5-1-2009

Topics in Compressed Sensing

Deanna Needell
Claremont McKenna College

Topics in Compressed Sensing

By

Deanna Needell
B.S. (University of Nevada, Reno) 2003
M.A. (University of California, Davis) 2005

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

MATHEMATICS

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Roman Vershynin
Roman Vershynin (Co-Chair)

Thomas Strohmer
Thomas Strohmer (Co-Chair)

Jesús DeLoera
Jesús DeLoera

Committee in Charge
2009

I dedicate this dissertation to all my loved ones, those with me today and those who have passed on.

Contents

1 Introduction **1**

1.1 Overview 1

1.1.1 Main Idea 1

1.1.2 Problem Formulation 3

1.2 Applications 6

1.2.1 Error Correction 6

1.2.2 Imaging 7

1.2.3 Radar 9

2 Major Algorithmic Approaches **10**

2.1 Basis Pursuit 10

2.1.1 Description 11

2.1.2 Restricted Isometry Condition 12

2.1.3 Main Theorems 15

2.1.4 Numerical Results 18

2.1.5 Linear Programming 19

2.1.6 Summary 22

2.2 Greedy Methods 23

2.2.1 Orthogonal Matching Pursuit 23
2.2.2 Stagewise Orthogonal Matching Pursuit 27
2.2.3 Combinatorial Methods 32

3 Contributions **36**

3.1 Regularized Orthogonal Matching Pursuit 36
3.1.1 Description 36
3.1.2 Main Theorems 39
3.1.3 Proofs of Theorems 43
3.1.4 Implementation and Runtime 63
3.1.5 Numerical Results 64
3.1.6 Summary 68
3.2 Compressive Sampling Matching Pursuit 70
3.2.1 Description 71
3.2.2 Main Theorem 74
3.2.3 Proofs of Theorems 77
3.2.4 Implementation and Runtime 105
3.2.5 Numerical Results 112
3.2.6 Summary 116
3.3 Reweighted L1-Minimization 120
3.3.1 Main Results 123
3.3.2 Proofs 125
3.3.3 Numerical Results and Convergence 132
3.4 Randomized Kaczmarz 136
3.4.1 Main Results 139
3.4.2 Numerical Examples 142

4 Summary **145**

A Matlab Code **149**

A.1 Basis Pursuit 149

A.2 Orthogonal Matching Pursuit 152

A.3 Regularized Orthogonal Matching Pursuit 154

A.4 Compressive Sampling Matching Pursuit 156

A.5 Reweighted L1 Minimization 158

A.6 Randomized Kaczmarz 160

References **162**

Topics in Compressed Sensing

Abstract

Compressed sensing has a wide range of applications that include error correction, imaging, radar and many more. Given a sparse signal in a high dimensional space, one wishes to reconstruct that signal accurately and efficiently from a number of linear measurements much less than its actual dimension. Although in theory it is clear that this is possible, the difficulty lies in the construction of algorithms that perform the recovery efficiently, as well as determining which kind of linear measurements allow for the reconstruction. There have been two distinct major approaches to sparse recovery that each present different benefits and shortcomings. The first, ℓ_1-minimization methods such as Basis Pursuit, use a linear optimization problem to recover the signal. This method provides strong guarantees and stability, but relies on Linear Programming, whose methods do not yet have strong polynomially bounded runtimes. The second approach uses greedy methods that compute the support of the signal iteratively. These methods are usually much faster than Basis Pursuit, but until recently had not been able to provide the same guarantees. This gap between the two approaches was bridged when we developed and analyzed the greedy algorithm Regularized Orthogonal Matching Pursuit (ROMP). ROMP provides similar guarantees to Basis Pursuit as well as the speed of a greedy algorithm. Our more recent algorithm Compressive Sampling Matching Pursuit (CoSaMP) improves upon these guarantees, and is optimal in every important aspect. Recent work has also been done on a reweighted version of the ℓ_1-minimization method that improves upon the original version in the recovery error and measurement requirements. These algorithms are discussed in detail, as well as previous work that serves as a foundation for sparse signal recovery.

