Compressive Sensing Applied to MIMO Radar and Sparse Disjoint Scenes

A Thesis

Submitted to the Faculty

of

Drexel University

by

Michael Francis Minner

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy

March 2016

Dedications

I dedicate this project to George and Catherine.

Acknowledgments

Thank you, Simon Foucart, for your indelible encouragement, guidance, and support these past years. You possess a deep-rooted passion for mathematics, an innate ability to convey even the most complex ideas with total clarity, and an inexhaustible attention to detail. Thank you for introducing me to compressive sensing and the rigors of mathematical analysis, for assisting me in my professional development and for prompting me to attend conferences, gives talks, and publish papers. I am especially grateful for your continued support these past fews years despite the physical distance between our respective universities. Lastly, thank you for your friendship.

Thank you to the entire Department of Mathematics at Drexel University, especially Hugo Woerdeman, Shari Moskow, Doug Wright, Andrew Hicks, Robert Boyer, Pawel Hitczenko, Thomas Yu, Sobha Philip, and Malinda Gilchrist, for your continued support during my time at Drexel.

Thank you to my teachers: Boris Datskovsky, Theodore Burkhardt, Dieter Forster, Chyan Long Lin, Maureen Keppard-Pedlow, Michael Fiocco, Andrew Dolan, Father Hurley, and Sister Judith. You instilled within me a curious mind, a desire to learn, and a strong work ethic.

Thank you to my mentors: Jovana Helms, Matthew Pugh, Johanna Burns, and Lauren Dalton. The passion you bring to your work each and every day is inspiring and the advice and support you have given me has helped me to grow and mature.

Thank you to my colleagues, whom I am proud to call friends: Jean-Luc Bouchot, Jingmin Chen, Phillip Gaudreau, Jeffrey Armstrong, Jason Aran, Caroline Shapcott, Heather Richardson, and Chung Wong. My time at Drexel has been enriching and rewarding thanks to you.

Thank you, Irish, Ward, Roache and Romito. Thank you, John, Tina, Rob, and Jackie. Thank you, Anna, Leslie, Magey and the Hallways. Thank you to the Nerd Herd, the Physics Crew, the Gang, and my Intern Peeps. Thanks for all the terrible jokes, yearbook quotes, Simpsons references, long hours at the gym, discussion based sit-downs, late nights getting pie, Wawa runs and amazing adventures. You have kept me grounded and I look forward to all of our future tomfoolery.

Thank you to my family for your love and support over the course of my life. Thank you, Douglas, Lizzy, Theresa, Brian, Nicky, Allison, Raymond, Michelle, and Ashley. Thank you, Barbara, Margaret, Raymond, Donna, Robert, Deborah, William, Marti, Emerick, Lori, Michael, and Anna May.

Thank you, George and Bette, Andrew and Meghan, and soon Andrew Thomas. I am blessed to have you in my life.

Thank you, Mom and Dad, for your unwavering belief in me throughout my life. Your love and support has meant the world to me.

Thank you, Lord, for always being there for me.

Table of Contents

List of Tables vii

List of Figures viii

Abstract x

1. Introduction 1

1.1 Background 1

1.2 Compressive Sensing 2

1.3 Motivations 4

1.4 Overview 7

2. Compressive Sensing 9

2.1 Introduction 9

2.2 Recovery 10

2.3 Measurement Matrix Properties 15

2.4 Random Matrices 28

2.5 Conclusion 31

3. Application to Radar 32

3.1 Introduction 32

3.2 Radar Background 37

3.3 Problem Formulation 40

3.4 Numerical Simulations 44

3.5 Results 46

3.6 Conclusion 54

4. Sparse Disjoint Structure 56

4.1 Introduction 56

4.2 Sparse Disjointed Vectors 58

4.3 Sufficient Number of Measurements 64
4.4 Necessary Number of Measurements 68
4.5 Conclusion 72
5. Conclusion 74
Bibliography 76
Appendix A: Code 79
A.1 Compressive Sensing MIMO Radar 79
A.2 Sparse Disjoint Vectors 104
Vita 115

List of Tables

List of Figures

1.1 ISAR / Matched Filter with fewer samples / Hybrid Matched Filter & CS, Fraunhofer TIRA [Ender 2010]. Reprinted from Signal Processing, 90/5, Ender, On compressive sensing applied to radar / CS ISAR imaging applied to real data, Pages No. 1410, Copyright (2010), with permission from Elsevier. 5

3.1 Simple diagram of a MIMO radar system setup. Each of the transmit antennas sends a signal to the scene. These signals reflect off of each target and return to each of the receiver antennas. In this example the scene has been discretized in azimuth (angle) and range (delay). 33

3.2 Visualization of a representative target scene discretized in range, azimuth and velocity (Doppler). Please note that while this depiction is a rectangular box, in actuality the azimuth axis would curve slightly. 41

3.3 Comparison of OMP, AISS, L1SQNN, and L1L2CNN algorithms with fixed SNR = 30 dB and a threshold of 0.001. 46

3.4 Comparison of ROC curves for L1SQNN with random arrays and ULA, SNR = 20 dB. . 47

3.5 ROC curves for L1SQNN with random arrays, SNR = 15 dB and various sparsity levels. 48

3.6 Families of ROC curves for L1SQNN with random arrays. Each family of curves corresponds to a different sparsity level (denoted by color), while individual curves within each family correspond to a different SNR (denoted by symbol). 49

3.7 ROC curves for L1SQNN with random arrays at various levels of SNR and different values of the regularization parameter. The upper set of curves in each graph corresponds to a sparsity of 5 (dots), the middle set corresponds to a sparsity of 8 (circles) and the lower set corresponds to a sparsity of 12 (stars). The figures for the noiseless case and for SNR = 50 dB each feature a magnified view of a portion of the ROC curves when the sparsity is 5. 50

3.8 The average number of iterations versus the sparsity at distinct SNR levels and with different values of the regularization parameter for L1SQNN with random arrays. 51

3.9 Comparison of various parameter values, β, in L1SQNN for the larger random array measurement matrices of size 4000×8000 with no noise. The upper set of curves corresponds to a sparsity of 5 (dots), the middle set corresponds to a sparsity of 8 (circles) and the lower set corresponds to a sparsity of 12 (stars). 52

3.10 Comparison of various bandwidth values, B, in L1SQNN with random arrays, SNR = 30 dB and various sparsity levels. 53

4.1 Pictorial representation of sparse disjointed vectors (hollow circles represent zero values). 58

4.2 Sketch of the dynamic program computing the best s-sparse d-disjointed approximations to $\mathbf{x} = (1, 0, 1, 2^{1/4}, 1, 0, 2^{-1/2})$ with $s = 3$ and $d = 1$ for $p = 2$ (left) and $p = 4$ (right). The italicized numbers in each table indicate values that are computed as part of the initialization of the algorithm. Observe that according to the tables, the output for the best 3-sparse 1-disjointed approximation to $\mathbf{x}$ for $p = 2$ is $\mathbf{x}_{p=2} = (1, 0, 1, 0, 1, 0, 0)$, while the output for $p = 4$ is $\mathbf{x}_{p=4} = (1, 0, 0, 2^{1/4}, 0, 0, 2^{-1/2})$. 61

4.3 Comparison of the classical IHT and IHT adapted to the sparse disjointed structure. Here one random 12-sparse 65-disjointed vector in $\mathbb{R}^{1000}$ is generated, then recovered by each method using the same random measurement matrix with $m = 200$. The values of entries of the original vector and the recovered vector are plotted together for each method. 62

4.4 Comparison of the efficacy of the classical IHT and IHT adapted to the sparse disjointed structure. Here $N = 1000$, $d = 10$ and $d = 20$, and the sparsity varies from 1 to 20. One hundred s-sparse and d-disjointed vectors and one hundred random measurement matrices, with $m = 200$, are generated combination of parameters. The ℓ_2-errors of the recovered vectors are compared with a fixed tolerance level to assess whether a particular simulation was successful. The percentage of successful recoveries for each method at each value of d are plotted together. 63

4.5 Illustration of the counting argument in the proof of Lemma 29. 69

Abstract

Compressive Sensing Applied to MIMO Radar and Sparse Disjoint Scenes

Michael Francis Minner

The purpose of remote sensing is to acquire information about an object through the propagation of electromagnetic waves, specifically radio waves for radar systems. However, these systems are constrained by the costly Nyquist sampling rate required to guarantee efficient recovery of the signal. The recent advancements of compressive sensing offer a means of efficiently recovering such signals with fewer measurements. This thesis investigates the feasibility of employing techniques from compressive sensing in on-grid MIMO radar in order to identify targets and estimate their locations and velocities. We develop a mathematical framework to model this problem then devise numerical simulations to assess how various parameters, such as the choice of recovery algorithm, antenna positioning, signal to noise ratio, etc., impact performance. The experimental formulation of this project leads to further theoretical questions concerning the benefits of incorporating an underlying signal structure within the compressive sensing framework. We pursue these concerns for the case of sparse and disjoint vectors. Our computational and analytical treatments illustrate that knowledge of the simultaneity of these structures within a signal provides no benefit in reducing the minimal number of measurements needed to robustly recover such vectors from noninflating measurements, regardless of the reconstruction algorithm.

Chapter 1: Introduction

Here we provide background information for the content of this dissertation. This includes a brief history of the field of research and a short primer on the fundamental problems compressive sensing seeks to resolve. We detail various applications of this mathematical framework to radar systems as well as recent research into so-called structured compressive sensing before highlighting the results presented in this thesis. We conclude by presenting an overview of the remaining chapters of this document.

1.1 Background

As the capacity (and demand) to generate and store information continues to grow, so does the need to effectively capture information, specifically the need to measure large real-world signals, be they images, sounds, or other forms of data. While the challenges and advancements in this area have gained widespread attention under the ubiquitous term "Big Data," here we are concerned with acquiring information in an efficient, stable and robust manner. *Compressive sensing* (or compressed sensing) is a mathematical framework that has gained popularity since it emerged in 2004 due to its surprising success at both measuring signals and compressing them. The crucial ingredient to this paradigm is *sparsity,* that is, many real-world signals contain only a small amount of vital information, relative to the signal's size, when represented in an appropriate manner. Thus, by acquiring these essential components, the signal can be captured with significantly fewer measurements than anticipated while still retaining an excellent approximation of the signal.

Compressive sensing requires two additional factors to succeed in practice as sparsity alone does not guarantee such results. The first is an appropriate measurement scheme, typically a random under-sampling procedure to minimize the number of measurements, that preserves signal information while reducing the dimension size for all sparse signals. The second is a reconstruction algorithm to provide rapid, accurate approximations based on the measurements and knowledge of the sampling scheme. We stress the importance of these two factors. If the method of compression can not effectively measure all appropriately sparse signals, then there are signals which we will always fail to capture and instead obtain poor approximations in practice. Second, if the available algorithms are sluggish, costly, and return estimations that are far from the true signal, then their utility is impractical in comparison to alternative, currently available approaches. Although the demands of compressive sensing are at times difficult to integrate in practice, this area of study offers a rich overlapping field of research in mathematics, engineering, and the sciences.

Understandably, this capability to accurately and quickly compress large data sets has found far-reaching applications. These include the single-pixel camera [Duarte et al. 2008], accelerated MRIs [Lustig et al. 2007], and metagenomic analysis [Koslicki et al. 2013], to briefly name but a few examples. Thanks in part to the overlapping nature of the foundations upon which compressive sensing has been built, such as analysis, optimization and probability theory, researchers in the field have made theoretical advancements in approximation theory [Foucart et al. 2010], matrix completion [Candès et al. 2013] and graph theory [Foucart and Rauhut 2013 Chapter 13]. In addition to reducing costs and computational complexity by minimizing the number of measurements, researchers have also explored the potential for achieving higher levels of resolution by incorporating techniques from the literature into current systems and maintaining the typical number of measurements. See [Candès and Fernandez-Granda 2014] and [Zhu and Bamler 2012] for examples of this work in *super-resolution*. As more people learn of the tools and advantages offered by compressive sensing, this field will continue to develop and expand in practice and influence.

1.2 Compressive Sensing

The achievements of compressive sensing have been attained by combining tools from many different areas of mathematics including linear algebra, convex analysis, harmonic analysis and much more. The earliest work in the area of sparse reconstruction is attributed to de Prony, in the late eighteenth century. His method, known as the Prony method, estimated the parameters of a signal, e.g. frequency, amplitude, etc., based on equidistant samples of the signal [Prony 1795]. In the 1960s and 1970s, ℓ_1-minimization found early applications in the context of sparse frequency estimation and through the work of geophysicists attempting to ascertain changes in subsurface layers [Taylor et al. 1979]. The end of the twentieth century saw further developments in the use of ℓ_1-minimization and greedy algorithms for the recovery of sparse solutions. Compressive sensing as it is now known, came into its own with the seminal works by Candès, Romberg, and Tao [Candès et al. 2006] and by Donoho [Donoho 2006]. Their works are credited for being the first to have combined ℓ_1-minimization with randomness in the measurement matrix to successfully solve underdetermined systems of equations. Since these early publications, the field has experienced wide-spread attention and tremendous growth.

The essential problem of compressive sensing is to construct an accurate approximation to a compressible vector $\mathbf{x} \in \mathbb{C}^N$ from a minimal number of linear measurements $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{y} \in \mathbb{C}^m$ and $\mathbf{A} \in \mathbb{C}^{m \times N}$, in a stable and robust manner. One would not expect such reconstruction to be possible when $m \ll N$; however, the additional a priori knowledge that $\mathbf{x}$ is sparse allows for such recovery. A vector is called *s-sparse* if the number of non-zero entries is at most s. A key finding of compressive sensing states that a measurement matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ of probabilistic construction with $m \geq Cs \ln(N/s)$, where C is a constant and s is the sparsity of the signal, will satisfy the *Restricted Isometry Property* (RIP) with high probability. This result is a computationally significant improvement over the hefty Nyquist-rate because the RIP guarantees that *any* s-sparse signal $\mathbf{x} \in \mathbb{C}^N$ can be recovered exactly. An intuitive answer as to why this is possible is that the "important" information of a sparse signal is stored within only a small number of locations; hence, one would expect the number of measurements needed to be proportional to these "intrinsic" elements of the signal.

The practical applications of compressive sensing are far reaching, but one must keep in mind the importance of both the design of the measurement matrix $\mathbf{A}$ and the method of reconstruction for a given setting. The standard algorithms, such as greedy and iterative procedures, require different input parameters upon initialization and may return their outputs at significantly different rates depending on the problem size. At the same time, the conditions one seeks on the measurement matrix to provide guarantees on the quality of the reconstructed approximation to the original signal are often onerous to verify. In fact no known constructions exist which match the theoretically optimal minimal number of measurements attainable via classes of random matrices; thus, if possible, one must utilize random matrices in order to circumvent this difficulty and achieve the desired guarantees with high probability. Accordingly, the application of compressive sensing requires a sparse representation for the data of interest and an undersampling scheme to compress the data while guaranteeing effective recovery via a prescribed method.

In practice, one may not possess a sparse representation for the data of interest or the physical constraints of the problem may inhibit the design of the measurement matrix. If a method to sparsify the data is available, this transformation may also cause conflict with the sampling scheme and obstruct any attempts to provide theoretical guarantees on the efficacy of the reconstruction algorithm. Hence, we find three areas of critical importance for the advancement of compressive sensing. First, we must investigate the parameters under which compressive sensing can feasibly be adopted to reduce the computational complexity of a given problem. Next, we must probe the signals of interest for underlying structure then assess how such information may be leveraged to further improve performance. Finally, we must explore how existing recovery methods compare in practice and, depending on the previous pursuits, develop new methods of signal recovery, or modify existing methods, to better serve a specific signal structure or better meet the conditions dictated by the problem. Keeping these considerations in mind, we focus our attention on a specific class of radar systems due to the increasing demand for improved resolution and performance from these systems in various civilian, industrial and military applications.

1.3 Motivations

As an illustration of the remarkable success of compressive sensing, consider Figure 1.1, presented in [Ender 2010]. Here, we see an application of compressive sensing to Inverse Synthetic Aperture Radar (ISAR) with real data from the TIRA (Tracking and Imaging Radar) system located at the Fraunhofer Institute for High Frequency Physics and Radar Techniques FHR, in Wachtberg, Germany. ISAR is an imaging method where a fixed antenna is used to image isolated rotating targets. The image on the left displays a satellite obtained from TIRA using ISAR with $1024 \times 1697 = 1,737,728$ samples in k-space. The data undergoes *polar reformatting* and interpolation to a rectangular grid in a (k_x, k_y) plane and an inverse Fourier transform is applied to generate the image. In the center we see the same satellite, but here 70 k_y values are selected at random so that only $1024 \times 70 = 71,680$ samples are used with a matched filter. These same samples are taken to generate the image on the right with a hybrid matched filter/compressive sensing technique. This yields a reduction in the number of samples by a factor of about 25!

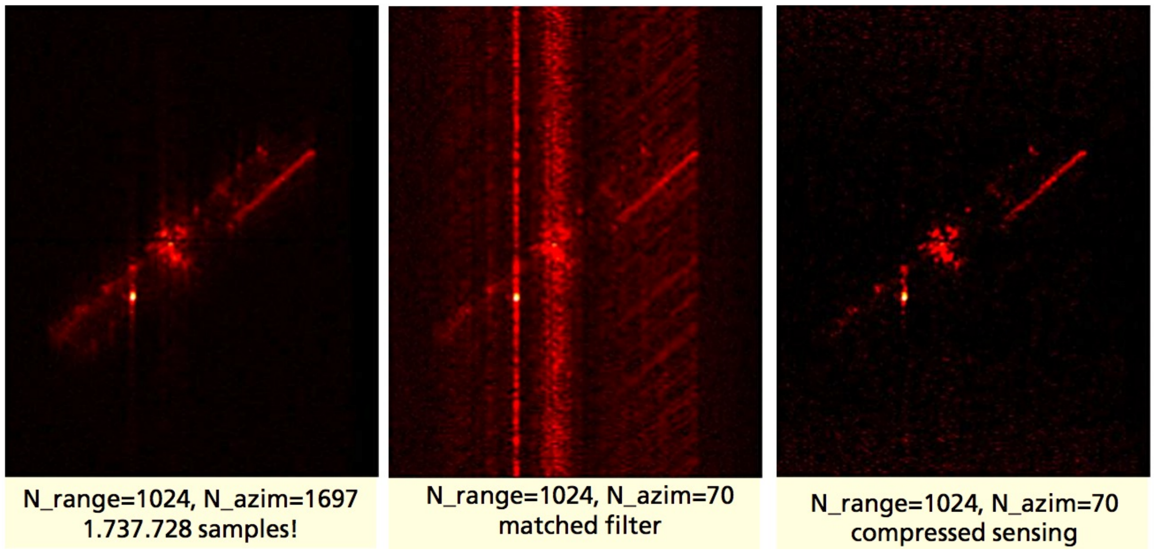


Figure 1.1: ISAR / Matched Filter with fewer samples / Hybrid Matched Filter & CS, Fraunhofer TIRA [Ender 2010]. Reprinted from Signal Processing, 90/5, Ender, On compressive sensing applied to radar / CS ISAR imaging applied to real data, Pages No. 1410, Copyright (2010), with permission from Elsevier.

We highlight a few recent publications to further illustrate the wide range of applications being investigated in this field. Although they are not the first to consider applying compressive sensing to Multiple Input Multiple Output (MIMO) radar, the authors of [Yu et al. 2011] explore this application further, with particular emphasis on analyzing the design matrix. More specifically, they propose methods of designing the measurement matrix for a general MIMO radar setting which seek to both decrease the coherence (See Chapter 2) of the matrix and improve the signal to interference ratio. The results established in [Zhu and Bamler 2012], one of a series of papers, incorporate compressive sensing in a reconstruction algorithm for Synthetic Aperture Radar Tomography. The problem they consider is a more general super-resolution problem, but they demonstrate that their compressive sensing based method is applicable and robust when recovering a 'reasonable' target scene with few sample acquisitions and low SNR. The author of [Ender 2012] presents a method of utilizing auto-focus, i.e. motion compensation, ISAR in reconstructing sparse scenes with compressive sensing. An iterative approach is introduced which compares differences in phase information from images captured over processing intervals to extrapolate other parameters of the target, such as translational and rotational movement, and reduce error, improving image quality. These examples are only a small sample of the diverse projects applying compressive sensing to radar, but they demonstrate the potential for further enhancements.

We also review additional publications to illustrate the recent work of incorporating structure in compressive sensing. The authors of [Baraniuk et al. 2010] propose a *model-based compressive sensing* framework as means to reduce the number measurements needed for robust signal recovery. Their approach requires the specified model to obey a *nested approximation property* and an associated *Restricted Amplification Property*, in analog to the standard RIP. They employ an adaptation of the CoSaMP (See Chapter 2) algorithm to further validate their theory with experimental data for wavelet trees and block sparsity models. However, the nested approximation property is highly restrictive and not valid for the specific structure we consider in Chapter 4. This work is followed in [Hegde et al. 2015] which introduces a new framework dubbed *approximation-tolerant model-based compressive sensing*. Their approach forgoes the requirement that the model-based projection of [Baraniuk et al. 2010] be exact and instead requires "approximate solutions for the model-projection." The theory is then applied to the Constrained Earth Mover's Distance structured sparsity model [Schmidt et al. 2013] to substantiate the analysis. The author of [Foucart 2011b] employs the hard thresholding pursuit (See Chapter 2) method in the recovery of jointly sparse signals, i.e., signals which share a support, and highlights the significant improvement it provides over reconstructing such signals individually. [Hegde et al. 2009] examines the structure we call *sparse and disjointed* in the context of representing neural spike trains under the model presented in [Baraniuk et al. 2010]. They present a formal statement on the number of measurements that are sufficient to guarantee robust recovery of sparse disjointed vectors. Lastly, though our brief review is by no means exhaustive, the researchers of [Oymak et al. 2015] consider the impact combinations of various norms possess in promoting distinct structures during recovery. They show that vectors exhibiting multiple structures require at least as many Gaussian random measurements for their recovery via combined convex relaxations as is possible through the convex relaxation associated with just one of the structures.

1.4 Overview

This dissertation examines the applicability of compressive sensing to MIMO radar and further explores the theoretical consequences that result from considering the naturally arising structure found in the aforementioned experimental work. The original motivation for this exposition resulted from participation in an introductory radar summer school that not only provided an excellent introduction into the fundamentals of radar systems but also an opportunity to discuss recent attempts to incorporate the results of compressive sensing into various aspects of radar technologies. After surveying the landscape of research in this field and discussing these findings with fellow researchers, MIMO radar was deemed most suitable for improvement via the techniques of compressive sensing. Thus, an experimental project was conceived to investigate the parameters under which various reconstruction algorithms would correctly detect point targets in a small, far-field scene. This endeavor combined distinct elements from recent works in both radar and compressive sensing to produce an original study, the formal results of which were published in [Minner 2015].

While conducting this investigation, a decision on how to formulate the problem within the programming code lead to an interesting observation. The few targets within the scene can not occupy the same locations; hence, when represented by a vector, their positions are separated from one another. Given that the vector, or scene, is sparse and separated, can we leverage this information to further reduce the minimal number of measurements needed to recover such vectors in a stable and robust manner? We termed this structure *sparse and disjointed* and adapted standard reconstruction algorithms to employ this additional knowledge in order to investigate the query computationally. This analysis facilitated our theoretical conclusions which are in the negative, i.e., incorporating information of this simultaneous structure when utilizing a noninflating measurement scheme does not reduce the optimal number of measurements need for robust reconstruction, regardless of the recovery method, within the compressive sensing framework. We analytically justify these findings which were also presented in [Foucart et al. 2015].

The remainder of this text proceeds as follows. First, in Chapter 2, we detail the fundamentals of compressive sensing: notation, definitions, and basic theory. We present various recovery algorithms, discuss pertinent properties of the measurement matrices, particularly random matrices, and examine the theoretical guarantees these elements can produce in application. This exposition follows the presentation of [Foucart and Rauhut 2013]. Next, in Chapter 3 we provide an introduction to radar and describe early applications of compressive sensing to radar systems. We mathematically formulate our problem of interest then design numerical simulations to investigate how various parameters can improve performance, the results of which are detailed at length. Chapter 4 introduces the notion of *sparse and disjoint* vectors, a specific structure found in the signals analyzed in Chapter 3. We establish a method for finding the best sparse and disjoint approximation to a given vector via dynamic programming. Furthermore, we determine the necessary and sufficient number of noninflating measurements to recover such vectors in the presence of noise. Chapter 5 concludes this dissertation with a brief summary and notes on the potential for future research. Appendix A contains MATLAB (The MathWorks, Inc., Natick, MA) files for our work in both MIMO radar and in sparse and disjoint vectors.

Chapter 2: Compressive Sensing

Here we provide an overview of compressive sensing, including notations, definitions and theorems. We examine several standard recovery algorithms, including greedy and iterative methods, detail measurement matrix properties, such as the Null Space Property and coherence, and highlight the associated guarantees these tools can produce. Additionally we consider random matrices and review their relationship to the Restricted Isometry Property in compressive sensing. This chapter will establish the necessary groundwork for the experimental and theoretical results of the two proceeding chapters.

2.1 Introduction

In classical sampling theory a continuous signal band limited to bandwidth B is completely determined from sampled measurements when the sampling frequency is greater than or equal to $2B$. The Nyquist-Shannon-Kotelnikov Theorem states this in precise terms:

Theorem 1. (Nyquist-Shannon-Kotelnikov) If the Fourier Transform $\hat{f}$ of a function f is compactly supported in $[-\frac{B}{2}, \frac{B}{2}]$, then f can be reconstructed from measurements $\{f(\frac{2\pi n}{B}), n \in \mathbb{Z}\}$ via

$$f(t) = \sum_{n=-\infty}^{\infty} f\left(\frac{2\pi n}{B}\right) \operatorname{sinc}\left(\frac{Bt}{2} - \pi n\right),$$

where

$$\operatorname{sinc}(x) = \begin{cases} \dfrac{\sin(x)}{x} & \text{if } x \neq 0, \\ 1 & \text{if } x = 0. \end{cases}$$

In other words, one must sample at a rate that is at least twice the highest frequency to guarantee successful recovery of the signal. For high bandwidth systems, such as radar systems, sampling at this rate can become prohibitively expensive. However, while this is a sufficient condition for perfect reconstruction, it is not, as we shall see, a necessary condition.

We now introduce definitions and notations essential to Compressive Sensing, and we follow the same presentation as in [Foucart and Rauhut 2013]. We begin by denoting $[N] := \{1, 2, ..., N\}$, i.e., the set of the first N integers, we also denote $\text{card}(S)$ as the cardinality of a set S, and formally define the previously mentioned notion of sparsity.

Definition 2. (See Definition 2.1 of [Foucart and Rauhut 2013]) The support of a vector $\mathbf{x} \in \mathbb{C}^N$ is the index set of its non-zero entries, $\text{supp}(\mathbf{x}) := \{j \in [N] : x_j \neq 0\}$. A vector $\mathbf{x} \in \mathbb{C}^N$ is called s-sparse if at most s of its entries are nonzero, i.e.,

$$\|\mathbf{x}\|_0 := \text{card}(\text{supp}(\mathbf{x})) \leq \text{s}.$$

In most practical settings a signal will not be perfectly s-sparse, but close to it; thus, we introduce the notions of *compressibility* and *the error of best s-term approximation.*

Definition 3. (See Definition 2.2 of [Foucart and Rauhut 2013]) For $p > 0$, the ℓ_p-error of best s-term approximation to a vector $\mathbf{x} \in \mathbb{C}^N$ is defined by

$$\sigma_s(\mathbf{x})_p := \inf\{\|\mathbf{x} - \mathbf{z}\|_p, \mathbf{z} \in \mathbb{C}^N \text{is s-sparse}\}.$$

We say that $\mathbf{x} \in \mathbb{C}^N$ is *compressible* if $\sigma_s(\mathbf{x})_p$ rapidly decays as s increases.

2.2 Recovery

Since we cannot take measurements with infinite precision and must often account for corruption and errors in our data, we are highly interested in reconstruction algorithms which not only remain stable and robust but also provide recovery in a reasonable amount of time. As an initial attempt at algorithmically recovering the sparse vector $\mathbf{x}$ from $\mathbf{y} = \mathbf{A}\mathbf{x}$, one may consider the optimization problem

$$\min \|\mathbf{z}\|_0 \;\text{ subject to }\; \mathbf{A}\mathbf{z} = \mathbf{y}. \tag{2.1}$$

This problem is impractical to solve, though. Essentially, one would need to consider all possible subsets of $[N]$ of size s. There are $\binom{N}{s}$ such sets and even in small test cases, this number becomes exponentially large. Since this problem is nonconvex and NP-hard, in its place one often considers the convex optimization problem

$$\min \|\mathbf{z}\|_1 \quad \text{subject to} \quad \mathbf{A}\mathbf{z} = \mathbf{y}. \tag{2.2}$$

The method for recovering $\mathbf{x}$ from (2.2) is known as *basis pursuit* or *ℓ_1-minimization* and the problem can be solved with reasonably fast algorithms from convex optimization, see for instance [Chambolle and Pock 2011] and [Daubechies et al. 2010]. Under appropriate assumptions the solution of (2.2) will match the solution of (2.1). Additionally, under similar assumptions, greedy algorithms, such as *Orthogonal Matching Pursuit* (OMP) and *Compressive Sampling Matching Pursuit* (CoSaMP) [Needell and Tropp 2009], and thresholding algorithms, such as *Iterative Hard Thresholding* (IHT) [Blumensath and Davies 2009b] and *Hard Thresholding Pursuit* (HTP) [Foucart 2011a], will also obtain solutions to (2.1). The pseudocode provided below for each of these methods follows the presentation of [Foucart and Rauhut 2013 See Chapter 3].

Highlighting the Orthogonal Matching Pursuit, this method builds the support of a target vector, $\mathbf{x}^n$, which best fits the measurements by adding one index to the support per iteration. The additional index is chosen to reduce the ℓ_2-norm of the residual $\mathbf{y} - \mathbf{A}\mathbf{x}^n$. This algorithm is detailed precisely below.

Orthogonal Matching Pursuit (OMP)

Input: measurement matrix $\mathbf{A}$, measurements $\mathbf{y}$.

Initialization: support $S^0 = \emptyset$, target vector $\mathbf{x}^0 = 0$.

Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$\mathbf{x}^{n+1} = S^n \cup \{j_{n+1} := \mathrm{argmax}\{|(\mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^n))_j|, j \in [N]\}\}, \tag{OMP$_1$}$$

$$\mathbf{x}^{n+1} = \mathrm{argmin}\{\|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2, \mathrm{supp}(\mathbf{z}) \subseteq S^{n+1}\}, \tag{OMP$_2$}$$

Output: the $\bar{n}$-sparse vector $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$.

