

https://dspace.org/

An l1-norm solution of under-determined linear algebraic systems using a hybrid method

Sejeso, Matthews Malebogo

2016

http://hdl.handle.net/10539/21638

Sejeso, Matthews Malebogo (2016) An l1-norm solution of under-determined linear algebraic systems using a hybrid method, University of Witwatersrand, Johannesburg, <http://wiredspace.wits.ac.za/handle/10539/21638>

Downloaded from a DSpace repository. https://dspace.org/

An ℓ_1-norm Solution of Under-determined Linear Algebraic Systems Using a Hybrid Method

Matthews Malebogo Sejeso

(Student No: 381834)

Supervised by: Prof Montaz Ali

School of Computer Science and Applied Mathematics,

University of the Witwatersrand, Johannesburg, South Africa.

October 2016

Submitted in fulfilment of Master's of Sciences



WITS
UNIVERSITY

Declaration

I, the undersigned, hereby declare that the work contained in this research project is my original work, and that any work done by others or by myself previously has been acknowledged and referenced accordingly. This dissertation has been submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, South Africa, in fulfilment of the requirements for the degree of Masters of Science. The work has not been submitted before for any degree or examination to any other institution.



Matthews Malebogo Sejeso, 24 October 2016

Abstract

The ℓ_1-norm solution to an under-determined system of linear equations $\mathbf{y} = \mathbf{Ax}$ is the sparsest solution to the system. In digital signal processing this mathematical problem is known as compressive sensing. Compressive sensing provides a mathematical framework for sampling and reconstructing an analogue signal at a rate far lower than the rate provided by the standard information theory. The reconstruction from few samples is possible using non-linear optimization algorithms provided that the signal is sparse and the sensing matrix in incoherent. The major algorithmic challenge in compressive sensing is to efficiently and effectively find sparse solutions from minimal measurements. General purpose optimization algorithms are not suitable for solving non-differentiable ℓ_1-minimization problem.

In this dissertation, we survey the major practical algorithms for finding ℓ_1-norm solution of under-determined linear system of equations. Specific attention is paid to computational issues, in which individual methods tends to perform well. We propose a hybrid algorithm that combines complementary strengths of the fixed-point method and the interior-point method. The strong feature of the fixed-point method is its speed, while the strength of the interior-point method is accuracy. The hybrid algorithm combine the two methods in a probabilistic manner. The algorithm tends to prioritise a method that is efficient and robust. The computational performance of the hybrid algorithm is tested on simple signal reconstruction problems. The hybrid algorithm is shown to produce similar recoverability of sparse solution as that of the fixed-point method and the interior-point method. Furthermore the proposed hybrid algorithm is comparative in terms of speed and accuracy with existing methods.

Acknowledgements

I would like to pass my sincere gratitude to my supervisor Prof. Montaz Ali for his guidance and support for making this work possible.

Special thanks to African Institute for Mathematical Science (AIMS) for their financial support through the Ben and Marry Turok Scholarship, and the School of Computer Sciences and Applied Mathematics for the afforded opportunity to complete this study.

To my family and friends I say thank you for your unconditional encouragement, support and love.

Contents

Abstract **ii**

1 Introduction **1**

1.1 Background 2

1.2 Problem Statement 4

1.3 Applications 6

1.4 Dissertation Organization 8

2 Fundamentals of Compressive Sensing Theory **9**

2.1 Sparse and Compressible Signal Models 10

2.2 Incoherence and Restricted Isometry Property 14

2.3 Sparse Signal Recovery 18

2.4 Phase Transition Analysis 22

2.5 Optimality for Non-differentiable Functions 23

3 Algorithms for Compressive Sensing **28**

3.1 Convex Relaxation Algorithms 30

3.2 Greedy Pursuit Algorithms 47

3.3 Iterative Thresholding Algorithms 59

3.4 Combinatorial Algorithms 66

3.5 Bayesian Algorithms 67

4 Hybrid Algorithm for ℓ_1-minimization **68**

4.1 Review of Fixed-point and Interior-point for Hybrid Algorithm 70

4.2 Probabilistic Hybrid Algorithm 73
4.3 Continuation Strategy 77

5 Computational Experiments **79**
5.1 Comparison of Hybrid Algorithm with Constituent Algorithms 79
5.2 Comparison of Hybrid Algorithm with Existing Algorithms 89

6 Conclusion **92**

References **100**

List of Figures

1.1 Nyquist-Shannon sampling vs compressive sampling 4

2.1 Compressive sensing measurement model. 10

2.2 Sparse representation of an image via a multi-scale wavelet transform. (a) Original image. (b) Haar wavelets representation. Large coefficients are represented by light pixels, while small coefficients are represented by dark pixels. Note that most of the wavelet coefficients are close to zero. (c) The wavelets coefficients obeys the power law decay. (d) Reconstruction of the original image by thresholding few large wavelets coefficients. 13

2.3 Best approximation of a point in $\mathbb{R}^2$ by a one-dimensional subspace using the ℓ_p-norms for $p = 2, 1$ and the ℓ_p quasi-norm with $p = \frac{1}{2}$. 19

2.4 Phase transition diagram. 23

3.1 Classification of compressed sensing algorithms. 29

4.1 Schematic diagram of the probabilistic hybrid algorithm. 69

5.1 Phase diagrams illustrating reconstruction ability of fixed-point algorithm: (a) signal reconstruction from noiseless measurements and (b) signal reconstruction from noisy measurements. 81

5.2 Phase diagrams illustrating reconstruction ability of interior-point algorithm: (a) signal reconstruction from noiseless measurements and (b) signal reconstruction from noisy measurements. 82

5.3 Phase diagrams illustrating reconstruction ability of probabilistic hybrid algorithm: (a) signal reconstruction from noiseless measurements and (b) signal reconstruction from noisy measurements. 82
5.4 Speed comparison for different problem sizes 85
5.5 An example of signal reconstruction with fixed-point algorithm. 86
5.6 An example of signal reconstruction with interior point algorithm. 87
5.7 An example of signal reconstruction with probabilistic hybrid algorithm. . . 87
5.8 Probability profile of reconstruction for signal length $n = 2^8$. 88
5.9 Probability profile of reconstruction for signal length $n = 2^{10}$. 88
5.10 Probability profile of reconstruction for signal length $n = 2^{13}$. 89

List of Tables

5.1 Parameters and their default values for hybrid algorithm. 83
5.2 Speed comparison for different problem size. The computation time is reported in seconds. 84
5.3 Comparison of solution accuracy. Root square mean error between the true signal and reconstructed solution. 85
5.4 Comparison of recoverability of hybrid algorithm with other existing algorithms. 90

Chapter 1

Introduction

Many situations in science and technology call for the solution of under-determined linear algebraic systems. In these situations, the features of most interest cannot be directly observed, but have to be inferred from observable quantities. In a wide range of cases, it suffices to assume a linear relationship between the features of interest and the observable quantities. Applications of most interest are those in which the observable quantities are much less than the unknown features, with the knowledge that the unknown features are sparse. Thus, we model the under-determined linear algebraic system, and the problem can be posed as the linear inverse problem. Linear inverse problems arise throughout engineering and mathematical sciences. In most applications, these problems are ill-conditioned; meaning they are very sensitive to the measurements. That is, any small change in measurements can lead to a radically different inference of the sparse solution. Finding the sparse solution of an under-determined linear system of equations has been intensively studied, largely because solving such system of equations constitutes a critical step in emerging methodology in digital signal processing - called *compressive sensing*.

Compressive sensing is a signal processing technique for efficiently acquiring and reconstructing a signal. This is done by finding solutions to under-determined linear algebraic system of equations. The principle of reconstruction is based on convex non-differential optimization theory. There are two requirements under which reconstruction is possible. The

first requirement is *sparsity*, which is the concept that the signal must have few non-zero components relative to its dimension. The second requirement is *incoherence*, which is the idea that the sampling domain and representation domain of the signal must not be correlated. These requirements are explored in details in **Chapter 2**. Since the inception of the fundamental mathematical framework of compressive sensing about a decade ago, there has been an increasing interest in this technique in various disciplines of engineering and mathematical sciences. This is due to high demand of effective and efficient signal processing algorithms. Compressive sensing promises dramatic reductions in cost and power consumption in wide range of signal processing applications. Most of these applications are possible due to the inherent sparsity of many real world signals.

1.1 Background

While the field of compressive sensing has recently gained a significant attention in the signal processing community, there had been hints in this direction dating back about half a century ago. Sparse approximation was used in Seismology as an attractive alternative for Least Square Solutions, see Claerbut and Muir (1973). Santosa and Symes (1986) suggested ℓ_1-norm to recover sparse spike signal. Rudin et al. (1992) used total variation minimization in image processing which is very close to ℓ_1-norm minimization in an attempt to solve these under-determine systems. Some of these developments were a direct result of the inconsistency of traditional signal sampling theory - *Nyquist-Shannon theory*. These inconsistencies will be discussed shortly. The idea of sparse approximation was revamped about a decade ago, when Candes et al. (2006a) and Donoho (2006) independently gave profound results regarding mathematical foundation of compressive sensing. Ever since, the field has witnessed a significant advancement. Below, we discuss the traditional way of sampling data known as Nyquist-Shannon theory. We will highlight some inconsistencies and undesirable effects of the theory. We will then connect it with compressive sensing, highlighting why this novel sampling theory is to revolutionise the digital future.

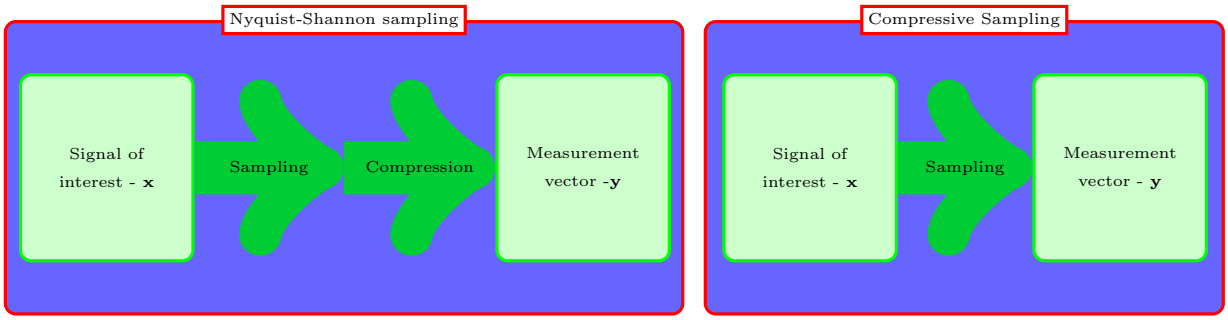
The theoretical foundation of acquiring signals was pioneered by Nyquist (1928) and Shannon (1949) on sampling continuous signals. The Nyquist-Shannon theory states that an analog signal can be reconstructed exactly from its samples, provided that it was sampled at a rate at least twice the highest frequency present in the signal. This rate is known as *Nyquist rate*. Due to Nyquist-Shannon theory, signal processing has shifted from analog domain to digital domain. Digitalization has enabled the creation of sensing systems that are robust, cheaper, and extensively used compared with its analog counterpart.

As a result of this success, the amount of data generated by sensing systems has grown tremendously. Unfortunately, in many applications in various emerging technologies the resulting Nyquist rate has serious repercussions. The Nyquist rate is so high such that too many samples are needed. Despite extraordinary advancement in computational power, this kind of acquisition of signals pose tremendous challenges in terms of computational resources. Additionally, prominent signal structures such as sparsity are not utilized. Generally, signals are not sparse. However, it is often possible for such signals to be sparse in some predetermined domain. The use of Nyquist-Shannon theory would heavily oversample sparse signals since much of the signal has little to no important information.

The conventional way of mitigating computational challenges involved in signal processing is often achieved by compression. Compression aims at finding the most concise representation of a signal with modest distortion. Typical examples of image compressing techniques are *lossless* and *lossy* compressions. Lossless compression minimizes the image size by identifying and eliminating statistical redundancy. No information is lost. On the other hand, lossy compression reduces the image size by identifying and removing unnecessary information. Figure 1.1a, summarises the concept signal acquisition using Nyquist-Shannon sampling with compression. In essence, the traditional signal processing technique uniformly samples a signal at Nyquist rate. Prior to transmission, the samples are compressed, discarding samples of minimal value, thus minimizing storage and ultimately computational cost.

Although, signal compression is useful, compressive sensing has emerged as a new framework for signal sampling. This framework enables a large reduction of sampling and computational cost for acquiring sparse signals. Compressive sensing exploits sparsity of the signal and

the incoherency of samples to achieve its goal. In essence, compressive sensing combines the sampling and compression into one step, see Figure 1.1b. Only samples that contain maximum information about the signal are obtained.



(a) (b)

Figure 1.1: Nyquist-Shannon sampling vs compressive sampling

1.2 Problem Statement

We aim to solve the inverse linear problem. In most applications the problem is ill-posed. Thus, one needs to apply regularization in order to obtain meaningful solutions. Over the last decade, sparsity constraints have emerged as a fundamental type of regularization. The adopted approach seeks a solution to a linear system while requiring that the unknown has few non-zero entries relative to its dimension. The problem can be stated as:

$$\text{Find sparse } \boldsymbol{x} \quad \text{such that } \boldsymbol{\Psi x} = \boldsymbol{y}.$$

Traditionally, the problem is posed as a ℓ_0-minimization problem - which minimize $\|\boldsymbol{x}\|_0$. Here $\|\boldsymbol{x}\|_0$ is the ℓ_0 quasi norm defined as $\|\boldsymbol{x}\|_0 = |\{i : x_i \neq 0\}|$. Under this formulation, a sparse signal can be reconstructed precisely due to the fact that ℓ_0 quasi norm function counts number of non-zero elements of the solution. Unfortunately, although ℓ_0 quasi norm returns the exact solution, the problem is combinatorial minimization and it is generally NP-hard. Algorithms that solve ℓ_0-minimization problem are computationally intractable.

Candes et al. (2006a) provided a breakthrough that overcome the drawback of ℓ_0-minimization problem. Their profound idea was to pose the problem as ℓ_1-minimization program,

$$\min \|\boldsymbol{x}\|_1, \quad \text{subject to} \quad \boldsymbol{\Psi x} = \boldsymbol{y}. \tag{1.2.1}$$

The ℓ_1-norm of $\boldsymbol{x}$ is defined as $\|\boldsymbol{x}\|_1 = \sum |x_i|$. The ℓ_1-norm promotes sparsity and the problem becomes a convex optimization program. Thus, algorithms for solving problem (1.2.1) are computationally tractable. The main challenge is the nondifferentiability of the problem. Additionally, in practical problems the size of $\boldsymbol{x}$ is usually large.

Recently, algorithms for solving the sparse problem (1.2.1) have been intensively studied. This is largely because solving such an equation constitutes a critical step in emerging methodology in digital signal processing. In fact, problem (1.2.1) forms the basis of compressive sensing theory. Alternatively, the problem is known as sparse approximation problem. The problem arises in numerous areas such as statistics see Tibshirani (1996), image processing see Duarte et al. (2008); Lustig et al. (2008), machine learning see Davenport et al. (2010) and many more. Mathematical optimization provides tools to conceptualize, formulate and solve many of these problems. However, due to nondifferentiability of the problem, standard optimization techniques are not capable of solving such formulation.

Goals: The goal of this dissertation, in a broad sense, is to develop an efficient and effective method for solving the sparse structured convex optimization problem. This will be achieved through the following:

- Survey and give an overview of algorithms for the sparse approximation problem. This will be achieved by describing their computational requirements and relationships between them.
- Develop a hybrid method in the sense that complementary strengths of two or more algorithms are combined to form a more robust algorithm. The newly developed algorithm should perform well on various applications reported in literature.
- Different applications of interest are discussed and computational performance of the proposed hybrid algorithm is tested.

An algorithm is said to be hybrid when there is a choice between at least two distinct algorithms, each of which solves the same problem. This is generally done by combining the desired features of each algorithm, so that the overall algorithm is better than the individual constituents. We propose a probabilistic hybrid algorithm that combines the important features of fixed-point method and interior-point method. In particular, we take advantage of the speed of fixed-point iteration and combine that with high accuracy of interior-point iteration. The resulting algorithm is shown to be comparatively efficient and robust when tested on benchmarked problems from different applications reported in literature.

1.3 Applications

Sparse recovery problem has applications in many areas and it has been applied to the following domain: compressive imaging, machine learning, sensor networks and so on. In practice, sparse signal arise in a very natural way. Hence, compressive sensing lends itself very well to many settings. Before describing compressive sensing theory, we motivate it with some applications.

- **Compressive Imaging.** Compressive sensing has far reaching implications on compressive imaging systems and cameras. It reduces the number of measurements. This practically implies reduction of power consumption, computational complexity and storage without sacrificing the spatial resolution. With the advent of single pixel camera, by RICE University, imaging system are due to be transformed drastically. The camera is based on a single photon detector adaptable to image at wavelengths which were impossible with conventional cameras, see Takhar et al. (2006); Wakin et al. (2006); Duarte et al. (2008). The advantage of single pixel camera over conventional camera is that a tiny number of measurements are required to create an image. The resulting image does not suffer from the usual aberration and focussing problems associated with the lens.

- **Medical Imaging.** Compressive sensing has its early success in medical imaging, particularly in Magnetic Resonance Imaging (MRI). MRI, like angiograms has sparsity properties in domain such as Fourier or wavelet basis, Lustig et al. (2007). Generally, MRI is a costly and time consuming process because of its data collection process which is dependent upon physical and physiological constraints. However, the introduction of compressive sensing based techniques has reduced the computational cost of collecting data while retaining the image quality. Sparse MRI is an active area of research in compressive sensing community. In the recent past, a number of algorithms have been specifically designed for it, see Lustig et al. (2007); Gamper et al. (2008); Lustig et al. (2008).

- **Microarrays Sequencing.** Microarray are used in biology to identify specimens, using fluorescent tags to identify where samples bind; most samples have only a few active parts, thus using ideas from compressive sensing it is possible to take many fewer measurements and still accurately infer which specimens are present. Recent work in industry has suggested that this approach is useful in practice, see Mohtashemi et al. (2010); Sheikh et al. (2007); Milenkovic et al. (2007).

- **Compressive Manifold Learning.** Normally, data possess merely s intrinsic degrees of freedom. This data is assumed to lie on a s-dimensional manifold in the n-dimensional space, where $s \ll n$. Once the manifold model is identified, any point can be represented using essentially s pieces of information. Manifold learning methods embed high-dimensional space data in a low-dimensional space while preserving their characteristic properties. Therefore compressive sensing fits well on these kind of problems. A detailed discussion of compressive manifold learning is given by Davenport et al. (2010).

An elaborate discussion of all these topics would go beyond the scope of this dissertation. Digital and Signal Processing keeps repository of latest developments in compressed sensing theory and applications.

1.4 Dissertation Organization

The dissertation is organized in the following manner. In **Chapter 2**, we review the compressed sensing theory. We discuss the design of the measurement matrix and state the condition that makes it possible to find a sparse solution from an under-determined linear algebraic system of equations. **Chapter 3** provides a detailed review of some existing methods for finding ℓ_1-norm solutions of under-determined systems of linear equations. In **Chapter 4**, we develop a probabilistic hybrid algorithm for finding the ℓ_1-norm solutions of an under-determined systems of equations. In **Chapter 5**, we demonstrate the computational performance of the developed probabilistic hybrid algorithm. A comparison to constituent algorithms and other existing algorithms is provided. We give a summary of the dissertation and possible further research options in **Chapter 6**.

Chapter 2

Fundamentals of Compressive Sensing Theory

Compressive sensing provides a mathematical framework for the sampling and reconstruction of sparse signals from few measurements acquired in a domain incoherent with the sparsity domain. There are two main research directions in this topic. The first is the theoretical foundation that will allow the minimum number of incoherent measurements necessary to sample and reconstruct an original sparse signal accurately with high probability. The second is to develop efficient optimization algorithms for signal reconstruction. We discuss some of the existing algorithms in **Chapter 3**. This chapter focuses on the first research direction.

The objective of this chapter is to address the following questions:

- How to design an under-determined sensing matrix that efficiently capture the main information of the measured signal?
- What is the minimum number of measurements necessary to reconstruct the original signal with high probability?
- What are the available optimization techniques that find sparse solutions of the linear inverse problem?

In addition to providing answers to the above questions, we will discuss how to evaluate the

performance of reconstruction methods and also present non-smooth optimization theory in brief.

2.1 Sparse and Compressible Signal Models

Signals can be thought of as real-valued functions having a domain which can be either continuous or discrete, and either infinite or finite. For computational simplicity, we restrict the discussion to signals that are discrete and finite. Thus, signals can be modelled as vectors in an appropriate vector space. Vector space allows us to apply intuition and tools from Euclidean geometry. We can describe, compare and analyse signals of interest using tools such as length and angles.

Let $\boldsymbol{x}$ be a discrete signal, which can be viewed as an $n \times 1$ column vector in $\mathbb{R}^n$. Given an orthonormal bases matrix $\boldsymbol{\Phi} \in \mathbb{R}^{n \times n}$ whose columns are the basis vectors $\{\boldsymbol{\phi}_i\}_{i=1}^n$, any signal $\boldsymbol{x}$ can be represented in terms of this basis as

$$\boldsymbol{x} = \sum_{i=1}^{n} z_i \boldsymbol{\phi}_i, \quad \text{or equivalently} \quad \boldsymbol{x} = \boldsymbol{\Phi} \boldsymbol{z}. \tag{2.1.1}$$

Here $\boldsymbol{z} \in \mathbb{R}^n$ is a column vector of coefficients. These coefficients are computed as $z_i = \langle \boldsymbol{x}, \boldsymbol{\phi}_i \rangle = \boldsymbol{\phi}_i^T \boldsymbol{x}$, where $\langle \cdot, \cdot \rangle$ is the inner product of its arguments and $(\cdot)^T$ denotes transposition operator acting on the input argument.

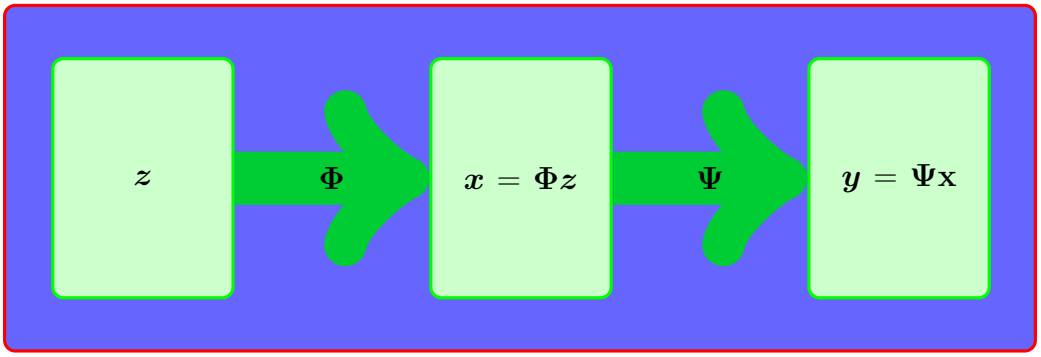


Figure 2.1: Compressive sensing measurement model.

In general sampling methodology is concerned with a collection of measurements y_i. Sampling is done by correlating the signal $\boldsymbol{x}$ with a set of sensing functions $\{\boldsymbol{\psi}_i\}_{i=1}^m$, that is

$$y_i = \langle \boldsymbol{x}, \boldsymbol{\psi}_i \rangle \text{ for } i = 1, \ldots, m \quad \text{or equivalently} \quad \boldsymbol{y} = \boldsymbol{\Psi}\boldsymbol{x}. \tag{2.1.2}$$

Here $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$ is called the sensing or measurement matrix, whose rows are formed by $\boldsymbol{\psi}_i^T$, see Figure 2.1. The choice of sensing functions $\boldsymbol{\psi}_i$ provides the flexibility in the design as how to gather information about $\boldsymbol{x}$. Combining equation (2.1.1) and (2.1.2), the compressive measurements can be written as $\boldsymbol{y} = \boldsymbol{\Psi}\boldsymbol{x} = \boldsymbol{\Psi}\boldsymbol{\Phi}\boldsymbol{z}$, and we define $\boldsymbol{A} = \boldsymbol{\Psi}\boldsymbol{\Phi} \in \mathbb{R}^{m \times n}$. The process is non-adaptive, meaning $\boldsymbol{\Psi}$ is fixed and does not depend on the signal $\boldsymbol{x}$.

The major challenge associated with compressed sensing is in the case where $m << n$. One needs to solve an under-determined system of linear equations in order to recover the original signal $\boldsymbol{x}$ from few measurements $\boldsymbol{y}$. From the basic theory of linear algebra, solving such system will produce infinitely many solutions since there are far much fewer equations than unknowns. Thus, it is necessary to impose constraints on candidate solutions to identify the desirable solution. A powerful constraint that can be used in this regard is sparsity. Sparsity of a signal is discussed in the next subsection.

2.1.1 Sparse Signals. A signal is said to be sparse if its number of non-zero components is small relative to its dimension. The sparsity of a signal is defined as the number of non-zero components in it. A signal is said to be s-sparse if its support is of cardinality less than or equal to s. Support of a signal $\boldsymbol{x} \in \mathbb{R}^n$ is defined as

$$supp(\boldsymbol{x}) := \{i : x_i \neq 0\}. \tag{2.1.3}$$

Typically, signals are not naturally sparse, but admit sparse representation in some appropriately chosen basis $\boldsymbol{\Phi}$, see Figure 2.2. We will refer to $\boldsymbol{\Phi}$ as the sparsity basis or sparsity matrix. In practice the observable signal is not directly sparse. Under such situations we refer to $\boldsymbol{x}$ as being s-sparse with understanding that we can express the signal as $\boldsymbol{x} = \boldsymbol{\Phi}\boldsymbol{z}$, with $supp(\boldsymbol{z}) \leq s$. The concept of sparsity is useful in practice. It reduces data storage and transmission which can be quite expensive in some applications. One needs to store and transmit coefficients of only dominating information, instead of redundant information.

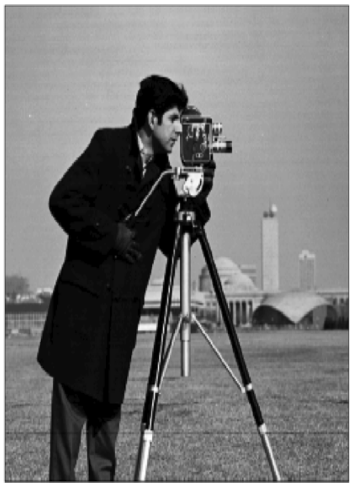
2.1.2 Compressible Signals. In practice, signals are often not exactly sparse, rather they are compressible. Compressible signals are characterized by a few large components and they can be approximated by sparse signals, Davenport et al. (2014). If one sets the small components to zero, the remaining large components can represent the original signal with hardly noticeable distortion. To quantify this, consider the rate at which the magnitude of components decay. Assume that there exists some basis such that the magnitude of components obey a power law decay. Specifically, if $\boldsymbol{x} = \boldsymbol{\Phi z}$ (i.e. $\boldsymbol{x}$ is compressible in the basis $\boldsymbol{\Phi}$) and by sorting the components of $\boldsymbol{z}$ such that $|z_1| \geq |z_2| \geq \cdots \geq |z_n|$, then the components obey the power law decay if there exists constants $C, q > 0$ such that

$$|z_i| \leq Ci^{-q}, \qquad \text{for } i = 1, 2, \ldots, n. \tag{2.1.4}$$

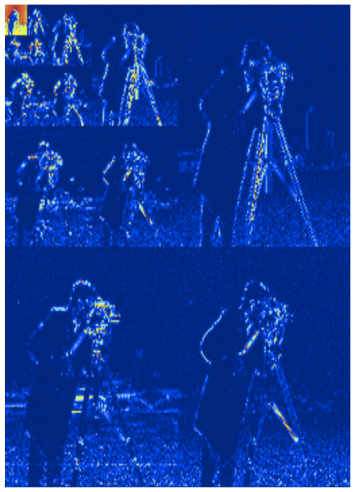
The larger the value of q, the faster the magnitudes of the signal decay, and the more compressible a signal is. Because the magnitudes of components decay so rapidly, compressible signals can be represented accurately by s components (the sparsity level s is far much less than n). The following example from Eldar and Kutyniok (2012) illustrates the concepts of sparsity and compressibility.

