Compressed Sensing and Approximate Message Passing for the Single-Pixel Camera

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona of

Universitat Politècnica de Catalunya

and in the Fakultät für Elektrotechnik und Informationstechnik of

Technische Universität Wien

by

Jordi Biosca Caro

In partial fulfilment

of the requirements for the degree in

TELECOMMUNICATION SYSTEMS ENGINEERING

Advisors: Norbert Görtz and Gregori Vazquez Grau

Barcelona, June 2018

Abstract

This thesis is based on the technique of Compressed Sensing and the study of the algorithms of Approximate Message Passing for the single-pixel camera. This thesis has been developed in the institute of telecommunication in the Technische Universität Wien (TU Wien) led by the research group of the professor Norbert Görtz. In the beginning of the thesis it is explained what a single-pixel camera is and it also talks about what compressed sensing is, later on, two necessarily iterative schemes (Iterative Hard Thresgolding i Iterative Soft Thresholding) are defined to understand the algorithms of Approximate Message Passing (AMP) and its Bayesian derivation (Bayesian Approximate Message Passing, BAMP). The theoretical explanation of the operation of these algorithms is implicit in the thesis, moreover, AMP and BAMP are implemented in Matlab code. This implementation allows to see the behaviour of these algorithms in different scenarios to fully understand the differences between them.

Resum

Aquest treball es basa en la tècnica del compressed sensing i l'estudi dels algoritmes d'Approximate Message Passing per la càmera d'un sol píxel. És un treball desenvolupat a l'institut de telecomunicacions de la Technische Universität Wien (TU Wien) liderat pel grup de recerca del professor Norbert Görtz, el qual ve liderant la recerca en aquest sector des de fa diversos anys. En l'inici del treball s'explica que és la càmera d'un sol píxel i també es parla de què és el compressed sensing, més endavant s'expliquen dos esquemes iteratius (Iterative Hard Thresgolding i Iterative Soft Thresholding) necessaris per entendre els algoritmes de Approximate Message Passing (AMP) i la seva derivant bayesiana (Bayesian Approximate Message Passing, BAMP). L'explicació teòrica del funcionament d'aquests algoritmes està implícit en el projecte, a més a més, aquests dos algoritmes estan implementats en codi Matlab. Aquesta implementació permet veure quin és el comportament d'aquests algoritmes per diferents escenaris i permet entendre l'algoritme de AMP i BAMP.

Resumen

Este trabajo se basa en la técnica del Compressed Sensing y en el estudio de los algoritmos de Approximate Message Passing para la cámara de un solo pixel. Es un trabajo desarrollado en el instituto de telecomunicaciones de la Technische Universität Wien (TU Wien) liderado por el grupo de investigación del profesor Norbert Görtz. En el principio del trabajo se explica que es la cámara de un solo pixel y también se habla de que es el Compressed Sensing, más adelante se explican dos esquemas iterativos (Iterative Hard Thresgolding i Iterative Soft Thresholding) necesarios para entender los algoritmos de Approximate Message Passing (AMP) y su derivada bayesiana (Bayesian Approximate Message Passing, BAMP). La explicación teórica del funcionamiento de estos algoritmos es implícita en el proyecto, además, estos dos últimos están implementados en código Matlab. Dicha implementación permite ver cuál es el comportamiento de estos algoritmos en diferentes escenarios y permite entender el algoritmo de AMP y su derivada bayesiana, BAMP.

Dedication

First of all, I would like to thank professor Norbert Görtz from the Technische Universität of Wien for being so kind and helpful during all the project in my stay in Vienna. He introduced me in the field of compressed sensing and approximate message passing algorithms. Whenever I had a doubt or a question he found the moment to solve it and explained to me. I would also like to thank all the patience and eases my supervisor in the Universitat Politècnica de Catalunya, Gregori Vazquez, gave me during this time.

Finally, a personal thank to all the people that has been supporting me during these months with special mention to my family and friends both in Barcelona and Vienna.

Revision history and approval record

Revision	Date	Purpose
0	30/05/2018	Document creation
1	20/06/2018	Document revision by Norbert Görtz
2	27/06/2018	Document revision by Gregori Vazquez

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Jordi Biosca Caro	jordi.bc94@gmail.com
Norbert Görtz	norbert.goertz@nt.tuwien.ac.at
Gregori Vazquez Grau	gregori.vazquez@upc.edu

Written by:		Reviewed and approved by:	
Date	30/05/2018	Date	
Name	Jordi Biosca Caro	Name	Norbert Görtz & Gregori Vazquez
Position	Project Author	Position	Project Supervisors

Table of contents

List of Figures

1. Introduction

This project describes what the Single-Pixel Camera is and talks about the compressed sensing and how important are sparse signals to allow the implementation of different algorithms such as Iterative Soft Thresholding (IST), Iterative Hard Thresholding (IHT), Approximate Messape Passing (AMP) and Bayesian Approximate Message Passing (BAMP) to recover images with a sample rating below the Nyquist-Shannon theorem. In this section there are explained which are the objectives of the thesis and the work plan.

1.1. Objectives

The main objective of this project is to find in the compressed sensing topic, also known as compressive sensing, compressive sampling or sparse sampling, the different algorithms and methods of work that are normally used. Compressed sensing (CS) is a signal processing technique for efficiently acquiring and reconstructing signal, by finding solutions to undetermined linear systems. The goal is to study some of these algorithms, to understand their operating and validate them to achieve optimal results. Nowadays, the fields involving the compressed sensing are in a wide ranging, such as signal processing, computational mathematics or sparse sampling. Its broad scope and generality has enabled several CS approximations in signal processing and compressing, solution of inverse problems, design of radiating systems, radar, through-the-wall imaging and antenna characterization. Imaging techniques also have a strong relation with CS include coded aperture and computational photography. In this thesis, we will talk about these relations, we will use the discrete cosine transform (DCT) as a way to obtain sparse signals and then apply different algorithms to see its behaviour.

1.2. Work plan

The work plan has followed the dates proposed in either the work proposal and critical review despite some modifications were made in the critical review after I decided to do the talk of the thesis in Technische Universität Wien instead of the Universitat Politècnica de Catalunya. So, finally the work packages of the project have been structured as follows:

1.2.1. Work Packages

Project: Introduction to the Single-Pixel Camera algorithm	WP ref: WP1	
Major constituent: Study		
Short description: Introduction to the Single-Pixel Camera. How it works? How can be applied? How signal processing is involved? Which are the possibilities of the technology? The first code version must be studied. It should be able to simulate the Single-Pixel Camera algorithm.	Planned start date: 01/03/2018 Planned end date: 09/03/2018	
	Start event: 01/03/2018 End event: 09/03/2018	
Internal task T1: Introduction to the Single-Pixel Camera Internal task T2: Analysis of the algorithm	Deliverables: NA	Dates: NA

Project: Refinement	WP ref: WP2	
Major constituent: Code implementation		
Short description: The first code version must be done. It should be able to identify the main characteristics of the Gaussian form of the signal, its DCT and the histogram.	Planned start date: 12/03/2018 Planned end date: 28/03/2018	
	Start event: 12/03/2018 End event: 28/03/2018	
Internal task T1: First code version implementations Internal task T2: Validations	Deliverables: Reporting	Dates: NA

Project: Study and algorithms implementation	WP ref: WP3
Major constituent: Study and algorithms implementation	
Short description: Study how data can be recovered. Implement all the necessary to recover the signal and find the best solution	Planned start date: 26/03/2018 Planned end date: 26/05/2018
	Start event: 26/03/2018

depending on the complexity, the noise, etc of the Single-Pixel Camera algorithm.	End event: 12/06/2018	
Internal task T1: Study and algorithms implementation Internal task T2: Validations and tests	Deliverables: Reporting	Dates: NA

Project: Documentation	WP ref: WP4	
Major constituent: Documentation		
Short description: To complete all the documentation required in the final bachelor thesis.	Planned start date: 01/03/2018 Planned end date: 10/07/2018	
	Start event: 01/03/2018 End event: 28/06/2018	
Internal task T1: Thesis proposal Internal task T2: Critical review Internal task T3: Thesis memory and revision	Deliverables: Documentation	Dates: Depending on the document

1.2.2. Gantt diagram

Where each task has been developed in the specified dates. Note that, although in the beginning of the project I was focused on the literature and study of the principles in the topic, the longest part was the implementation and validation of the algorithm as we can see in the next Gantt diagram:

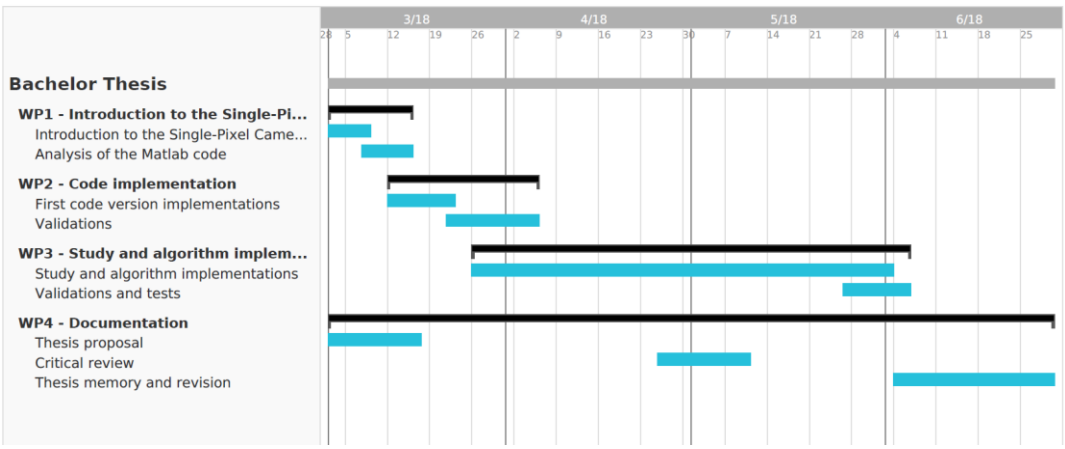


Figure 1. Gantt diagram of the thesis

2. State of the art of the technology used or applied in this thesis:

In this section there is a review of what a single-pixel camera is and an introduction to the compressed sensing. This, is used as presentation of the main topics of the thesis and it contains the main features one needs to know to fully understand the technologies explained in the following sections. Firstly, it is explained what a single-pixel camera is, its advantages and disadvantages. Secondly, there is an exhaustive analysis of the compressed sensing technique.

2.1. The Single-Pixel Camera

Nowadays, the standard digital cameras use a large number of photo sensors to determine the amount of light in each area of the image, we refer these areas as pixels. Thanks to the maturity of technologies (CCD and CMOS) is possible to obtain, for example, 12.000.000 pixels in a 12-megapixel camera because they work very well for optical light and is relatively cheap. However, these technologies only work in the optical range and for other kind of illumination, such as infrared or ultraviolet cameras, it is more difficult and expensive to produce.

At this point, one possible solution is to use a single-pixel camera. The single-pixel camera works with just one light sensor able to measure the entire image, this allows the use of one really good light sensor instead of the millions very cheap ones. What this sensor exploits it is the technique of compressed sensing. As we will see in the next sections, compressed sensing is a technique able to recover signals below the Nyquist rate.

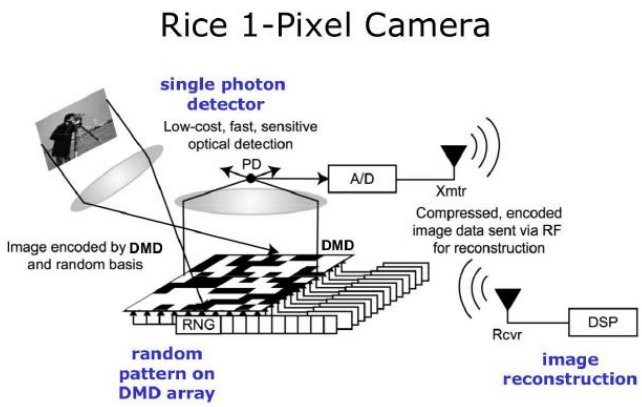


Figure 2. Single-pixel camera structure

The functioning of the single-pixel is to reflect the image to a digital micro mirror device (DMD) through a lens onto a photodiode. The DMD used is an array of microscopic mirrors which can be individually titled. These mirrors are titled in such a way that some of the pixels are focused onto a light absorber and the others to the photodiode. If we do this procedure thousands of times and

measuring the intensity of the light for each different mirror, we are able to physically realize the sampling matrix of a compressed sensing system.

On the one hand, the advantage in this case is that any radiation can be reflected from a mirror, including UV and IR, then this allows to create these kind of cameras much cheaper than is currently available. On the other hand, the main drawback is that the measurements must be taken in series rather than all at once as in a traditional camera. For example, in a 1024x1024 image we need 25% of the pixels for a total of about 260.0000 total pixels. The DMD can move positions in (worst case) 20μs, allowing the entire image to be captured in 5,25s. However, optical sensors stabilize much faster (20 ns for IR, 4ns for optical) so there is potential for imaging to become faster if the switching speed of the mirrors increases. Despite this situation even at this speed the camera could be used for many non-video applications.

If we take a closer look of how it works, as explained in [1], the object is sensed through a set of random patterns – a different one for each of the measurements. Those measurements with different random patterns are used to computationally reconstruct the scene. The implemented code of the thesis uses this algorithm as we will see in the next sections of the thesis.

2.2. Compressed Sensing

First of all, it is important to define what compressed sensing is and a proper definition could be: a technique to sample compressible signals below the Nyquist rate, whilst still allowing near optimal reconstruction of the signal.

Compressed Sensing (CS) is an attractive, rapidly growing field that has captivated considerable attention in electrical engineering, applied mathematics, statistics and computer science. Since its initial introduction several years ago, a huge flood of results have been obtained, both of theoretical and practical nature. CS offers a framework for simultaneous sensing and compression of finite-dimensional vectors, that relies on linear dimensionality reduction. Quite surprisingly, it predicts that sparse high-dimensional signals can be recovered from highly incomplete measurements by using effective algorithms.

In electrical engineering the use of a band-limited signal, with f_g as the highest frequency of the signal needs a sampling rate $f_s > 2f_g$ as explained for the Nyquist-Shannon sampling theorem. Therefore, with a signal that consists of a few non-zero samples, e.g., single short impulse, the result is a very large (almost infinite) bandwidth that can cause problems of memory in terms of computations. If instead of this, one uses a band-limited discrete time signal, e.g., a Discrete Fourier Transform (DFT) that produces few non-zero coefficients, then we obtain what is called a

sparse signal in the Fourier-domain. During the project the tests will be developed in images so, it will be used the two-dimensional Discrete Cosine Transform (DCT) which leaves few dominant coefficients, those are quantized and stored and the other ones are dropped (JPEG source-coding). In conclusion, it is possible to have an accurate signal representation with fewer coefficients than the originally sampled.

The question is why sparse signal models are so popular?