Acknowledgments

There were many people who helped me in my graduate career, and I would like to take this opportunity to thank them.

I am extremely grateful to my adviser, Prof. Roman Vershynin. I have been very fortunate to have an adviser who gave me the freedom to explore my area on my own, and also the incredible guidance to steer me down a different path when I reached a dead end. Roman taught me how to think about mathematics in a way that challenges me and helps me succeed. He is an extraordinary teacher, both in the classroom and as an adviser.

My co-adviser, Prof. Thomas Strohmer, has been very helpful throughout my graduate career, and especially in my last year. He always made time for me to talk about ideas and review papers, despite an overwhelming schedule. His insightful comments and constructive criticisms have been invaluable to me.

Prof. Jesús DeLoera has been very inspiring to me throughout my time at Davis. His instruction style has had a great impact on the way I teach, and I strive to make my classroom environment as enjoyable as he always makes his.

Prof. Janko Gravner is an amazing teacher, and challenges his students to meet high standards. The challenges he presented me have given me a respect and fondness for the subject that still helps in my research today.

I would also like to thank all the staff at UC Davis, and especially Celia Davis, Perry Gee, and Jessica Potts for all their help and patience. They had a huge impact on my success at Davis, and I am so grateful for them.

My friends both in Davis and afar have helped to keep me sane throughout the graduate process. Our time spent together has been a blessing, and I thank them for

all they have done for me. I especially thank Blake for all his love and support. His unwaivering confidence in my abilities has made me a stronger person.

My very special thanks to my family. My father, George, has always provided me with quiet encouragement and strength. My mother, Debbie, has shown me how to persevere through life challenges and has always been there for me. My sister, Crystal, has helped me keep the joy in my life, and given me a wonderful nephew, Jack. My grandmother and late grandfather were a constant support and helped teach me the rewards of hard work.

Finally, thank you to the NSF for the financial support that helped fund parts of the research in this dissertation.

Chapter 1

Introduction

1.1 Overview

1.1.1 Main Idea

The phrase *compressed sensing* refers to the problem of realizing a sparse input x using few linear measurements that possess some incoherence properties. The field originated recently from an unfavorable opinion about the current signal compression methodology. The conventional scheme in signal processing, acquiring the entire signal and then compressing it, was questioned by Donoho [20]. Indeed, this technique uses tremendous resources to acquire often very large signals, just to throw away information during compression. The natural question then is whether we can combine these two processes, and directly sense the signal or its essential parts using few linear measurements. Recent work in compressed sensing has answered this question in positive, and the field continues to rapidly produce encouraging results.

The key objective in compressed sensing (also referred to as sparse signal recovery or compressive sampling) is to reconstruct a signal accurately and efficiently from a

set of few non-adaptive linear measurements. Signals in this context are vectors, many of which in the applications will represent images. Of course, linear algebra easily shows that in general it is not possible to reconstruct an arbitrary signal from an incomplete set of linear measurements. Thus one must restrict the domain in which the signals belong. To this end, we consider *sparse* signals, those with few non-zero coordinates. It is now known that many signals such as real-world images or audio signals are sparse either in this sense, or with respect to a different basis.

Since sparse signals lie in a lower dimensional space, one would think intuitively that they may be represented by few linear measurements. This is indeed correct, but the difficulty is determining in which lower dimensional subspace such a signal lies. That is, we may know that the signal has few non-zero coordinates, but we do not know which coordinates those are. It is thus clear that we may not reconstruct such signals using a simple linear operator, and that the recovery requires more sophisticated techniques. The compressed sensing field has provided many recovery algorithms, most with provable as well as empirical results.