Example 2.1. *(Eldar and Kutyniok (2012)). Natural images are characterised by large smooth or textual regions and relatively few sharp edges, see Figure 2.2a. Signals with this structure are known to be very nearly sparse when represented using discrete wavelet transform, see Figure 2.2b. This particular type of wavelet transform is called the Haar wavelet transform. The general concepts of wavelet is to recursively divide the image into low and high frequency components. The lowest frequency components provide the coarse scale approximation of the image, while the higher frequency components fill in the details and resolves edges. The wavelet transformation clearly shows that only few coefficients are non-zero. The image is sparse under the wavelets basis. Figure 2.2c shows the plot of sorted absolute values of the components of Haar wavelet transform. The reconstructed image after keeping only a small fraction of components with large magnitude is shown in Figure 2.2d. This reconstruction provides a good approximation of the original image. Hence the signal can be efficiently reconstructed by, thresholding the components, to obtain a s-sparse good*

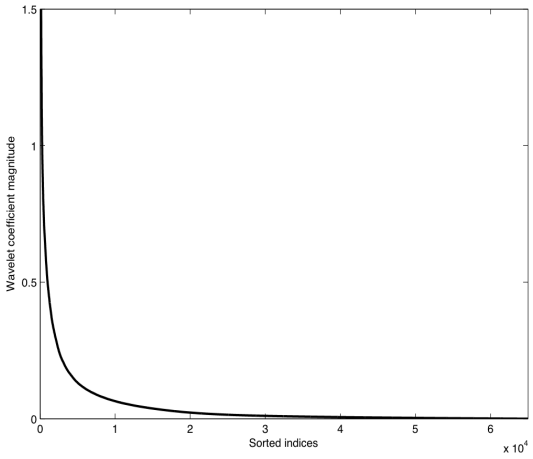
approximate signal.



(a) Original image



(b) Haar wavelets representation



(c) Sorted magnitudes of Haar wavelets coefficients.



(d) Reconstruction of the original image.

Figure 2.2: Sparse representation of an image via a multi-scale wavelet transform. (a) Original image. (b) Haar wavelets representation. Large coefficients are represented by light pixels, while small coefficients are represented by dark pixels. Note that most of the wavelet coefficients are close to zero. (c) The wavelets coefficients obeys the power law decay. (d) Reconstruction of the original image by thresholding few large wavelets coefficients.

2.2 Incoherence and Restricted Isometry Property

The sparsity of the solution vector is a necessary but not a sufficient condition for finding a unique solution to an under-determined system of linear equations. A sufficient condition is associated with the design of the measurement matrix $\boldsymbol{\Psi}$. Rather than presenting a direct design procedure, we present some conditions on the measurement matrix that will guarantee the recovery of unique sparse solution. Such conditions include *incoherence* between the measurement matrix $\boldsymbol{\Psi}$ and the sparsity matrix $\boldsymbol{\Phi}$. Incoherence is related to *restricted isometric property*. Restricted isometric property is associated with the matrix $\boldsymbol{A} = \boldsymbol{\Psi}\boldsymbol{\Phi}$, and it is more powerful than the incoherence, Candes and Wakin (2008); Baraniuk et al. (2008); Davenport and Wakin (2010).

2.2.1 Incoherence. Incoherence extends the duality between the measurement matrix $\boldsymbol{\Psi}$ and sparsity matrix $\boldsymbol{\Phi}$. It asserts that, if a signal is sparse in $\boldsymbol{\Phi}$, then sampling of measurements should be well spread out in $\bar{\boldsymbol{\Psi}}$. Note that, $\bar{\boldsymbol{\Psi}}$ is a set of all possible sensing basis vectors, while $\boldsymbol{\Psi}$ is formed by selecting only m basis vectors from $\bar{\boldsymbol{\Psi}}$. The concept of incoherence is often easily understood through mutual coherence. Consider the following definition:

Definition 2.1 (Mutual Coherence, Candes and Wakin (2008))**.** *The coherence (denoted by μ) between measurement matrix $\bar{\boldsymbol{\Psi}} \in \mathbb{R}^{n\times n}$ and sparsity matrix $\boldsymbol{\Phi} \in \mathbb{R}^{n\times n}$ is defined as*

$$\mu(\bar{\boldsymbol{\Psi}}, \boldsymbol{\Phi}) = \max_{1<i,j<n} |\langle \boldsymbol{\psi}_i, \boldsymbol{\phi}_j \rangle|. \tag{2.2.1}$$

Coherence measures the absolute largest correlation between any two columns of $\bar{\boldsymbol{\Psi}}$ and $\boldsymbol{\Phi}$. Assuming that $\bar{\boldsymbol{\Psi}}$ and $\boldsymbol{\Phi}$ are orthonormal, then it follows from linear algebra that the range of mutual coherence is bounded by $\mu(\bar{\boldsymbol{\Psi}}, \boldsymbol{\Phi}) \in [1, \sqrt{n}]$. If two bases are correlated then coherence is large. Otherwise, coherence is small and the bases are said to be incoherent. Compressive sensing is mainly concerned with low coherence between measurement basis and sparsity basis. The following theorem by Candes and Romberg (2006a) relates low mutual coherence with recoverability of sparse solution from ℓ_1-minimization formulation.

Theorem 2.1 (Recoverability from mutual coherence, Candes and Romberg (2006a))**.** *Assume that the measurements are noise free and suppose* $\boldsymbol{x} \in \mathbb{R}^n$ *is* s*-sparse. Let* $\boldsymbol{\Psi}$ *be constructed by selecting* m *basis vectors from* $\bar{\boldsymbol{\Psi}}$ *uniformly at random. If*

$$m \geq C \cdot s \cdot \mu(\bar{\boldsymbol{\Psi}}, \boldsymbol{\Phi})^2 \cdot \log(n), \tag{2.2.2}$$

for some positive constant C*, then with high probability* $\boldsymbol{x}$ *is a unique solution of* ℓ_1*-minimization problem.*

The above theorem indicates that in addition to the size and sparsity of signals the required number of measurements depend on the mutual coherence. It provides some insights on how the choice of $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ affect the efficiency of compressive sensing schemes. The role of coherence is completely transparent; the smaller the coherence, the fewer measurements needed. In fact, if mutual coherence is equal or close to one, then any order of $s \log n$ measurements suffice to reconstruct a signal of length n from the ℓ_1-minimization problem, Candes and Romberg (2006a). **Example 2.2** provides pairs of matrices $\bar{\boldsymbol{\Psi}}$ and $\boldsymbol{\Phi}$ that have low coherence.

Example 2.2 (Candes and Wakin (2008))**.** *The following are examples of basis pair* $(\bar{\boldsymbol{\Psi}}, \boldsymbol{\Phi})$ *that exhibits maximal incoherence.*

- ***$\bar{\boldsymbol{\Psi}}$ is spike basis and $\boldsymbol{\Phi}$ is Fourier basis.*** *In this case, the* k*-th basis vector of* $\bar{\boldsymbol{\Psi}}$ *is* $\boldsymbol{\psi}_k(t) = \delta(t-k)$*, where* δ *is the Dirac function and the* j*-th basis vector of* $\boldsymbol{\Phi}$ *is* $\boldsymbol{\phi}_j(t) = n^{-\frac{1}{2}} e^{-2i\pi jt}$*, where* $i = \sqrt{-1}$ *is the imaginary number. This corresponds to the classical sampling scheme in time or space. Here* $\mu(\bar{\boldsymbol{\Psi}}, \boldsymbol{\Phi}) = 1$ *and therefore the basis pair* $\bar{\boldsymbol{\Psi}}$ *and* $\boldsymbol{\Phi}$ *have minimal mutual coherence.*
- ***$\bar{\boldsymbol{\Psi}}$ is a random matrix and $\boldsymbol{\Phi}$ is any fixed basis.*** *With high probability, the coherence between any orthonormal basis* $\bar{\boldsymbol{\Psi}}$ *selected at random and any fixed basis* $\boldsymbol{\Phi}$ *is about* $\sqrt{2 \log n}$*. This is applicable when entries of* $\bar{\boldsymbol{\Psi}}$ *are samples of independent and identically distributed (iid) random variables from Gaussian or Bernoulli distributions.*

2.2.2 Restricted Isometric Property (RIP). Theorem 2.1 shows the recoverability of noise free measurements and exact sparse signals. Often in practice, signals are not exactly sparse but are compressible and they are contaminated with noise. The issue here is whether or not is it possible to obtain accurate stable reconstruction of such signals from under-sampled measurements. Under such situations, the RIP plays a similar role as incoherence. RIP provides sufficient conditions on the matrix $\boldsymbol{A} = \boldsymbol{\Psi}\boldsymbol{\Phi}$ that ensure recoverability of unique sparse solution. RIP was first introduced by Candes and Tao (2005) and has proved to be very useful in studying the general robustness of compressive sensing. Consider the following definition:

Definition 2.2 (Restricted Isometric Property, Candes and Wakin (2008))**.** *A matrix* $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ *is said to satisfy the restricted isometric property of order* s*, if there exist a small number* $\delta_s \in (0,1)$ *such that*

$$(1-\delta_s)\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{A}\boldsymbol{x}\|_2^2 \leq (1+\delta_s)\|\boldsymbol{x}\|_2^2 \tag{2.2.3}$$

hold for all s*-sparse signals* $\boldsymbol{x} \in \mathbb{R}^n$*, where* $\|\boldsymbol{x}\|_2 = \left(\sum x_i^2\right)^{1/2}$ *is the* ℓ_2*-norm.*

When a matrix $\boldsymbol{A}$ satisfies the RIP, then the Euclidean length of any s-sparse vector $\boldsymbol{x}$ is approximately preserved by taking $\boldsymbol{A}\boldsymbol{x}$, which in turn implies that s-sparse vector cannot be in $\mathcal{N}(\boldsymbol{A}) = \{\boldsymbol{x} : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}\}$, the null space of $\boldsymbol{A}$, Candes and Wakin (2008). Clearly, this is very important as otherwise there will be no hope of reconstructing these s-sparse signals. The RIP can also be interpreted as, all subsets of s columns of $\boldsymbol{A}$ being nearly orthogonal.

To make use of the RIP, we need to establish the kind of matrices that have small restricted isometric constant δ_s. We also need to determine the connection between the RIP, number of required measurements m and the sparsity level s. Although it is difficult to test whether a given matrix satisfies the RIP condition, it has been shown that with high probability, random matrices satisfy the RIP with few measurements. Consider the following matrices from Candes and Wakin (2008), which are examples of random matrices commonly used in compressive sensing.

- **Gaussian measurements** - entries of the matrix $\boldsymbol{A}$ are chosen independently from normal distribution with mean zero and variance $1/m$.
- **Bernoulli measurements** - entries of the matrix $\boldsymbol{A}$ are independently chosen from symmetric Bernoulli distribution $\mathbb{P}(\boldsymbol{A}_{ij} = \frac{1}{\sqrt{m}}) = \frac{1}{2}$.

There are different measurement bounds for random matrices, but the following theorem by Baraniuk et al. (2008) presents a paraphrase of an elegant result that sums up the general idea.

Theorem 2.2 (Measurement bounds based on the RIP by Baraniuk et al. (2008))**.** *Let the entries of the matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be drawn from independently identical distribution (i.i.d) according to either a zero-mean Gaussian distribution or Bernoulli distribution, and scaled so the expected value of each column is one. For any integer $s = 1, 2, \ldots$ and δ_s, with overwhelming probability the matrix $\boldsymbol{A}$ will satisfy the restricted isometry property with a constant $\delta_s \in (0, 1)$ as long as*

$$m \geq C \cdot s \cdot \log\left(\frac{n}{s}\right), \tag{2.2.4}$$

for some constant C, which depends on each instance.

It has been shown by Baraniuk et al. (2008) that for any given $\boldsymbol{\Phi}$, if $\boldsymbol{\Psi}$ is selected as one of previously mentioned distributions, then with overwhelming probability, the matrix $\boldsymbol{A} = \boldsymbol{\Psi}\boldsymbol{\Phi}$ obeys the RIP provided that the inequality (2.2.4) is satisfied. Note that this random measurement matrix $\boldsymbol{\Psi}$ is in a sense universal, Baraniuk et al. (2008). That is the sparsity basis need not to be known when designing the measurement system. The sole purpose of the sparsity matrix $\boldsymbol{\Phi}$ is to induce sparsity on the signal. The measurement matrix $\boldsymbol{\Psi}$ is the most important element is compressive sensing. Therefore, from now onwards we shall use $\boldsymbol{\Psi}$ with the knowledge that the measurement matrix is fully represented by $\boldsymbol{A} = \boldsymbol{\Psi}\boldsymbol{\Phi}$.

It is worth noting another useful result in compressive sensing concerning Fourier matrices. Fourier matrices are significant since the fast Fourier transform allows fast matrix factorization. The number of required measurement is similar to (2.2.4), except that the $\log(\frac{n}{s})$ factor is replaced by $\log^4(n)$, Candes and Wakin (2008). This bound may be lowered

in future, so it is not necessarily indicative that Fourier matrices are inferior to random matrices.

2.3 Sparse Signal Recovery

Compressive sensing protocol acquires linear measurements $\boldsymbol{y} \in \mathbb{R}^m$ of a sparse signal $\boldsymbol{x} \in \mathbb{R}^n$ in an incoherent domain $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$. It invokes optimization techniques to recover the sparse signal. This protocol essentially acquires the signal in compressed form and it uses an efficient decoder to decompress the signal; this is the role of ℓ_1-minimization. In order to reconstruct a signal $\boldsymbol{x}$ from the measurement $\boldsymbol{y}$, we need to solve the following linear algebraic system,

$$\boldsymbol{\Psi x} = \boldsymbol{y}. \tag{2.3.1}$$

An important characteristic of equation (2.3.1) is that the size of the observed signal $\boldsymbol{y}$ is much smaller than the size of original signal $\boldsymbol{x}$; the equation is under-determined. Linear algebraic theory asserts that it is impossible to find an exact solution from an under-determined system of linear equations. Accordingly, there are infinitely many solutions to equation (2.3.1). The solution has the form $\boldsymbol{x}^* = \boldsymbol{x} + \mathcal{N}(\boldsymbol{\Psi})$, where $\boldsymbol{x}$ satisfies equation (2.3.1) and $\mathcal{N}(\boldsymbol{\Psi})$ is the null space of $\boldsymbol{\Psi}$.

Since the original vector $\boldsymbol{x}$ is known to be sparse, the problem of finding the desired solution can be phrased as an optimization problem. The objective is to minimize an appropriate measure of sparsity while simultaneously satisfying the constraints defined by (2.3.1). This can be expressed mathematically as;

$$\boldsymbol{x}^* = \arg \min_{\boldsymbol{x}} F(\boldsymbol{x}) \quad \text{subject to } \boldsymbol{\Psi x} = \boldsymbol{y}, \tag{2.3.2}$$

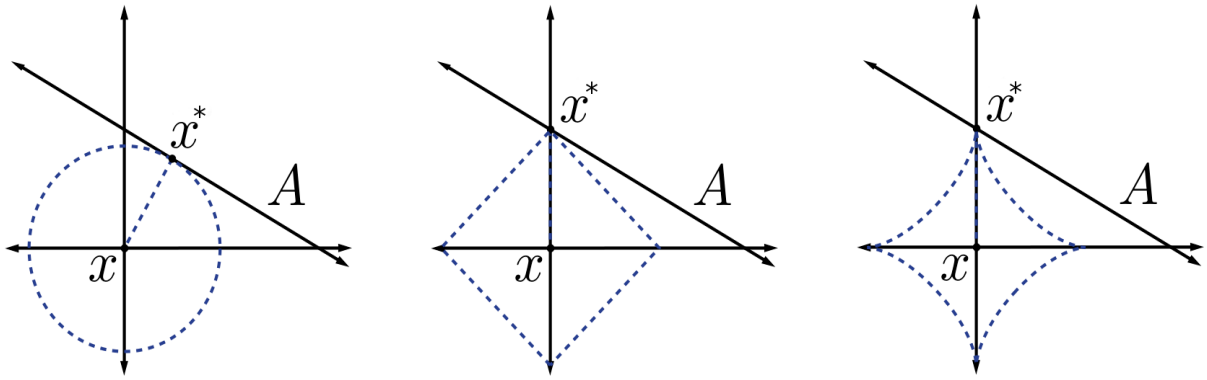
where $F : \mathbb{R}^n \to \mathbb{R}$ is an objective function and it should promote sparsity in the solution. We are interested in the function of the form

$$F(\boldsymbol{x}) := \|\boldsymbol{x}\|_p = \left(\sum_i^n |x_i|^p \right)^{\frac{1}{p}}, \tag{2.3.3}$$

where $p > 0$ and x_i is the i-th component of $\boldsymbol{x}$. Equation (2.3.3) expresses the ℓ_p-norm of the vector $\boldsymbol{x}$. Note that in the case where $p < 1$, the triangular inequality is not satisfied

and the norm lose one of its important properties, which is convexity. Thus ℓ_p-norm for $0 < p < 1$ is not actually a norm but rather a quasi-norm. Figure 2.3, shows the graphical representation of ℓ_p-norm for different values of p, imposed with a linear constraint.

Geometrically, the sparse vector should always lie near or along the coordinate axes (for $\mathbb{R}^2$ it must lie along the axes). The straight line represents all the points $\boldsymbol{x}$ that satisfy the constraint $\boldsymbol{\Psi x} = \boldsymbol{y}$. Due to randomness of $\boldsymbol{\Psi}$, the line would be oriented at a random angle. The dotted lines represent ℓ_p ball, $\mathcal{B}_p := \{\boldsymbol{x} : \|\boldsymbol{x}\|_p \leq r\}$ of radius r. We shall use this to motivate the use of ℓ_1-norm in compressed sensing.



(a) ℓ_2-norm (b) ℓ_1-norm (c) $\ell_{\frac{1}{2}}$-norm

Figure 2.3: Best approximation of a point in $\mathbb{R}^2$ by a one-dimensional subspace using the ℓ_p-norms for $p = 2, 1$ and the ℓ_p quasi-norm with $p = \frac{1}{2}$.

2.3.1 The ℓ_0-norm Minimization. Traditionally, the sparse approximation problem is solved through the ℓ_0-minimization approach. Given the observed measurements $\boldsymbol{y}$ and the knowledge that the original signal is sparse, it is possible to recover $\boldsymbol{x}$ by solving an optimization problem of the form,

$$\boldsymbol{x}^* = \arg\min \|\boldsymbol{x}\|_0 \qquad \text{subject to} \quad \boldsymbol{\Psi x} = \boldsymbol{y}, \tag{2.3.4}$$

where $\|\boldsymbol{x}\|_0$ is the number of elements in the support set $supp(\boldsymbol{x})$, that is $\|\boldsymbol{x}\|_0 = |supp(\boldsymbol{x})|$. Note $\|\cdot\|_0$ is not a norm by definition, rather a quasi-norm. Solving this problem is equivalent to selecting s columns of $\boldsymbol{\Psi}$ that best represent the observed vector $\boldsymbol{y}$. This requires a search through all n choose s possible subsets of columns of $\boldsymbol{\Psi}$. In principle, the strategy is very effective in finding the exact sparse solution.

Although it is possible to recover sparse solutions precisely, it is infeasible to solve such a combinatorial optimization problem. The problem is generally NP-Hard, Natarajan (1995). Algorithms that solves equation (2.3.4) are computationally intractable. In addition to this computational challenge, it has been shown that program (2.3.4) yields a solution which is not robust to noise, Mashud and Kaushik (2009).

2.3.2 The ℓ_2-norm Minimization. Since the ℓ_0-minimization problem in equation (2.3.4) is computationally infeasible to solve, an alternative approach is needed to solve the problem efficiently. A common avenue is to use the least square approach, based on the ℓ_2-minimization problem,

$$\boldsymbol{x}^* = \arg\min \|\boldsymbol{x}\|_2 \qquad \text{subject to} \quad \boldsymbol{\Psi x} = \boldsymbol{y}. \tag{2.3.5}$$

Equation (2.3.5) can be solved through Tikhnov regularization, Neumaier (1998),

$$\min_{\boldsymbol{x}} \quad \frac{1}{2}\|\boldsymbol{\Psi x} - \boldsymbol{y}\|_2^2 + \mu\|\boldsymbol{x}\|_2^2, \tag{2.3.6}$$

where $\mu > 0$ is the penalty parameter. There exists an exact analytical solution of equation (2.3.6) given by

$$\boldsymbol{x}^* = \boldsymbol{\Psi}^T \left(\boldsymbol{\Psi}\boldsymbol{\Psi}^T + \mu \boldsymbol{I}_m\right)^{-1} \boldsymbol{y}, \tag{2.3.7}$$

where $\boldsymbol{I}_m$ is identity matrix of size $m \times m$. In most cases, the least square approach produces poor results; the solution vector is generally not sparse.

To demonstrate the effect of ℓ_2-norm on the solution, we consider the example in Figure 2.3a. The convexity of ℓ_2-norm implies that a unique solution to equation (2.3.5) exists. However, the solution is generally not sparse. The ℓ_2 minimizer of problem (2.3.5) is a point on the constraint line closest to the origin. This point can be found by blowing the $\mathcal{B}_2$ ball until it contacts the constraint line $\boldsymbol{\Psi x} = \boldsymbol{y}$. Accordingly, the optimal point $\boldsymbol{x}^*$ will always be situated away from the coordinate axes. Thus, it is clear that the ℓ_2-norm would always return a unique non-sparse solution. This concept also extends to high dimensional vectors. Accordingly, the ℓ_2-norm is not appropriate for finding s-sparse solution. We seek a sparse promoting norm.

2.3.3 The ℓ_1-norm Minimization. Candes et al. (2006a) provide mathematical foundations that show if a measurement matrix satisfies certain quantitative properties (such as incoherence and the RIP), problem (2.3.4) can be transformed into something computationally tractable. This can be successfully done by replacing the non-convex function $\|\cdot\|_0$ with its closest convex approximate function $\|\cdot\|_1$. The equivalent convex optimization problem is,

$$\boldsymbol{x}^* = \arg\min \|\boldsymbol{x}\|_1 \qquad \text{subject to} \quad \boldsymbol{\Psi x} = \boldsymbol{y}. \tag{2.3.8}$$

This problem is known as the *Basis Pursuit* in literature. The value of $p = 1$ is the smallest p value for which ℓ_p-norm is convex. Due to its convexity, basis pursuit is numerically feasible to solve. Most importantly, ℓ_1-minimization is greatly utilized because it provides improved sparsity relative to least square solution. To illustrate how well basis pursuit promotes sparsity we consider Figure 2.3b. Blowing the $\mathcal{B}_1$ ball as before, the first point of contact $\boldsymbol{x}^*$ with the constraint line $\boldsymbol{\Psi x} = \boldsymbol{y}$ coincides with the exact minimizer and it is aligned with the coordinate axes. This will always be the case for a unique sparse solution vector. Accordingly, ℓ_1-norm promotes sparsity. Depending on the orientation of $\boldsymbol{\Psi x} = \boldsymbol{y}$, the constrained line and the side of $\mathcal{B}_1$ ball may have the same gradient. In such instances one have infinitely many solutions and most of which are not sparse. We will ignore such case.

There exists theoretical guarantees with profound results on the effectiveness of basis pursuit. Candes and Tao (2005) showed that for any measurement matrix that satisfy the restricted isometric property, basis pursuit (2.3.8) recovers all sparse signals exactly. This is illustrated in the following theorem:

Theorem 2.3 (Exact recovery of basis pursuit, Candes and Tao (2005))**.** *Assume that the measurement matrix $\boldsymbol{\Psi}$ satisfies the restricted isometric property. Then every s-sparse vector $\boldsymbol{x}$ can be exactly recovered from the measurements $\boldsymbol{y} = \boldsymbol{\Psi x}$, as a unique solution to the optimization problem (2.3.8).*

The above guarantee is uniform. That is, once the measurement matrix $\boldsymbol{\Psi}$ satisfies the restricted isometric property, basis pursuit correctly recovers all sparse vectors.

2.3.4 The ℓ_p-norm Minimization for $0 < p < 1$. The use of ℓ_1-norm works well due to its geometry. One might think of using ℓ_p-norm for some $0 < p < 1$. Indeed, the geometry of such quasi-norm would return even better sparsity, see Figure 2.3c. Ge et al. (2011) used this approach compressive sensing. However, recovery methods using such approach have not yet provided rigorous optimal results. The program (2.3.8) has an advantage over those with $0 < p < 1$ because of convexity.

2.4 Phase Transition Analysis

The performance of a compressive sensing algorithm can be evaluated via phase transition analysis. This analysis was first introduced by Donoho and Tanner (2009a) as a means of studying the equivalence of ℓ_1 and ℓ_0. The equivalence is an indication of how sparsity and the number of measurements affect the success of basis pursuit. The technique has been adopted by other researchers in compressive sensing community as a means of comparing reconstruction ability of different algorithms, Donoho et al. (2006).

The description of phase transition analysis is as follow. Let $\delta = \frac{m}{n}$, represent a normalized measure of indeterminacy factor and $\rho = \frac{s}{m}$, represent a normalized measure of sparsity. Phase plot depicts some performance statistics as a function of δ and ρ, over a phase spaced $(\delta, \rho) \in [0, 1] \times [0, 1]$. Each pair (δ, ρ) in the space indicates the difficulty of reconstruction. For example, a pair with small δ and large ρ represents harder reconstruction. The challenge is that a problem has large number of non-zero components with few measurements. Theoretical analysis and empirical results have shown that there are two regions in the phase space, Donoho and Tanner (2009a); Donoho et al. (2006). A success region, where the probability of precise reconstruction of the sparse signal tends to one and a failure region, where the probability of precise reconstruction tends to zero. The phenomenon of moving from one region to the other is referred to as phase transition. The boundary where the two regions intercept is called the phase transition line, Donoho and Tanner (2009a), see Figure 2.4. The higher the phase transition line, the better the performance of the reconstruction algorithm. Only the phase transition line of ℓ_1-minimization problem has been rigorously

proven. The transition line has been calculated from computational geometry by Donoho and Tanner (2009a,b, 2005), and also empirically observed by Donoho et al. (2006); Berinde et al. (2008).

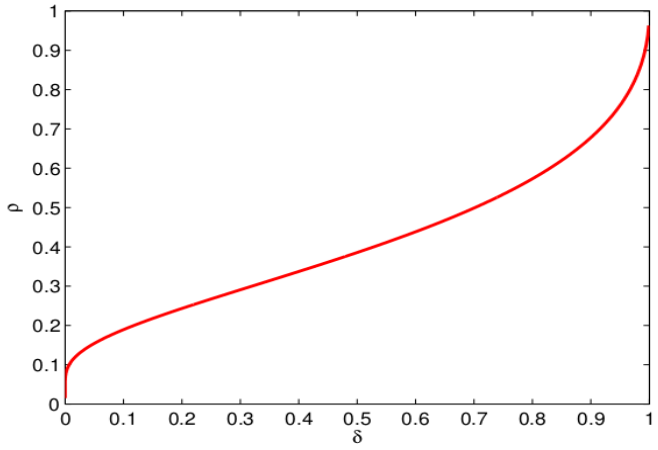


Figure 2.4: Phase transition diagram.

2.5 Optimality for Non-differentiable Functions

In this section, important concepts regarding optimality condition of non-differential convex optimization problems are discussed. Consider an optimization problem of the form

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := g(\boldsymbol{x}) + f(\boldsymbol{x}), \tag{2.5.1}$$

where $F : \mathbb{R}^n \to \mathbb{R}$ is a composite multivariate real-valued function with a convex differentiable term $f : \mathbb{R}^n \to \mathbb{R}$ and a convex non-differentiable term $g : \mathbb{R}^n \to \mathbb{R}$. The presence of the non-differentiable term makes the problem difficult to solve in a sense that conventional optimization concepts do not work. The compressive sensing problem is cast as a constrained optimization problem. Through regularization it can be easily expressed as unconstrained optimization problem of the form (2.5.1). Hence, we will present some useful concepts relating to non-differentiable convex unconstrained optimization.

2.5.1 Gradient and Subgradient. Gradient often relates to differentiable functions and sub-gradient relates to non-differentiable functions. Below, we to discuss a subgradient since the overall objective function in our study is non-differentiable.

Definition 2.3 (Subdifferential, Neal and Stephen (2013)). *Let $F : \mathbb{R}^n \to \mathbb{R}$ be a convex function and a vector $\boldsymbol{x} \in \mathbb{R}^n$ be given. Then the subdifferential of F at $\boldsymbol{x}$ is defined as:*

$$\partial F(\boldsymbol{x}) := \{\boldsymbol{z} \in \mathbb{R}^n : F(\boldsymbol{x}) + \boldsymbol{z}^T(\boldsymbol{x}' - \boldsymbol{x}) \leq F(\boldsymbol{x}') \text{ for all vectors } \boldsymbol{x}' \in \mathbb{R}^n\}. \tag{2.5.2}$$

The elements of $\partial F(\boldsymbol{x})$ are called subgradients of F at $\boldsymbol{x}$.

Definition 2.3 actually states that any subgradient $\boldsymbol{z}$ in $\partial F(\boldsymbol{x})$ defines an affine function $\boldsymbol{x}' \to F(\boldsymbol{x}) + \boldsymbol{z}^T(\boldsymbol{x}' - \boldsymbol{x})$ which is tangent to the graph of the function F. Moreover, there is a bijection between such tangent affine functions and the subgradient. The following proposition illustrates the usefulness of the subdifferential in the study of non-differentiable optimization problems.