Closer analysis of (OMP$_2$) reveals that $\mathbf{x}^{n+1}$ can be chosen by an alternate form $\mathbf{x}^{n+1} := \mathbf{A}^\dagger_{S^{n+1}}\mathbf{y}$, where $\mathbf{A}^\dagger_{S^{n+1}} = (\mathbf{A}^*_{S^{n+1}}\mathbf{A}_{S^{n+1}})^{-1}\mathbf{A}^*_{S^{n+1}}$. By definition, $\mathbf{x}^{n+1}$ is characterized by recognizing $\mathbf{Az}$ as the orthogonal projection of $\mathbf{y}$ onto the space of $\{\mathbf{Az}, \mathrm{supp}(\mathbf{z}) \subseteq S^{n+1}\}$. Thus, we have

$$\langle \mathbf{y} - \mathbf{A}\mathbf{x}^{n+1}, \mathbf{A}\mathbf{z}\rangle = 0, \quad \text{for all} \quad \mathbf{z} \in \mathbb{C}^N \quad \text{with} \quad \mathrm{supp}(\mathbf{z}) \subseteq S^{n+1},$$
$$\langle \mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^{n+1}), \mathbf{z}\rangle = 0, \quad \text{for all} \quad \mathbf{z} \in \mathbb{C}^N \quad \text{with} \quad \mathrm{supp}(\mathbf{z}) \subseteq S^{n+1}.$$

However, this necessitates

$$(\mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^{n+1}))_{S^{n+1}} = 0,$$
$$\mathbf{A}^*_{S^{n+1}}\mathbf{y} = \mathbf{A}^*_{S^{n+1}}\mathbf{A}_{S^{n+1}}\mathbf{x}^{n+1},$$
$$\mathbf{x}^{n+1} := \mathbf{A}^\dagger_{S^{n+1}}\mathbf{y}.$$

CoSaMP proceeds by incorporating information about the support from previous iterations to build a best fit for the data, then pruning the support of the approximation by retaining the s largest absolute entries in magnitude before updating the estimate of the original signal $\mathbf{x}$. This algorithm incorporates two operators: $L_s(\mathbf{x})$, which returns the index set of the s largest entries of $\mathbf{x}$ in magnitude, and $H_s(\mathbf{x})$, which is the *hard thresholding operator* of order s and returns the vector $\mathbf{x}$ restricted to the support outputted by $L_s(\mathbf{x})$. The hard thresholding operator will retain the largest entries in magnitude of a vector and set all other entries to zero.

Compressive sampling matching pursuit (CoSaMP)

Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, sparsity level s.

Initialization: s-sparse target vector $\mathbf{x}^0$, typically $\mathbf{x}^0 = \mathbf{0}$.

Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$S^{n+1} = \operatorname{supp}(\mathbf{x}^n) \cup L_{2s}(A^*(y - Ax^n)) \tag{CoSaMP$_1$}$$

$$\mathbf{u}^{n+1} = \underset{\mathbf{z}\in\mathbb{C}^N}{\operatorname{argmin}} \{\|\mathbf{y} - \mathbf{A}\mathbf{x}^n\|_2, \ \ \operatorname{supp}(\mathbf{z}) \subset S^{n+1}\}, \tag{CoSaMP$_2$}$$

$$x^{n+1} = H_s(\mathbf{u}^{n+1}). \tag{CoSaMP$_3$}$$

Output: the s-sparse vector $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$.

IHT, as the name implies, employs the hard thresholding operator to iteratively build approximations to the target vector $\mathbf{x}$. This method comes from attempting to solve the system $\mathbf{A}^*\mathbf{y} = \mathbf{A}^*\mathbf{A}\mathbf{x}$ and is based on a fixed point method.

Iterative hard thresholding (IHT)

Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, sparsity level s.

Initialization: s-sparse target vector $\mathbf{x}^0$, typically $\mathbf{x}^0 = \mathbf{0}$.

Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$\mathbf{x}^{n+1} = H_s(\mathbf{x}^n + \mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^n)). \tag{IHT}$$

Output: the s-sparse vector $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$.

Since IHT does not perform an orthogonal projection to find a best s-term approximation, modifying the algorithm to incorporate this step results in the HTP. This method determines a candidate for the support of the vector $\mathbf{x}$ then finds the best approximation with this support to fit the data.

Hard thresholding pursuit (HTP)

Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, sparsity level s.

Initialization: s-sparse target vector $\mathbf{x}^0$, typically $\mathbf{x}^0 = \mathbf{0}$.

Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$S^{n+1} = L_s(\mathbf{x}^n + \mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^n)), \tag{HTP$_1$}$$

$$\mathbf{x}^{n+1} = \operatorname*{argmin}_{\mathbf{z} \in \mathbb{C}^N} \{\|\mathbf{y} - \mathbf{A}\mathbf{x}^n\|_2, \;\; \mathrm{supp}(\mathbf{z}) \subset S^{n+1}\}. \tag{HTP$_2$}$$

Output: the s-sparse vector $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$.

Each of the aforementioned algorithms have straightforward implementations and work well in practice; the choice of which is most effective will depend on the parameters of the given problem. We note that Orthogonal Matching Pursuit and Basis Pursuit, which is not an algorithm itself and may instead be solved via any number of existing methods, do not require an estimate of the sparsity level as inputs while other algorithms do require this information. Empirical tests can provide insight into which approach is most appropriate as the problem size alone can exhibit a profound effect on the run time of these algorithms, e.g. OMP may slow down if the sparsity is not sufficiently small due to the projection step. Researchers continue to produce an ever-growing library of additional methods and variants designed with compressive sensing in mind but their details are beyond the scope of this document.

2.3 Measurement Matrix Properties

As previously mentioned, the design of the sensing matrix is vital to compressive sensing, we are therefore interested in analyzing properties of the matrix which will guarantee the exact recovery of a sparse vector or nearly exact recovery of a compressible vector via the algorithms presented above. Hence, we recall the following definitions and, associated lemmas, propositions, and theorems beginning, with the *null space property.* We stress that the presentation of this section, including the proofs, is adopted from [Foucart and Rauhut 2013] and not our own.

Definition 4. (See Definition 4.1 of [Foucart and Rauhut 2013]) A matrix $\mathbf{A} \in \mathbb{C}^{m\times N}$ is said to satisfy the null space property relative to a set $S \subset [N]$ if

$$\|\mathbf{v}_S\|_1 < \|\mathbf{v}_{\bar{S}}\|_1 \text{ for all } \mathbf{v} \in \ker \mathbf{A} \setminus \{0\}. \tag{2.3}$$

The matrix is said to satisfy the null space property of order s if it satisfies the null space property relative to any set $S \subset [N]$ with $\text{card}(s) \leq s$.

Here $\bar{S}$ is the complement of the support S and $\mathbf{v}_S$ is the vector $\mathbf{v} \in \mathbb{C}^N$ restricted to the elements on the support of S with all other elements 0. This condition essentially requires that the vectors in the kernel of $\mathbf{A}$ are far from being sparse. Intuitively, if this were not the case then a non-zero sparse vector $\mathbf{v} \in \ker \mathbf{A}$ would yield $\mathbf{y} = \mathbf{A}\mathbf{v} = \mathbf{0}$, so that the zero vector would be reconstructed instead of $\mathbf{v}$. The following theorem highlights the importance of this property.

Theorem 5. (See Theorem 4.4 of [Foucart and Rauhut 2013]) Given a matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$, every vector $\mathbf{x} \in \mathbb{C}^N$ supported on a set S is the unique solution of (2.2) with $\mathbf{y} = \mathbf{A}\mathbf{x}$ if and only if $\mathbf{A}$ satisfies the null space property of order s.

Proof. ($\Rightarrow$)We suppose that for a given index set S, every $\mathbf{x} \in \mathbb{C}^N$ supported on S is the unique minimizer of (2.2). Then for all $\mathbf{v} \in \ker \mathbf{A} \setminus \{0\}$ we have $\mathbf{A}\mathbf{v}_S = -\mathbf{A}\mathbf{v}_{\bar{S}}$; thus, $\mathbf{v}_S$ is unique minimizer of (2.2) but with $\mathbf{v}_S$ replacing $\mathbf{x}$. Since $-\mathbf{v}_{\bar{S}} \neq \mathbf{v}_S$, else $\mathbf{v} = 0$, we must have that $\|\mathbf{v}_S\|_1 < \|\mathbf{v}_{\bar{S}}\|_1$.

($\Leftarrow$) Supposing that (2.3) holds relative to a set S, we consider a vector $\mathbf{x} \in \mathbb{C}^N$ supported on S and vector $\mathbf{z} \in \mathbb{C}^N, \mathbf{z} \neq \mathbf{x}$, with $\mathbf{A}\mathbf{z} = \mathbf{A}\mathbf{x}$. Defining $\mathbf{v} := \mathbf{x} - \mathbf{z} \in \ker \mathbf{A} \setminus \{0\}$, we have

$$\begin{aligned} \|\mathbf{x}\|_1 \leq \|\mathbf{x} - \mathbf{z}_S\|_1 + \|\mathbf{z}_S\|_1 = \|\mathbf{v}_S\|_1 + \|\mathbf{z}_S\|_1 \\ < \|\mathbf{v}_{\bar{S}}\|_1 + \|\mathbf{z}_S\|_1 = \|-\mathbf{z}_{\bar{S}}\|_1 + \|\mathbf{z}_S\|_1 = \|\mathbf{z}\|_1. \end{aligned}$$

Hence (2.2) is satisfied. □

Theorem 5 provides a necessary and sufficient condition for exact recovery via ℓ_1-minimization. If we allow the set S to vary then the following theorem is a direct result of Theorem 5.

Theorem 6. (See Theorem 4.5 of [Foucart and Rauhut 2013]) Given a matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$, every s-sparse vector $\mathbf{x} \in \mathbb{C}^N$ is the unique solution of (2.2) with $\mathbf{y} = \mathbf{A}\mathbf{x}$ if and only if $\mathbf{A}$ satisfies the null space property of order s.

Theorem 6 demonstrates that for every $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{x}$ is s-sparse, ℓ_1-minimization (2.2) actually solves the ℓ_0-minimization problem (2.1) when the null space property of order s holds. This is due to the fact that, assuming every s-sparse vector $\mathbf{x}$ is recovered via ℓ_1-minimization from $\mathbf{y} = \mathbf{A}\mathbf{x}$, if $\mathbf{z}$ is the solution of the ℓ_0-minimization problem (2.1) then $\|\mathbf{z}\|_0 \leq \|\mathbf{x}\|_0$. However, by Theorem 6, every s-sparse vector is the unique solution of (2.2); thus, $\mathbf{x} = \mathbf{z}$.

The issue of *stability* arises when the vector under consideration is not exactly s-sparse, but rather close to s-sparse, i.e. *compressible*. If we enhance the restriction of Equation (2.3), we arrive at the so-called *stable null space property*.

Definition 7. (See Definition 4.11 of [Foucart and Rauhut 2013]) A matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ satisfies the stable null space property with constant $0 < \rho < 1$ relative to a set $S \subset [N]$ if

$$\|\mathbf{v}_S\|_1 \leq \rho \|\mathbf{v}_{\bar{S}}\|_1 \text{ for all } \mathbf{v} \in \ker \mathbf{A}. \tag{2.4}$$

The matrix is said to satisfy the stable null space property of order s with constant $0 < \rho < 1$ if it satisfies the stable null space property with constant $0 < \rho < 1$ relative to any set $S \subset [N]$ with $\text{card}(S) \leq s$.

This restriction leads to the following result, which we state without proof.

Theorem 8. (See Theorem 4.12 of [Foucart and Rauhut 2013]) Suppose a matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ satisfies the stable null space property of order s with constant $0 < \rho < 1$. Then, for any vector $\mathbf{x} \in \mathbb{C}^N$, a solution $\mathbf{x}^\sharp$ of (2.2) with $\mathbf{y} = \mathbf{A}\mathbf{x}$ approximate the vector $\mathbf{x}$ with ℓ_1-error

$$\|\mathbf{x} - \mathbf{x}^\sharp\|_1 \leq \frac{2(1+\rho)}{(1-\rho)} \sigma_s(\mathbf{x})_1. \tag{2.5}$$

We highlight one of the differences between Theorem 6 and Theorem 8: although one can no longer guarantee the exact recovery of $\mathbf{x}$, the strengthening of Equation (2.3) to Equation (2.4) allows for the recovery of any $\mathbf{x}$, rather than just s-sparse $\mathbf{x}$, and provides a bound on the quality of the reconstruction in terms of the ℓ_1-error of the best s-term approximation. This result can be strengthened further, see [Foucart and Rauhut 2013 Theorem 4.14].

As previously mentioned, we are limited in the precision to which we can measure signals; thus, we seek measurement processes which are robust in order to still capture measurements that have been corrupted with noise and other possible errors.

Definition 9. (See Definition 4.21 of [Foucart and Rauhut 2013]) Given $q \geq 1$, the matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ is said to satisfy the ℓ_q-robust null space property of order s (with respect to $\|\cdot\|$) with constants $0 < \rho < 1$ and $\tau > 0$ if, for any set $S \subset [N]$ with $\operatorname{card}(S) \leq s$,

$$\|\mathbf{v}_S\|_q \leq \frac{\rho}{s^{1-1/q}} \|\mathbf{v}_{\bar{S}}\|_1 + \tau \|\mathbf{A}\mathbf{v}\| \text{ for all } \mathbf{v} \in \mathbb{C}^N. \tag{2.6}$$

This enhancement of the null space property leads to the following guarantee, also presented without proof, on the solution of the optimization problem, known as quadratically constrained basis pursuit,

$$\underset{\mathbf{z} \in \mathbb{C}^N}{\text{minimize}} \;\; \|\mathbf{z}\|_1 \;\; \text{subject to} \;\; \|\mathbf{A}\mathbf{z} - \mathbf{y}\|_2 \leq \eta. \tag{2.7}$$

Theorem 10. (See Theorem 4.22 of [Foucart and Rauhut 2013]) Suppose the matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ satisfies the ℓ_2-robust null space property of order s with constants $0 < \rho < 1$ and $\tau > 0$. Then, for any vector $\mathbf{x} \in \mathbb{C}^N$, a solution $\mathbf{x}^\sharp$ of (2.7) with $\|\cdot\| = \|\cdot\|_2$, $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$ and $\|\mathbf{e}\|_2 \leq \eta$ approximates the vector $\mathbf{x}$ with ℓ_p-error

$$\|\mathbf{x} - \mathbf{x}^\sharp\|_p \leq \frac{C}{s^{1-1/p}} \sigma_s(\mathbf{x})_1 + D s^{1/p-1/2} \eta, \qquad 1 \leq p \leq 2, \tag{2.8}$$

for some constants $C, D > 0$ depending only on ρ and τ.

Interpreting this result, we see that in return for putting an additional restriction on the measurement matrix, one obtains a bound on the reconstruction error that depends on both the best s-term approximation and the error in the measurements. Aside from the issue of the norms involved, this is the best one could reasonably hope to achieve.

Next, we introduce the *coherence*, which provides a basic measure of the suitability of a measurement matrix for compressive sensing.

Definition 11. (See Definition 5.1 of [Foucart and Rauhut 2013]) Let the matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ have ℓ_2-normalized columns $\mathbf{a}_1, ..., \mathbf{a}_N$. The coherence $\mu = \mu(\mathbf{A})$ of the matrix $\mathbf{A}$ is defined as

$$\mu := \max_{1 \leq i \neq j \leq N} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle|. \tag{2.9}$$

Notice that μ is always between 0 and 1. The coherence is often easier to calculate in place of verifying whether or not a matrix satisfies the (stable or robust) null space property, which is NP-hard. We will need the following proposition in order to prove the next theorem.

Proposition 12. (See Proposition 3.5 of [Foucart and Rauhut 2013]) Given a matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$, every non-zero vector $\mathbf{x} \in \mathbb{C}^N$ supported on a set S of size s is unequivocally recovered from $\mathbf{y} = \mathbf{A}\mathbf{x}$ after s steps of OMP if and only if the following two conditions both hold:

(1) $\mathbf{A}_S$ is injective,

(2) $\max_{j \in S} |(\mathbf{A}^*\mathbf{r})_j)| > \max_{\ell \in \bar{S}} |(\mathbf{A}^*\mathbf{r})_\ell)|$ for all $\mathbf{r} \in \{\mathbf{A}\mathbf{z}, \operatorname{supp}(\mathbf{z}) \subseteq S\}\{\mathbf{0}\}$.

Proof. ($\Rightarrow$) Suppose that OMP recovers all s-sparse vectors with support S after at most s iterations. Note that for two vectors $\mathbf{x}, \mathbf{z}$ supported on S with $\mathbf{x} \neq \mathbf{z}$, we must have $\mathbf{A}\mathbf{x} \neq \mathbf{A}\mathbf{z}$ and $\mathbf{A}_S\mathbf{x} \neq \mathbf{A}_S\mathbf{z}$; hence, $\mathbf{A}_S$ is injective, which is condition (1). Now consider the vector $\mathbf{x}$ with support S. At the first iteration of OMP, we choose $j_1 \in S$. This means that for all $\ell \in \bar{S}$,

$$|\mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^0)|_{j_1} > |\mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^0)|_\ell,$$

but since $\mathbf{A}\mathbf{x}^0 = \mathbf{0}$, because $\mathbf{x}^0 = \mathbf{0}$, and since the first index j_1 always stays in the target support, this implies

$$\max_{j \in S} |(\mathbf{A}^*\mathbf{y})_j| > \max_{\ell \in \bar{S}} |(\mathbf{A}^*\mathbf{y})_\ell|,$$

which is condition (2).

($\Leftarrow$) Now we suppose (1) and (2) hold and that $\mathbf{A}\mathbf{x}^1 \neq \mathbf{y}, ..., \mathbf{A}\mathbf{x}^{s-1} \neq \mathbf{y}$, otherwise there is nothing to do; we will show that for $0 \leq n \leq s$, S^n is a subset of S of size n. This will imply that

$S^s = S$, then (OMP$_2$) will yield $\mathbf{A}\mathbf{x}^s = \mathbf{y}$ and in turn, since $\mathbf{A}_S$ is injective, $\mathbf{x}^s = \mathbf{x}$. We proceed as follows: Given $0 \leq n \leq s-1$, then $S^n \subseteq S$ gives $\mathbf{r}^n = \mathbf{y} - \mathbf{A}\mathbf{x}^n \in \{\mathbf{A}\mathbf{z}, \mathrm{supp}(\mathbf{z}) \subseteq S\}$ so that (2) yields $j_{n+1} \in S$ and $S^{n+1} = S^n \cup \{j_{n+1}\} \subseteq S$ by (OMP$_1$). This inductively proves that $S^n \subseteq S$ for any $0 \leq n \leq s$. Now, given $1 \leq n \leq s-1$, we recall the orthogonality characterization from (OMP$_2$),

$$\langle \mathbf{y} - \mathbf{A}\mathbf{x}^n, \mathbf{A}\mathbf{r}^n \rangle = 0 \text{ whenever } \mathrm{supp}(\mathbf{r}^n) \subseteq S^n,$$

$$\langle \mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^n), \mathbf{r}^n \rangle = 0 \text{ whenever } \mathrm{supp}(\mathbf{r}^n) \subseteq S^n,$$

$$(\mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^n))_{S^n} = (\mathbf{A}^*\mathbf{r}^n)_{S^n} = 0.$$

Thus, by (OMP$_1$), the index j_{n+1} does not lie in S^n, otherwise $\mathbf{A}^*\mathbf{r}^n = \mathbf{0}$ so that $\mathbf{r}^n = \mathbf{0}$ from (2). Hence, we have inductively proven that S^n is a set of size $|n|$. The completes the proof. $\square$

An alternate way to formulate the necessary and sufficient condition of Proposition 12 is the so-called *exact recovery condition* (See Remark 3.6 of [Foucart and Rauhut 2013]), which is

$$\|\mathbf{A}_S^\dagger \mathbf{A}_{\bar{S}}\|_{1\to 1} < 1. \tag{2.10}$$

The existence of $\mathbf{A}_S^\dagger = (\mathbf{A}_S^*\mathbf{A}_S)^{-1}\mathbf{A}_S^*$ is equivalent to the injectivity of $\mathbf{A}_S$, which is condition (1). Additionally, condition (2) is equivalent to

$$\|\mathbf{A}_S^*\mathbf{A}_S\mathbf{u}\|_\infty > \|\mathbf{A}_{\bar{S}}^*\mathbf{A}_S\mathbf{u}\|_\infty \text{ for all } \mathbf{u} \in \mathbb{C}^s \setminus \{0\}.$$

Substituting $\mathbf{v} = \mathbf{A}_S^*\mathbf{A}_S\mathbf{u}$ into the above expression yields

$$\|\mathbf{v}\|_\infty > \|\mathbf{A}_{\bar{S}}^*\mathbf{A}_S(\mathbf{A}_S^*\mathbf{A}_S)^{-1}\mathbf{v}\|_\infty = \|\mathbf{A}_{\bar{S}}^*(\mathbf{A}_S^\dagger)^*\mathbf{v}\|_\infty \text{ for all } \mathbf{v} \in \mathbb{C}^s \setminus \{0\}.$$

Thus, we have $\|\mathbf{A}_{\bar{S}}^*(\mathbf{A}_S^\dagger)^*\|_{\infty\to\infty} < 1$, i.e. $\|\mathbf{A}_S^\dagger\mathbf{A}_{\bar{S}}\|_{1\to 1} < 1$.

The following theorem illustrates the applicability of the coherence.

Theorem 13. (See Corollary 5.4 and Theorem 5.14 of [Foucart and Rauhut 2013]) Given $\mathbf{A} \in \mathbb{C}^{m\times N}$ with ℓ_2-normalized columns, if $\mu < \dfrac{1}{2s-1}$, then every s-sparse vector $\mathbf{x} \in \mathbb{C}^N$ is exactly recovered from $\mathbf{y} = \mathbf{A}\mathbf{x}$ after at most s iterations of OMP.

Proof. We need to prove that for any S with $\mathrm{card}(S) = s$ conditions (1) and (2) of Proposition 12 hold. We note that $(\mathbf{A}^*\mathbf{r})_j = \langle \mathbf{A}^*\mathbf{r}, \mathbf{e}_j\rangle = \langle \mathbf{r}, \mathbf{A}\mathbf{e}_j\rangle = \langle \mathbf{r}, \mathbf{a}_j\rangle$, so that condition (2) can be expressed as:

$$(2) \qquad \max_{j\in S} |\langle \mathbf{r}, \mathbf{a}_j\rangle| > \max_{\ell\in \bar{S}} |\langle \mathbf{r}, \mathbf{a}_\ell\rangle| \text{ for all } \mathbf{r} \in \{\mathbf{A}\mathbf{z}, \mathrm{supp}(\mathbf{z}) \subseteq S\} \setminus \{\mathbf{0}\}.$$

We prove condition (2) as follows: Let $\mathbf{a}_1, ..., \mathbf{a}_N$ denote the ℓ_2-normalized columns of $\mathbf{A}$, let $\mathbf{r} := \sum_{i\in S} r_i \mathbf{a}_i$ and choose $k \in S$ so that $|r_k| = \max_{i\in S} |r_i| > 0$. Then for $\ell \in \bar{S}$ we have

$$|\langle \mathbf{r}, \mathbf{a}_\ell\rangle| = \sum_{i\in S} |\langle r_i \mathbf{a}_i, \mathbf{a}_\ell\rangle| = \sum_{i\in S} |r_i|\,|\langle \mathbf{a}_i, \mathbf{a}_\ell\rangle| \leq \mu \sum_{i\in S} |r_i|.$$

Additionally, we have

$$\begin{aligned} |\langle \mathbf{r}, \mathbf{a}_k\rangle| &= \Big|\sum_{i\in S} r_i \langle \mathbf{a}_i, \mathbf{a}_k\rangle\Big| = \Big|r_k + \sum_{i\in S, i\neq k} r_i \langle \mathbf{a}_i, \mathbf{a}_k\rangle\Big| \\ &\geq |r_k| - \Big|\sum_{i\in S, i\neq k} r_i \langle \mathbf{a}_i, \mathbf{a}_k\rangle\Big| \geq |r_k| - \sum_{i\in S, i\neq k} \mu |r_i| \\ &= |r_k| - \mu \sum_{i\in S} |r_i| + \mu |r_k| = |r_k|(1+\mu) - \mu \sum_{i\in S} |r_i|. \end{aligned}$$

We note that we will have $|\langle \mathbf{r}, \mathbf{a}_\ell\rangle| \leq |\langle \mathbf{r}, \mathbf{a}_k\rangle|$ when

$$\mu \sum_{i\in S} |r_i| \leq |r_k|(1+\mu) - \mu \sum_{i\in S} |r_i|,$$

$$2\mu \sum_{i\in S} |r_i| \leq |r_k|(1+\mu).$$

However,

$$2\mu \sum_{i\in S} |r_i| \leq 2\mu |r_k| s,$$

so we need $2\mu s < 1 + \mu$, or equivalently $\mu(2s-1) < 1$, which yields $\mu < \dfrac{1}{2s-1}$ as desired. $\square$

We omit the proof of condition (1) for the sake of brevity. One approach utilizes the introduction of the ℓ_1-coherence function, $\mu_1(s)$, and an additional theorem which bounds the eigenvalues of the matrix $\mathbf{A}_S^*\mathbf{A}_S$ by way of Gershgorin's disc theorem in terms of this new function. The invertibility follows from verifying that the smallest eigenvalue of $\mathbf{A}_S^*\mathbf{A}_S$ satisfies $\lambda_{\min} > 0$ and the use of inequalities relating $\mu, \mu_1(s),$ and $\mu_1(s-1)$.

We note that the successful recovery of all vectors supported on a set S via $\operatorname{card}(S)$ steps of orthogonal matching pursuit also ensures the recovery of all vectors with support S via basis pursuit. This is due in part to the fact that for $\mathbf{v} \in \ker \mathbf{A} \setminus \{0\}$, then $\mathbf{A}_S\mathbf{v}_S = -\mathbf{A}_{\bar{S}}\mathbf{v}_{\bar{S}}$, where, again, $\mathbf{A}_S$ is the submatrix of $\mathbf{A}$ restricted to the columns indexed by the support S and $\mathbf{v}_S$ is the vector $\mathbf{v} \in \mathbb{C}^{\operatorname{card}(S)}$ restricted to the elements on the support of S. Hence,

$$\|\mathbf{v}_S\|_1 = \|\mathbf{A}_S^\dagger \mathbf{A}_S \mathbf{v}_S\|_1 = \|\mathbf{A}_S^\dagger \mathbf{A}_{\bar{S}} \mathbf{v}_{\bar{S}}\|_1 \le \|\mathbf{A}_S^\dagger \mathbf{A}_{\bar{S}}\|_{1\to 1} \|\mathbf{v}_{\bar{S}}\|_1 < \|\mathbf{v}_{\bar{S}}\|_1.$$

Here we have used the *exact recovery condition*, $\|\mathbf{A}_S^\dagger \mathbf{A}_{\bar{S}}\|_{1\to 1} < 1$, of equation (2.10). Thus, the null space property relative to S, equation (2.3), holds and, in turn, Theorem 5 applies.

Although not proven here, one limitation of the coherence is a lower bound known as the *Welch bound*, see Theorem 5.7 of [Foucart and Rauhut 2013]: For a matrix $\mathbf{A} \in \mathbb{C}^{m\times N}$ with ℓ_2-normalized columns, the coherence of $\mathbf{A}$ is bounded below by

$$\mu \ge \sqrt{\frac{N-m}{m(N-1)}}.$$

Due to the requirement that $m < N$, the construction of matrices $\mathbf{A} \in \mathbb{C}^{m\times N}$ which meet the *Welch bound* proves quite difficult. See Chapter 3 for one such construction which nearly matches this bound in application to radar. Furthermore, if we consider the inequality of Theorem 13, which was $\mu < \dfrac{1}{2s-1}$, and note that in the case of large N, the expression for the Welch bound behaves as $\dfrac{1}{\sqrt{m}}$, then, combining these inequalities and rearranging the terms leads to $m \ge Cs^2$, where C is a constant. This indicates that one can minimize the coherence of a matrix by taking m on the order of the square of the sparsity. Hence, while the coherence is straightforward to calculate and matrix constructions exist with near minimal coherence, as previously mentioned, and as we shall see, this order of measurements is not optimal.

Finally, we present the *Restricted Isometry Property.*

Definition 14. (See Definition 6.1 of [Foucart and Rauhut 2013]) The sth restricted isometry constant $\delta_s = \delta_s(\mathbf{A})$ of a matrix $\mathbf{A} \in \mathbb{C}^{m\times N}$ is the smallest $\delta \geq 0$ such that

$$(1-\delta)\|\mathbf{x}\|_2^2 \leq \|\mathbf{A}\mathbf{x}\|_2^2 \leq (1+\delta)\|\mathbf{x}\|_2^2 \tag{2.11}$$

holds for all s-sparse vectors $\mathbf{x} \in \mathbb{C}^N$. An equivalent formulation is given by

$$\delta_s = \max_{S\subseteq[N],\mathrm{card}(S)\leq s} \|\mathbf{A}_S^*\mathbf{A}_S - \mathbf{Id}\|_{2\to 2}. \tag{2.12}$$

The matrix $\mathbf{A}$ satisfies the restricted isometry property if δ_s is small for sufficiently large s.

From the definition, we observe that the restricted isometry constant of order s considers all s-tuples of the columns of $\mathbf{A}$ and compare this to the coherence, which only involves pairs of columns of $\mathbf{A}$; thus, the restricted isometry constant is more general and better suited to assess the quality of the measurement matrix. This transfer from pairs to s-tuples, however, bears an obvious trade-off as the restricted isometry property can quickly become arduous to verify as this is an NP-hard problem. We note that $\delta_1 = 0$ since $\|\mathbf{A}\mathbf{e}_j\|_2^2 = \|\mathbf{e}_j\|_2^2$ for all $j \in [N]$ when $\mathbf{A}$ has ℓ_2-normalized columns. Additionally, one can also show that $\delta_2 = \mu$ and $\delta_s \leq (s-1)\mu$. Equation (2.12) indicates that each column submatrix $\mathbf{A}_S$ with $S \subseteq [N]$ and card$(S)\leq s$ has singular values close to 1, i.e., within the interval $[1-\delta_s, 1+\delta_s]$, and is therefore injective for $\delta_s < 1$.

Unfortunately, deterministic constructions of matrices whose restricted isometry constants demonstrably satisfy $\delta_s \leq \delta$ with a minimal number of measurements m are unknown. However, a collection of random matrices, to be introduced shortly, can be shown to satisfy such conditions with high probability. The following theorem highlights the significance of the restricted isometry property.

Theorem 15. (See Theorem 6.9 of [Foucart and Rauhut 2013]) Suppose that the 2sth restricted isometry constant of the matrix $\mathbf{A} \in \mathbb{C}^{m\times N}$ satisfies

$$\delta_{2s} < \frac{1}{3}. \tag{2.13}$$

Then every s-sparse vector $\mathbf{x} \in \mathbb{C}^N$ is the unique solution of

$$\min_{\mathbf{z}\in\mathbb{C}^N} \|\mathbf{z}\|_1 \text{ subject to } \mathbf{Az} = \mathbf{Ax}.$$

The proof of Theorem 15 will require the following two ingredients.