There are several important traits that an optimal recovery algorithm must possess. The algorithm needs to be fast, so that it can efficiently recover signals in practice. Of course, minimal storage requirements as well would be ideal. The algorithm should provide uniform guarantees, meaning that given a specific method of acquiring linear measurements, the algorithm recovers all sparse signals (possibly with high probability). Ideally, the algorithm would require as few linear measurements as possible. Linear algebra shows us that if a signal has s non-zero coordinates, then recovery is theoretically possible with just $2s$ measurements. However, recovery using only this property would require searching through the exponentially large set of all possible lower dimensional subspaces, and so in practice is not numerically fea-

sible. Thus in the more realistic setting, we may need slightly more measurements. Finally, we wish our ideal recovery algorithm to be stable. This means that if the signal or its measurements are perturbed slightly, then the recovery should still be approximately accurate. This is essential, since in practice we often encounter not only noisy signals or measurements, but also signals that are not exactly sparse, but close to being sparse. For example, *compressible* signals are those whose coefficients decay according to some power law. Many signals in practice are compressible, such as smooth signals or signals whose variations are bounded.

1.1.2 Problem Formulation

Since we will be looking at the reconstruction of sparse vectors, we need a way to quantify the sparsity of a vector. We say that a d-dimensional signal x is s-sparse if it has s or fewer non-zero coordinates:

$$x \in \mathbb{R}^d, \quad \|x\|_0 := |\mathrm{supp}(x)| \leq s \ll d,$$

where we note that $\|\cdot\|_0$ is a quasi-norm. For $1 \leq p < \infty$, we denote by $\|\cdot\|_p$ the usual p-norm,

$$\|x\|_p := \big(\sum_{i=1}^{d} |x_i|^p\big)^{1/p},$$

and $\|x\|_\infty = \max |x_i|$. In practice, signals are often encountered that are not exactly sparse, but whose coefficients decay rapidly. As mentioned, compressible signals are those satisfying a power law decay:

$$|x_i^*| \leq Ri^{(-1/q)}, \tag{1.1.1}$$

where x^* is a non-increasing rearrangement of x, R is some positive constant, and $0 < q < 1$. Note that in particular, sparse signals are compressible.

Sparse recovery algorithms reconstruct sparse signals from a small set of non-adaptive linear measurements. Each measurement can be viewed as an inner product with the signal $x \in \mathbb{R}^d$ and some vector $\phi_i \in \mathbb{R}^d$ (or in $\mathbb{C}^d$)[1]. If we collect m measurements in this way, we may then consider the $m \times d$ measurement matrix Φ whose columns are the vectors ϕ_i. We can then view the sparse recovery problem as the recovery of the s-sparse signal $x \in \mathbb{R}^d$ from its measurement vector $u = \Phi x \in \mathbb{R}^m$. One of the theoretically simplest ways to recover such a vector from its measurements $u = \Phi x$ is to solve the ℓ_0-minimization problem

$$\min_{z \in \mathbb{R}^d} \|z\|_0 \quad \text{subject to} \quad \Phi z = u. \tag{1.1.2}$$

If x is s-sparse and Φ is one-to-one on all $2s$-sparse vectors, then the minimizer to (1.1.2) must be the signal x. Indeed, if the minimizer is z, then since x is a feasible solution, z must be s-sparse as well. Since $\Phi z = u$, $z - x$ must be in the kernel of Φ. But $z - x$ is $2s$-sparse, and since Φ is one-to-one on all such vectors, we must have that $z = x$. Thus this ℓ_0-minimization problem works perfectly in theory. However, it is not numerically feasible and is NP-Hard in general [49, Sec. 9.2.2].

Fortunately, work in compressed sensing has provided us numerically feasible alternatives to this NP-Hard problem. One major approach, Basis Pursuit, relaxes the ℓ_0-minimization problem to an ℓ_1-minimization problem. Basis Pursuit requires a condition on the measurement matrix Φ stronger than the simple injectivity on sparse vectors, but many kinds of matrices have been shown to satisfy this condition with

[1] Although similar results hold for measurements taken over the complex numbers, for simplicity of presentation we only consider real numbers throughout.

number of measurements $m = s \log^{O(1)} d$. The ℓ_1-minimization approach provides uniform guarantees and stability, but relies on methods in Linear Programming. Since there is yet no known strongly polynomial bound, or more importantly, no *linear bound* on the runtime of such methods, these approaches are often not optimally fast.