Proposition 2.1 (First order optimality of non-differentiable function, Neal and Stephen (2013)). *For any non-differentiable convex function $F : \mathbb{R}^n \to \mathbb{R}$, a point $\boldsymbol{x} \in \mathbb{R}^n$ is a global minimization of F if and only if the condition $\mathbf{0} \in \partial F(\boldsymbol{x})$ holds.*

The concept of a subdifferential is mainly useful for non-differentiable functions. If F is differentiable at $\boldsymbol{x}$, then the set $\partial F(\boldsymbol{x})$ would be a singleton $\{\nabla F(\boldsymbol{x})\}$. Moreover the optimality condition of **Proposition 2.1** reduces to the classical first-order optimality condition $\nabla F(\boldsymbol{x}) = \mathbf{0}$. The above concepts are used in the following section to derive the shrinkage operator. Shrinkage operator plays an important role in finding the solution of ℓ_1-norm minimization.

2.5.2 Proximal Operator. Analogous to Newtons methods, which are standard tools for solving unconstrained differentiable optimization problems, proximal methods are standard tools to address non-differentiable optimization problems. The basic operation of proximal techniques is evaluating the *proximal operator* of a given function. **Definition 2.4** below describes the proximal operator.

Definition 2.4 (Proximal Operator, Neal and Stephen (2013)). *The scaled proximal operator* $\boldsymbol{prox}_{\mu f} : \mathbb{R}^n \to \mathbb{R}^n$ *of a convex function* $f : \mathbb{R}^n \to \mathbb{R}$ *at the proximal point* $\boldsymbol{v} \in \mathbb{R}^n$ *is defined as the minimization problem*

$$\boldsymbol{prox}_{\mu f}(\boldsymbol{v}) := \ arg \ \min_{\boldsymbol{x}} \left\{ f(\boldsymbol{x}) + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2 \right\}, \tag{2.5.3}$$

where $\mu > 0$ *is the scaling factor.*

There is a close connection between the proximal operator and fixed-point theory. In particular, the fixed-point of the proximal operator of f is precisely the minimizers of f. In other words, if $\boldsymbol{x}^*$ is the optimal point then the following statement is true:

$$\boldsymbol{x}^* = \boldsymbol{prox}_{\mu f}(\boldsymbol{x}^*) \quad \text{if and only if} \quad \boldsymbol{x}^* = \arg \min_{\boldsymbol{x}} f(\boldsymbol{x}). \tag{2.5.4}$$

Below, we prove statement (2.5.4) and also demonstrate the connection of the proximal operator with subdifferential. Let $\boldsymbol{prox}_{\mu f}(\boldsymbol{v})$ be given as (2.5.3). Then, with the knowledge that $f(\boldsymbol{x})$ is not differentiable, one can evoke the first order optimality condition to find the optimum of the proximal minimization problem (2.5.3) as:

$$\mathbf{0} \in \partial \left(f(\boldsymbol{x}) + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2 \right) = \partial f(\boldsymbol{x}) + \frac{1}{\mu}(\boldsymbol{x} - \boldsymbol{v}). \tag{2.5.5}$$

Here, $\partial f(\boldsymbol{x})$ is the subdifferential of f at $\boldsymbol{x}$, and is defined as in **Definition 2.3**. Setting $\boldsymbol{x} = \boldsymbol{v} = \boldsymbol{x}^*$, it follows that $\mathbf{0} \in \partial f(\boldsymbol{x}^*)$ and $\boldsymbol{x}^*$ minimizes f.

To argue the converse, let $\boldsymbol{x}^*$ be the minimizer of f, then $f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)$ for all $\boldsymbol{x}$ in the domain of f. Thus, for any $\boldsymbol{x}$, the following is true

$$f(\boldsymbol{x}) + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{x}^*\|_2^2 \geq f(\boldsymbol{x}^*) = f(\boldsymbol{x}^*) + \frac{1}{2\mu}\|\boldsymbol{x}^* - \boldsymbol{x}^*\|_2^2. \tag{2.5.6}$$

Therefore $\boldsymbol{x}^*$ minimizes the function $f(\boldsymbol{x}) + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{x}^*\|_2^2$. It follows that $\boldsymbol{x}^* = \boldsymbol{prox}_{\mu f}(\boldsymbol{x}^*)$. Hence, this suggests that the proximal operator is a very useful tool for solving non-differential optimization problems. In fact, most convex optimization algorithms for solving ℓ_1-minimization problems involve the evaluation of the proximal operator.

This work is concerned with the proximal operator in which the function $f(\boldsymbol{x})$ is defined as ℓ_1-norm, that is $f(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$. Below, we present another related operator which will be

useful in subsequent chapters. Substitute $f(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ into the proximal operator (2.5.3) and solve the resulting minimization problem,

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := \|\boldsymbol{x}\|_1 + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2. \tag{2.5.7}$$

We first characterise the subdifferential of the convex function $\|\boldsymbol{x}\|_1$ as

$$(\partial\|\boldsymbol{x}\|_1)_i = \begin{cases} \{1\} & \text{if } x_i > 0, \\ [-1, 1] & \text{if } x_i = 0, \\ \{-1\} & \text{if } x_i < 0. \end{cases} \tag{2.5.8}$$

Using (2.5.8) the sub-differential of problem (2.5.7), is given componentwise as

$$\begin{aligned} \partial F(\boldsymbol{x}) &= \partial\left(\|\boldsymbol{x}\|_1 + \frac{1}{2\mu}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2\right), \\ &= \partial\|\boldsymbol{x}\|_1 + \frac{1}{\mu}(\boldsymbol{x} - \boldsymbol{v}), \\ &= \begin{cases} \{x_i - v_i + \mu\}/\mu & \text{if } x_i > 0, \\ [-\mu - v_i, \mu - v_i]/\mu & \text{if } x_i = 0, \\ \{x_i - v_i - \mu\}/\mu & \text{if } x_i < 0. \end{cases} \quad \text{for all } i = 1, 2, \ldots, n, \end{aligned} \tag{2.5.9}$$

Let X^* represent a set of all optimal solutions of problem (2.5.7). The first order optimality condition for non-differentiable function asserts that $\boldsymbol{x} \in X^* \iff \boldsymbol{0} \in \partial F(\boldsymbol{x})$. Therefore, we have the following cases for which the subdifferential contains zero:

- if $v_i > +\mu$, the sub-differential is zero is when $x_i = v_i - \mu > 0$.
- if $v_i \in [-\mu, \mu]$, the sub-differential is zero at $x_i = 0$.
- if $v_i < -\mu$, the sub-differential is zero is when $x_i = v_i + \mu < 0$.

Hence, the i-th component of the minimizer $\boldsymbol{x}^*$ of problem (2.5.7) is

$$x_i^* = \begin{cases} v_i - \mu & \text{if } v_i > \mu, \\ 0 & \text{if } v_i \in [-\mu, \mu], \\ v_i + \mu & \text{if } v_i < -\mu. \end{cases} \tag{2.5.10}$$

From equation (2.5.10), we define the *shrinkage operator* as

$$\mathcal{S}(v,\mu) := sign(v) \times \max\{|v| - \mu, 0\}, \quad \text{where} \quad sign(v) = \begin{cases} 1 & \text{if } v > 0, \\ 0 & \text{if } v = 0, \\ -1 & \text{if } v < 0. \end{cases} \tag{2.5.11}$$

Lastly, the solution of the proximal ℓ_1-minimization problem (2.5.3) is

$$\boldsymbol{prox}_{\mu\|\boldsymbol{x}\|_1}(\boldsymbol{v}) = \mathcal{S}(\boldsymbol{v},\mu). \tag{2.5.12}$$

This computation is done componentwise. It turns out that in order to solve the ℓ_1-minimization problem, one needs to transform the problem to a form similar to proximal operator (2.5.3). Then an approximate solution can be obtain via equation (2.5.12). In **Chapter 3** we present algorithm for solving the ℓ_1-minimization problem. The above theory plays an important role.

Chapter 3

Algorithms for Compressive Sensing

This chapter discusses a number of algorithmic approaches that address the compressive sensing problem. Algorithms for solving a compressive sensing problem are broadly divided into at least five major categories, as illustrated in Figure 3.1. A brief description of what each category entails is as follows:

1. *Convex relaxation* - replaces the ℓ_1-norm problem with a convex optimization problem, and then solves the convex problem with algorithms that exploit problem structure.

2. *Greedy pursuit* - iteratively refines a sparse solution by successively identifying one or more components that yield the greatest improvement in quality.

3. *Iterative thresholding* - can be thought of as an extension of the classical gradient approach to deal with non-differentiable objective functions.

4. *Combinatorial algorithms* - search through all possible support sets, possibly using cutting-plane methods to reduce the number of possibilities. However, most algorithms here employ the group testing technique.

5. *Bayesian (Nonconvex optimization)* - relaxes the ℓ_0 problem to a related nonconvex problem and attempts to identify an optimal solution.

An elaboration of individual algorithm is provided on subsequent sections of this chapter. While it is possible to describe as many as possible algorithms here, we restrict the discussion



Figure 3.1: Classification of compressed sensing algorithms.

to the original algorithmic ideas of a specific technique and explain their advancement thereof. The Digital and Signal Processing group at RICE university keeps an up to date repository of the latest articles in compressed sensing.

The understanding of how these different algorithms perform is of paramount interest in this study. We identify and implement all the necessary improvements in the design of effective and efficient hybrid algorithm. The design of sparse recovery algorithms is guided by various criteria that quantify its efficiency and effectiveness. Some important criteria in the development and analysis of compressive sensing algorithms are defined below, Needell and Tropp (2009):

- *Speed*: Most applications in compressive sensing deal with large scale problems. Hence, a good algorithm should strive to efficiently recover an optimal signal using minimal computational resources.
- *Performance guarantees*: A range of geometric conditions that aid in preserving signal information were fully discussed in the previous chapter. Thus, a good algorithm should satisfy the restricted isomeric property and the incoherence condition.
- *Robustness to noise*: Real world signals are not exactly sparse and involve uncertainty (for example perturbation of input signal or contamination by unstructured noise). A good algorithm should provide stable and consistent reconstruction of real sparse signals.

These criteria are important in understanding and comparing computational performance of compressive sensing algorithms. In addition to a detailed description of how a particular algorithm moves from one iteration to the next, we also state its main results. Main results are specified in terms of one or more of the above mentioned criteria.

3.1 Convex Relaxation Algorithms

In a general sense, convex relaxation algorithms solve the ℓ_1-minimization problem. A fundamental approach is to replace the ℓ_0-minimization combinatorial problem with a convex optimization problem. Convex relaxation algorithms solve the convex program by exploiting the problem structure. The algorithms are an extension of classical optimization techniques adapted to deal with the non-differentiability of the objective function. Naturally, the

ℓ_1-norm is the closest convex function that promotes sparsity, see **section 2.3** for the full motivation. We consider the convex program, *Basis Pursuit*:

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1, \quad \text{subject to} \quad \boldsymbol{y} = \boldsymbol{\Psi}\boldsymbol{x}. \tag{3.1.1}$$

In the case of noiseless scenarios, Basis Pursuit can be solved exactly provided that mutual coherence is small, see **Theorem 2.1**.

In any practical system, one cannot discard the presence of noise. Therefore, compressive sensing should be able to deal with noisy measurements. In such cases, the measurements are modelled as

$$\boldsymbol{y} = \boldsymbol{\Psi}\boldsymbol{x} + \boldsymbol{e}, \tag{3.1.2}$$

where $\boldsymbol{e}$ is a noise vector and we assume that noise is bounded by a positive constant ε, i.e $\|\boldsymbol{e}\|_2 \leq \varepsilon$. In the presence of noise, finding a sparse solution vector $\boldsymbol{x}$ of (3.1.2) can be approached via several equivalent formulations. Basis Pursuit is modified and expressed as *Basis Pursuit Denoising*:

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1, \quad \text{subject to } \|\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{x}\|_2 \leq \varepsilon_1, \tag{3.1.3}$$

where ε_1 quantifies the uncertainty about the measurement $\boldsymbol{y}$. This formulation is often preferred because a reasonable estimate of ε_1 may be known. Alternatively, problem (3.1.3) can be recast in its Lagrangian form as

$$\min_{\boldsymbol{x}} \mu\|\boldsymbol{x}\|_1 + \frac{1}{2}\|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2^2, \tag{3.1.4}$$

where $\mu > 0$ is the penalty parameter controlling the trade-off between accuracy of sparsity and speed of convergence. This approach is frequently discussed in literature due to its unconstrained nature.

In statistics, Tibshirani (1996) proposed the least absolute shrinkage and selector operator (LASSO). It minimizes the residual sum of squared error, under the constraint of ℓ_1-norm on the solution vector. Mathematically, the problem is stated as

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2^2, \quad \text{subject to} \quad \|\boldsymbol{x}\|_1 \leq \varepsilon_2. \tag{3.1.5}$$

Standard optimization theory asserts that the three problems (3.1.3), (3.1.4) and (3.1.5) are equivalent for specific choices of parameters $\varepsilon_1, \mu, \varepsilon_2$. Note, the theoretical equivalence does not necessary mean all these three problems are just as easy to solve. Generally, the constrained optimization problems are harder to solve than unconstrained problems. In literature, problem (3.1.4) is preferred over the other two formulations.

Solving one of the above mentioned noisy sparse approximation formulations will recover the exact solution with an error proportional to noise. The following theorem by Candes et al. (2006b) connects these reconstruction approaches with RIP. Results of the theorem serve as a benchmark of robustness to noise for convex relaxation algorithms.

Theorem 3.1 (Reconstruction from noisy measurements, Candes et al. (2006b))**.** *Assume that the measurement matrix* $\boldsymbol{\Psi}$ *satisfies the restricted isometric condition with isometric constant* $\delta_s = \sqrt{2} - 1$*. Then a sparse approximate solution* $\boldsymbol{x}^*$ *of problem (3.1.3), obeys the error bound*

$$\|\boldsymbol{x}^* - \boldsymbol{x}\|_1 \leq C_1 \cdot \varepsilon_1 + C_2 \cdot \frac{\|\boldsymbol{x} - \boldsymbol{x}_s\|_1}{s}, \tag{3.1.6}$$

where $\boldsymbol{x}$ *is a true solution and* $\boldsymbol{x}_s$ *is a vector* $\boldsymbol{x}$ *with all but* s *largest components set to zero, and* C_1 *and* C_2 *are constants.*

Problems (3.1.3), (3.1.4) and (3.1.5) pose two key challenges which relate to specific practical and theoretical encounters with signals. Firstly, real world problems are invariably large scale which lead to optimization involving millions of variables. This is well beyond any standard optimization software packages. Secondly, the objective function and/or constraints are not smooth. Standard smoothing techniques do not produce good results, Bobin and Becker (2009). Thus, the conventional optimization algorithms are not effective or even practical. These unique challenges encountered in the context of compressive sensing have led to a considerable development of sparse recovery algorithms within the optimization community. We discuss some of the popular algorithms, with a particular focus on how they deal with the non-smoothness of objective functions.

3.1.1 Fixed Point Continuation Active Set. The *Fixed Point Continuation Active Set* (FPC-AS) was proposed by Wen et al. (2010) as an efficient algorithm for solving problems of the form (3.1.4). The algorithm is divided into two stages that are performed iteratively. In the first stage a first-order iterative shrinkage scheme is employed. Its aim is to quickly estimate a subset of components of $\boldsymbol{x}$ that are likely to be non-zero in the optimal solution. In the second stage a second-order limited-memory quasi-Newton method is used to solve a smooth version of (3.1.4). The smooth quadratic function is formed by restricting the problem to indices of non-zero components of $\boldsymbol{x}$ computed in the first stage of the algorithm. These two stages are embedded in a continuation approach. Continuation strategy solves the problem for decreasing sequence of μ (detailed discussion of continuation is given later). The continuation strategy has been empirically shown to significantly improve the convergence of the algorithm, Hale et al. (2008).

FPC-AS is an improvement of *Fixed Point Continuation* (FPC) algorithm developed by Hale et al. (2008). FPC is used as a shrinkage scheme in the first stage of the two-stage FPC-AS algorithm. FPC is based on the ideas of *operator-splitting* from Combettes et al. (2005), and the continuation strategy. Consider a general ℓ_1-minimization problem of the form

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mu\|\boldsymbol{x}\|_1 + f(\boldsymbol{x}), \tag{3.1.7}$$

where $\mu > 0$ and $f : \mathbb{R}^n \to \mathbb{R}$ is a real valued function which is assumed to be continuously differentiable and convex. This function is essentially $f(\boldsymbol{x}) = \|\boldsymbol{\Psi x} - \boldsymbol{y}\|_2^2$ in the compressive sensing settings. The main idea is based on the fixed point theory. The fixed point iteration is derived from subdifferential optimality condition. It is basically a composition of shrinkage and a gradient descent step.

To derive the fixed point iteration, let us write the objective function of problem (3.1.7) as a sum of two functions in form $F(\boldsymbol{x}) = g(\boldsymbol{x}) + f(\boldsymbol{x})$, where the function $g(\boldsymbol{x})$ represents a convex non-differentiable term, while $f(\boldsymbol{x})$ is a convex continuously differentiable function. A point $\boldsymbol{x}$ in the optimal set X^* minimizes $F(\boldsymbol{x})$ if and only if

$$\mathbf{0} \in \partial F(\boldsymbol{x}^*) = \partial g(\boldsymbol{x}^*) + \nabla f(\boldsymbol{x}^*). \tag{3.1.8}$$

Here $\partial g(\boldsymbol{x})$ is the subdifferential of g at $\boldsymbol{x}$ and $\nabla f(\boldsymbol{x})$ is the gradient of f at $\boldsymbol{x}$. Let us

multiply (3.1.8) by a positive parameter α, the reason for this will become clear later. The optimality condition is still true if and only if the following equivalent statements are true,

$$\begin{aligned}
\mathbf{0} &\in \alpha\partial g(\boldsymbol{x}) + \alpha\nabla f(\boldsymbol{x}) \\
\mathbf{0} &\in (\alpha\partial g(\boldsymbol{x}) + \boldsymbol{x}) + (\alpha\nabla f(\boldsymbol{x}) - \boldsymbol{x}) \\
\mathbf{0} &\in (\boldsymbol{I} + \alpha\partial g)(\boldsymbol{x}) - (\boldsymbol{I} - \alpha\nabla f)(\boldsymbol{x}) \\
(\boldsymbol{I} - \alpha\nabla f)(\boldsymbol{x}) &\in (\boldsymbol{I} + \alpha\partial g)(\boldsymbol{x}) \\
\boldsymbol{x} &= (\boldsymbol{I} + \alpha\partial g)^{-1}(\boldsymbol{I} - \alpha\nabla f)(\boldsymbol{x}),
\end{aligned} \tag{3.1.9}$$

Combettes et al. (2005). Here $\boldsymbol{I}$ is the identity operator and $(\cdot)^{-1}$ is the inverse operator. Equation (3.1.9) implies that $\boldsymbol{x}$ minimizes F if and only if it is a fixed point of the *forward-backward operator*:

$$(\boldsymbol{I} + \alpha\partial g)^{-1}(\boldsymbol{I} - \alpha\nabla f), \tag{3.1.10}$$

Combettes et al. (2005). We can repeatedly apply this operator to obtain a fixed point and thus find a solution of problem (3.1.7). However, we first need to evaluate $(\boldsymbol{I} + \alpha\partial g)^{-1}$. In fact the expression is equivalent to the proximal operator. To prove this assertion consider the following argument. By definition, if $\boldsymbol{x} \in (\boldsymbol{I} + \alpha\partial g)^{-1}(\boldsymbol{v})$, then

$$\boldsymbol{v} \in (\boldsymbol{I} + \alpha\partial g)(\boldsymbol{x}) = \boldsymbol{x} + \alpha\partial g(\boldsymbol{x}). \tag{3.1.11}$$

This equation can be expressed as

$$\mathbf{0} \in \partial g(\boldsymbol{x}) + \frac{1}{\alpha}(\boldsymbol{x} - \boldsymbol{v}), \tag{3.1.12}$$

which is equivalent to

$$\mathbf{0} \in \partial\left(g(\boldsymbol{x}) + \frac{1}{2\alpha}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2\right), \tag{3.1.13}$$

where the subdifferential is with respect to $\boldsymbol{x}$. As in **section 2.5**, this is the necessary and sufficient condition for $\boldsymbol{x}$ to minimize a convex function which is defined by the proximal operator

$$\boldsymbol{prox}_{\alpha g}(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}} g(\boldsymbol{x}) + \frac{1}{2\alpha}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2. \tag{3.1.14}$$

Hence, we connect everything above to ℓ_1-minimization problem (3.1.7). Since $g(\boldsymbol{x}) = \mu\|\boldsymbol{x}\|_1$, it follows from the proximal operator (3.1.14) that the explicit solution of problem (3.1.7) is given by the shrinkage operator, see **section 2.5**. Thus, the k-th iterate of a fixed point iteration is

$$\begin{aligned} \boldsymbol{x}^k &= (\boldsymbol{I} + \alpha\partial g)^{-1} (\boldsymbol{I} - \alpha\nabla f) (\boldsymbol{x}^{k-1}) \\ &= \mathcal{S}\left(\boldsymbol{x}^{k-1} - \alpha\nabla f(\boldsymbol{x}^{k-1}), \alpha\mu\right). \end{aligned} \tag{3.1.15}$$

The fixed point iteration is basically a composition of two mappings. The first mapping, $\boldsymbol{x}^{k-1} - \alpha\nabla f(\boldsymbol{x}^{k-1})$, resembles the gradient descent step of a differentiable term f, with the step size α. Thus, α was introduced early for this purpose. The second mapping is the shrinkage step with $\mathcal{S}$. It reduces or discards the magnitude of each component of $\boldsymbol{x}^{k-1} - \alpha\nabla f(\boldsymbol{x}^{k-1})$ by an amount less than or equal to $\alpha\mu$, thus deducing the sparsity.

The convergence of fixed point algorithm is depended upon two parameters α and μ. Hale et al. (2008) implemented the fixed point iteration with a fixed stepsize α, with the requirement that

$$0 < \alpha < \frac{2}{\lambda_{max}}, \tag{3.1.16}$$

where $\lambda_{\max}$ is the largest eigenvalue of the Hessian $\nabla^2 f$. Since a bad choice of α usually lead to a slow rate of convergence, one can choose the stepsize in a dynamic way. One can employ line search to guarantee global convergence. Furthermore, since practical problems are normally large scale, it may be computationally expensive to find eigenvalues of the Hessian. An efficient strategy for dynamic updating of α^k was developed by Barzilai and Borwein (1988). The strategy requires few storage and very inexpensive computations. The Barzilai-Borwein strategy defines α^k in a manner that it mimics the quasi-Newton method, specifically

$$\alpha_{BB}^k = \frac{\left(\Delta\boldsymbol{x}^k\right)^T \Delta\boldsymbol{x}^k}{\left(\Delta\boldsymbol{x}^k\right)^T \Delta\boldsymbol{f}^k}, \quad \text{or} \quad \alpha_{BB}^k = \frac{\left(\Delta\boldsymbol{x}^k\right)^T \Delta\boldsymbol{f}^k}{\left(\Delta\boldsymbol{f}^k\right)^T \Delta\boldsymbol{f}^k}, \tag{3.1.17}$$

where

$$\Delta\boldsymbol{x}^k = \boldsymbol{x}^k - \boldsymbol{x}^{k-1}, \quad \text{and} \quad \Delta\boldsymbol{f}^k = \nabla f(\boldsymbol{x}^k) - \nabla f(\boldsymbol{x}^{k-1}). \tag{3.1.18}$$

To avoid the parameter α^k being either too small or too large, one can choose the step size based on the following restrictions

$$\alpha^k = \max\left\{\alpha_{min}, \min\left\{\alpha_{BB}^k, \alpha_{max}\right\}\right\}, \tag{3.1.19}$$

where $0 < \alpha_{min} \leq \alpha_{max} < \infty$ are predetermined fixed constants. This step is useful in most of convex optimization algorithm in general.

The second parameter that affects convergence of fixed point iteration is μ. To enhance convergence through μ a strategy termed *continuation* is incorporated. In a continuation procedure instead of solving problem (3.1.7) for fixed μ one solves a sequence of problems

$$\boldsymbol{x}_{\mu^k}^* := \arg\min_{\boldsymbol{x}} F_{\mu^k}(\boldsymbol{x}) := \mu^k \|\boldsymbol{x}\|_1 + f(\boldsymbol{x}), \tag{3.1.20}$$

where $\mu^0 > \mu^1 > \mu^2, > \cdots, > \mu$. The solution $\boldsymbol{x}_{\mu^{k-1}}^*$ is used as the initial estimate solution of the problem (3.1.20) with μ^k. Hale et al. (2008) have empirically shown that using a shrinkage scheme to obtain $\boldsymbol{x}_{\mu^k}^*$ in a continuation strategy is superior to applying a basic shrinkage scheme to (3.1.7) directly. Experiments by Wright et al. (2009) have further confirmed the effectiveness of continuation. A pseudo code of fixed point continuation algorithm is presented in **Algorithm 1**.

Algorithm 1 Fixed Point Continuation Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$, measurement vector $\boldsymbol{y}$, and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set the initial solution $\boldsymbol{x}^0$. Select $\mu^0 > \mu^1 > \mu^2, > \cdots, > \mu^K = \mu$. Set all other parameters which are algorithm dependent.
2: *for* $k = 1, 2, \ldots, K$ *do*:
3: Set $\mu = \mu^k$ and $l = 0$.
4: *While* halting condition is not met *do*:
5: $l = l + 1$
6: Compute the stepsize α^l via (3.1.19).
7: Update solution $\boldsymbol{x}^l$ by solving (3.1.20) via (3.1.15).
8: *endWhile*
9: Set $\boldsymbol{x}^k = \boldsymbol{x}^l$.
10: *endFor*

The FPC-AS is developed from the convergence properties of FPC. Although the shrinkage scheme in FPC is very easy to compute, it takes large number of iterations to achieve an acceptable level of accuracy. The remarkable observation is that the shrinkage yields the iterate $\boldsymbol{x}^k$ with the same support and signs as those of the optimal solution $\boldsymbol{x}^*$ for all k greater than a finite number of iterations under mild conditions. Consider the following theorem by Hale et al. (2008), it states the global convergence of FPC.

Theorem 3.2. *Let f be a convex function that is bounded below, and let α satisfy*

$$0 < \alpha < \frac{2}{\lambda_{max}}, \tag{3.1.21}$$

where λ_{max} is the maximum eigenvalue of Hessian $\nabla^2 f$. The sequence $\{\boldsymbol{x}^k\}$ generated by the fixed-point iterations (3.1.15) from any starting point $\boldsymbol{x}^0$, converges to optimal point $\boldsymbol{x}^* \in X^*$. In addition the following statements holds true for all but finitely many iterations

$$x_i^k = x_i^*, \quad \text{for all } i \in S^c,$$
$$sign\left(\left[\boldsymbol{x}^k - \alpha \nabla f(\boldsymbol{x}^k)\right]_i\right) = sign\left(\left[\boldsymbol{x}^* - \alpha \nabla f(\boldsymbol{x}^*)\right]_i\right) \quad \text{for all } i \in S,$$

where $S = supp(\boldsymbol{x})$ is the support set of $\boldsymbol{x}$ and S^c is complimentary of the support set.

In practice the above results mean that after a moderate number of iterations $S^c(\boldsymbol{x}^*) \subseteq S^c(\boldsymbol{x}^k)$ holds, but it often takes many more iterations before the algorithm reaches the optimal support set. In other words, shrinkage is very effective in obtaining a support superset, but it is not efficient in recovering signal values. Therefore, FPC-AS utilises the first property of shrinkage scheme and overcomes the second property.

The second stage of FPC-AS find the optimal values of the sparse solution by solving a smooth subspace optimization problem. Once a good approximation of the support set S is found in the first stage a subspace optimization problem is defined as

$$\min_{\boldsymbol{x}} F_\mu(\boldsymbol{x}) := \mu \times sign(\boldsymbol{x}^k_{|S})^T \boldsymbol{x}^k_{|S} + f(\boldsymbol{x}). \tag{3.1.22}$$

Here $\boldsymbol{x}_{|S}$ represents a vector that keeps the components corresponding to indices of the set S while setting all other components to zero. This problem can be easily reformulated as a constrained problem. Wen et al. (2010) used a limited-memory quasi-Newton method based on the work by Zhu et al. (1997) to solve the resulting bounded constraint problem.

