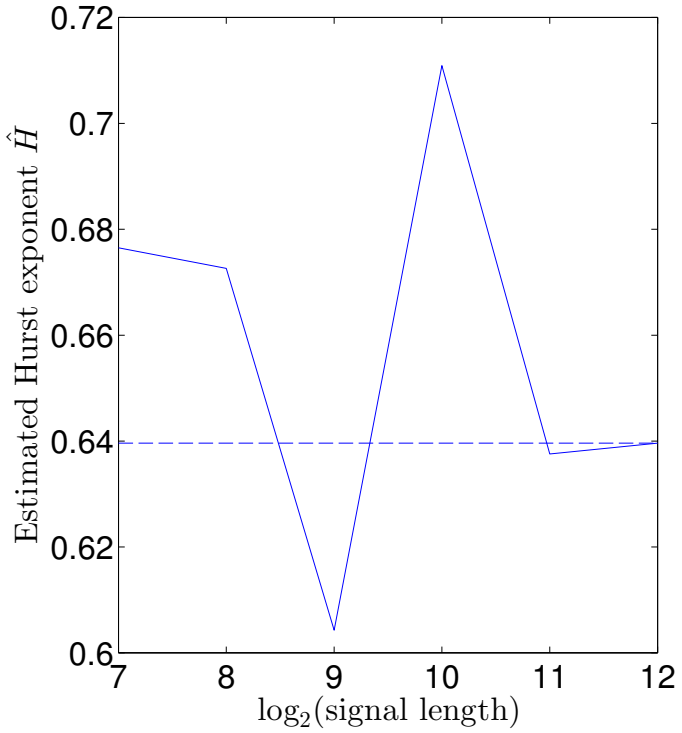
We can alter this method in order for the slope to find $\hat{H}$ as the slope of the curve directly by plotting $\log(\sigma_i m_i)$ along the y-axis.

This approach has already been implemented and made available on the MATLAB Central File Exchange, and this version was used in these tests. The function is available at http://www.mathworks.com/matlabcentral/fileexchange/9842-hurst-exponent.

The problem discussed in [101] was the signal length needed to accurately find the Hurst exponent of the whole time series. The problem is illustrated in figure 5.14, where a time series $\mathbf{x}$ of length $N = 2^{12} = 4096$ was created, and the algorithm described above was applied. The slope was then calculated. Then, we repeated the process using only the first $2048, 1024, 512, 256$ and 128 elements of the signal. We note that when attempting to estimate the Hurst exponent from only a fraction of the data set, the result fluctuates wildly. This is further explored in [101].



(a) The log-log lines found using different lengths of the signal



(b) The estimated Hurst exponents found using different lengths of the signal

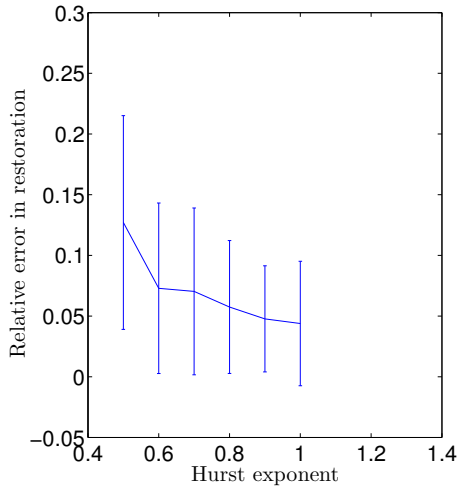
Figure 5.14: The figures show the results of attempting to reconstruct the signal from only the first signal samples.

It can be shown that the Fourier spectrum of such a series on average will be proportional to $1/f^{\beta}$, where $\beta = 2H + 1$ [102]. For this reason, we expect the series to be fairly compressible for large H, and for a signal restoration to be efficient. It has also been shown that for Daubechies wavelets (a family of wavelets we have not discussed), the wavelet detail spaces will follow the power law $w_n \propto 1/n^{1/2+H}$ [102], showing that wavelets will also be a suitable candidate for such a restoration. The theoretical argument in the mentioned article only uses the general scaling relation valid for any MRA wavelet, so it is very reasonable to expect the same for the Haar and CDF 9/7 wavelets as well.

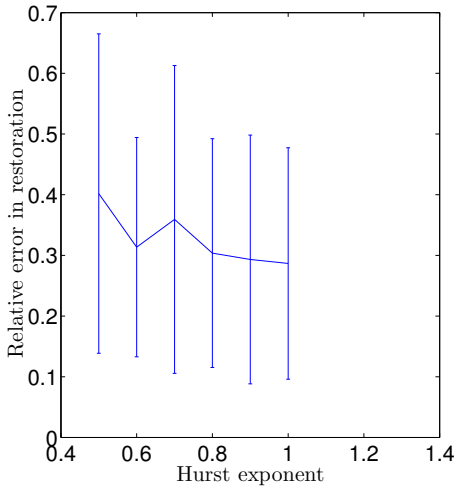
Some work has already been done with the goal of restoring an under-sampled time series of Brownian motion (see [103]). Here we will check the quality of such a restored signal by measuring the Hurst exponent before and after the under-sampling and restoration. The question of whether such parameters can be recovered well by CS techniques was posed by Ma [99], but has not been explored much in literature.

We will create datasets using the midpoint displacement method, which allows us to create series which will on average have a Hurst exponent which can be chosen freely. The following experiment was performed. For Hurst exponents $H = 0.5, 0.6, 0.7, 0.8, 0.9, 1.0$, 100 experiments where performed where either $N_{us} = N/2$ or $N_{us} = N/4$ samples were chosen randomly. The experiment was repeated for $N = 128$ and $N = 1024$, as well as for a DFT and a CDF 9/7 DWT. For the DWT, a random projection was applied first, in order to ensure an optimal restoration. The Hurst exponent of the restored signal, H_r, was then measured, and the relative error $|H - H_r|/H$ was then measured. The results are shown in figure 5.15.

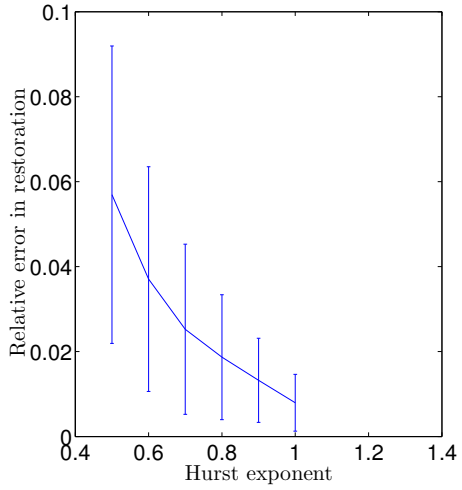
The results turned out to be quite poor. For $N_{us} = N/2$ the estimates are decent, for smaller N_{us} however, the errors are large. This is because all the Fourier coefficients are needed to give the signal its multi-scale features. We also note that the wavelet transform gives slightly better results overall, but this approach remains unimpressive.



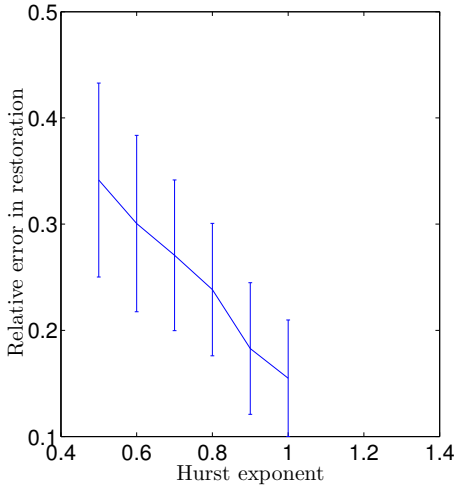
(a) DFT, $N = 128$, $N_{us} = N/2$



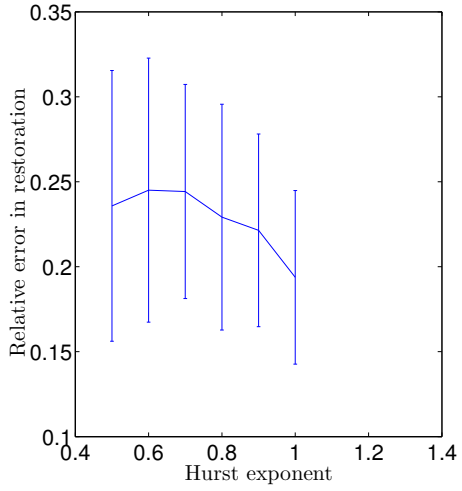
(b) DFT, $N = 128$, $N_{us} = N/4$



(c) DFT, $N = 1024$, $N_{us} = N/2$



(d) DFT, $N = 1024$, $N_{us} = N/4$



(e) CDF 9/7 DWT, $N = 1024$, $N_{us} = N/4$

Figure 5.15: The figures shows the relative errors when measuring the Hurst exponent of restored signals.

5.6 Learned Dictionaries for characterization of rough surfaces

We have tried to find the Hurst exponent for rough surfaces by looking for sparse representations in several known bases. The problem with this approach is that components in these bases (Fourier, Wavelet), only provide information on particular length/frequency scales. For a correct calculation of the Hurst exponent, one must consider the signals properties on all length scales, which does not work well with a sparse representation, unless the basis elements also contain information on all length scales.

One approach to a basis or frame with such properties is to consider a basis consisting of several different rough surfaces. We will call such a collection of suitable functions a dictionary, drawing on the terminology from section 3.7. A dictionary may may have a number of elements far larger than the dimensionality of the system, but it does not necessarily have full rank (i.e. it can not be reduced to a basis).

Variants of this approach is to look at a dictionary of transforms of rough surfaces, such as dictionaries of DFTs, DWTs or differences. The idea is that, when looked at the right way, different surfaces with the same Hurst exponent will be similar.

The goal of such an approach would obviously be that our method, when selecting M measurement points in whichever basis we choose, should outperform simply selecting the M first points of the signal and calculate the Hurst exponent based on those samples.