2.2.1. Problem setting: compressed sensing for sparse signals

On the one hand, we have a n −dimensional signal $x \in \mathbb{R}^n$ that is also sparse after some linear transformation (DCT for example). Then we assume that x is a *s*-sparse signal, i.e., it has at most *s* non-zero components (sparsity level). Taking $m \ll n$ linear measurements according to the next expression

$$y_{[m]} = A_{[m\times n]} \cdot x_{[n]} + w_{[m]} \quad \text{with} \quad m \ll n$$
$$\boldsymbol{y} = \boldsymbol{Ax} + \boldsymbol{w} \quad \text{(blond means vector notation)} \tag{1}$$

the matrix A is assumed to have its full possible rank m and its components are independently drawn realizations of a real random variable, e.g., has a Gaussian[1] distribution. Moreover, the matrix column vectors $A_j, j = 1, \dots, n$ (each of dimensions $m \times 1$) are assumed to have zero mean and be normalized[2] to unit l_2-norm[3], i.e.,

$$A = \{A_1, A_2, \dots, A_n\} \tag{2}$$

with,

[1] In [2], [3] systematic designs of measurement matrices have been investigated by state evolution; in this thesis we stick for simplicity to the traditional "random Gaussian" designs for the measurement matrix A but I would like to point out that AMP and BAMP will also work for non-Gaussian designs.
[2] When the measurement matrix A is defined by the system designer, the column-normalization can always be implemented; it simplifies notation. The normalization is, however, not necessarily required.
[3] The l_p −norm of a vector $x = \{x_j, j = 1,2, \dots, n\}$ is defined by

$$\left||x|\right|_p = (\sum_{j=!}^{n} |x_j|^p)^{\frac{1}{p}}, \text{ with p=1,2,...,}$$

and the "maximum-norm" is defined as

$$\left||x|\right|_\infty = \max_j |x_j|,$$

with x_j the components of the vector.

$$\|A_j\|_2 = 1 \; \forall \, j \tag{3}$$

The measurements are assumed to be affected by Gaussian noise that is modelled by the addition of the $m-$ dimensional noise vector $w \equiv \{w_j, j = 1, \dots, m\}$. The components w_j of the noise vector are assumed to be independent and identically distributed Gaussian with variance $\sigma^2 > 0$[4].

The problem is to find the vector x given the measurements y and the matrix A. As the number m of measurements in (1) is smaller than the number n of unknowns in x, the problem is undetermined: it is impossible to simply invert (1), even if there was no measurement of noise. One can try, however, to form a good estimate of x, and to do this one can exploit extra information about that x that may be available:

1) If x is sparse the solution $\hat{x}$ (estimated solution) would be supposed to contain few non-zero components.
2) The vector x might be known to have a large number of components that take values ± 1.
3) The probability distribution $p_x(x)$ (or a mixed distribution/density) of x might be known.

In what follows we will focus on case 3). In practice, the given $p_x(x)$ may, however, not be the true distribution but it may be a good model for it or even only be useful to find a good solution to the problem of estimating x from a given observation vector y. By choosing a suitable distribution $p_x(x)$, the other two cases 1) and 2) are also covered. If, for instance, a signal of dimension n is known to have $s \ll n$ non-zero components (this the sparse case 1), we could set the probability $\epsilon = (n - s)/n$ (to have a zero-component) and define the prior distribution (in fact a probability density function (pdf) with a delta-function that centres the probability mas ϵ at the single value $x = 0$) for the components x_j of x according to

$$p_{x_j}(x_j) = \epsilon\delta(x_j) + (1 - \epsilon) f_{X_j}(x_j) \; \forall j = 1, \dots, n \tag{4}$$

And assume that the components of x are all independent. The function $f_{X_j}(x_j)$ in (4) can be any probability density function, corresponding, e.g., to a uniform or a Gaussian distribution.

In case 2), the vector components might take the values +1 or -1 with equal probabilities of $\epsilon/2$ while the components take values from some other distribution $f_{X_j}(x_j)$ with probability $1 - \epsilon$. The corresponding pdf would read

[4] As here is the first time we talk about noise, from now on we define that components w_j of the noise vector are assumed to be independent and identically distributed Gaussian with variance $\sigma^2 > 0$

$$p_{X_j}(x_j) = \frac{\epsilon}{2}\,\delta(x_j + 1) + \frac{\epsilon}{2}\,\delta(x_j - 1) + (1 - \epsilon) f_{X_j}(x_j) \qquad \forall j = 1, \ldots, n \tag{5}$$

2.2.2. Estimation problem

As a formal, deterministic inversion of the measurement equation is impossible and we have to resort to an estimation of the signal vector, an optimization criterion must be defined. A very common approach from estimation theory is to pick the signal vector $\hat{x}$ as a solution that minimizes the expectation of the mean squared error, given the observation y. With our notation this is

$$\hat{x} = argmin_{\hat{x}}\, \mathbb{E}_{X,W}\{\, ||X - \hat{x}||_2^2 \mid Y = y \,\} \tag{6}$$

In our problem setting, the measurement vector y is deterministically known, so it appears as a condition in the expectation in (6). Moreover, the expectation is taken over two random components (indicated by capital letters) which are the measurement noise vector W and the unknown signal vector X for which we want to find an estimate $\hat{x}$. The solution to (6) is well-known from estimation theory and easy to derive: partial derivatives of (6) are taken for the vector components $\hat{x}_j$ of $\hat{x}$ and those derivatives are set to zero. The result equals

$$\hat{x} = \mathbb{E}_{X,W}\{X | Y = y\}, \tag{7}$$

i.e., the best estimate of x is the conditional expectation of the signal vector, given the observation y.

To understand why it is difficult to evaluate (7) for large signal dimension n, we write out the expectation using Bayes rule:

$$\hat{x} = \int_{\mathbb{R}_n} x\, p_{X|Y}(x|y)dx = \frac{1}{p_Y(y)} \int_{\mathbb{R}_n} x \underbrace{p_{Y|X}(y|x)}_{measurament\ noise} \underbrace{p_X(x)}_{signal\ prior} dx \tag{8}$$

In (8), $p_{X|Y}(x|y)$ is the conditional probability density function (pdf) of the signal vector x given an observation y (posterior pdf); for the second equality, Bayes rule was used (i.e., $p_{X|Y}(x \mid y) = p_{Y|X}(y \mid x)p_X(x)/p_Y(y)$). The pdf $p_X(x)$ is the signal prior and the pdf $p_{Y|X}(y \mid x)$ describes the noisy measurement process. For independent, Gaussian noise components w_j with variance σ^2 we obtain

$$p_{Y|X}(y \mid x) = \prod_{k=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} e^{-(y_k-(Ax)_k)^2/(2\sigma^2)} \tag{9}$$

When we write $(Ax)_k$ for the k-th component of the vector that results from the matrix multiplication Ax. Note that using (1) the k-th noise component is explicitly given by $w_k = y_k - (Ax)_k$. Re-introducing vector notation in the exponent, the pdf in (9) can also be written as follows:

$$\begin{aligned} p_{Y|X}(y \mid x) &= \frac{1}{\left(\sqrt{2\pi}\sigma\right)^m} \exp\left(-\sum_{k=1}^{m} \frac{(y_k-(Ax)_k)^2}{2\sigma^2}\right) \\ &= \frac{1}{\left(\sqrt{2\pi}\sigma\right)^m} \exp\left(-\frac{1}{2\sigma^2}\left|\left|y-Ax\right|\right|_2^2\right) \end{aligned} \tag{10}$$

If the components of the signal vector x are also independent, and each of the components has the pdf $p_{X_j}(x_j)$, we can write the prior pdf according to

$$p_X(x) = \prod_{j=1}^{n} p_{X_j}(x_j) \tag{11}$$

The unconditional pdf $p_Y(y)$ of the observations can be computed by the marginalization

$$p_Y(y) = \int_{\mathbb{R}_n} p_{\boldsymbol{Y}|\boldsymbol{X}}(y|x) p_{\boldsymbol{X}}(x) dx \tag{12}$$

involving the same pdfs as in (9) and (11).

We insert (9) and (11) into (8) to obtain

$$\hat{x} = \frac{1}{p_Y(y)} \int_{\mathbb{R}_n} x \prod_{k=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} e^{-(y_k-(Ax)_k)^2/(2\sigma^2)} \prod_{j=1}^{n} p_{X_j}(x_j) dx \tag{13}$$

Even though $p_X(x)$ can be decoupled into n factors (due to the assumed independence of the vector components of the prior), the integration can not be carried out separately over single components x_j because each of the m factors $e^{-(y_k-(Ax)_k)^2/(2\sigma^2)}$ requires the full vector x, i.e. the term cannot be easily decomposed into n independent factors. Therefore, to solve (13) a high-dimensional integration is required for each new observation y making this approach infeasible in practice (particularly when practically relevant dimensions of $n \gg 100$ are considered).

2.2.3. Remarks

In many problem settings the prior pdfs $p_X(x_j)$ are unknown, but some sort of prior knowledge (such as a vague notion of sparsity) can be assumed. In those cases, one can try to define a class of pdfs that all fulfil some structural constraint (e.g., a sparsity condition) and then search among this class of pdfs for the worst one, for which then the best possible estimator has to be designed (this is called a minimax–problem, see e.g. [4], [5], for more details).

If the prior pdf $p_{X_j}(x_j)$ is unknown, one can also choose that is convenient to simplify (13) (the following discussion is again based on [4] and [5]). Such a simplifying choice would be $p_{X_j}(x_j) = ce^{-h(x_j)/\sigma^2}$, with c a normalizing factor that ensures that $p_{X_j}(x_j)$ integrates to 1 and $h(x_j)$ a suitable (non-negative) function that can incorporate prior vague knowledge about the signal components (such as sparsity). If we use the right-hand side of (10) we can write (13) as follows

$$\begin{aligned}\hat{x} &= \frac{1}{p_Y(y)}\int_{\mathbb{R}^n} x\cdot\frac{c^n}{\left(\sqrt{2\pi}\sigma\right)^m}\cdot\exp\left(-\frac{1}{2\sigma^2}\left||y-Ax\right||_2^2-\frac{1}{\sigma^2}\sum_{j=1}^{n}h(x_j)\right)dx \\ &= \frac{1}{p_Y(y)}\frac{c^n}{\left(\sqrt{2\pi}\sigma\right)^m}\int_{\mathbb{R}^n} x\cdot\exp\left(-\frac{1}{\sigma^2}\left(\underbrace{\frac{1}{2}\left||y-Ax\right||_2^2+\sum_{j=1}^{n}h(x_j)}_{\doteq L(x)\geq 0}\right)\right)dx\end{aligned} \tag{14}$$

If we now think of small noise, i.e., $\sigma^2 \to 0$, the exponential under term under the integral is dominated by the smallest value that $L(x) \geq 0$ can take, i.e., the posterior pdf under the integral (which includes the constant and $p_Y(y)$ in front of it) will approach a delta-distribution that takes non-zero value for a particular vector $\tilde{x}$, i.e., for $\sigma^2 \to 0$ (14) can be written as

$$\hat{x} =^{\sigma^2\to 0} \int_{\mathbb{R}^n} x\delta(x-\tilde{x})dx = \tilde{x} \qquad \text{with} \qquad \tilde{x} = \arg\min_x L(x) \tag{15}$$

so the mean squared error estimate at low noise approaches that arguments $\tilde{x}$ for which $L()$ takes its smallest value. Therefore, instead of (13) we can write for low noise variance

$$\hat{x} = \arg\min_{\tilde{x}}\left(\frac{1}{2}\left||y-A\tilde{x}\right||_2^2+\sum_{j=1}^{n}h(\tilde{x}_j)\right) \tag{16}$$

The problem (15) has been studied extensively, and for a sparse signal vector the function $h(x_j) = \lambda|x_j|$ has turned out to be a good choice (see e.g., [6], [7]) that at the same time has the charm to lead to the convex problem

$$\hat{x} = \arg\min_{\tilde{x}} \left(\frac{1}{2} \left| |y - A\tilde{x}| \right|_2^2 + \lambda \left| |\tilde{x}| \right|_1 \right), \tag{17}$$

Which can be solved in reasonable time for small-to-moderate dimension n (the solution (17) is referred to as the LASSO[5] in the literature). The fact that the l_0-norm -- which would strictly measure sparsity in (17) but which would cause combinatorial search complexity – can be replaced by the l_1-norm, with the solutions being the same in most cases. This has been a major factor that has driven research in compressed sensing. The factor $\lambda > 0$ in (17) controls the trade-off between the sparsity of an achieved solution (measured by $\left| |\tilde{x}| \right|_1$) and the accuracy (measured by $\left| |y - Ax| \right|_2^2$) of the solution in terms of reproducing the observation vector. This factor has to be chosen somehow, and there is no general simple rule of how to do that. The reason is that the structural constraints used to define (17) are vague (as opposed to the well-defined problem solved by (13) when the pdfs of the priors are known) and therefore the free factor λ is the price one has to pay for a problem that is only defined vaguely. At some point when implementing (17), however one has to be specific about λ. The knowledge about sparsity is hidden in the choice of λ, once we are specific about the value of λ we are implicitly specific about the sparsity of the solution we want.

Some approaches to solve the LASSO will be discussed in the following sections of the thesis, the Iterative Soft Thresholding (IST) is a greedy method to solve the LASSO, as well as Approximate Message Passing (AMP) that is another scheme to solve the LASSO. This algorithm produces significantly better results with faster convergence and can be extended, in particular a Bayesian-optimal version (BAMP) exists for known signal prior.

In this section we would like to point out that (13) is significantly complicated to compute. The approaches to solve the problem include minimax strategies, i.e., the “best solution for the worst priors”, as well as one that considers the low-noise case, for which the problem (17) has been derived. It is unclear what happens, if those schemes are used in a high-noise setting and moreover, one has to cope with adjusting parameter λ.

In the thesis, the discussion of compressed sensing and approaches to solve the underdetermined inverse problem of deducing x from the noisy measurements y in (1) is very much focused on a sparse signal model for x. This is for very good reasons, as many practically interesting problems such as image denoising and inpainting, blind source separation, deconvolution, channel estimation in wireless communications (just to name a few) can be tracked back to sparse signal

[5] Least Absolute Shrinkage and Selection Operator, [6].

models, often not with the signal itself being sparse, but rather in some base (e.g. images after a discrete cosine transform). But compressed sensing as such is no restricted to sparse signals or any other special signal structure, latter can in fact be freely chosen by appropriate priors. In what follows we will, therefore, assume at a certain point that we do know[6] the prior pdfs to solve the well-defined problem (13), which directly leads to Bayesian-optimal Approximate Message Passing (BAMP)- an algorithm without any free factors.

[6] If the prior is unknown, it may be estimated within the iterations of Approximate Message Passing (AMP).

3. Iterative schemes

As explained in the previously section we have a n −dimensional signal $x \in \mathbb{R}^n$ that is also sparse after some linear transformation. Then we assume that x is a s-sparse, i.e., it has at most s non-zero components (sparsity level). Taking $m \ll n$ linear measurements according to the next expression:

$$y_{[m]} = A_{[m \times n]} \cdot x_{[n]} + w_{[m]} \quad \text{with} \quad m \ll n \tag{18}$$

With A the $m \times n$ sensing matrix and w the $m \times 1$ measurement noise vector.