FPC-AS starts by executing shrinkage on the first phase and switch to second phase of subspace optimization once a good approximation of support set is found. Let $\zeta > 0$ and $\varepsilon_f, \varepsilon_g \in (0, 1)$ be some predetermined fixed constants. The shrinkage scheme switch at the k-th iteration if either of the following conditions hold,

$$\alpha^{k-1} \frac{\left\| \left[\nabla f\left(\boldsymbol{x}^k\right)\right]_{|S^c} \right\|_2}{\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|_2} > \zeta \quad \text{and} \quad \left\| \left[\nabla f\left(\boldsymbol{x}^k\right) - \mu\right]_{|S^c} \right\|_2 \leq \varepsilon_g \max\left\{\|\boldsymbol{x}^k\|_2, 1\right\}, \tag{3.1.23}$$

or

$$\left|F_\mu\left(\boldsymbol{x}^{k-1}\right) - F_\mu\left(\boldsymbol{x}^k\right)\right| \leq \varepsilon_f \max\left\{\left|F_\mu\left(\boldsymbol{x}^k\right)\right|, \left|F_\mu\left(\boldsymbol{x}^{k-1}\right)\right|, 1\right\}. \tag{3.1.24}$$

The justification of these conditions is based on the convergence property of line search implementation, Wen et al. (2010). These conditions are chosen so as to ensure the switching is not done prematurely. On one hand, a subspace optimization must be started as soon as possible. On the other hand, the complementary set of the support set that defines the subspace optimization problem must be as accurate as possible. Condition (3.1.23) ensures that a sufficient number of iterations are done before switching; while condition (3.1.24) avoids slow convergence.

The two stages are embedded under the continuation strategy to improve the rate of convergence. There are several halting conditions that can be used to terminate the algorithm. First, the algorithm can be stopped when the relative change of the solution falls below some tolerance value, that is $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|_2 \leq tol$. Secondly, one can terminate the algorithm when a norm of the gradient of smooth term is significantly small, that is $\left\|\nabla f(\boldsymbol{x}^k)\right\|_2 \leq tol$. These are the most suitable stopping conditions. Other conditions such as the limiting number of iterations can be used. A complete FPC-AS algorithm is described in **Algorithm 2**. FPC-AS exhibits state-of-the-art performance in terms of both its speed and ability to recover sparse signals.

Algorithm 2 Fixed Point Continuation Active Set Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m\times n}$, measurement vector $\boldsymbol{y}$, and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

Initialization: Set the initial solution $\boldsymbol{x}^0$. Select $\mu^0 > \mu^1 > \mu^2, > \cdots, > \mu^K = \mu$. Set all other parameters which are algorithm dependent.
for each $\mu^0, \mu^1, \ldots, \mu^K$ *do*
  While halting condition is not met *do*:
    Compute the stepsize α^k via (3.1.19).
    Update solution $\boldsymbol{x}^k$ by solving (3.1.20) via (3.1.15).
    Compute the support set $S = supp(\boldsymbol{x}^k)$.
    if either of condition (3.1.23) or (3.1.24) holds *do*
      Update solution by solving subspace problem (3.1.22).
    endif
  endWhile
endFor

3.1.2 Truncated Newton Interior Point Method. The ℓ_1-minimization problem (3.1.3) can be transformed to a convex quadratic program. The resulting problem can be solved by standard convex optimization methods such as interior point methods. The ℓ_1-magic by Candes and Romberg (2006b), provides high-quality general purpose implementation of standard interior point methods which can handle small to medium size problems. Kim et al.

(2007) propose a specialised interior point method for solving large-scale ℓ_1-minimization problem. The algorithm is termed *Truncated Newton Interior Point Method* (TNIPM) and it is normally referred to as $\ell_1-\ell_s$ in literature. TNIPM makes use of duality of ℓ_1-minimization problem and it uses preconditioned conjugate gradient to compute the search direction. Kim et al. (2007) have empirically shown that $\ell_1 - \ell_s$ is comparatively efficient and it scales well to large problems.

TNIPM uses a convex transformation to turn problem (3.1.4) into a bounded constrained convex quadratic problem:

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mu \sum_{i}^{n} u_i + \|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2^2, \tag{3.1.25}$$

$$\text{subject to} \quad -u_i \leq x_i \leq u_i,$$

where $\boldsymbol{u} \in \mathbb{R}^n$. A differentiable convex, logarithmic barrier function $\rho(\boldsymbol{x}, \boldsymbol{u})$, is further constructed for the bounded linear inequality constraints,

$$\rho(\boldsymbol{x}, \boldsymbol{u}) = -\left[\sum_{i}^{n} log(u_i + x_i) + \sum_{i}^{n} log(u_i - x_i)\right]. \tag{3.1.26}$$

The logarithmic barrier function is defined in the domain $\{(\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^n \times \mathbb{R}^n | \, |x_i| \leq u_i, \forall i\}$. The following optimization function can be obtained by augmenting the logarithmic function,

$$\min_{\boldsymbol{x},\boldsymbol{u}} \hat{F}(\boldsymbol{x}, \boldsymbol{u}) := t\mu \sum_{i}^{n} u_i + t\|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \rho(\boldsymbol{x}, \boldsymbol{u}), \tag{3.1.27}$$

where $t > 0$. The resulting objective function is differentiable, strictly convex and bounded below. Therefore problem (3.1.27) has a unique minimizer $(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t))$. This defines a curve in $\mathbb{R}^n \times \mathbb{R}^n$, parametrized by t, called the central path. The optimal solution is also shown to be dual feasible, Kim et al. (2007).

As a primal interior-point method, Kim et al. (2007) computed a sequence of points on the central path for an increasing sequence of t, and minimized the objective function $\hat{F}$ for each t using a truncated Newton's method. It is computationally expensive to compute the Hessian and ultimately obtain the search direction in order to solve problem (3.1.27). A preconditioned conjugate gradient algorithm is used to compute the search direction,

for which more details are provided by Kim et al. (2007). The interior point method was customized by the author in several ways. First, the dual feasible point and the associated duality gap are computed in a cheap fashion. These variables are necessary because the stopping condition is based on the duality gap. Secondly, the central path parameter t can be fixed, but Kim et al. (2007) updates t dynamically to achieve a robust convergence when combined with the preconditioned conjugate gradient algorithm. Thirdly, different options depending of the problem size are adapted for solving the Newton's system. Small and medium size problems are solved by direct methods such as Cholesky factorization, see Cotter et al. (1999) for details of this method. Large problems are solved using iterative methods such as conjugate gradient. Kim et al. (2007) provided a full discussion on the algorithm. A pseudo code of TNIPM is presented in **Algorithm 3**.

Algorithm 3 Truncated Newton Interior Point Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$, measurement vector $\boldsymbol{y}$, and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set $k = 0$. Initialize the solution $\boldsymbol{x}^0$ and set $\boldsymbol{u} = \mathbf{1}_n$. Set all other algorithm depended parameters.
2: *While* halting condition is not met *do*:
3: k = k + 1.
4: Compute the search direction $\boldsymbol{d}^{k-1}$, as approximate solution of the Newton system using preconditioned gradient method.
5: Compute the step size s^{k-1} by backtracking line search.
6: Update solution as $\left[\boldsymbol{x}^k; \boldsymbol{u}^k\right] = \left[\boldsymbol{x}^{k-1}; \boldsymbol{u}^{k-1}\right] + s^{k-1}\boldsymbol{d}^{k-1}$.
7: *endWhile*

The computational resources required by TNIPM are relatively moderate. Thus, TNIPM is able to solve very large problems, for which forming the Hessian would be prohibitively expensive. The computational speed of the algorithm is largely dominated by the calculation of the search direction. In particular, it is proportional to the product of total number of preconditioned conjugate gradient steps and the cost of each step. The total number of preconditioned conjugate gradient steps depends on the regularization parameter μ and

the desired accuracy. Small value of μ and high accuracy would lead to large number of preconditioned conjugate gradient steps. However, the implementation of the algorithm can solve the large sparse problems very efficiently with high accuracy, Kim et al. (2007).

3.1.3 Gradient Projection for Sparse Recovery - GPSR. Mathematical problems can be formulated in different ways and the most desirable formulation has been the smooth convex optimization problem. Figueiredo et al. (2007) developed the Gradient Projection for Sparse Recovery (GPSR) algorithm for solving Basis Pursuit Denoising:

$$\arg \min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mu \|\boldsymbol{x}\|_1 + \frac{1}{2}\|\boldsymbol{\Psi x} - \boldsymbol{y}\|_2^2, \tag{3.1.28}$$

where $\mu > 0$. GPSR reformulates problem (3.1.28), as a smooth convex quadratic program with bounded constraints. The approach starts by decomposing a signal $\boldsymbol{x}$ into its positive and negative components by defining the variables. Lets $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ be new variables which satisfy

$$|x_i| = u_i + v_i, \quad \text{for all} \quad i = 1, 2, \ldots, n. \tag{3.1.29}$$

Here $u_i = \max\{+x_i, 0\}$ and $v_i = \max\{-x_i, 0\}$. Moreover this choice of components satisfy $\boldsymbol{x} = \boldsymbol{u} - \boldsymbol{v}$. The linearisation of the ℓ_1-norm becomes

$$\begin{aligned} \|\boldsymbol{x}\|_1 &= \sum_{i=1}^{n} |x_i|, \\ &= \sum_{i=1}^{n} u_i + \sum_{i=1}^{n} v_i, \\ &= \mathbf{1}_n^T \boldsymbol{u} + \mathbf{1}_n^T \boldsymbol{v}, \end{aligned} \tag{3.1.30}$$

with $\boldsymbol{u}, \boldsymbol{v} \geq \mathbf{0}$ and $\mathbf{1}_n \in \mathbb{R}^n$ is a column vector whose entries are all ones. Using the above linearisation technique, problem (3.1.28) is transformed in the following manner,

$$
\begin{aligned}
\min_{\boldsymbol{u},\boldsymbol{v}} F(\boldsymbol{u},\boldsymbol{v}) &= \mu \mathbf{1}_n^T \boldsymbol{u} + \mu \mathbf{1}_n^T \boldsymbol{v} + \frac{1}{2}\|\boldsymbol{\Psi}(\boldsymbol{u}-\boldsymbol{v}) - \boldsymbol{y}\|_2^2, \\
&= \mu \mathbf{1}_n^T \boldsymbol{u} + \mu \mathbf{1}_n^T \boldsymbol{v} + \frac{1}{2}\left(\boldsymbol{\Psi}\boldsymbol{u} - \boldsymbol{\Psi}\boldsymbol{v} - \boldsymbol{y}\right)^T \left(\boldsymbol{\Psi}\boldsymbol{u} - \boldsymbol{\Psi}\boldsymbol{v} - \boldsymbol{y}\right), \\
&= \mu \mathbf{1}_n^T \boldsymbol{u} + \mu \mathbf{1}_n^T \boldsymbol{v} + \frac{1}{2}\left\{\boldsymbol{u}^T\left(\boldsymbol{\Psi}^T\boldsymbol{\Psi}\right)\boldsymbol{u} - \boldsymbol{u}^T\left(\boldsymbol{\Psi}^T\boldsymbol{\Psi}\right)\boldsymbol{v} \ldots\right. \\
&\qquad \left. - \boldsymbol{v}^T\left(\boldsymbol{\Psi}^T\boldsymbol{\Psi}\right)\boldsymbol{u} + \boldsymbol{u}^T\left(\boldsymbol{\Psi}^T\boldsymbol{\Psi}\right)\boldsymbol{u} - 2\boldsymbol{u}^T\boldsymbol{\Psi}^T\boldsymbol{y} + 2\boldsymbol{v}^T\boldsymbol{\Psi}^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y}\right\}, \\
&= \mu \mathbf{1}_n^T \boldsymbol{u} + \mu \mathbf{1}_n^T \boldsymbol{v} - \boldsymbol{u}\left(\boldsymbol{\Psi}^T\boldsymbol{y}\right)^T + \boldsymbol{v}\left(\boldsymbol{\Psi}^T\boldsymbol{y}\right)^T \ldots \\
&\qquad + \frac{1}{2}[\boldsymbol{u}^T \quad \boldsymbol{v}^T]\begin{bmatrix} +\boldsymbol{\Psi}^T\boldsymbol{u} - \boldsymbol{\Psi}^T\boldsymbol{v} \\ -\boldsymbol{\Psi}^T\boldsymbol{u} + \boldsymbol{\Psi}^T\boldsymbol{v} \end{bmatrix} + \boldsymbol{y}^T\boldsymbol{y}, \\
&= [\boldsymbol{u}^T \quad \boldsymbol{v}^T]\begin{bmatrix} \mu\mathbf{1}_n \\ \mu\mathbf{1}_n \end{bmatrix} + [\boldsymbol{u} \quad \boldsymbol{v}]\begin{bmatrix} -\boldsymbol{\Psi}^T\boldsymbol{y} \\ +\boldsymbol{\Psi}^T\boldsymbol{y} \end{bmatrix} \ldots \\
&\qquad + \frac{1}{2}[\boldsymbol{u}^T \quad \boldsymbol{v}^T]\begin{bmatrix} +\boldsymbol{\Psi}^T\boldsymbol{\Psi} & -\boldsymbol{\Psi}^T\boldsymbol{\Psi} \\ -\boldsymbol{\Psi}^T\boldsymbol{\Psi} & +\boldsymbol{\Psi}^T\boldsymbol{\Psi} \end{bmatrix}\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} + \boldsymbol{y}^T\boldsymbol{y}.
\end{aligned}
$$

Let us introduce the new variables as

$$
\boldsymbol{z} = \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} \qquad \boldsymbol{c} = \mu\mathbf{1}_{2n} + \begin{bmatrix} -\boldsymbol{\Psi}^T\boldsymbol{y} \\ \boldsymbol{\Psi}^T\boldsymbol{y} \end{bmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{\Psi}^T\boldsymbol{\Psi} & -\boldsymbol{\Psi}^T\boldsymbol{\Psi} \\ -\boldsymbol{\Psi}^T\boldsymbol{\Psi} & \boldsymbol{\Psi}^T\boldsymbol{\Psi} \end{bmatrix}, \tag{3.1.31}
$$

and problem (3.1.28) can be equivalently rewritten as the bounded constrained quadratic program

$$
\arg\ \min_{\boldsymbol{z}} F(\boldsymbol{z}) := \boldsymbol{c}^T\boldsymbol{z} + \frac{1}{2}\boldsymbol{z}^T\boldsymbol{B}\boldsymbol{z}, \quad \text{subject to } \boldsymbol{z} \geq 0. \tag{3.1.32}
$$

Although, the technique transforms a non-differentiable problem to a differentiable problem, the linearisation comes with a burden: the dimension of the new formulation doubles. This is an issue of serious concern in practice. However, the increase in dimension has minor impact on the complexity of the algorithm. Matrix operation involving $\boldsymbol{B}$ (in the case of computing ∇F and $\boldsymbol{z}^T\boldsymbol{B}\boldsymbol{z}$) can be performed more economically by exploring its structure. The operations reduce to matrix multiplication involving either $\boldsymbol{\Psi}$ or $\boldsymbol{\Psi}^T$, Figueiredo et al. (2007). Thus with good implementation of matrix multiplication the proposed gradient projection method can be fast.

The proposed gradient projection algorithm solves problem (3.1.32) iteratively and the k-th iteration consists of two steps. First step chooses an appropriate step size $\alpha^{k-1} > 0$, it then searches the solution along the negative gradient and project it onto a positive orthant as

$$\boldsymbol{w}^k = \left(\boldsymbol{z}^{k-1} - \alpha^{k-1}\nabla F(\boldsymbol{z}^{k-1})\right)_+ . \tag{3.1.33}$$

Here the projection $(\cdot)_+$ restricts the components of the argument vector to positive values by setting all negative components to zeros. The projection is necessary due to the constraint on the solution vector $\boldsymbol{z}$. The second step chose the parameter $\beta^{k-1} \in (0, 1)$ and updates the solution

$$\boldsymbol{z}^k = \boldsymbol{z}^{k-1} + \beta^{k-1}(\boldsymbol{w}^k - \boldsymbol{z}^{k-1}). \tag{3.1.34}$$

GPSR algorithm continues this until some convergence criterion is satisfied. The main issue is how to choose the step size α^k and weight β^k at each iteration. Step size is chosen such that a decrease on the objective function is achieved at each iteration.

Basic implementation of GPSR employs backtracking line search to select the step size. Backtracking line search avoids large step size that overshot the new point, which may result in increasing the objective function. At the same time it also avoids small step size, which may cause slow convergence of the algorithm. Figueiredo et al. (2007) uses the initial guess for α^k, that would yield an exact minimiser of F along the descent directions. Specifically initial step size is given as:

$$\alpha_0^k = \frac{\left(\boldsymbol{g}^k\right)^T \boldsymbol{g}^k}{\left(\boldsymbol{g}^k\right)^T \boldsymbol{B}\boldsymbol{g}^k}, \tag{3.1.35}$$

where for $i = 1, \ldots, 2n$

$$g_i^k = \begin{cases} (\nabla F(\boldsymbol{z}^k))_i & \text{if} \quad z_i^k > 0 \text{ or } (\nabla F(\boldsymbol{z}^k))_i < 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.1.36}$$

The components of the vector $\boldsymbol{g^k}$ gives an indication of the favourable directions where there is a possibility of zeroing some complements of the solution. Figueiredo et al. (2007) keeps the algorithm stable by removing potentially non-productive step sizes by selecting the middle value of the set $\{\alpha_{min}, \alpha_0^k, \alpha_{max}\}$. Denoting the middle value as α, the GPSR

algorithm uses this value as the initial guess of the step size in the backtracking line search. The value is sequentially reduced and the step size α^{k-1} is accepted as the first value such that the Armirjo condition is satisfied. The Armijo condition is specified as:

$$F(\boldsymbol{z}^{k-1} + \alpha^{k-1}\boldsymbol{d}^{k-1}) \leq F(\boldsymbol{z}^{k-1}) + \alpha^{k-1}\sigma\nabla F(\boldsymbol{z}^{k-1})^T\boldsymbol{d}^{k-1}, \tag{3.1.37}$$

where parameter $\sigma \in (0, \frac{1}{2})$ is fixed and $\boldsymbol{d}^k$ is the search direction. The parameter β^{k-1} in equation (3.1.34) is updated in way that GPSR algorithm mimics the convergence property of the Newton method, specifically

$$\beta^{k-1} = \frac{(\boldsymbol{z}^{k-1} - \boldsymbol{z}^{k-2})^T(\boldsymbol{z}^{k-1} - \boldsymbol{z}^{k-2})}{(\boldsymbol{z}^{k-1} - \boldsymbol{z}^{k-2})^T\boldsymbol{g}^{k-1}}. \tag{3.1.38}$$

The algorithm is normally terminated when the relative change of solution norm falls below some tolerance. After the convergence of the above iteration, Figueiredo et al. (2007) employed debiasing to improve the accuracy of the algorithm. In the debiasing procedure a subspace problem restricted to non-zero components of the solution is solved to find optimal signal values. The algorithm also benefits from the use of the continuation strategy, which improves the convergence rate. The GPSR is a computationally light iterative approach to solving the convex relaxation principle of Basis Pursuit. The complete algorithm is defined as in **Algorithm** 4.

3.1.4 Augmented Lagrangian Algorithm - SALSA. Split Augmented Lagrangian algorithm was proposed by (Afonso et al., 2009), for solving *Basis Pursuit Denosing* in the context of image restoration. The approach is based on the principle of variable splitting to obtain an equivalent constrained optimization problem. The resulting transformation is then addressed with *Alternating Direction Method of Multipliers* which belongs to a family of Augmented Lagrangian methods.

Consider an unconstrained optimization problem in which the objective function is a sum of two functions:

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := g(\boldsymbol{x}) + f(\boldsymbol{x}), \tag{3.1.39}$$

where $g, f : \mathbb{R}^n \to \mathbb{R}$. The idea is to split the variable $\boldsymbol{x}$ into a pair of variables, e.g. $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{z} \in \mathbb{R}^n$, each to serve as an argument of each of the two functions. Then minimizing

Algorithm 4 Gradient Projection for Sparse Recovery Algorithm

Input: An observed signal $\boldsymbol{y} \in \boldsymbol{R}^m$, a sensing matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$ and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Construct and initialise $\boldsymbol{z}^0$, choose $\beta \in (0, 1)$, and set $k = 0$.
2: *While* not convergence *do*:
3: k = k + 1.
4: Compute α_0^k via (3.1.35), and set α to be middle value of the set $\{\alpha_{min}, \alpha_0^k, \alpha_{max}\}$.
5: Backtracking line search: choose α^k to be the first value such that the Armijo condition (3.1.37) is satisfied.
6: Update the $\boldsymbol{w}^k$ via (3.1.33)
7: Compute β^{k-1} via (3.1.38) and update the solution $\boldsymbol{z}^k$ via (3.1.34).
8: do convergence test.
9: *endWhile*

the constrained problem

$$\min_{\boldsymbol{x},\boldsymbol{z}} \hat{F}(\boldsymbol{x}, \boldsymbol{z}) := g(\boldsymbol{z}) + f(\boldsymbol{x}), \quad \text{subject to } \boldsymbol{x} = \boldsymbol{z}, \tag{3.1.40}$$

is equivalent to (3.1.39), since in the feasible set $\{(\boldsymbol{x}, \boldsymbol{z}) : \boldsymbol{x} = \boldsymbol{z}\}$ the objective function in (3.1.39) coincides with that of (3.1.40). The simple rationale behind variable splitting is that it may be simple to optimize one problem over the other. In fact there exist Lagrangian methods as efficient techniques for solving constrained optimization problem.

A similar variable splitting approach underlies a recent development of split Bregman methods, Goldstein and Osher (2008) which was proposed to address ℓ_1-regularization problems. In these methods, the constrained problem is addressed using a Bragman iterative algorithms, which has been shown to be equivalent to Augmented Lagrangian methods, Yin et al. (2008).

3.2 Greedy Pursuit Algorithms

This section, discusses a class of algorithms collectively referred to as greedy pursuit algorithms. Generally speaking, a greedy algorithm refers to any algorithm following the metaheuristic of choosing the best immediate local optimum. This strategy can find the global optimum for some optimization problems, but not for all, Cormen et al. (2012). In compressive sensing, the main principle of greedy algorithms is to detect or estimate the underlying support set of a sparse signal vector, followed by evaluation of associated signal values. For the detection of support set, greedy algorithms typically use the *matched filter*. For estimating the associated signal values, they normally solve the least squares problem, Mallat and Zhang (1993).

The basic principle of the matched filter detector is to filter measurements that match features of interest in the signal. In compressive sensing, a very important feature is associated with active columns of the measurement matrix $\boldsymbol{\Psi}$. At the optimal solution, active columns correspond to the non-zero components of the sparse signal $\boldsymbol{x}$. This follows from the fact that for a s-sparse signal $\boldsymbol{x} \in \mathbb{R}^n$, the measurements $\boldsymbol{y} = \boldsymbol{\Psi x}$ in $\mathbb{R}^m$, are merely a weighted linear combination of s columns of $\boldsymbol{\Psi}$. Let $\boldsymbol{r} \in \mathbb{R}^m$ be a known quantity (typically residual), then matched filter detection is based on computing the absolute correlation vector $|\boldsymbol{\Psi}^T \boldsymbol{r}|$, and selecting one or several indices corresponding to the largest components. Due to correlation and measurement of noise, the results of matched filter are usually not used directly as estimator of the target signal $\boldsymbol{x}$, but instead as a detector for the support set S. Once S is known, least squares estimation is employed to estimate the signal values of the active components of $\boldsymbol{x}$.

Least squares estimation is one of the main tools for many of greedy algorithms. The standard compressive sensing problem cannot be solved directly with least squares, since it is an under-determined set of linear algebraic equations. However, if the true support set S of $\boldsymbol{x}$ is accurately detected, then one has $\boldsymbol{\Psi x} = \boldsymbol{\Psi}_S \boldsymbol{x}_{|S}$, in the case of noiseless measurements. Here, $\boldsymbol{x}_{|S}$ represents a signal that keeps the components corresponding to indices in the support set S while setting the rest of the components to zero. Also, $\boldsymbol{\Psi}_S$ denotes the column

sub-matrix of $\boldsymbol{\Psi}$ whose columns are listed in S. Since the cardinality of S is less than m, the measurements, $\boldsymbol{y} = \boldsymbol{\Psi}_S \boldsymbol{x}_{|S}$ represents an over-determined set of linear equations, to which least squares can be used to find the unique solution. The typical least squares estimate of the signal is a solution $\boldsymbol{x}^*_{|S}$ of the problem:

$$\boldsymbol{x}^*_{|S} = \arg \min_{x_{|S}} \|\boldsymbol{\Psi}_S \boldsymbol{x}_{|s} - \boldsymbol{y}\|_2^2. \tag{3.2.1}$$

The least squares is a quadratic problem. One can find the exact solution by evoking the first order optimality condition of problem (3.2.1):

$$\begin{aligned}
\mathbf{0} &= \frac{\partial}{\partial \boldsymbol{x}_{|S}} \|\boldsymbol{\Psi}_S \boldsymbol{x}_{|s} - \boldsymbol{y}\|_2^2, \\
&= \frac{\partial}{\partial \boldsymbol{x}_{|S}} \left[\left(\boldsymbol{\Psi}_S \boldsymbol{x}_{|s} - \boldsymbol{y} \right)^T \left(\boldsymbol{\Psi}_S \boldsymbol{x}_{|s} - \boldsymbol{y} \right) \right], \\
&= \frac{\partial}{\partial \boldsymbol{x}_{|S}} \left[\boldsymbol{x}_{|S}^T \boldsymbol{\Psi}_S^T \boldsymbol{\Psi}_S \boldsymbol{x}_{|S} - \boldsymbol{x}_{|S}^T \boldsymbol{\Psi}_S^T \boldsymbol{y} - \boldsymbol{y} \boldsymbol{\Psi}_S^T \boldsymbol{x}_{|S} + \boldsymbol{y}^T \boldsymbol{y} \right] \\
&= -2\boldsymbol{\Psi}_S^T \boldsymbol{y} + 2\boldsymbol{\Psi}_S^T \boldsymbol{\Psi}_S \boldsymbol{x}_{|S}.
\end{aligned} \tag{3.2.2}$$

Here we only consider the case where $\boldsymbol{\Psi}_S$ has full column rank, in that case the analytical solution of problem (3.2.1) is,

$$\boldsymbol{x}^*_{|S} = \boldsymbol{\Psi}_S^\dagger \boldsymbol{y} = \left(\boldsymbol{\Psi}_S^T \boldsymbol{\Psi}_S \right)^{-1} \boldsymbol{\Psi}_S^T \boldsymbol{y}. \tag{3.2.3}$$

Here $\boldsymbol{\Psi}^\dagger = \left(\boldsymbol{\Psi}^T \boldsymbol{\Psi} \right)^{-1} \boldsymbol{\Psi}^T$, denotes the pseudo inverse of the matrix $\boldsymbol{\Psi}$. In practice, the true support set S is usually not accessible, but one can only access an estimate of a support set. If one can indeed accurately detect the true support set and there is no noise present, then (3.2.3) gives the values of the active components in $\boldsymbol{x}$.

In literature, there are two main types of greedy algorithms. The first categorization is that the algorithm is either *serial* or *parallel*. In serial implementation, the matched filter selects one component at each iteration. While in parallel implementation, multiple columns can be selected simultaneously. The second categorization is that the algorithm is either *irreversible* or *reversible*. In irreversible reconstruction once a component is added to the support set, it remains there forever. While the reversible construction allows modification of the support set. Early greedy methods tend to implement serial-irreversible approach,

while contemporary algorithms trend to implement parallel-reversible approach. Below we discuss some of the greedy algorithms which have attracted interests in literature.

3.2.1 Orthogonal Matching Pursuit. Earlier greedy algorithms include Matching Pursuit (MP), Mallat and Zhang (1993), and its improved version Orthogonal Matching Pursuit (OMP), Pati et al. (1993); Tropp and Gribert (2007). Using the fact that the measurements $\boldsymbol{y}$ are just the weighted sum of active columns in measurement matrix $\boldsymbol{\Psi}$, these algorithms iteratively decompose a signal into linear combination of columns from measurement matrix $\boldsymbol{\Psi}$. At each iteration, they use a greedy rule to identify a column in $\boldsymbol{\Psi}$ that best approximates the signal and add the corresponding index to the support set of the signal.