Lemma 16. (See Lemma 6.10 of [Foucart and Rauhut 2013]) For $q > p > 0$, if $\mathbf{u} \in \mathbb{C}^s$ and $\mathbf{v} \in \mathbb{C}^t$ satisfy

$$\max_{i\in[s]} |u_i| \le \min_{j\in[t]} |v_j|, \tag{2.14}$$

then

$$\|\mathbf{u}\|_q \le \frac{s^{1/q}}{t^{1/p}} \|\mathbf{v}\|_p, \tag{2.15}$$

and in particular when $p = 1$, $q = 2$ and $s = t$,

$$\|\mathbf{u}\|_2 \le \frac{1}{\sqrt{s}} \|\mathbf{v}\|_1. \tag{2.16}$$

Proof. Simply observe

$$\frac{\|\mathbf{u}\|_q}{s^{1/q}} = \left[\frac{1}{s}\sum_{i=1}^{s} |u_i|^q\right]^{1/q} \le \max_{1\le i\le s} |u_i|,$$

and

$$\frac{\|\mathbf{v}\|_p}{t^{1/p}} = \left[\frac{1}{t}\sum_{j=1}^{t} |v_j|^p\right]^{1/p} \ge \min_{1\le j\le t} |v_j|,$$

then with (2.14) the result of (2.15) is immediate. □

Proposition 17. (See Proposition 6.3 of [Foucart and Rauhut 2013]) Let $\mathbf{u}, \mathbf{v} \in \mathbb{C}^N$ be vectors such that $\|\mathbf{u}\|_0 \leq s$ and $\|\mathbf{v}\|_0 \leq t$. If they have disjoint supports, i.e. supp($\mathbf{u}$) $\cap$ supp($\mathbf{v}$)$= \emptyset$, then

$$|\langle \mathbf{Au}, \mathbf{Av} \rangle| \leq \delta_{s+t} \|\mathbf{u}\|_2 \|\mathbf{v}\|_2. \tag{2.17}$$

Proof. Let $S :=$ supp($\mathbf{u}$) $\cup$ supp($\mathbf{v}$), then since $\mathbf{u}$ and $\mathbf{v}$ have disjoint supports $\langle \mathbf{u}_S, \mathbf{v}_S \rangle = 0$. Hence, we have

$$\begin{aligned} |\langle \mathbf{Au}, \mathbf{Av} \rangle| &= |\langle \mathbf{A}_S \mathbf{u}_S, \mathbf{A}_S \mathbf{v}_S \rangle - \langle \mathbf{u}_S, \mathbf{v}_S \rangle| = |\langle (\mathbf{A}_S^* \mathbf{A}_S - \mathbf{Id}) \mathbf{u}_S, \mathbf{v}_S \rangle| \\ &\leq \|(\mathbf{A}_S^* \mathbf{A}_S - \mathbf{Id}) \mathbf{u}_S\|_2 \|\mathbf{v}_S\|_2 \leq \|\mathbf{A}_S^* \mathbf{A}_S - \mathbf{Id}\|_{2 \to 2} \|\mathbf{u}_S\|_2 \|\mathbf{v}_S\|_2. \end{aligned}$$

Applying (2.12) and noting that $\|\mathbf{u}_S\|_2 = \|\mathbf{u}\|_2$ and $\|\mathbf{v}_S\|_2 = \|\mathbf{v}\|_2$ the proof is complete. $\square$

Now we prove Theorem 15.

Proof. We will prove that $\mathbf{A}$ satisfies the null space property of order s, but in a slightly alternate form:

$$\|\mathbf{v}_S\|_1 < \frac{1}{2}\|\mathbf{v}\|_1 \text{ for all } \mathbf{v} \in \ker \mathbf{A} \setminus \{0\} \text{ and all } S \subseteq [N] \text{ with } \operatorname{card}(S) = s.$$

This results from simply adding $\|\mathbf{v}_S\|_1$ to both sides of (2.3). We will actually show the following stronger statement:

$$\|\mathbf{v}_S\|_2 \leq \frac{\rho}{2\sqrt{s}}\|\mathbf{v}\|_1 \text{ for all } \mathbf{v} \in \ker \mathbf{A} \setminus \{0\} \text{ and all } S \subseteq [N] \text{ with } \operatorname{card}(S) = s,$$

where

$$\rho := \frac{2\delta_{2s}}{1 - \delta_{2s}}$$

satisfies $\rho < 1$ when $\delta_{2s} < 1/3$. We note that this inequality results from the observation that $\|\mathbf{v}_S\|_1 \leq \sqrt{s}\|\mathbf{v}_S\|_2$ and then requiring $\sqrt{s}\|\mathbf{v}_S\|_2 \leq \frac{\rho}{2}\|\mathbf{v}\|_1$.

Take $\mathbf{v} \in \ker \mathbf{A} \setminus \{0\}$, and consider an index set $S =: S_0$ of the s largest elements of $\mathbf{v}$ in modulus.

We can then partition the complement $\overline{S_0}$ of S_0 in [N] as $\overline{S_0} = S_1 \cup S_2 \cup ...$, where

$$\begin{aligned} S_1: &\quad \text{index set of s largest elements in modulus of } \mathbf{v} \text{ in } \overline{S_0}, \\ S_2: &\quad \text{index set of s largest elements in modulus of } \mathbf{v} \text{ in } \overline{S_0 \cup S_1}, \end{aligned}$$

etc. Note the last such S_i may contain fewer than s elements, but this will not hinder the proof. Since $\mathbf{v} \in \ker\mathbf{A}$, we have $\mathbf{A}(\mathbf{v}_{S_0}) = \mathbf{A}(-\mathbf{v}_{S_1} - \mathbf{v}_{S_2} - \cdots)$; thus,

$$\begin{aligned} \|\mathbf{v}_{S_0}\|_2^2 \leq \frac{1}{1-\delta_{2s}}\|\mathbf{A}(\mathbf{v}_{S_0})\|_2^2 &= \frac{1}{1-\delta_{2s}}\langle \mathbf{A}(\mathbf{v}_{S_0}), \mathbf{A}(-\mathbf{v}_{S_1}) + \mathbf{A}(-\mathbf{v}_{S_2}) + \cdots \rangle \\ &= \frac{1}{1-\delta_{2s}}\sum_{k\geq 1}\langle \mathbf{A}(\mathbf{v}_{S_0}), \mathbf{A}(-\mathbf{v}_{S_k})\rangle. \end{aligned}$$

Substituting (2.17) of Proposition 17 into this result gives

$$\|\mathbf{v}_{S_0}\|_2 \leq \frac{\delta_{2s}}{1-\delta_{2s}}\sum_{k\geq 1}\|\mathbf{v}_{S_k}\|_2 = \frac{\rho}{2}\sum_{k\geq 1}\|\mathbf{v}_{S_k}\|_2. \tag{2.18}$$

When $k \geq 1$, the elements of $\mathbf{v}_{S_k}$ do not exceed the elements of $\mathbf{v}_{S_{k-1}}$; hence, Lemma 16 gives

$$\|\mathbf{v}_{S_k}\|_2 \leq \frac{1}{\sqrt{s}}\|\mathbf{v}_{S_{k-1}}\|_1.$$

Therefore, we have

$$\|\mathbf{v}_{S_0}\|_2 \leq \frac{\rho}{2\sqrt{s}}\sum_{k\geq 1}\|\mathbf{v}_{S_{k-1}}\|_1 \leq \frac{\rho}{2\sqrt{s}}\|\mathbf{v}\|_1,$$

as desired. □

Next we extend this result to incorporate stable and robust recovery via the quadratically constrained basis pursuit, see Equation (2.7).

Theorem 18. Suppose that the 2sth restricted isometry constant of the matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ satisfies

$$\delta_{2s} < \frac{1}{3}.$$

Then for any $\mathbf{x} \in \mathbb{C}^N$, a solution $\mathbf{x}^*$ of Equation (2.7), with $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$ and $\|\mathbf{e}\|_2 \leq \eta$, approximates the vector $\mathbf{x}$ with ℓ_p error:

$$\|\mathbf{x} - \mathbf{x}^*\|_p \leq \frac{C}{s^{1-1/p}} \sigma_s(\mathbf{x})_1 + D s^{1/p - 1/2} \eta, \qquad 1 \leq p \leq 2, \tag{2.19}$$

where C and D are constants depending on $\rho = \dfrac{\delta_{2s}}{(1-\delta_{2s})}$ and $\tau = \dfrac{\sqrt{1+\delta_{2s}}}{1-\delta_{2s}}$.[1]

Proof. Let $\mathbf{A} \in \mathbb{C}^{m \times N}$ with $2s$th restricted isometry constant $\delta_{2s} < \frac{1}{3}$, let $\mathbf{v} \in \mathbb{C}^N$, and consider a partition of the set $[N] = S_0 \cup S_1 \cup S_2 \cup ...$ as defined in the previous proof. Thus, we must have $\mathbf{A}\mathbf{v}_{S_0} = \mathbf{A}\mathbf{v} - \sum_{k \geq 1} \mathbf{A}\mathbf{v}_k$. We expand $\|\mathbf{v}_{S_0}\|_2^2$ as follows via the RIP and Proposition 17:

$$\begin{aligned}
\|\mathbf{v}_{S_0}\|_2^2 &\leq \frac{1}{1-\delta_{2s}} \|\mathbf{A}\mathbf{v}_{S_0}\|_2^2 = \frac{1}{1-\delta_{2s}} \langle \mathbf{A}\mathbf{v}_{S_0}, \mathbf{A}\mathbf{v} - \mathbf{A}\mathbf{v}_{S_1} - \mathbf{A}\mathbf{v}_{S_2} - \cdots \rangle \\
&= \frac{1}{1-\delta_{2s}} \langle \mathbf{A}\mathbf{v}_{S_0}, \mathbf{A}\mathbf{v} \rangle + \frac{1}{1-\delta_{2s}} \sum_{k \geq 1} \langle \mathbf{A}\mathbf{v}_{S_0}, \mathbf{A}(-\mathbf{v}_{S_k}) \rangle \\
&\leq \frac{1}{1-\delta_{2s}} \|\mathbf{A}\mathbf{v}_{S_0}\|_2 \|\mathbf{A}\mathbf{v}\|_2 + \frac{1}{1-\delta_{2s}} \sum_{k \geq 1} \delta_{2s} \|\mathbf{v}_{S_0}\|_2 \|\mathbf{v}_k\|_2 \\
&= \frac{\delta_{2s}}{1-\delta_{2s}} \|\mathbf{v}_{S_0}\|_2 \sum_{k \geq 1} \|\mathbf{v}_k\|_2 + \frac{\sqrt{1+\delta_{2s}}}{1-\delta_{2s}} \|\mathbf{v}_{S_0}\|_2 \|\mathbf{A}\mathbf{v}\|_2.
\end{aligned}$$

Now we may cancel a $\|\mathbf{v}_{S_0}\|_2$ from both sides, then recognize that by definition $\|\mathbf{v}_{S_1}\|_2 \leq \|\mathbf{v}_{S_0}\|_2$ and by (2.16) of Lemma 16 $\|\mathbf{v}_{S_k}\|_2 \leq \frac{1}{\sqrt{s}} \|\mathbf{v}_{S_{k-1}}\|_2$ for $k \geq 2$. After a rearrangement of terms we arrive at

$$\left(1 - \frac{\delta_{2s}}{1-\delta_{2s}}\right) \|\mathbf{v}_{S_0}\|_2 \leq \frac{1}{\sqrt{s}} \left(\frac{\delta_{2s}}{1-\delta_{2s}}\right) \sum_{k \geq 2} \|\mathbf{v}_{k-1}\|_1 + \frac{\sqrt{1+\delta_{2s}}}{1-\delta_{2s}} \|\mathbf{A}\mathbf{v}\|_2.$$

[1]This result and the proof are left as an exercise for the reader in [Foucart and Rauhut 2013], see Exercise 6.12.

Setting $\rho := \dfrac{\delta_{2s}}{1-2\delta_{2s}} < 1$, so $\delta_{2s} < \dfrac{1}{3}$, and $\tau := \dfrac{\sqrt{1+\delta_{2s}}}{1-2\delta_{2s}} > 0$, since $\delta_{2S} < \dfrac{1}{3}$, we now have

$$\|\mathbf{v}_{S_0}\|_2 \leq \frac{\rho}{\sqrt{s}}\|\mathbf{v}_{\overline{S_0}}\|_1 + \tau\|\mathbf{A}\mathbf{v}\|_2,$$

which matches the ℓ_q-robust null space property of Definition 9 for $q = 2$. Therefore, we can apply Theorem 10 to obtain the desired result. One may consult with [Foucart and Rauhut 2013 Theorem 4.25] for the derivation of the constants $C = \dfrac{(1+\rho)^2}{1-\rho}$ and $D = \dfrac{(3+\rho)\tau}{1-\rho}$. □

We note that the bound of Theorem 15 can be relaxed further to roughly $\delta_{2s} < \dfrac{1}{\sqrt{2}} \approx 0.7071$ and additional bounds can be found for the thresholding and greedy algorithms, such as $\delta_{3s} < \dfrac{1}{\sqrt{3}} \approx 0.5773$ for IHT and HTP and $\delta_{13s} < \dfrac{1}{6}$ for OMP. Furthermore, the $2s$th restricted isometry constant of Theorem 18 can also be relaxed to $\delta_{2s} < .6247$ while also ensuring stability and robustness. A multitude of similar bounds can be derived for various recovery guarantees, see [Foucart and Rauhut 2013 Chapter 6].

2.4 Random Matrices

Random matrices are quite valuable to compressive sensing because under appropriate conditions they can be shown to satisfy the Restricted Isometry Property with high probability. We define one class of such random matrices below.

Definition 19. (See Definition 9.1 of [Foucart and Rauhut 2013]) Consider a matrix $\mathbf{A} \in \mathbb{R}^{m\times N}$ whose elements are random variables. If all entries of $\mathbf{A}$ are independent mean-zero subgaussian random variables with variance 1 and constants $\beta, \theta > 0$ such that

$$\mathbb{P}(|\mathbf{A}_{j,k}| \geq t) \leq \beta \exp(-\theta t^2) \quad \text{for all} \quad t > 0, j \in [m], k \in [N],$$

then $\mathbf{A}$ is called a subgaussian random matrix.

We can also define Guassian matrices matrices as those matrices whose entries are independent standard Gaussian random variables, and similarly Bernoulli matrices are matrices whose entries are independent Rademacher variables. We note that Gaussian and Bernoulli matrices are subgaussian matrices. Next we state, without proof, an essential theorem and a corollary which highlight the importance of such matrices.

Theorem 20. (See Theorem 9.2 of [Foucart and Rauhut 2013]) Let $\mathbf{A}$ be an $m \times N$ subgaussian random matrix. Then there exists a constant $C > 0$ (which only depends upon the subgaussian parameters β, θ) such that the restricted isometry constant of $\frac{1}{\sqrt{m}}\mathbf{A}$ satisfies $\delta_s \leq \delta$ with probability at least $1 - \epsilon$ provided

$$m \geq C\delta^{-2}(s\ln(eN/s) + \ln(2\epsilon^{-1})).$$

Corollary 21. (See Corollary 9.3 of [Foucart and Rauhut 2013]) Let $\mathbf{A}$ be an $m \times N$ subgaussian random matrix. Let $s < N, \epsilon \in (0, 1)$ such that

$$m \geq C_1 s\ln(eN/s) + C_2\ln(2\epsilon^{-1}),$$

for some constants $C_1, C_2 > 0$ only depending on the subgaussian parameters β, θ. Then with probability of at least $1 - \epsilon$ every s-sparse vector $\mathbf{x}$ is recovered from $\mathbf{y} = \mathbf{Ax}$ via ℓ_1-minimization.

Notice that for $\epsilon = 2\exp(-m/(2C_2))$, Corollary 21 ensures recovery of all s-sparse vectors via ℓ_1-minimization with probability at least $1 - 2\exp(-m/(2C_2))$ using a subgaussian random matrix provided

$$m \geq 2C_1 s\ln(eN/s).$$

This condition can be shown to be optimal using *Gelfand Widths*, in the following sense: Given a matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ with $2s$th restricted isometry constant $\delta_{2s} < 1/3$, say, then $\mathbf{A}$ must have a number of rows $m \geq cs\ln(eN/s)$, for some constant $c > 0$ depending only on δ_{2s}, see [Foucart and Rauhut 2013 Chapters 10 and 11]. We also observe that this result improves upon the number of measurements needed when relying upon the coherence of the matrix to guarantee sparse recovery. The proof for these results stems from combining a concentration inequality, a covering argument for unit balls and an upper bound on $\binom{N}{s}$, see [Foucart and Rauhut 2013 Chapter 9]. Fortunately, as one may by now expect, these results maybe extended to guarantee stability and robustness in the reconstruction.

Theorem 22. (See Theorem 9.13 of [Foucart and Rauhut 2013]) Let $\mathbf{A}$ be an $m \times N$ subgaussian random matrix. Then there exists constants $C_1, C_2 > 0$ (which only depend upon the subgaussian parameters β, θ) and universal constants $D_1, D_2 > 0$ such that if, for $\epsilon \in (0,1)$,

$$m \geq C_1 s \ln(eN/s) + C_2 \ln(2\epsilon^{-1}),$$

then the following statement holds with probability at least $1-\epsilon$ uniformly for all vectors $\mathbf{x} \in \mathbb{C}^N$: given $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$ with $\|\mathbf{e}\|_2 \leq \sqrt{m}\eta$ for some $\eta \geq 0$, a solution $\mathbf{x}^\sharp$ of

$$\underset{\mathbf{z} \in \mathbb{C}^N}{\text{minimize}} \;\; \|\mathbf{z}\|_1 \;\; \text{subject to} \;\; \|\mathbf{A}\mathbf{z} - \mathbf{y}\|_2 \leq \sqrt{m}\eta. \tag{2.20}$$

satisfies

$$\|\mathbf{x} - \mathbf{x}^\sharp\|_2 \leq \frac{D_1}{s^{1/2}} \sigma_s(\mathbf{x})_1 + D_2 \eta,$$
$$\|\mathbf{x} - \mathbf{x}^\sharp\|_1 \leq D_1 \sigma_s(\mathbf{x})_1 + D_2 \sqrt{s}\eta.$$

The proof of this theorem involves rearranging the optimization problem by dividing through by $\sqrt{m}$ on both sides, then applying Theorem 18 with $p = 1$ and $p = 2$ and Theorem 20. We stress that for the appropriate choice of $\epsilon = 2\exp(-m/(2C_2))$, this result guarantees the stable and robust recovery of all s-sparse vectors via quadratically constrained basis pursuit with high probability when employing an $m \times N$ subgaussian random matrix with

$$m \geq 2C_1 s \ln(eN/s).$$

This number of measurements cannot be improved upon, see [Foucart and Rauhut 2013 Chapter 10]. This result can also be expanded for the greedy and thresholding algorithms of Section 2.2.

2.5 Conclusion

We have introduced much of the language, tools and concepts necessary to utilize compressive sensing in practice and further develop the theory. We will rely upon the algorithms presented in this chapter, and their modifications, in our radar experiments and our work in structured compressive sensing, while the theoretical results will be echoed in our findings in Chapter 4. The guarantees provided by coherence and the RIP are highly sought after in practice, but we must stress that such assurances of stable and robust recovery are difficult to attain when the conditions of a physical problem prevent the direct use of such random matrices as described above. For example, since random matrices provide such guarantees with high (in fact often a very high) probability, when a system is built according to such design, one has to hope the technology will perform as intended. These considerations are not meant to impede progress in this field, but rather to recognize that compressive sensing is not a panacea. This field requires a great attention to detail and careful analysis to properly manifest growth and improvement over existing practices.

Chapter 3: Application to Radar

We explore the feasibility of applying the framework of compressive sensing to MIMO radar systems. We assess the landscape of recent work in this area of research and explain how our experiments integrate elements from select publications. After providing a primer on radar systems, we mathematically formulate our investigation via a three-fold azimuth-delay-Doppler discretization for target detection and parameter estimation. We utilize a co-located random sensor array and transmit distinct linear chirps to a small scene with few, slowly moving targets. Relying upon standard far-field and narrowband assumptions, we analyze the efficacy of various recovery algorithms in determining the parameters of the scene through numerical simulations, with particular focus on the ℓ_1-squared Nonnegative Regularization method. We detail the various specifications of our experiments and breakdown the results.

3.1 Introduction

Multiple-input multiple-output (MIMO) radar systems have garnered significant interest in recent years for the purpose of accurately detecting targets. These systems incorporate multiple antennas to transmit signals to a target scene and receive and process the reflected echoes. Depending on the positioning of the antennas, i.e., widely separated or co-located, they can provide enhanced target detection and parameter estimation. In particular, a co-located MIMO radar system which transmits waveforms with distinct frequencies can yield improved spatial resolution over similar setups, such as phased-array radar systems. See [Li and Stoica 2009] for a detailed analysis of MIMO radar. Figure 3.1 illustrates the basic setup of a MIMO radar system with a discretization of the target scene in azimuth (angle) and radial range (time delay).

As discussed in Chapter 1, the advancements of compressive sensing have attracted wide-spread attention as a means of efficiently recovering sparse (or compressible) signals. We previously illustrated in Chapter 2 the key finding that a random measurement matrix $\mathbf{A} \in \mathbb{C}^{m \times N}$ with

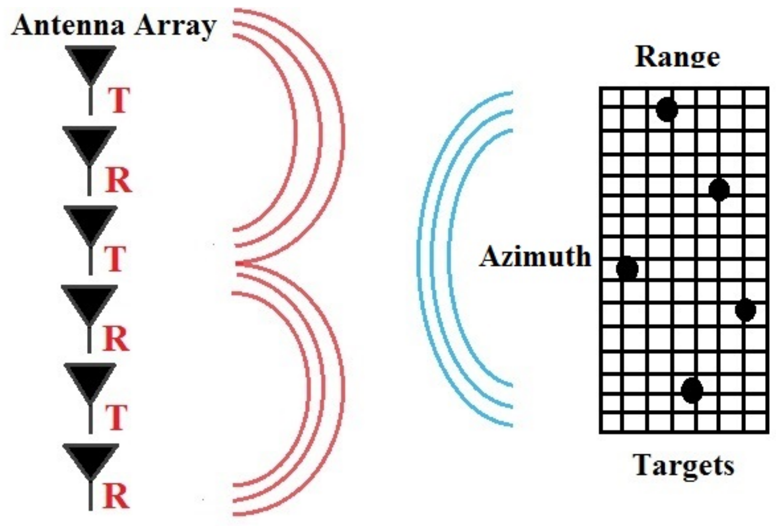


Figure 3.1: Simple diagram of a MIMO radar system setup. Each of the transmit antennas sends a signal to the scene. These signals reflect off of each target and return to each of the receiver antennas. In this example the scene has been discretized in azimuth (angle) and range (delay).

$m \geq Cs\ln(N/s)$, where C is a constant and s is the sparsity of the signal, will fulfill the *Restricted Isometry Property* with high probability. Once more, we note the significance of this result as it greatly reduces the minimal number of measurements needed to recover any s-sparse signal $\mathbf{x} \in \mathbb{C}^N$, particularly as compared to the Nyquist-rate. The signal $\mathbf{x}$ can be reconstructed by solving the convex optimization problem called *basis pursuit* or *ℓ_1-minimization*, see Equation (2.2), or in the more pertinent case of noisy measurements via the *quadratically constrained basis pursuit*, see Equation (2.7).

The efficacy of compressive sensing hinges on the sparsity, or compressibility, of the signal one seeks to recover. Fortunately the discretization of a target scene containing only a few point scatterers in the desired domain, such as angle, time-delay and Doppler, leads to an advantageous sparsity constraint since only a few of the bins in the domain, i.e. possible locations in the scene, contain a target in comparison to the total number of bins. Thus, techniques from compressive sensing have the potential to reduce costs without degrading resolution in detecting the reflectors. A great deal of research has been carried out in investigating the applicability of compressive sensing to a wide variety of radar systems and various aspects of radar signal processing. See Chapter 1 for such examples and [Ender 2010] and [Potter et al. 2010] for surveys of this field. We discuss two early works in this area below.

Perhaps the earliest attempt to apply the results of compressive sensing to radar technologies was presented in [Baraniuk and Steeghs 2007]. The authors highlight inefficiencies in current radar systems to compress a sparse signal: the full signal must be captured at the Nyquist sampling rate, then a complete set of coefficients must be calculated in order to represent the signal in an appropriate sparsifying basis, e.g. $\mathbf{x} = \sum_i \theta_i \boldsymbol{\psi}_i$ where θ_i are coefficients for the basis vectors $\boldsymbol{\psi}_i$. Next, the s largest, when $\mathbf{x}$ is at most s-sparse, coefficients must be located and stored while the rest of the coefficients are discarded. They note that for wideband signals, the Nyquist rate will demand a large number of samples resulting in a heavier cost for computing the θ_i even though most of these values will be thrown away. Furthermore, the locations of the s largest coefficients will change with each signal; hence, they must be recalculated with each new signal. All of this results in a vast amount of information to process, which is further limited by current Analog to Digital conversion technologies and the need for a matched filter to correlate the received signals.

The method they propose to overcome these inefficiencies is as follows: Find a representation to express the received signal in a sparse or compressible manner, e.g. delta functions for a set of point targets or a wavelet expansion for smooth targets. Thus, the signal takes the form $\mathbf{x} = \sum_i \theta_i \boldsymbol{\psi}_i = \boldsymbol{\Psi}\boldsymbol{\theta}$, where $\boldsymbol{\Psi} \in \mathbb{C}^{N \times N}$ is a matrix of the set of basis vectors $\boldsymbol{\psi}_i$, and the vector $\boldsymbol{\theta} \in \mathbb{C}^N$ gives information about the target or scene, such as range, velocity, etc. When considering a simplified 1D range problem they choose the basis vectors to be delta spikes and information about the range of target is stored in $\boldsymbol{\theta}$. Next, compute measurements of the signal via $\mathbf{y} = \boldsymbol{\Phi}\mathbf{x}$, where $\boldsymbol{\Phi} \in \mathbb{C}^{m \times N}$ is a suitable measurement matrix and m is taken such that the new matrix $\boldsymbol{\Phi}\boldsymbol{\Psi}$ will satisfy the RIP with high probability. Since time-translated and frequency-modulated versions of the transmitted chirp signals form a dictionary that is *incoherent* with many of the representation bases, i.e., the product of the matrices will have a small coherence, they serve as good candidates for the rows of the sampling matrix. They also note the applicability of random matrices, in particular a quasi-Toeplitz matrix whose entries are a pseudorandom noise sequence of independent and identically distributed (i.i.d.) Bernoulli or i.i.d. Gaussian measurements, to satisfy the RIP for different transmitted radar waveforms, as opposed to just chirp signals. By incorporating both of these steps, they propose the design of new radar technology without a matched filter and with an Analog to Digital conversion which operates at a sub-Nyquist rate proportional to the target's compressibility. They demonstrate the feasibility of their approach with a test case by recovering a sparse scene of point targets via OMP while sampling at only half the Nyquist rate.

A separate proposal for a new 'stylized' compressive sensing radar was described in [Strohmer and Friedlander 2009]. Whereas [Baraniuk and Steeghs 2007] stressed the details of implementation and hardware in the case of stationary targets, the authors of [Herman and Strohmer 2009] not only consider moving targets but also provide a rigorous treatment of the mathematics needed to increase the resolution of radar imaging systems via compressive sensing. The central component of their analysis hinges upon the use of the *Alltop* vector, defined for prime $m \geq 5$ as

$$f_\ell = e^{2\pi i \ell^3/m}, \qquad \ell \in [m],$$

as appropriate waveforms to exploit the sparsity of a scene in the time-frequency domain. They combine this approach with a collection of $m \times m$ matrices, known as the translation and modulation operators,

$$T = \begin{pmatrix} 0 & & & 1 \\ 1 & 0 & & \\ & \ddots & \ddots & \\ 0 & & 1 & 0 \end{pmatrix} \text{ and } M = \begin{pmatrix} \omega_m^0 & & & 0 \\ & \omega_m^1 & & \\ & & \ddots & \\ 0 & & & \omega_m^{m-1} \end{pmatrix},$$

where $\omega_m = e^{2\pi i/m}$ is the m-th root of unity, to form a *Gabor frame* $\mathbf{\Phi} \in \mathbb{C}^{m \times m^2}$. This explicit construction leads to a matrix with coherence $\mu = \frac{1}{\sqrt{m}}$, which is only slightly above the *Welch bound*, see Chapter 2; hence, they are able to present guarantees on the recovery of a sparse (or compressible) target vector $\mathbf{x}$ from $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$.

The authors go on to note that classical radar systems are limited in their capacity to resolve distinct targets which are close together when represented on a discretized time-frequency grid due to the radar uncertainty principle. A simplified form of this principle is as follows: a waveform with good range (time-delay) resolution will have poor Doppler (frequency) resolution and a waveform with good Doppler resolution will have poor range resolution. If a classical radar system utilized the *Alltop* vector to reconstruct such a scene, a matched filter would be employed to correlate the received signal; however, even in the case of no noise, a great deal of interference will arise due to the uncertainty principle. The presented 'stylized' compressive sensing radar improves resolution by discarding the need for a matched filter while still sampling at the Nyquist rate.

Next, we highlight two recent publications which serve as the basis for this work. The researchers of [Strohmer and Wang 2013] provide an excellent mathematical framework which incorporates Compressive Sensing for the recovery of on-grid targets in azimuth-range-Doppler via a MIMO radar system. They employ random sensor arrays and special waveforms, namely the so-called Kerdock waveforms, in their setup and present a detailed mathematical analysis on the accurate detection of targets in such a setting. The target vector representing the scene is recovered via the *Debiased LASSO*, which is a variation of the well-known *LASSO*, from a number of measurements corresponding to the product of the number of receivers and the number of time samples taken.

The authors of [He et al. 2013] devise an adaptive procedure to detect off-grid targets in azimuth and range with a MIMO radar system which features a uniform linear array (ULA), i.e., the transmit and receive antennas are placed on an axis so that the distance between antennas is uniform and based on the reference carrier frequency, and transmits linear chirps with distinct frequencies. They highlight the performance of their algorithm with numerical simulations and compare the results to alternative recovery methods. These authors rely upon the orthogonality of their waveforms to obtain a number of measurements equal to the product of the number of transmitters, the number of receivers and the number of samples taken.

Our approach incorporates elements from both of these works. Namely, we utilize a random array MIMO radar system which transmits linear chirps and obtains a number of measurements equal to the product of the number of transmitters, the number of receivers and the number of time samples in order to detect on-grid targets in the azimuth, time-delay and Doppler domain. Various numerical simulations are performed in this framework for a small target scene containing only a few slowly moving point scatterers. We initially utilize several different algorithms for recovery, then focus solely on the ℓ_1-squared Nonnegative Regularization (L1SQNN) from [Foucart and Koslicki 2014] due to its superior performance in comparison to the other selected methods. The choice of the random sensor array in place of a ULA is justified by a set of results directly comparing the two setups. We further analyze how changes in the sparsity level, signal-to-noise ratio (SNR), problem size, regularization parameters and bandwidth can impact the reconstruction. A collection of selected Matlab files for this work has been included in Appendix A.1.