To reconstruct the signal x with the values obtained in y after the compression it is impossible to just invert de value of A as the problem is undetermined because of $m < n$, one could say that the observation vector y is an incomplete description of x.

The performance to guarantee the reconstruction should exploit the sparsity to resolve the uncertainty and the relations of n , m , s so reconstruction of x from y will be successful with high probability.

There are many different algorithms to exploit this situation some of them are explained below.

3.1. Iterative Hard Thresholding (IHT)

The first one is the Iterative Hard Thresholding (IHT), this algorithm has two different incarnations of the CS recovery problem to consider. On one hand, the l_0-regularized problem sets

$$\hat{x} = \arg\min_{\tilde{x}} (\left|\left|y - A\tilde{x}\right|\right|_2^2 + \lambda \left|\left|\tilde{x}\right|\right|_0) \tag{19}$$

On the other hand, the s-sparse problem defines if the level s is known a priori (or a solution with some particular sparsity s is desired) the optimization problem as:

$$\hat{x} = \arg\min_{\tilde{x}} \left(\left|\left|y - A\tilde{x}\right|\right|_2^2\right) \text{ subject to } \left|\left|\tilde{x}\right|\right|_0 \leq s \tag{20}$$

In both cases we assume that the measurement matrix has a norm $\left|\left|A\right|\right|_2 < 1$ by normalization of (18).

3.1.1. l_0-regularized problem

The problem is with solving the l_0-regularized (also any other l_p-regularized, with $p \neq 2$) problem. In this case, the minimization in (19) is for the vector $\hat{x}$ and can be in very large dimensions (more than $n = 1000$). If we write out the cost function assuming column-vectors everywhere we obtain:

$$\begin{aligned} C_{l_0}(\tilde{x}) &= \left||y - A\tilde{x}|\right|_2^2 + \lambda\left||\tilde{x}|\right|_0 = (y - A\tilde{x})^*(y - A\tilde{x}) + \lambda\left||\tilde{x}|\right|_0 \\ &= \left||y|\right|_2^2 - 2\tilde{x}^*A^*y + (A\tilde{x})^*(A\tilde{x}) + \lambda\left||\tilde{x}|\right|_0 \\ &= \left||y|\right|_2^2 - 2\tilde{x}^*A^*y + \tilde{x}^*A^*A\tilde{x} + \lambda\left||\tilde{x}|\right|_0 \end{aligned} \tag{21}$$

Now, we would like to express the solution of (21) as a sum over the components $\tilde{x}_j$ of $\tilde{\mathrm{x}}$, so the minimization can be conducted separately in each component:

$$C_{l_0}(\tilde{x}) = \sum_{j=1}^{n} [-2\tilde{x}_j^*(A^*y)_j + \tilde{x}_j{}^*(A^*A\tilde{x})_j + \left|\tilde{x}_j\right|_0\lambda] + \underbrace{\left||y|\right|_2^2}_{\text{const}} \tag{22}$$

With $(x)_j$ the j-th component of the vector x and $|\tilde{x}_0| = 1$ if $\tilde{x}_j \neq 0$ ($|\tilde{x}_0| = 0$ for $\tilde{x}_j = 0$). Therefore, the term $\tilde{x}_j{}^*(A^*A\tilde{x})_j$ does not allow for separation of the minimization problem.

To solve this problem, we will use the surrogate cost function [8] instead of the desired cost function, with the auxiliary variable z we could know of the same dimension as $\tilde{x}$:

$$C_{l_0}^S(\tilde{x}, z) = \underbrace{\left||y - A\tilde{x}|\right|_2^2 + \lambda\left||\tilde{x}|\right|_0}_{=C_{l_0}(\tilde{x})} + \underbrace{\left||\tilde{x} - z|\right|_2^2 - \left||A\tilde{x} - Az|\right|_2^2}_{\geq 0} \quad (>0) \tag{23}$$

Accordingly, for $\tilde{x} = z$ we have $C_{l_0}^S(\tilde{x}, \tilde{x}) = C_{l_0}(\tilde{x})$ so minimizing $C_{l_0}^S(\tilde{x}, \tilde{x})$ for $\tilde{x}$ will minimize the original cost function ($C_{l_0}(\tilde{x})$) and generally $C_{l_0}^S(\tilde{x}, x)$ will be a majorization of $C_{l_0}(\tilde{x})$ (if matrix norm[7]) $\left||A|\right|_2 < 1$. Hence, minimizing $C_{l_0}^S(\tilde{x}, z)$ means we minimize an upper limit of $C_{l_0}(\tilde{x})$.

Then, if we develop the surrogate cost function we can write:

$$C_{l_o}^S(\tilde{x}, z) = \left||y - A\tilde{x}|\right|_2^2 + \lambda\left||\tilde{x}|\right|_0 + \left||\tilde{x} - z|\right|_2^2 - \left||A\tilde{x} - Az|\right|_2^2 \tag{24}$$

[7] Note the definition of the matrix norm ("largest possible stretching factor")
$\left||A|\right|_2 \doteq \max_{||w||_2=1} \left||Aw|\right|_2 < 1$
So $\left||A|\right|_2 < 1$ implies that $\left||A(\tilde{x} - z)|\right|_2^2 < \left||\tilde{x} - z|\right|_2^2$ and, thus, $C_{l_0}^S(\tilde{x}, z) \geq C_{l_0}(\tilde{x})\ \ \forall z$

$$= \left||y|\right|_2^2 - 2\tilde{x}^* A^* y + \cancel{(A\tilde{x})^*(A\tilde{x})} + \lambda\left|\tilde{x}\right|_0 + \left||\tilde{x}|\right|_2^2 - 2\tilde{x}^* z + \left||z|\right|_2^2$$
$$- [\cancel{(A\tilde{x})^*(A\tilde{x})} - 2\tilde{x}^* A^* A z + \left||Az|\right|_2^2]$$

$$= \left||\tilde{x}|\right|_2^2 - 2\tilde{x}^* z - 2\tilde{x}^* A^* y - 2\tilde{x}^* A^* A z + \lambda\left|\tilde{x}\right|_0 + \left||y|\right|_2^2 + \left||z|\right|_2^2 - \left||Az|\right|_2^2$$

For real variables and matrixes (always assumed here) $x^* = x^T$ and $A^* = A^T$. We denote by A_j the j-th column of the $m \times n$ matrix A ($m < n$). Now we can write the terms in (24) that depend on the vector $\tilde{x}$ as sums over its components $\tilde{x}_j$:

$$C_{l_0}^S(\tilde{x}, z) = \sum_{j=1}^{n} \left[\tilde{x}_j^2 - 2\tilde{x}_j\left(z_j + (A^T y)_j - (A^T A z)_j\right)\right]$$
$$+ \sum_{j=1}^{n} \lambda|\tilde{x}|_0 + \left||y|\right|_2^2 + \underbrace{\left||z|\right|_2^2 - \left||Az|\right|_2^2}_{>0} \tag{25}$$

With the $()_j$ the j-th component of the vector within the brackets and $\left|\tilde{x}_j\right|_0 \equiv \begin{cases} 1 & \text{if } \tilde{x}_j \neq 0 \\ 0 & \text{if } \tilde{x}_j = 0 \end{cases}$

Further simplified:

$$C_{l_0}^S(\tilde{x}, z) = \sum_{j=1}^{n} \underbrace{[\tilde{x}_j^2 - 2\tilde{x}_j\left(z_j + (A^T y)_j - (A^T A z)_j\right) + \lambda|\tilde{x}|_0]}_{D(\tilde{x}_j)}$$
$$\underbrace{+ \left||y|\right|_2^2 + \underbrace{\left||z|\right|_2^2 - \left||Az|\right|_2^2}_{>0}}_{K \equiv f(\tilde{x})} \tag{26}$$

The idea is now to minimize $C_{l_0}^S(\tilde{x}, z)$ by variation of $\tilde{x}$ for given z. As $C_{l_0}^S(\tilde{x}, z) = \sum_{j=1}^{n} D(\tilde{x}_j) + K \geq 0 \;\; \forall \tilde{x}$ we can minimize $C_{l_0}^S(\tilde{x}, z)$ by minimizing the element-wise cost $D(\tilde{x}_j)$ (ignoring the constant K) for each vector component $\tilde{x}_j$ independently. When minimizing $D(\tilde{x}_j)$ over $\tilde{x}_j$ we have to distinguish two types of solutions:

a) $\tilde{x}_j = 0 \rightarrow \left|\tilde{x}_j\right|_0 = 0 \rightarrow D(\tilde{x}_j) = 0$

b) $\tilde{x}_j \neq 0 \rightarrow \left|\tilde{x}_j\right|_0 = 1 \rightarrow D(\tilde{x}_j) = \tilde{x}_j^2 - 2\tilde{x}_j\left(z_j + \left(A^T(y - Az)\right)_j\right) + \lambda$

In this case we obtain the minimum by

$$\frac{dD(\tilde{x}_j)}{d\tilde{x}_j} = 2\tilde{x}_j - 2\left(z_j + \left(A^T(y - Az)\right)_j\right) + \lambda = 0 \tag{27}$$

So we find the minimum at $\tilde{x}_j^{opt} = z_j + \left(A^T(y - Az)\right)_j$ which be plug back into $D(\tilde{x})$:

$$D\left(\tilde{x}_j^{opt}\right)^2 - 2\left(\tilde{x}_j^{opt}\right)^2 + \lambda = \lambda - \left(\tilde{x}_j^{opt}\right)^2 \tag{28}$$

We will pick either the solution a) or b) – whatever minimizes $D(\tilde{x}_j)$:

- ➔ As for a) $\tilde{x}_j = 0$ and the cost is $D(0) = 0$ we will always pick this solution, unless the second solution b) leads to a negative cost
- ➔ $D\left(\tilde{x}_j^{opt}\right) = \lambda - \left(\tilde{x}_j^{opt}\right)^2$ is negative, whenever $\left|\tilde{x}_j^{opt}\right| = \left|z_j + \left(A^T(y - Az)\right)_j\right| > \sqrt{\lambda}$ (with λ the regularization parameter chosen in the original problem(2.1.2)).

In summary, $C_{l_0}^S(\tilde{x}, z)$ in (23) is minimized when the components $\tilde{x}_j, j = 1,2, \dots$ of the solution vector $\tilde{x} \doteq \{\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_n\}$ are chosen as follows:

$$\hat{x}_j = \tilde{x}_j^{opt} = \begin{cases} z_j + \left(A^T(y - Az)\right)_j & \text{if } \left|z_j + \left(A^T(y - Az)\right)_j\right| > \sqrt{\lambda} \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

As the notation in (29) is bulky, we define the Hard Thresholding Operator (with threshold τ):

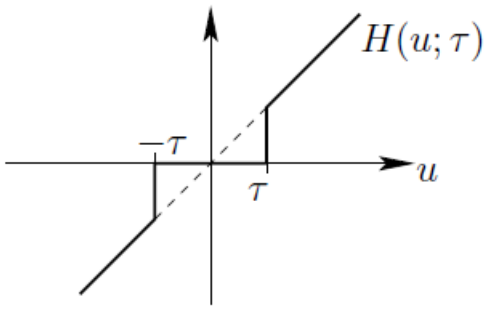


Figure 3. Function of the Hard Thresholding Operator

$$H(u;\tau) = \begin{cases} u & \text{if } |u| > \tau \\ 0 & \text{otherwise} \end{cases} \tag{30}$$

It operates component-wise when applied to a vector $\boldsymbol{u}$ instead of a scalar u.

With (30) the solution (29) reads in vector notation (with $H(.,.)$ applied component-wise)

$$\hat{x} = H(z + \left(A^T(y - Az)\right);\ \sqrt{\lambda}) \tag{31}$$

Finally, this equation is the solution of the minimization of the surrogate cost function (23) for a given auxiliary variable z.

We analyse the possible different solutions for a given z, we have that if $z = \hat{x}$ then $C_{l_0}^S(\hat{x}, z) = C_{l_0}(\hat{x})$, i.e., a solution of the original problem. Otherwise, if $z \neq \hat{x}$, $C_{l_0}^S(\hat{x}, z)$ is still minimized by

(29) and due to $C_{l_0}^S(\hat{x}, z) \geq C_{l_0}(\hat{x})$ we still minimize an upper bound of the original cost function $C_{l_0}(\cdot)$. Hence, it is reasonable to assume that, if $z \neq \hat{x}$, the solution $\hat{x}$ of (31) will be closer than z to the desired solution that minimizes $C_{l_0}(\cdot)$. This motivates to iterate (31) – with the iteration counter i – and use the previous solution of (31) as the value for z in the next step $i+1$ when solving (31) again, i.e.,

$$\hat{x}^{(i+1)} = H\left(\hat{x}^{(i)} + \left(A^T\left(y - A\hat{x}^{(i)}\right)\right); \sqrt{\lambda}\right) \tag{32}$$

Finally, the algorithm of the IHT for the l_0-regularized problem is:

$$\hat{x}^{(i+1)} = H\left(\underbrace{\hat{x}^{(i)} + A^T(y - A\hat{x}^{(i)})}_{=u}; \sqrt{\lambda}\right) \quad i = 0, 1, 2, 3, \ldots \tag{33}$$

with the hard thresholding operator (applied component-wise):

$$H(u;\tau) = \begin{cases} u & \text{if } |u| > \tau \\ 0 & \text{otherwise} \end{cases} \tag{34}$$

and with $\left|\left|A\right|\right|_2 \doteq \max_{||x||_2=1} \left|\left|Ax\right|\right|_2 = \sqrt{\max\{\text{eig}(A^*A)\}} < 1$.

As have already explained before, the parameter λ controls the trade-off between accuracy and sparsity in (19), we also define the start of the algorithm with $\hat{x}^{(0)} = 0$, as is not a critical choice. We also define a suitable stopping criterion that iterates until $\left|\left|\hat{x}^{(i+1)} - \hat{x}^{(i)}\right|\right|_2 < \epsilon \left|\left|\hat{x}^{(i)}\right|\right|_2$, with $\epsilon < 10^{-4}$.

The algorithm can also be applied to signal vectors x that are not strictly sparse, the resulting error will contain a component, due to the best s-sparse approximation. This means the algorithm is robust in the sense that it can cope with measurement noise as well as not exactly sparse signals, this is rather important in practice.