The other main approach uses greedy algorithms such as Orthogonal Matching Pursuit [62], Stagewise Orthogonal Matching Pursuit [23], or Iterative Thresholding [27, 3]. Many of these methods calculate the support of the signal iteratively. Most of these approaches work for specific measurement matrices with number of measurements $m = O(s \log d)$. Once the support S of the signal has been calculated, the signal x can be reconstructed from its measurements $u = \Phi x$ as $x = (\Phi_S)^\dagger u$, where Φ_S denotes the measurement matrix Φ restricted to the columns indexed by S and $\dagger$ denotes the pseudoinverse. Greedy approaches are fast, both in theory and practice, but have lacked both stability and uniform guarantees.

There has thus existed a gap between the approaches. The ℓ_1-minimization methods have provided strong guarantees but have lacked in optimally fast runtimes, while greedy algorithms although fast, have lacked in optimal guarantees. We bridged this gap in the two approaches with our new algorithm Regularized Orthogonal Matching Pursuit (ROMP). ROMP provides similar uniform guarantees and stability results as those of Basis Pursuit, but is an iterative algorithm so also provides the speed of the greedy approach. Our next algorithm, Compressive Sampling Matching Pursuit (CoSaMP) improves upon the results of ROMP, and is the first algorithm in sparse recovery to be provably optimal in every important aspect.

1.2 Applications

The sparse recovery problem has applications in many areas, ranging from medicine and coding theory to astronomy and geophysics. Sparse signals arise in practice in very natural ways, so compressed sensing lends itself well to many settings. Three direct applications are error correction, imaging, and radar.

1.2.1 Error Correction

When signals are sent from one party to another, the signal is usually encoded and gathers errors. Because the errors usually occur in few places, sparse recovery can be used to reconstruct the signal from the corrupted encoded data. This error correction problem is a classic problem in coding theory. Coding theory usually assumes the data values live in some finite field, but there are many practical applications for encoding over the continuous reals. In digital communications, for example, one wishes to protect results of onboard computations that are real-valued. These computations are performed by circuits that experience faults caused by effects of the outside world. This and many other examples are difficult real-world problems of error correction.

The error correction problem is formulated as follows. Consider a m-dimensional input vector $f \in \mathbb{R}^m$ that we wish to transmit reliably to a remote receiver. In coding theory, this vector is referred to as the "plaintext." We transmit the measurements $z = Af$ (or "ciphertext") where A is the $d \times m$ measurement matrix, or the *linear code*. It is clear that if the linear code A has full rank, we can recover the input vector f from the ciphertext z. But as is often the case in practice, we consider the setting where the ciphertext z has been corrupted. We then wish to reconstruct the input signal f from the corrupted measurements $z' = Af + e$ where $e \in \mathbb{R}^N$ is the sparse error vector. To realize this in the usual compressed sensing setting, consider

a matrix B whose kernel is the range of A. Apply B to both sides of the equation $z' = Af + e$ to get $Bz' = Be$. Set $y = Bz'$ and the problem becomes reconstructing the sparse vector e from its linear measurements y. Once we have recovered the error vector e, we have access to the actual measurements Af and since A is full rank can recover the input signal f.

1.2.2 Imaging

Many images both in nature and otherwise are sparse with respect to some basis. Because of this, many applications in imaging are able to take advantage of the tools provided by Compressed Sensing. The typical digital camera today records every pixel in an image before compressing that data and storing the compressed image. Due to the use of silicon, everyday digital cameras today can operate in the megapixel range. A natural question asks why we need to acquire this abundance of data, just to throw most of it away immediately. This notion sparked the emerging theory of Compressive Imaging. In this new framework, the idea is to directly acquire random linear measurements of an image without the burdensome step of capturing every pixel initially.