Figure 5.16 shows the result of such a preliminary investigation, where we test dictionaries of surfaces, differences, DFTs and DWTs. Here $N = 512$, and the number of samples is $N_{us} = 64$ in 5.16(a) and $N_{us} = 32$ in 5.16(b). The number of learning surfaces were $M = 2000$, with H in the range $[0.45, 1.05]$. The Fourier and Wavelet approaches seem to give poor performances, while the surface and difference approaches seem more promising. It is worth noting that in 5.16(b), there were too few points to calculate the Hurst exponent by simply sampling the first N_{us} samples.

We should also check the convergence properties of this approach. In a second experiment, we choose a fixed H, in this case $H = 0.7$, and attempt to find the Hurst exponent using different numbers of samples. This should error should converge to 0 for all methods. This holds, but it is interesting to note that the convergence appears better for the straight forward sampling approach. This is shown in 5.16(c).

We should check if this methods predicts the surface curve. We do a simple check where we sample 64 points in the basis we investigate, and plot the restored curves. The result is shown in figure 5.16(d), and shows that only the surface dictionary has this property.

So far we have used dictionaries of $M = 2000$ surfaces in all our experiments. It is interesting to check the effects of using different fewer surfaces in our simulations. We can easily test this, and the result is shown in figure 5.17(a), where we have used an subsampling factor of $1/8$, and a Hurst parameter of $H = 0.7$. We might expect the result to drop with the number of surfaces, and this is true for the direct surface measurements. However, for the difference based method, the number of test surfaces does not significantly effect the results.

Another feature we are interested in is whether this method can be used to distinguish smaller differences in the Hurst parameter if the dictionary is more fine-tuned around a specific set of Hurst parameter values. Once again, we can simply test this. We can create a dictionary of differences with Hurst parameters in the range $[0.74, 0.80]$, and attempt to restore surfaces with Hurst parameters within this range. The result of this is shown in figure 5.17(b). We note that while there might be some improvement compared to the wide-scale searches in terms of

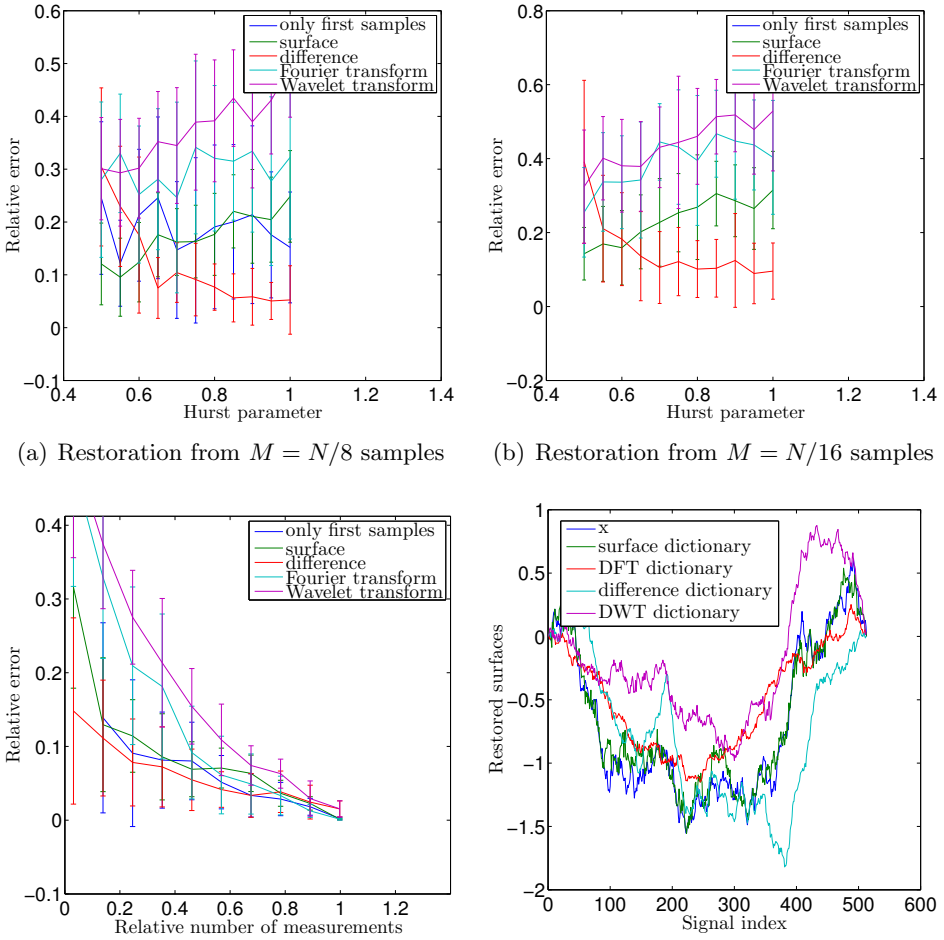
the relative error, the result is not adequate to distinguish such similar Hurst parameters.

Finally, there is no a priori reason to spread the samples randomly out over the the entire domain. Aliasing effects do not necessarily occur with this dictionary. We perform two more tests: one where all the samples are taken from the first elements of the surface (or the relevant transform of the surface), and one where the samples are spread out equidistantly. The results of this test is shown in figure 5.18.

Somewhat surprisingly, the approach of sampling only the first few points shakes up the results quite a bit. The difference approach is still superior, but now the Fourier measurements follow right behind. The surface measurements, however, now produce poor results. This approach also greatly improves the performance of the difference dictionary for low Hurst parameters.

The impact of sampling equidistantly seems insignificant for most of the methods, except for the Fourier dictionary, which improves its performances with this sampling scheme compared to random sampling. It still performs better when only the first Fourier coefficients are sampled, however.

To conclude, based on the method where we used only the first samples, and restored the signal using a dictionary of differences, we have here presented a method for estimating the Hurst exponent of a signal, using only a small fraction of the elements needed normally.



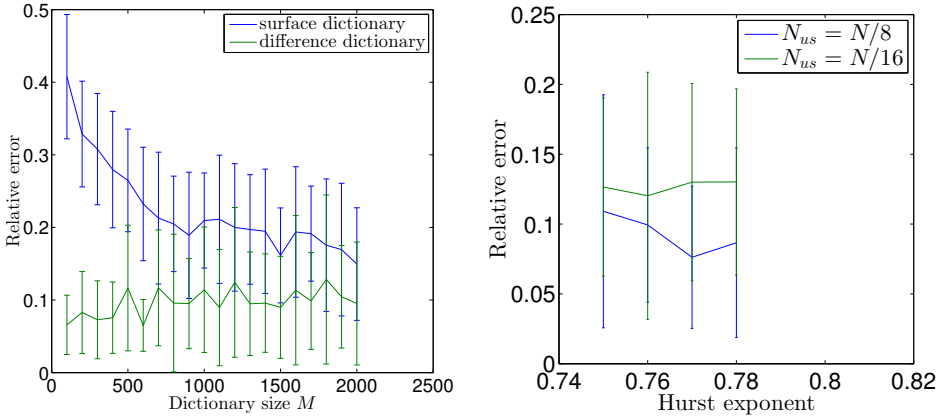
(a) Restoration from $M = N/8$ samples

(b) Restoration from $M = N/16$ samples

(c) Hurst estimation error vs. the number of measurements made

(d) Attempted restoration of the curve using various subsample schemes

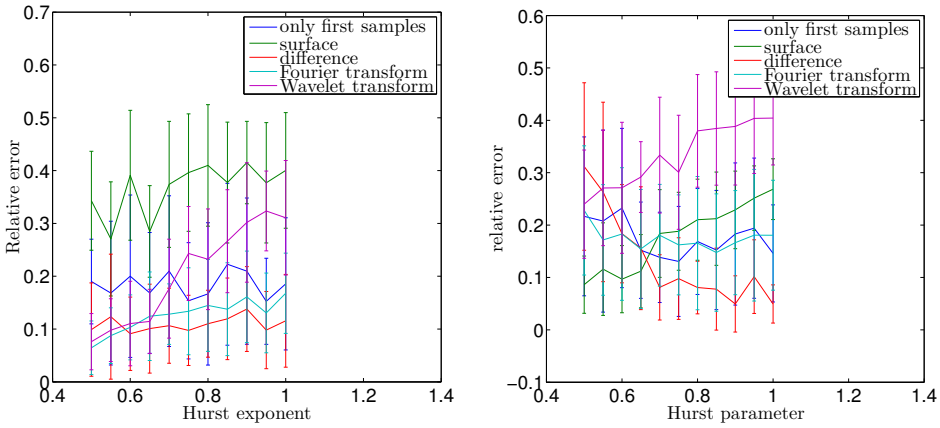
Figure 5.16: Preliminary tests for the restoration of Hurst exponents. Figure 5.16(a) and 5.16(b) show the error in the restoration for different Hurst exponents, figure 5.16(c) shows the error when the number of samples is varied and the Hurst exponent is kept fixed, and finally figure 5.16(d) shows the attempt to restore the surface curve using the various methods.



(a) The relative error for a fixed Hurst exponent for various dictionary sizes

(b) The effect of only selecting Hurst exponents from a small interval. Only the difference dictionary was tested.

Figure 5.17: The effects of the number of surfaces and different ranges of Hurst parameters used to create the dictionary.



(a) $N_{us} = N/8$, first elements sampled

(b) $N_{us} = N/8$, equidistant sampling

(c) $N_{us} = N/16$, first elements sampled

(d) $N_{us} = N/16$, equidistant sampling

Figure 5.18: Tests using different sampling schemes.

5.7 Super-Resolution based on curvelet sparsity

This is not strictly connected to physics in itself. Rather, it reflects original work that was made while working on this thesis, and introduces concepts which will be relevant later.