3.1.2. s-sparse problem

As defined in (20) the algorithm is made to guarantee a sparse solution (as opposed to the l_0-regularized problem). The concept in this case is similar as above (23), optimizing a surrogate cost function [8] with the auxiliary variable z of the same dimension as $\tilde{x}$:

$$C_s^S(\tilde{x},z)=\underbrace{\left||y-A\tilde{x}|\right|_2^2}_{=C_s(\tilde{x})}+\underbrace{\left||\tilde{x}-z|\right|_2^2-\left||A(\tilde{x}-z)|\right|_2^2}_{\geq 0 \text{ as } ||A||_2<1} \quad (>0) \tag{35}$$[8]

As above, (35) can be decoupled into sum of terms over the signal components. We obtain the same as (26), only without a constraint scaled by λ.

$$C_s^S(\tilde{x},z)=\sum_{j=1}^{n}\underbrace{[\tilde{x}_j^2-2\tilde{x}_j\left(z_j+\left(A^T(y-Az)\right)_j\right)]}_{D(\tilde{x}_j)}+\underbrace{\left||y|\right|_2^2+\underbrace{\left||z|\right|_2^2-\left||Az|\right|_2^2}_{>0}}_{K\equiv f(\tilde{x})} \tag{36}$$

Therefore, if we minimize the cost-contributions $D(\tilde{x}_j)$ of the components separately we obtain:

$$\frac{dD(\tilde{x}_j)}{d\tilde{x}_j}=2\tilde{x}_j-2\left(z_j+\left(A^T(y-Az)\right)_j\right)=0 \tag{37}$$

so we find the minimum at $x_j^{opt}=z_j+\left(A^T(y-Az)\right)_j$ which once if plugged back into $D(\tilde{x}_j)$:

$$D\left(\tilde{x}_j^{opt}\right)=\left(\tilde{x}_j^{opt}\right)^2-2\left(\tilde{x}_j^{opt}\right)^2=-\left(\tilde{x}_j^{opt}\right)^2 \tag{38}$$

Consequently, the total cost is:

$$C_s^S(\tilde{x},z)=\sum_{j=1}^{n}\underbrace{[\left(z_j+\left(A^T(y-Az)\right)_j\right)]}_{\tilde{x}_j^{opt}}+\underbrace{\left||y|\right|_2^2+\underbrace{\left||z|\right|_2^2-\left||Az|\right|_2^2}_{>0}}_{K\equiv f(\tilde{x})} \tag{39}$$

The total cost takes its smallest value for a given number s of non-zero components, when the s largest (in magnitude) components $\tilde{x}_j^{opt}$ are picked.

A simplified notation as (31) defines

$$\hat{x}=H(z+\left(A^T(y-Az)\right);\tau_s) \tag{40}$$

with the hard thresholding operator (known already from above; again applied component wise)

$$H(u;\tau)=\begin{cases}u & \text{if } |u|>\tau_s\\ 0 & \text{otherwise}\end{cases} \tag{41}$$

[8] Note: no regularization here, but the constraint $\left||\tilde{x}|\right|_0\leq s$ is inserted later

and the threshold τ_s selected as the s-largest magnitude of the vector $u = z + \left(A^T(y - Az)\right)$. A common alternative is to defined the s-sparse vector-base hard thresholding operator $H_s(u)$ that keeps the components with the s largest magnitudes and sets the rest to zero.

Therefore, the s-sparse iterative hard thresholding algorithm reads

$$\hat{x}^{(i+1)} = H_s\left(\hat{x}^{(i)} + A^T(y - A\hat{x}^{(i)})\right) \quad i = 0, 1, 2, 3, \ldots \tag{42}$$

with $\left|\left|A\right|\right|_2 \doteq \max_{||x||_2=1} \left|\left|Ax\right|\right|_2 = \sqrt{\max\{\text{eig}(A^*A)\}} < 1$.

As in the l_0-regularized problem, we define the start of the algorithm with $\hat{x}^{(0)} = 0$, as is not a critical choice. We also define a suitable stopping criterion that iterates until $\left|\left|\hat{x}^{(i+1)} - \hat{x}^{(i)}\right|\right|_2 < \epsilon \left|\left|\hat{x}^{(i)}\right|\right|_2$, with $\epsilon < 10^{-4}$. Its important to denote that this algorithm can handle huge vector dimension (as opposed to many other schemes).

The algorithm can also be applied to signal vectors x that are not strictly sparse, the resulting error will contain a component, due to the best s-sparse approximation. This means the algorithm is robust in the sense that it can cope with measurement noise as well as not exactly sparse signals, this is (as in l_0-regularized problem).

The performance guarantee is a rather strong statement, but it has a disadvantage because the Restricted Isometry Property (RIP) is required for A which is hard to check for a given measurement matrix. Another, performance guarantee is also of the "uniform type", ensuring error bounds and convergences guarantees for any possible source vector: this is again a rather strong statement. In practice we may often be more interested in average performance guarantee rather than in one for the worst case (as the one above). In addition, we cannot draw strong conclusions from a worst-case guarantee for the average-case, so one way to obtain "average-case guarantees" is to run simulations.

3.2. Iterative Soft Thresholding (IST)

The problem setting is the same one as explained in (18) the first part of this section. As we have seen in the previous algorithm, IHT has a combinatorial complexity due to l_0-pseudo-norm. For the Iterative Soft Thresholding (IST) algorithm we change the norm to l_1 instead of l_0 as a common relaxation of the problem:

$$\hat{x} = \arg\min_{\tilde{x}} \left(\frac{1}{2} \left|\left| y - A\tilde{x} \right|\right|_2^2 + \lambda \left|\left| \tilde{x} \right|\right|_1 \right) \tag{43}$$

The difficulties in this case come from the convex optimization problem (still rather complex for high dimension n), the Least Absolute Shrinkage and Selection Operator (LASSO) problem, also called Basis Pursuit Denoising (BPDN). The l_1-norm also promotes sparsity that often leads to the same result as for l_0-norm.

For convenience, instead of the desired cost function in (44) we normalize differently:

$$\hat{x} = \arg\min_{\tilde{x}} \left(\left|\left| y - A\tilde{x} \right|\right|_2^2 + \tilde{\lambda} \left|\left| \tilde{x} \right|\right|_1 \right) \qquad (\text{so } \tilde{\lambda} = 2\lambda) \tag{44}$$

Similar as for IHT (see (23)) we consider a surrogate cost function with the auxiliary variable z of the same dimension as $\tilde{x}$:

$$C_{l_1}^S(\tilde{x}, z) = \underbrace{\left|\left| y - A\tilde{x} \right|\right|_2^2 + \tilde{\lambda} \left|\left| \tilde{x} \right|\right|_0}_{= C_{l_1}(\tilde{x})} + \underbrace{\left|\left| \tilde{x} - z \right|\right|_2^2 - \left|\left| A(\tilde{x} - z) \right|\right|_2^2}_{\geq 0 \; as \; \left|\left| A \right|\right|_2 < 1} \qquad (> 0) \tag{45}$$

Again as for IHT we request $\left|\left| A \right|\right|_2 \doteq \max_{||w||_2 = 1} \left|\left| Aw \right|\right|_2$, then if $\left|\left| A \right|\right|_2 < 1$ implies that $\left|\left| A(\tilde{x} - z) \right|\right|_2^2 < \left|\left| \tilde{x} - z \right|\right|_2^2$ and, hence, $C_{l_1}^S(\tilde{x}, z) \geq C_{l_1}(\tilde{x}) \;\; \forall z$. Accordingly, for $\tilde{x} = z$ we have $C_{l_1}^S(\tilde{x}, \tilde{x}) = C_{l_1}(\tilde{x})$ so minimizing $C_{l_1}^S(\tilde{x}, \tilde{x})$ for $\tilde{x}$ will minimize the original cost function ($C_{l_1}(\tilde{x})$) and generally $C_{l_1}^S(\tilde{x}, x)$ will be a majorization of $C_{l_1}(\tilde{x})$). Hence, minimizing $C_{l_1}^S(\tilde{x}, z)$ means we minimize an upper limit of $C_{l_1}(\tilde{x})$. With exactly the same steps as for derivation of (27), we re-write (46) as a sum over independent contributions to the total cost:

$$\begin{aligned} & C_{l_1}^S(\tilde{x}, z) \\ &= \sum_{j=1}^{n} [\underbrace{\left(\tilde{x}_j^2 - 2\tilde{x}_j \left(z_j + \left(A^T(y - Az) \right)_j \right) \right)}_{D(\tilde{x}_j)} + \tilde{\lambda} |\tilde{x}_j|] \\ &+ \underbrace{\left|\left| y \right|\right|_2^2 + \underbrace{\left|\left| z \right|\right|_2^2 - \left|\left| Az \right|\right|_2^2}_{>0}}_{K \equiv f(\tilde{x})} \end{aligned} \tag{46}$$

As $C_{l_1}^S(\tilde{x}, z) = \sum_{j=1}^{n} D(\tilde{x}_j) + K \geq 0 \; \forall \tilde{x}$ we can minimize $C_{l_1}^S(\tilde{x}, z)$ by minimizing independently the element-wise cost $D(\tilde{x}_j)$ for each component $\tilde{x}_j$ ignoring the constant K. At that point, when minimizing

$$D(\tilde{x}_j) = \tilde{x}_j^2 - 2\tilde{x}_j \left(\underbrace{z_j + \left(A^T(y - Az)\right)_j}_{\doteq r_j} \right) + \tilde{\lambda}|\tilde{x}_j| \tag{47}$$

over $\tilde{x}_j$ we have to distinguish solutions with $|\tilde{x}_j| \geq 0$ and $|\tilde{x}_j| < 0$. Note that r_j can take any real value.

a) $\tilde{x}_j \geq 0 \implies |\tilde{x}_j| = \tilde{x}_j$ hence (noting $\tilde{\lambda} = 2\lambda$)

$$D(\tilde{x}_j) = \tilde{x}_j^2 - 2\tilde{x}_j(r_j - \lambda) = \tilde{x}_j\left(\tilde{x}_j - 2(r_j - \lambda)\right) \tag{48}$$

In this case we obtain a minimum by

$$\frac{dD(\tilde{x}_j)}{d\tilde{x}_j} = 2\tilde{x}_j - 2(r_j - \lambda) = 0 \;\Rightarrow\; \tilde{x}_j^{opt} = (r_j - \lambda) \quad \forall r_j \geq \lambda \tag{49}$$

we plug (50) back into $D(\tilde{x}_j)$ in (49):

$$D\left(\tilde{x}_j^{opt}\right) = \tilde{x}_j^{opt}\left(\tilde{x}_j^{opt} - 2\tilde{x}_j^{opt}\right) - \left(\tilde{x}_j^{opt}\right)^2 \tag{50}$$

so as long as $r_j > \lambda$ we obtain negative "cost" (ignoring constant K) and the optimal solution is given by

$$\tilde{x}_j^{opt} = r_j - \lambda \quad \text{if} \quad r_j \geq \lambda \tag{51}$$

for $r_j < \lambda$ the smallest possible value $D(\tilde{x}_j) = 0$ is taken at $\tilde{x}_j \geq 0$. In summary we have for $\hat{x}_j \geq 0$:

$$\hat{x}_j = \tilde{x}_j^{opt} = \begin{cases} r_j - \lambda & \text{if } r_j \geq \lambda \quad \text{then } D(\hat{x}_j) \leq 0 \\ 0 & otherwise \quad \text{then } D(\hat{x}_j) = 0 \end{cases} \tag{52}$$

b) $\tilde{x}_j < 0 \;\Rightarrow\; |\tilde{x}_j| = -\tilde{x}_j$, hence (noting $\tilde{\lambda} = 2\lambda$)

$$D(\tilde{x}_j) = \tilde{x}_j^2 - 2\tilde{x}_j(r_j + \lambda) = \tilde{x}_j\left(\tilde{x}_j - 2(r_j + \lambda)\right) \tag{53}$$

We obtain a minimum by

$$\frac{dD(\tilde{x}_j)}{d\tilde{x}_j} = 2\tilde{x}_j - 2(r_j + \lambda) = 0 \;\Rightarrow\; \tilde{x}_j^{opt} = (r_j + \lambda) \quad \forall r_j < -\lambda \tag{54}$$

we plug (55) back into $D(\tilde{x}_j)$ in (54):

$$D\left(\tilde{x}_j^{opt}\right) = \tilde{x}_j^{opt}\left(\tilde{x}_j^{opt} - 2\tilde{x}_j^{opt}\right) - \left(\tilde{x}_j^{opt}\right)^2 \tag{55}$$

so as long as $r_j < -\lambda$ we obtain negative "cost" (ignoring constant K) and the optimal solution is given by

$$\tilde{x}_j^{opt} = r_j + \lambda \quad \text{if} \quad r_j < -\lambda \tag{56}$$

for $r_j \geq -\lambda$ the optimal solution must be located at the corner of the definition region $\tilde{x}_j < 0$, as (55) has only one solution. We take the again cost functions (without constants):

$$D(\tilde{x}_j) = \tilde{x}_j\left(\tilde{x}_j - 2(r_j + \lambda)\right) \geq 0 \quad \text{for } r_j \geq -\lambda \quad \text{and } x_j < 0 \tag{57}$$

hence, for $r_j \geq -\lambda$ the smallest possible value $D(\tilde{x}_j) = 0$ is taken at $\tilde{x}_j = 0$. Finally, for $\hat{x}_j < 0$:

$$\hat{x}_j = \tilde{x}_j^{opt} = \begin{cases} r_j + \lambda & \text{if } r_j < -\lambda \quad \text{then } D(\hat{x}_j) < 0 \\ 0 & otherwise \quad \text{then } D(\hat{x}_j) = 0 \end{cases} \tag{58}$$

Given the two solutions a) and b) above, which both apply for any real number r_j, we still have to decide which one to pick for a given value r_j. As the regularization parameter $\lambda \geq 0$ there is only a chance to obtain a negative cost, when solution a) is chosen for $r_j > 0$ (negative cost if $r_j > \lambda$, zero cost in all other cases). The same principle applies to values $r_j < 0$. Therefore, the overall solution reads:

$$\hat{x}_j = \begin{cases} r_j - \lambda & \text{if} \quad r_j > \lambda & \text{then } D(\hat{x}_j) < 0 \\ 0 & \text{if} \quad -\lambda \leq r_j \leq \lambda & \text{then } D(\hat{x}_j) = 0 \\ r_j + \lambda & \text{if} \quad r_j < -\lambda & \text{then } D(\hat{x}_j) < 0 \end{cases} \tag{59}$$

With these conditions the solution using the Soft Thresholding operator $\eta(r_j;\lambda)$ is:

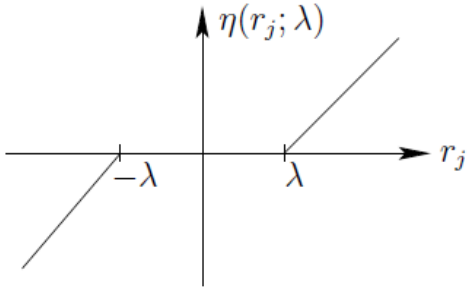


Figure 4. Function of the Soft Thresholding Operator

$$\hat{x}_j = \eta(r_j;\lambda) = \begin{cases} r_j - \lambda & \text{if} \quad r_j > \lambda \\ 0 & \text{if} \quad -\lambda \le r_j \le \lambda \\ r_j + \lambda & \text{if} \quad r_j < -\lambda \end{cases} \tag{60}$$

$$= sign(r_j)\max(|r_j| - \lambda, 0) \tag{61}$$

The Soft Thresholding operator is applied component-wise in vector notation. Certainly, the surrogate cost function $C_{l_1}^S(\tilde{x}, z)$ in (46) is minimized for given measurements y and a given auxiliary variable z by

$$\hat{x} = \eta\left(z + \left(A^T(y - Az)\right);\lambda\right) \tag{62}$$

Similar as for IHT, we motivate iterations by considering that $C_{l_1}^S(\hat{x}, z) \le C_{l_1}^S(\tilde{x}, z)\ \ \forall\tilde{x}$. So, if we estimate $\hat{x}^{(i)}$, we can use it as an auxiliary variable $z = \hat{x}^{(i)}$ and improve this estimate by minimizing $C_{l_1}^S(\tilde{x}, x^{(i)})$. This is achieved by application of (63), i.e.,

$$\hat{x}^{(i+1)} = \eta\left(x^{(i)} + \left(A^T\left(y - Ax^{(i)}\right)\right);\lambda\right) \tag{63}$$

This is the scheme we call Iterative Soft Threholding, it was introduced for Compressed Sensing in [8]. As well as the IHT we start iterations with $\hat{x}^{(0)} = 0$ as is not a critical choice. The threshold parameter λ, according to the derivation just adopted from the cost function might not be the best choice to really minimize $C_{l_1}(\tilde{x})$, as the derivation minimizes a surrogate cost function and not $C_{l_1}(\tilde{x})$ directly.