In MP, the main quantity of the matched filter detector is the residual $\boldsymbol{r} \in \mathbb{R}^m$. The residual is defined as $\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{\Psi x}$ and it represents the yet to be explained portion of the measurements. Let $\boldsymbol{\Psi}$ be a matrix with normalised columns $\|\boldsymbol{\psi}_i\|_2 = 1$, then the matched filter in MP calculates the absolute correlation between columns of $\boldsymbol{\Psi}$ and the residual $\boldsymbol{r}$:

$$|\boldsymbol{\psi}_i^T \boldsymbol{r}|, \quad \text{for all} \quad i. \tag{3.2.4}$$

The greedy rule selects the i-th component which corresponds to the maximum correlation. Once this column is selected one has a better representation of the signal, since the new index i has been added to the current support set. The second fundamental step in MP is concerned with updating the residual and the i-th component of the signal. At the k-th iteration, these quantities are respectively updated as

$$\boldsymbol{r}^k = \boldsymbol{r}^{k-1} - \langle \boldsymbol{r}^{k-1}, \boldsymbol{\psi}_i \rangle \boldsymbol{\psi}_i, \quad \text{and} \tag{3.2.5}$$

$$\boldsymbol{x}_i^k = \boldsymbol{x}_i^{k-1} + \langle \boldsymbol{r}^k, \boldsymbol{\psi}_i \rangle. \tag{3.2.6}$$

The procedure is repeated until convergence is reached. At convergence, the suitable halting criterion is when the norm of $\boldsymbol{r}^k$ becomes smaller than some tolerance. It is known that the residual norm $\|\boldsymbol{r}^k\|$ converges linearly to zero, see Mallat and Zhang (1993). Thus, the algorithm terminates in a finite number of iterations. MP is fully described by a pseudo code in **Algorithm 5**.

MP is a serial type of greedy algorithm. It selects one column of $\boldsymbol{\Psi}$ at each iteration, and it does not keep track of the support set. Thus MP, will often repeat the selection of columns

Algorithm 5 Matching Pursuit Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$ and measurement vector $\boldsymbol{y}$.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set $k = 0$. Set the initial solution $\boldsymbol{x}^0 = \boldsymbol{0}$. Set the residual $\boldsymbol{r}^0 = \boldsymbol{y}$.

2: *While* halting condition is not met *do*:

3: Set $k = k + 1$

4: Select index i so that $\max_i |\boldsymbol{\psi}_i^T \boldsymbol{r}^k|$.

5: Update the residual $\boldsymbol{r}^k$, via (3.2.5).

6: Update the i-th component of the signal $\boldsymbol{x}_i^k$, via (3.2.6).

7: *endWhile*

in an attempt to improve the quality of the approximation. In general, matching filtering of MP is not good. The selection of a column does not always provide good performance in practice. To better understand this, let us study the i-th component of the signal $\boldsymbol{x}$:

$$\begin{aligned} x_i &= \boldsymbol{\psi}_i^T \boldsymbol{y}, \\ &= \boldsymbol{\psi}_i^T \sum_j^n \boldsymbol{\psi}_j x_j, \\ &= \boldsymbol{\psi}_i^T \boldsymbol{\psi}_i x_i + \sum_{j \neq i}^n \boldsymbol{\psi}_i^T \boldsymbol{\psi}_j x_j, \\ &= x_i + \sum_{j \neq i}^n \boldsymbol{\psi}_i^T \boldsymbol{\psi}_j x_j. \end{aligned} \tag{3.2.7}$$

Notice, if $\boldsymbol{\Psi}$ is not orthogonal, then the second term of (3.2.7) is not equal to zero. Although one can argue that this term is small, it has undesirable effects on the final solution. To illuminate the resulting effects, one would orthogonalise the newly selected column $\boldsymbol{\psi}_i$ with all columns of $\boldsymbol{\Psi}$ corresponding to the current support set. Hence the algorithm resulting from this modification is called *Orthogonal Matching Pursuit*.

Orthogonal matching pursuit (OMP) is an improvement of matching pursuit. The principle is the same, at every iteration a column is picked from the measurement matrix that best approximates the signal. However, instead of subtracting the contribution of the column $\boldsymbol{\psi}_i$ with which the residual $\boldsymbol{r}$ is maximally correlated, one has to compute the projection of

$\boldsymbol{r}$ onto the orthogonal subspace spanned by the columns corresponding to the support set. That is after each iteration OMP orthogonalises the columns corresponding to the support set. This removes erroneous term in equation (3.2.7) and it ensures that no column would be selected for the second time on subsequent iterations. Thus the maximum number of iterations are bounded.

The k-th iteration of OMP can be completely described in four steps:

- *Selection:* The algorithm selects the column of measurement matrix $\boldsymbol{\Psi}$ that is maximally correlated with the current residual. That is

$$i^* = \arg \max_i |\boldsymbol{\psi}_i^T \boldsymbol{r}^k|, \tag{3.2.8}$$

  where i^* denoted the index of the column which is most correlated with the residual.

- *Support Merge:* Unlike MP, OMP keeps track of already selected columns. The newly selected index i^* is united with the previous support set. That is

$$S^k = S^{k-1} \cup \{i^*\}. \tag{3.2.9}$$

- *Estimation:* Instead of only updating the i^* component of signal estimate, OMP solves subspace least squares problem. The components of the signal are restricted to the current support set S^k. That is

$$\boldsymbol{x}^k = \arg\min_{\boldsymbol{x}} \|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2, \quad \text{subject to} \quad supp(\boldsymbol{x}) = S^k. \tag{3.2.10}$$

  The exact solution of problem (3.2.10) can be obtained via equation (3.2.3).

- *Residual update:* The main feature of OMP is orthogonalization. The residual is updated as

$$\boldsymbol{r}^k = \boldsymbol{y} - \boldsymbol{\Psi}_S \boldsymbol{x}_{|S} = (\boldsymbol{I} - \mathcal{P}_k)\, \boldsymbol{y}, \tag{3.2.11}$$

  where $\mathcal{P}^k = \boldsymbol{\Psi}_S \left(\boldsymbol{\Psi}_S^T \boldsymbol{\Psi}_S\right)^{-1} \boldsymbol{\Psi}_S^T$ is the orthogonal projection onto the span of the columns of $\boldsymbol{\Psi}$ restricted to the current support set $S = S^k$.

The above procedures in the k-th iteration are repeated until convergence sets in. Due to orthogonalization, OMP converges in no more than s iterations, with s being the sparsity level of the problem. Comparing computational performance, this is a huge improvement over MP. In MP one terminates the algorithm when residual falls below some predetermine tolerance value. OMP is fully described in a pseudo code in **Algorithm 6**.

Algorithm 6 Orthogonal Matching Pursuit Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$ and measurement vector $\boldsymbol{y}$.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set $k = 0$. Set the initial solution $\boldsymbol{x}^0 = \boldsymbol{0}$. Set the residual $\boldsymbol{r}^0 = \boldsymbol{y}$. Set the initial support set $S^0 = \emptyset$.
2: *While* halting condition is not met *do*:
3: Set $k = k + 1$
4: Select index i so that $\max_i |\boldsymbol{\psi}_i^T \boldsymbol{r}^k|$, then update the support set $S^k = S^{k-1} \cup i$.
5: Update the signal estimate $\boldsymbol{x}^k$, via (3.2.10).
6: Update the residual $\boldsymbol{r}^k$, via (3.2.11).
7: *endWhile*

The main computational cost of OMP is due to matrix multiplication in the least squares problems. Methods involving Cholesky factorisation by Cotter et al. (1999), QR factorisation by Mallat and Zhang (1992) and gradient methods have been proposed. OMP is provably a fast algorithm, both computationally and theoretically. However, the guarantees accompanying OMP for sparse recovery are weaker than those associated with convex optimization based techniques. In fact the main results of OMP are given in the following theorem by Tropp and Gribert (2007):

Theorem 3.3 (Recoverability of OMP, Tropp and Gribert (2007))**.** *Fix* $\delta \in (0, 0.36)$ *and choose* $m \geq sClog\left(\frac{n}{\delta}\right)$*. Assume that* $\boldsymbol{x} \in \mathbb{R}^n$ *is* s*-sparse and let* $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$ *have independently and identically distributed entries from the Gaussian distribution* $\mathcal{N}\left(0, \frac{1}{m}\right)$*. Then, given the measurements* $\boldsymbol{y} = \boldsymbol{\Psi}\boldsymbol{x}$*, OMP can reconstruct the signal* $\boldsymbol{x}$ *with probability exceeding* $1 - 2\delta$*, where the constant* C *satisfies* $C \leq 20$*.*

The results are very restrictive in terms of the measurement matrix and the probability

of success is quite low. These result are not as strong as those associated with convex optimization techniques. Furthermore reconstruction guarantees are not uniform, that is it cannot be shown that a signal measurement matrix can be used to recover every possible s-sparse signal. Another issue with OMP is robustness to noise. There are no known proven results about noise bounds. The total complexity of OMP increases linearly with sparsity and measurements.

3.2.2 Stagewise Orthogonal Matching Pursuit. Orthogonal Matching pursuit is not effective when the signal is not very sparse. The computational complexity of OMP increases linearly with sparsity level s of the signal $\boldsymbol{x}$. Practical problems are often large scale and the signal is not very sparse, for example in image processing applications. Thus in such situations, OMP is ineffective. Therefore, OMP has been revised in a way to find less accurate solution for such large scale problems in reasonable time. The modified version of OMP was proposed by Donoho et al. (2006) and termed the resulting algorithm Stagewise Orthogonal Matching Pursuit (StOMP).

The main difference with OMP, is with the matched filter. That is the manner in which StOMP selects columns from the measurement matrix. Specifically, StOMP is characterized by adding multiple columns to the support set at each iteration instead of considering a single column as in the case of OMP. This is done by thresholding. One selects all columns whose absolute correlation with residual is greater than a specially chosen threshold value. The selection of correct correlation value is important, as the performance of the algorithm depends upon it. More specifically, the threshold parameter τ, is calculated at each iteration as

$$\tau^k = t^k \frac{\|\boldsymbol{r}^{k-1}\|_2}{\sqrt{n}}, \tag{3.2.12}$$

where $\boldsymbol{r}^{k-1}$ is the residual. Donoho et al. (2006), suggest the ideal value for the parameter t to be in the domain $2 \leq t \leq 3$. The idea behind the thresholding strategy is simply to allow many columns to enter the support set, thus improving efficiency of the algorithm. Beside thresholding step, StOMP is very much similar to OMP. The k-th iteration is outline in four steps as:

- *Selection:* The algorithm applies matched filter on the current residual, by computing a vector $\boldsymbol{c}^k$ of residual correlation:

$$\boldsymbol{c}^k = \boldsymbol{\Psi}^T \boldsymbol{r}^{k-1}, \tag{3.2.13}$$

  which contains a small number of significant non-zero components. The selection step is completed by performing hard thresholding to find the significant non-zero components. Threshold yields a small set J^k of large coordinates defined as

$$J^k = \{i : |\boldsymbol{c}^k(i)| \geq \tau^k\}, \tag{3.2.14}$$

  where $\boldsymbol{c}^k$ is defined by (3.2.13).

- *Support merge:* A set of newly selected components is united with the previous support set,

$$S^k = S^{k-1} \cup J^k. \tag{3.2.15}$$

  After the support merge, the algorithm orthogonalise the newly identified columns with the existing ones.

- *Estimation:* Similar to OMP, the StOMP algorithm solves the subspace least squares problem (3.2.10). The problem is restricted to components associated with the index of the current support set and its analytical solution is given by equation (3.2.3).

- *Residual update:* The final step updates the residual via equation (3.2.11).

The above steps are repeated until convergence is achieved. StOMP is fully described in a pseudo code in **Algorithm 7**. Just like any other greedy algorithms, the convenient stopping condition is when the residual norm falls below some specified tolerance. Although StOMP has a structure similar to that of OMP but the fact that StOMP selects multiple components at each iteration improve its computational complexity. The convergence is faster and the algorithm converges in much less number of iterations compared to OMP.

3.2.3 Compressive Sampling Matching Pursuit. Matching pursuit algorithms, OMP and StOMP discussed so far, mitigate issues of computational complexity encountered in

Algorithm 7 Stagewise Orthogonal Matching Pursuit Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$ and measurement vector $\boldsymbol{y}$.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set $k = 0$. Set the initial solution $\boldsymbol{x}^0 = \mathbf{0}$. Set the residual $\boldsymbol{r}^0 = \boldsymbol{y}$. Set the initial support set $S^0 = \emptyset$.

2: *While* halting condition is not met *do*:

3: Set $k = k + 1$

4: Compute the residual correlation vector, $\boldsymbol{c}^k = \boldsymbol{\Psi}^T \boldsymbol{r}^{k-1}$.

5: Compute the threshold parameter, $\tau^k = t^k \frac{\|\boldsymbol{r}^{k-1}\|}{\sqrt{n}}$, where $2 \leq t^k \leq 3$.

6: Select the components that are significant non-zero from the residual correlation, $J^k = \{i : |\boldsymbol{c}^k(i)| \geq \tau^k\}$, and then update the support set $S^k = S^{k-1} \cup J^k$.

7: Update the signal estimate $\boldsymbol{x}^k$, via (3.2.10).

8: Update the residual $\boldsymbol{r}^k$, via (3.2.11).

9: *endWhile*

convex optimization based methods. However, these algorithms do not have strong uniform recovery guarantees, similar to those associated with convex optimization algorithms. Recent developments of this type of algorithms attempt to bridge the gap between uniformity and complexity. In Dai and Milenkovic (2009), Space Pursuit (SP) is proposed. SP has the same computational complexity as those of OMP, but with reconstruction accuracy on the same order as those of convex optimization algorithms. In Needell and Vershynin (2009, 2010), an algorithm termed Regularized Orthogonal Matching Pursuit (ROMP), is proposed. ROMP combines the speed and ease of implementation of greedy algorithms with the strong performance guarantees of convex optimization algorithms.

In Needell and Tropp (2009), an algorithm termed Compressive Sampling Matching Pursuit (CoSaMP) is discussed. CoSaMP provides the same performance guarantees as those of best optimization based algorithms, and its complexity is dominated by matrix vector multiplication. CoSaMP and SP incorporate the ideas from convex optimization and combinatorial algorithms. Convex relaxation algorithms succeed with a very small number of measurements, but they tend to be computationally burdensome. On the other hand,

it will be seen in later section that combinatorial algorithms are extremely fast. They are sublinear in the length of the signal, but require a large number of measurements that may not be easy to acquire. CoSaMP is at the heart of greedy algorithm, it guarantees speed and it provides rigorous error bounds. Here we discuss CoSaMP as representative of recent developments of greedy algorithms.

CoSaMP uses the approach inspired by the restricted isometry property. Suppose that measurement matrix $\boldsymbol{\Psi}$ has restricted isometry constant $\delta_s \ll 1$. For an s-sparse signal $\boldsymbol{x}$, the vector $\boldsymbol{u} = \boldsymbol{\Psi}^T\boldsymbol{\Psi}\boldsymbol{x}$ can serve as a proxy for the signal because the energy in each set of s components of $\boldsymbol{u}$ approximates the energy in the corresponding s components of $\boldsymbol{x}$. In particular, the largest s entries of the proxy $\boldsymbol{u}$ point towards the largest s entries of the signal $\boldsymbol{x}$. Since the measurements have the form $\boldsymbol{y} = \boldsymbol{\Psi}\boldsymbol{x}$, we can obtain the proxy just by applying the matrix $\boldsymbol{\Psi}^T$ to the measurements.

The algorithm invokes this idea iteratively to approximate the target signal. At each iteration the current approximation induces a residual, a part of the target signal that has not been approximated. As the algorithm processes, the samples are updated so that they reflect the current residual. These samples are used to construct a proxy for the residual, which permits the identification of the large components in the residual. This step yields a tentative support for the next approximation. Then the measurements are used to estimate the approximation on this support set using least squares model. This process is repeated until we have found the recoverable energy in the signal.

Let $\boldsymbol{x}_{(s)}$ represent the signal that keeps s largest components of $\boldsymbol{x}$ while setting all other components to zero. For an index set S, let $\boldsymbol{x}_{|S}$ represent the signal that keeps the components whose index are specified by S while setting the rest of the components to zero. Also, let $\boldsymbol{\Psi}_S$ denotes the column submatrix of $\boldsymbol{\Psi}$ whose columns are listed in S. The CoSaMP algorithm consist of five major steps:

- *Identification:* The algorithm forms a signal proxy $\boldsymbol{u} = \boldsymbol{\Psi}^T\boldsymbol{r}$ from the current measurements and identify the support set Ω of cardinality $2s$ formed by the largest components, that is $\Omega = \boldsymbol{u}_{(2s)}$.

- *Support Merger:* The algorithm merges the newly identified support Ω with the support of the sparse signal in the previous iteration, that is $S = \Omega \cup supp(\boldsymbol{x}^{k-1})$.
- *Estimation:* The algorithm estimates the signal by solving a least squares problem to the components on the merged set, that is $\boldsymbol{x}^* = \arg\ \min_{\boldsymbol{x}_{|S}} \|\boldsymbol{\Psi}_S \boldsymbol{x}_{|S} - \boldsymbol{y}\|_2^2$.
- *Pruning:* The algorithm produces a new approximation by retaining s largest components of the least squares solution, that is $\boldsymbol{x}^k = \boldsymbol{x}^*_{(s)}$.
- *Residual update:* Finally, the algorithm update the the residual $\boldsymbol{r}$, which contains the signal that is yet to be approximated.

The algorithm stops when the halting condition is met. There are three major approaches to halting the algorithm. In the first approach, the algorithm terminates after a fixed number of iterations. In the second approach, the algorithm stops when ℓ_2-norm of the residual falls below some fixed tolerance, i.e. $\|\boldsymbol{r}\|_2 \leq \epsilon_r$ for very small $\epsilon_r > 0$. The choice of different approaches and parameters dictates the trade-off between accuracy of approximation and computational complexity. The complete CoSaMP is presented with a pseudo code in **Algorithm 8**.

Algorithm 8 Compressive Sampling Matching Pursuit Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$, measurement vector $\boldsymbol{y}$, and sparsity level s.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set $k = 0$. Set the initial solution $\boldsymbol{x}^0 = \mathbf{0}$. Set the residual $\boldsymbol{r}^0 = \boldsymbol{y}$. Set the initial support set $S^0 = \emptyset$.
2: *While* halting condition is not met *do*:
3: Set $k = k + 1$
4: Form signal proxy $\boldsymbol{u}^k = \boldsymbol{\Psi}^T \boldsymbol{r}^{k-1}$
5: *Identify* largest signal components $\Omega = \boldsymbol{u}_{(2s)}$
6: *Merge* support $S^k = \Omega \cup supp(\boldsymbol{x}^{k-1})$
7: *Estimate* signal using by solving $\boldsymbol{x}^* = \arg\ \min_{\boldsymbol{x}_{|S}} \|\boldsymbol{\Psi}_S \boldsymbol{x}_{|S} - \boldsymbol{y}\|_2^2$ and set $\boldsymbol{x}^*_{|S^c} = 0$.
8: *Prune* the signal to obtain the next approximation $\boldsymbol{x}^k = \boldsymbol{x}^*_s$
9: *Update* the current residual $\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{x}^k$
10: *endWhile*

The main results of CoSaMP are stated in terms of restricted isometry property and relate to the accuracy of reconstruction, and computational complexity of the algorithm. Consider the following theorem by Needell and Tropp (2009):

Theorem 3.4 (Recoverability of CoSaMP, Needell and Tropp (2009))**.** *Suppose the measurement matrix $\Psi \in \mathbb{R}^{m \times n}$ has the restricted isometry constant of $\delta_{2s} \leq c$. Let $\boldsymbol{y} = \boldsymbol{\Psi}\boldsymbol{x} + \boldsymbol{e}$ be a vector of samples of an arbitrary signal, contaminated with arbitrary noise $\boldsymbol{e}$. For a given precision parameter ε, CoSaMP produces an s-sparse approximation $\boldsymbol{x}^*$ of the original signal $\boldsymbol{x}_0$ that satisfies*

$$\|\boldsymbol{x}^* - \boldsymbol{x}_0\|_2 \leq C \times \max\left\{\varepsilon, \frac{1}{\sqrt{s}}\|\boldsymbol{x}^* - \boldsymbol{x}^*_{(s/2)}\|_1 + \|\boldsymbol{e}\|_2\right\}. \tag{3.2.16}$$

Here $\boldsymbol{x}_{(s/2)}$ is a best $(s/2)$-sparse approximation of $\boldsymbol{x}$. The running time is $\mathcal{O}\left(\mathcal{L} \times log\left(\|\frac{\boldsymbol{x}^*\|_2}{\varepsilon}\right)\right)$, where $\mathcal{L}$ bounds the cost of a matrix-vector multiply with $\boldsymbol{\Psi}$ or $\boldsymbol{\Psi}^T$. Working storage is $\mathcal{O}(n)$.

These results are as strong as those associated with the convex optimization algorithm. The CoSaMP algorithm only requires the measurement matrix to satisfy a bound of restricted isometry constant. Many types of random matrices have this property. Thus the algorithm

is suitable to use in many measurement technologies. The reconstruction guarantees are uniform. The error bound demonstrates that the algorithm succeeds for all signals, even when there is noise in the measurement. The efficiency and effectiveness of the algorithm depend on the signal structure and measurement matrix.

3.3 Iterative Thresholding Algorithms

Greedy pursuit algorithms are shown to be efficient in terms of speed and they are easy to implement. However, they do not have recovery guarantees as strong as algorithms based on convex relaxation. The algorithms discussed in this section are aimed at bridging this gap. They are fairly easy to implement and can be extremely fast but also show the strong performance guarantees available when compared with the algorithms based on convex relaxation. The general idea of iterative thresholding algorithms is to approximate the objective function to a form where concepts such as shrinkage threshold can be used. These methods offer a powerful set of tools for addressing compressive sensing problem. We concentrate on two algorithms, the Iterative Hard Thresholding (IHT) which has received less attention in literature as compared to a second algorithm, Iterative Soft Thresholding (IST). We will discuss different variations of iterative soft thresholding.

3.3.1 Iterative Soft-Thresholding. *Iterative Shrinkage-Thresholding Algorithm* (ISTA) has been studied in literature under various context and framework, see Daubechies et al. (2004); Blumensath and Davies (2008); Combettes et al. (2005). In particular, ISTA was proposed to handle a class of convex unconstrained optimization problems arising in image processing and other inverse linear problems. The algorithm is of first order and can be viewed as an extension of classical gradient algorithm. In practice, the ISTA algorithm is attractive due to its simplicity and it is adequate to solve large scale problems. The idea of the algorithm is to build at each iteration a regularization of the linearised differentiable function part in the objective. The linearised problem is much simpler to solve than the original problem.

ISTA considers the following general formulation of nondifferentiable unconstrained

optimization:

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := g(\boldsymbol{x}) + f(\boldsymbol{x}), \tag{3.3.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex continuously differentiable function and $g : \mathbb{R}^n \to \mathbb{R}$ is convex but non-differentiable. The following assumptions on the above formulation are made. Firstly, the function f is assumed to have Lipschitz continuous gradient ∇f. That is, there exists some Lipschitz constant $L > 0$ such that:

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{u})\| \leq L\|\boldsymbol{x} - \boldsymbol{u}\|, \quad \text{for every } \boldsymbol{x}, \boldsymbol{u} \in \mathbb{R}^n, \tag{3.3.2}$$

Beck and Teboulle (2009). The second assumption that brings success to this approach is that g must be separable. A function is separable if it can be written as a sum of functions of its arguments, that is $g(\boldsymbol{x}) = \sum g_i(x_i)$. It is easy to see that ℓ_1-norm is separable functions. The third assumption is that problem (3.3.1) is solvable, that is for an optimal set of solutions X^*, there exist $\boldsymbol{x}^* \in X^*$, such that $\boldsymbol{x}^*$ is the minimizer.

The main idea of the algorithm is to reduce problem (3.3.1) to a much simpler subproblem which can be solved efficiently. At the k-th iteration, the subproblem is a local quadratic approximation of the differentiation term:

$$\boldsymbol{x}^k = \arg\ \min_{\boldsymbol{x}} \left\{ g(\boldsymbol{x}) + f(x^{k-1}) + \nabla f(\boldsymbol{x}^{k-1})^T \left(\boldsymbol{x} - \boldsymbol{x}^{k-1}\right) + \frac{1}{2\alpha^{k-1}} \left\|\boldsymbol{x} - \boldsymbol{x}^{k-1}\right\|_2^2 \right\}. \tag{3.3.3}$$

The parameter $\alpha^{k-1} > 0$, which often acts as the step size in iterative algorithms, relates with Lipschitz constant, Beck and Teboulle (2009). Problem (3.3.3) is the direct results of applying the Taylor approximation to the differentiable term. We seek the optimal minimizer and not the function value. Thus ignoring the constant $f(\boldsymbol{x}^{k-1})$, we can add a constant like

$\|f(x^{k-1})\|^2$. Using this fact, equation (3.3.3) is simplified as

$$\begin{aligned}
\boldsymbol{x}^k &= \arg\ \min_{\boldsymbol{x}} \Bigg\{ g(\boldsymbol{x}) + f(x^{k-1}) + \boldsymbol{x}^T \nabla f(\boldsymbol{x}^{k-1}) - \left(\boldsymbol{x}^{k-1}\right)^T \nabla f(\boldsymbol{x}^{k-1}) + \dots \\
&\qquad \frac{1}{2\alpha^{k-1}} \left[\boldsymbol{x}^T\boldsymbol{x} - 2\boldsymbol{x}^T\boldsymbol{x}^{k-1} + \left(\boldsymbol{x}^{k-1}\right)^T \boldsymbol{x}^{k-1} \right] \Bigg\}, \\
&= \arg\ \min_{\boldsymbol{x}} \Bigg\{ g(\boldsymbol{x}) + \frac{1}{2\alpha^{k-1}} \boldsymbol{x}^T\boldsymbol{x} + \boldsymbol{x}^T \nabla f(\boldsymbol{x}^{k-1}) - \frac{1}{\alpha^{k-1}} \boldsymbol{x}^T\boldsymbol{x}^{k-1} + \dots \\
&\qquad f(x^{k-1}) - \left(\boldsymbol{x}^{k-1}\right)^T \nabla f(\boldsymbol{x}^{k-1}) + \frac{1}{2\alpha^{k-1}} \left(\boldsymbol{x}^{k-1}\right)^T \boldsymbol{x}^{k-1} \Bigg\}, \\
&\approx \arg\ \min_{\boldsymbol{x}} \Bigg\{ g(\boldsymbol{x}) + \frac{1}{2\alpha^{k-1}} \boldsymbol{x}^T\boldsymbol{x} - \frac{1}{\alpha^{k-1}} \boldsymbol{x}^T \left[\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right] + \dots \\
&\qquad \frac{1}{2\alpha^{k-1}} \left[\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right]^T \left[\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right] \Bigg\}, \\
&= \arg\ \min_{\boldsymbol{x}} \Bigg\{ g(\boldsymbol{x}) + \frac{1}{2\alpha^{k-1}} \left[\boldsymbol{x} - \left(\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right) \right]^T \left[\boldsymbol{x} - \left(\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right) \right] \Bigg\}.
\end{aligned} \tag{3.3.4}$$

In compressive sensing the non-differentiable term is given as $g(\boldsymbol{x}) = \mu\|\boldsymbol{x}\|_1$, where $\mu > 0$ is the regularisation parameter. The parameter μ controls the trade-off between sparsity and accuracy of reconstruction. Substituting, these functions into equation (3.3.4) the iteration reduces to:

$$\boldsymbol{x}^k = \arg\ \min_{\boldsymbol{x}} \left\{ \mu\|\boldsymbol{x}\|_1 + \frac{1}{2\alpha^{k-1}} \left\| \boldsymbol{x} - \left(\boldsymbol{x}^{k-1} + \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right) \right\|_2^2 \right\}. \tag{3.3.5}$$

The approach is very similar to the shrinkage operator which was discussed under the fixed-point method, see **Section 3.1.1**. In fact the minimization problem (3.3.5) takes the exact form of the proximal operator, for which there exist a close form solution. Hence the iterative procedure for solving sparse approximation problem is explicitly given as:

$$\begin{aligned}
\boldsymbol{x}^k &= \mathcal{S}\left(\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}), \mu\alpha^{k-1} \right), \\
&= sign\left(\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right) \max\left\{ \left| \boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}) \right| - \mu\alpha^{k-1}, \boldsymbol{0} \right\} \\
&= soft\left(\boldsymbol{x}^{k-1} - \alpha^{k-1} \nabla f(\boldsymbol{x}^{k-1}), \mu\alpha^{k-1} \right).
\end{aligned} \tag{3.3.6}$$

Here $\mathcal{S}$ is the shrinkage operator and it is termed $soft$ operator, thus the name *iterative soft-thresholding algorithm*. The computation of $soft$ operator is done component-wise. As mentioned before, the iteration is basically an extension of classical gradient methods. One takes the gradient descent step of the differentiable function, followed by soft-thresholding to induce sparsity on the solution.