3.2 Radar Background

Radar (short for RAdio Detection And Ranging) is a technology that was originally designed for target detection via radio waves. The basic concept central to radar is the following: generate a pulse at time $t = 0$ with speed c, the pulse will travel until it reflects from an object at range R and return at time τ. Hence, since the pulse travelled a total distance of $2R$, we can determine the location of the target from $R = c\tau/2$.

Since radio waves are able to pass through clouds and rain and in some cases penetrate through foliage and debris, radar systems can operate in all weather. Radar systems do not require light; thus, they can be used during the day or at night, a major advantage over most optical systems. Additionally, since radar waves mostly scatter from objects whose size is on the same order of magnitude as the wavelength, radar imaging can resolve targets whose length scales range from centimeters to meters.

One of the earliest developments in the history of radar occurred on 18th May 1904 at the Hohenzollern Bridge, in Cologne, Germany, with one of Christian Hülsmeyer's earliest public demonstrations of his patented "Telemobiloscope." Designed with collision avoidance in mind, the device alerted observers to the presence of an approaching ship by sounding a bell, but the device did not calculate distance and only had limited range. (People could hear the ships coming before the device detected them!) Further developments in radar technologies were made throughout the 1900s, especially during wartimes with advances later adopted for civilian use. Today, radar technologies have been greatly expanded to numerous applications throughout society, including high resolution imaging, weather forecasts and navigation.

Currently there are numerous types of radar systems depending upon the configuration of transmitting and receiving arrays, the design of the antenna, the chosen scanning method, etc. The basic setup, however, is as follows (Our presentation and notation are adopted from [Cheney and Borden 2009]). An incident signal $s_{inc}(t)$, typically a pulse in fielded systems, is multiplied by a rapidly varying carrier wave with angular frequency ω_0. Thus, the signal acts as an envelope for the carrier wave and the product, called the modulated signal, takes the form

$$p(t) = s_{inc}(t)\cos(\omega_0 t) = Re\{s_{inc}(t)e^{-i\omega_0 t}\}.$$

The modulated signal is often amplified by a transmitter and used to excite currents on the antenna, which then radiates electromagnetic waves. These waves scatter off targets in the scene and return, exciting currents on the antenna once again. Since these new currents generate low time-varying voltages that are often below the threshold of thermal noise in the system, a low-noise amplifier amplifies the signal, which is then filtered to remove the high-frequency of the carrier wave. The demodulated received signal $s_{rec}(t)$ is then processed.

Filtering is done via convolution of the signal with the inverse Fourier transform of a transfer function, in other words, by taking the inverse Fourier transform of the signal, multiplying this with the transfer function and finally taking the Fourier transform of this product. The transfer function is used to filter undesired frequencies, e.g. a low-pass filter is given below

$$\mathcal{H}_b(\nu) = \begin{cases} 1 & \text{if } |\nu| < b, \\ 0 & \text{else.} \end{cases} \tag{3.1}$$

In this case, the filter removes all frequencies outside the band of $(-b, b)$. The inverse Fourier transform of $\mathcal{H}_b$, denoted $h(t)$, is called the impulse response of the filter.

In an idealized setting, the received signal $s_{rec}(t)$ that is reflected from a fixed point target is a time-delayed version of the transmitted signal with noise, which is typically represented by a random process. For example,

$$s_{rec}(t) = \rho s(t-\tau) + n(t). \tag{3.2}$$

Here τ is the 'location' of the target from the radar system, i.e., we can convert the physical radial range distance, R, to the time-delay quantity, τ via $\tau = 2R/c$. The factor ρ results from the fact that the radar waves produced by the antenna experience a decay in power proportional to $1/R^2$ and upon scattering suffer another decay in power proportional to $1/R^2$; hence, the received signals have a total power decay that is proportional to $1/R^4$. We note that ρ gives information about the 'reflectivity' of the target and will depend on its position, velocity and other properties in a non-idealized setting. Due to the loss of power, these signals are often subsumed by thermal noise. A technique to overcome this difficulty, known as pulse compression, utilizes coded pulses and a matched filter to recover the signal. A common choice of waveforms for the pulse is the *chirp*, that is a pulse with constant amplitude but whose instantaneous frequency is a linear function of time. The instantaneous frequency is the time derivative of the pulse's angular frequency. An example is given below:

$$s(t) = e^{i(\omega_0 t + \alpha t^2/2)} I_T(t), \quad \text{so that} \quad \frac{d}{dt}(\omega_0 t + \alpha t^2/2) = \omega_0 + \alpha t.$$

Here $I_T(t)$ is the characteristic function on $[0, T]$, and T is the duration of the pulse. In order to maximize the Signal to Noise Ratio (SNR), which is the ratio of the power of the signal to the power of the noise, one must choose an appropriate filter. This is achieved by taking the impulse response to be $h(t) = s^*(-t)$, which is the matched filter.

The essential concept of pulse compression is as follows: As the pulse is transmitted, different parts are coded with different instantaneous frequencies. When the pulses are received, a matched filter is applied so that different frequencies are delayed by different amounts of time. This concentrates all of the energy so that it is released at the same time and the signal is properly received.

Lastly, we note that when the target is moving, the received signal will exhibit a Doppler shift, so that Equation (3.2) becomes

$$s_{rec}(t) = \rho s(t-\tau)e^{-i\omega_D(t-\tau)} + n(t).$$

When accounting for multiple stationary targets, Equation (3.2) will become a summation of separate signals

$$s_{rec}(t) = \sum_{\tau'} \rho(\tau')s(t-\tau') + n(t),$$

and in the case of moving targets, combining this with the previous equation yields

$$s_{rec}(t) = \sum_{\tau'}\sum_{\omega'} \rho(\tau',\omega')s(t-\tau')e^{-i\omega'(t-\tau')} + n(t).$$

3.3 Problem Formulation

Consider a co-located MIMO radar system with M randomly positioned transmit antennas and N randomly positioned receive antennas in the sensor array. Each of the transmitters repeatedly sends a waveform $s_m(t)$, for $m = 1, ..., M$, which are orthogonal to each other and narrowband, to reflect off P point targets in a far-field scene and return to the N receive antennas. These returning signals are observed over a duration T_d. We discretize the scene in azimuth, delay (radial range) and Doppler (radial velocity) with U angle bins, V delay bins and W Doppler bins and associated discretization steps $\Delta_\theta, \Delta_\tau$, and Δ_υ. Hence, targets located exactly on the grid correspond to a location $(\theta_k, \tau_k, \upsilon_k) = (\theta_{\text{ref}} + \text{u}_{(k)}\Delta_\theta, \tau_{\text{ref}} + \text{v}_{(k)}\Delta_\tau, \upsilon_{\text{ref}} + \text{w}_{(k)}\Delta_\upsilon)$, where $k = 1, ..., K$, with $K = UVW$, $\theta_{\text{ref}}, \tau_{\text{ref}}$, and υ_{ref} are reference values in the respective domains, and where $(\text{u}_{(k)}, \text{v}_{(k)}, \text{w}_{(k)})$ represents the k-th bin in the discretization of the scene. Figure 3.2 provides a representative visual for this setup. These point targets possess nonzero complex reflectivity coefficients x_k and are assumed to be (slow) moving with constant velocities. If there is no target at grid point $(\theta_k, \tau_k, \upsilon_k)$, then the associated $x_k = 0$. We let $S \subset \{1, 2, \ldots, K\}$ denote the locations of the targets, i.e., $S := \{k \in [K] : x_k \neq 0\}$ and $|S| = P$.

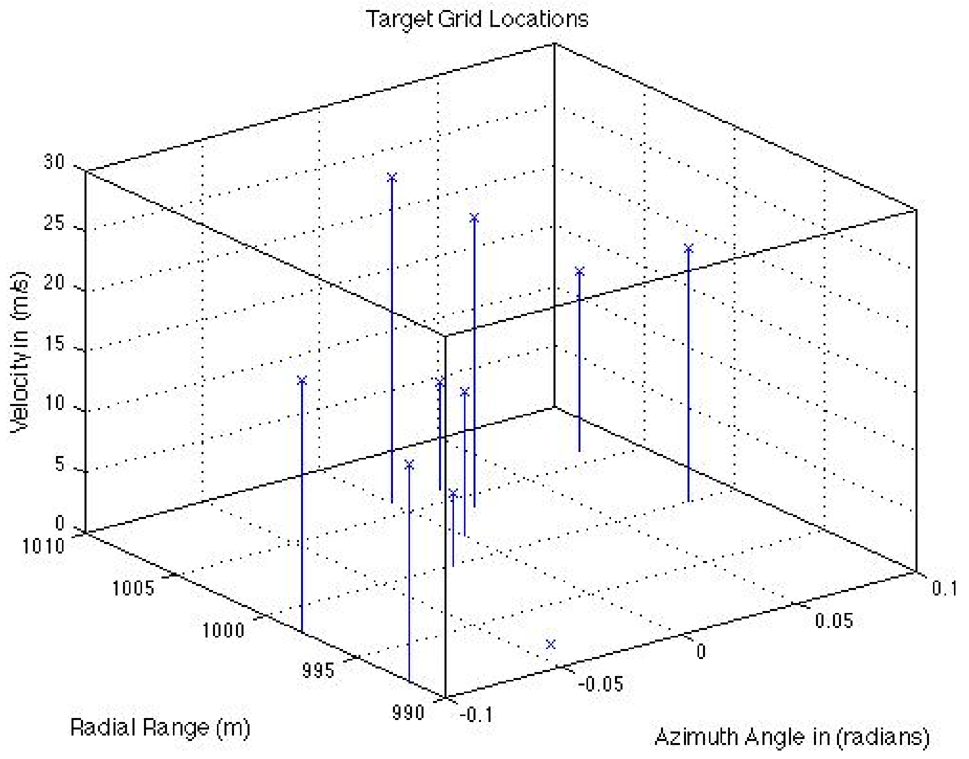


Figure 3.2: Visualization of a representative target scene discretized in range, azimuth and velocity (Doppler). Please note that while this depiction is a rectangular box, in actuality the azimuth axis would curve slightly.

We introduce the array manifolds for the small target scene:

$$\mathbf{a}(\theta_k) = \left[1, e^{i\frac{2\pi}{\lambda}dt_2\theta_k}, ..., e^{i\frac{2\pi}{\lambda}dt_M\theta_k}\right]^T, \text{ and} \tag{3.3}$$

$$\mathbf{b}(\theta_k) = \left[1, e^{i\frac{2\pi}{\lambda}dr_2\theta_k}, ..., e^{i\frac{2\pi}{\lambda}dr_N\theta_k}\right]^T, \tag{3.4}$$

where λ is the reference carrier wavelength, dt_m, for $m = 2, ..., M$, is the distance from the m-th transmitter to the first transmitter, and dr_n, for $n = 2, ..., N$, is the distance from the n-th receiver to the first receiver. (We are using the approximation $\sin(\theta_k) \approx \theta_k$ since we will only consider small angles, i.e. $|\theta_k| < 0.1$ radians.) Hence, under the narrowband assumption and after orthogonal separation, the signal received from the m-th transmitter at the n-th receiver at time t is given by

$$z_{mn}(t) = \sum_{k \in S} x_k b_n(\theta_k) a_m(\theta_k) s_m(t - \tau_k) \exp\left[-i2\pi \upsilon_k t\right] + e_{mn}(t), \tag{3.5}$$

where $b_n(\theta_k)$ is the n-th entry of $\mathbf{b}(\theta_k)$ and $a_m(\theta_k)$ is the m-th entry of $\mathbf{a}(\theta_k)$ and $e_{mn}(t)$ is noise.

We employ Linear Frequency Modulated (LFM) chirps of the form

$$s_m(t) = \exp\left[i2\pi \left(\frac{\alpha}{2} t^2 + f_m t\right)\right] I_T(t), \tag{3.6}$$

where α is the chirp rate, T is the pulse duration, $f_m = f_{\text{ref}} + m\alpha T$ is the carrier frequency for a specified reference carrier frequency f_{ref}, and $I_T(t)$ is the characteristic function on $[0, T]$. Comparing this expression to the equation presented in the previous section, we note that we have transformed from the angular frequency ω via $\omega = 2\pi f$. We suppose that the reference range for the scene is R_{ref}, i.e., the distance from the radar system where the target scene begins; thus, the reference time delay is $\tau_{\text{ref}} = 2R_{\text{ref}}/c$ and for LFM chirps we obtain the following:

$$\begin{aligned} s_m(t - \tau_k) s_m^*(t - \tau_{\text{ref}}) = I_T(t - \tau_{\text{ref}}) I_T(t - \tau_k) \\ \times \exp\left[-i2\pi (f_m + \alpha t')(\tau_k - \tau_{\text{ref}})\right] \\ \times \exp\left[i\pi\alpha(\tau_k - \tau_{\text{ref}})^2\right] \end{aligned} \tag{3.7}$$

where $t' = t - \tau_{\text{ref}}$ and $t' \in [0, T_d]$. The quadratic term is known as the residual video phase and can be removed according to [Carrara et al. 1995]. Thus, after dechirping, the measurements we obtain between the m-th transmitter and the n-th receiver at time t' take the form:

$$\begin{aligned} y_{mn}(t') = \sum_{k=1}^{K} x_k b_n(\theta_k) a_m(\theta_k) \\ \times \exp\left[-i2\pi (f_m + \alpha t')(\tau_k - \tau_{\text{ref}})\right] \\ \times \exp\left[-i2\pi \upsilon_k (t' + \tau_{\text{ref}})\right] + e_{mn}(t). \end{aligned} \tag{3.8}$$

Note that contributions to the summation only result from the P point targets, however, since it is unknown a priori which x_k are nonzero, we sum over all K possible locations. We consider the measurements at times t'_q for $q = 1, ..., Q$. After substituting the appropriate expressions into (3.8), we have

$$\begin{aligned} y_{mn}(t'_q) = \sum_{k=1}^{K} x_k \exp\left[\frac{i2\pi}{\lambda}(dt_m + dr_n)\theta_k\right] & \\ \times \exp\left[-i2\pi(f_m + \alpha t'_q)(\tau_k - \tau_{\text{ref}})\right] & \\ \times \exp\left[-i2\pi\upsilon_k(t'_q + \tau_{\text{ref}})\right] + e_{mn}(t'_q). & \qquad (3.9) \end{aligned}$$

Our objective now is to recover $\{x_k, \theta_k, \tau_k, \upsilon_k\}_{k=1}^{K}$ from the set of $\{y_{mn}(t'_q)\}$, where $m = 1, ..., M$, $n = 1, ..., N$, and $q = 1, ..., Q$. Hence, we introduce the vectorization operation vec$(\cdot)$ and define $\mathbf{y} := \text{vec}\left(y_{mn}(t'_q)\right)$, $\mathbf{e} := \text{vec}\left(e_{mn}(t_q)\right)$, and

$$\begin{aligned} \mathbf{A}_k := \text{vec}\Bigg(\exp\left[i2\pi\left(\frac{dt_m + dr_n}{\lambda}\theta_k - \upsilon_k(t'_q + \tau_{\text{ref}})\right)\right] & \\ \times \exp\left[-i2\pi(f_m + \alpha t'_q)(\tau_k - \tau_{\text{ref}})\right]\Bigg), & \qquad (3.10) \end{aligned}$$

which are all vectors of size $MNQ \times 1$. We store the $\mathbf{A}_k$'s via:

$$\mathbf{A} = \Big(\mathbf{A}_1 \,|\, \mathbf{A}_2 \,|\, \cdots \,|\, \mathbf{A}_K\Big). \qquad (3.11)$$

Letting $\mathbf{x} = [x_1, x_2, \ldots, x_K]^T$, we arrive at the standard Compressive Sensing framework

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}. \qquad (3.12)$$

Here $\mathbf{y}$ is the set of measurements we obtain from our measurement matrix $\mathbf{A}$, the P-sparse target scene $\mathbf{x}$ and the noise vector $\mathbf{e}$. Thus, our goal is to recover $\mathbf{x}$ from $\mathbf{y}$ and $\mathbf{A}$ and in turn estimate θ_k, τ_k, and υ_k associated with each nonzero x_k.

3.4 Numerical Simulations

We are unable to establish the RIP for the measurement matrix $\mathbf{A}$; thus, we perform simulations in Matlab to investigate the efficacy of various reconstruction algorithms in recovering sparse vectors from Equation (3.12) since we lack any theoretical guarantees provided by the RIP. The following parameters remain unchanged throughout the simulations: the reference carrier frequency is $f_{\text{ref}} = 10$ GHz, the bandwidth of each transmitted signal is $B = 15$ MHz, the pulse duration is $T = 2\ \mu$s (hence the chirp rate is $\alpha = 7.5 \times 10^{-12}$ Hz/s), the azimuth angle ranges (in radians) from -0.1 to 0.1, the radial range values are between 990 m and 1010 m, and the target velocities range from 0 m/s to 30 m/s. We consider $M = 5$ transmitters, $N = 5$ receivers, $Q = 20$ time samples, $U = 10$ azimuth bins, $V = 10$ time-delay bins, $W = 10$ Doppler bins, (thus $K = 1000$ bins all together) for the majority of the simulations, but we also double all of these values to explore how increasing the problem size impacts the recovery. We also investigate how the bandwidth affects the reconstruction in the final experiment. For each simulation set the sparsity level of the target varies over some fixed collection of values, a number of measurement matrices are generated for each sparsity level and a specified number of random target vectors are generated for each matrix.

The transmit and receive antenna positions are generated independently according to the uniform distribution on $[0, \frac{MN}{2}]$ as in [Strohmer and Wang 2013], while the locations of the point targets in the scene are chosen iteratively. A location in the azimuth-delay grid is selected at random; if the bin does not already contain a target then a Doppler value is selected at random and the new azimuth-delay-Doppler location is added to the support of the target scene. Otherwise, a new azimuth-delay location is chosen at random and the process repeats until the target vector contains the correct number of point scatterers. (See section 3.6 for further remarks on this and its implications for structured compressive sensing.) The targets are each given a unit reflectivity coefficient. After the measurements are taken, they are corrupted by complex, circularly symmetric Gaussian noise, for a designated SNR level. Recovery is then performed with Orthogonal Matching Pursuit (OMP), Adaptive Inverse Scale Space (AISS) [Burger et al. 2013], ℓ_1-squared Nonnegative Regularization (L1SQNN) [Foucart and Koslicki 2014] and/or ℓ_1/ℓ_2 Constrained Nonnegative Regularization (L1L2CNN) from the YALL1 software package [Zhang et al. 2011]. In the case of no noise, *basis pursuit* with a nonnegativity constraint is selected from YALL1 in place of L1L2CNN. The Orthogonal Matching Pursuit was previously presented in Chapter 2. The other methods are variants on *basis pursuit denoising* [Chen et al. 1998]:

$$\underset{\mathbf{z}\in\mathbb{R}^N}{\text{minimize}}\, \|\mathbf{z}\|_1 + \nu\|\mathbf{A}\mathbf{z}-\mathbf{y}\|_2^2. \tag{3.13}$$

Specifically, the L1SQNN method solves

$$\underset{\mathbf{z}\in\mathbb{R}^N}{\text{minimize}}\, \|\mathbf{z}\|_1^2 + \beta^2\|\mathbf{A}\mathbf{z}-\mathbf{y}\|_2^2 \;\text{ subject to }\; \mathbf{z}\geq 0, \tag{3.14}$$

while the L1L2CNN method solves

$$\begin{aligned}\underset{\mathbf{z}\in\mathbb{R}^N}{\text{minimize}}\;\; \|\mathbf{z}\|_1 \;\text{ subject to }\; &\|\mathbf{A}\mathbf{z}-\mathbf{y}\|_2 \leq \delta \\ \text{and }\; &\mathbf{z}\geq 0,\end{aligned} \tag{3.15}$$

where $\|\mathbf{e}\| \leq \delta$. As the name implies, the AISS approach relies on Inverse Scale Space methods and Bregman Iterations to iteratively solve lower dimension problems in seeking a solution to (2.2). Since we have assumed the point scatterers possess a unit reflectivity, we utilize the nonnegativity constraints in L1SQNN and L1L2CNN; however, for scatterers with complex reflectivity coefficients, variants of these two methods can be employed which drop the nonnegative constraint. While this collection is by no means exhaustive, we selected these algorithms for the following reasons: OMP is commonly used and easy to implement, the YALL1 package is readily available online, and L1SQNN and AISS are both more recently developed methods.

A set of threshold levels is used to zero out the entries of the recovered vector which fall below the specified threshold in magnitude so that all nonzero entries after thresholding are classified as targets in the scene. Throughout the simulations, the following quantities are calculated and averaged for each sparsity level: the probability of detection, the probability of false alarm, the relative error,

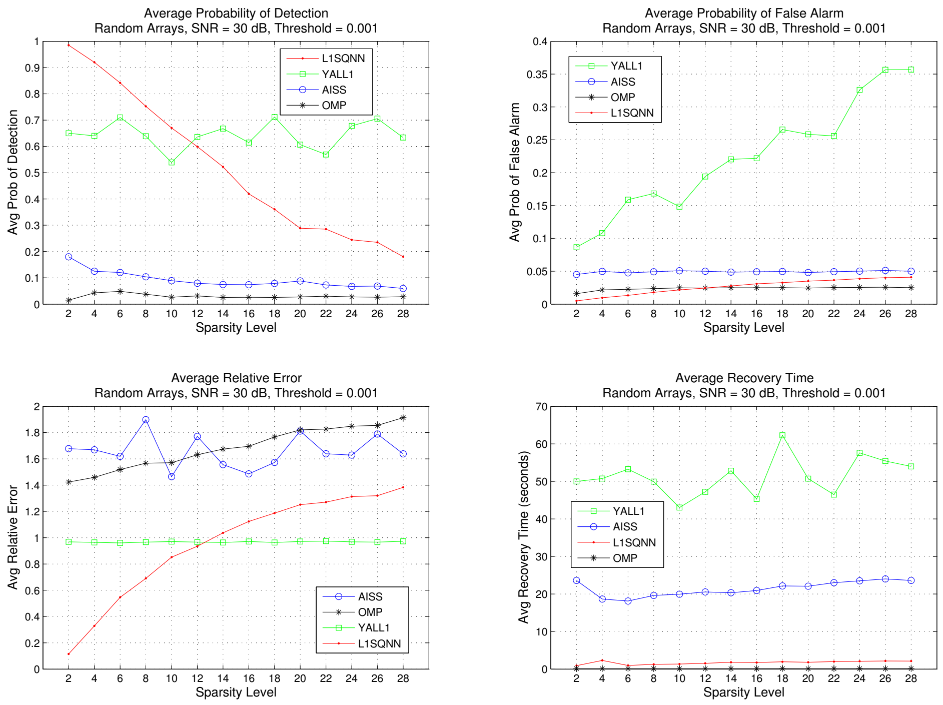


Figure 3.3: Comparison of OMP, AISS, L1SQNN, and L1L2CNN algorithms with fixed SNR = 30 dB and a threshold of 0.001.

and the number of iterations and amount of time required for the algorithms to terminate. The probability of detection is calculated by dividing the number of correctly identified targets, after thresholding, by the number of true targets present in the scene. Similarly, the probability of false alarm is calculated by dividing the number of falsely identified targets, after thresholding, by the number of vacant locations in the scene. The relative error for a recovered vector post-thresholding, denoted $\tilde{\mathbf{x}}$, is simply $\|\tilde{\mathbf{x}} - \mathbf{x}\|_2 / \|\mathbf{x}\|_2$. For several simulation sets we plot Receiver Operating Characteristic (ROC) curves [Richards 2005], i.e., the probability of detection plotted against the probability of false alarm, which illustrate how lowering the threshold level increases both the probability of detection and probability of false alarm.

3.5 Results

The initial set of simulations were designed to provide a rough comparison of the performance of the previously discussed recovery algorithms. The SNR is fixed at 30 dB, a low threshold level of 0.001

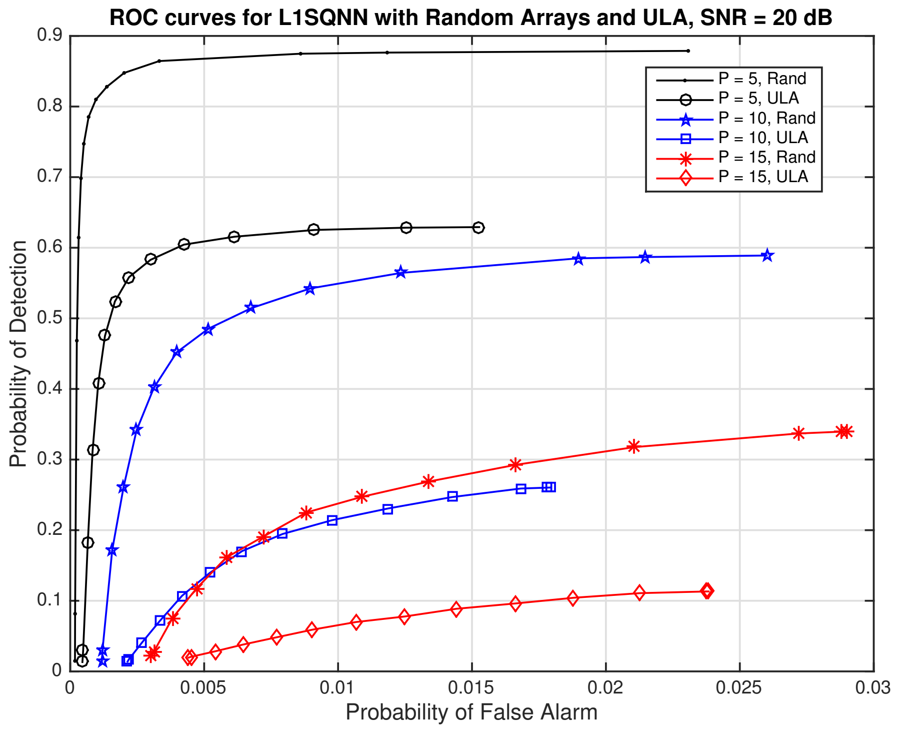


Figure 3.4: Comparison of ROC curves for L1SQNN with random arrays and ULA, SNR = 20 dB.

is selected, and $\beta = 30$ is chosen as the parameter for L1SQNN, while the noise level is used in the L1L2CNN method from YALL1. A total of 100 simulations, i.e., 10 matrices applied to 10 random target vectors, are performed for each sparsity level, which ranged from 2 to 28 with increments of 2. As shown in Fig. 3.3, the L1SQNN algorithm offers superior recovery in comparison to the other three methods in all categories. Although the L1L2CNN algorithm provides a better probability of detection and lower relative error for higher sparsity counts, this benefit comes at the cost of a higher probability of false alarm, by an order of magnitude, and much longer run times on average. Also, while not shown here, separate simulations reveal that the AISS method outperforms the L1SQNN method in the case of no noise, though it also requires a longer run time. Thus, due to the enhanced performance, we focus on the L1SQNN algorithm for the remaining simulations.

Next we consider both the random array and uniform linear array MIMO radar setups to justify the use of the random sensor arrays. The ULA system has receiver antennas positioned in a line

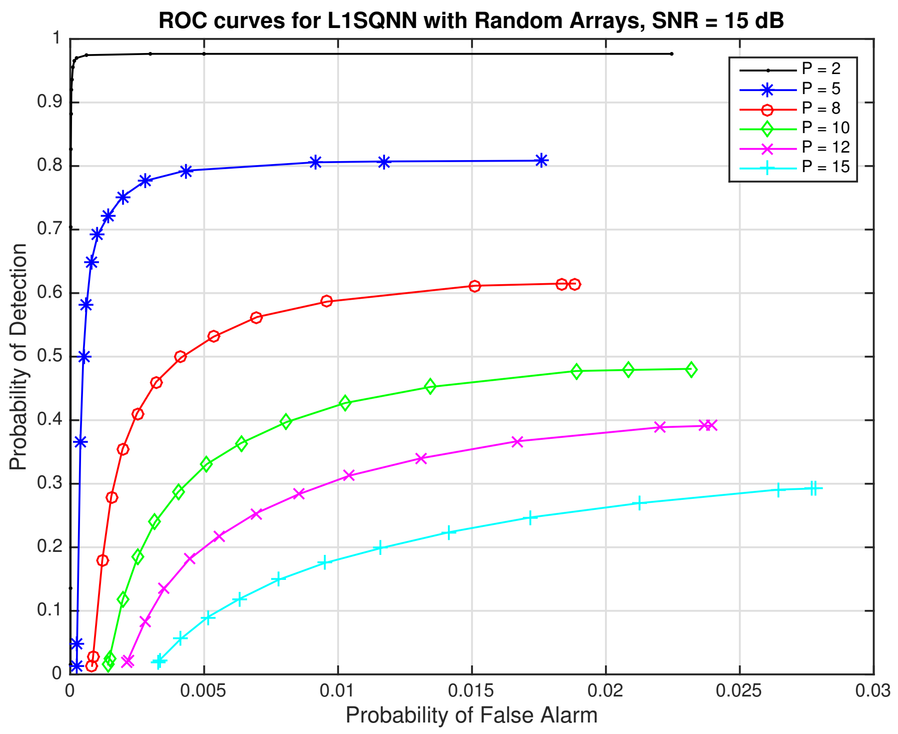


Figure 3.5: ROC curves for L1SQNN with random arrays, SNR = 15 dB and various sparsity levels.

with a uniform separation distance of λ_{ref}, i.e., the reference carrier wavelength $\lambda_{\text{ref}} = c/f_{\text{ref}}$, and transmit antennas similarly positioned but with a uniform separation distance of $2N\lambda_{\text{ref}}$. Here the SNR is fixed at 20 dB, $\beta = 9$ is chosen as the parameter for L1SQNN and a set of threshold values is taken from $[0.0001, 0.999]$. The ULA-based matrix, which never changes since it is completely deterministic, and a random array-based matrix are each used to separately measure and recover the same vector at each step of the simulations. A total of 500 simulations, i.e., 20 random array matrices (and 1 ULA matrix) applied to 25 target vectors, are performed for each sparsity level, which ranges from 5 to 15. As highlighted by the ROC curves in Fig. 3.4, the random sensor array systems provide superior performance over the ULA system.