We won't consider other versions of IST in this thesis but is good to know that there is an accelerated version introduced in [10].

3.3. Heuristics for Iterative Recovery

First of all, we start from noiseless measurement equation $y_{[m]} = A_{[m\times n]} \cdot x_{[n]}$ with $m = n$. If A is orthogonal, i.e., $A^* = A^{-1}$ (and $A^* = A^T$ for A real: always assumed), then the solution could be fined by simple inversion, i.e., $A^* y = x$. However, if we assume $m < n$, A can't be orthogonal and is not invertible but we could still write:

$$A^*y = A^*Ax = x + \underbrace{(A^*A - I)}_{\doteq H} x \tag{64}$$

The term Hx can be interpreted as "noise", due to random entries of A, so $A^*y = x + noise \doteq \tilde{y}$. De-noising from the literature (e.g. [3],[11]): appropriately tuned (λ) soft thresholding of (x + noise) will reduce noise and lead to an estimate $\hat{x}^{(1)}$ of x with smaller Mean Squared Error (MSE). Then, de-noising of sparse signal by soft thresholding $\eta()$, applied component-wise to $\tilde{y} \doteq x + noise$:

$$\hat{x}^{(1)} = \eta(\tilde{y}; \lambda) = \begin{Bmatrix} \tilde{y} - \lambda & \text{if} & \tilde{y} > \lambda \\ 0 & \text{if} & -\lambda \le \tilde{y} \le \lambda \\ \tilde{y} + \lambda & \text{if} & \tilde{y} < -\lambda \end{Bmatrix} = sign(\tilde{y}) \max|\tilde{y}| - \lambda, 0) \tag{65}$$

If $\hat{x}^{(1)}$ is a better estimate of x than $\tilde{y} = A^*y$, we have $\left|\left|x - \hat{x}^{(1)}\right|\right|_2^2 < \left|\left|x - A^*y\right|\right|_2^2$. Then if we use iterative soft thresholding with the initialisation $\hat{x}^{(0)} = 0$:

$$\begin{aligned} \hat{x}^{(i+1)} &= \eta(A^*\left(y - A\hat{x}^{(i)} + \hat{x}^{(i)}; \lambda\right) \qquad i = 0, 1, 2, \dots \\ &= \eta(A^*A\left(x - \hat{x}^{(i)}\right) + \hat{x}^{(i)};\ \lambda\right) \\ &= \eta(A^*A\left(x - \hat{x}^{(i)}\right) + \hat{x}^{(i)} + x - x;\ \lambda\right) \\ &= \eta(A^*A\left(x - \hat{x}^{(i)}\right) - (x - \hat{x}^{(i)}) + x;\ \lambda\right) \\ &= \eta(H\left(x - \hat{x}^{(i)}\right) + x;\ \lambda\right) \qquad \text{with} \quad \mathrm{H} \doteq \mathrm{A^*A} - \mathrm{I} \end{aligned} \tag{66}$$

With the appropriate threshold λ the "noise" $H(x - \hat{x}^{(i)})$ gets smaller (in MSE) with each iteration. Ultimately, $\hat{x}^{(i)} \overset{i\to\infty}{\longrightarrow} x$, so $\hat{x}^{(i+1)} = \eta(x; \lambda)$ with $\lambda \to 0$ then with decreasing noise, the threshold λ must approach zero, as only then we obtain a fixed point $x = \eta(x; 0)$ at the desired solution. Therefore, the exact recovery for noiseless measurements are only possible with adaptive λ. The choice of λ with noisy measurements can be seen as a problem where extra measurement noise adds to the "noise" due to undersampling ($m < n$). Additionally, IHT (for the l_0-regularized problem) has the same problem with adjusting the threshold.

Finally, now that we know how all these algorithms work we are ready to recover our signal, but there are still some difficulties. First of all, one could say that the results are not the best possible, moreover, is not clear how we choose the parameters of λ and τ and the convergence of the signal is slow.

In an extensive numerical study in [6] several iterative recovery algorithms were optimally tuned for given undersampling factor $\delta = m/n$, this parameter is known from the measurement matrix.

4. Approximate Message Passing (AMP) and Bayesian derivation (BAMP)

Approximate Message Passing (AMP) is a rather new topic and as yet the field has not settled. In this section we will jump from the IST to new AMP algorithm thanks to the graphical model for the LASSO and the min-sum algorithm.

As it has seen in the previous sections of the thesis we set the general noisy problem

$$y_{[m]} = A_{[m\times n]} \cdot x_{[n]} + w_{[m]} \quad \text{with} \quad m \ll n$$
$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w} \tag{67}$$

Where y is the observation vector, w the measurement vector noise with variance σ_w^2 and x is the sparse signal to be recovered. The measurement matric $A = \{A_{:1}, A_{:2}, \dots, A_{:n}\}$, assume normalized column vectors $\left|\left|A_{:j}\right|\right|_2 = 1$.

4.1. Graphical model for the LASSO

At this point, we want to find a some sparse solution and this lead us to the Least Absolute Shrinkage and Selection Operator (LASSO). This is the method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. In our case, we represent the LASSO problem in the so-called Lagrangian form

$$\hat{x} = \arg\min_{\tilde{x}} \left(\underbrace{\frac{1}{2}\left|\left|y - A\tilde{x}\right|\right|_2^2 + \lambda\left|\left|\tilde{x}\right|\right|_1}_{C_{l_1}(\tilde{x})} \right) \tag{68}$$

Therefore, our cost function $C_{l_1}(\tilde{x})$ expressed as the q^{th} component

$$C_{l_1}(\tilde{x}) = \frac{1}{2}\left|\left|y - A\tilde{x}\right|\right|_2^2 + \lambda\left|\left|\tilde{x}\right|\right|_1 = \frac{1}{2}\sum_{q=1}^{m}\left(y_q - (A\tilde{x})_q\right)^2 + \lambda\sum_{j=1}^{n}|\tilde{x}_j| \tag{69}$$

Where $(A\tilde{x})_q$ denotes the q^{th} component of the "measurement" resulting from a particular choice for $\tilde{x}$. Then, if we develop the cost function in (70)

$$C_{l_1}(\tilde{x}) = \frac{1}{2}\sum_{q=1}^{m}\left(y_q - A_{q:}\tilde{x}\right)^2 + \lambda\sum_{j=1}^{n}|\tilde{x}_j| \\ = \sum_{q\in F}\underbrace{\frac{1}{2}\left(y_q - A_{q:}\tilde{x}\right)^2}_{C_q(\tilde{x})} + \sum_{j\in V}\underbrace{\lambda|\tilde{x}_j|}_{C_j(\tilde{x})} \tag{70}$$

with $F = \{1, 2, \dots, m\}$ the set of all measurement indices and $V = \{1, 2, \dots, n\}$ the set of all indices of the components of the signal vector. This notation is introduced to better match the cost function to standard graph notation.

Now the goal is to minimize the cost function by suitable choice of the optimization variables, i.e., the components $\tilde{x}_j, j = 1, \dots, n$ of the signal vector $\tilde{x}$. As well known, a direct minimization of the first part in (71) will be prohibitively complex in very high dimensions n (even though is a convex problem). The principle of a graph-based approach is that one tries to minimize the total cost by local minimization of the components $C_q(\tilde{x})$ and $C_j(\tilde{x})$ of the cost function. Note that we had achieved this before by manipulation of the cost function such that the minimization problem could be algebraically decomposed into independent sums of costs per optimization variable $\tilde{x}_j$. It is important to know too that depending on the matrix A, the sum-terms $C_1(\tilde{x})$ may not all contain all signal components $\tilde{x}_j$.

Once the cost functions are settled, we define the general graph for measurement matrix $A = \{a_{q,j}\}, q = 1, \dots, m; j = 1, \dots, n$. We assume that the graph is fully connected, i.e., $a_{q,j} \neq 0\ \forall q$ and $\forall j$, as it is a common case in compressed sensing.

Then if we represent the measurement matrix in a graph model we obtain:

$$A = \begin{Bmatrix} a_{11} & a_{12} & a_{1j} & a_{14} & \dots & a_{1n} \\ a_{q1} & a_{q2} & a_{qj} & a_{q4} & \dots & a_{qn} \\ a_{31} & a_{32} & a_{3j} & a_{34} & \dots & a_{3n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & a_{mj} & a_{m4} & \dots & a_{mn} \end{Bmatrix}$$

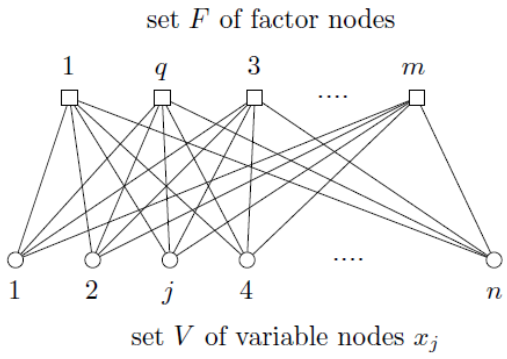


Figure 5. Graph representation for the LASSO problem

Once, the graph model is settled we use the min-sum algorithm to minimize our cost function.

4.2. Min-sum algorithm

The min-sum algorithm is an established scheme to minimize a cost function on a graph, for background details see **¡Error! No se encuentra el origen de la referencia.**. In this case, we apply the algorithm to the last part of (71) and we obtain for the message-update rules for all $j = 1, \dots, n$ and $q = 1, \dots, m$:

variable-node j to factor-node q:
$$v_{j\to q}^{(i+1)}(\tilde{x}_j) = \lambda|\tilde{x}_j| + \sum_{\tilde{q}\in\partial j\setminus q} f_{q\to j}^{(i)}(\tilde{x}_j) \tag{71}$$

factor-node q to variable-node j:
$$f_{q\to j}^{(i+1)}(\tilde{x}_j) = \min_{\tilde{x}_{\partial q\setminus j}} \left\{ \frac{1}{2}\left(y_q - A_{q:}\tilde{x}\right)^2 + \sum_{\tilde{\jmath}\in\partial q\setminus j} v_{j\to q}^{(i)}(\tilde{x}_j) \right\} \tag{72}$$

The final step is to estimate $\hat{x}_j$ after convergence of the messages $v_{j\to q}^{(i+1)}(\tilde{x}_j)$ and $f_{q\to j}^{(i+1)}(\tilde{x}_j)$:

$$\hat{x}_j = \arg\min_{\tilde{x}\in\mathbb{R}} \left\{ \lambda|\tilde{x}_j| + \sum_{q\in\partial j} f_{q\to j}^{(i+1)}(\tilde{x}_j) \right\} \tag{73}$$

As we can see the message $f_{q\to j}^{(i+1)}(\tilde{x}_j)$ depends on $\tilde{x}_j$ only through the term $a_{qj}\tilde{x}_j$, then, we can assume that this function is approximated by a Taylor series that is developed around the (for the moment unknown) x_j-value that minimizes the local cost function. This minimum later appears as a number that describes the function $f_{q\to j}^{(i+1)}(\tilde{x}_j)$. The minimum-operation in $f_{q\to j}^{(i+1)}(\tilde{x}_j)$ (for the LASSO cost function) enforces a solution that can be expressed by soft thresholding $\eta(\cdot;\cdot)$ [Section 3.2]. During this procedure of the somewhat complicated derivation, various terms can be safely dropped under the assumption that the matrix columns are normalized (unit-norms columns of A) and that the dimension considered are large [10].

As an intermediate result we obtain a new (and approximate) message-passing rules. The message-update rules for iteration i for all $j = 1, \dots, n$ and $q = 1, \dots, m$ are:

$$\tilde{x}_{j\to q}^{(i+1)} = \eta\left(\sum_{\tilde{q}=1,\tilde{q}\neq q}^{m} A_{\tilde{q}j} z_{q\to j}^{(i)}; \theta_i \right) \tag{74}$$

$$z_{q\to j}^{(i)} = y_q - \left(\sum_{\tilde{\jmath}=1,\tilde{\jmath}\neq j}^{n} A_{\tilde{q}\tilde{\jmath}} \tilde{x}_{j\to q}^{(i)} \right) \tag{75}$$

The final step consists in estimate $\hat{x}_j$ after convergence of the messages $\tilde{x}_{j\to q}^{(i+1)}$ and $r_{q\to j}^{(i)}$:

$$\hat{x}_j = \eta\left(\sum_{q=1}^{m} A_{qj} z_{q\to j}^{(i)};\ \theta_i\right) \tag{76}$$

This message passing algorithm is still very complicated, as for each signal component j the iterations must be run separately with the sums in (75) and (76) having to be computed also for each different value of q and j.

In the final steps, the message passing scheme above is further simplified, again by dropping terms in expanded version that can be neglected due to matrix-columns normalization and large dimension assumed. The final result is of the type:

$$\hat{x}^{(i)} = \eta(\hat{x}^{(i-1)} + A^T z^{(i-1)};\ \theta_i) \tag{77}$$

$$z^{(i)} = y - A\hat{x}^{(i)} + b^{(i-1)} z^{(i-1)} \tag{78}$$

with signal-dependent constants θ_i and $b^{(i-1)}$ and i the iterations counter and soft-thresholding operator seen in the [Section 3.2]

$$\hat{x}_j = \eta(r_j;\lambda) = \begin{Bmatrix} r_j - \lambda & \text{if} & r_j > \lambda \\ 0 & \text{if} & -\lambda \le r_j \le \lambda \\ r_j + \lambda & \text{if} & r_j < -\lambda \end{Bmatrix} = sign(r_j)\max(|r_j| - \lambda, 0) \tag{79}$$

that is applied component-wise to the vector $r_j = \hat{x}^{(i-1)} + A^T z^{(i-1)}$.

Note that this algorithm is very similar to IST, the difference is the term $b^{(i-1)} z^{(i-1)}$ in (80).