5.7.1 Introduction

The problem of super-resolution amounts to solving the inverse problem of estimating a high resolution signal $\mathbf{x} \in \mathbb{R}^n$ from only m measurements, where $m < n$. In a general setting, the measurements may also be contaminated by an additive noise, $\mathbf{w}$, yielding the problem formulation

$$\mathbf{x} = U\mathbf{y} + \mathbf{w}, \tag{5.20}$$

where U is some operator.

Image interpolation is an important example of super-resolution. For images, a bicubic interpolation often provide nearly the best result among linear operators [104].

Much work has already been done regarding super-resolution using CS, some using learned dictionaries as representation bases [105], [106], some non-adaptive bases [107]. Additionally, other super-resolution schemes based on sparsity have been proposed [108] as well other DCUT-based methods [109], [110].

Here we will suggest a new way of looking at the super-resolution problem, connecting it to the problem of restoring information from incomplete measurements in CS. For an image $\mathbf{x}$, consider its 1-level discrete Haar-wavelet transform (DWT). An example is shown in Figure 5.19 using the "Lena" image. The upper left part of the DWT represents a low-resolution of the image. Because of this, we can consider a sensing matrix Φ which computes only the low-frequency part of the Haar DWT.

The low resolution image will be quite jagged, and it is reasonable that the curvelet transform of can be made sparser by smoothening the image. For this reason, we will let Ψ be the curvelet transform. We want to find the sparsest possible curvelet transform such that the low-res version of the inverse CT matches to the low resolution version of the original image. Of course, in a realistic setting, we would simply start with the low resolution image and use the same procedure.

Once this method has been established, it is easy to change the CT for some other curve-promoting basis, as well as change the amount of levels we want to upscale the image, simple by letting Φ correspond to an m-level DWT, where $m > 1$.

5.7.2 Experimental results

The outlined method was tested using MCALab [43], which can solve the ℓ_1-minimization problem

$$\begin{aligned} &\text{minimize} && \|\mathbf{z}\|_1 \\ &\text{subject to} && \|\Phi\Psi\mathbf{z} - \mathbf{y}\|_2 \leq \sigma, \end{aligned} \tag{5.21}$$

by Morphological Component Analysis (MCA) [111], as well as the alternative formulation

$$\text{minimize}_{\mathbf{z},\sigma_\epsilon} \frac{1}{\sigma_\epsilon^2}\|\Phi\Psi\mathbf{z} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{z}\|_1, \tag{5.22}$$

based on the estimation maximation (EM) algorithm [112].

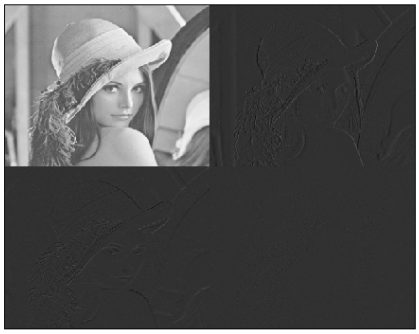


Figure 5.19: The Lena image and its Haar DWT.

$m = 1$	zero-filling	bicubic	EM	MCA
Lena	31.56	34.31	33.03	34.71
Mandril	27.28	29.66	29.52	30.93
Boat	28.57	30.11	29.71	30.61
Cameraman	31.12	36.01	33.92	35.55
Straws	19.05	20.54	21.91	21.53
Peppers	29.93	32.07	31.04	32.06
Avg. gain	-2.534	0	-0.5978	0.4465

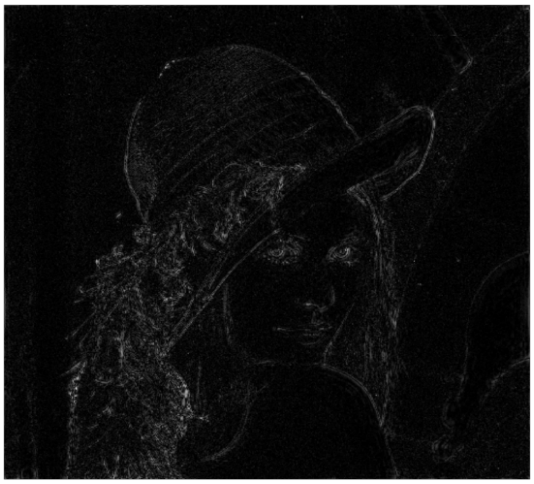
Table 5.2: Results for the curvelet transform for super-resolution, upscaled by a factor of 2×2. The columns show the PSNR value of the restorations. The average gain is compared to the bicubic interpolation.

The approach was tested on the test images "Lena", "Peppers", "Mandril", "Cameraman", "Boat" and "Straws", see Figure 2.4. These are all accessible through the USC-SIPI image database [6]. The results are shown in tables 5.2. Figure 5.20 shows the restored Lena image using the different methods, along with the absolute errors. Note that in particular the error for the MCA based algorithm is less pronounced along edges, which would, along with an improved PSNR, seem to indicate that the method is successful at improving sharpness of the image.

(a) Bicubic

(b) Bicubic error

(c) EM



(d) EM error

(e) MCA

(f) MCA error

Figure 5.20: A super-resolution reconstruction for an $m = 1$ level down-sampling, along with errors, for different methods.

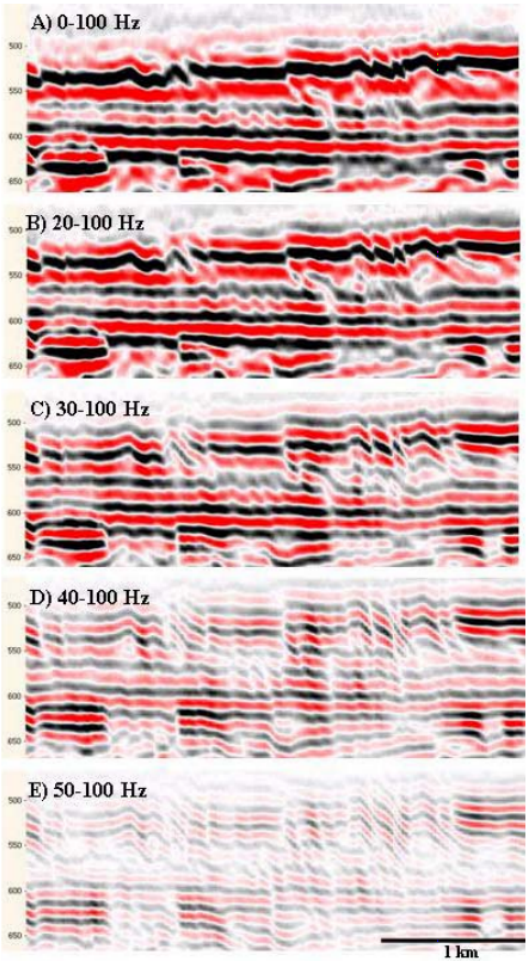


Figure 5.21: Seismic data acquired with different signal frequencies. The figure is from http://folk.uio.no/hanakrem/svalex/E-learning/geophysics/Seismic_resolution.pdf

5.8 Seismic data restoration based on data with varying resolution

In seismic such as the one described at the beginning of section 5.3, there is always a trade-off between the amplitude of the signal and the resolution. The is controlled by the frequencies used in the data acquisition, as illustrated in figure 5.21. Often, this is mitigated by sampling the signal at high frequencies in only a select few places, and relying on low-resolution measurements for most of the image. In this section we will explore how to use the framework we have developed in order to improve on sampled data in this form.

Using terminology from section 3.3, we would line to make measurements along different lines in each of the resolution spaces $V_m, V_{m-1}, \ldots, V_0$. This operation is linear, but it cannot be written as a simple direct product of two operators. Rather, an explicit sensing matrix like that would require a large sum of direct products. However, we may use the DWT theory developed as an inspiration for how to model this process in a way that will not require an unreasonable

amount of memory or CPU usage. There are two clear ways of doing this.

1. We could simply select samples in the m-level DWT. This has the advantage of giving orthogonal measurements, and is very simple to implement. However, the DWT is really the representation of the basis $V_1 \otimes W_1 \otimes W_2 \otimes W_m$, which means we have to chose our samples in a very specific way if we want the samples to reflect samples made in V_m, V_{m-1}, $\ldots, V_0$.

2. Another Approach is to collect samples from each of the spaces $V_m, V_{m-1}, \ldots, V_0$ as we work our way down through the stages of the DWT. There measurements might not be orthogonal (we might measure a particular line both with high and low resolution, giving some overlap).

Let us begin by exploring the first option. In order to ensure that we collect samples from the DWT which corresponds to only making samples in $V_m, \ldots, V_0$, we must make sure that if we sample a particular area in W_k, then we must make sure that we have sampled the same area in $W_{k-1}, \ldots W_0, V_0$, i.e. we must make sure that we do not add fine details to an area of the image if we have not added coarse details. We must also sample from all the detail spaces $W_m^{(0,1)}$, $W_m^{(1,0)}$ and $W_m^{(1,1)}$

In an 1-level DWT this would mean that we do not sample from the detail spaces W_0 unless we have already sampled the low-resolution space V_0, as illustrated in figures 5.22(a)-5.22(b). In a 2-level DWT we may only sample from the finest detail spaces W_1 if we sample the corresponding areas in the detail spaces W_0 and the low resolution space, as illustrated in figures 5.22(c)-5.22(f).

Generating acceptable masks can be done in the following way.

1. Sample all of the low-resolution space.

2. Make some mask of traces for one of the first detail spaces, and copy this mask for every other detail space.

3. The mask for the next detail space can now be generated by doubling the resolution of the previous detail space mask, and then removing some number of traces.

The result of a few experiments using this strategy is shown in table 5.3. The result indicates that as long as we have a decent number of total samples, we can improve the signal significantly using this method. The result of one experiment is shown in figure 5.23

In order to implement the second method, we may draw some inspiration from the idea used in Atomic Decomposition. In atomic decomposition, we tried to restore the signal from an overcomplete dictionary, consisting of two or more bases or frames. Here, we would instead like to make our measurements from such a dictionary. The dictionary is the combination of the low-resolution parts of various m-level DWTs, as well as the original data itself.