Focusing on random sensor arrays, we examine how L1SQNN performs as the sparsity level increases. The SNR is decreased to 15 dB but remains fixed throughout the experiment, while $\beta = 5$ is chosen as regularization parameter. We perform 1000 simulations, i.e., 25 matrices applied to

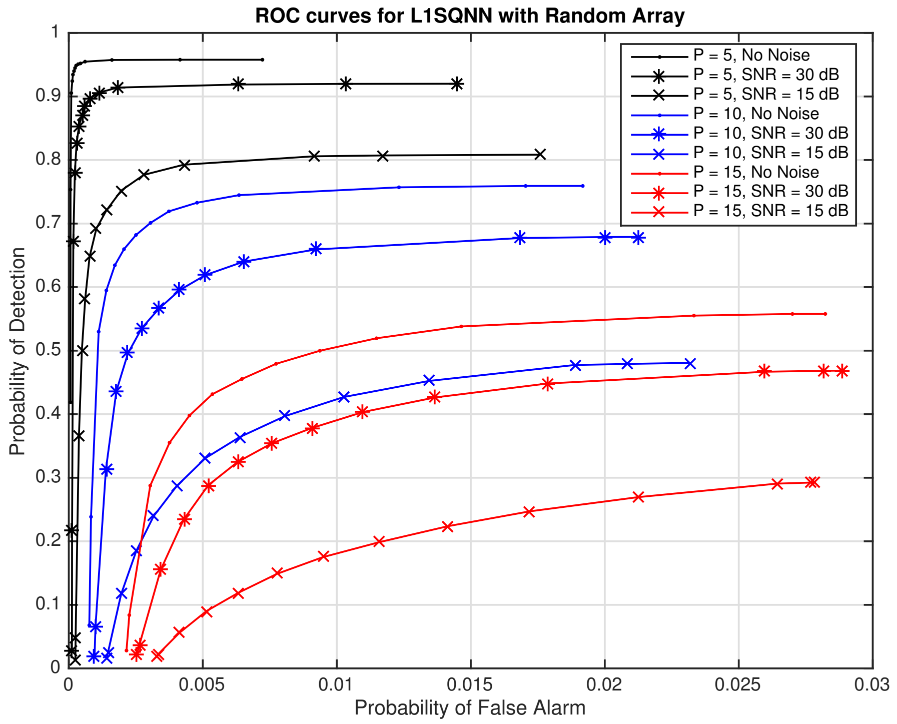


Figure 3.6: Families of ROC curves for L1SQNN with random arrays. Each family of curves corresponds to a different sparsity level (denoted by color), while individual curves within each family correspond to a different SNR (denoted by symbol).

40 vectors, at each sparsity level, which now ranges from 2 to 15. The ROC curves in Fig. 3.5 demonstrate a graceful decay in performance as the number of targets increases. Additionally, comparing the random array ROC curves from Fig. 3.4, where the SNR was 20 dB, to the curves in Fig. 3.5 with the same respective sparsity counts also illustrates a reasonable decline in performance as the SNR decreases. This decline is clearly highlighted in Fig. 3.6 which presents the performance of L1SQNN at three different levels of SNR for three distinct sparsity levels.

For the next set of simulations, we investigate how changing the parameter of L1SQNN impacts its performance with random sensor arrays. As before, we run 400 simulations for each sparsity level with a set of threshold values taken from $[0.0001, 0.999]$; however, we consider different levels of SNR separately. For each SNR, we perform the recovery with distinct values of the regularization parameter β. The results are displayed in Fig. 3.7 and 3.8. Each subplot in Fig. 3.7 contains separate families of ROC curves which correspond to the sparsity levels of 5 (dots), 8 (circles) and 12 (stars),

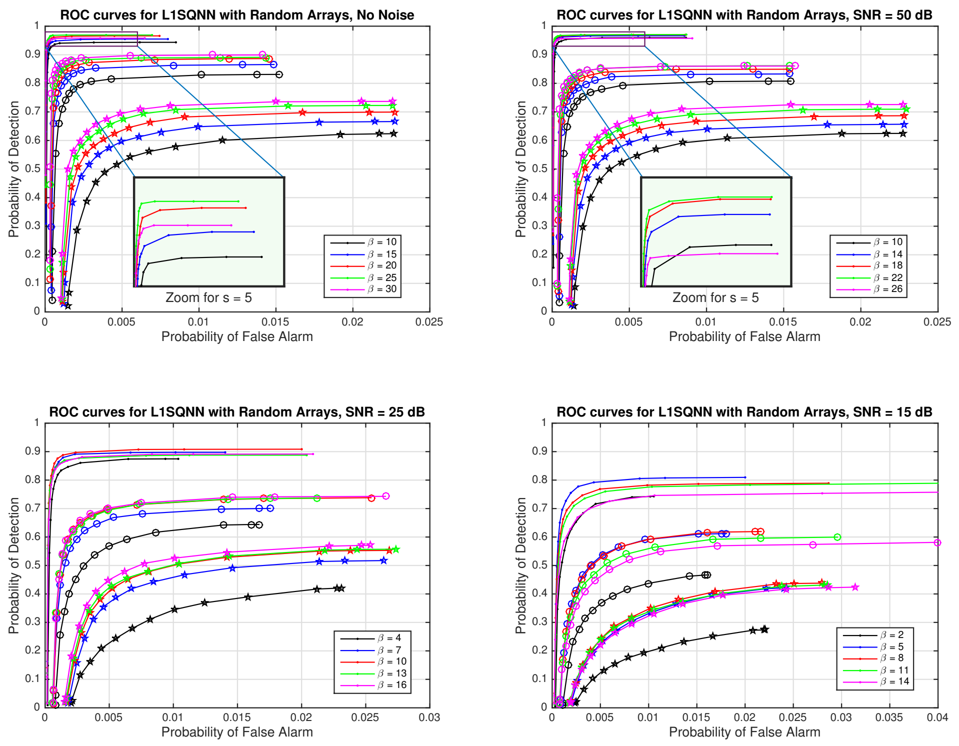


Figure 3.7: ROC curves for L1SQNN with random arrays at various levels of SNR and different values of the regularization parameter. The upper set of curves in each graph corresponds to a sparsity of 5 (dots), the middle set corresponds to a sparsity of 8 (circles) and the lower set corresponds to a sparsity of 12 (stars). The figures for the noiseless case and for SNR = 50 dB each feature a magnified view of a portion of the ROC curves when the sparsity is 5.

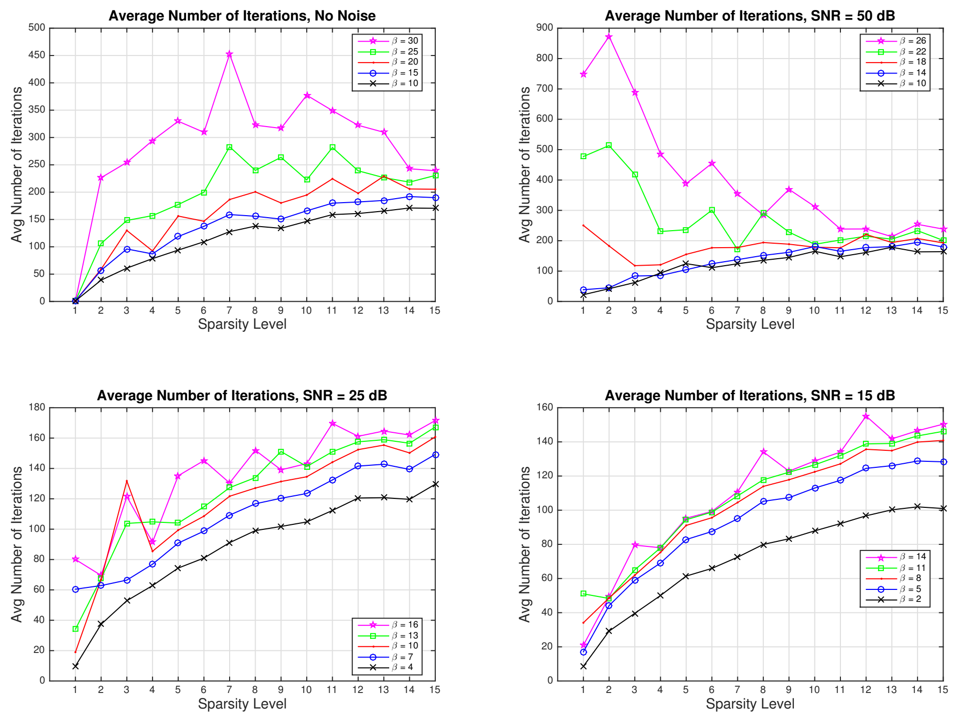


Figure 3.8: The average number of iterations versus the sparsity at distinct SNR levels and with different values of the regularization parameter for L1SQNN with random arrays.

respectively. Since the curves within each family represent a different value of the parameter β, these figures indicate that, for a fixed SNR, the value of β should increase to enhance performance as the sparsity increases. Furthermore, one can observe from Fig. 3.7 that as the SNR rises, the regularization parameter should increase to improve performance overall. Comparing the plots in Fig. 3.8 reveals, for select parameter values, a sharp increase in the average number of iterations needed for the method to terminate while moving from the noise-free scenario to the case where a small amount of noise corrupts the measurements, particularly at low sparsity levels. As the SNR continues to deteriorate, these apparent differences diminish. Fig. 3.7 and 3.8 highlight the importance of fine-tuning the regularization parameter to attain the desired probability of detection or probability of false alarm. However, if the SNR and sparsity level are a priori unknown, then more advanced techniques may be used to estimate these quantities and update the parameter accordingly, if, for example, using a constant false alarm rate (CFAR) system [Skolnik 2001].

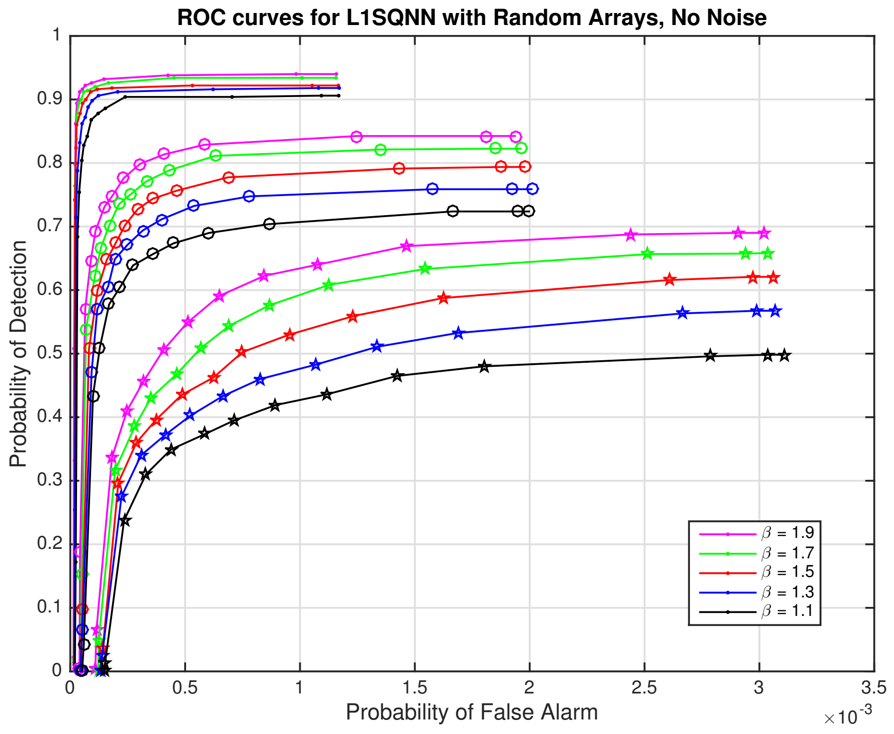


Figure 3.9: Comparison of various parameter values, β, in L1SQNN for the larger random array measurement matrices of size 4000×8000 with no noise. The upper set of curves corresponds to a sparsity of 5 (dots), the middle set corresponds to a sparsity of 8 (circles) and the lower set corresponds to a sparsity of 12 (stars).

The previous simulations are repeated on a smaller scale but for a larger problem: the number of transmitters, receivers, sample times, and bins in each domain are doubled, hence, the dimensions of the measurement matrix have increased from 500×1000 to 4000×8000. However, only 100 simulations are performed, i.e., 10 random sensor array measurement matrices are applied to 10 random target vectors. The results for the noise-free scenario are presented in Fig. 3.9. The most noticeable difference between the ROC curves of Fig. 3.9 and those in the noise-free plot from Fig. 3.7 is the order of magnitude decrease in the probability of false alarm for the larger problem. This is appropriate given the increase in the number of bins from 1000 to 8000. Comparing these figures also exposes a heightened sensitivity to the regularization parameter since smaller variations in β lead to more pronounced changes in the ROC curves, as displayed in Fig. 3.9. Although not shown here, the number of iterations is consistent with previous simulations, but the average amount of

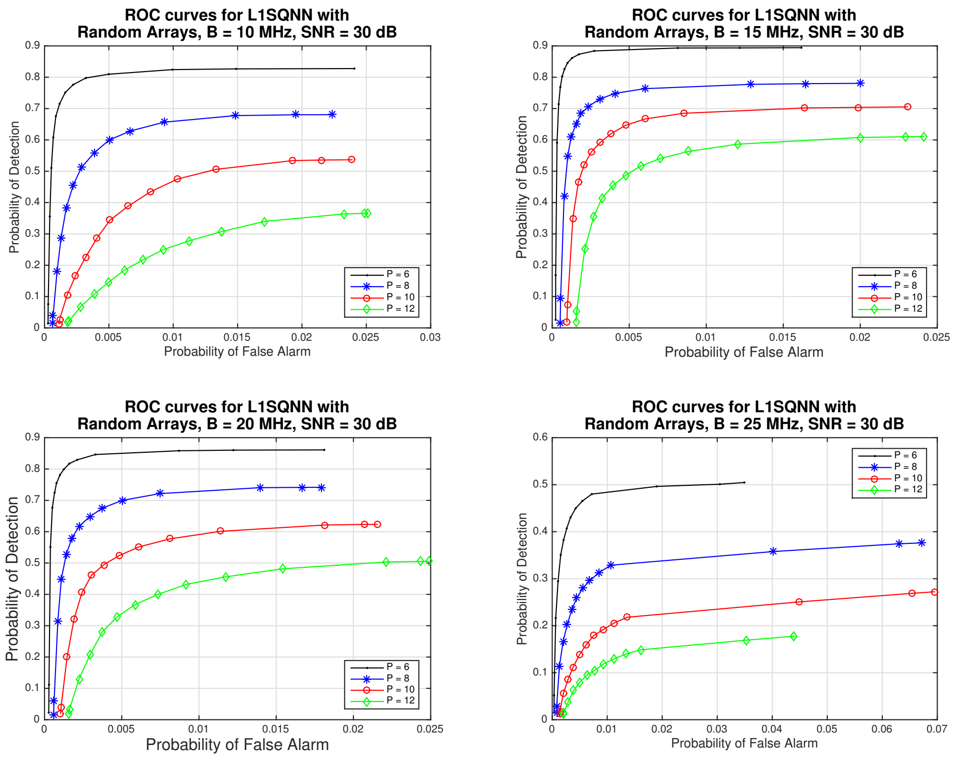


Figure 3.10: Comparison of various bandwidth values, B, in L1SQNN with random arrays, SNR $=$ 30 dB and various sparsity levels.

time needed for the algorithm to terminate is significantly longer, by at least an order of magnitude, depending on the sparsity level, and increases for greater values of β.

As a final experiment, we explore how changing the bandwidth impacts the performance. We select a set of values for the bandwidth, B, and at each value run 800 simulations for a predetermined collection of sparsity levels. The reconstruction is performed with a fixed regularization parameter, $\beta = 15$, along with a set of threshold values taken from $[0.0001, 0.999]$ and a constant SNR of 30 dB. The results are displayed in Fig. 3.10. The subplots in Fig. 3.10 correspond to the bandwidth values of $10, 15, 20$, and 25 MHz, respectively, while the curves within each subplot represent a different sparsity level. Noting how each ROC curve for a fixed sparsity changes across the subplots, one can observe that as the bandwidth rises, the performance improves then decays. Further simulations indicate that the performance improves again briefly as the bandwidth continues to increase but then drops off sharply.

The results from each of these sets of simulations indicate the following: for the specified physical parameters, this sparse vector recovery problem requires a significantly low sparsity level to achieve meaningful performance. As the sparsity level increases from just a few scatterers and as the SNR decreases from the noiseless setting, the performance consistently decays, though typically in a graceful manner. Although increasing the number of bins in each domain does not greatly inhibit the results, it does lead to a significant increase in the run time. This is due to the high computational complexity which results from discretizing in azimuth, delay and Doppler instead of simply one or two of these domains.

3.6 Conclusion

We have combined elements from recent work in [Strohmer and Wang 2013] and [He et al. 2013] to further investigate the applicability of Compressive Sensing to a MIMO radar system. Specifically, we have considered a random array MIMO radar system which transmits linear chirps and utilizes orthogonal separation and dechirping to acquire additional measurements while probing a small target scene in the azimuth-delay-Doppler domain. The various simulations, some of which are included in Appendix A.1 , demonstrate superior performance when only a few point scatterers are present in the scene. The ℓ_1-squared Nonnegative Regularization method from [Foucart and Koslicki 2014] provides enhanced recovery in the presence of noise over the other algorithms considered, however, as is always the case, the regularization parameter must be finely tuned to achieve a desired false alarm rate.

This work is intended as an initial step in exploring the feasibility of applying techniques from Compressive Sensing to the off-grid MIMO radar problem in the azimuth, time-delay and Doppler domain. However, when programming the vectorized target scene for our experiments we noted an interesting observation: due to the fact that targets could not occupy the same range and azimuth location simultaneously they were often separated from each other due to the additional Doppler domain. This is not surprising because we made the reasonable assumption that the targets would not be colliding into one another in our setup. Yet, our experiments consistently generated sparse vectors whose nonzero entries were disjoint from one another due to the presence of a large number of zeros between any two pairs of targets. In keeping with the considerations laid out in Section 1.2, we investigate this underlying structure from a general perspective, i.e., outside of the context of radar system, in the proceeding chapter.

Chapter 4: Sparse Disjoint Structure

We investigate the minimal number of linear measurements needed to recover sparse disjointed vectors robustly in the presence of measurement error. We discuss basic facts about sparse disjointed vectors and detail how projections onto the set of sparse disjointed vectors can be computed by dynamic programming. The ability to compute these projections would allow for the modification of virtually all sparse recovery iterative greedy algorithms to fit the sparse disjointed framework, but we focus only on iterative hard thresholding (IHT)—arguably the simplest of these algorithms. We provide a justification that robust uniform recovery can be carried out efficiently based on random measurements (which are noninflating, see Section 4.4) provided their number has order at least $m_{\mathsf{spa\&dis}}$, see Theorem 23. Finally, we present the crucial result that robust uniform recovery schemes for sparse disjointed vectors cannot exist if the number of noninflating measurements has order less than $m_{\mathsf{spa\&dis}}$.

4.1 Introduction

As previously introduced in Chapter 1 and in following the observation made in Section 3.6, here we study vectors that are simultaneously s-sparse and d-disjointed. A vector $\mathbf{x} \in \mathbb{C}^N$ is said to be d-disjointed if there are always at least d zero entries between two nonzero entries, i.e., if $|j-i| > d$ for all distinct $i, j \in \operatorname{supp}(\mathbf{x})$. Sparse disjointed vectors also serve as a pertinent model for neural spike trains, see [Hegde et al. 2009] which already established recovery results similar to those presented in Section 4.3. However, we consider the question of the *minimal* number of measurements needed for robust uniform recovery of sparse disjointed vectors. As presented in Section 2.4, the uniform recovery of s-sparse vectors is achievable from

$$m \asymp m_{\mathsf{spa}} := s \ln\left(e\frac{N}{s}\right) \tag{4.1}$$

random linear measurements. This can be carried out efficiently via convex optimization or iterative greedy algorithms and the recovery is both robust with respect to measurement error and stable with respect to sparsity defect. The number of measurements in (4.1) is optimal when stability is required. The uniform recovery of d-disjointed vectors is achievable from

$$m \asymp m_{\mathsf{dis}} := \frac{N}{d} \tag{4.2}$$

deterministic Fourier measurements and it can be carried out efficiently using convex optimization (see [Candès and Fernandez-Granda 2014 Corollary 1.4]). The number of measurements (4.2) is easily seen to be optimal, even without requiring stability. We give an informal statement of our main result regarding simultaneously sparse and disjointed vectors below.

Theorem 23. The minimal number of noninflating measurements needed to achieve robust uniform recovery of s-sparse d-disjointed vectors is of the order of

$$m_{\mathsf{spa\&dis}} := s \ln\left(e \frac{N - d(s-1)}{s} \right). \tag{4.3}$$

The significance of this result lies in the interpretation: for $m_{\mathsf{spa\&dis}}$ to be of smaller order than m_{spa}, we need $t := (N - d(s-1))/s \leq N/(2s)$; but then $d = (N - st)/(s-1) \geq (N - N/2)/(s-1) \geq N/(2s)$, i.e., $N/d \leq 2s$, which implies that m_{dis} is of smaller order than $m_{\mathsf{spa\&dis}}$. In short, we have arrived at

$$m_{\mathsf{spa\&dis}} \asymp \min\left\{ m_{\mathsf{spa}}, m_{\mathsf{dis}} \right\}. \tag{4.4}$$

Expressed differently, there is no benefit in knowing the simultaneity of sparsity and disjointness as far as the number of noniflating measurements is concerned. This recalls the message of [Oymak et al. 2015], which showed that vectors possessing certain structures simultaneously require at least as many Gaussian random measurements for their recovery via combined convex relaxations as what could have been achieved via the convex relaxation associated to one of the structures. Our result is narrower since it focuses on a particular simultaneity of structures, but no limitation is placed on the nature of the recovery algorithm and the measurements are only assumed to be noninflating instead of Gaussian. Finally, restricting to ℓ_1-minimization and Gaussian measurements would have been irrelevant here, because even nonuniform recovery, i.e., the recovery of a single sparse vector—a fortiori of a disjointed one—already requires a number of measurements of order at least m_{spa}, as inferred from known results on phase transition (see [Donoho and Tanner 2009] for the original arguments and [Amelunxen et al. 2014] for recent arguments).

4.2 Sparse Disjointed Vectors

We note first that sparsity and disjointness are not totally independent structures, since a highly disjointed vector is automatically quite sparse when its length N is fixed. Alternatively, an exactly s-sparse vector that is d-disjointed cannot have too small a length, i.e. $N \geq s + d(s-1)$, because there must be s nonzero entries and at least d zero entries in each of the $s-1$ spaces between them. In connection with (4.3), we re-express this inequality as $N - d(s-1) \geq s$ to highlight that the logarithmic factor is at least equal to 1. Figure 4.1 depicts a useful, alternative way to think of an s-sparse d-disjointed vector. Namely, we artificially insert d zero entries at the right end, hence forming a vector of length $N+d$ containing s blocks of size $d+1$. Each block consists of one nonzero entry followed by d zero entries. Then, identifying every block with a condensed object of length 1 reveals a one-to-one correspondence between s-sparse d-disjointed vectors of length N and s-sparse vectors of length $N - d(s-1)$. This correspondence will be used again in Section 4.4, but for now it explains the following fact already employed in [Hegde et al. 2009], which is based on the well-known number of ways to distribute n identical objects amongst k groups.

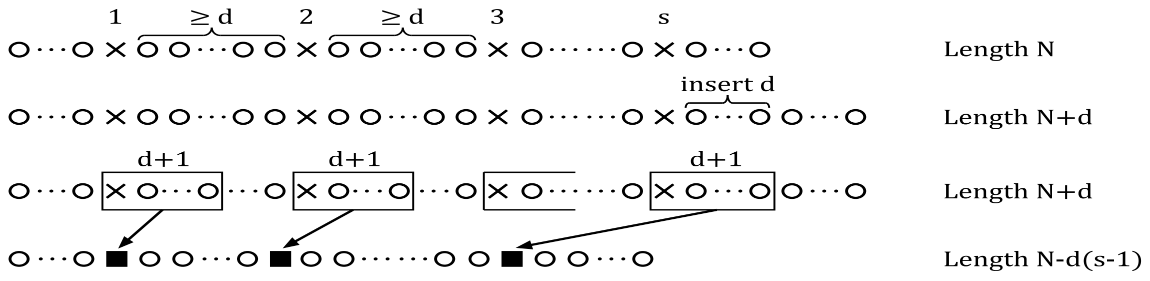


Figure 4.1: Pictorial representation of sparse disjointed vectors (hollow circles represent zero values).

Fact 24. The number of d-disjointed subsets of $[\![1 : N]\!]$ with size s is $\binom{N - d(s-1)}{s}$.

This formula could instead be justified by the inductive process at the basis of Fact 25 below, which concerns the computation of best approximations by s-sparse d-disjointed vectors in ℓ_p for $p \in (0, \infty)$. This task is not immediate, unlike the computation of the best approximations by s-sparse vectors (called hard thresholding), which simply consists of keeping the s largest absolute entries and setting the other entries to zero. Perhaps counterintuitively, the largest absolute entry need not be part of the support of the best approximation in the sparse disjointed case[1], as illustrated by the example $\mathbf{x} = (1, 0, 1, 2^{1/4}, 1, 0, 2^{-1/2})$ whose best 3-sparse 1-separated approximation is $(1, 0, 1, 0, 1, 0, 0)$ for $p = 2$. Note that the best approximation is $(1, 0, 0, 2^{1/4}, 0, 0, 2^{-1/2})$ for $p = 4$, highlighting another difference with the sparse case, namely the possible dependence on p of the best approximation. The strategy adopted in [Hegde et al. 2009] for computing best sparse disjointed approximations consisted in recasting the problem as an integer program and relaxing it to a linear program proved to yield the same solution. We propose a different approach here. The corresponding Matlab implementation, along with an illustrative set of experiments, are included in Appendix A.2.

Fact 25. The best ℓ_p-approximation of a vector $\mathbf{x} \in \mathbb{C}^N$ from the set of s-sparse d-disjointed vectors can be efficiently computed by dynamic programming for any $p \in (0, \infty)$.

Dynamic programming in general refers to methods for solving optimization problems by breaking them into smaller subproblems. The solutions of these smaller optimization problems are then synthesized in an orderly fashion to solve the original optimization problem. See [Isaev 2006] for examples of dynamic programming in applications to computational gene sequence alignment. The program of Fact 25 determines the error of best approximation $F(N, s)$, where $F(n, r)$ is defined for $n \in [\![1 : N]\!]$ and $r \in [\![0 : s]\!]$ by

$$F(n, r) := \min \left\{ \sum_{j=1}^{n} |x_j - z_j|^p : \mathbf{z} \in \mathbb{C}^n \text{ is } r\text{-sparse } d\text{-disjointed} \right\}. \tag{4.5}$$

[1] In particular, the nested approximation property of [Baraniuk et al. 2010], as mentioned in Section 1.3 does not hold in this case.

We claim that, for $n \in [\![d+2, N]\!]$ and $r \in [\![1, s]\!]$,

$$F(n,r) = \min \begin{cases} F(n-1,r) + |x_n|^p, \\ F(n-d-1, r-1) + \sum\limits_{j=n-d}^{n-1} |x_j|^p. \end{cases} \tag{4.6}$$

This relation simply distinguishes between a zero and a nonzero value at the last entry of the minimizer for $F(n,r)$.

We establish first the lower estimate on $F(n,r)$ by considering a minimizer $\widehat{\mathbf{x}} \in \mathbb{C}^n$ for $F(n,r)$: if $\widehat{x}_n = 0$, then

$$F(n,r) = \sum_{j=1}^{n-1} |x_j - \widehat{x}_j|^p + |x_n|^p \geq F(n-1,r) + |x_n|^p,$$

since $\widehat{\mathbf{x}}_{[\![1:n-1]\!]} \in \mathbb{C}^{n-1}$ is r-sparse d-disjointed; if $\widehat{x}_n \neq 0$, so that $\widehat{x}_{n-d} = \widehat{x}_{n-d+1} = \cdots = \widehat{x}_{n-1} = 0$ by d-disjointedness, then

$$F(n,r) = \sum_{j=1}^{n-d-1} |x_j - \widehat{x}_j|^p + \sum_{j=n-d}^{n-1} |x_j|^p + |x_n - \widehat{x}_n|^p \geq F(n-d-1, r-1) + \sum_{j=n-d}^{n-1} |x_j|^p,$$

since $\widehat{\mathbf{x}}_{[\![1:n-d-1]\!]} \in \mathbb{C}^{n-d-1}$ is $(r-1)$-sparse d-disjointed. Next, we establish the upper estimate on $F(n,r)$ by separating cases for the minimum in (4.6): if $F(n-1,r) + |x_n|^p$ is the smallest value, selecting a minimizer $\widetilde{\mathbf{x}} \in \mathbb{C}^{n-1}$ for $F(n-1,r)$ and considering the r-sparse d-disjointed vector $\widehat{\mathbf{x}} := (\widetilde{\mathbf{x}}, 0) \in \mathbb{C}^n$ yields

$$F(n,r) \leq \sum_{j=1}^{n} |x_j - \widehat{x}_j|^p = \sum_{j=1}^{n-1} |x_j - \widetilde{x}_j|^p + |x_n|^p = F(n-1,r) + |x_n|^p;$$

if $F(n-d-1, r-1) + \sum_{j=n-d}^{n-1} |x_j|^p$ is the smallest value, selecting a minimizer $\widetilde{\mathbf{x}} \in \mathbb{C}^{n-d-1}$ for $F(n-d-1, r-1)$ and considering the r-sparse d-separated vector $\widehat{\mathbf{x}} := (\widetilde{\mathbf{x}}, 0, \ldots, 0, x_n) \in \mathbb{C}^n$ yields

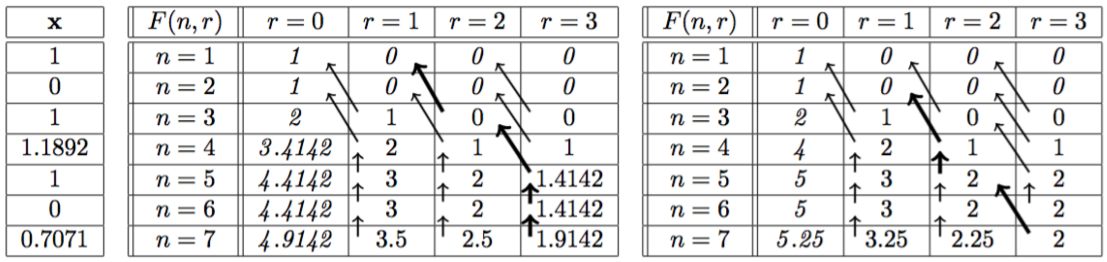
$$F(n,r) \leq \sum_{j=1}^{n} |x_j - \widehat{x}_j|^p = \sum_{j=1}^{n-d-1} |x_j - \widetilde{x}_j|^p + \sum_{j=n-d}^{n-1} |x_j|^p = F(n-d-1, r-1) + \sum_{j=n-d}^{n-1} |x_j|^p.$$

The relation (4.6) is now fully justified; thus, we can fill in a table of values for $F(n,r)$ from the initial values

$$F(n,0) = \|\mathbf{x}_{[\![1:n]\!]}\|_p^p, \quad n \in [\![1:N]\!],$$

$$F(n,r) = \|\mathbf{x}_{[\![1:n]\!]}\|_p^p - \max_{j \in [\![1:n]\!]} |x_j|^p, \quad n \in [\![1:d+1]\!], r \in [\![1:s]\!].$$

The latter relation reflects the absence of exactly r-sparse d-disjointed vectors in $\mathbb{C}^n$ when $r \geq 2$ and $n \leq d+1$. Let us observe that, according to (4.6), determining one entry of the table generated by (4.5) requires $\mathcal{O}(d)$ arithmetic operations and that there are $\mathcal{O}(sN)$ entries in F, so computing the error of best approximation by s-sparse d-disjointed vectors requires a total of $\mathcal{O}(dsN) = \mathcal{O}(N^2)$ arithmetic operations (to compare with $\mathcal{O}(N^{3.5})$ for the linear programming strategy of [Hegde et al. 2009]). As for the best approximation itself, we need to keep track of the cases producing the minima in (4.6)[2]. Using arrows as in Figure 4.2, an arrow pointing northwest from the (n,r)th box indicates that the index n is part of the support of the best approximation, while an arrow pointing north indicates that the index is not part of the support. Once the support is determined, the best approximation is the vector equal to $\mathbf{x}$ on this support.