4.3. Approximate Message Passing (AMP)

The Approximate Message Passing (AMP) scheme has constants/thresholds that must be determined. We can find two schemes depending on whether the measurement noise variance is known or unknown. As we have solved the LASSO problem, there remains a free parameter λ that is a tuning factor to control achieved sparsity of the solution and it depends on the measurement noise variance.

4.3.1. AMP I

In the first algorithm of AMP the measurement noise variance σ^2 is known and based on [10] the solution of the algorithm for iterations $i = 1, 2, \ldots$ is:

$$\hat{x}^{(i)} = \eta\left(\hat{x}^{(i-1)} + A^T z^{(i-1)}; \sqrt{\beta c^{(i-1)}}\right)$$

$$b^{(i-1)} = \frac{1}{m}\left|\left|\hat{x}^{(i)}\right|\right|_0 \tag{80}$$

$$z^{(i)} = y - A\hat{x}^{(i)} + b^{(i-1)} z^{(i-1)}$$

$$c^{(i)} = \sigma^2 + c^{(i-1)} b^{(i-1)}$$

And the initial parameters are set when $i = 0$ as described above:

$$\begin{aligned} &\hat{x}^{(0)} = 0_{n\times 1} && \text{(signal vector; dimension n} > \text{m)} \\ &z^{(0)} = y && \text{(dimensions: m} \times 1) \\ &c^{(0)} = \sigma^2 + \frac{1}{m}\left|\left|z^{(0)}\right|\right|_2^2 && \text{(scalar)} \end{aligned} \tag{81}$$

As we can see the proposed solution for the adaptation of the "effective-noise variance", $c^{(i)}$, and σ^2 the actual measurement noise variance is interesting. Applying the unilateral Z-transform (time index $k = i - 1$) to $c^{(i)}$ we obtain:

$$C(z) = \frac{\sigma^2}{1-b}\left(\frac{z}{z-1} - b\frac{z}{z-b}\right) + bc^{(-1)}\frac{z}{z-b} \tag{82}$$

and the solution in time-domain is

$$\begin{aligned} c^{(k)} &= \frac{\sigma^2}{1-b}\left(1 - b^{k+1}\right)\sigma[k] + c^{(-1)} b^{k+1}\sigma[k] \\ &= \frac{\sigma^2}{1-b}\,\sigma[k] + \left(c^{(-1)} - \frac{\sigma^2}{1-b}\right) b^{k+1}\sigma[k] \end{aligned} \tag{83}$$

First, if we assume that the scheme recovers the correct s-sparse solution then $b^{(i-1)} = \frac{1}{m}\left|\left|\hat{x}^{(i)}\right|\right|_0 = \frac{s}{m}$. Therefore, we treat $b^{(i-1)} = b = \frac{s}{m} < 1$ as a constant. Note that $\lim_{k\to\infty} b^{k+1}\sigma[k] = 0$, as $|b| < 1$, so

$$\lim_{k\to\infty} c^{(k)} = \frac{\sigma^2}{1-b} > \sigma^2 \tag{84}$$

As $c^{(k)}$ is the noise variance in the current iteration that is used in the thresholding function, this means that the effective noise variance $c^{(k)}$ in the n signal components is larger than the measurement noise variance σ^2 in the m measurements, so it causes extra noise because of the undersampling.

Second, if we now consider that $b^{(i-1)}$ is not a constant, but for whatever reason $\left|\left|\hat{x}^{(i)}\right|\right|_0$ is close to m we obtain $b^{(i-1)} \to 1$. This means $c^{(k)}$ become very large, as $(1-b)$ approaches zero. From a maths-perspective this may still be good, as large c means that in the next iteration most components will be set to zero by the soft thresholder $\eta()$ and then $c^{(i)} = \sigma^2$.

Third, and last option, is when $b > 1$, then the implementation is set to avoid trouble by limiting $b^{(i-1)} = \max\left\{b^{(i-1)}, \frac{m-1}{m}\right\}$. A bigger result of $b^{(i-1)}$ could result to divergences of the AMP I algorithm. These divergences are solved in the second algorithm of approximate message passing.

4.3.2. AMP II

In this algorithm the noise variance σ^2 is unknown, therefore we have:

$$\begin{aligned} \hat{x}^{(i)} &= \eta\left(\hat{x}^{(i-1)} + A^T z^{(i-1)}; \sqrt{\beta c^{(i-1)}}\right) \\ z^{(i)} &= y - A\hat{x}^{(i)} + z^{(i-1)} \frac{1}{m}\left|\left|\hat{x}^{(i)}\right|\right|_0 \\ c^{(i)} &= \frac{1}{m}\left|\left|z^{(i)}\right|\right|_2^2 \end{aligned} \tag{85}$$

and the initializations at $i = 0$:

$$\begin{aligned} \hat{x}^{(0)} &= 0_{n\times 1} && \text{(signal vector; dimension n} > \text{m)} \\ z^{(0)} &= y && \text{(dimensions: m} \times 1) \\ c^{(0)} &= \frac{1}{m}\left|\left|z^{(0)}\right|\right|_2^2 && \text{(scalar)} \end{aligned} \tag{86}$$

In this case, thanks to the new definition of c (as we do not know the noise variance σ^2) we are able to solve the problem of divergence we had in the first approximate message passing algorithm.

In both cases AMP algorithms I and II can use the stopping criterion known from other iterative schemes: iterate, until $\left|\left|\hat{x}^{(i+1)} - \hat{x}^{(i)}\right|\right|_2 < \epsilon \left|\left|\hat{x}^{(i)}\right|\right|_2$, with $\epsilon < 10^{-4}$.

4.4. Bayesian Approximate Message Passing (BAMP)

In addition to this, if we come back to the estimation problem of other sections. We picked a prior of the form $p_{X_j}(x_j) = c\, e^{-h(x_j)/\sigma^2}$ using $h(x_j) = \lambda|x_j|$ for sparse signals and considering small noise $\sigma^2 \to 0$, this led us to the well-known LASSO problem. All these assumptions were justified to have a primary motivation for those approximations, also because of the fact that the prior pdf is indeed often unknown, but is known that the solution shall be sparse. With that a graph-based approach was used to solve the LASSO, then the min-sum algorithm with various approximations and in the large-system limit ($n \to \infty$) led the solution to the Approximate Message Passing. But what will happen if we know the pdf, does it change something? The BAMP algorithm responds this problem as we can see above.

The problem setting is the same as AMP, in particular the columns of the measurement matrix A are assumed to be normalized to one. Derivation in [2] and [4]. The basic concept is the same as for AMP, it starts from

$$\hat{\boldsymbol{x}} = \frac{1}{p_{\boldsymbol{Y}}(y)} \int_{\mathbb{R}_n} x \frac{1}{\left(\sqrt{2\pi}\sigma\right)^m} \exp\left(-\frac{1}{2\sigma^2}\left||y - Ax|\right|_2^2\right) \prod_{j=1}^{n} p_{X_j}(x_j)\, dx \tag{87}$$

and it is necessary to find a graph-model and apply message-passing algorithm, but this time no low-noise assumption as it is no longer the goal to get the LASSO problem (see [13], Sec. 6.2]). The approximations are similar as for AMP, also including $n \to \infty$ to obtain a feasible algorithm. The key difference is the signal prior, $p_{X_j}(x_j)$, that is now used and not approximated by something "reasonable" from an exponential family. For simplicity the same pdf is assumed for all source vector component $x_j, j = 1, \dots, n$ and all components are independent.

4.4.1. BAMP I

This algorithm is related to the first scheme in AMP because in this case the measurement noise variance σ^2 is known as well. Therefore. in the first iteration we have:

$$\begin{aligned} \hat{x}^{(0)} &= 0_{n\times 1} && \text{(signal vector; dimension n} > \text{m)} \\ z^{(0)} &= y && \text{(dimensions: m} \times 1) \\ c^{(0)} &= \sigma^2 + \frac{1}{m}\left||z^{(0)}|\right|_2^2 && \text{(scalar)} \end{aligned} \tag{88}$$

No difference between both algorithms in this case, the disparity comes when the iteration starts, then algorithm is

$$
\begin{aligned}
u^{(i-1)} &= \hat{x}^{(i-1)} + A^T z^{(i-1)} \\
\hat{x}_j &= F\left(u_j^{(i-1)}; c^{(i-1)}\right) \\
v_j &= G\left(u_j^{(i-1)}; c^{(i-1)}\right) \\
q_j^{(i-1)} &= F'\left(u_j^{(i-1)}; c^{(i-1)}\right) \\
\hat{x}^{(i)} &= \left\{\hat{x}_1^{(i)}, \hat{x}_2^{(i)}, \dots, \hat{x}_n^{(i)}\right\}^T \\
z^{(i)} &= y - A\hat{x}^{(i)} + z^{(i-1)} \frac{1}{m} \sum_{j=1}^{n} q_j^{(i-1)} \\
c^{(i)} &= \sigma^2 + \frac{1}{m} \sum_{j=1}^{n} v_j^t
\end{aligned}
\tag{89}
$$

with the scalar operators (neglecting the iteration index $i-1$ of the scalar arguments).

And with new variables introduced here defined as

$$
\begin{aligned}
F(u_j; c) &= \mathbb{E}_{X_j}\{X_j | U_j = u_j\} \\
G(u_j; c) &= Var_{X_j}\{X_j | U_j = u_j\} \\
F'(u_j; c) &= \frac{d}{du_j} F(u_j; c)
\end{aligned}
\tag{90}
$$

4.4.2. BAMP II

In this case, the measurement noise variance σ^2 is unknown. The initializations are exactly the same as the AMP II algorithm and what changes is the scheme for the iterations.

$$
\begin{aligned}
&\hat{x}^{(0)} = 0_{n\times 1} && \text{(signal vector; dimension n} > \text{m)} \\
&z^{(0)} = y && \text{(dimensions: m} \times 1) \\
&c^{(0)} = \frac{1}{m}\left|\left|z^{(0)}\right|\right|_2^2 && \text{(scalar)}
\end{aligned}
\tag{91}
$$

Then, for the iteration $i = 1, 2, \dots$:

$$
\begin{aligned}
u^{(i-1)} &= \hat{x}^{(i-1)} + A^T z^{(i-1)} \\
\hat{x}^{(i)} &= F\left(u^{(i-1)}; c^{(i-1)}\right) \\
z^{(i)} &= y - A\hat{x}^{(i)} + z^{(i-1)} \frac{1}{m} \sum_{j=1}^{n} F'\left(u^{(i-1)}; c^{(i-1)}\right) \\
c^{(i)} &= \frac{1}{m} \left|\left| z^{(i)} \right|\right|_2^2
\end{aligned}
\tag{92}
$$

If the prior pdf takes a simple form, we can get explicit and relatively simple equations for the MMSE estimator $F\left(u_j; c\right)$, the variance $G(u_j; c)$ of the estimate and the derivative $F'(u_j; c)$. This is useful for an implementation so one can try to obtain each depending on the prior.

In this thesis, we will focus our attention in solving the BAMP algorithm for sparse signals, more specifically in the ones that have a Gaussian prior (with zero mean). Therefore, the computations for F, G and F' will be done for this prior. In the Appendix B of the thesis one can find how this variables are obtained and which is the final result used in the practice.

5. Results

The results obtained in this thesis are based on the theory explained before. What has been done is an implementation of the algorithms of message passing (AMP and BAMP) to compare the differences between them. Some explanation of the code and the results obtained are described below, moreover, one can find all the Matlab code in the Appendix A.

First of all, the algorithm vectorise the image, i.e., stack all columns into one column-vector of x dimensions $n \times 1$, we also create a random pattern of $m \times n$ measurements and we obtain the standard compressed sensing measurement model as:

$$y_{[M]} = A_{[M\times N]} \cdot x_{[N]} + w_{[M]} \text{ with } M \ll N \tag{93}$$

Each of the components of $y_{[M]}$ represents one measurement that results one of the random patterns. The vector $w_{[M]}$ is used to model the measurement of the noise (can be zero). We need to exploit the structure of the image, that is, for instance, sparsity in the DCT/Fourier domain because the image itself will usually not be sparse.

A brief explanation of DCT, in that case the DCT-II used for images is defined as follows

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left(\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right) \quad \text{for} \quad k = 0,1,\dots,N-1 \tag{94}$$

$$x_n = \frac{1}{2}X_0 + \sum_{k=1}^{N-1} X_k \cos\left(\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right) \quad \text{for} \quad n = 0,1,\dots,N-1 \tag{95}$$

If we come back to our problem then it is written following this matrix notation with $D_{[N\times N]}$ the DCT-Transform matrix:

$$X_{[N]} = D_{[N\times N]} x_{[N]} \tag{96}$$

and the inverse transform

$$x_{[N]} = D^T_{[N\times N]} X_{[N]} \tag{97}$$

Note that $\left(D_{[N\times N]}\right)^{-1} = D^T_{[N\times N]}$ because DCT is a unitary transform. In the implementation we will avoid to use the matrix $D_{[N\times N]}$ directly, as for image dimensions of 1000×1000, we have a DCT

matrix of the dimensions $10^6 \times 10^6$. With the DCT-matrix we form the compressed sensing measurement equation used for recovery:

$$\begin{aligned} y_{[M]} &= A_{[M\times N]} \cdot x_{[N]} + w_{[M]} \text{ with } M \ll N \\ &= \underbrace{A_{[m\times n]} D^T_{[N\times N]}}_{=\Phi_{[M\times N]}} X_{[N]} + w_{[M]} \end{aligned} \tag{98}$$

With the compound measurement matrix $\Phi_{[M\times N]}$ we now have a sparse vector $X_{[N]}$ of DCT-coefficients we can recover by using the functions of AMP and BAMP created for the thesis. Note that the size of the measurement matrix $\Phi_{[M\times N]}$ is much smaller than the $N \times N$ DCT-matrix D when $M \ll N$.

After recovery – from which we obtain $\hat{X}_{[N]}$ – we need to transform back to the image domain, i.e.,

$$\hat{x}_{[N]} = D^T_{[N\times N]} \hat{X}_{[N]} \tag{99}$$

Finally, the vector $\hat{x}_{[N]}$ of recovered pixels has to be reshaped into the original pixels.

Once, we know how to compress and recover the image we could jump to the next step. After the compressing of the image we send the signal to our algorithms that return the new signal to be decompressed after their iterations.

Now, we are going to show some results of the code and some plots made to compare and decide which one is the algorithm that is able to reconstruct the best image. In the code, there are explicit some plots and figures.

The first one shows the DCT-coefficients of the original image and the resulting ones from the algorithms of AMP and BAMP. In the code is also implemented a variation of the BAMP algorithm that, instead of using the Gaussian model for all the components of the 40 images, computes the model for each of the components of the 40 images, i.e., if the size of the image is 100×100 we obtain a sigma for the 10.000 components. Therefore, what we have is a Gaussian model for each of the pixels of all the images instead one model for all the pixels and images. For sure, this is an accurate model and as we will see below the results are better for the BAMP-pointwise recovery.