The easiest way to implement this is to implement this in our toy model is the following:

1. Generate all the m-level DWTs of the signal X, and store these as one long signal $\mathbf{x}$.

2. Mask all the detail spaces of all the DWTs, as well as the non-sampled parts of the low-resolution spaces.

Number of measurements	Transform	PSNR
32%	zero-filling	32.5
	DFT	32.54
	DCUT2	33.21
	Wave Atom	33.31
57%	zero-filling	35.79
	DFT	37.53
	DCUT2	39.73
	Wave Atom	39.13
67%	zero-filling	38.42
	DFT	41.40
	DCUT2	45.73
	Wave Atom	45.49
78%	zero-filling	40.30
	DFT	43.31
	DCUT2	47.29
	Wave Atom	47.25

Table 5.3: The table shows the result of single CS restorations on seismic data with varied resolution using the first method.

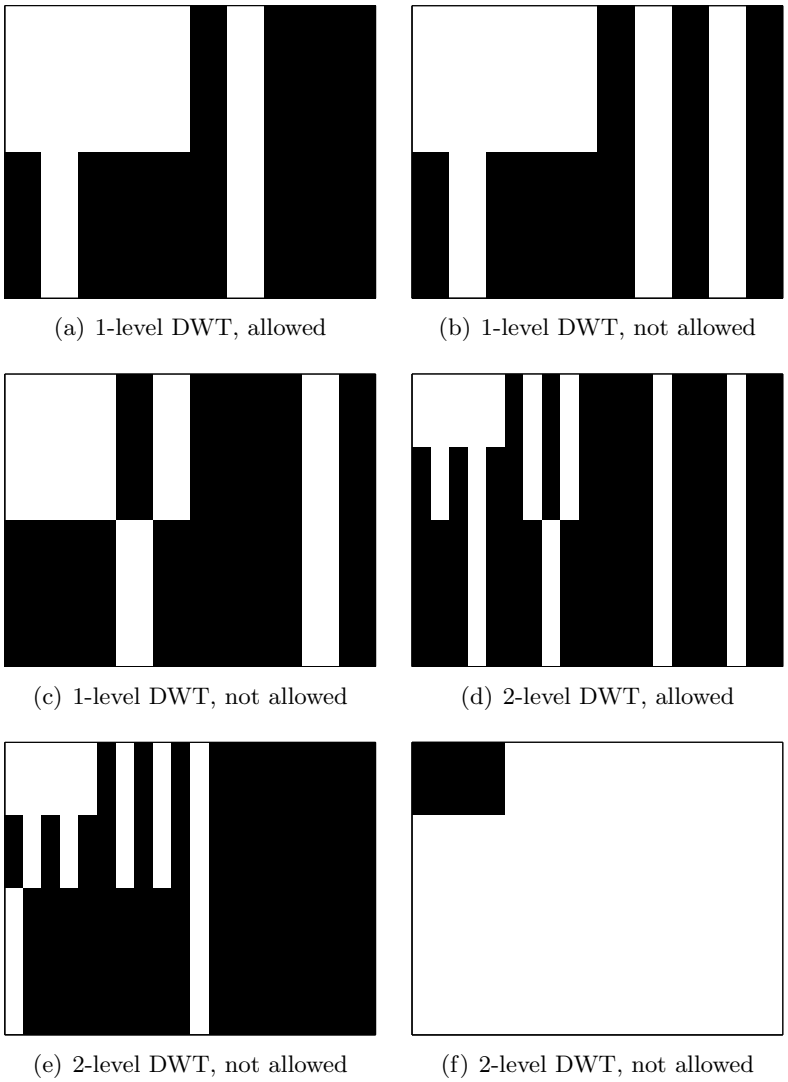
3. Now we attempt to find the sparsest possible set of DFT coefficients consistent with all these measurements (or any other transform).

4. Once the sparse coefficients have been found, we apply the inverse transform to found the restored signal.

The fact that we overproduce a lot of the DWT coefficients which take part in several of the m-level DWTs does not really matter, as we only compare the signal to the non-masked coefficients. The result of a few experiments using this method is shown in table 5.4. The results are comparable to the other method. Note that in this case the zero-filling would not make sense, as the sensing vectors are not orthogonal.

Number of measurements	Transform	PSNR
55%	zero-filling	N/A
	DFT	34.74
	DCUT2	38.44
	Wave Atom	38.38
65%	zero-filling	N/A
	DFT	36.72
	DCUT2	42.37
	Wave Atom	42.52

Table 5.4: The table shows the result of single CS restorations on seismic data with varied resolution using the second method.

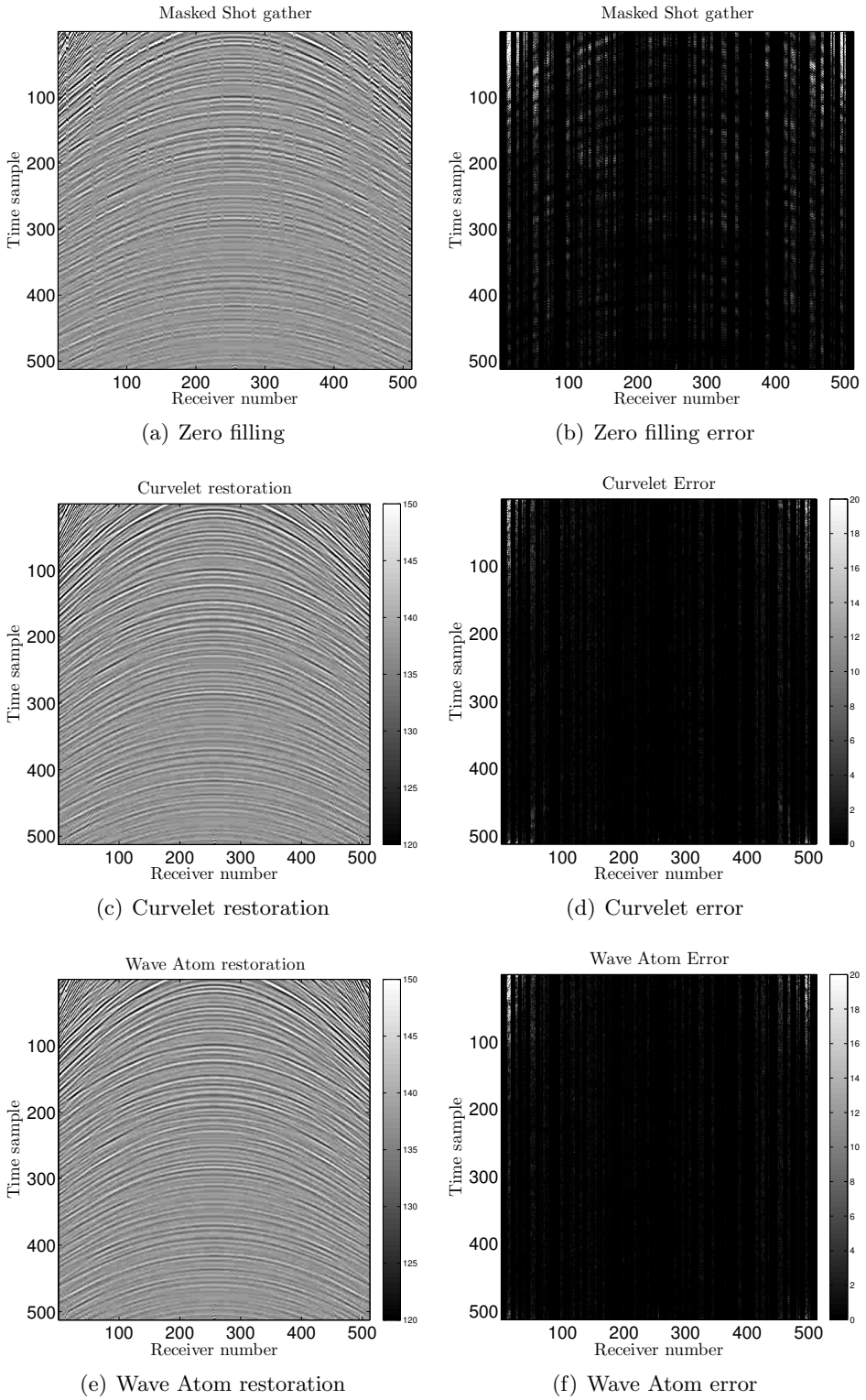


(a) 1-level DWT, allowed (b) 1-level DWT, not allowed

(c) 1-level DWT, not allowed (d) 2-level DWT, allowed

(e) 2-level DWT, not allowed (f) 2-level DWT, not allowed

Figure 5.22: The figures shows different attempts to create masks in the DWT spectrum of an image which corresponds to measurements in the spaces V_m. 5.22(a) and 5.22(c) are allowed. 5.22(d) is not allowed, as we have sampled some of the detail spaces, but not all (see the rightmost white line in the image, which is not there for the lower left detail space). 5.22(b) is not allowed as we have sampled details in an area where we have not sampled the low-resolution signal. 5.22(e) is not allowed, as we have sampled fine details in an area where we have not sampled coarse details (i.e. we sample the leftmost line of the fine detail space). 5.22(f) is not allowed as sample from both detail spaces, but do not sample from the low-resolution space.



(a) Zero filling

(b) Zero filling error

(c) Curvelet restoration

(d) Curvelet error

(e) Wave Atom restoration

(f) Wave Atom error

Figure 5.23: The figures show the result of restoring an image where 67% of the DWT spectrum of a 4-level DWT was sampled. All the lowest resolution measurements were samples, and the detail space sampling mask followed the rules laid out in the text.