$\mathbf{x}$
1
0
1
1.1892
1
0
0.7071

$F(n,r)$	$r=0$	$r=1$	$r=2$	$r=3$
$n=1$	*1*	*0*	*0*	*0*
$n=2$	*1*	*0*	*0*	*0*
$n=3$	*2*	1	0	0
$n=4$	*3.4142*	2	1	1
$n=5$	*4.4142*	3	2	1.4142
$n=6$	*4.4142*	3	2	1.4142
$n=7$	*4.9142*	3.5	2.5	1.9142

$F(n,r)$	$r=0$	$r=1$	$r=2$	$r=3$
$n=1$	*1*	*0*	*0*	*0*
$n=2$	*1*	*0*	*0*	*0*
$n=3$	*2*	1	0	0
$n=4$	*4*	2	1	1
$n=5$	*5*	3	2	2
$n=6$	*5*	3	2	2
$n=7$	*5.25*	3.25	2.25	2

Figure 4.2: Sketch of the dynamic program computing the best s-sparse d-disjointed approximations to $\mathbf{x} = (1, 0, 1, 2^{1/4}, 1, 0, 2^{-1/2})$ with $s = 3$ and $d = 1$ for $p = 2$ (left) and $p = 4$ (right). The italicized numbers in each table indicate values that are computed as part of the initialization of the algorithm. Observe that according to the tables, the output for the best 3-sparse 1-disjointed approximation to $\mathbf{x}$ for $p = 2$ is $\mathbf{x}_{p=2} = (1, 0, 1, 0, 1, 0, 0)$, while the output for $p = 4$ is $\mathbf{x}_{p=4} = (1, 0, 0, 2^{1/4}, 0, 0, 2^{-1/2})$.

The example highlighted in Figure 4.2 is included in Appendix A.2, as previously mentioned, along with an additional experiment to highlight a situation in which the iterative hard thresholding

[2]we break possible ties arbitrarily by choosing preferentially the second case

algorithm adapted for sparse and disjoint approximations outperforms the original IHT method. See Section 4.3 for the pseudocode and Appendix A.2 for the implementation of this algorithm. Figure 4.3 presents the difference in the outputs of each method by plotting the value associated with each index of the original vector in blue and the returned value in red. Additionally, a collection of simulations were performed to investigate how these algorithms perform with a fixed value of $N = 1000$, two different values of $d = 10$ and $d = 20$, and sparsity levels ranging from 1 to 20. For this experiment, one hundred s-sparse and d-disjointed vectors and one hundred random measurement matrices, with $m = 200$, were generated. The vectors were measured precisely, i.e., without noise, then IHT and the modified IHT methods were called to approximate the original signals. A comparison of the percentage of successful recoveries as the sparsity increases is displayed in Figure 4.4. Notice that although the modified IHT slightly outperforms the classical IHT, both algorithms perform roughly the same overall and exhibit a similar graceful decay.

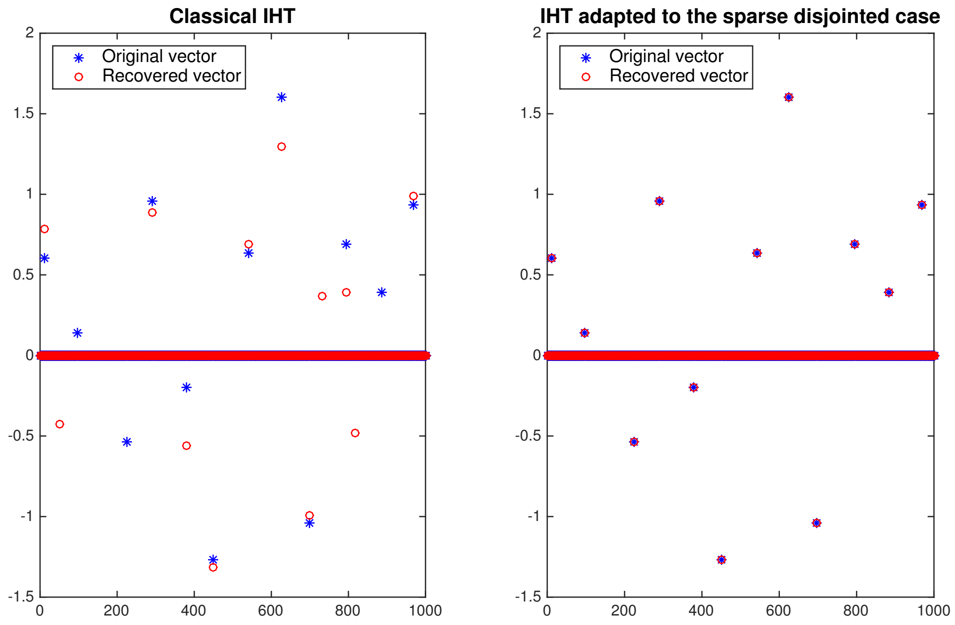


Figure 4.3: Comparison of the classical IHT and IHT adapted to the sparse disjointed structure. Here one random 12-sparse 65-disjointed vector in $\mathbb{R}^{1000}$ is generated, then recovered by each method using the same random measurement matrix with $m = 200$. The values of entries of the original vector and the recovered vector are plotted together for each method.

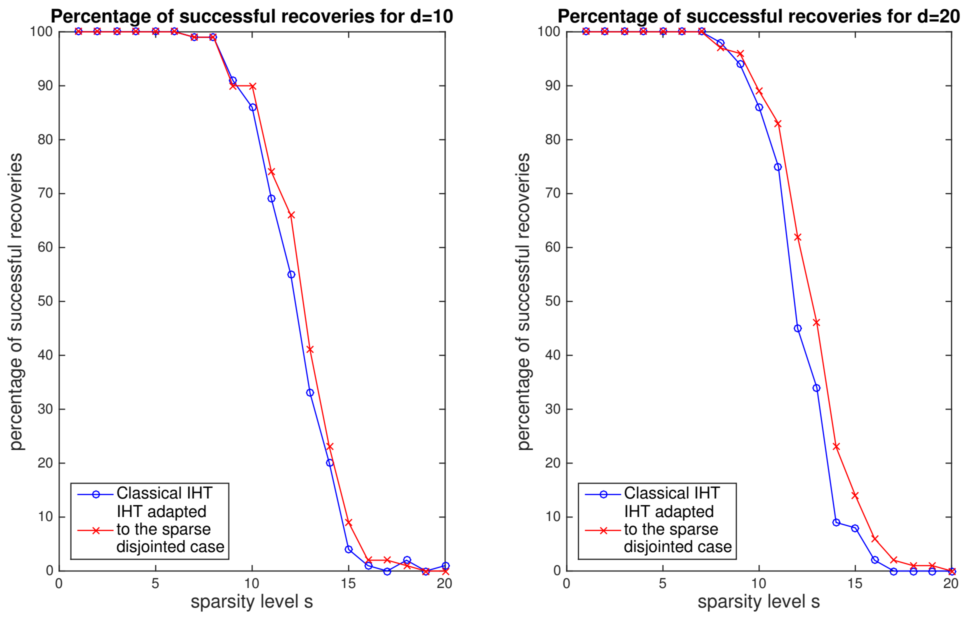


Figure 4.4: Comparison of the efficacy of the classical IHT and IHT adapted to the sparse disjointed structure. Here $N = 1000$, $d = 10$ and $d = 20$, and the sparsity varies from 1 to 20. One hundred s-sparse and d-disjointed vectors and one hundred random measurement matrices, with $m = 200$, are generated combination of parameters. The ℓ_2-errors of the recovered vectors are compared with a fixed tolerance level to assess whether a particular simulation was successful. The percentage of successful recoveries for each method at each value of d are plotted together.

4.3 Sufficient Number of Measurements

Here we show that a number of measurements proportional to $m_{\mathsf{spa\&dis}}$ is enough to guarantee robust recovery of sparse disjointed vectors. Such a result was already stated in [Hegde et al. 2009 Theorem 3], see [Baraniuk et al. 2010] for the proof. However, the algorithm considered in these articles is an adaptation of CoSaMP, whereas we analyze a natural adaptation of IHT. Our approach also simplifies the existing arguments. The main tool we employ is a restricted-isometry-like property valid in the general context of union of subspaces, see [Blumensath and Davies 2009a Theorem 3.3]. The slight difference with the theorem stated below is that the scaling in δ has been reduced from $\ln(1/\delta)/\delta^2$ to the optimal $1/\delta^2$.

Theorem 26. Let $\delta \in (0,1)$ and let $\mathbf{A} \in \mathbb{C}^{m\times N}$ be populated by independent identically distributed subgaussian random variables with variance $1/m$. Then, with probability at least $1-2\exp(-c\delta^2 m)$,

$$(1-\delta)\|\mathbf{z}+\mathbf{z}'+\mathbf{z}''\|_2^2 \le \|\mathbf{A}(\mathbf{z}+\mathbf{z}'+\mathbf{z}'')\|_2^2 \le (1+\delta)\|\mathbf{z}+\mathbf{z}'+\mathbf{z}''\|_2^2 \tag{4.7}$$

for all s-sparse d-disjointed $\mathbf{z},\mathbf{z}',\mathbf{z}'' \in \mathbb{C}^N$, provided

$$m \ge \frac{C}{\delta^2} s \ln\left(e\frac{N-d(s-1)}{s}\right).$$

The constants $c, C > 0$ depend only on the subgaussian distribution.

We note that the need for $\mathbf{z},\mathbf{z}'$, and $\mathbf{z}''$ will be made clear in the proof of the upcoming Proposition 27.

Proof. We prove the equivalent statement that, with probability at least $1-2\exp(-c\delta^2 m)$,

$$\|\mathbf{A}_T^*\mathbf{A}_T - \mathbf{I}\|_{2\to 2} \le \delta$$

for all sets $T \subseteq [\![1:N]\!]$ of the form $S\cup S'\cup S''$ where S, S', S'' are d-disjointed subsets of $[\![1:N]\!]$ with size s. Notice this echoes the alternate form of the RIP as presented in (2.12). Such sets are of size at most $3s$ and their number is upper-bounded by the cube of the number of d-disjointed subsets with size s, i.e., by

$$\binom{N-d(s-1)}{s}^3 \leq \left(e\frac{N-d(s-1)}{s}\right)^{3s}.$$

This is a straightforward estimate, confer with Lemma C.5 of [Foucart and Rauhut 2013] for a proof. Now, if T is fixed, the following inequality holds (see e.g. [Foucart and Rauhut 2013 Theorem 9.9 and Equation (9.12)])

$$\mathbb{P}\left(\|\mathbf{A}_T^*\mathbf{A}_T - \mathbf{I}\|_{2\to 2} > \delta\right) \leq 2\exp\left(-c'\delta^2 m + c''s\right)$$

with constants $c', c'' > 0$ depending only on the subgaussian distribution, confer with Definition 19. Taking a union bound over all possible T, we see that the desired result holds with failure probability at most

$$\begin{aligned}\left(e\frac{N-d(s-1)}{s}\right)^{3s} 2\exp\left(-c'\delta^2 m + c''s\right) &\leq 2\exp\left(-c'\delta^2 m + (c''+3)s\ln\left(e\frac{N-d(s-1)}{s}\right)\right)\\ &\leq 2\exp(-c'\delta^2 m/2),\end{aligned}$$

where the last inequality holds if one imposes $m \geq [2(c''+3)/c']\delta^{-2}s\ln(e(N-d(s-1))/s)$. □

The restricted-isometry-like property will not be used directly in the form (4.7), but rather through its two consequences below.

Proposition 27. Suppose that $\mathbf{A}$ satisfies (4.7). Then, for all s-sparse d-disjointed $\mathbf{x}, \mathbf{x}', \mathbf{x}'' \in \mathbb{C}^N$,

$$|\langle \mathbf{x}-\mathbf{x}', (\mathbf{A}^*\mathbf{A}-\mathbf{I})(\mathbf{x}-\mathbf{x}'')\rangle| \leq \delta\|\mathbf{x}-\mathbf{x}'\|_2\|\mathbf{x}-\mathbf{x}''\|_2, \tag{4.8}$$

and, for all $\mathbf{e} \in \mathbb{C}^m$ and all d-disjointed subsets S, S' of $[\![1:N]\!]$ with size s,

$$\|(\mathbf{A}^*\mathbf{e})_{S\cup S'}\|_2 \leq \sqrt{1+\delta}\,\|\mathbf{e}\|_2. \tag{4.9}$$

Proof. Setting $\mathbf{u} = e^{i\theta}(\mathbf{x} - \mathbf{x}')/\|\mathbf{x} - \mathbf{x}'\|_2$ and $\mathbf{v} = e^{iv}(\mathbf{x} - \mathbf{x}'')/\|\mathbf{x} - \mathbf{x}''\|_2^2$ for properly chosen $\theta, \upsilon \in [-\pi, \pi]$, we have

$$\begin{aligned}\frac{|\langle \mathbf{x} - \mathbf{x}', (\mathbf{A}^*\mathbf{A} - \mathbf{I})(\mathbf{x} - \mathbf{x}'')\rangle|}{\|\mathbf{x} - \mathbf{x}'\|_2\|\mathbf{x} - \mathbf{x}''\|_2} &= \mathrm{Re}\langle \mathbf{u}, (\mathbf{A}^*\mathbf{A} - \mathbf{I})\mathbf{v}\rangle = \mathrm{Re}\langle \mathbf{A}\mathbf{u}, \mathbf{A}\mathbf{v}\rangle - \mathrm{Re}\langle \mathbf{u}, \mathbf{v}\rangle \\ &= \frac{1}{4}\left(\|\mathbf{A}(\mathbf{u}+\mathbf{v})\|_2^2 - \|\mathbf{A}(\mathbf{u}-\mathbf{v})\|_2^2\right) - \frac{1}{4}\left(\|\mathbf{u}+\mathbf{v}\|_2^2 - \|\mathbf{u}-\mathbf{v}\|_2^2\right) \\ &\leq \frac{1}{4}|\|\mathbf{A}(\mathbf{u}+\mathbf{v})\|_2^2 - \|\mathbf{u}+\mathbf{v}\|_2^2| + \frac{1}{4}|\|\mathbf{A}(\mathbf{u}-\mathbf{v})\|_2^2 - \|\mathbf{u}-\mathbf{v}\|_2^2|.\end{aligned}$$

Noticing that both $\mathbf{u} + \mathbf{v}$ and $\mathbf{u} - \mathbf{v}$ take the form $\mathbf{z} + \mathbf{z}' + \mathbf{z}''$ for some s-sparse d-disjointed $\mathbf{z}, \mathbf{z}', \mathbf{z}'' \in \mathbb{C}^N$, we apply (4.7) to deduce

$$\frac{|\langle \mathbf{x} - \mathbf{x}', (\mathbf{A}^*\mathbf{A} - \mathbf{I})(\mathbf{x} - \mathbf{x}'')\rangle|}{\|\mathbf{x} - \mathbf{x}'\|_2\|\mathbf{x} - \mathbf{x}''\|_2} \leq \frac{\delta}{4}\left(\|\mathbf{u}+\mathbf{v}\|_2^2 + \|\mathbf{u}-\mathbf{v}\|_2^2\right) = \frac{\delta}{2}\left(\|\mathbf{u}\|_2^2 + \|\mathbf{v}\|_2^2\right) = \delta.$$

This shows the desired inequality (4.8). To prove inequality (4.9), we write

$$\|(\mathbf{A}^*\mathbf{e})_{S\cup S'}\|_2^2 = \langle (\mathbf{A}^*\mathbf{e})_{S\cup S'}, \mathbf{A}^*\mathbf{e}\rangle = \langle \mathbf{A}((\mathbf{A}^*\mathbf{e})_{S\cup S'}), \mathbf{e}\rangle \leq \|\mathbf{A}((\mathbf{A}^*\mathbf{e})_{S\cup S'})\|_2\|\mathbf{e}\|_2.$$

It now suffices to notice that $(\mathbf{A}^*\mathbf{e})_{S\cup S'}$ takes the form $\mathbf{z} + \mathbf{z}' + \mathbf{z}''$ for some s-sparse d-disjointed $\mathbf{z}, \mathbf{z}', \mathbf{z}'' \in \mathbb{C}^N$ (with $\mathbf{z}'' = \mathbf{0}$) to derive $\|\mathbf{A}((\mathbf{A}^*\mathbf{e})_{S\cup S'})\|_2 \leq \sqrt{1+\delta}\,\|(\mathbf{A}^*\mathbf{e})_{S\cup S'}\|_2$ from (4.7). The inequality (4.9) follows after simplifying by $\|(\mathbf{A}^*\mathbf{e})_{S\cup S'}\|_2$. $\square$

With Proposition 27 at hand, robust uniform recovery is quickly established for the iterative hard thresholding algorithm adapted to the framework of s-sparse d-disjointed vectors. In the description of this algorithm below, $\mathbf{P}_{s,d}$ represents the projection onto (i.e., best ℓ_2-approximation by) s-sparse d-disjointed vectors discussed in Section 4.1. An implementation of this algorithm is included in Appendix A.2

Sparse disjointed iterative hard thresholding (SDIHT)

Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, sparsity level s, disjointedness level d.

Initialization: s-sparse d-disjointed vector $\mathbf{x}^0$, typically $\mathbf{x}^0 = \mathbf{0}$.

Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$\mathbf{x}^{n+1} = \mathbf{P}_{s,d}(\mathbf{x}^n + \mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^n)). \tag{SDIHT}$$

Output: the s-sparse d-disjointed vector $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$.

Theorem 28. Suppose that $\mathbf{A} \in \mathbb{C}^{m \times N}$ satisfies (4.7) with $\delta < 1/2$. Then for every s-sparse d-disjointed vector $\mathbf{x} \in \mathbb{C}^N$ acquired via $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e} \in \mathbb{C}^m$ with an adversarial error $\mathbf{e} \in \mathbb{C}^m$, the output $\mathbf{x}^\sharp := \lim\limits_{n \to \infty} \mathbf{x}^n$ of IHT approximates $\mathbf{x}$ with ℓ_2-error

$$\|\mathbf{x} - \mathbf{x}^\sharp\|_2 \leq D\|\mathbf{e}\|_2, \tag{4.10}$$

where $D > 0$ is a constant depending only on δ. In particular, this conclusion is valid with high probability on the draw of a matrix populated by independent zero-mean Gaussian entries with variance $1/m$ provided

$$m \geq C\, m_{\mathsf{spa\&dis}}$$

for some absolute constant $C > 0$.

Proof. We observe that $\mathbf{x}^{n+1}$ is a better s-sparse d-disjointed ℓ_2-approximation to the vector $\mathbf{x}^n + \mathbf{A}^*(\mathbf{y} - \mathbf{A}\mathbf{x}^n) = \mathbf{x}^n + \mathbf{A}^*\mathbf{A}(\mathbf{x} - \mathbf{x}^n) + \mathbf{A}^*\mathbf{e}$ than $\mathbf{x}$ is to derive

$$\|(\mathbf{x}^n + \mathbf{A}^*\mathbf{A}(\mathbf{x} - \mathbf{x}^n) + \mathbf{A}^*\mathbf{e}) - \mathbf{x}^{n+1}\|_2^2 \leq \|(\mathbf{x}^n + \mathbf{A}^*\mathbf{A}(\mathbf{x} - \mathbf{x}^n) + \mathbf{A}^*\mathbf{e}) - \mathbf{x}\|_2^2,$$

i.e.,

$$\|\mathbf{x} - \mathbf{x}^{n+1} + (\mathbf{A}^*\mathbf{A} - \mathbf{I})(\mathbf{x} - \mathbf{x}^n) + \mathbf{A}^*\mathbf{e}\|_2^2 \leq \|(\mathbf{A}^*\mathbf{A} - \mathbf{I})(\mathbf{x} - \mathbf{x}^n) + \mathbf{A}^*\mathbf{e}\|_2^2.$$

After expanding the squares and rearranging, we deduce

$$\begin{aligned}\|\mathbf{x}-\mathbf{x}^{n+1}\|_2^2 &\leq -2\langle \mathbf{x}-\mathbf{x}^{n+1}, (\mathbf{A}^*\mathbf{A}-\mathbf{I})(\mathbf{x}-\mathbf{x}^n)+\mathbf{A}^*\mathbf{e}\rangle \\ &\leq 2|\langle \mathbf{x}-\mathbf{x}^{n+1}, (\mathbf{A}^*\mathbf{A}-\mathbf{I})(\mathbf{x}-\mathbf{x}^n)\rangle| + 2\|\mathbf{x}-\mathbf{x}^{n+1}\|_2 \|(\mathbf{A}^*\mathbf{e})_{S\cup S^{n+1}}\|_2,\end{aligned}$$

where S and S^{n+1} denote the supports of $\mathbf{x}$ and $\mathbf{x}^{n+1}$, respectively. Applying (4.8) and (4.9) and simplifying by $\|\mathbf{x}-\mathbf{x}^{n+1}\|_2$ gives

$$\|\mathbf{x}-\mathbf{x}^{n+1}\|_2 \leq 2\delta\|\mathbf{x}-\mathbf{x}^n\|_2 + 2\sqrt{1+\delta}\,\|\mathbf{e}\|_2.$$

With $\delta < 1/2$, this inequality readily implies the desired result (4.10) with $D := 2\sqrt{1+\delta}/(1-2\delta)$. Combining it with Theorem 26 yields the rest of the statement. □

4.4 Necessary Number of Measurements

We now show that a number of noninflating measurements at least proportional to $m_{\mathsf{spa\&dis}}$ is necessary to ensure robust recovery of sparse disjointed vectors. By noninflating measurements with constant c, relative to the s-sparse d-disjointed model, we mean that the matrix $\mathbf{A}$ associated with the measurement process satisfies,

$$\|\mathbf{A}\mathbf{z}\|_2 \leq c\|\mathbf{z}\|_2 \qquad \text{whenever } \mathbf{z}\in\mathbb{C}^N \text{ is } s\text{-sparse } d\text{-disjointed}.$$

Figuratively, the energy of a signal with the targeted structure is not inflated by the measurement process. According to Theorems 26 and 28, a random measurement process is likely to be noninflating (with constant $c \leq 1+\delta$) and it enables robust uniform recovery of s-sparse d-disjointed vectors when the number of measurements obeys $m \geq C\, m_{\mathsf{spa\&dis}}$. We show below that this is optimal. The key to the argument is a generalization of a lemma used in [Foucart et al. 2010] (see also [Foucart and Rauhut 2013 Lemma 10.12]).

Lemma 29. There exist

$$n \geq \left(\frac{N - d(s-1)}{c_1 s} \right)^{c_2 s} \tag{4.11}$$

d-disjointed subsets $S_1, \dots, S_n$ of $[\![1 : N]\!]$ such that

$$\operatorname{card}(S_i) = s \quad \text{for all } i \qquad \text{and} \qquad \operatorname{card}(S_i \cap S_j) < \frac{s}{2} \quad \text{for all } i \neq j.$$

The constant $c_1, c_2 > 0$ are universal—precisely, one can take $c_1 = 12e$ and $c_2 = 1/2$.

Proof. Let $\mathcal{A}$ be the collection of all d-disjointed subsets of $[\![1 : N]\!]$ with size s. Let us fix an arbitrary $S_1 \in \mathcal{A}$. We then consider the collection $\mathcal{A}_1$ of sets in $\mathcal{A}$ whose intersection with S_1 has size $s/2$ or more, i.e.,

$$\mathcal{A}_1 = \bigcup_{j=\lceil s/2 \rceil}^{s} \mathcal{A}_1^j, \qquad \text{where} \quad \mathcal{A}_1^j := \{ S \in \mathcal{A} : \operatorname{card}(S_1 \cap S) = j \}.$$

We claim that, for any $j \in [\![\lceil s/2 \rceil : s]\!]$,

$$\operatorname{card}(\mathcal{A}_1^j) \leq \binom{s}{j} \binom{N - d(s-1)}{s-j} \leq \binom{s}{j} \binom{N - d(s-1)}{\lfloor s/2 \rfloor}. \tag{4.12}$$

The first factor upper-bounds the possible choices of the intersection $J := S_1 \cap S$. The second factor

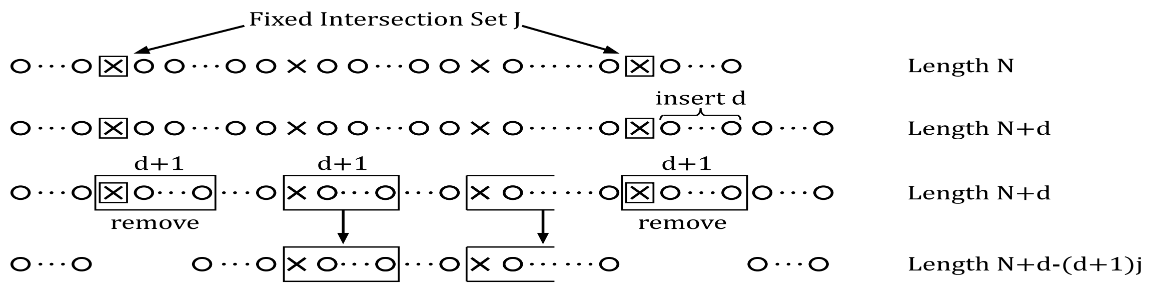


Figure 4.5: Illustration of the counting argument in the proof of Lemma 29.

upper-bounds the number of d-disjointed subsets of $[\![1 : N]\!]$ with size s whose intersection with S_1 is a fixed set J of size j: indeed, by thinking of s-sparse d-disjointed vectors of length N as s blocks with size $d+1$ inside a set with size $N+d$, as we did in Figure 4.1, we observe in Figure 4.5 that the d-disjointed subsets of $[\![1 : N]\!]$ with size s whose intersection with S_1 equals J inject into the

d-disjointed subsets of $[\![1 : N-(d+1)j]\!]$ with size $s-j$ by the process of removing the blocks attached to J, so the desired number is at most

$$\binom{N-(d+1)j-d(s-j-1)}{s-j} = \binom{N-d(s-1)-j}{s-j} \le \binom{N-d(s-1)}{s-j}.$$

This finishes the justification of the first inequality in (4.12). The second inequality holds because $s-j \le \lfloor s/2 \rfloor \le \lceil (N-d(s-1))/2 \rceil$. It then follows from (4.12) that

$$\operatorname{card}(\mathcal{A}_1) \le \sum_{j=\lceil s/2 \rceil}^{s} \binom{s}{j} \binom{N-d(s-1)}{\lfloor s/2 \rfloor} \le 2^s \binom{N-d(s-1)}{\lfloor s/2 \rfloor}.$$

Let us now fix an arbitrary set $S_2 \in \mathcal{A} \setminus \mathcal{A}_1$, provided the latter is nonempty. We consider the collection $\mathcal{A}_2$ of sets in $\mathcal{A} \setminus \mathcal{A}_1$ whose intersection with S_2 has size $s/2$ or more, i.e.,

$$\mathcal{A}_2 = \bigcup_{j=\lceil s/2 \rceil}^{s} \mathcal{A}_2^j, \qquad \text{where} \quad \mathcal{A}_2^j := \{ S \in \mathcal{A} \setminus \mathcal{A}_1 : \operatorname{card}(S_2 \cap S) = j \}.$$

The same reasoning as before yields

$$\operatorname{card}(\mathcal{A}_2) \le 2^s \binom{N-d(s-1)}{\lfloor s/2 \rfloor}.$$

We repeat the procedure of selecting sets $S_1, \ldots, S_n$ until $\mathcal{A} \setminus (\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_n)$ becomes empty. In this way, for any $i < j$, the condition $\operatorname{card}(S_i \cap S_j) < s/2$ is automatically fulfilled by virtue of $S_j \in \mathcal{A} \setminus (\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_{j-1}) \subseteq \mathcal{A} \setminus (\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_i)$. Finally, the number n of subsets satisfies

$$n \ge \frac{\operatorname{card}(\mathcal{A})}{\max_{i \in [\![1:n]\!]} \operatorname{card}(\mathcal{A}_i)} \ge \frac{\binom{N-d(s-1)}{s}}{2^s \binom{N-d(s-1)}{\lfloor s/2 \rfloor}} \ge \frac{\left(\frac{N-d(s-1)}{s}\right)^s}{2^s \left(e \frac{N-d(s-1)}{\lfloor s/2 \rfloor}\right)^{\lfloor s/2 \rfloor}} \ge \frac{\left(\frac{N-d(s-1)}{s}\right)^s}{2^s \left(e \frac{N-d(s-1)}{s/3}\right)^{s/2}}$$
$$= \left(\frac{N-d(s-1)}{12es}\right)^{s/2}. \qquad \square$$

We can now turn to the main result of this section.

Theorem 30. Let $\mathbf{A} \in \mathbb{C}^{m \times N}$ be the matrix of a noninflating measurement process with constant c relative to the s-sparse d-disjointed model and let $\Delta : \mathbb{C}^m \to \mathbb{C}^N$ be a reconstruction map providing a robustness estimate

$$\|\mathbf{x} - \Delta(\mathbf{A}\mathbf{x} + \mathbf{e})\|_2 \leq D\|\mathbf{e}\|_2 \tag{4.13}$$

which is valid for all s-sparse d-disjointed vectors $\mathbf{x} \in \mathbb{C}^N$ and all measurement error $\mathbf{e} \in \mathbb{C}^m$. Then the number m of measurements is lower-bounded as

$$m \geq Cs \ln\left(e\frac{N - d(s-1)}{s}\right).$$

The constant $C > 0$ depends only on c and D.