The first result of the Matlab code shows which are differences between each of the DCT-coefficients of the algorithms implemented. It also shows how is the histogram of this signal before the reconstruction.

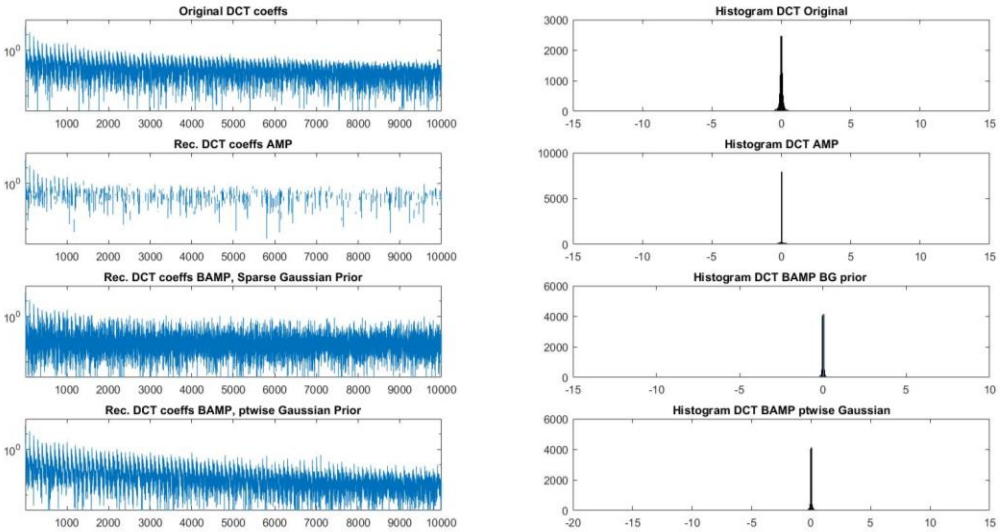


Figure 6. DCT and histogram for the same image but different algorithms

The next plot of the code shows how these DCT-coefficients are distributed in 2D and as in the first picture, we can appreciate the differences between all the algorithms.

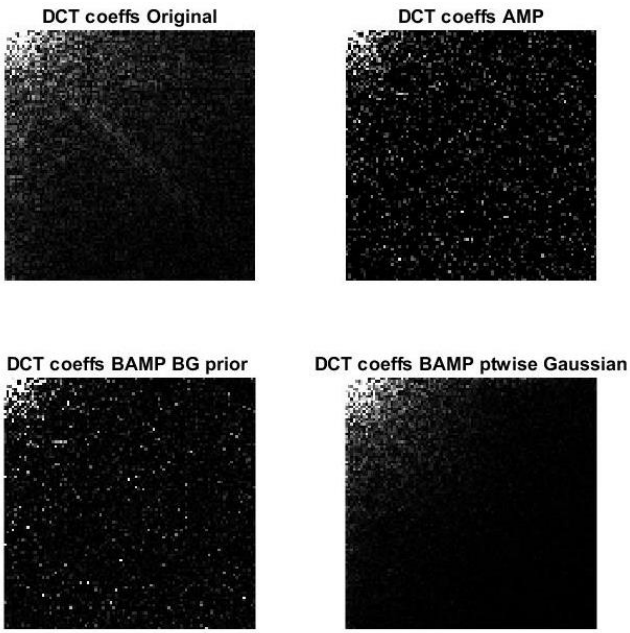


Figure 7. DCT coefficients of the original image, AMP, BAMP and BAMP-pointwise

In the third figure, the image reconstruction is done and we can compare the differences between the images reconstructed and the original one. In this figure we can appreciate the difference between each algorithm.

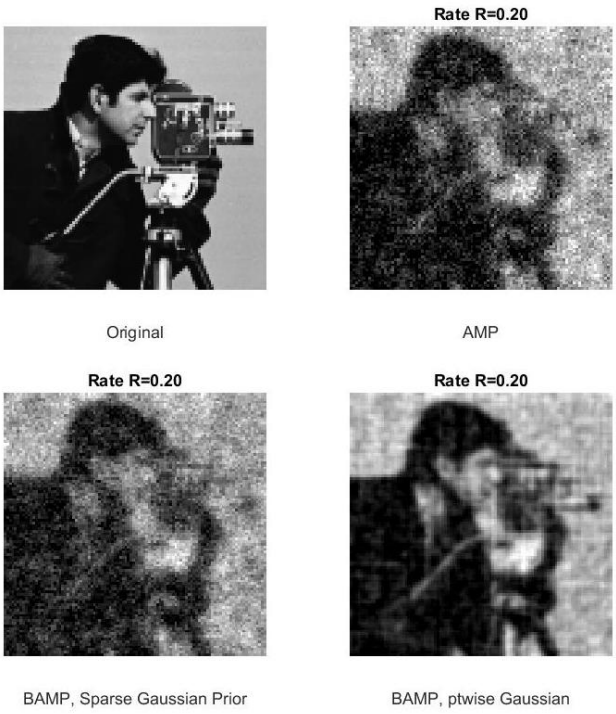
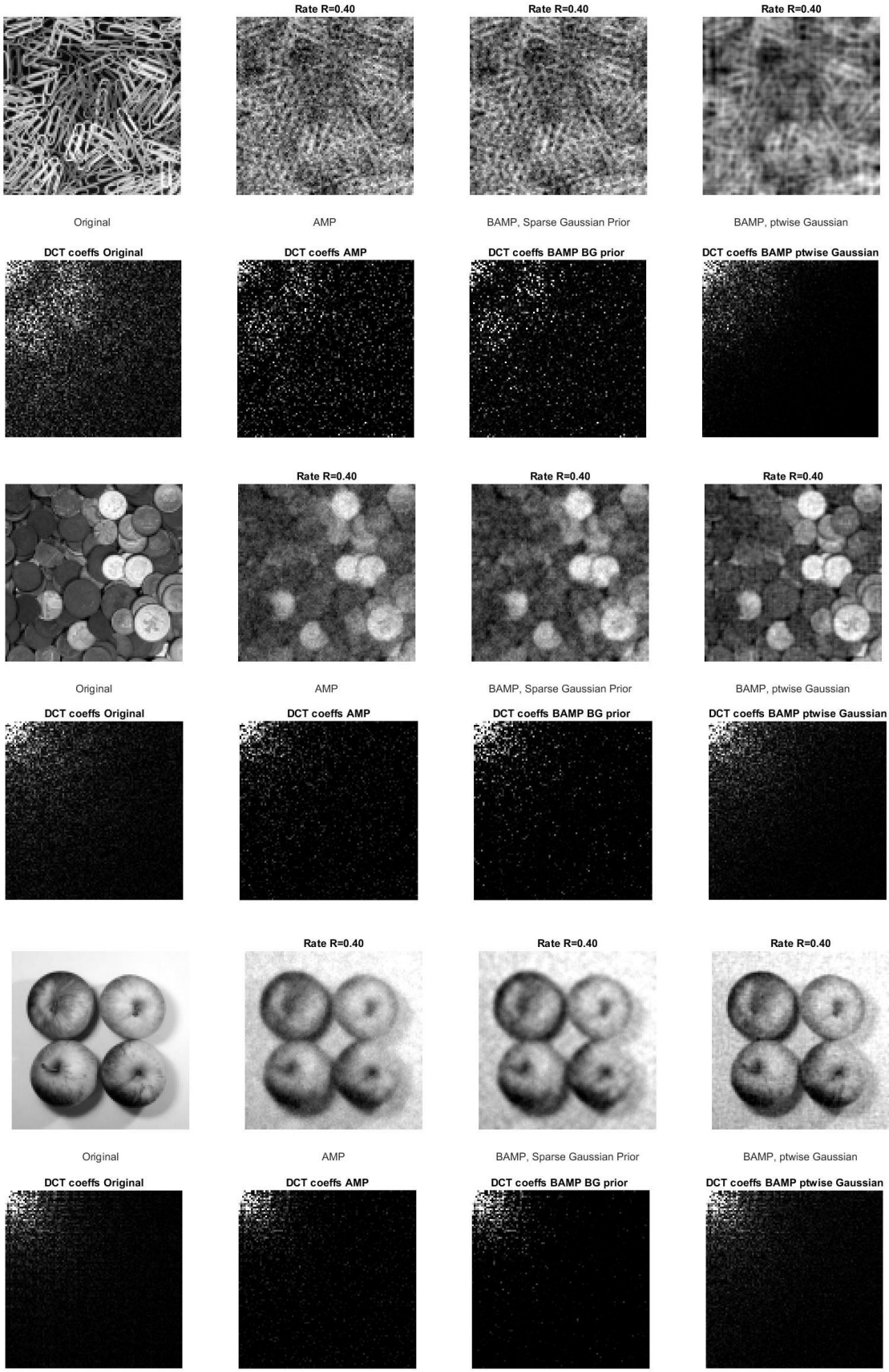
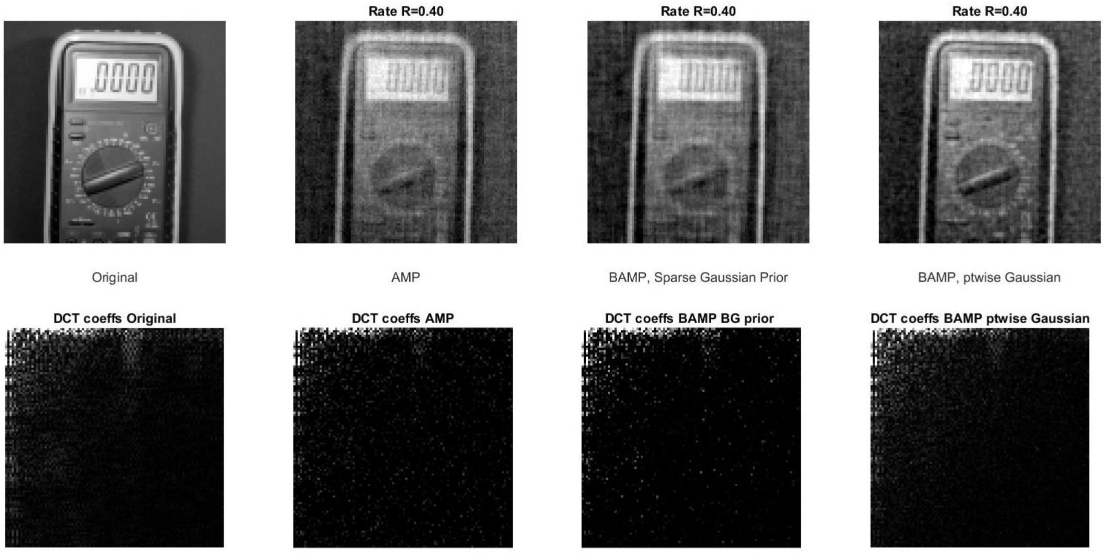


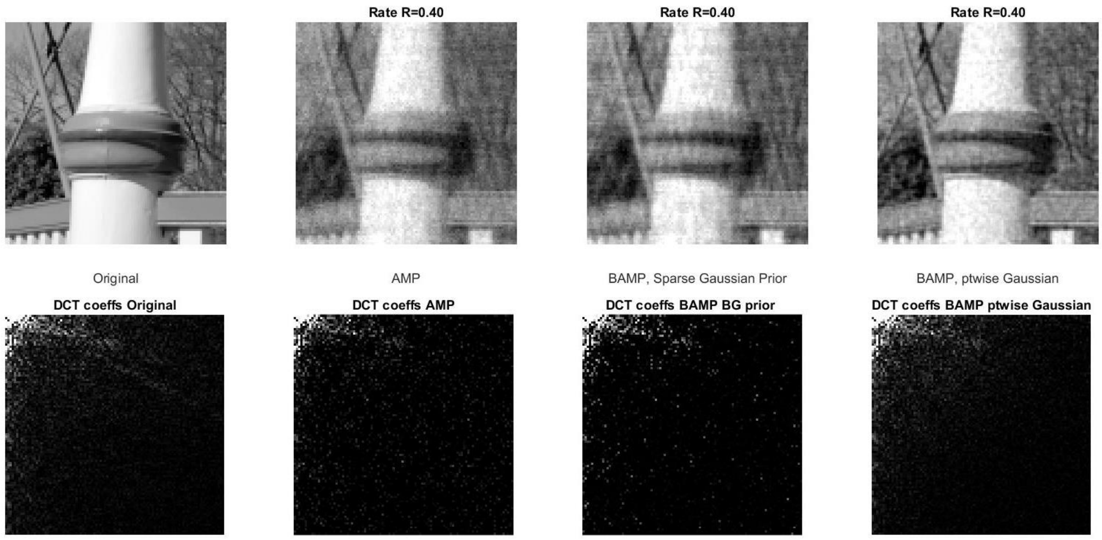
Figure 8. Image reconstruction for the different algorithms

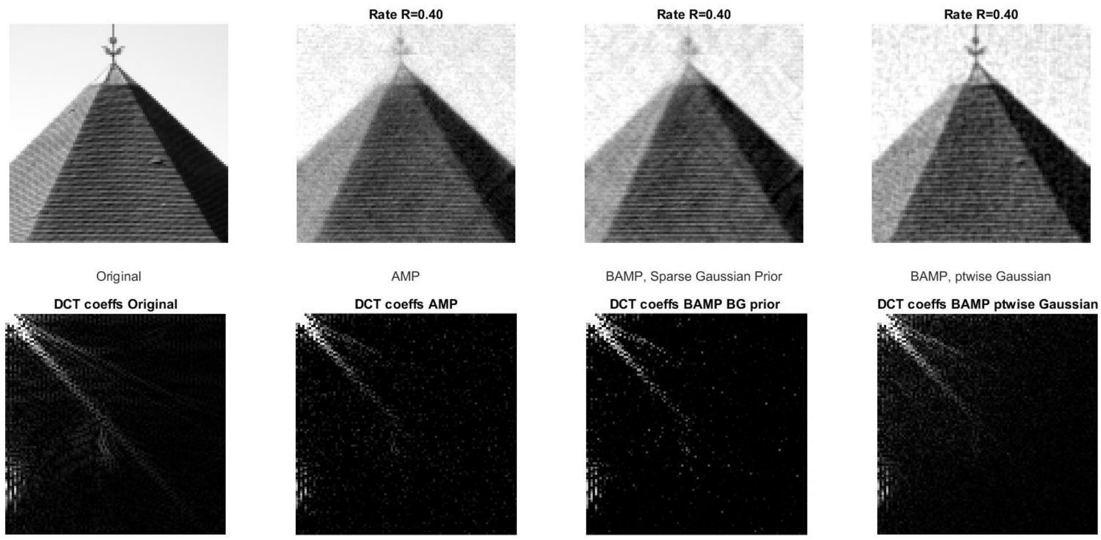
In all these three figures is easy to value which is the best algorithm. As in the first one is difficult distinguish which histogram is more similar as the original one, if we take a look to the DCT-coefficients in the left-side of the first figure and the second one it is clear that the BAMP-pointwise algorithm has the most similar coefficients referred to the original one. In the third figure we can see the differences reflected in the reconstruction of the images for each algorithm, despite the reconstruction not fits perfectly to the original one. We can still differentiate more characteristics in the last picture while in the first one is difficult to realise which is the object.