5.9 Seismic Data Restoration based on learned dictionaries

Here, we will go back to the problem of inpainting missing shots in a seismic signal. Rather than any of the fixed bases we have used so far, we will here try to restore the signal based on a learned dictionary of image patches. Such dictionaries have been explored for image denoising ([113], [114], [115]) super-resolution ([105], [106], [116]) as well as the general compressed sensing problem ([117], [118], [119]), but the problem of inpainting is less well explored (some mentions are made in [115]). We briefly introduced the concept of learned dictionaries in 5.6, but here we will use some more finesse in constructing our dictionary.

The reason for using patches is two-fold. First, the learned dictionaries do not have fast implementations, nor can they be written as simple Kronecker products of 1-D operators, which means that dictionaries of full scale signals are impossible both to store and use, as exemplified in section 3.4.2, where we noted that an explicit representation of the 2-D DFT of a 512×512 image would require a $512^2 \times 512^2$ complex matrix, which in turn would use 1024 GB of memory. Secondly, we expect a signal to behave in a highly coherent way locally (in patches), but the signal might look less coherent over large distances in the data, making it more difficult to construct a basis in which the signal is sparse.

In order to create a suitable dictionary of patches, we can rely heavily on the established theory for denoising (if a learned dictionary is effective for denoising, it is reasonable that it is effective for inpainting). Ahron et al. published an algorithm for training a dictionary to represent certain data in a sparse way [120], called the K-SVD algorithm. For the sake of brevity, we will not go into detail on the numerics on the algorithm, but we note that several efficient MATLAB implementations already exist (notably, R. Rubinstein et al. published an efficient version [121], which is available online [122]). The main points of the algorithm are the following:

1. The algorithm is first fed some initial dictionary, as well as a large set of training patches. A good initial dictionary will ensure a fast convergence, but generally, any starting point is acceptable. Typically, an overcomplete Discrete Cosine Transform (DCT), a real-valued version of the Fourier transform, is a suitable starting point for many signals. The training data should be relevant to the data we want to restore later using our trained dictionary.

2. Then, the dictionary is iteratively updated, to make the training patches have an as sparse as possible representation in the data set. The dictionary elements are changed in order to improve the sparsity.

Figure 5.24 shows the result of such a training process, when the training data is made from random patches from figure 5.4. Note that the basis of patches effectively have a lot in common with the wave atom and curvelet bases, as they have local (due to the patches) oscillatory behavior along various angles. The dictionary patches were chosen to be of size 16×16. The size of the patches must be selected to ensure that the area that should be inpainted (i.e. single pixels) is small compared to the size of the patches. At the same time, the patches should be small enough to ensure that the behavior of the signal is fairly coherent across the patch. For denoising, a typical patch size is 8×8, but through testing this turned out to be too small for inpainting.

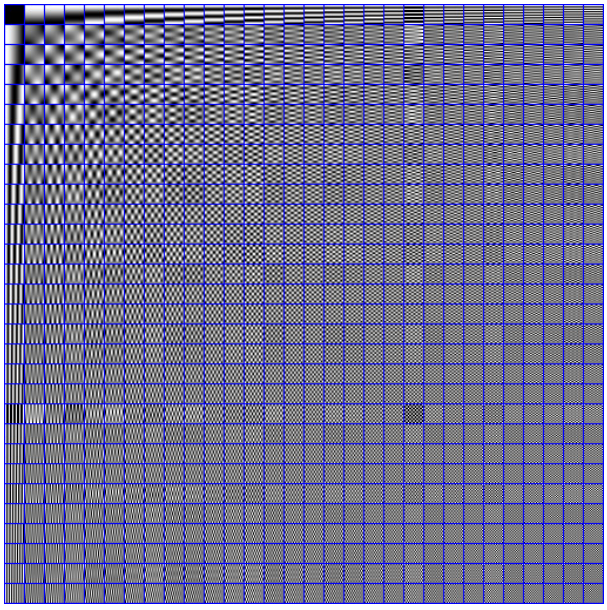
In setting up the experiment for inpainting seismic data, we should make sure that the missing shots are not on the en of the patches, as this would be outpainting, a much more

Number of measurements	Transform	PSNR
75%	zero-filling	11.41
	Wave Atom	46.52
	Learned dictionary	48.68

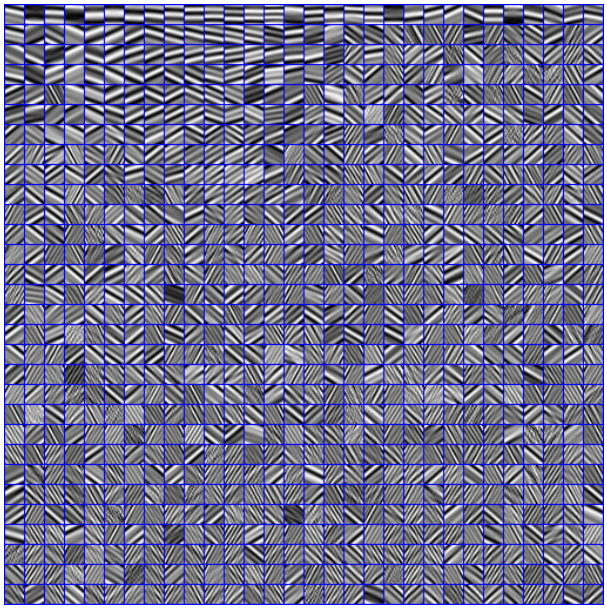
Table 5.5: The table shows the result of single CS restoration on seismic data using a learned dictionary, compared to the Wave Atom transform.

difficult problem than inpainting. In order to do this, we remove every 4th column of the data, starting with the 2nd column. The result of the restoration is shown in figure 5.25. The same restoration was attempted with the Wave Atom transform, and the learned dictionary gave a better result. The results are compared in table 5.5

Some notes should be made on this result. First, the training patches were made by selecting random patches from the same data set. This is somewhat unfair, and was done due to lack of available data. In addition, in order to achieve good results, I had to relax the minimization somewhat (i.e. solve (P1-σ) rather than (P1)). This improves the result for the learned dictionary, but a similar relaxation does not improve the result for the Wave Atom restoration. In the experiment, I set $\sigma = 5$. The experiment was carried out using the spgl1-package. Finally, we should note that this result is likely not extendable to cases with a very high ratio of samples missing to the same degree as the other bases, without altering the experiment parameters (at the very least, one would expect to need larger patches).

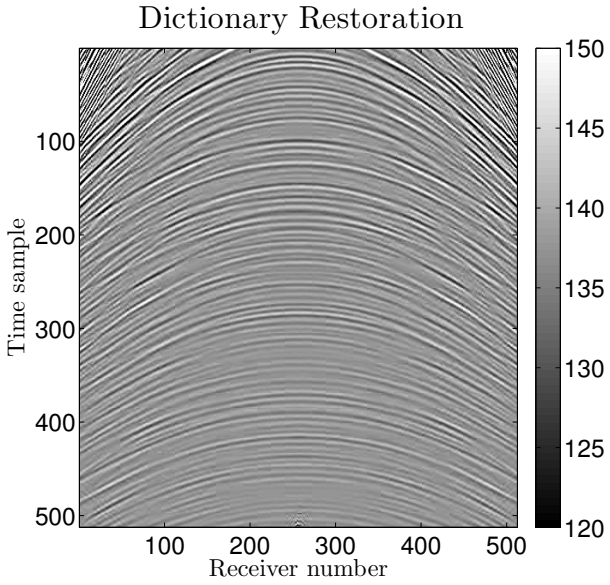


(a) Original dictionary

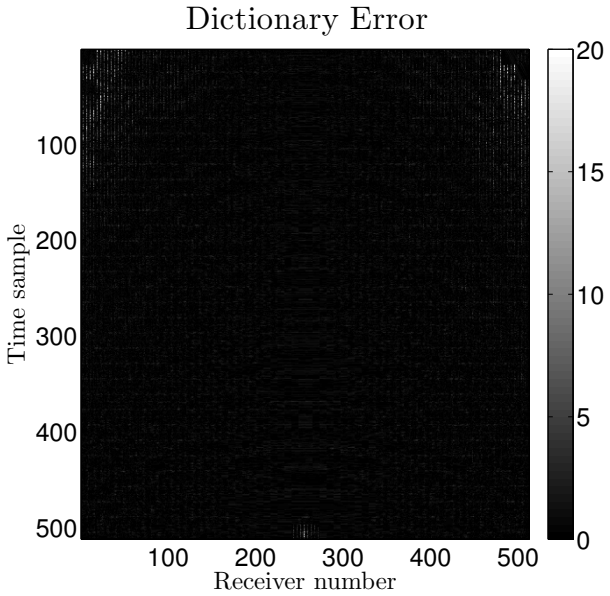


(b) Trained dictionary

Figure 5.24: The image show an initial dictionary of 16×16 pixels, and the result of training the dictionary to provide sparse representations of seismic data.



(a) Original dictionary



(b) Trained dictionary

Figure 5.25: The figure shows the result of inpainting a seismic data set using learned dictionaries.

Chapter 6

Conclusion and discussion

6.1 Summary

After briefly introducing classical digital signal theory in chapter 2, we studied in great detail various ways to construct sparsifying bases in chapter 3, from classics like the Fourier and the wavelet bases, to modern approaches such as the curvelet, which are not usually studied at a sub-graduate level. At the start of chapter 4, we introduced the general problem in filling in missing or corrupted information in a signal, in the form of linear inverse problems. In section 4.1.1 we demonstrated how sparsity could give very good results for signal restoration by experimenting with signal restoration in 1-D and 2-D. We noted, however that sparsity alone was not enough to ensure a high quality restoration, by performing experiments using wavelet based inpainting, with poor results. The rest of chapter 4 was then spent on introducing the theory of compressed sensing, in order to explain why some approaches work and some do not. We also developed some theory for good robust ways of sensing a 2-D signal, notably corollary 4.9.7 and the phase plots in figure 4.20.