Proof. With each S_i of Lemma 29, we associate an s-sparse d-disjointed vector $\mathbf{x}^i \in \mathbb{C}^N$ defined by

$$x^i_\ell = \begin{cases} 1/\sqrt{s}, & \text{if } \ell \in S_i, \\ 0, & \text{if } \ell \notin S_i. \end{cases}$$

We notice that each $\mathbf{x}^i$ satisfies $\|\mathbf{x}^i\|_2 = 1$. We also notice that each $\mathbf{x}^i - \mathbf{x}^j$, $i \neq j$, is supported on the symmetric difference $(S_i \cup S_j) \setminus (S_i \cap S_j)$, which has size larger than s, and since its nonzero entries are $\pm 1/\sqrt{s}$, we have

$$\|\mathbf{x}^i - \mathbf{x}^j\|_2 > 1, \quad i \neq j.$$

Setting $\rho := 1/(2D)$, let us consider the balls in $\mathbb{C}^m \equiv \mathbb{R}^{2m}$ with radius ρ and centered at the $\mathbf{A}\mathbf{x}^i$, i.e.,

$$\mathcal{B}_i := \mathbf{A}\mathbf{x}^i + \rho B_2^m.$$

Using the noninflating property, we easily see that each ball $\mathcal{B}_i$ is contained in $(c+\rho)B_2^m$. This implies that

$$\mathrm{Vol}\left(\bigcup_{i=1}^n \mathcal{B}_i\right) \leq \mathrm{Vol}\left((c+\rho)B_2^m\right) = (c+\rho)^{2m}\mathrm{Vol}(B_2^m). \tag{4.14}$$

Moreover, we claim that the balls $\mathcal{B}_i$ are disjoint. Indeed, if $\mathcal{B}_i \cap \mathcal{B}_j \neq \emptyset$ for some $i \neq j$, then there would exist $\mathbf{e}^i, \mathbf{e}^j \in B_2^m$ such that $\mathbf{A}\mathbf{x}^i + \rho\mathbf{e}^i = \mathbf{A}\mathbf{x}^j + \rho\mathbf{e}^j =: \mathbf{y}$. Exploiting (4.13), we could then write

$$1 < \|\mathbf{x}^i - \mathbf{x}^j\|_2 \le \|\mathbf{x}^i - \Delta(\mathbf{y})\|_2 + \|\mathbf{x}^j - \Delta(\mathbf{y})\|_2 \le D\|\rho\mathbf{e}^i\|_2 + D\|\rho\mathbf{e}^j\|_2 \le 2D\rho = 1,$$

which is absurd. It follows that

$$\mathrm{Vol}\left(\bigcup_{i=1}^{n} \mathcal{B}_i\right) = \sum_{i=1}^{n} \mathrm{Vol}(\mathcal{B}_i) = n\mathrm{Vol}(\rho B_2^m) = n\rho^{2m}\mathrm{Vol}(B_2^m). \tag{4.15}$$

Putting (4.14) and (4.15) together before making use of (4.11), we obtain

$$\left(1 + \frac{c}{\rho}\right)^{2m} \ge n, \qquad \text{hence} \quad 2m\ln\left(1 + \frac{c}{\rho}\right) \ge \ln(n) \ge c_2 s \ln\left(\frac{N - d(s-1)}{c_1 s}\right).$$

In view of $\rho = 1/(2D)$, we have arrived at

$$\frac{m}{s} \ge c_3 \ln\left(\frac{N - d(s-1)}{c_1 s}\right), \qquad c_3 := \frac{c_2}{2\ln(1 + 2cD)}.$$

To conclude, we take the obvious lower bound $m/s \ge 1$ into account to derive

$$\left(\frac{1}{c_3} + \ln(ec_1)\right)\frac{m}{s} \ge \ln\left(\frac{N - d(s-1)}{c_1 s}\right) + \ln(ec_1) = \ln\left(e\frac{N - d(s-1)}{s}\right).$$

This is the desired result with $C := c_3/(1 + c_3\ln(ec_1))$. □

4.5 Conclusion

We have found that simultaneous knowledge of both sparsity and disjointedness within a signal provides no benefit in reducing the optimal number of measurements needed for uniform robust recovery as compared to knowledge of only one such structure when employing noninflating measurements. Although this result is in the negative, we have still expanded the theory of structured compressive sensing as our class of measurement matrices is more general than the standard Gaussian random matrices and we place no restriction on the method of recovery. Our findings have been rigorously proven by naturally extending the proofs for the result concerning the standard sparse model of Chapter 2 to the case of sparse and disjoint vectors. Additionally, we have developed an effective algorithm for calculating the best s-sparse d-disjoint approximation to a vector via dynamic programming and supplied the associated code for our work in Appendix A.2.

Chapter 5: Conclusion

We have explored the feasibility of applying the techniques of compressive sensing to on-grid MIMO radar and incorporating disjointedness and sparsity together within the compressive sensing framework. While our findings are in the negative, these analyses have highlighted several considerations for future work in these research areas.

For the specific physical problem we established in Chapter 3, we found that our approach performed well when only a few targets were present in the scene. As the signal-to-noise ratio decreased, the reconstruction decayed considerably. Although this method used far fewer measurements than required via the Nyquist rate, more measurements should be taken to improve performance and offset these two impediments. Our data not only reinforce the notion that random positioning of the antennas in MIMO radar enhances the efficacy of the compressive sensing framework but also emphasize the crucial need for fine-tuning of the parameters of the selected reconstruction algorithms and the benefit this can provide. Alternative recovery methods should also be explored as the ℓ_1-squared Nonnegative Regularization method method provided surprisingly superior performance over other approaches.

This research was intended as a first step towards the case of off-grid MIMO radar in azimuth, delay, and Doppler domains, however, all attempts to account for discrepancies between a target's true location and the closest grid location were unsuccessful. Further work is vital in this area for compressive sensing-based MIMO radar systems, and any other adapted radar system, to succeed in practice, even if the number of considered domains is reduced from the three presented here. Depending on the system design, alternative signals may outperform linear chirps, improve target parameter estimation, and enhance the suitability of the resulting measurement matrix for compressive sensing. Additionally, establishing restricted isometry property guarantees for the matrices that result from the various combinations of the previously mentioned parameters is of critical importance for this field.

The experimental investigation of MIMO radar lead to a natural question concerning the structure of the vectors we were attempting to recover. We termed this structure *sparse and disjoint* and provided a complete treatment of its viability within compressive sensing. After discussing the essential details of this structure, we devised a method for projecting onto the set of sparse and disjoint vectors through dynamic programming. This procedure and its adaptation into the classical iterative hard thresholding algorithm were utilized in our experiments probing whether or not this structural knowledge can be leveraged to reduce the minimal number of measurements needed to robustly recover sparse disjointed vectors uniformly. Unfortunately, our findings were in the negative, i.e., simultaneous knowledge of these two structures does not reduce the optimal number of measurements for uniform robust recovery, as compared to knowing just one of these structures, when using a noninflating measurement scheme regardless of the recovery algorithm.

We arrived at the theoretical justifications for this claim by separately examining the sufficient and necessary number of measurements needed for robust recovery. Our results echo the standard theory for sparse vectors. We note that there is still an open question as to whether or not algorithms adapted to sparse and disjoint vectors can provide a significant reduction in the computational complexity of a problem in application. Furthermore, the theoretical implications for non-uniform sparse disjoint robust recovery are still to be explored. Other structures also warrant similar considerations. Deriving theoretical results as presented here will further expand the field of compressive sensing as such developments have the potential to increase its utility in practice. As technological advancements continue to necessitate more efficient methods of data compression that reduce computational complexity and provide higher levels of resolution, i.e., more for less, compressive sensing is poised to play a significant role in addressing these demands.

Bibliography

D. Amelunxen, M. Lotz, M. McCoy, and J. Tropp. Living on the edge: phase transitions in convex programs with random data. *Information and Inference*, 2014. doi: 10.1093/imaiai/iau005.

R. Baraniuk and P. Steeghs. Compressive radar imaging. In *Radar Conference, 2007 IEEE*, pages 128–133. IEEE, 2007.

R. Baraniuk, V. Cevher, M. Duarte, and C. Hegde. Model-based compressive sensing. *Information Theory, IEEE Transactions on*, 56(4):1982–2001, 2010.

T. Blumensath and M. Davies. Sampling theorems for signals from the union of finite-dimensional linear subspaces. *Information Theory, IEEE Transactions on*, 55(4):1872–1882, 2009a.

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

M. Burger, M. Möller, M. Benning, and S. Osher. An adaptive inverse scale space method for compressed sensing. *Mathematics of Computation*, 82(281):269–299, 2013.

E. Candès and C. Fernandez-Granda. Towards a mathematical theory of super-resolution. *Communications on Pure and Applied Mathematics*, 67(6):906–956, 2014.

E. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

E. Candès, T. Strohmer, and V. Voroninski. Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming. *Communications on Pure and Applied Mathematics*, 66(8):1241–1274, 2013.

W. Carrara, R. Goodman, and R. Majewski. *Spotlight synthetic aperture radar - Signal processing algorithms*. Artech House, 1995.

A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM journal on scientific computing*, 20(1):33–61, 1998.

M. Cheney and B. Borden. *Fundamentals of Radar Imaging*. Society for Industrial and Applied Mathematics, 2009.

I. Daubechies, R. DeVore, M. Fornasier, and C. Güntürk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 63(1):1–38, 2010.

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

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

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, 2008.

J. Ender. On compressive sensing applied to radar. *Signal Processing*, 90(5):1402–1414, 2010.

J. Ender. Autofocusing ISAR images via sparse representation. In *Synthetic Aperture Radar, 2012. EUSAR. 9th European Conference on*, pages 203–206. VDE, 2012.

S. Foucart. Hard thresholding pursuit: An algorithm for compressive sensing. *SIAM Journal on Numerical Analysis*, 49(6):2543–2563, 2011a.

S. Foucart. Recovering jointly sparse vectors via hard thresholding pursuit. *Proc. Sampling Theory and Applications (SampTA)*, 2011b.

S. Foucart and D. Koslicki. Sparse recovery by means of nonnegative least squares. *Signal Processing Letters, IEEE*, 21(4):498–502, 2014.

S. Foucart and H. Rauhut. *A mathematical introduction to compressive sensing.* Springer, 2013.

S. Foucart, A. Pajor, H. Rauhut, and T. Ullrich. The Gelfand widths of ℓ_p-balls for $0 < p \leq 1$. *Journal of Complexity*, 26(6):629–640, 2010.

S. Foucart, M. Minner, and T. Needham. Sparse disjointed recovery from noninflating measurements. *Applied and Computational Harmonic Analysis*, 39(3):558–567, 2015.

X. He, C. Liu, B. Liu, and D. Wang. Sparse frequency diverse MIMO radar imaging for off-grid target based on adaptive iterative MAP. *Remote Sensing*, 5(2):631–647, 2013.

C. Hegde, M. Duarte, and V. Cevher. Compressive sensing recovery of spike trains using a structured sparsity model. In *Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, 2009.

C. Hegde, P. Indyk, and L. Schmidt. Approximation algorithms for model-based compressive sensing. *Information Theory, IEEE Transactions on*, 61(9):5129–5147, 2015.

M. Herman and T. Strohmer. High-resolution radar via compressed sensing. *Signal Processing, IEEE Transactions on*, 57(6):2275–2284, 2009.

A. Isaev. *Introduction to mathematical methods in bioinformatics.* Springer Science & Business Media, 2006.

D. Koslicki, S. Foucart, and G. Rosen. Quikr: a method for rapid reconstruction of bacterial communities via compressive sensing. *Bioinformatics*, page btt336, 2013.

J. Li and P. Stoica. *MIMO Radar Signal Processing. 2009.* Wiley, New York, 2009.

M. Lustig, D. Donoho, and J. Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic resonance in medicine*, 58(6):1182–1195, 2007.

M. Minner. Compressed sensing in on-grid MIMO radar. *The Scientific World Journal*, 2015. doi: 10.1155/2015/397878.

D. Needell and J. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

S. Oymak, A. Jalali, M. Fazel, Y. Eldar, and B. Hassibi. Simultaneously structured models with application to sparse and low-rank matrices. *Information Theory, IEEE Transactions on*, 61(5): 2886–2908, 2015.

L. Potter, E. Ertin, J. Parker, and M. Cetin. Sparsity and compressed sensing in radar imaging. *Proceedings of the IEEE*, 98(6):1006–1020, 2010.

R. Prony. Essai expérimental et analytique sur les lois de la dilatabilité des fluides élastiques et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alkool, à différentes températures. *Journal de l'École Polytechnique*, 1(2):24–76, 1795.

M. Richards. *Fundamentals of radar signal processing.* Tata McGraw-Hill Education, 2005.

L. Schmidt, C. Hegde, and P. Indyk. The constrained earth mover distance model, with applications to compressive sensing. In *10th Intl. Conf. on Sampling Theory and Appl.(SAMPTA)*, 2013.

M. Skolnik. *Introduction to Radar Systems.* McGraw-Hill, 2001.

T. Strohmer and B. Friedlander. Compressed sensing for MIMO radar-algorithms and performance. In *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, pages 464–468. IEEE, 2009.

T. Strohmer and H. Wang. Accurate imaging of moving targets via random sensor arrays and kerdock codes. *Inverse Problems*, 29(8):085001, 2013.

H. Taylor, S. Banks, and J. McCoy. Deconvolution with the ℓ_1 norm. *Geophysics*, 44(1):39–52, 1979.

Y. Yu, A. P. Petropulu, and H. V. Poor. Measurement matrix design for compressive sensing–based MIMO radar. *Signal Processing, IEEE Transactions on*, 59(11):5338–5352, 2011.

Y. Zhang, J. Yang, and W. Yin. YALL1: Your ALgorithms for L1, online at yall1.blogs.rice.edu, 2011.

X. Zhu and R. Bamler. Super-resolution power and robustness of compressive sensing for spectral estimation with application to spaceborne tomographic SAR. *Geoscience and Remote Sensing, IEEE Transactions on*, 50(1):247–258, 2012.

Appendix A: Code

This appendix includes reproductions of the Matlab code associated with the presentation of Chapters 3 and 4. Much of the code is already documented in-line but brief explanations are provided to supplement the material.

A.1 Compressive Sensing MIMO Radar

The first experiment detailed in Section 3.5 compares different algorithms at different sparsity levels with a fixed SNR and a fixed threshold value. However, only the L1SQNN method is included to save space. The code may be easily modified to incorporate other algorithms aside from those mentioned in the experiment.

```
%% Experiment 1
% Compressed Sensing in On-Grid MIMO Radar
% Test Simulations, SNR = 30dB, random array
% Only L1 Square Non Negative Regularization displayed
% Michael F Minner

%% Simulation information

% For each sparsity level, generate N_S signals, N_M matrices
Sparsity_Levels=2:2:28;

N_S=10;N_A=10;N_ss=size(Sparsity_Levels,2);  % Number of signals, random arrays,
    sparsity levels
N_Sim=N_S*N_A;                              % Number of simulations for each sparsity
    level

threshold=[.001,.01,.1];                    % Consider different threshold levels for
    detection
N_thresh=size(threshold,2);                 % Number of thresholds considered
```

```

SNR=30;             % Signal to noise ratio in dB

L1SQNN.AvgProbD=zeros(N_ss,N_thresh);
L1SQNN.AvgProbFA=zeros(N_ss,N_thresh);
L1SQNN.AvgRelErr=zeros(N_ss,N_thresh);
L1SQNN.AvgRelErrNT=zeros(N_ss,1);
L1SQNN.AvgTime=zeros(N_ss,1);
L1SQNN.AvgNbrItr=zeros(N_ss,1);

L1SQNN_ProbD=zeros(N_Sim,N_thresh);
L1SQNN_ProbFA=zeros(N_Sim,N_thresh);
L1SQNN_RelErr=zeros(N_Sim,N_thresh);
L1SQNN_RelErrNT=zeros(N_Sim,1);
L1SQNN_Time=zeros(N_Sim,1);
L1SQNN_NbrItr=zeros(N_Sim,1);

%% Initialization

M=5;          % # of Transmitters
N=5;          % # of Receivers.
Q=15;          % # of samples

U=10;          % # of Azimuth Bins
V=10;          % # of Range Bins
W=10;          % # of Doppler Bins

K=U*V*W;       % total # of Bins

P=10;           % sparsity of the target scene

c=3e8;              % speed of light

B=15e6;             % Bandwidth of each transmitted signal 15MHz
```

```
T=2e-6;             % Waveform pulse duration
f_c=1e10;           % Carrier frequency of the first transmitter 10GHz
lambda=c/f_c;       % Reference carrier wavelength, lambda = c/f_c
alpha=B/T;          % Chirp rate, alpha=B/T

Range_U=[-.1,.1];                 % range of angles (-.1 to .1) in radians
Range_V=[990,1010];               % range of distances in meters (980 to 1020 meters)
Range_W=[0,30];                   % range of speeds (0 to 30 meters per second)

Range_tau=2.*Range_V./c;          % range of time delays
Range_upsilon=2/lambda.*Range_W;  % range of Doppler shifts

Delta_U=(Range_U(2)-Range_U(1))/U;        % step size in each domain
Delta_V=(Range_V(2)-Range_V(1))/V;
Delta_W=(Range_W(2)-Range_W(1))/W;

Delta_tau=(Range_tau(2)-Range_tau(1))/V;
Delta_upsilon=(Range_upsilon(2)-Range_upsilon(1))/W;

Values_U=Range_U(1):Delta_U:Range_U(2)-Delta_U; % values for each bin
Values_V=Range_V(1):Delta_V:Range_V(2)-Delta_V; % remove additional final bin
Values_W=Range_W(1):Delta_W:Range_W(2)-Delta_W;

Values_tau=Range_tau(1):Delta_tau:Range_tau(2)-Delta_tau;
Values_upsilon=Range_upsilon(1):Delta_upsilon:Range_upsilon(2)-Delta_upsilon;

% Sample in time only for targets in the scene...
T_d=Range_tau(2)-Range_tau(1);  % Sample time duration
Delta_Q=T_d/Q;                  % Sample times
Values_T_d=Range_tau(1):Delta_Q:Range_tau(2)-Delta_Q;

%% Simulations

timeStart=tic;
```

```
index1=1; % Index for the total number of simulations performed

index4=1; % Index for sparsity
for ss=Sparsity_Levels   % Vary the sparsity level

    index2=1; % Index for number of simulations per sparsity

    % Calculations for measurement matrices
    [~, weight,A_F, A_G, A_H] = LinearChirp( M,Q,V,W,alpha,T,f_c,Values_tau,
    Values_upsilon, Values_T_d );

    for ii=1:N_A          % The number of signals to generate

        % Random Antenna Locations
        [ p,q ] = AntennaPositionRandom(M,N);

        % Build Antenna Arrays
        [A_T,A_R] = AntennaArray( p,q,lambda, M,N,U,Values_U);

        % Measurement Matrix
        [ A ] = MeasurementMatrixVectorized(A_R, A_T, A_F, A_G, A_H, weight,M,N,Q,U,V
    ,W,K);

        for kk=1:N_S

            [signal, grid, supp] = TargetScene(ss,U,V,W,K);

            % All ones on the support
            signal(supp)=ones(ss,1);

            % Take Measurements
            y=A*signal;

            % Complex separation
            x=[real(signal);imag(signal)];
            B=[real(A),-imag(A);imag(A),real(A)];
            z=B*x;

            % Generate Noise
            PowerSignal=norm(signal(supp)).^2;
            NoiseStdDev=sqrt(PowerSignal/(2*M*N*Q*power(10,SNR/10)));              %
    Standard deviation of the noise
            noise=NoiseStdDev.*randn(M*N*Q,1)+NoiseStdDev*1i.*randn(M*N*Q,1);    %
    Complex Gaussian noise

            %noisey measurments
            y=y+noise;
            z=[real(y);imag(y)];

            % RECOVERY

            % L1SQNN
            tStartL1SQNN=tic;
            [xL1SQNN,~,~,NbrItrL1SQNN] = l1sqnnreg(z,B,30);
            L1SQNN_Time(index2,1)=toc(tStartL1SQNN);
            L1SQNN_NbrItr(index2,1)=NbrItrL1SQNN;
            xL1SQNN=xL1SQNN(1:K)+1i.*xL1SQNN(K+1,end);
            L1SQNN_RelErrNT(index2,1)=norm(xL1SQNN-signal)/sqrt(PowerSignal);
            xL1SQNN_Abs=abs(xL1SQNN);

            index3=1;
            for t=threshold

                Removed_L1SQNN=find(xL1SQNN_Abs<t);
                xL1SQNN(Removed_L1SQNN)=0;
                xL1SQNN_Abs(Removed_L1SQNN)=0;
                SuppL1SQNN=find(xL1SQNN_Abs>=t);
                L1SQNN_RelErr(index2,index3)=norm(xL1SQNN-signal)/sqrt(PowerSignal);
                L1SQNN_ProbD(index2,index3)=sum(xL1SQNN_Abs(supp,1)>=t)/ss;
                L1SQNN_ProbFA(index2,index3)=sum(xL1SQNN_Abs(setdiff(SuppL1SQNN,supp)
    ,1)>=t)/(K-ss);
                index3=index3+1;

            end
            display(['current sparsity: ' num2str(ss) ' current simulation: ' num2str
    (index2)])
            index2=index2+1;
            index1=index1+1;
        end
    end

    index4=index4+1;

end
timeStop=toc(timeStart)
```

The second experiment compares a random array and uniform linear array setup in the MIMO radar system.

```
%% Experiment 2
% Compressed Sensing in On-Grid MIMO Radar
% Test Simulations, SNR = 15, 20, 30, Inf dB
% Compare Random Array and Uniform Linear Array (ULA)
% L1 Square Non Negative Regularization
% L1SQNN parameter lambda varies [depends on SNR level]
% Size of the problem is 500*1000
% Sparsity: 3,5,6,9,10,15,20

% Analyze different threshold levels
% [0.0,0.0001,0.001,0.01,0.1,0.2,0.3,0.4,0.5,0.6,0.7,.8,0.9,.99,.999]

% Michael F Minner

```

```
%% Simulation information

% For each sparsity level, generate N_S signals, N_M matrices
Sparsity_Levels=[3,5,6,9,10,15,20];

N_S=25;N_A=20;N_ss=size(Sparsity_Levels,2);  % Number of signals, random arrays,
    sparsity levels
N_Sim=N_S*N_A;                               % Number of simulations for each
    sparsity level

threshold=[0.0,0.0001,0.001,0.01,0.1,0.2,0.3,0.4,0.5,0.6,0.7,.8,0.9,.99,.999];
% Consider different threshold levels for detection

N_thresh=size(threshold,2);                  % Number of thresholds considered

SNR=[Inf,30,20,15];                                        % Signal to Noise Ratio in
    dB
N_snr=size(SNR,2);                           % Number of Signal to Noise Ratios


Par_L1SQ=[24,15,9,5];                          % Consider different parameter values
    for lambda
N_par=size(Par_L1SQ,2);                      % Number of Parameter values

% Average of the probabilities of detection
% and false alarms for each threshold level
% Average relative error with thresholding
% Average relative error without thresholding
% Average time to evaluate OMP
% Average number of iterations for OMP

L1SQNNR.AvgProbD=zeros(N_ss,N_thresh, N_snr);       % Random Array
L1SQNNR.AvgProbFA=zeros(N_ss,N_thresh, N_snr);
L1SQNNR.AvgRelErr=zeros(N_ss,N_thresh, N_snr);
```

```
L1SQNNR.AvgRelErrS=zeros(N_ss,N_thresh, N_snr);
L1SQNNR.AvgTime=zeros(N_ss, 1, N_snr);
L1SQNNR.AvgNbrItr=zeros(N_ss, 1, N_snr);

L1SQNNR_ProbD=zeros(N_Sim,N_thresh, N_snr);
L1SQNNR_ProbFA=zeros(N_Sim,N_thresh, N_snr);
L1SQNNR_RelErr=zeros(N_Sim,N_thresh, N_snr);
L1SQNNR_RelErrS=zeros(N_Sim,N_thresh, N_snr);
L1SQNNR_Time=zeros(N_Sim, 1, N_snr);
L1SQNNR_NbrItr=zeros(N_Sim, 1, N_snr);

L1SQNNU.AvgProbD=zeros(N_ss,N_thresh, N_snr);          % ULA
L1SQNNU.AvgProbFA=zeros(N_ss,N_thresh, N_snr);
L1SQNNU.AvgRelErr=zeros(N_ss,N_thresh, N_snr);
L1SQNNU.AvgRelErrS=zeros(N_ss,N_thresh, N_snr);
L1SQNNU.AvgTime=zeros(N_ss, 1, N_snr);
L1SQNNU.AvgNbrItr=zeros(N_ss, 1, N_snr);

L1SQNNU_ProbD=zeros(N_Sim,N_thresh, N_snr);
L1SQNNU_ProbFA=zeros(N_Sim,N_thresh, N_snr);
L1SQNNU_RelErr=zeros(N_Sim,N_thresh, N_snr);
L1SQNNU_RelErrS=zeros(N_Sim,N_thresh, N_snr);
L1SQNNU_Time=zeros(N_Sim, 1, N_snr);
L1SQNNU_NbrItr=zeros(N_Sim, 1, N_snr);

% each row is a separate simulation,
% each column is a different threshold
% each set of rows and columns corresponds to a different SNR level.

%% Initialization

M=5;          % # of Transmitters
N=5;          % # of Receivers.
Q=20;          % # of samples
```

```

U=10;          % # of Azimuth Bins
V=10;          % # of Range Bins
W=10;          % # of Doppler Bins

K=U*V*W;       % total # of Bins

c=3e8;              % speed of light

B=15e6;             % Bandwidth of each transmitted signal 15MHz
T=2e-6;             % Waveform pulse duration
f_c=1e10;           % Carrier frequency of the first transmitter 10GHz
lambda=c/f_c;       % Reference carrier wavelength, lambda = c/f_c
alpha=B/T;          % Chirp rate, alpha=B/T

Range_U=[-.1,.1];                   % range of angles (-.1 to .1) in radians
Range_V=[990,1010];                 % range of distances in meters (980 to 1020 meters)
Range_W=[0,30];                     % range of speeds (0 to 30 meters per second)

Range_tau=2.*Range_V./c;            % range of time delays
Range_upsilon=2/lambda.*Range_W;    % range of Doppler shifts

Delta_U=(Range_U(2)-Range_U(1))/U;        % step size in each domain
Delta_V=(Range_V(2)-Range_V(1))/V;
Delta_W=(Range_W(2)-Range_W(1))/W;

Delta_tau=(Range_tau(2)-Range_tau(1))/V;
Delta_upsilon=(Range_upsilon(2)-Range_upsilon(1))/W;

Values_U=Range_U(1):Delta_U:Range_U(2)-Delta_U; % values for each bin
Values_V=Range_V(1):Delta_V:Range_V(2)-Delta_V; % remove additional final bin
Values_W=Range_W(1):Delta_W:Range_W(2)-Delta_W;

Values_tau=Range_tau(1):Delta_tau:Range_tau(2)-Delta_tau;
```

```
Values_upsilon=Range_upsilon(1):Delta_upsilon:Range_upsilon(2)-Delta_upsilon;

% Sample in time only for targets in the scene...
T_d=Range_tau(2)-Range_tau(1);  % Sample time duration
Delta_Q=T_d/Q;                  % Sample times
Values_T_d=Range_tau(1):Delta_Q:Range_tau(2)-Delta_Q;

%% Simulations

timeStart=tic;
index1=1; % Index for the total number of simulations performed

index4=1; % Index for sparsity

for ss=Sparsity_Levels   % Vary the sparsity level

    index2=1; % Index for number of simulations per sparsity

    % Calculations for measurement matrices
    [~, weight,A_F, A_G, A_H] = LinearChirp( M,Q,V,W,alpha,T,f_c,Values_tau,
    Values_upsilon, Values_T_d );

    for ii=1:N_A          % The number of matrices to generate

        % Random Antenna Locations
        [ p1,q1 ] = AntennaPositionRandom(M,N);

        % ULA positions
        [ p2,q2] = AntennaPositionUniform(M,N,lambda);

        % Build Antenna Arrays
        [A_T1,A_R1] = AntennaArray( p1,q1,lambda, M,N,U,Values_U);
        [A_T2,A_R2] = AntennaArray( p2,q2,lambda, M,N,U,Values_U);

        % Measurement Matrices
        [ A1 ] = MeasurementMatrixVectorized(A_R1, A_T1, A_F, A_G, A_H, weight,M,N,Q,
    U,V,W,K);
        [ A2 ] = MeasurementMatrixVectorized(A_R2, A_T2, A_F, A_G, A_H, weight,M,N,Q,
    U,V,W,K);

        for kk=1:N_S % The number of signals to generate

            %Mag=2;          % Magnitude of target reflectivities
            [signal, grid, supp] = TargetScene(ss,U,V,W,K);

            % All ones on the support
            signal(supp)=ones(ss,1);

            % Take Measurements
            y1=A1*signal;
            y2=A2*signal;

            % Complex separation
            x=[real(signal);imag(signal)];
            B1=[real(A1),-imag(A1);imag(A1),real(A1)];
            z1=B1*x;

            B2=[real(A2),-imag(A2);imag(A2),real(A2)];
            z2=B2*x;

            % Signal Power
            PowerSignal=norm(signal(supp)).^2;
            snr=SNR;

            index5=1;
            for snr=SNR

                if snr==Inf
```

```

                tStartL1SQNNR=tic;
                display(['current SNR: ' num2str(snr)])
                [xL1SQNNR,~,~,NbrItrL1SQNNR] = l1sqnnreg(z1,B1,Par_L1SQ(index5));
                L1SQNNR_Time(index2,1,index5)=toc(tStartL1SQNNR);
                L1SQNNR_NbrItr(index2,1,index5)=NbrItrL1SQNNR;
                xL1SQNNR=xL1SQNNR(1:K)+1i.*xL1SQNNR(K+1,end);
                xL1SQNNR_Abs=abs(xL1SQNNR);

                tStartL1SQNNU=tic;
                display(['current SNR: ' num2str(snr)])
                [xL1SQNNU,~,~,NbrItrL1SQNNU] = l1sqnnreg(z2,B2,Par_L1SQ(index5));
                L1SQNNU_Time(index2,1,index5)=toc(tStartL1SQNNU);
                L1SQNNU_NbrItr(index2,1,index5)=NbrItrL1SQNNU;
                xL1SQNNU=xL1SQNNU(1:K)+1i.*xL1SQNNU(K+1,end);
                xL1SQNNU_Abs=abs(xL1SQNNU);

            else
                % Generate Noise

                NoiseStdDev=sqrt(PowerSignal/(2*M*N*Q*power(10,snr/10)));
                % Standard deviation of the noise
                noise=NoiseStdDev.*randn(M*N*Q,1)+NoiseStdDev*1i.*randn(M*N*Q,1);
                % Complex Gaussian noise

                %noisey measurments
                y1N=y1+noise;
                z1N=[real(y1N);imag(y1N)];

                y2N=y2+noise;
                z2N=[real(y2N);imag(y2N)];

                % Recovery