The convergence properties of the algorithm depends upon the step size α^{k-1}, which is related to Lipschitz constant $L(f)$ of differentiable function f, in fact for fixed step size $\alpha^{k-1} = L(f)$. In sparse approximation problem the Lipschitz constant $L(f) = \frac{2}{\lambda_{max}(\boldsymbol{\Psi}^T\boldsymbol{\Psi})}$, where $\lambda_{max}(\cdot)$ is the maximum eigenvalue of the input matrix. In practice, it is often difficult to compute the Lipschitz constant explicitly, so backtracking line search is used. The backtracking starts with a large estimate of the stepsize α and iteratively shrinks it. One can initialize α using the Brazilai-Borwein strategy, Barzilai and Borwein (1988). The shrinking continues until the following condition holds:

$$F(\boldsymbol{x}^k) \leq \hat{F}(\boldsymbol{x}^k) \tag{3.3.7}$$

where $\boldsymbol{x}^k$ is given by equation (3.3.6). The algorithm is normally terminated when the relative change of the objective value or solution norm falls below some specified tolerance. **Algorithm 9** illustrates the algorithmic procedure to Iterative Soft-Thresholding.

Algorithm 9 Iterative Shrinkage-Thresholding Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m\times n}$, measurement vector $\boldsymbol{y}$, and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set $k = 0$. Set the initial solution $\boldsymbol{x}^0$.
2: *While* not convergence *do*
3: $k \leftarrow k + 1$.
4: *while* condition (3.3.7) is not true *do*
5: Update α^k using backtracking line search.
6: *endWhile*
7: Update the solution $\boldsymbol{x}^k$ via (3.3.6).
8: *endWhile*

Although it is well known that first order methods are practically efficient on large scale problems, ISTA is still considered to have slow convergence. In fact ISTA possesses a sublinear global rate of convergence satisfying:

$$F(\boldsymbol{x}^k) - F(\boldsymbol{x}^*) \leq L(f)\frac{\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2}{2k} \sim \mathcal{O}(1/k), \quad \text{for all } \boldsymbol{x}^* \in X^*, \tag{3.3.8}$$

see Beck and Teboulle (2009) for the derivations. Research works have been done to accelerate the rate of convergence of ISTA. The devised algorithm are aimed at improving the speed of ISTA, while retaining the computational effort and simplicity of ISTA. The Two Step ISTA (TwISTA) proposed

by Bioucas-Dias and Figueiredo (2007) speeds up ISTA by computing the next approximate solution based not only on the previous one, but on two previously computed approximate solutions. A more efficient method that uses a similar technique was proposed by Beck and Teboulle (2009) and the algorithm was named Fast Interactive Shrinkage-Thresholding (FISTA).

FISTA improves the convergence rate of ISTA from $\mathcal{O}(1/k)$ to $\mathcal{O}(1/k^2)$. The method preserves the computational simplicity of ISTA and it does not require any additional expensive effort at each iteration. Rather FISTA just uses an additional point that is smartly chosen and easy to compute. Let $\boldsymbol{z}^k$ be smartly chosen point, which is initiated as $\boldsymbol{z}^0 = \boldsymbol{x}^0$. For some parameter t^k, the k-th iteration of the algorithm is given as:

$$\boldsymbol{x}^k = soft\left(\boldsymbol{z}^{k-1} - \alpha^{k-1}\nabla f(\boldsymbol{z}^{k-1}), \mu\alpha^{k-1}\right), \tag{3.3.9}$$

$$t^k = \frac{1+\sqrt{1+(t^{k-1})^2}}{2}, \tag{3.3.10}$$

$$\boldsymbol{z}^k = \boldsymbol{x}^k + \left(\frac{t^{k-1}-1}{t^k}\right)(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}). \tag{3.3.11}$$

All other parameters and the algorithmic structure are exactly the same as that of ISTA. Hence the complete FISTA method is presented in **Algorithm 10**

Algorithm 10 Fast Iterative Shrinkage-Thresholding Algorithm .

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m\times n}$, measurement vector $\boldsymbol{y}$, and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

Initialization: Set $k = 0$. Set the initial solution $\boldsymbol{x}^0$.
While not convergence *do*
  $k \leftarrow k + 1$.
  while condition (3.3.7) is not true *do*
    Update α^k using backtracking line search.
  endWhile
  Update the solution $\boldsymbol{x}^k$ via (3.3.9).
  Update the parameter t^k via (3.3.10).
  Update the point $\boldsymbol{z}^k$ via (3.3.11).
endWhile

FISTA preserves computational simplicity of ISTA but with global rate of convergence given as:

$$F(\boldsymbol{x}^k) - F(\boldsymbol{x}^*) \leq 2L(f)\frac{\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2}{(k+1)^2} \sim \mathcal{O}(1/k^2), \quad \text{for all } \boldsymbol{x}^* \in X^*, \tag{3.3.12}$$

see Beck and Teboulle (2009) for the full derivation. The results are also demonstrated to be practically significantly better than those of ISTA, Beck and Teboulle (2009).

3.3.2 Iterative Hard Thresholding. Motivated by the development of Kingsbury (2001); Kingsbury and Reeves (2003), the Iterative Hard Thresholding (IHT) was independently introduced by Blumensath and Davies (2008, 2009) and Portilla (2009). IHT solves a traditional formulation of sparse approximation problem. The ℓ_0-regularization optimization problem is defined as follow

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mu\|\boldsymbol{x}\|_0 + \|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2^2. \tag{3.3.13}$$

This formulation is non-convex and it is known to be NP-hard in general. Therefore two common themes have been adopted to approximately solve the problem: greedy optimization strategies and relaxation of the objective function. Greedy strategies, such as Matching Pursuit are relatively fast iterative procedures that have been used extensively in practical applications. The performance of these methods is not guaranteed in general and only under very strict conditions they can be shown to optimise the above objective function. On the other hand relaxation methods replace the ℓ_0-norm constraint by an almost everywhere differentiable and often convex objective function. IHT adopts these themes in the following manner.

Instead of optimizing equation (3.3.13) directly, a surrogate objective function is introduced:

$$\hat{F}(\boldsymbol{x}, \boldsymbol{z}) := \mu\|\boldsymbol{x}\|_0 + \|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2^2 - \|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{\Psi}\boldsymbol{z}\|_2^2 + \|\boldsymbol{x} - \boldsymbol{z}\|_2^2. \tag{3.3.14}$$

The advantage of a surrogate objective function is that $F(\boldsymbol{x}) = \hat{F}(\boldsymbol{x}, \boldsymbol{x})$ and it can be written as:

$$\hat{F}(\boldsymbol{x}, \boldsymbol{z}) = \sum_{i=1}^{n} \left[x_i^2 - 2x_i\left(z_i + \boldsymbol{\psi}_i^T\boldsymbol{y} - \boldsymbol{\psi}_i^T\boldsymbol{\Psi}\boldsymbol{z}\right) + \mu|x_i|^0\right] + \|\boldsymbol{y}\|_2^2 + \|\boldsymbol{z}\|_2^2 - \|\boldsymbol{\Psi}\boldsymbol{z}\|_2^2, \tag{3.3.15}$$

where $|x_i|^0$ is one $x_i \neq 0$ and zero otherwise. The components of $\boldsymbol{x}$ are separated and the minimization of equation (3.3.15) can be done component wise. Two cases arise when the component x_i is zero and when it is non-zero. For simplicity the constant terms on a surrogate function is ignored. The constant term does not affect the solution, but it rather shifts the objective value up or down. In the first case $x_i = 0$, a surrogate objective function reduces to a constant. In

the second case $x_i \neq 0$, the minimization is achieved at $x_i^* = z_i + \boldsymbol{\psi}_i^T \boldsymbol{y} - \boldsymbol{\psi}_i^T \boldsymbol{\Psi} \boldsymbol{z}$. Thus the optimal objective value is,

$$\hat{F}(\boldsymbol{x}, \boldsymbol{z}) = \begin{cases} 0 & \text{if} \quad x_i = 0, \\ -\left(z_i + \boldsymbol{\psi}_i^T \boldsymbol{y} - \boldsymbol{\psi}_i^T \boldsymbol{\Psi} \boldsymbol{z}\right)^2 & \text{if} \quad x_i = z_i + \boldsymbol{\psi}_i^T \boldsymbol{y} - \boldsymbol{\psi}_i^T \boldsymbol{\Psi} \boldsymbol{z}. \end{cases} \tag{3.3.16}$$

We are interest is finding the optimal solution and hence the minimization of equation (3.3.15) is attained at

$$\boldsymbol{x} = H_{\sqrt{\mu}} \left(\boldsymbol{z} + \boldsymbol{\Psi}^T (\boldsymbol{y} - \boldsymbol{\Psi} \boldsymbol{z}) \right), \tag{3.3.17}$$

where H_τ is the component-wise hard thresholding operator defined as,

$$H_\tau(u) = \begin{cases} 0 & \text{if} \quad u \leq \tau, \\ u & \text{if} \quad u > \tau. \end{cases} \tag{3.3.18}$$

The minimum does not need to be unique whenever $x_i = z_i + \boldsymbol{\psi}_i^T \mathbf{y} - \boldsymbol{\psi}_i^T \boldsymbol{\Psi} \boldsymbol{z}$, however using the strict inequality in the definition of the thresholding operator guarantees a unique update, (Blumensath and Davies, 2008). Hence, starting from $\boldsymbol{x}^0 = \mathbf{0}$, the next iterate of iterative hard thresholding algorithms is defined as,

$$\boldsymbol{x}^{k+1} = H_{\sqrt{\mu}} \left(\boldsymbol{x}^k + \boldsymbol{\Psi}^T (\boldsymbol{y} - \boldsymbol{\Psi} \boldsymbol{x}^k) \right). \tag{3.3.19}$$

The iteration is due to the fact that at convergence $\boldsymbol{x}^{k+1} \approx \boldsymbol{x}^k$ and $F(\boldsymbol{x}^{k+1}) = \hat{F}(\boldsymbol{x}^{k+1}, \boldsymbol{x}^k)$. Blumensath and Davies (2008), have shown that the algorithm is guaranteed to converge whenever the inequality $0 < \lambda_{\max} \leq 1$ is satisfied, where $\lambda_{\max}$ is the maximum eigenvalue of $\boldsymbol{\Psi}^T \boldsymbol{\Psi}$. The iteration (3.3.19) is normally executed until a suitable termination condition is satisfied. A pseudo code of Iterative Hard Thresholding is shown in **Algorithm 11**.

Algorithm 11 Iterative Hard Thresholding Algorithm.

Input: An observed signal $\boldsymbol{y} \in \mathbb{R}^m$, a sensing matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$ and regularization parameter μ.

Output: An s-sparse coefficient vector $\boldsymbol{x}$.

1: *Initialization*: Set $k = 0$. Set the initial solution $\boldsymbol{x}^0 = \boldsymbol{0}$.

2: *While* not convergence *do*:

3: $k = k + 1$

4: Update solution via $\boldsymbol{x}^{k+1} = H_{\sqrt{\mu}}\left(\boldsymbol{x}^k + \boldsymbol{\Psi}^T(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{x}^k)\right)$.

5: *endWhile*

3.4 Combinatorial Algorithms

The major benefit of greedy algorithms is its computational efficiency in terms of speed, both theoretical and empirical. Combinatorial algorithms provide better performance, they are faster but impose very strict requirements on the measurement matrix. They have been shown to recover sparse vectors but with more measurements than ℓ_1-minimisation. These algorithms take advantage of highly structured matrices that support very fast signal reconstruction through group testing. Some of the combinatorial approaches reported in literature include Chaining Pursuit, Gilbert et al. (2006), and Heavy Hitters on Steroids (HHS), Gilbert et al. (2007), just to mention a few.

Many combinatorial methods employ the idea of group testing. Group testing arose during World War II, Dorfman (1943), in the context of blood testing for syphilis where very few patients had disease and it was too expensive to test every individual. The idea has been successfully adapted to compressive sensing to find a support of a sparse vector $\boldsymbol{x}$. The group testing problem supposes that a vector $\boldsymbol{x}$ has n elements, of which an unknown subset of s elements are anomalous and need to be identified. The goal is to design a collection of tests that will allow identification of the support of $\boldsymbol{x}$ while minimizing the number of tests performed. In simple practical setting, these tests are represented by a binary matrix $\boldsymbol{\Psi}$ whose entries $\boldsymbol{\Psi}_{ij}$ is one if and only if j^{th} item is used in the i^{th} test. If the output of the test is linear with respect to input, then the problem of recovering a vector $\boldsymbol{x}$ is equivalent to the standard sparse recovery problem.

3.5 Bayesian Algorithms

Under the Bayesian framework we depart from the conventional deterministic signal framework. Signals that have been considered up to this point are fixed. Bayesian framework assumes that the sparse signals arise from a known probability distribution. Theoretical work needed to discuss this type of algorithms digress from the conventional sparse recovery techniques discussed so far. No known algorithms are accompanied by guarantees on the number of measurements or the fidelity of signal reconstruction. There are no well-defined notion of error reconstruction. Different techniques reported in literature include, *sparse recovery via belief propagation*, Saravorham et al. (2008), *sparse Bayesian learning*, Tipping (2001) and *Bayesian compressed sensing*, Shihao et al. (2010).

We do not discuss the methods in details here. However, such methods provide insight into the development of recovery algorithms for rich class of signals and may be considered for practical interest.

Five algorithmic approaches for finding ℓ_1-norm solution of under-determined system of equations are discussed. Each approach has a set of advantages and disadvantages. We focused on three approaches. Convex algorithms provided strong theoretical results, but lacked the speed. Greedy algorithms provided speed, but lacked stability and theoretical guarantees such as restricted isomery property. Iterative shrinkage-thresholding algorithms bridges the gap between convex algorithms and greedy algorithms. Other approaches such as combinatorial algorithms and non-convex algorithms are of no interest to the ultimate goal of the dissertation.

Chapter 4

Hybrid Algorithm for ℓ_1-minimization

In this chapter, we propose a hybrid algorithm for finding ℓ_1-norm solution of under-determined linear algebraic systems. In a hybrid algorithm there is a choice between at least two distinct algorithms, each solving the same problem. The purpose is to combine the desirable features of each algorithm, so that the overall algorithm produces better performance than the individual algorithm. We propose a hybrid algorithm that combines important features of *fixed-point* scheme and *interior-point* scheme. Fixed-point scheme is based on shrinkage technique. It is an iterative first order method which is very efficient in identifying the support set of the sparse solution. Interior-point scheme is based on truncated Newtons method and it is very accurate. The proposed hybrid algorithm alternates between these optimization techniques in a probabilistic manner. The resulting combination is embedded under continuation to improve convergence of the overall algorithm.

The idea of hybrid algorithm is not new in compressed sensing. However the existing hybrid algorithms are not similar to the proposed hybrid algorithm. The existing algorithms by Wen et al. (2010) and Shi et al. (2010) are similar in structure. This kind of algorithm work in two stages that combines two types of optimization iterations. In the first stage, a shrinkage technique is used to estimate a subset of components of $\boldsymbol{x}$ that are likely to be non-zero in an optimal solution. The method is quick to estimate this set, but it takes large number of iterations before the optimal support set is reached. In the second stage, a smooth subspace optimization problem is solved. The subspace problem involves minimization of a smaller and smooth quadratic function. The quadratic function is formed by limiting components of $\boldsymbol{x}$ that correspond to the approximate support set

computed in the first stage. The technique balances the speed and accuracy of reconstruction. The proposed hybrid algorithm is different from this structure of hybrid algorithms presented by Wen et al. (2010) and Shi et al. (2010).

The proposed algorithm alternates between two optimization techniques in a probabilistic manner. This idea was introduced by Kaelo and Ali (2006) in the context of differential evolution algorithm. We adopt these idea to develop a new iterative algorithm for solving ℓ_1-minimization problem. In the proposed hybrid algorithm at the k-th iteration the fixed-point scheme is used with probability p^k and the interior-point scheme is used with probability $(1 - p^k)$. Depending on how a scheme preforms at each iteration, that particular scheme is rewarded for good performance or penalised for failure. The performance is decided on the basis of efficiency and effectiveness. The hybrid algorithm trends to prioritise a scheme that is fast and accurate. Figure 4.1, presents a schematic illustration of the proposed algorithm.

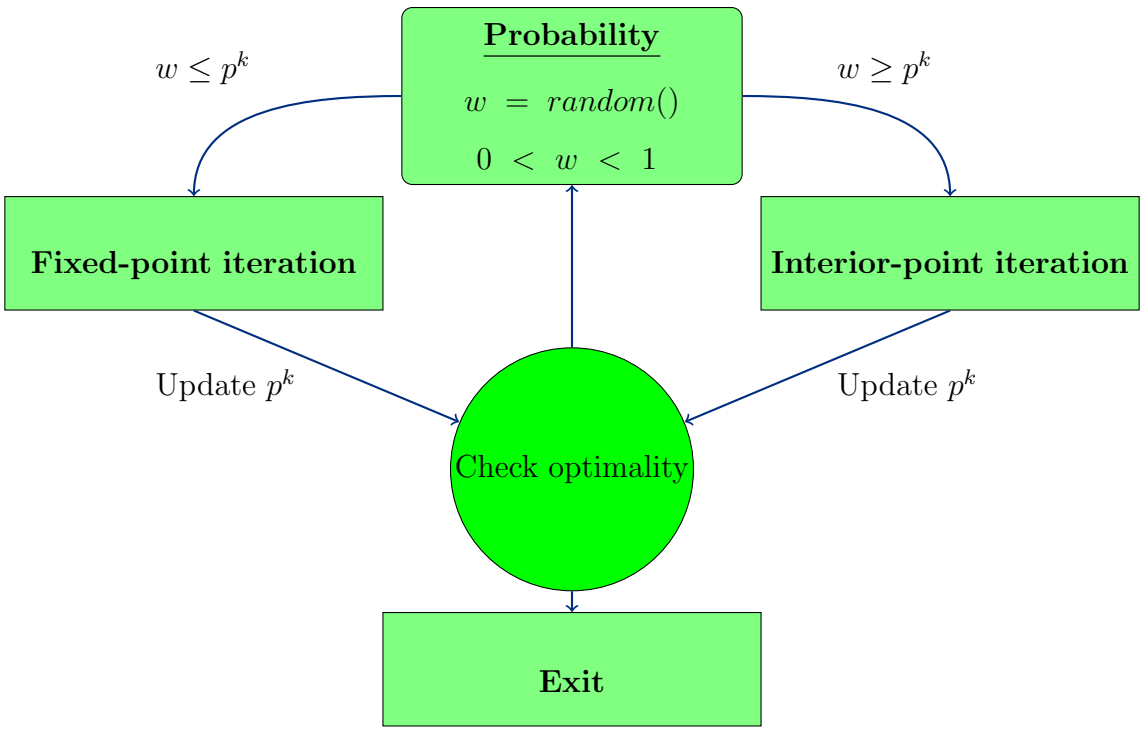


Figure 4.1: Schematic diagram of the probabilistic hybrid algorithm.

4.1 Review of Fixed-point and Interior-point for Hybrid Algorithm

This section reviews the fixed-point and interior-point iterations as they are implemented in the proposed probabilistic hybrid algorithm. The proposed hybrid algorithm solves the following general ℓ_1-norm optimization problem:

$$\arg \min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mu\|\boldsymbol{x}\|_1 + f(\boldsymbol{x}), \tag{4.1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable convex function and $\mu > 0$. The model (4.1.1), naturally extends to sparse approximation/compressive sensing problem. In which case $f(\boldsymbol{x}) = \|\boldsymbol{\Psi x} - \boldsymbol{y}\|_2^2$, for some sparse vector $\boldsymbol{x} \in \mathbb{R}^n$, and measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m\times n}$ designed to be incoherent with sparsity domain, see **Chapter 2**. Problem (4.1.1) always has a solution but it is not necessarily unique. A good algorithm must be able to recover the maximally sparse solution. A full discussion of fixed-point and interior-point algorithms is presented in **Chapter 3**, where we review important and useful results, and develop some insight of how to implement them in the hybrid algorithm.

4.1.1 Fixed-point iteration. A fixed point iteration implemented on the hybrid algorithm was proposed by Hale et al. (2008) as a fast algorithm for large scale ℓ_1-norm optimization problems. The algorithm uses an iterative shrinkage procedure to approximately solve problem (4.1.1). Specifically, the procedure iteratively computes:

$$\boldsymbol{x}^k = \mathcal{S}\left(\boldsymbol{x}^{k-1} - \alpha^{k-1}\nabla f(\boldsymbol{x}^{k-1}), \mu\alpha^{k-1}\right), \tag{4.1.2}$$

where the shrinkage operator is define as:

$$\mathcal{S}(x, \tau) = sign(x) \times \max\{|x| - \tau, 0\}. \tag{4.1.3}$$

Note that for a vector $\boldsymbol{x}$ the operation is evaluated component-wise. The fixed-point iteration is a first-order method which is basically a composition of two mappings. The first mapping is a gradient descent step, one takes a step proportional to the negative of the gradient of differentiable term $f(\boldsymbol{x})$. The results are passed on as input to the second mapping, the shrinkage step. In the shrinkage step, one reduces the ℓ_1-norm by shrinking the magnitude of each non-zero components in the input vector.

In the fixed-point method by Hale et al. (2008), the parameter α is fixed so that the fixed-point iteration is a contraction at every iteration. Since a bad choice of the value α usually slows down the rate of convergence, we implement the iteration with Barzilai-Borwein (BB) strategy, Barzilai and Borwein (1988). In the BB strategy, α^k is updated dynamically as:

$$\alpha^k = \frac{\left(\Delta \boldsymbol{x}^k\right)^T \Delta \boldsymbol{x}^k}{\left(\Delta \boldsymbol{x}^k\right)^T \Delta \boldsymbol{f}^k}, \tag{4.1.4}$$

where $\Delta \boldsymbol{x}^k = \boldsymbol{x}^k - \boldsymbol{x}^{k-1}$ and $\Delta \boldsymbol{f}^k = \nabla f(\boldsymbol{x}^k) - \nabla f(\boldsymbol{x}^{k-1})$. The BB is defined such that it mimics the Newton's method, and it significantly improves the performance of shrinkage. Fixed-point iteration based on shrinkage is very efficient in identifying the support set and is accurate for signals with small magnitude. Since $\mu\alpha^k$ is normally small, the shrinkage does not do well for signal with large magnitude.

We also incorporate a line search to improve the convergence of the algorithm. **Theorem 3.2** states the convergence results of the fixed-point algorithm. The theoretical convergence properties of the algorithm includes global convergence and the identification of support set after a finite number of iterations, Hale et al. (2008). A line search method is based on the properties of the search direction $\boldsymbol{d}^{k-1}$, determined by the shrinkage operator as:

$$\boldsymbol{d}^{k-1} = \boldsymbol{x}^{k+} - \boldsymbol{x}^{k-1}, \quad \text{where} \quad \boldsymbol{x}^{k+} = \mathcal{S}\left(\boldsymbol{x}^{k-1} - \alpha^{k-1}\nabla f(\boldsymbol{x}^{k-1}), \mu\alpha^{k-1}\right). \tag{4.1.5}$$

At each iteration one computes the step length s^{k-1}, and update the solution as:

$$\boldsymbol{x}^k = \boldsymbol{x}^{k-1} + s^{k-1}\boldsymbol{d}^{k-1}. \tag{4.1.6}$$

A sequence of step length candidates are identified and a decision is made to accept one when certain conditions hold. At each iteration we initialise the length as $s_0^{k-1} = 1$, and gradually decrease it until a sufficient decrease condition is met. In particular, we geometrically backtrack the step length, letting $s_l^{k-1} = \eta^l s_0^{k-1}$, for some $\eta \in (0,1)$ and $l = 0, 1, \ldots$, until the following Armijo's condtion is satisfied:

$$F(\boldsymbol{x}^{k-1} + s^{k-1}\boldsymbol{d}^{k-1}) \leq F(\boldsymbol{x}^{k-1}) + \hat{\rho}s^{k-1}\Pi^{k-1}. \tag{4.1.7}$$

Here $\hat{\rho} \in (0, \frac{1}{2})$ and for non-differentiable objective function Π^{k-1} is derived by Wen et al. (2010) as:

$$\Pi^{k-1} := \nabla f(x^{k-1})^T \boldsymbol{d}^{k-1} + \mu\|\boldsymbol{x}^{k+}\|_1 + \mu\|\boldsymbol{x}^{k-1}\|_1 \leq \boldsymbol{0}, \tag{4.1.8}$$

The Armijo condition for the line search stipulates that the step length s^{k-1} in the search direction $\boldsymbol{d}^{k-1}$ should produce a sufficient decrease of the objective function $F(\boldsymbol{x})$. The implementation of the line search is monotone, so the objective function is guaranteed to decrease at each iteration.

4.1.2 Interior-point iteration. The interior-point iteration implemented on the hybrid algorithm was proposed by Kim et al. (2007) as a fast algorithm for large scale ℓ_1-norm optimization problems. The algorithm reformulates problem (4.1.1) and uses primal interior-point method to approximately solve,

$$\min_{\boldsymbol{x},\boldsymbol{u}} \hat{F}(\boldsymbol{x},\boldsymbol{u}) := \mu \sum_{i}^{n} u_i + \|\boldsymbol{\Psi}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \rho(\boldsymbol{x},\boldsymbol{u}). \tag{4.1.9}$$

Here $\rho(\boldsymbol{x},\boldsymbol{u})$ is the logarithmic barrier function defined as

$$\rho(\boldsymbol{x},\boldsymbol{u}) = -\left[\sum_{i}^{n} log(u_i + x_i) + \sum_{i}^{n} log(u_i - x_i)\right]. \tag{4.1.10}$$

The dimension of the problem is doubled, with the restriction that $-u_i \leq x_i \leq u_i$. This constraint must be satisfied at all points of the algorithm. At the k-th iteration, before continuing with interior-point iteration we make an update $\boldsymbol{u}^k = \boldsymbol{x}^k$, whenever $\boldsymbol{x}^{k-1}$ was updated using fixed-point iteration. The primal barrier interior-point method updates the solution as:

$$\begin{bmatrix} \boldsymbol{x}^k \\ \boldsymbol{u}^k \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}^{k-1} \\ \boldsymbol{u}^{k-1} \end{bmatrix} + s^{k-1} \begin{bmatrix} \Delta\boldsymbol{x}^{k-1} \\ \Delta\boldsymbol{u}^{k-1} \end{bmatrix}. \tag{4.1.11}$$

Let us define the search direction as $\boldsymbol{d}^{k-1} = \left[\Delta\boldsymbol{x}^{k-1}; \Delta\boldsymbol{u}^{k-1}\right] \in \mathbb{R}^{2n}$. The search direction computed as the solution to the Newton system:

$$\nabla^2 \hat{F}(\boldsymbol{x}^{k-1}, \boldsymbol{u}^{k-1}) \boldsymbol{d}^{k-1} = -\nabla \hat{F}(\boldsymbol{x}^{k-1}, \boldsymbol{u}^{k-1}). \tag{4.1.12}$$

It is possible but computationally expensive to calculate the search direction in each iteration. We adopt the idea of Kim et al. (2007), that the search direction be computed as an approximate solution to the Newton system (4.1.12), using preconditioned conjugate gradient method. Kim et al. (2007) provides a detailed discussion of the preconditioned conjugate gradient algorithm.

The step length s^{k-1} is found using backtracking Armijo line search, similar to our discussion under fixed point iteration. The only exception is that, here we optimize a continuously differentiable function $\hat{F}(\boldsymbol{x},\boldsymbol{u})$, given by (4.1.9). Thus, the Armijo condition (4.1.7), must rather be satisfied with the quantity:

$$\Pi^{k-1} = \nabla \hat{F}(\boldsymbol{x}^{k-1}, \boldsymbol{u}^{k-1})^T \boldsymbol{d}^{k-1}. \tag{4.1.13}$$

4.2 Probabilistic Hybrid Algorithm

The probabilistic hybrid algorithm combines complementary strengths of fixed-point iteration and interior-point iteration. A fixed-point iteration is computationally fast and memory friendly but it is very slow for producing high accurate solution. It requires a large number of iterations before it reaches acceptable accuracy. On the other hand, interior-point iteration is very accurate, but it is computationally expense. In general, first order methods tend to struggle when the objective function is not well posed, while second order methods can easily find solution in such situation. The rationale for the proposed hybrid algorithm is to take advantage of the different computational strengths of the two methods and use them for optimal speed up while attaining high accuracy. Additionally, the algorithm should be able to solve a larger number of problems.

The probabilistic algorithm solves problem (4.1.1), using probability distribution over two distinct optimizations techniques. These optimization techniques are namely *fixed-point* iteration and *interior-point* iteration. Let $\mathcal{A}_1$ represent the updating of the solution using the fixed-point iteration and $\mathcal{A}_2$ be an updating using the interior-point iteration. At the k-th iteration the solution is approximated using either $\mathcal{A}_1$ or $\mathcal{A}_2$ depending on probability p^k. Initially, the two schemes $\mathcal{A}_1$ and $\mathcal{A}_2$ are equally likely to succeed. Thus, an initial probability of $p^0 = \frac{1}{2}$ is assigned to each scheme. This probability is updated according to some rules depending on computational the performance of the scheme. The performance measure is designed to measure both efficiency and effectiveness of the iteration. The detail of performance measure will be discussed shortly. A scheme is rewarded for being successful, thus we increase the probability of using that particular scheme. Otherwise a scheme is penalised for its failure, in which case we decrease the probability of using that particular scheme. This probabilistic adaptation guides the hybrid algorithm in deciding on which scheme is mostly used when updating the solution. This allows the algorithm to angle towards a scheme that solves a given problem in a most efficient and robust manner.