Armed with the results and theory from the first chapters, we then highlighted some previous application of compressed sensing methods to areas of science related to computational physics. We then showcased some new attempted concepts. First, we showed that the traditional approach of partial samples in some basis does *not* necessarily restore all the statistical properties of the signal, highlighted by poor performance in restoring the Hurst exponent of a rough surface. This question was highlighted by Ma [99], but has not been explored in published literature. We also noted that by generating a dictionary with the "right" statistical properties, we could obtain much better results. The word "right" here is ambiguous; it simply means a basis which produces good results! We will get back to this point in the discussion. We then introduced the concept of super-resolution by inpainting the wavelet detail spaces of a signal using curvelet sparsity. This approach is not explored in the literature surveyed for this thesis, and outperforms other curvelet based super-resolution schemes, such as the one presented by Mallat and Guoshen [108]. This framework was then applied to the problem of filling in seismic data in the case where measurements are made with varying resolution. Finally, we once again approached the problem of seismic inpainting, this time using a trained dictionary. Using this method, we were able to reconstruct the signal with higher quality than using the frames explored earlier in the chapter.

It is relevant at this point to revisit the goals made at the outset of this project, stated in the introduction. We certainly developed a solid understanding of the current theoretical

results in CS, and were able to apply CS for simply reconstructions. We also spent some time on the formalism required to implement CS in 2-D, and performed CS reconstructions for fairly large data sets (512×512 samples), using methods which could easily scale to even much larger data sets, by applying fast operators of order $\mathcal{O}(N \log_2 N)$. While we did not reproduce the CS MRI results, we reproduces results in seismic imaging and molecular dynamics reconstructions, and developed a skillset where a reproduction of MRI results would be straight forward. This was not done, as it would not have produced any useful insights in itself. We were also able to study problems which have not been explored previously in CS literature, and suggested new methods for further investigation.

6.2 Discussion

The most useful end product of my thesis work, is a solid understanding of the principles of CS. By working through other published applications, I have learned a very useful and unique skill, which will undoubtedly prove itself useful in future work. I am certain that the understanding of how to apply CS to a problem, and the ability to understand when it can be applied, will be highly sought after in both industry and academia as the theory matures further and becomes more well known.

That being said, the work done in this thesis also holds potential. There are four key results which may, in time, be matured to the point of being suitable for stand-alone publications; the 2-D random sensing results, the investigation of rough surfaces, the approach to super-resolution and the applications of learned dictionaries to seismic inpainting. Given the short time span of a Master's degree, all of these results to be polished further, and some more understanding of the underlying mechanisms would still be useful to further explain the results in both super-resolution, Hurst exponent estimation and dictionary-based inpainting.

Compressed sensing is a relevant tool whenever data points are expensive to make (where expensive can refer both to the price of making the measurement equipment, or the time needed to make sufficient measurements), and can be used to great effect in many areas of science when applied correctly. One should however note the ambiguity in this statement. What does a "correct" application entail? Because of the large gap between the current theory and state-of-the-art applications, one must essentially try a compressed sensing approach on manufactured problems, much like we have done in this thesis, and decide on whether or not the application is suited for CS. We have also noted that while the a CS-based restoration way give a small error in terms of signal values, the statistical properties of the signal may be altered, a property which is largely overlooked in existing literature.

If one considers using a CS method for a particular problem, one must naturally find some representation which the relevant signal should be sparse in, whether it be some non-adaptive frame or a learned dictionary. One must then consider whether a sampling scheme suited for CS is possible in practice. A completely random sensing of seismic data using Gaussian sensing matrices would most likely perform better than the case of missing some fraction of traces, but such a sampling scheme is unrealistic in practice, and thus of little interest in this case. This also discourages us from trying to use wavelets as the sparsifying basis, as we noted that wavelets are not able to perform high quality inpainting. One then has to consider whether the available subsampling scheme will cause aliasing effects. In section 5.9 we used a equidistant set of missing traces in the data. This approach works for the learned dictionaries and the wave atom, but would give terrible results for a Fourier based reconstruction, as it would give aliasing

effects similar to those explored in section 4.3. We have not discussed how to avoid aliasing effects explicitly for the curvelets or wave atoms, as very little theory exists for this at the time of writing. We note, however, that a randomized sampling will usually avoid any major aliasing effects.

We also note here that while the performance guarantees give bounds in terms of the ℓ_p-norm error, the reconstructed signal might display significantly different statistical properties than that of the original surface, even when the ℓ_p-norm error is small. This is a feature often overlooked in existing literature, which may be of great importance to physicist looking to use this method.

6.3 Future work

For a physicist, I do not believe working on compressed sensing full time is currently the most useful way to spend their time at the outset of their career. As the methods are still not well known among physicist, you might and up with a solution, without having any relevant problems to apply it to. Rather, it is useful to be aware of this method, and then expose yourself to other areas of physics. Then, there is a good chance that applications will emerge. For instance, in many lab settings, many experiments will be greatly dictated by the time it takes to make measurements of sufficient quality. Experimentalists may need to reserve time to use equipment such as CT scanners or electron microscopes, which takes some time to collect a sufficient number of samples. Knowledge of CS may prove itself useful in such cases in order to cut down the time spent acquiring the needed samples. As noted by the applications to molecular dynamics, CS can also prove itself useful in a purely computational setting.

Immediate future work in CS will likely involve further studies of the original work in this thesis. Each of the highlighted results require further theoretical understanding. Outstanding questions are:

- For the random orthogonal sensing matrices, why are we seemingly not able to guarantee reconstructions for the range where $k \ll M$? What are the effects of changing to similar but slightly different schemes such as $A = R_T((Q_1 \otimes Q_2) \otimes (Q_3 \otimes Q_4))$, where Q_i are random orthogonalized matrices of size $N^{1/4} \otimes N^{1/4}$, or sampling from two different sets of Kronecker products $A = R_T^{(1)}(Q_1 \otimes Q_2) + R_T^{(2)}(Q_3 \otimes Q_4)$? The first suggestion may produce results of lower quality, but will have a faster implementation, while the second procedure may produce higher quality results at the cost of higher computation times.

- For the estimation of Hurst exponents based on partial data and learned dictionaries, more work should be done in order to understand why the difference based reconstruction works well, while the other approaches do not. The method should also be tested further on real data, which was not readily available during the work on this thesis.

- For the learned dictionaries applied to the seismic data inpainting problem, much more testing is needed. First, a larger amount of similar data sets should be obtained, in order to train the dictionary with patches from one group of data sets and then apply this dictionary to another group. A much larger scale test should also be run, similar to that in figure 5.7 for the non-adaptive frames.

There is also prospects of starting up a small group for CS activity at the University of Oslo, with collaborations between the physics and mathematics department, of which I hope to take

part. Apart from this, my future work in CS will depend largely on the applications which may appear as I move on to my graduate work in computational physics.

List of Abbreviations

BOS Bounded Orthonormal System

BP Basis pursuit

BPDN Basis Pursuit De-Noising

CDF Cohen-Daubechies-Feauveau

CRT Contiuous Ridgelet Transform

CS Compressed Sensing

CTCT Continuous Time Cosine Transform

CTFT Continuous Time Fourier Transform

DCT Discrete Cosine Transform

DCUT1 First Generation Discrete Curvelet Transform

DCUT2 2nd generation Discrete Curvelet Transform

DFT Discrete Fourier Transform

DRAT Discrete Radon Transform

DRIT Discrete Ridgelet Transform

DSP Digital Signal Processing

DWT Discrete Wavelet Transform

ERP Exact Recovery Property

ETF Equiangular Tight Frame

FBP Finite Back-projection

FFT Fast Fourier Transform

ICTCT Inverse Continuous Time Cosine Transform

IDFT Inverse Discrete Fourier Transform

IDWT Inverse Discrete Fourier Transforms

LIP Linear Inverse Problem

MIP Mutual Incoherence Property

MRA Multi-resolution Analysis

MRI Magnetic Resonance Imaging

MSE Mean Square Error

NSP Null Space Property

NSP-2 Alternate definition of the NSP

PSNR Peak Signal-to-Noise Ratio

RIP Restricted Isometry Property

RNSP Robust Null Space Property

SNR Signal-to-Noise Ratio

TV Total Variation

List of Figures

List of Tables

Bibliography

[1] E. J. Candès and M. B. Wakin. "An introduction to compressive sampling". In: *Signal Processing Magazine, IEEE* 25.2 (2008), pp. 21–30.

[2] M. Lustig et al. "Compressed sensing MRI". In: *Signal Processing Magazine, IEEE* 25.2 (2008), pp. 72–82.

[3] X. Andrade, J. N. Sanders, and A. Aspuru-Guzik. "Application of compressed sensing to the simulation of atomic systems". In: *Proceedings of the National Academy of Sciences* 109.35 (2012), pp. 13928–13933.

[4] J. N. Sanders et al. "Compressed Sensing for Multidimensional Spectroscopy Experiments". In: *The Journal of Physical Chemistry Letters* 3.18 (2012), pp. 2697–2702.

[5] F. J. Herrmann and G. Hennenfent. "Non-parametric seismic data recovery with curvelet frames". In: *Geophysical Journal International* 173.1 (2008), pp. 233–248.

[6] *The USC-SIPI Image Database*. Nov. 2013. URL: http://sipi.usc.edu/database.

[7] M. L. Boas. *Mathematical Methods in the Physical Sciences*. Third Edition. Wiley, 2006, pp. 356–358. ISBN: 978-0-471-19826-0.

[8] T. Lindstrøm. *Mathematical Analysis (Compendium)*. 2013. Chap. 7.

[9] B. A. Cipra. "The best of the 20th century: Editors name top 10 algorithms". In: *SIAM news* 33.4 (), pp. 1–2.

[10] D. C. Lay. *Linear Algebra and Its Applications*. Thrid Edition.

[11] W. Sweldens. "The lifting scheme: A construction of second generation wavelets". In: *SIAM Journal on Mathematical Analysis* 29.2 (1998), pp. 511–546.