```

```
                    % L1SQNN
                    tStartL1SQNNR=tic;
                    display(['current SNR: ' num2str(snr)])
                    [xL1SQNNR,~,~,NbrItrL1SQNNR] = l1sqnnreg(z1N,B1,Par_L1SQ(index5))
    ;
                    L1SQNNR_Time(index2,1,index5)=toc(tStartL1SQNNR);
                    L1SQNNR_NbrItr(index2,1,index5)=NbrItrL1SQNNR;
                    xL1SQNNR=xL1SQNNR(1:K)+1i.*xL1SQNNR(K+1,end);
                    xL1SQNNR_Abs=abs(xL1SQNNR);

                    tStartL1SQNNU=tic;
                    display(['current SNR: ' num2str(snr)])
                    [xL1SQNNU,~,~,NbrItrL1SQNNU] = l1sqnnreg(z2N,B2,Par_L1SQ(index5))
    ;
                    L1SQNNU_Time(index2,1,index5)=toc(tStartL1SQNNU);
                    L1SQNNU_NbrItr(index2,1,index5)=NbrItrL1SQNNU;
                    xL1SQNNU=xL1SQNNU(1:K)+1i.*xL1SQNNU(K+1,end);
                    xL1SQNNU_Abs=abs(xL1SQNNU);

                end

                index3=1;
                for t=threshold

                    Removed_L1SQNNR=find(xL1SQNNR_Abs<=t);
                    xL1SQNNR(Removed_L1SQNNR)=0;
                    xL1SQNNR_Abs(Removed_L1SQNNR)=0;
                    SuppL1SQNNR=find(xL1SQNNR_Abs>t);
                    L1SQNNR_RelErr(index2,index3,index5)=norm(xL1SQNNR-signal)/sqrt(
    PowerSignal);
                    L1SQNNR_ProbD(index2,index3,index5)=sum(xL1SQNNR_Abs(supp,1)>=t)/
    ss;
                    L1SQNNR_ProbFA(index2,index3,index5)=sum(xL1SQNNR_Abs(setdiff(
    SuppL1SQNNR,supp),1)>=t)/(K-ss);
```

```
                    L1SQNNR_RelErrS(index2,index3,index5)=norm(xL1SQNNR(supp)-signal(supp))/sqrt(PowerSignal);


                    Removed_L1SQNNU=find(xL1SQNNU_Abs<=t);
                    xL1SQNNU(Removed_L1SQNNU)=0;
                    xL1SQNNU_Abs(Removed_L1SQNNU)=0;
                    SuppL1SQNNU=find(xL1SQNNU_Abs>t);
                    L1SQNNU_RelErr(index2,index3,index5)=norm(xL1SQNNU-signal)/sqrt(PowerSignal);
                    L1SQNNU_ProbD(index2,index3,index5)=sum(xL1SQNNU_Abs(supp,1)>=t)/ss;
                    L1SQNNU_ProbFA(index2,index3,index5)=sum(xL1SQNNU_Abs(setdiff(SuppL1SQNNU,supp),1)>=t)/(K-ss);
                    L1SQNNU_RelErrS(index2,index3,index5)=norm(xL1SQNNU(supp)-signal(supp))/sqrt(PowerSignal);
                    index3=index3+1;
                end

                index5=index5+1;

            end
            display(['current sparsity: ' num2str(ss) ' current simulation: ' num2str(index2)])
            index2=index2+1;
            index1=index1+1;
        end
    end

    L1SQNNR.AvgProbD(index4,:,:)=mean(L1SQNNR_ProbD,1);
    L1SQNNR.AvgProbFA(index4,:,:)=mean(L1SQNNR_ProbFA,1);
    L1SQNNR.AvgRelErr(index4,:,:)=mean(L1SQNNR_RelErr,1);
    L1SQNNR.AvgRelErrS(index4,:,:)=mean(L1SQNNR_RelErrS,1);
    L1SQNNR.AvgTime(index4,1,:)=mean(L1SQNNR_Time,1);
    L1SQNNR.AvgNbrItr(index4,1,:)=mean(L1SQNNR_NbrItr,1);

    L1SQNNU.AvgProbD(index4,:,:)=mean(L1SQNNU_ProbD,1);
    L1SQNNU.AvgProbFA(index4,:,:)=mean(L1SQNNU_ProbFA,1);
    L1SQNNU.AvgRelErr(index4,:,:)=mean(L1SQNNU_RelErr,1);
    L1SQNNU.AvgRelErrS(index4,:,:)=mean(L1SQNNU_RelErrS,1);
    L1SQNNU.AvgTime(index4,1,:)=mean(L1SQNNU_Time,1);
    L1SQNNU.AvgNbrItr(index4,1,:)=mean(L1SQNNU_NbrItr,1);

    index4=index4+1;
end

timeStop=toc(timeStart)
```

Please note that to avoid redundancy and save space, we do not include the third, fifth or sixth experiments. The code included here can be adapted to reproduce these experiments, as detailed in Section 3.5. The final experiment included here, and the fourth experiment described in Section 3.5, generates families of ROC Curves for L1SQNN as the parameter of β varies at different values of SNR and separate sparsity levels.

```
%% Experiment 4

% Compressed Sensing in On-Grid MIMO Radar
% Test Simulations, SNR = varying Levels (15,20,25,30,50,Inf dB)
% Random array
% L1 Square Non Negative Regularization
% Families of ROC Curves for L1SQNN as parameter lambda varies [depends on SNR level]
% Size of the problem is 500*1000

% Michael F Minner

%% Clear
clear all;

```

```
%% Simulation information

% For each sparsity level, generate N_S signals, N_M matrices
Sparsity_Levels=1:15;

N_S=20;N_A=20;N_ss=size(Sparsity_Levels,2);
% Number of signals, random arrays, sparsity levels
N_Sim=N_S*N_A;
% Number of simulations for each sparsity level

threshold=[0.0,0.0001,0.001,0.01,0.1,0.2,0.3,0.4,0.5,0.6,0.7,.8,0.9,.99,.999];
% Consider different threshold levels for detection
N_thresh=size(threshold,2);              % Number of thresholds considered

SNR=Inf;                                 % Signal to Noise Ratio in dB
N_snr=size(SNR,2);                       % Number of Signal to Noise Ratios

% Consider different parameter values for lambda
Par_L1SQ=[10,15,20,25,30];
N_par=size(Par_L1SQ,2);                        % Number of Parameter values

% Average of the probabilities of detection
% and false alarms for each threshold level
% Average relative error with thresholding
% Average relative error without thresholding
% Average time to evaluate OMP
% Average number of iterations for OMP

L1SQNNR.AvgProbD=zeros(N_ss,N_thresh, N_par);        % Random Array
L1SQNNR.AvgProbFA=zeros(N_ss,N_thresh, N_par);
L1SQNNR.AvgRelErr=zeros(N_ss,N_thresh, N_par);
L1SQNNR.AvgRelErrS=zeros(N_ss,N_thresh, N_par);
L1SQNNR.AvgTime=zeros(N_ss, 1, N_par);
L1SQNNR.AvgNbrItr=zeros(N_ss, 1, N_par);
```

```

L1SQNNR_ProbD=zeros(N_Sim,N_thresh, N_par);
L1SQNNR_ProbFA=zeros(N_Sim,N_thresh, N_par);
L1SQNNR_RelErr=zeros(N_Sim,N_thresh, N_par);
L1SQNNR_RelErrS=zeros(N_Sim,N_thresh, N_par);
L1SQNNR_Time=zeros(N_Sim, 1, N_par);
L1SQNNR_NbrItr=zeros(N_Sim, 1, N_par);

% each row is a separate simulation,
% each column is a different threshold
% each set of rows and columns corresponds to a different SNR level.

%% Initialization

M=5;        % # of Transmitters
N=5;        % # of Receivers.
Q=20;        % # of samples

U=10;        % # of Azimuth Bins
V=10;        % # of Range Bins
W=10;        % # of Doppler Bins

K=U*V*W;      % total # of Bins

c=3e8;            % speed of light

B=15e6;           % Bandwidth of each transmitted signal 15MHz
T=2e-6;           % Waveform pulse duration
f_c=1e10;         % Carrier frequency of the first transmitter 10GHz
lambda=c/f_c;     % Reference carrier wavelength, lambda = c/f_c
alpha=B/T;        % Chirp rate, alpha=B/T

Range_U=[-.1,.1];               % range of angles (-.1 to .1) in radians
Range_V=[990,1010];             % range of distances in meters (980 to 1020 meters)
```

```
Range_W=[0,30];                    % range of speeds (0 to 30 meters per second)

Range_tau=2.*Range_V./c;           % range of time delays
Range_upsilon=2/lambda.*Range_W;   % range of Doppler shifts

Delta_U=(Range_U(2)-Range_U(1))/U;        % step size in each domain
Delta_V=(Range_V(2)-Range_V(1))/V;
Delta_W=(Range_W(2)-Range_W(1))/W;

Delta_tau=(Range_tau(2)-Range_tau(1))/V;
Delta_upsilon=(Range_upsilon(2)-Range_upsilon(1))/W;

Values_U=Range_U(1):Delta_U:Range_U(2)-Delta_U; % values for each bin
Values_V=Range_V(1):Delta_V:Range_V(2)-Delta_V; % remove additional final bin
Values_W=Range_W(1):Delta_W:Range_W(2)-Delta_W;

Values_tau=Range_tau(1):Delta_tau:Range_tau(2)-Delta_tau;
Values_upsilon=Range_upsilon(1):Delta_upsilon:Range_upsilon(2)-Delta_upsilon;

% Sample in time only for targets in the scene...
T_d=Range_tau(2)-Range_tau(1);  % Sample time duration
Delta_Q=T_d/Q;                   % Sample times
Values_T_d=Range_tau(1):Delta_Q:Range_tau(2)-Delta_Q;

%% Simulations

timeStart=tic;
index1=1; % Index for the total number of simulations performed

index4=1; % Index for sparsity

for ss=Sparsity_Levels   % Vary the sparsity level

    index2=1; % Index for number of simulations per sparsity
```

```

% Calculations for measurement matrices
[~, weight,A_F, A_G, A_H] = LinearChirp( M,Q,V,W,alpha,T,f_c,Values_tau,
Values_upsilon, Values_T_d );

for ii=1:N_A          % The number of matrices to generate

    % Random Antenna Locations
    [ p1,q1 ] = AntennaPositionRandom(M,N);

    % Build Antenna Arrays
    [A_T1,A_R1] = AntennaArray( p1,q1,lambda, M,N,U,Values_U);

    % Measurement Matrices
    [ A1 ] = MeasurementMatrixVectorized(A_R1, A_T1, A_F, A_G, A_H, weight,M,N,Q,
U,V,W,K);

    for kk=1:N_S  % The number of signals to generate

        [signal, grid, supp] = TargetScene(ss,U,V,W,K);

        % All ones on the support
        signal(supp)=ones(ss,1);

        % Take Measurements
        y1=A1*signal;

        % Complex separation
        x=[real(signal);imag(signal)];
        B1=[real(A1),-imag(A1);imag(A1),real(A1)];
        z1=B1*x;

        % Signal Power
        PowerSignal=norm(signal(supp)).^2;
```

```
            snr=SNR;
            index5=1;
            for parameter=Par_L1SQ

                if snr==Inf
                    % No noise
                    tStartL1SQNNR=tic;
                    display(['current parameter: ' num2str(parameter)])
                    [xL1SQNNR,~,~,NbrItrL1SQNNR] = l1sqnnreg(z1,B1,parameter);
                    L1SQNNR_Time(index2,1,index5)=toc(tStartL1SQNNR);
                    L1SQNNR_NbrItr(index2,1,index5)=NbrItrL1SQNNR;
                    xL1SQNNR=xL1SQNNR(1:K)+1i.*xL1SQNNR(K+1,end);
                    xL1SQNNR_Abs=abs(xL1SQNNR);

                else
                    % Generate Noise

                    NoiseStdDev=sqrt(PowerSignal/(2*M*N*Q*power(10,snr/10)));
                    % Standard deviation of the noise
                    noise=NoiseStdDev.*randn(M*N*Q,1)+NoiseStdDev*1i.*randn(M*N*Q,1);
                    % Complex Gaussian noise

                    %noisey measurments
                    y1N=y1+noise;
                    z1N=[real(y1N);imag(y1N)];

                    % Recovery

                    % L1SQNN
                    tStartL1SQNNR=tic;
                    display(['current parameter: ' num2str(parameter)])
                    [xL1SQNNR,~,~,NbrItrL1SQNNR] = l1sqnnreg(z1N,B1,parameter);
                    L1SQNNR_Time(index2,1,index5)=toc(tStartL1SQNNR);
                    L1SQNNR_NbrItr(index2,1,index5)=NbrItrL1SQNNR;
                    xL1SQNNR=xL1SQNNR(1:K)+1i.*xL1SQNNR(K+1,end);
                    xL1SQNNR_Abs=abs(xL1SQNNR);
                end

                index3=1;
                for t=threshold

                    Removed_L1SQNNR=find(xL1SQNNR_Abs<t);
                    xL1SQNNR(Removed_L1SQNNR)=0;
                    xL1SQNNR_Abs(Removed_L1SQNNR)=0;
                    SuppL1SQNNR=find(xL1SQNNR_Abs>=t);
                    L1SQNNR_RelErr(index2,index3,index5)=norm(xL1SQNNR-signal)/sqrt(
    PowerSignal);
                    L1SQNNR_ProbD(index2,index3,index5)=sum(xL1SQNNR_Abs(supp,1)>=t)/
    ss;
                    L1SQNNR_ProbFA(index2,index3,index5)=sum(xL1SQNNR_Abs(setdiff(
    SuppL1SQNNR,supp),1)>=t)/(K-ss);
                    L1SQNNR_RelErrS(index2,index3,index5)=norm(xL1SQNNR(supp)-signal(
    supp))/sqrt(PowerSignal);

                    index3=index3+1;
                end

                index5=index5+1;

            end
            display(['current sparsity: ' num2str(ss) ' current simulation: ' num2str
    (index2)])
            index2=index2+1;
            index1=index1+1;
        end
    end

    L1SQNNR.AvgProbD(index4,:,:)=mean(L1SQNNR_ProbD,1);
```

```
    L1SQNNR.AvgProbFA(index4,:,:)=mean(L1SQNNR_ProbFA,1);
    L1SQNNR.AvgRelErr(index4,:,:)=mean(L1SQNNR_RelErr,1);
    L1SQNNR.AvgRelErrS(index4,:,:)=mean(L1SQNNR_RelErrS,1);
    L1SQNNR.AvgTime(index4,1,:)=mean(L1SQNNR_Time,1);
    L1SQNNR.AvgNbrItr(index4,1,:)=mean(L1SQNNR_NbrItr,1);

    index4=index4+1;
end

timeStop=toc(timeStart)
```

The following auxiliary files are called upon in the main experiments. Their functions are detailed with in each piece of code.

Antenna Position (Random)

```
function [ p,q ] = AntennaPositionRandom(M,N)
% Generate the random locations of the M transmit antennas
% and the N receive antennas. Uniform Random Distribution.
% Michael F Minner

 L=N*M/2;               % locations of antennas
 p1=sort(L.*rand(M,1)); % p are transmit antenna locations
 q1=sort(L.*rand(N,1)); % q are receive antenna locations

% distances between antennas
p=p1-p1(1);
q=q1-q1(1);

end
```

Antenna Position (Uniform)

```
function [ p,q] = AntennaPositionUniform(M,N,lambda)
% Generate the random locations of the M transmit antennas
% and the N receive antennas. Uniform Linear Array.
% Michael F Minner

% ULA
p1=2*N*lambda*(1:M).';  % p are transmit antenna locations
q1=lambda*(1:N).';      % q are receive antenna locations

% distances between antennas
p=p1-p1(1);
q=q1-q1(1);
```

Antenna Array

```
function [A_T,A_R] = AntennaArray( p,q,lambda,M,N,U,Values_U)
% Build Antenna Arrays A_T and A_R
% Matrices to store array manifolds for each angle by column

A_T=zeros(M,U);
A_R=zeros(N,U);

a=exp(2*pi*1i/lambda);
a_T=a.^p;             % array manifolds for azimuth domain
a_R=a.^q;

for j=1:U
    A_T(:,j)= a_T.^Values_U(j);
    A_R(:,j)= a_R.^Values_U(j);
end

end
```

Linear Chirp

```
function [ chirp1, weight, A_F, A_G, A_H ] = LinearChirp( M,Q,V,W,alpha,T,f_c,
     Values_tau, Values_upsilon, Values_T_d )
% Create Matrix to store linear chirps where chirp(m,q)=s_m(t_q).
% Also create Angle, Delay and Doppler matrices for measurement matrix
% without vectorization.
% Michael F Minner
```

```

chirp1 = zeros(M,Q);         % Store LFM chirps
A_F = zeros(M,V);            % Values needed for measurement matrix
A_G = zeros(Q,V);
A_H = zeros(Q,W);

t= Values_T_d;               % values of t_q'

f_m = f_c+alpha*T;           % values of f_m = f_c+m*alpha*T

weight=ones(M,1);            % no weights
                             % values of tau_k - tau_0
Values_tau_shifted=Values_tau-Values_tau(1);

for m=1:M
     chirp1(m,:) = exp(1i*2*pi*(.5*alpha.*t.^2+f_m.*t));
     A_F(m,:)=exp(-1i*2*pi*f_m.*Values_tau_shifted);
     f_m = f_m+alpha*T;
end

for q=1:Q
     A_G(q,:) = exp(-1i*2*pi*alpha*t(q).*Values_tau_shifted);
     A_H(q,:) = exp(-1i*2*pi*(t(q)+Values_tau(1)).*Values_upsilon);
end


end
```

Measurement Matrix (Vectorized)

```
function [ A ] = MeasurementMatrixVectorized(A_R, A_T, A_F, A_G, A_H, weight,M,N,Q,U,
     V,W,K)
% Measurement Matrix with Angle, Range and Velocity Vectorized
% Michael F Minner

A=zeros(M*N*Q,K);                 % measurment matrix
```

```
index=1;                     % keep track of which column of A we are filling in

% A_R and A_T store array manifolds for each angle by column

for u=1:U
    for v=1:V
        for w=1:W
            a=A_G(:,v).*A_H(:,w);
            b=kron(A_R(:,u),a);
            c=(weight(:,1).*A_T(:,u)).*A_F(:,v);
            A(:,index)=kron(c,b);
            index=index+1;
        end
    end
end

end
```

Scene Location

```
function [k] = SceneLocate(ii,jj,kk,U,V,W)
% ii,jj,kk is the current grid location
% U is the number of Azimuth bins (not needed in current setup)
% V is the number of Range bins
% W is the number of Doppler bins
% Michael F Minner

k=kk+(jj-1).*W+(ii-1).*(V*W);   % position in U*V*W scene vector

end
```

Target Scene

```
function [ signal, grid, supp] = TargetScene(P,U,V,W,K)
% Create a 'scene' containing P targets.
% Michael F Minner

```

```
signal=zeros(K,1);            % initialize the signal

% build support in azimuth and range, then add Doppler
grid=zeros(3,P);
grid(:,1)=[randsample(U,1);randsample(V,1);randsample(W,1)];    % first target
    location
flag=1;
while flag<P
    ii=randsample(U,1);        % possible azimuth and range location
    jj=randsample(V,1);
    if max((ii==grid(1,:))+(jj==grid(2,:)))~=2  % check if bin is unoccupied
        kk=randsample(W,1);   % doppler for target
        flag=flag+1;
        grid(:,flag)=[ii;jj;kk]; % add new target to support
    end
end
supp=sort(SceneLocate(grid(1,:),grid(2,:),grid(3,:),U,V,W));

end
```

A.2 Sparse Disjoint Vectors

This file contains three separate experiments. The first experiment finds the best s-sparse d-disjointed approximation to a vector, once with the ℓ_2-norm and again with the ℓ_4 norm to highlight the impact of the norm in the approximation. The second experiment recovers a single sparse disjointed vector via the classical IHT and via the DIHT (see Section 4.3). In this example the DIHT provides superior reconstruction. The final experiment, which is detailed at the end of Section 4.2, runs several simulations to compare the efficacy of IHT and DIHT in recovering sparse disjoint vectors.

Sparse Disjoint Experiments

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file contains the reproducible experiments accompanying the article
% SPARSE DISJOINTED RECOVERY FROM NONINFLATING MEASUREMENTS
% by Foucart, Minner, and Needham
% Created 3 Sept 2014
% Last modified 12 Sept 2014
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Experiment 1
% finding best s-sparse d-disjointed approximations
x = [1; 0; 1; 2^(1/4); 1; 0; 2^(-1/2)];
s = 3; d = 1;
% best approximation in the 2-norm --- note that it does not contain the
% largest absolute entry of x
[z2,S2,error2] = baSpaDis(x,s,d,2);
z2 = z2'
% best approximation in the 4-norm --- note that it is different from the
% best approximation in the 2-norm
[z4,S4,error4] = baSpaDis(x,s,d,4);
z4 = z4'

%% Experiment 2
% recovery of a single sparse disjointed via iterative thresholding algorithms

% set the paramaters for the experiments
N = 1000; m = 200; s = 12; d = 65;
% create a sparse disjointed vector
rng(6)  % specify the random number generator to always produce the same experiment
suppCollapsed = sort(randsample(N-d*(s-1),s));
supp = suppCollapsed + d*(0:s-1)';
x = zeros(N,1);
x(supp) = randn(s,1);
% define the measurement matrix and the measurement vector
A = randn(m,N)/sqrt(m);
y = A*x;
% recover using classical IHT
xiht = iht(y,A,s);
recoveryErrorIHT = norm(x-xiht)
```

```
% recover using IHT adapted to the sparse disjointed case
xihtSpaDis = ihtSpaDis(y,A,s,d);
recoveryErrorIHTSpaDis = norm(x-xihtSpaDis)
% visualize the recovered vectors versus the original vector
figure(1)
subplot(1,2,1)
plot(1:N,x,'b*',1:N,xiht,'ro')
legend('Original vector','Recovered vector')
title('Classical IHT')
subplot(1,2,2)
plot(1:N,x,'b*',1:N,xihtSpaDis,'ro')
legend('Original vector','Recovered vector')
title('IHT adapted to the sparse disjointed case')

%% Experiment 3
% percentage of recovery success via iterative thresholding algorithms

% set the paramaters for the experiments (two values of d are considered)
nTests = 100;
N = 1000; m = 200;
d1 = 10; d2 = 20;
smin = 1; smax = 20;
recoveryErrorsIHT1 = nan(nTests,smax-smin+1);
recoveryErrorsIHT2 = nan(nTests,smax-smin+1);
recoveryErrorsIHTSpaDis1 = nan(nTests,smax-smin+1);
recoveryErrorsIHTSpaDis2 = nan(nTests,smax-smin+1);
tic;
for n=1:nTests
   n
   for s = smin:smax
      % create sparse disjointed vectors
      suppCollapsed = sort(randsample(N-d1*(s-1),s));
      supp = suppCollapsed + d1*(0:s-1)';
      x1 = zeros(N,1);
```

```
        x1(supp) = randn(s,1);
        suppCollapsed = sort(randsample(N-d2*(s-1),s));
        supp = suppCollapsed + d2*(0:s-1)';
        x2 = zeros(N,1);
        x2(supp) = randn(s,1);
        % define the measurement matrix and the measurement vector
        A = randn(m,N)/sqrt(m);
        y1 = A*x1;
        y2 = A*x2;
        % recover using classical IHT
        x1iht = iht(y1,A,s);
        recoveryErrorsIHT1(n,s-smin+1) = norm(x1-x1iht);
        x2iht = iht(y2,A,s);
        recoveryErrorsIHT2(n,s-smin+1) = norm(x2-x2iht);
        % recover using IHT adapted to the sparse disjointed case
        x1ihtSpaDis = ihtSpaDis(y1,A,s,d1);
        recoveryErrorsIHTSpaDis1(n,s-smin+1) = norm(x1-x1ihtSpaDis);
        x2ihtSpaDis = ihtSpaDis(y2,A,s,d2);
        recoveryErrorsIHTSpaDis2(n,s-smin+1) = norm(x2-x2ihtSpaDis);
    end
end
timeExp3 = toc;
% percentage of recovery success
tol = 1e-3;
percentageSuccessIHT1 = 100* sum( recoveryErrorsIHT1 < tol ) / nTests ;
percentageSuccessIHT2 = 100* sum( recoveryErrorsIHT2 < tol ) / nTests ;
percentageSuccessIHTSpaDis1 = 100* sum( recoveryErrorsIHTSpaDis1 < tol ) / nTests ;
percentageSuccessIHTSpaDis2 = 100* sum( recoveryErrorsIHTSpaDis2 < tol ) / nTests ;
save('Exp3.mat')

%% visualization of the results
load('Exp3.mat')
figure(2)
% success of IHT and IHTSpaDis for d=d1
```

```
subplot(1,2,1)
plot(smin:smax,percentageSuccessIHT1,'b',smin:smax,percentageSuccessIHTSpaDis1,'g')
legend('Classical IHT',['IHT adapted' 10 'to the sparse' 10 'disjointed case'],...
    'Location', 'SouthWest')
title(strcat('Percentage of successful recoveries for d=',num2str(d1)))
xlabel('sparsity level s')
ylabel('percentage of successful recoveries')
% success of IHT and IHTSpaDis for d=d2
subplot(1,2,2)
plot(smin:smax,percentageSuccessIHT2,'b',smin:smax,percentageSuccessIHTSpaDis2,'g')
legend('Classical IHT',['IHT adapted' 10 'to the sparse' 10 'disjointed case'],...
    'Location', 'SouthWest')
title(strcat('Percentage of successful recoveries for d=',num2str(d2)))
xlabel('sparsity level s')
ylabel('percentage of successful recoveries')
```

The following functions are required for the above experiments. Their primary purposes are detailed immediately within each file. The first is the dynamic programming algorithm for finding the best s-sparse d-disjointed approximation to a vector.

```
% baSpaDis  Computes best sparse disjointed approximations by dynamic programming
%
% Finds the best approximation to a vector x
% by an s-sparse d-disjointed vector in the p-norm,
% i.e., the minimizer z of norm(x,z,p)
% subject to z is s-sparse d-disjointed
%
% Usage: [z,S,error] = baSpaDis(x,s,d,p)
%
% x: the vector to be approximated
% s: the sparsity level
% d: the disjointness parameter
% p: the exponent of the norm (optional, default=2)
%
% z: the best s-sparse d-disjointed approximation
% S: the support of the best approximation
% error: the error of best approximation, i.e., norm(x-z,p)

% Written by Simon Foucart and Michael Minner
% Send comments to simon.foucart@centraliens.net

function [z,S,error] = baSpaDis(x,s,d,p)

if nargin < 4
   p=2;
end

N = length(x);
absxp = abs(x).^p;
values = nan(N,s+1);
pointers = nan(N,s+1);

% initialize of the table of values
% first column
values(:,1) = cumsum(absxp);
% first d+1 rows
for n=1:d+1
   values(n,2:s+1) = (sum(absxp(1:n))-max(absxp(1:n))) * ones(1,s);
end

% fill in the table of values, row by row
for n=d+2:N
   for rr=2:s+1   %rr represents r+1
      val1 = values(n-1,rr)+absxp(n);
      val2 = values(n-d-1,rr-1)+sum(absxp(n-d:n-1));
      if val1 < val2
         values(n,rr) = val1;
         pointers(n,rr) = 0;   % "0" means that the arrow points north
      else
```

```
            values(n,rr) = val2;
            pointers(n,rr) = 1;   % "1" means that the arrow points northeast
        end
    end
end

% return the error of best approximation
error = values(N,s+1)^(1/p);

% construct the support by backtracking
n = N;
rr = s+1;
S = [];
while ( n > d+1 && rr > 1 )
    if pointers(n,rr) == 0
        n = n-1;
    else
        S = [n,S]; % where is the right place?
        n = n-d-1;
        rr = rr-1;
    end
end
if rr > 1
    [~,j] = max(absxp(1:n));
    S = [j,S];
end

% return the best approximation
z = zeros(N,1);
z(S) = x(S);

end
```

The next function is an implementation of the classical iterative hard thresholding algorithm.

```
% iht  runs the Iterative Hard Thresholding
%
% Usage: [x,S,normRes,nIter] = iht(y,A,s,x0,maxnIter,tolRes)
%
% y: the measurement vector
% A: the measurement matrix
% s: the sparsity parameter
% x0: initial vector (optional, default=0)
% maxnIter: number of iterations not to be exceeded (optional, default=500)
% tolRes: threshold for the L2-norm of the residual to stop the algorithm (optional,
    default=1e-4)
%
% x: column vector
% S: support of x
% normRes: the norm of the residual
% nbIter: the number of performed iterations
%
% SF and MFM (created 27 May 2012, modified 9 Sept 2014)

function [x,S,normRes,nIter] = iht(y,A,s,x0,maxnIter,tolRes)

[~,N]=size(A);
if nargin < 6
   tolRes = 1e-4;
end
if nargin < 5
   maxnIter = 500;
end
if nargin < 4
   x0=zeros(N,1);
end

x = x0;
res = y-A*x;
```

```
normRes = norm(res);
nIter=0;

while ( nIter < maxnIter && normRes > tolRes )
    u = x+A'*res;
    [~,sorted_idx] = sort(abs(u),'descend');
    S = sorted_idx(1:s);
    x=zeros(N,1); x(S)=u(S);
    res = y-A*x;
    normRes=norm(res);
    nIter = nIter+1;
end

end
```

Finally, we present the sparse disjointed iterative hard thresholding method.

```
% ihtSpaDis  runs the Sparse Disjointed Iterative Hard Thresholding
%
% Usage: [x,S,normRes,nIter] = ihtSpaDis(y,A,s,d,x0,maxnIter,tolRes)
%
% y: the measurement vector
% A: the measurement matrix
% s: the sparsity parameter
% d: the disjointness parameter
% x0: initial vector (optional, default=0)
% maxnIter: number of iterations not to be exceeded (optional, default=500)
% tolRes: threshold for the L2-norm of the residual to stop the algorithm (optional,
    default=1e-4)
%
% x: column vector
% S: support of x
% normRes: the norm of the residual
% nbIter: the number of performed iterations
%
```

```
% SF and MFM (created 3 Sept 2014, modified 12 Sept 2014)

function [x,S,normRes,nIter] = ihtSpaDis(y,A,s,d,x0,maxnIter,tolRes)

[~,N]=size(A);
if nargin < 7
   tolRes = 1e-4;
end
if nargin < 6
   maxnIter = 500;
end
if nargin < 5
   x0=zeros(N,1);
end

x = x0;
res = y-A*x;
normRes = norm(res);
nIter=0;

while ( nIter < maxnIter && normRes > tolRes )
    u = x+A'*res;
    x = baSpaDis(u,s,d);
    res = y-A*x;
    normRes=norm(res);
    nIter = nIter+1;
end

if nIter == maxnIter
    fprintf('Maximum number of iterations reached \n')
end

if normRes <= tolRes
    fprintf('Tolerance on the norm of the residual reached \n')
```

```
end

end
```

Vita

Michael F. Minner

Drexel University, Department of Mathematics, michael.f.minner@gmail.com

Education

Ph.D. Mathematics, Drexel University, Philadelphia, PA (September 2010 - March 2016)

Advisor: Dr. Simon Foucart. GPA: 3.98

M.S. Mathematics, Drexel University, Philadelphia, PA, (2012) GPA: 3.95

B.S. Mathematics and Physics, Temple University, Philadelphia, PA, (2006 - 2010) GPA: 3.94

Summa cum Laude, Distinction in Major, Dean's List, Honors Program Certificate

Publications

Compressed Sensing in On-Grid MIMO Radar, The Scientific World Journal, 2015.

Sparse Disjointed Recovery from Noninflating Measurements, Applied and Computational Harmonic Analysis, 2015, with S. Foucart and T. Needham.

Experience

Graduate Teaching Assistant, Drexel University, Fall 2010 - Winter 2016

Sandia National Laboratories Internship, Livermore, California, Summer 2014 & 2015

Industrial Mathematics and Statistical Modeling Workshop, Raleigh, North Carolina, July 2014

4th International Summer School on Radar/SAR, Rolandseck, Germany, July 2012

Co-President, Vice-President, Drexel's SIAM *Student Chapter*, Fall 2012 - 2014

President, MathBytes Graduate Student Organization, Fall 2011 - Summer 2012

Honors and Awards

Albert Herr Teaching Assistant Award, Math Department, Drexel University, Spring 2014

Highly Commended Teaching Excellence Award Drexel University, Spring 2014

Student Chapter Certificate of Recognition, SIAM, Drexel University, Spring 2014

Graduate Student Travel Award, Drexel University, Winter 2013