The two techniques $\mathcal{A}_1$ and $\mathcal{A}_2$ are combined so that the new iterate $\boldsymbol{x}^k$ is either generated with some probability p_k using $\mathcal{A}_1$ or with probability $q^k = 1-p^k$ using $\mathcal{A}_2$. If the solution $\boldsymbol{x}^k$ is updated, say using $\mathcal{A}_1$, and the process is deemed efficient and effective then the probability for using $\mathcal{A}_1$ is increased. The following rule is used for increasing the probability for $\mathcal{A}_1$,

$$p^k = p^{k-1} + \beta p^{k-1}(1 - p^{k-1}). \tag{4.2.1}$$

Here $\beta \in [0, 1]$ determines the rate at which probability p^k is changed. For simplicity the hybrid

algorithm is implemented with a fixed value of β. In a more sophisticated implementation, one can dynamically update β at each iteration. In that case β^k will be chosen to reflect the proportion of success of the scheme. If however the procedure for update $\boldsymbol{x}^k$ was inefficient then the probability of using $\mathcal{A}_1$ is decreased using,

$$p^k = p^{k-1} - \beta p^{k-1}(1 - p^{k-1}). \tag{4.2.2}$$

Updating the probability p^k of using $\mathcal{A}_1$, automatically changes the probability q^k of using $\mathcal{A}_2$ accordingly.

This adaptive procedure tends to let the algorithm decide for itself which scheme to use most in updating the solution for given specific problem. The algorithm tends to prioritise a scheme which is computationally efficient and robust so that it solves a given problem in a much efficient way. This is done by increasing p^k whenever $\mathcal{A}_1$ gives more favourable approximate solution, thus decreasing the probability for using $\mathcal{A}_2$. On the other hand, whenever $\mathcal{A}_2$ produces better results we increase q^k, thus decreasing probability of using $\mathcal{A}_1$. We force p^k, for $k \geq 1$, to stay in the range $[0.1, 0.9]$. In $p^k \notin [0.1, 0.9]$ we use clipping such that when $p^k \leq 0.1$ we set $p^k = 0.1$ and when $p^k \geq 0.9$ we set $p^k = 0.9$. The rationale behind this is to avoid a complete switch to one scheme. Combining the fixed-point iteration and interior-point iteration in this fashion, we get a complete hybrid algorithm as summarised in **Algorithm 12**.

Algorithm 12 Probabilistic Hybrid Algorithm.

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m\times n}$, measurement vector $\boldsymbol{y}$, and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

Initialization: Set iteration count $k = 0$ and probability $p^0 = \frac{1}{2}$. Set the initial solution $\boldsymbol{x}^0$.
While halting condition is not met *do*:
  Set $k = k + 1$.
  Generate a random number $\omega \in (0, 1)$.
  if $\omega \leq p^k$ *do*:
  ***** *Fixed-point iteration* *****
    Compute the shrinkage step size α^{k-1} by BB strategy (4.1.4).
    Update the approximate solution $\boldsymbol{x}^{k+} = \mathcal{S}\left(\boldsymbol{x}^{k-1} - \alpha^{k-1}\nabla f(\boldsymbol{x}^{\boldsymbol{k-1}}), \mu\alpha^{k-1}\right)$.
    Calculate the search direction $\boldsymbol{d}^{k-1} = \boldsymbol{x}^{k+} - \boldsymbol{x}^{k-1}$ and set $s^{k-1} = 1$.
    if Amijo condition (4.1.7) with (4.1.8) is not met *do*:
      Keep bracktraking $s^{k-1} = \eta s^{k-1}$
    endif
    Update the solution $\boldsymbol{x}^k = \boldsymbol{x}^{k-1} + s^{k-1}\boldsymbol{d}^{k-1}$.
  else if $\omega \geq p^k$ *do*:
  ***** *Interior-point iteration* *****
    Update $\boldsymbol{u}$ such that $-u_i \leq x_i \leq u_i$.
    Calculate the search direction $\boldsymbol{d}^{k-1}$ as approximate solution of Newton system, using preconditioned conjugate gradient method. Set $s^{k-1} = 1$.
    if Amijo condition (4.1.7) with (4.1.13) is not met *do*:
      Keep bracktraking $s^{k-1} = \eta s^{k-1}$
    endif
    Update the solution $\left[\boldsymbol{x}^k; \boldsymbol{u}^k\right] = \left[\boldsymbol{x}^{k-1}; \boldsymbol{u}^{k-1}\right] + s^{k-1}\boldsymbol{d}^{k-1}$.
  endif
  Evaluate the performance of the iteration.
  Update probability p^k via (4.2.1) or (4.2.2). Make sure $p^k \in [0.1, 0.9]$.
endWhile

4.2.1 Performance Evaluation. The success of the probabilistic hybrid algorithm depends on the effective choice of the performance evaluation of each scheme. An ideal performance measure would be a metric that can quantify the speed and accuracy of a given scheme at each iteration. It is very difficult to find such a matric. However, one can rather use convergence properties of the algorithm to evaluate the performance of the schemes. We adopt the following conditions based on the backtracking Armijo line search to reward or penalise the scheme:

- **Condition one:** Armijo line search parameters:

$$F(\boldsymbol{x}^{k-1} + s^{k-1}\boldsymbol{d}^k) \geq F(\boldsymbol{x}^{k-1}) + (1-\hat{\rho})s^{k-1}\Pi^{k-1}, \tag{4.2.3}$$

  where the variables and parameters have the usual meaning. The parameter Π^{k-1} is algorithm dependent, see equation (4.1.8) and (4.1.13) for the representation of the parameter for fixed-point algorithm and interior-point algorithm respectively. The justification of the condition is that we want to optimally select the step length s^{k-1}. A too small or too large step length does not provide a sufficient decrease of the objective value. Armijo line search avoids large step length. Using this condition, we try to avoid a selection of small step size.

- **Condition two:** Goldstein condition

$$\boldsymbol{d}^k \nabla f(\boldsymbol{x}^k) \geq \sigma \boldsymbol{d}^k \nabla f(\boldsymbol{x}^{k-1}), \tag{4.2.4}$$

  where $f(\boldsymbol{x})$ is the the differentiable term of the objective function. Here $\sigma \in (\rho, 1)$.

- **Condition three:** Relative change of the objective function:

$$\left|F(\boldsymbol{x}^k) - F(\boldsymbol{x}^{k-1})\right| \leq \varepsilon_f \max\left\{\left|F(\boldsymbol{x}^k)\right|, \left|F(\boldsymbol{x}^{k-1})\right|, 1\right\}, \tag{4.2.5}$$

  where $\varepsilon_f > 0$, is the tolerance value. The justification of the condition is that we want to avoid slow convergence of the algorithm. If it happens that the algorithm cannot make sufficient progress, then the relative change of the objective function over two consecutive iterations is usually small. Hence, stratification of this condition implies the algorithm is stagnating.

The scheme is considered to be successful if all the conditions holds true. In that manner, a scheme is rewarded by increasing the probability of using that particular scheme. Otherwise if any of the condition is violated, a scheme is penalised. The probability of using that particular scheme is decreased. Hence the hybrid algorithm tends to prioritise the scheme that converges fast.

4.3 Continuation Strategy

The regularization parameter $\mu > 0$ controls the trade-off between sparsity and accuracy of the method that solve the ℓ_1-minimization problem:

$$\arg \min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mu\|\boldsymbol{x}\|_1 + f(\boldsymbol{x}). \tag{4.3.1}$$

Moreover, Hale et al. (2008) have shown that the convergence property of the fixed-point iteration is affected by μ. Thus a good choice of the parameter is very important. However, it is practically difficult to choose μ, so as to ensure optimal performance of the algorithm. The solution to problem (4.3.1) is typically sparse and as μ decreases, its tends to be sparser. In the continuation strategy, the ℓ_1-minimization problem (4.3.1) is solved for decreasing values of μ. The idea is related to the homotopy algorithm in statistics and the strategy has been empirically shown to be successful when applied to ℓ_1-minimization problem, see Hale et al. (2008); Wright et al. (2009); Figueiredo et al. (2007).

Given the regularization path such that $\mu^1 > \mu^2 > \cdots > \mu^L = \mu$, the continuation procedure solves problem (4.3.1) for each μ^k. The solution obtained from a previous subpath associated with μ^{k-1} is used as the initial estimate solution of the next problem associated with μ^k. One of the challenges in using continuation is to choose the regularization path that leads to fastest global running time. We embed the hybrid algorithm in a continuation procedure by utilising the adoptive continuation proposed by Wright et al. (2009). Using the fact that a value of μ too small slows down the algorithm and a value too large generates trivial zero solution. In fact for $f(\boldsymbol{x}) = \|\boldsymbol{\Psi x} - \boldsymbol{y}\|_2^2$, any $\mu \geq \|\boldsymbol{\Psi}^T\boldsymbol{y}\|_\infty$ the unique solution to problem (4.3.1) is the zero vector. The condition can be easily obtained from the optimality condition of ℓ_1-minimisation problem (4.3.1). Inspired by this fact, Wright et al. (2009) proposed adaptive continuation procedure presented in **Algorithm 13**. Numerical results in the next chapter are produced without implementing continuation strategy.

Algorithm 13 Probabilistic Hybrid Algorithm - Continuation .

Input: A measurement matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m \times n}$, measurement vector $\boldsymbol{y}$, and μ.

Output: An s-sparse approximate solution $\boldsymbol{x}$.

1: *Initialization*: Set iteration count $k = 0$. Set the initial solution $\boldsymbol{x}^0$ and set $\boldsymbol{r}^0 = \boldsymbol{y}$.

2: *while* $\mu^k > \mu$ *do*:

3: Set $k = k + 1$.

4: $\mu^k = \max\{\xi \|\boldsymbol{\Psi}^T \boldsymbol{r}^{k-1}\|_\infty, \mu\}$, where $\xi \in (0, 1)$.

5: Update solution using probabilist hybrid algorithm with input $\boldsymbol{\Psi}$, $\boldsymbol{y}$, μ^k and set the initial solution as $\boldsymbol{x}^{k-1}$.

6: $\boldsymbol{r}^k = \boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{x}^k$.

7: *endWhile*

The fixed-point algorithm and interior-point algorithm are combine to form a hybrid algorithm. Fixed-point was chosen for its speed while interior-point was recognized for its high accuracy. The algorithms were combined in a probabilistic manner, at each iteration the algorithm that performs well is rewarded, otherwise penalised. The most challenging task is the design of the performance evaluation to determine the effectiveness and efficiency at each iteration. An ideal performance measure is metric that can quantify the speed and accuracy of a given scheme at each iteration. Here a metric based on the convergence of line search is developed.

Chapter 5

Computational Experiments

This chapter presents numerical results, which illustrate the computational performance of the hybrid algorithm. The computational performance is carried out by testing the reconstruction ability, speed and accuracy of the hybrid algorithm. Computational results are produced without implementation of the continuation strategy. The performance is compared with some of the existing algorithms on selected applications. The investigations listed here pertain to small sample studies on synthetic data. All results were obtained using: *Matlab R2013a, Intel Core i5, 3.4 GHz processor with 8 GB memory running on Linux operating system*

5.1 Comparison of Hybrid Algorithm with Constituent Algorithms

In this section, we carry out a numerical comparison of the proposed hybrid algorithm with its constituent algorithms: the fixed-point algorithm and interior-point algorithm. We conducted the experiments with our implementation of these algorithms. We investigate the quality of reconstruction, the speed and accuracy of reconstruction of the algorithms and comment on the probability profile of the hybrid algorithm.

5.1.1 Quality of Reconstruction. We investigate how well the proposed hybrid algorithm recovers the sparse solution. Specifically, we study reconstruction ability of an hybrid algorithm and

compare its performance with its constituent algorithms. To achieve this we use phase transition analysis. Phase transition analysis provides a way of studying the equivalence of ℓ_1 and ℓ_0, by indicating how sparsity and number of measurements affect the success of an algorithm that solves the ℓ_1-minimization problem. Let $\delta = \frac{m}{n}$, represent a normalized measure of indeterminacy factor and $\rho = \frac{s}{m}$, represent a normalised measure of sparsity. A typical phase plot will depict some performance statistics as a function of δ and ρ, over a phase space $(\delta, \rho) \in [0,1] \times [0,1]$. Each pair (δ, ρ) in the space indicates the difficulty of reconstruction.

Experimental setup 1: We use simulated data. The original signal $\boldsymbol{x}_0$, was generated to be a random vector of size n containing s non-zero components. These non-zero components were generated randomly from the standard normal distribution and placed at random locations within the signal vector. Although numerous other signals could be used for testing, and undoubtedly the performance of algorithms would differ from signal to signal. The main objective of the empirical results is to serve only as a guideline to their performance. The measurement matrix $\boldsymbol{\Psi}$ of size $m \times n$ was generated randomly in accordance to Gaussian entries. That is the entries of $\boldsymbol{\Psi}$ were chosen from the normal distribution with zero mean and variance of $1/m$.

In practice, real signals are corrupted with noise. So to make the results relevant and realistic, both the original signal $\boldsymbol{x}_0$ and measurements $\boldsymbol{y}$ were contaminated with additive Gaussian noise. The measure were generated as:

$$\boldsymbol{y} = \boldsymbol{\Psi}\left(\boldsymbol{x}_0 + \boldsymbol{e}_x\right) + \boldsymbol{e}_y, \tag{5.1.1}$$

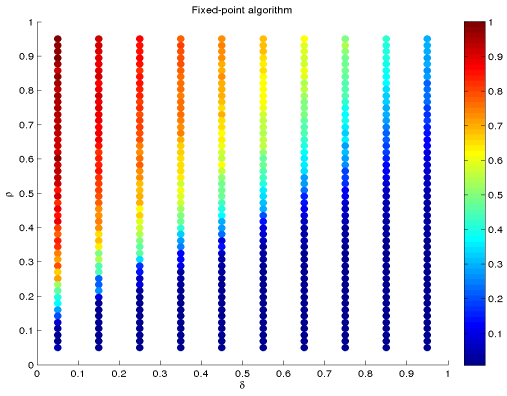
where $\boldsymbol{e}_x$ and $\boldsymbol{e}_y$ are signal and measurement noise vectors respectively. The elements of noise vectors were generated randomly from independent and identical normal distributions. Two instances of measurement were used: noiseless measurement, where $\boldsymbol{e}_x = \boldsymbol{e}_y = 0$ and noise measurements for which the components of noise vectors $\boldsymbol{e}_x$ and $\boldsymbol{e}_y$ were drawn from the normal distribution of $\mathcal{N}(0, 10^{-5})$ and $\mathcal{N}(0, 10^{-3})$ respectively.

The input of the algorithm was the measurements $\boldsymbol{y}$, the measurement matrix $\boldsymbol{\Psi}$ and other parameters of the algorithms. The algorithms had to return the optimal solution $\boldsymbol{x}^*$, which must be the best approximation of the original signal $\boldsymbol{x}_0$. The quality of reconstruction was determined via the root mean squared error between the original signal $\boldsymbol{x}_0$ and recovered signal $\boldsymbol{x}^*$:

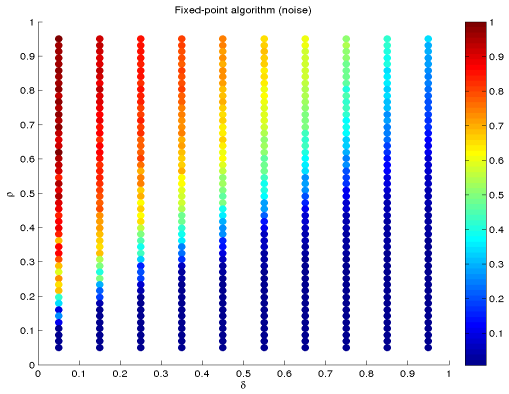
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[(x_0)_i - x_i^*\right]^2}. \tag{5.1.2}$$

The use of $RSME$ makes a good general purpose error metric for numerical predictions.

The phase diagram is formed by a grid of δ-ρ values, with δ, ρ arranged through 50 and 10 equispaced points respectively in the interval $(0, 1)$. Each point (δ, ρ) in the phase plot represents the average of root mean square error over 50 trails. The experiment was carried out on a small scale signal with fixed length $n = 512$ and the regularization parameter was fixed at $\mu = 0.01$. For each configuration, the algorithms were terminated when the relative change of the estimate solution falls below 10^{-5}, that is $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|/\|\boldsymbol{x}^{k-1}\| \leq 10^{-5}$. Otherwise the algorithms were forced to terminate when they reach maximum iterations of 1000. The same stopping condition was used for all algorithms. The results in this section were obtained by running our implementation of the three algorithms: fixed-point algorithm, interior-point algorithm and the combination through the probabilistic hybrid algorithm.

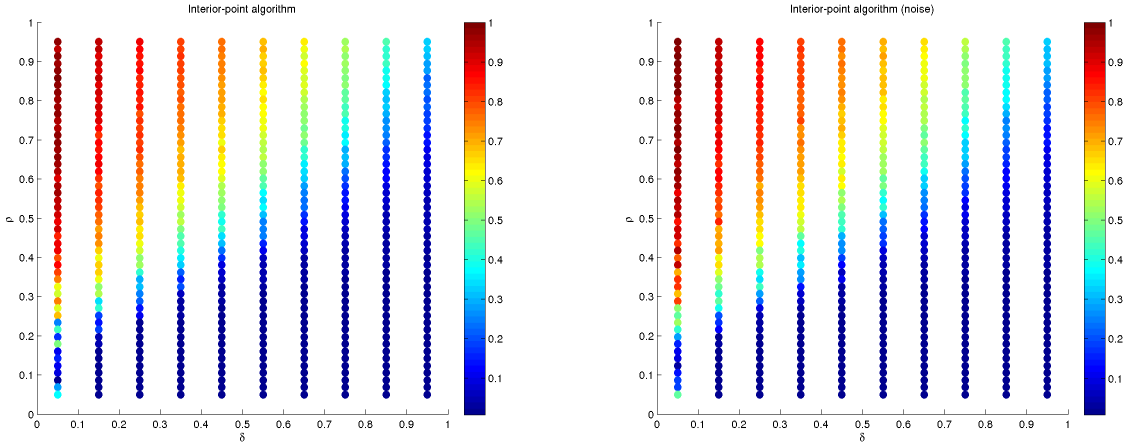


(a) Fixed-point algorithm noiseless measurements



(b) Fixed-point algorithm noisy measurements

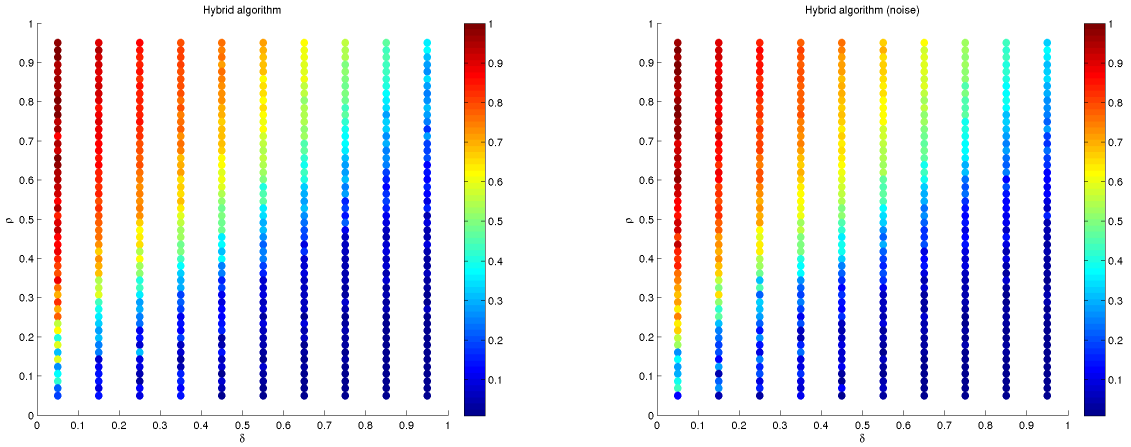
Figure 5.1: Phase diagrams illustrating reconstruction ability of fixed-point algorithm: (a) signal reconstruction from noiseless measurements and (b) signal reconstruction from noisy measurements.



(a) Interior-point algorithm noiseless measurements

(b) Interior-point algorithm noisy measurements

Figure 5.2: Phase diagrams illustrating reconstruction ability of interior-point algorithm: (a) signal reconstruction from noiseless measurements and (b) signal reconstruction from noisy measurements.



(a) Probabilistic hybrid algorithm noiseless measurements

(b) Probabilistic hybrid algorithm noisy measurements

Figure 5.3: Phase diagrams illustrating reconstruction ability of probabilistic hybrid algorithm: (a) signal reconstruction from noiseless measurements and (b) signal reconstruction from noisy measurements.

Discussion: Figures 5.1, 5.2 and 5.3, demonstrate the reconstruction ability of fixed-point algorithm, interior-point algorithm and probabilistic hybrid algorithm respectively. All the phase diagrams show the expected behaviour. We observe a highly accurate reconstruction for small ρ and large δ. This correspond to low sparsity level with large number of measurements. Conversely, when the signal is not sparse and few measurements taken we witness the failure of reconstruction which corresponds to large ρ and small δ. The original signal is recovered well as long as it is sparse enough and there are enough measurements.

The blue region of the phase diagrams indicates excellent reconstruction of sparse signals, while the red region represents poor reconstruction. The transition line is indicated by the ends of the blue region. All the points below the transition curve indicate the equivalence of ℓ_1-norm and ℓ_0-norm reconstruction. The transition from success to failure occurs at different ρ for different δ. The location of the transition curve depends on the recovery algorithm. The presence of noise does not have much influence on the qualitative shape of the transition curve. All the three algorithm-phase diagrams seen to exhibit no observable difference. The proposed probabilist hybrid algorithm has the same reconstruction ability as those of fixed-point algorithm and interior-point algorithm.

From the above experiment we were able to determine default value of parameters for the probabilistic hybrid algorithm. The values were tuned to produce good efficiency and effectiveness, to achieve the best possible phase diagram. Table 5.1 provides a summary of the parameters and their default values chosen for the hybrid algorithm.

Parameter	Default value
β	The factor for changing probability $\beta = 0.5$
η	The factor for decreasing step size in backtracking line search $\eta = 0.5$ (see equation (4.1.7))
$\hat{\rho}$	Armijo's condition parameter $\hat{\rho} = 0.003$
σ	The parameter for Goldstein condition $\sigma = 0.04$
ε_f	Tolerance for relative objective value $\varepsilon_f = 10^{-2}$

Table 5.1: Parameters and their default values for hybrid algorithm.

5.1.2 Speed and Accuracy. In this section, numerical experiments were carried out to study speed and accuracy of the proposed hybrid algorithm. Moreover, we study how scaling of problem size affects the speed and accuracy. We compare the efficiency of the hybrid algorithm with fixed-point algorithm and interior-point algorithm.

Experimental setup 2: The data was generated as in experiment 1 with measurements contaminated with noise. The speed and accuracy were determined for different problem sizes. We increase the signal length in the base 2 power from 2^8 to 2^{13}. In all problem sizes the indeterminacy ratio was fixed at $\delta = \frac{m}{n} = 0.4$ and sparsity ratio at $\rho = \frac{s}{m} = 2$. The regularization parameter was set to $\mu = 0.01$. All the algorithms were run until the same convergence criteria were met, where the relative change of the solution norm falls below the tolerance value of 10^{-5}. Experiments for each dimension were carried out on 50 different sets of random data. We compare the means and variances of the computation time for each dimension. We also evaluate the accuracy of the solution by looking at the reconstruction error. The error was computed using mean squared error between the true signal and reconstructed signal.

n	Fixed-point		Interior-point		Hybrid	
	mean	variance	mean	variance	mean	variance
2^8	0.0490	0.0023	0.1908	0.0033	0.1144	0.0028
2^9	0.0780	0.0017	0.2828	0.0036	0.1592	0.0039
2^{10}	0.1822	0.0036	0.5574	0.0053	0.2922	0.0099
2^{11}	1.4484	0.0172	3.3868	0.0204	1.8632	0.0508
2^{12}	6.8610	0.0788	14.8048	0.1195	7.7110	0.2143
2^{13}	30.6888	0.2687	72.0256	0.4927	31.1426	1.0573

Table 5.2: Speed comparison for different problem size. The computation time is reported in seconds.

n	Fixed-point	Interior-point	Hybrid
2^8	0.0023	0.0013	0.0036
2^9	0.0026	0.0014	0.0036
2^{10}	0.0022	0.0015	0.0034
2^{11}	0.0025	0.0015	0.0032
2^{12}	0.0027	0.0015	0.0039
2^{13}	0.0023	0.0015	0.0030

Table 5.3: Comparison of solution accuracy. Root square mean error between the true signal and reconstructed solution.

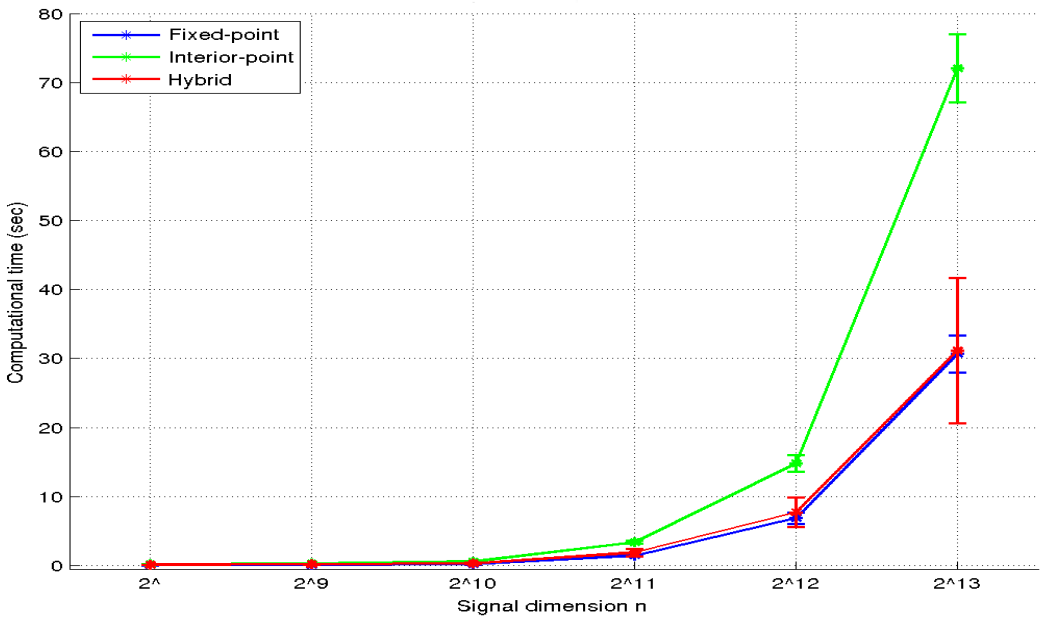


Figure 5.4: Speed comparison for different problem sizes

Discussion: Table 5.2 presents a summary of computational speed for the fixed-point, interior-point and hybrid algorithms. The fixed-point algorithm appears to be more efficient as it achieved less computational time in all dimensions, followed by the hybrid algorithm. Figure 5.4, plots the computational time as a function of dimension for better illustration of results. It can be clearly seen that interior-point (green) algorithm scales poor as compared to the fixed-point(blue) algorithm and hybrid (red) algorithm. A possible reason for the scaling is that interior-point

algorithm doubles the size of the problem, which for large problem this becomes a very serious issue in practice. It is noteworthy that the hybrid algorithm achieves comparative efficiency compared to fixed point algorithm, an algorithm that is recognized for its good efficiency.

We also evaluated the accuracy of the reconstructed solution of the algorithms, the results are summarized if Table 5.3. The accuracy was measured as the root mean square error between the true signal and reconstructed signal, see equation (5.1.2). The interior-point algorithm achieves better accuracy, followed by fixed-point algorithm then the hybrid algorithm. All the algorithm achieve similar order of accuracy of 10^{-3}. Figures 5.5, 5.6 and 5.7, provide visual examples of the reconstruction of fixed-point algorithm, interior-point algorithm and probabilistic hybrid algorithm, respectively. Hence, the hybrid algorithm is comparative to fixed-point algorithm in terms of speed and comparative to both fixed-point and interior-point algorithm in terms of accuracy. The results are also supported by the phase diagrams. All the algorithms achieves similar phase diagrams.