[12] J. Sakurai. *Modern Quantum Mechanics*. 2nd Edition. Addison-Wesley, 2010. ISBN: 978-0805382914.

[13] E. J. Candès and D. L. Donoho. "Continuous curvelet transform: I. Resolution of the wavefront set". In: *Applied and Computational Harmonic Analysis* 19.2 (2005), pp. 162–197.

[14] E. J. Candès and D. L. Donoho. "Continuous curvelet transform: II. Discretization and frames". In: *Applied and Computational Harmonic Analysis* 19.2 (2005), pp. 198–222.

[15] E. J. Candès and D. L. Donoho. "New tight frames of curvelets and optimal representations of objects with piecewise C2 singularities". In: *Communications on pure and applied mathematics* 57.2 (2004), pp. 219–266.

[16] W. T. Freeman and E. H. Adelson. "The design and use of steerable filters". In: *IEEE Transactions on Pattern analysis and machine intelligence* 13.9 (1991), pp. 891–906.

[17] E. P. Simoncelli et al. "Shiftable multiscale transforms". In: *Information Theory, IEEE Transactions on* 38.2 (1992), pp. 587–607.

[18] T. S. Lee. "Image representation using 2D Gabor wavelets". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 18.10 (1996), pp. 959–971.

[19] D. L. Donoho. "Wedgelets: Nearly minimax estimation of edges". In: *Annals of statistics* (1999), pp. 859–897.

[20] X. Huo and D. Donoho. "Beamlets and multiscale image analysis". In: *Multiscale and Multiresolution Methods* (2002), pp. 149–196.

[21] S. Mallat and G. Peyré. "A review of bandlet methods for geometrical image representation". In: *Numerical Algorithms* 44.3 (2007), pp. 205–234.

[22] E. Le Pennec and S. Mallat. "Sparse geometric image representations with bandelets". In: *Image Processing, IEEE Transactions on* 14.4 (2005), pp. 423–438.

[23] G. Yu, S. Mallat, et al. "Sparse super-resolution with space matching pursuits". In: *SPARS'09-Signal Processing with Adaptive Sparse Structured Representations*. 2009.

[24] D. Labate et al. "Sparse multidimensional representation using shearlets". In: *Optics & Photonics 2005*. International Society for Optics and Photonics. 2005, 59140U–59140U.

[25] K. Guo and D. Labate. "Optimally sparse multidimensional representation using shearlets". In: *SIAM journal on mathematical analysis* 39.1 (2007), pp. 298–318.

[26] L. Demanet and L. Ying. "Wave atoms and sparsity of oscillatory patterns". In: *Applied and Computational Harmonic Analysis* 23.3 (2007), pp. 368–387.

[27] R. M. Willett and R. D. Nowak. "Platelets: a multiscale approach for recovering edges and surfaces in photon-limited medical imaging". In: *Medical Imaging, IEEE Transactions on* 22.3 (2003), pp. 332–350.

[28] Y. M. Lu and M. N. Do. "Multidimensional directional filter banks and surfacelets". In: *Image Processing, IEEE Transactions on* 16.4 (2007), pp. 918–931.

[29] M. N. Do and M. Vetterli. "The finite ridgelet transform for image representation". In: *Image Processing, IEEE Transactions on* 12.1 (2003), pp. 16–28.

[30] E. J. Candès. "Ridgelets: Theory and Applications". PhD thesis. Department of Statistics, Stanford University.

[31] A. Averbuch et al. *Fast Slant Stack: A notion of Radon transform for data in a Cartesian grid which is rapidly computible, algebraically exact, geometrically faithful and invertible*. Department of Statistics, Stanford University, 2001.

[32] J. Fadili, J.-L. Starck, et al. "Curvelets and Ridgelets." In: *Encyclopedia of Complexity and Systems Science* 3 (2009), pp. 1718–1738.

[33] *Frame of a vector space*. June 2014. URL: http://en.wikipedia.org/wiki/Frame_of_a_vector_space.

[34] R. J. Duffin and A. C. Schaeffer. "A class of nonharmonic Fourier series". In: *Transactions of the American Mathematical Society* (1952), pp. 341–366.

[35] J. Ma and G. Plonka. "The curvelet transform". In: *Signal Processing Magazine, IEEE* 27.2 (2010), pp. 118–133.

[36] R. Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), pp. 267–288.

[37] S. S. Chen, D. L. Donoho, and M. A. Saunders. "Atomic decomposition by basis pursuit". In: *SIAM journal on scientific computing* 20.1 (1998), pp. 33–61.

[38] *Shepp-Logan Reconstruction Phantom*. Nov. 2013. URL: http://www.cs.tut.fi/~comsens.

[39] *Seismic measurements presentation*. Mar. 2014. URL: http://www.cgg.com/popup_page.aspx?cid=1-24-165.

[40] *Sparse Spikes Deconvolution with L1 Pursuit*. Mar. 2014. URL: https://www.ceremade.dauphine.fr/~peyre/numerical-tour/tours/sparsity_2_sparsespikes_bp/#10.

[41] J.-L. Starck, D. L. Donoho, and E. J. Candès. "Very high quality image restoration by combining wavelets and curvelets". In: *International Symposium on Optical Science and Technology*. International Society for Optics and Photonics. 2001, pp. 9–19.

[42] *Survey of inpainting methods*. Mar. 2014. URL: http://math.univ-lyon1.fr/~masnou/fichiers/publications/survey.pdf.

[43] J. Fadili et al. "MCALab: Reproducible research in signal and image decomposition and inpainting". In: *IEEE Computing in Science and Engineering* 12.1 (2010), pp. 44–63.

[44] D. L. Donoho and X. Huo. "Uncertainty principles and ideal atomic decomposition". In: *Information Theory, IEEE Transactions on* 47.7 (2001), pp. 2845–2862.

[45] M. Elad, P. Milanfar, and R. Rubinstein. "Analysis versus synthesis in signal priors". In: *Inverse problems* 23.3 (2007), p. 947.

[46] Y. C. Eldar and G. Kutyniuk. *Compressed sensing*. Cambridge, 2012. ISBN: 978-1-107-00558-7.

[47] S. Foucart and H. Rauhut. *A mathematical introduction to compressive sensing*. Springer, 2013.

[48] J.-L. Starck, F. Murtagh, and J. M. Fadili. *Sparse image and signal processing: wavelets, curvelets, morphological diversity*. Cambridge University Press, 2010.

[49] E. Candès and J. Romberg. "Sparsity and incoherence in compressive sampling". In: *Inverse problems* 23.3 (2007), p. 969.

[50] R. Coifman, F Geshwind, and Y. Meyer. "Noiselets". In: *Applied and Computational Harmonic Analysis* 10.1 (2001), pp. 27–44.

[51] T. Tuma, P. Hurley, et al. "On the incoherence of noiselet and Haar bases". In: *SAMPTA'09, International Conference on Sampling Theory and Applications*. 2009.

[52] *A geometric intuition for the null space property*. June 2014. URL: http://dustingmixon.wordpress.com/2013/10/28/a-geometric-intuition-for-the-null-space-property/.

[53] L. Welch. "Lower bounds on the maximum cross correlation of signals (Corresp.)" In: *Information Theory, IEEE Transactions on* 20.3 (1974), pp. 397–399.

[54] S. A. Gershgorin. "Uber die abgrenzung der eigenwerte einer matrix". In: . 6 (1931), pp. 749–754.

[55] L. Gan et al. "Analysis of the statistical restricted isometry property for deterministic sensing matrices using Stein's method". In: *Preprint* (2009).

[56] D. Needell and J. A. Tropp. “CoSaMP: Iterative signal recovery from incomplete and inaccurate samples”. In: *Applied and Computational Harmonic Analysis* 26.3 (2009), pp. 301–321.

[57] M. A. Davenport. “Random observations on random observations: Sparse signal acquisition and processing”. PhD thesis. Citeseer, 2010.

[58] E. J. Candès and T. Tao. “Decoding by linear programming”. In: *Information Theory, IEEE Transactions on* 51.12 (2005), pp. 4203–4215.

[59] E. J. Candès, J. K. Romberg, and T. Tao. “Stable signal recovery from incomplete and inaccurate measurements”. In: *Communications on pure and applied mathematics* 59.8 (2006), pp. 1207–1223.

[60] E. J. Candès. “The restricted isometry property and its implications for compressed sensing”. In: *Comptes Rendus Mathematique* 346.9 (2008), pp. 589–592.

[61] S. Foucart and M.-J. Lai. “Sparsest solutions of underdetermined linear systems via q-minimization for 0¡ q 1”. In: *Applied and Computational Harmonic Analysis* 26.3 (2009), pp. 395–407.

[62] S. Foucart. “A note on guaranteed sparse recovery via 1-minimization”. In: *Applied and Computational Harmonic Analysis* 29.1 (2010), pp. 97–103.

[63] T. T. Cai, L. Wang, and G. Xu. “Shifting inequality and recovery of sparse signals”. In: *Signal Processing, IEEE Transactions on* 58.3 (2010), pp. 1300–1308.

[64] T. T. Cai, L. Wang, and G. Xu. “New bounds for restricted isometry constants”. In: *Information Theory, IEEE Transactions on* 56.9 (2010), pp. 4388–4394.

[65] Q. Mo and S. Li. “New bounds on the restricted isometry constant δ2k”. In: *Applied and Computational Harmonic Analysis* 31.3 (2011), pp. 460–468.

[66] J. Andersson and J.-O. Strömberg. “On the theorem of uniform recovery of structured random matrices”. In: *Preprint* (2013).

[67] M. E. Davies and R. Gribonval. “Restricted isometry constants where sparse recovery can fail for”. In: *Information Theory, IEEE Transactions on* 55.5 (2009), pp. 2203–2214.

[68] T. T. Cai and A. Zhang. “Sharp RIP bound for sparse signal and low-rank matrix recovery”. In: *Applied and Computational Harmonic Analysis* 35.1 (2013), pp. 74–93.