The next figures, show for more images the differences between algorithms. As 40 images were used to create the Gaussian model for the BAMP algorithm we took advantage of it and if one wants it is possible to plot all the images. In the following pictures there are some examples of the images used. In this case the undersampling ratio is 0.4 instead of the 0.2 above.


Rate R=0.40
Rate R=0.40
Rate R=0.40
Original
AMP
BAMP, Sparse Gaussian Prior
BAMP, ptwise Gaussian
DCT coeffs Original
DCT coeffs AMP
DCT coeffs BAMP BG prior
DCT coeffs BAMP ptwise Gaussian
Rate R=0.40
Rate R=0.40
Rate R=0.40
Original
AMP
BAMP, Sparse Gaussian Prior
BAMP, ptwise Gaussian
DCT coeffs Original
DCT coeffs AMP
DCT coeffs BAMP BG prior
DCT coeffs BAMP ptwise Gaussian
Rate R=0.40
Rate R=0.40
Rate R=0.40
Original
AMP
BAMP, Sparse Gaussian Prior
BAMP, ptwise Gaussian
DCT coeffs Original
DCT coeffs AMP
DCT coeffs BAMP BG prior
DCT coeffs BAMP ptwise Gaussian


Rate R=0.40
Rate R=0.40
Rate R=0.40
Original
AMP
BAMP, Sparse Gaussian Prior
BAMP, ptwise Gaussian
DCT coeffs Original
DCT coeffs AMP
DCT coeffs BAMP BG prior
DCT coeffs BAMP ptwise Gaussian


Rate R=0.40
Rate R=0.40
Rate R=0.40
Original
AMP
BAMP, Sparse Gaussian Prior
BAMP, ptwise Gaussian
DCT coeffs Original
DCT coeffs AMP
DCT coeffs BAMP BG prior
DCT coeffs BAMP ptwise Gaussian


Rate R=0.40
Rate R=0.40
Rate R=0.40
Original
AMP
BAMP, Sparse Gaussian Prior
BAMP, ptwise Gaussian
DCT coeffs Original
DCT coeffs AMP
DCT coeffs BAMP BG prior
DCT coeffs BAMP ptwise Gaussian

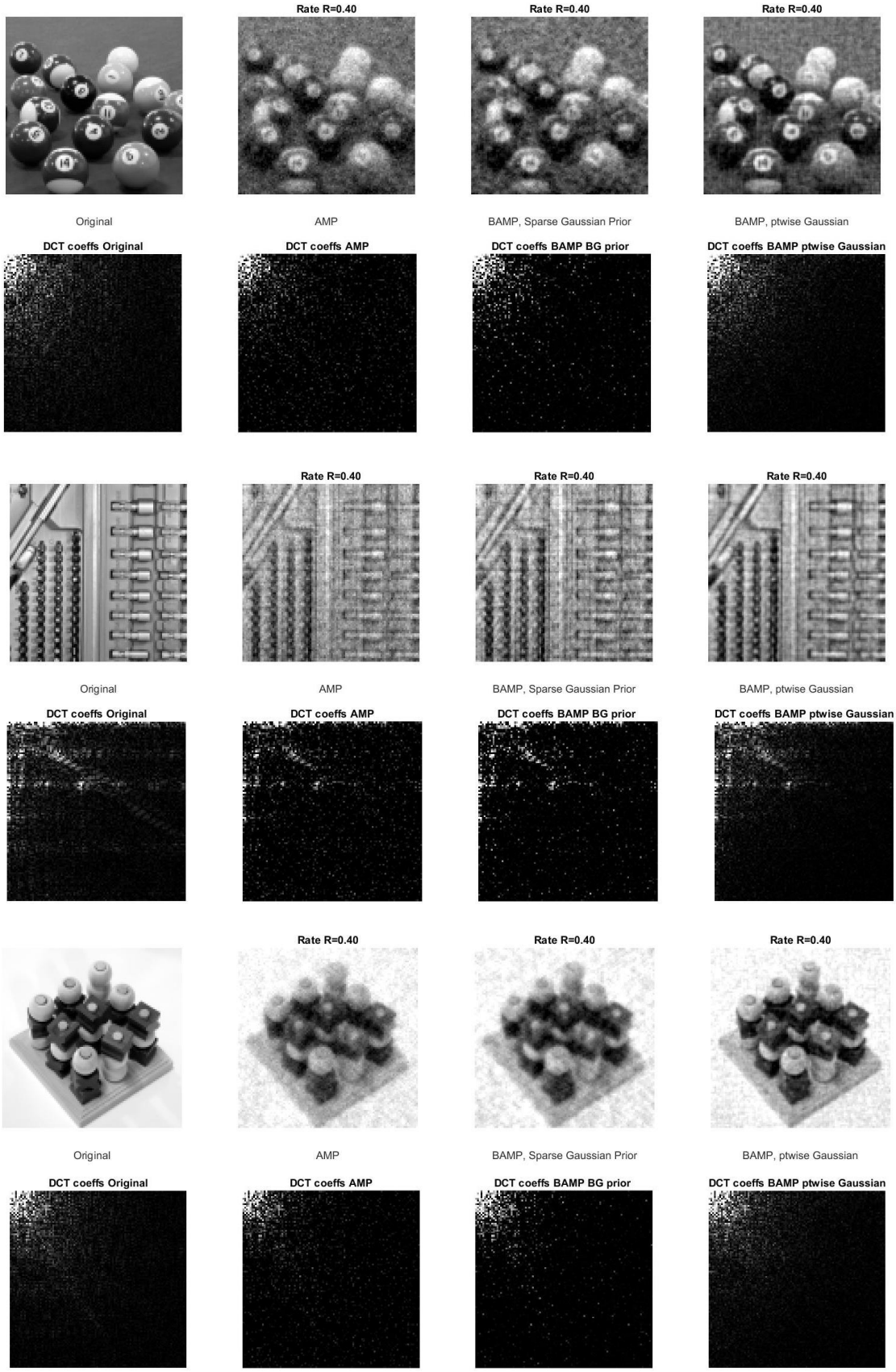


Figure 9. Comparison of the reconstruction and DCT for different images

Another interesting result is to compare the same image but changing the undersampling ratio. In this case we can really appreciate that for low undersampling ratios the BAMP-pointwise algorithm recovers much better the image than the others and for higher undersampling ratios the difference is not that significant.

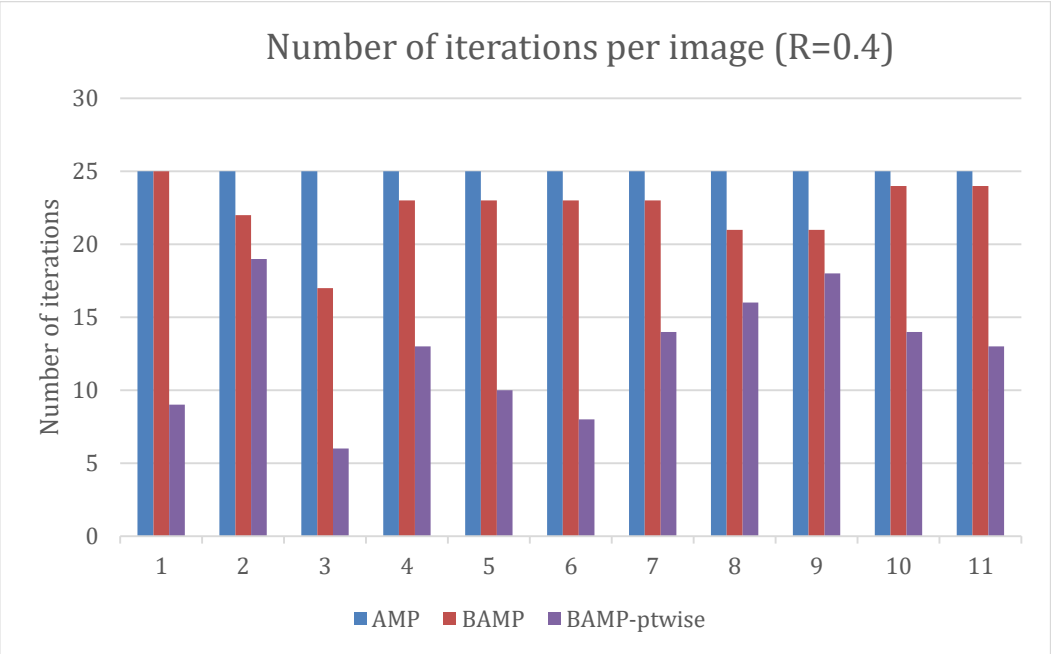
Figure 10. Study of the different image reconstructions for different undersampling ratios

All the measurements before were done without the noise measurement, but if we add to our implementation a Gaussian noise and we take a look at the behaviour for different signal to noise ratio we obtain:

Figure 11. Study of the different image reconstructions for different SNR

Also in this case, the behaviour of the BAMP-pointwise algorithm is still having the best results.

Finally, once explained the differences between algorithms in terms of DCT-coefficients, signal to noise ratio, undersampling ratios we talk now in terms of computation. We refer to the best computation algorithm to the one that needs less iterations to find a suitable result before the stopping criterion breaks. If we simulate the first images and one creates a figure for the number of iterations needed, we obtain:



As can be seen here the AMP algorithm breaks at the 25th iteration while in the worst case of the BAMP-pointwise is at the 19th one. Therefore, the BAMP-pointwise algorithm beats the other two in terms of computation as well.

If one wants to try the Matlab code one will be able to obtain the same figures as in the first figures 7, 8, 9 and 10. The code to obtain the last two pictures are not included in this thesis as they require more computational time.

Note that the results in practice are the expected from the theoretical part.

6. Budget

The main costs of this project comes from the salary of the researches and time spent in it as the software used is free thanks to university license for Matlab. The team for the development of this thesis is formed by two senior engineers (the advisors Norbert Görtz and Gregori Vazquez) and myself as a junior engineer. Considering that the total duration of the project was 18 weeks as presented in the Gantt diagram this would be the budget of the project:

Position	**Amount**	**Wage/hour**	**Dedication**	**Total**
Junior engineer	1	8,00 €/h	25 h/week	3,600 €
Senior engineer	2	20,00 €/h	1 h/week	720 €
			Total	**4,320 €**

7. Conclusions and future development:

During all the project has been demonstrated that theoretically the most advanced compressive sampling algorithms are supposed to obtain better results. The most advanced CS recovery schemes allow for higher resolution, speed and lower measurement ratios. It has been demonstrated that the Bayesian algorithm beats the approximate message passing, also it is fair to say that the pointwise derivation of the BAMP improves the results obtained by the original one. Taking advantage of the Gaussian model obtained pointwise, i.e., collecting different Gaussian models for each of the pixels of 40 training images. The figure 11 shows how is the evolution of the algorithms for different undersampling ratios there we can appreciate that the BAMP-pointwise has the best results possible when it is about lower ratios. Also for high lower measurement noise ratios as it has been seen in the figure 12. Although, all the algorithms are very efficient computationally the table of iterations shows that the algorithm that needs less iterations based on our stopping criterion is the BAMP-pointwise as well.

Despite, it is clear that BAMP-pointwise algorithm is the best I would also like to point out that compressed sensing is really an improvement and all the algorithms (AMP and BAMP) used are able to estimate the image with a certain degree of similarity compared to the original one even using by far less signal measurements. This topic as explained during the thesis is a highly growing field and I believe its applications in mathematics, electrical engineering, statistics and computer science will be in the future relevant for our daily lifes.

In what concerns us, one of things I found the most relevant is the fact of the importance of the pdf. Just estimating the Gaussian for some training images the results are considerably much better, this encourage me to keep studying on this field to discover and to research more algorithms able to estimate images for smaller undersampling ratios, smaller signal to noise ratios and even more efficient computationally.

Future developments of this project could be to apply the same algorithms in Matlab but instead of using black and white images use colour ones. Moreover, as this thesis has been done in software it would be great to try to implement a single-pixel camera hardware to compare if the results expected are the same. The research and implementation for new algorithms in compressed sensing is another development to be done in the future.

Bibliography:

[1] DMD-based Compressive Imaging & Spectroscopy Ting Sun Dharmpal Takhar A 1-Pixel Camera & Beyond - Ting Sun, Dharmpal Takhar, Marco Duarte, Jason Laska, Richard Baraniuk, & Kevin Kelly - http://people.ee.duke.edu/~lcarin/kelly.pdf .

[2] F. Krzakala M. Mézard F. Sausset Y. F. Sun L. Zdeborová "Statistical-physics-based reconstruction in compressed sensing" Phys. Rev. X vol. 2 pp. 021005 May 2012

[3] D. L. Donoho, A. Javanmard and A. Montanari, "Information-Theoretically Optimal Compressed Sensing via Spatial Coupling and Approximate Message Passing," in IEEE Transactions on Information Theory, vol. 59, no. 11, pp. 7434-7464, Nov. 2013.

[4] Eldar, Yonina C., and Gitta Kutyniok, eds. *Compressed sensing: theory and applications*. Cambridge University Press, 2012.

[5] Montanari, Andrea. "Graphical models concepts in compressed sensing." *Compressed Sensing: Theory and Applications* (2012): 394-438.

[6] Tibshirani, Robert. "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society. Series B (Methodological)* (1996): 267-288.

[7] Donoho, David L. "For most large underdetermined systems of linear equations the minimal $\ell 1$-norm solution is also the sparsest solution." *Communications on pure and applied mathematics* 59.6 (2006): 797-829.

[8] Donoho, David L., Arian Maleki, and Andrea Montanari. "Message-passing algorithms for compressed sensing." *Proceedings of the National Academy of Sciences* 106.45 (2009): 18914-18919.

[9] Donoho, David L., Arian Maleki, and Andrea Montanari. "How to design message passing algorithms for compressed sensing." *preprint* (2011).

[10] Bayati, Mohsen, and Andrea Montanari. "The dynamics of message passing on dense graphs, with applications to compressed sensing." *IEEE Transactions on Information Theory*57.2 (2011): 764-785.

[11] Krzakala, Florent, et al. "Probabilistic reconstruction in compressed sensing: algorithms, phase diagrams, and threshold achieving matrices." *Journal of Statistical Mechanics: Theory and Experiment* 2012.08 (2012): P08009.

[12] Goertz, Norbert, et al. "Iterative recovery of dense signals from incomplete measurements." *IEEE Signal Processing Letters*21.9 (2014): 1059-1063.

[13] Som, Subhojit, and Philip Schniter. "Compressive imaging using approximate message passing and a Markov-tree prior." *IEEE transactions on signal processing* 60.7 (2012): 3439-3448.

Glossary

CS: Compressed Sensing

AMP: Approximate Message Passing

BAMP: Bayesian Approximate Message Passing

IHT: Iterative Hard Thresholding

IST: Iterative Soft Thresholding

LASSO: Least Absolute Shrinkage and Selection Operator

MSE: Mean Squared Error