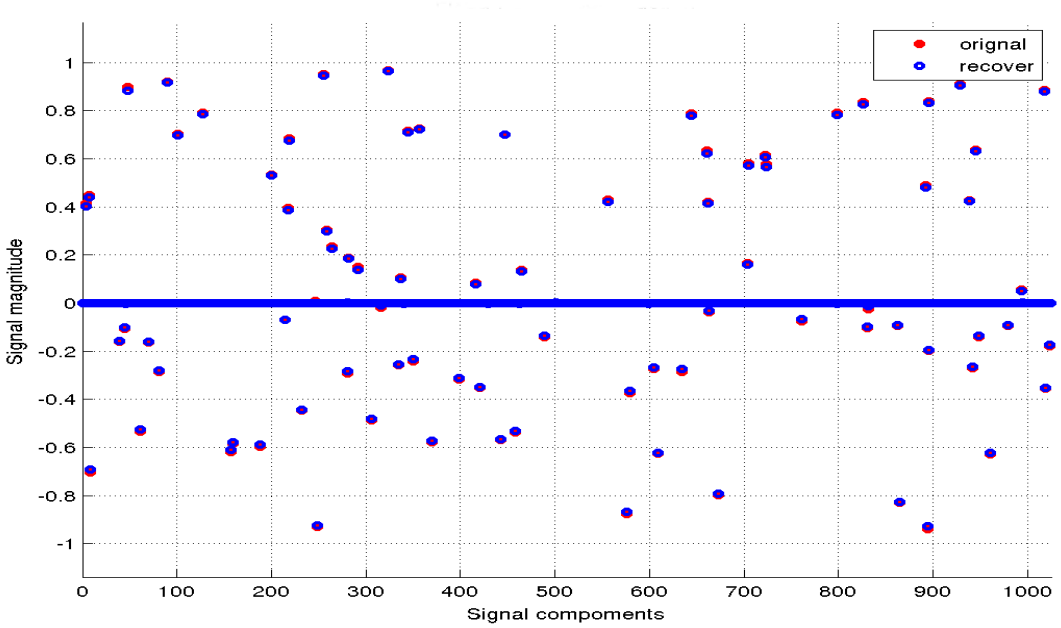


Figure 5.5: An example of signal reconstruction with fixed-point algorithm.

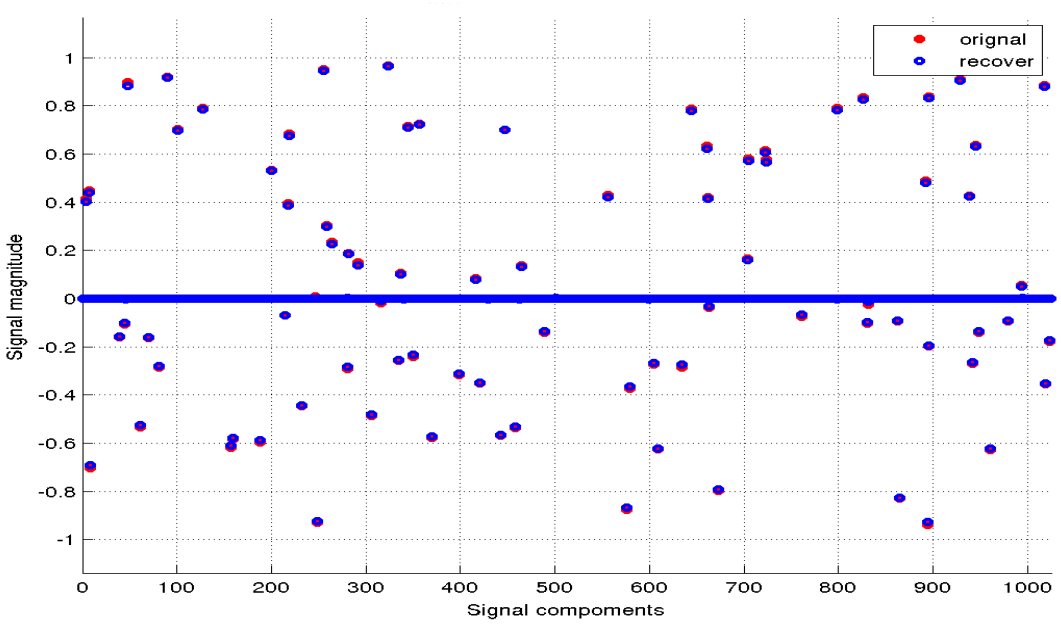


Figure 5.6: An example of signal reconstruction with interior point algorithm.

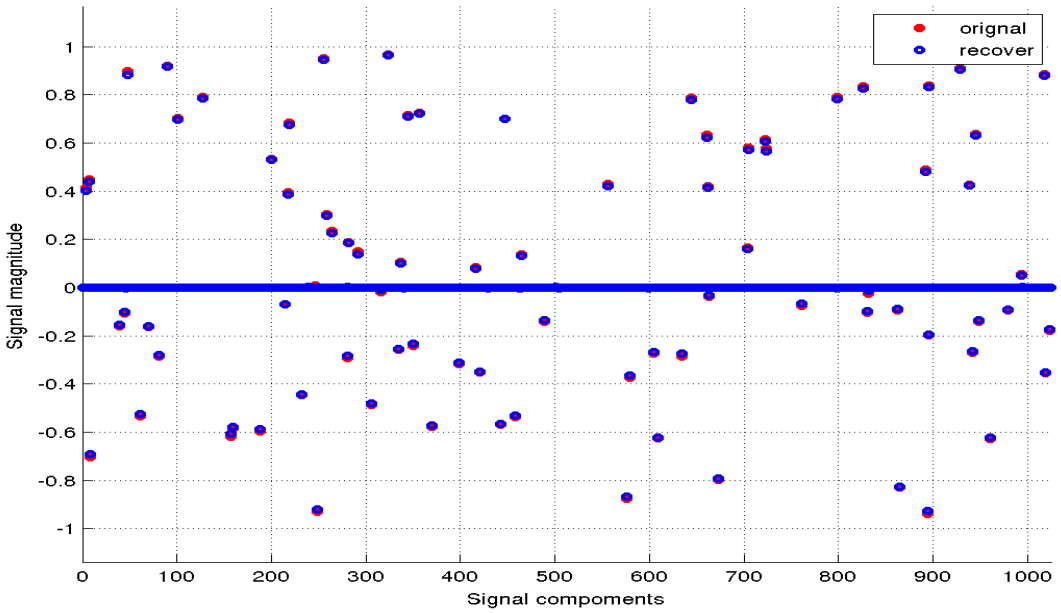


Figure 5.7: An example of signal reconstruction with probabilistic hybrid algorithm.

5.1.3 Probability Profile. Here we investigate the behaviour of probability with the evolution iterations. The behaviour of probability attained would provide some insight into how the performance evaluation conditions are met. It should also indicate how algorithms performs for a particular problem. From the results in **experimental setup 2**, we randomly picked some probability profiles for different dimensions. Figures 5.8, 5.9 and 5.10, present the random selection of probability profiles.

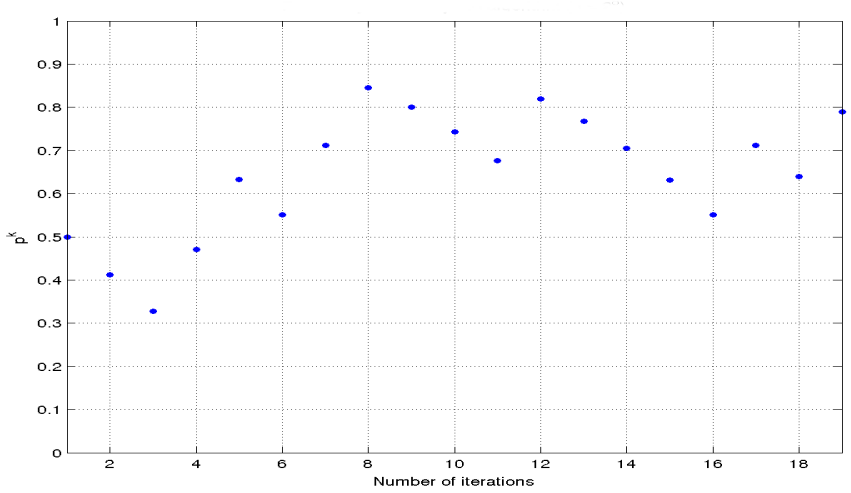


Figure 5.8: Probability profile of reconstruction for signal length $n = 2^8$.

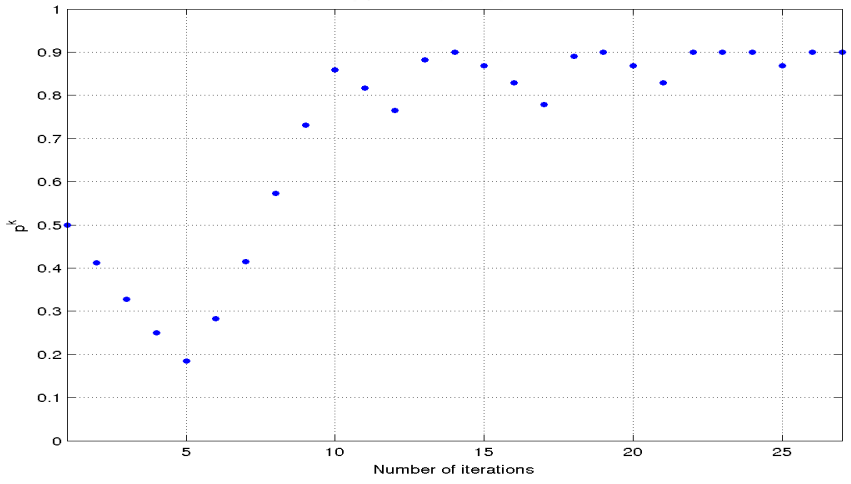


Figure 5.9: Probability profile of reconstruction for signal length $n = 2^{10}$.

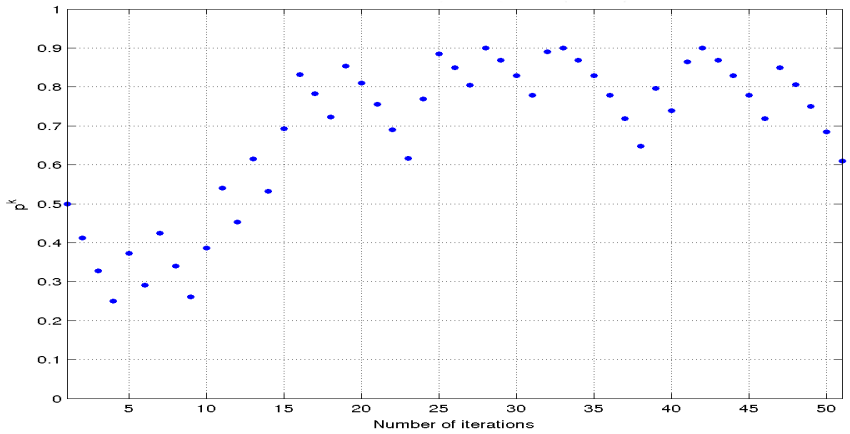


Figure 5.10: Probability profile of reconstruction for signal length $n = 2^{13}$.

Discussion: The evaluation favours the fixed-point algorithm more often than interior-point algorithm. In general the probabilistic hybrid algorithm tends to prioritise the fixed-point algorithm, at least for this specific problem, where the measurement matrix is Gaussian. This explains why the hybrid algorithm scales in a similar fashion with fixed-point algorithm, see Figure 5.4.

5.2 Comparison of Hybrid Algorithm with Existing Algorithms

This section, presents the comparison of computational performance of the proposed probabilistic hybrid algorithm with several existing algorithms in literature. We compare the hybrid algorithm with second order Truncated Newtons Interior Point method ($\ell_1 - \ell_2$) by Kim et al. (2007), first order Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) by Beck and Teboulle (2009), and a greedy Compressive Sampling Matching Pursuit (CoSaMP) by Needell and Tropp (2009). All these algorithms were discussed in **Chapter 3** and each algorithm represents one of the three algorithmic categories we focused on. The algorithms are recognized for their speed and high accuracy. The Matlab codes used here were implementation by the respective authors.

The setup was similar to that of **Experimental setup 1**. Here consider only a small scale problem. The original signal $\boldsymbol{x}_0$, was generated to be random vector of size $n = 512$ containing

$s = 32$ non-zero components. These non-zero components were generated randomly from standard normal distribution and placed at random locations within the signal vector. The number of measurements were set to $m = 128$. The measurements were generated via equation (5.1.1), with no noise. We tested the recoverability on two types of signal values. One which resembles low signal values. The signal values were randomly selected in the range [-1, 1]. In the second problem, the signal resembled high signal values, the values randomly generated with order of magnitude of 10. Numerous other signals could be used for testing, and undoubtedly the performance of algorithms would differ from signal to signal

All the algorithms were run until the same convergence criteria were met, where the relative change of the solution norm falls below the tolerance value of 10^{-5}. Experiments for each dimension were carried out on 50 different sets of random data. In all the experiments the regularization parameter was set to $\mu = 0.001$. All the other algorithm dependent parameters were set as suggested by the respective authors. Note that CoSaMP does not use μ and the algorithm requires the estimate of sparsity level as input. We have provided the algorithm the exact sparsity. Table 5.4 reports the mean values of computation time, relative error computed as $\|\boldsymbol{x}^* - \boldsymbol{x}_0\|/\|\boldsymbol{x}_0\|$, the ℓ_1-norm of the solution and the norm of the residual $\|\boldsymbol{\Psi}\boldsymbol{x}^* - \boldsymbol{y}\|_2$.

Problem	Algorithm	Time	rel. error	$\|\boldsymbol{x}^*\|_1$	$\|\boldsymbol{\Psi}\boldsymbol{x}^* - \boldsymbol{y}\|_2$
small	Hybrid	0.0732	3.1250e-03	1.5065e+01	4.5459e-03
$\boldsymbol{x}$	$\ell_1 - \ell_s$	0.7419	1.8632e-03	1.5026e+01	3.6663e-03
values	CoSaMP	0.0713	7.1880e-06	1.5025e+01	7.4354e-04
	FISTA	0.0193	2.6406e-03	1.5011e+01	2.3877e-03
large	Hybrid	0.0843	5.3950e-03	4.8811e+02	4.9803e-03
$\boldsymbol{x}$	$\ell_1 - \ell_s$	0.8328	1.1524e-03	4.8636e+02	4.1502e-03
values	CoSaMP	0.0696	8.1180e-06	4.8671e+02	2.6267e-04
	FISTA	0.0175	8.9833e-03	4.8653e+02	2.7107e-03

Table 5.4: Comparison of recoverability of hybrid algorithm with other existing algorithms.

Discussion: Overall the difference in signal values does not have any significant influence in the result, except in the ℓ_1-norm. The greedy algorithm, CoSaMP is superior in terms of accuracy and FISTA is superior in terms of speed. The hybrid algorithm is comparable to CoSaMP in terms of speed and achieves similar accuracy as both $\ell_1 - \ell_s$ and FISTA.

A set of computational experiments are carried to demonstrate the performance ability of the hybrid algorithm. The hybrid algorithm is shown to have similar reconstruction ability as fixed-point algorithm and interior algorithm. With no good performance evaluation the algorithm tends to prioritize the fixed-point algorithm. The hybrid algorithm was numerically shown to be comparable with other existing algorithms in terms speed and accuracy.

Chapter 6

Conclusion

Compressive sensing is a new and fast growing field of applied mathematics that addresses the shortcomings of conventional signal compression. With the knowledge that a given signal has few non-zero components relative to its dimension, compressive sensing seeks to reconstruct the signal from few nonadaptive linear measurements. This is equivalent to finding ℓ_1-norm solution of under-determined linear algebraic system of equations.

Fundamental theory of compressive sensing was reviewed, in which it became clear that it is possible to accurately reconstruct a signal from the under-determined linear system. Compressive sensing relies on two principles: sparsity, which pertains to the signals of interest, and incoherence, which pertains to the sampling of the signal. Sparsity expresses the idea that information contained in the signal may be much smaller than the dimension of the signal. Incoherence extends the idea that signal having sparse representation in some domain must be spread out in the measurement domain. Thus one needs to clearly distinguish the source of information. Not any measurement matrix is good for compressive sensing, but through restricted isometry property one can design a good measurement matrix that can capture all the information about the signal. Gaussian matrix does well in compressive sensing.

A survey of algorithmic approaches for finding ℓ_1-norm solution of under-determined system of equations was carried out. As work in this area developed, five algorithmic approaches to the problem emerged, each with its own set of advantages and disadvantages. We focused on three approaches. Convex algorithms provide strong theoretical results, but lack the speed.

Greedy algorithms provide speed, but lack stability and theoretical guarantees such as restricted isometry property. Iterative shrinkage-thresholding algorithms bridges the gap between convex algorithms and greedy algorithms. These algorithms are easy to implement, they are fast and they provide theoretical guarantees. Other approaches such as combinatorial algorithms and non-convex algorithms were of no interest to the ultimate goal of the dissertation.

The goal of the dissertation was achieved by combining two convex algorithms, the fixed-point algorithm and interior-point algorithm. Fixed-point was chosen for its speed while interior-point was recognized for its high accuracy. The algorithms were combined in a probabilistic manner, at each iteration the algorithm that performs well was rewarded, otherwise penalised. Perhaps the most challenging task was the design of the performance evaluation at each iteration. With no good performance evaluation the algorithm tends to prioritize the fixed-point algorithm. The hybrid algorithm was numerically shown to be comparable with other existing algorithms.

The future work includes: an investigation of performance evaluation of the algorithms at each iteration, and performing the experiments for different measurement matrices and trying the different problems such as problems from image processing. In our future research we would like to develop technique that will harness the speed of greedy algorithms and theoretical guarantees of convex algorithm.

References

M. V. Afonso, J. M. Boiucas-Dais, and M. A. Frigueiredo. Fast image recovery using variable splitting and constrained optimization. *IEEE Transactions on image Processing*, 19(4):2345 – 2356, 2009.

R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253 – 263, 2008.

J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(0):141 – 148, 1988.

A. Beck and M. Teboulle. A fast iterative shrinkage thresholding algorithm for linear inverse problems. *SIAM*, 2(1):183 – 202, 2009.

R. Berinde, A. C., Gilber, P. Indyk, H. Karloff, and M. J. Strauss. Combining geometry and combinatorics: A unified approach to sparse signal recovery. *IEEE, 46th Annual Allerton Conference on Communication, Control, and Computing, Urbana-Champaign, IL*, 2008.

J. Bioucas-Dias and M. Figueiredo. A new twist: Two-step iterative shrinkage/thresholding algorithm for image restoration. *IEEE Transactions, Image Processing*, 16(12):2992 – 3004, 2007.

T. Blumensath and M. Davies. Iterative thresholding for sparse approximation. *Fourier Analysis and Applications*, 14(5):629 – 654, 2008.

T. Blumensath and M. Davies. Iterative hard thresholding for compressive sensing. *Applied and Computational Harmonic Analysis*, 27(3):265 – 274, 2009.

J. Bobin and S. Becker. Nesta: A fast and accurate first-order method for sparse recovery. Technical report, Califonia Institute of Techonology, 2009.

E. Candes and J. Romberg. Sparsity and incoherence in compressive sampling. *IEEE Selected Topics in Signal Processing*, 1(4):586 – 597, 2006a.

E. Candes and J. Romberg. ℓ_1-magic. *A collection of MATLAB Routines for solving the convex optimization programs central to compressive sampling, www.acm.caltech.edu/l1magic*, 2006b.

E. Candes and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203 – 4215, 2005.

E. Candes and M. Wakin. An introduction for compressive sampling. *IEEE Signal Processing Magazine*, 1(4):21 – 30, 2008.

E. J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions*, 52(2): 489 – 509, 2006a.

E. J. Candes, J. Romberg, and T. Tao. Stable signal recovery form incomplete and inaccurate measurements. *Communications of Pure and Applied Mathematics*, 59(8):1207 – 1223, 2006b.

J. F. Claerbut and F. Muir. Robust modeling with erratic data. *Geophysics*, 38(5):826 – 844, 1973.

P. L. Combettes, E. Va, and E. R. Wajs. Signal recovery by proximal forward-backward splitting. *SIAM Journal on Multiscale Modeling and Simulation*, 4(4):1107 – 1130, 2005.

Y. C. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, editors. *Introduction to algorithms*. The MIT press, Cambridge Massachusetts London, 2012.

S. F. Cotter, R. Adler, R. Rao, and K. Kreutz-Delgado. Forward sequential algorithms for best basis selection. *Vision, Image and Signal Processing, IEE Proceedings*, 146(5):235 – 244, 1999.

W. Dai and O. Milenkovic. Space pursuit for compressive sensing signal reconstruction. *IEEE Transactions on Information Theory*, 55(5):2230 – 2249, 2009.

I. Daubechies, M. Defrise, and C. Mol. An iterative thresholding algorithm for linear inverse problems with sparsity constranint. *Communication of Pure and Applied Mathematics*, 57(5): 1413 – 1457, 2004.

M. Davenport and M. Wakin. Analyis of orthogonal matching pursuit using the restricted isometry property. *IEEE Transactions on Information Theory*, 56(9):4395 – 4401, 2010.

M. Davenport, C. Hegde, M. Duarte, and R. Baraniuk. High-dimensional data fusion via joint manifold learningy. *IEEE Transactions on Information Theory*, 56:395 – 401, 2010.

M. Davenport, C. Hegde, M. Duarte, and R. Baraniuk. An introduction to compressive sensing. *OpenStax CNX. http://cnx.org/contents/f70b6ba0-b9f0-460f-8828-e8fc6179e65f@5.12.*, 2014.

Digital and Signal Processing. Compressed sensing resources. RICE University, Digital and Signal Processing group, http://dsp.rice.edu/cs, Accessed June 2015.

D. L. Donoho. Compressed sensing. *Information Theory, IEEE Transactions*, 52(4):1289 – 1306, 2006.

D. L. Donoho and J. Tanner. Neighborliness of randomly-projected simplices in high dimensions. *Proceedings of the National Academy of Sciences*, 102(27):9452 – 9457, 2005.

D. L. Donoho and J. Tanner. Observed universality of phase transitions in high-dimensional geometry, with applications in modern signal processing and data analysis. *Philosofical Transactions, Royal Society A*, 367:4273 – 42793, 2009a.

D. L. Donoho and J. Tanner. Counting faces of randomly projected polytopes when the projection radically lowers dimension. *American Mathematical Society*, 22(1):1 – 53, 2009b.

D. L. Donoho, Y. Tsaig, I. Drori, and J. L.Starck. Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit. *Information Theory, IEEE Transactions*, 52(4):1289 – 1306, 2006.

R. Dorfman. The detection of defective members of large populations. *Annals of Mathematical Statistics*, 142:436 – 440, 1943.

M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing magazine*, 25(2):83 – 91, 2008.

Y. C. Eldar and G. Kutyniok, editors. *Compressed Sensing: Theory and Applications*. Cambridge university press, New York, 2012.

M. A. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sprase reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journals of Selected Topics in Signal Processing*, 1(4):586 – 597, 2007.

U. Gamper, P. Boesiger, and S. Kozerke. Compressed sensing in dynamic mri. *Magnetic Resonance in Medicine*, 59(2):365 – 373, 2008.

D. Ge, X. Jiang, and Y. Ye. Note of the complexity of ℓ_p minimization. *Springer, Mathematical Programming*, 129(2):285 – 299, 2011.

A. Gilbert, M. Strauss, J. Tropp, and R. Vershynin. Algorithmic linear dimension reduction in ℓ_1 norm for sparse vector. *44th Annual Allerton Conference on Communication, Control, and Computing*, 2006.

A. Gilbert, M. Strauss, and R. Vershynin. One sketch for all: Fast algorithm for compressed sensing. *39th ACM Symposium on Theory of Computing*, 2007.

T. Goldstein and S. Osher. The split bregman algorithm for ℓ_1 rigularized problems. *Techical report 08-09, Computaional and Applied Mathematics, University of California, Los Angeles*, 2008.

E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-minimization: Methodology and convergence. *SIAM Journal on Optimization*, 19(3):1107 – 1130, 2008.

P. Kaelo and M. M. Ali. Some variants of the controlled random search algorithm for global optimization. *Optimization Theory and Applied Mathematics*, 130(2):253 – 264, 2006.

S. J. Kim, k. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE Journals of Selected Topics in Signal Processing*, 1(4):606 –617, 2007.

N. G. Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Journal of Applied and Computational Harmonic Analysis*, 10(2):234–253, 2001.

N. G. Kingsbury and T. H. Reeves. Iterative image coding with overcomplete complex wavelet transforms. *Journal of Applied and Computational Harmonic Analysis*, 10(2):234–253, 2003.

M. Lustig, D. L. Donoho, and J. M. Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly. Compressed sensing mri. *IEEE Signal Processing Magazine*, 25(2):72–82, 2008.

S. Mallat and Z. Zhang. Adaptive time-frequency decomposition with matching pursuits: Time-frequency and time-scale analysis. *Proceedings of the IEEE-SP International Symposium*, 41(12):7–10, 1992.

S. G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

H. Mashud and M. Kaushik. An approximate ℓ_0-norm minimization algorithm for compressed sensing. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 41: 3365–3368, 2009.

O. Milenkovic, R. Baraniuk, and T. Simunic-Rosing. Compressed sensing meets bioinformatics: A novel dna microarray design. *Processing of Working Information Theory and Applications, San Diego, CA*, 2007.

M. Mohtashemi, H. Smith, F. Sutton, D. Walburger, and J. Diggans. Sparse sensing dna microarray-based biosensor: Is it feasible. *IEEE Sensor Applications*, 41(12):3397–3415, 2010.

B. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24 (2):227 – 234, 1995.

P. Neal and B. Stephen. Proximal algorithms. *Foundation and Trends in Optimization*, 1(3):123 – 231, 2013.

D. Needell and J. A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(13):301 – 321, 2009.

D. Needell and R. Vershynin. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Foundations of Computational Mathematics*, 9(3):317–334, 2009.

D. Needell and R. Vershynin. Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *IEEE Selected Topics on Signal Processing*, 4(2):310 – 316, 2010.

A. Neumaier. Solving ill conditioned and singular linear systems: A tutorial on regularization. *SIAM Review*, 40(2):636 – 666, 1998.

H. Nyquist. Certain topics in telegraph. *Transaction of American Institute of Electrical Engineers*, 47(2):617 – 644, 1928.

Y. Pati, R. Rexaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with appliction to wavelet decomposition. In *Signals, Systems and Computers, The Twenty-Seventh Asilomar Conference*, volume 1, pages 40–44. IEEE, 1993.

J. Portilla. Image restoration through ℓ_1 analysis-based sparse optimization in tight frames. *IEEE International Conference on Image Processing*, 47(2):3909 – 3912, 2009.

L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259 – 268, 1992.

F. Santosa and W. Symes. Linear inversion of band-limited reflection seismograms. *SIAM Journal on Scientific and Statistical Computing*, 7:1307 – 1309, 1986.

S. Saravorham, D. Baron, and R. Baraniuk. Compressed sensing reconstruction via belief propagation. *IEEE Transactions on Signal Processing*, 56(6):2346 – 2356, 2008.

C. E. Shannon. Communication in the presence of noise. *Proceedings of Institute of Radio Engineers*, 37(1):10 – 21, 1949.

M. Sheikh, O. Milenkovic, and R. Baraniuk. Compressed sensing dna microarrays. *Techical report TREE-0706, RICE University, ECE Department*, 2007.

J. Shi, W. Yin, S. Osher, and P. Sajda. A fast hybrid algorithm for large scale ℓ_1-regularized logistic regression. *Machine learing Research*, 11:713 – 741, 2010.

J. Shihao, Y. Xue, and L. Carin. Bayesian compressive sensing. *Signal Processing, IEEE Transactions*, 11:713 – 741, 2010.

D. Takhar, J. Laska, M. Wakin, M. Duarte, D. Baron, S. Sarvotham, K. Kelly, and R. Baraniuk. A new compressive imaging camera architecture using optimal-domain compression. *IST SPIE Computational Imaging IV*, 60(65):10 – 21, 2006.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Royal Statistical Society, Series B (Methodogical)*, 58(1):267 – 288, 1996.

M. Tipping. Sparse bayesian learning and the relevance vector machine. *Machine Learning Research*, 1:211 – 244, 2001.

J. A. Tropp and A. C. Gribert. Signal recovery from partial information via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.

M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk. An architecture for compressive imaging. *IEEE International Conference on Image Processing*, 37(1):1273 – 1276, 2006.

Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization and continuation. *Siam Journal on Scientific Computing*, 32 (4):1809 –1831, 2010.

S. J. Wright, R. D. Nowak, and M. A. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transitions in Signal Processing*, 57(1):2479 – 2493, 2009.

W. Yin, S. Osher, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for ℓ_1 minimization with application to compressed sensing. *SIAM Journal on Inaging Science*, 1:143 – 168, 2008.

C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transitions on Mathematical Software*, 23:550 – 560, 1997.