[69] J. M. Renes et al. “Symmetric informationally complete quantum measurements”. In: *arXiv preprint quant-ph/0310075* (2003).

[70] T. Strohmer and R. W. Heath Jr. “Grassmannian frames with applications to coding and communication”. In: *Applied and computational harmonic analysis* 14.3 (2003), pp. 257–275.

[71] P. Xia, S. Zhou, and G. B. Giannakis. “Achieving the Welch bound with difference sets”. In: *Information Theory, IEEE Transactions on* 51.5 (2005), pp. 1900–1907.

[72] S. Mendelson, A. Pajor, and N. Tomczak-Jaegermann. “Uniform uncertainty principle for Bernoulli and subgaussian ensembles”. In: *Constructive Approximation* 28.3 (2008), pp. 277–289.

[73] R. Baraniuk et al. “A simple proof of the restricted isometry property for random matrices”. In: *Constructive Approximation* 28.3 (2008), pp. 253–263.

[74] H. Rauhut. "Compressive sensing and structured random matrices". In: *Theoretical foundations and numerical methods for sparse recovery* 9 (2010), pp. 1–92.

[75] H. Rauhut and R. Ward. "Sparse Legendre expansions via 1-minimization". In: *Journal of approximation theory* 164.5 (2012), pp. 517–533.

[76] E. J. Candès and Y. Plan. "A probabilistic and RIPless theory of compressed sensing". In: *Information Theory, IEEE Transactions on* 57.11 (2011), pp. 7235–7254.

[77] E. J. Candès and T. Tao. "Near-optimal signal recovery from random projections: Universal encoding strategies?" In: *Information Theory, IEEE Transactions on* 52.12 (2006), pp. 5406–5425.

[78] M. Rudelson and R. Vershynin. "On sparse reconstruction from Fourier and Gaussian measurements". In: *Communications on Pure and Applied Mathematics* 61.8 (2008), pp. 1025–1045.

[79] M. Cheraghchi, V. Guruswami, and A. Velingker. "Restricted isometry of Fourier matrices and list decodability of random linear codes". In: *SIAM Journal on Computing* 42.5 (2013), pp. 1888–1914.

[80] H. Rauhut. "Circulant and Toeplitz matrices in compressed sensing". In: *arXiv preprint arXiv:0902.4394* (2009).

[81] H. Rauhut and R. Ward. "Sparse recovery for spherical harmonic expansions". In: *arXiv preprint arXiv:1102.4097* (2011).

[82] M. F. Duarte and R. G. Baraniuk. "Kronecker product matrices for compressive sensing". In: *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE. 2010, pp. 3650–3653.

[83] S. Jokar. "Sparse recovery and Kronecker products." In: *CISS*. 2010, pp. 1–4.

[84] M. F. Duarte and R. G. Baraniuk. "Kronecker compressive sensing". In: *Image Processing, IEEE Transactions on* 21.2 (2012), pp. 494–504.

[85] R. Thompson. "Principal submatrices IX: Interlacing inequalities for singular values of submatrices". In: *Linear Algebra and its Applications* 5.1 (1972), pp. 1–12.

[86] A. Eftekhari, M. Babaie-Zadeh, and H. Abrishami Moghaddam. "Two-dimensional random projection". In: *Signal Processing* 91.7 (2011), pp. 1589–1603.

[87] M. E. Muller. "A note on a method for generating points uniformly on n-dimensional spheres". In: *Communications of the ACM* 2.4 (1959), pp. 19–20.

[88] B. Adcock et al. "Breaking the coherence barrier: asymptotic incoherence and asymptotic sparsity in compressed sensing". In: *arXiv preprint arXiv:1302.0561* (2013).

[89] E. Candèes and J. Romberg. "l1-magic: Recovery of sparse signals via convex programming". In: *URL: www. acm. caltech. edu/l1magic/downloads/l1magic. pdf* 4 (2005).

[90] E. van den Berg and M. P. Friedlander. "Probing the Pareto frontier for basis pursuit solutions". In: *SIAM Journal on Scientific Computing* 31.2 (2008), pp. 890–912.

[91] E Van Den Berg and M. Friedlander. "SPGL1: A solver for large-scale sparse reconstruction". In: *Online: http://www. cs. ubc. ca/labs/scl/spgl1* (2007).

[92] H. Hassanieh et al. "Nearly optimal sparse Fourier transform". In: *Proceedings of the 44th symposium on Theory of Computing*. ACM. 2012, pp. 563–578.

[93] S. S. Vasanawala et al. "Practical parallel imaging compressed sensing MRI: Summary of two years of experience in accelerating body MRI of pediatric patients". In: *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*. IEEE. 2011, pp. 1039–1043.

[94] J. Bobin, J.-L. Starck, and R. Ottensamer. "Compressed sensing in astronomy". In: *Selected Topics in Signal Processing, IEEE Journal of* 2.5 (2008), pp. 718–726.

[95] E. van den Berg and M. P. Friedlander. *Spot – A Linear-Operator Toolbox*. June 2014. URL: http://www.cs.ubc.ca/labs/scl/spot/.

[96] W. Tang, J. Ma, and F. J. Herrmann. "Optimized compressed sensing for curvelet-based seismic data reconstruction". In: *preprint* 280 (2009).

[97] F. J. Herrmann, P. Moghaddam, and C. C. Stolk. "Sparsity-and continuity-promoting seismic image recovery with curvelet frames". In: *Applied and Computational Harmonic Analysis* 24.2 (2008), pp. 150–173.

[98] R. Shahidi et al. "Application of randomized sampling schemes to curvelet-based sparsity-promoting seismic data recovery". In: *Geophysical Prospecting* 61.5 (Sept. 2013), pp. 973–997.

[99] J. Ma. "Compressed sensing for surface characterization and metrology". In: *Instrumentation and Measurement, IEEE Transactions on* 59.6 (2010), pp. 1600–1615.

[100] K Yabana et al. "Real-time, real-space implementation of the linear response time-dependent density-functional theory". In: *physica status solidi (b)* 243.5 (2006), pp. 1121–1138.

[101] J. B. Bassingthwaighte and G. M. Raymond. "Evaluation of the dispersional analysis method for fractal time series". In: *Annals of biomedical engineering* 23.4 (1995), pp. 491–505.

[102] I. Simonsen, A. Hansen, and O. M. Nes. "Determination of the Hurst exponent by use of wavelet transforms". In: *Physical Review E* 58.3 (1998), p. 2779.

[103] A. B. Suksmono. "Reconstruction of fractional Brownian motion signals from its sparse samples based on compressive sampling". In: *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*. IEEE. 2011, pp. 1–6.

[104] R. Keys. "Cubic convolution interpolation for digital image processing". In: *Acoustics, Speech and Signal Processing, IEEE Transactions on* 29.6 (1981), pp. 1153–1160.

[105] J. Yang et al. "Image super-resolution as sparse representation of raw image patches". In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. 2008, pp. 1–8.

[106] J. Yang et al. "Image super-resolution via sparse representation". In: *Image Processing, IEEE Transactions on* 19.11 (2010), pp. 2861–2873.

[107] P. Sen and S. Darabi. "Compressive image super-resolution". In: *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*. IEEE. 2009, pp. 1235–1242.

[108] S. Mallat and G. Yu. "Super-resolution with sparse mixing estimators". In: *Image Processing, IEEE Transactions on* 19.11 (2010), pp. 2889–2900.

[109] A. A. Patil, R. Singhai, and J. Singhai. "Curvelet transform based super-resolution using sub-pixel image registration". In: *Computer Science and Electronic Engineering Conference (CEEC), 2010 2nd*. IEEE. 2010, pp. 1–6.

[110] D. K. Davis and R. C. Roy. "Hybrid Super Resolution using SWT and CT". In: *International Journal of Computer Applications* 59.7 (2012).

[111] M. Elad et al. "Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)". In: *Applied and Computational Harmonic Analysis* 19.3 (2005), pp. 340–358.

[112] M.-J. Fadili and J.-L. Starck. "Em algorithm for sparse representation-based image inpainting". In: *Image Processing, 2005. ICIP 2005. IEEE International Conference on*. Vol. 2. IEEE. 2005, pp. II–61.

[113] M. Elad and M. Aharon. "Image denoising via sparse and redundant representations over learned dictionaries". In: *Image Processing, IEEE Transactions on* 15.12 (2006), pp. 3736–3745.

[114] W. Dong et al. "Sparsity-based image denoising via dictionary learning and structural clustering". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE. 2011, pp. 457–464.

[115] J. Mairal, M. Elad, and G. Sapiro. "Sparse representation for color image restoration". In: *Image Processing, IEEE Transactions on* 17.1 (2008), pp. 53–69.

[116] W. Dong et al. "Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization". In: *Image Processing, IEEE Transactions on* 20.7 (2011), pp. 1838–1857.

[117] H. Rauhut, K. Schnass, and P. Vandergheynst. "Compressed sensing and redundant dictionaries". In: *Information Theory, IEEE Transactions on* 54.5 (2008), pp. 2210–2219.

[118] S. Ravishankar and Y. Bresler. "MR image reconstruction from highly undersampled k-space data by dictionary learning". In: *Medical Imaging, IEEE Transactions on* 30.5 (2011), pp. 1028–1041.

[119] T. Goldstein and S. Osher. "The split Bregman method for L1-regularized problems". In: *SIAM Journal on Imaging Sciences* 2.2 (2009), pp. 323–343.

[120] M. Aharon, M. Elad, and A. Bruckstein. "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation". In: *Signal Processing, IEEE Transactions on* 54.11 (2006), pp. 4311–4322.

[121] R. Rubinstein, M. Zibulevsky, and M. Elad. "Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit". In: *CS Technion* (2008), p. 40.

[122] *Efficient MATLAB implementation of K-SVD*. May 2014. URL: http://www.cs.technion.ac.il/~ronrubin/software.html.