Several issues from this of course arise. The first problem is how to reconstruct the image from its random linear measurements. The solution to this problem is provided by Compressed Sensing. The next issue lies in actually sampling the random linear measurements without first acquiring the entire image. Researchers [25] are working on the construction of a device to do just that. Coined the "single-pixel" compressive sampling camera, this camera consists of a digital micromirror device (DMD), two lenses, a *single* photon detector and an analog-to-digital (A/D) converter. The first lens focuses the light onto the DMD. Each mirror on the DMD corresponds to a pixel

in the image, and can be tilted toward or away from the second lens. This operation is analogous to creating inner products with random vectors. This light is then collected by the lens and focused onto the photon detector where the measurement is computed. This optical computer computes the random linear measurements of the image in this way and passes those to a digital computer that reconstructs the image.

Since this camera utilizes only one photon detector, its design is a stark contrast to the usual large photon detector array in most cameras. The single-pixel compressive sampling camera also operates at a much broader range of the light spectrum than traditional cameras that use silicon. For example, because silicon cannot capture a wide range of the spectrum, a digital camera to capture infrared images is much more complex and costly.

Compressed Sensing is also used in medical imaging, in particular with magnetic resonance (MR) images which sample Fourier coefficients of an image. MR images are implicitly sparse and can thus capitalize on the theories of Compressed Sensing. Some MR images such as angiograms are sparse in their actual pixel representation, whereas more complicated MR images are sparse with respect to some other basis, such as the wavelet Fourier basis. MR imaging in general is very time costly, as the speed of data collection is limited by physical and physiological constraints. Thus it is extremely beneficial to reduce the number of measurements collected without sacrificing quality of the MR image. Compressed Sensing again provides exactly this, and many Compressed Sensing algorithms have been specifically designed with MR images in mind [36, 46].

1.2.3 Radar

There are many other applications to compressed sensing (see [13]), and one additional application is Compressive Radar Imaging. A standard radar system transmits some sort of pulse (for example a linear chirp), and then uses a matched filter to correlate the signal received with that pulse. The receiver uses a pulse compression system along with a high-rate analog to digital (A/D) converter. This conventional approach is not only complicated and expensive, but the resolution of targets in this classical framework is limited by the radar uncertainty principle. Compressive Radar Imaging tackles these problems by discretizing the time-frequency plane into a grid and considering each possible target scene as a matrix. If the number of targets is small enough, then the grid will be sparsely populated, and we can employ Compressed Sensing techniques to recover the target scene. See [1, 39] for more details.

Chapter 2

Major Algorithmic Approaches

Compressed Sensing has provided many methods to solve the sparse recovery problem and thus its applications. There are two major algorithmic approaches to this problem. The first relies on an optimization problem which can be solved using linear programming, while the second approach takes advantage of the speed of greedy algorithms. Both approaches have advantages and disadvantages which are discussed throughout this chapter along with descriptions of the algorithms themselves. First we discuss Basis Pursuit, a method that utilizes a linear program to solve the sparse recovery problem.

2.1 Basis Pursuit

Recall that sparse recovery can be formulated as the generally NP-Hard problem (1.1.2) to recover a signal x. Donoho and his collaborators showed (see e.g. [21]) that for certain measurement matrices Φ, this hard problem is equivalent to its relaxation,

$$\min_{z \in \mathbb{R}^d} \|z\|_1 \quad \text{subject to} \quad \Phi z = u. \tag{2.1.1}$$

Candès and Tao proved that for measurement matrices satisfying a certain quantitative property, the programs (1.1.2) and (2.1.1) are equivalent [6].

2.1.1 Description

Since the problem (1.1.2) is not numerically feasible, it is clear that if one is to solve the problem efficiently, a different approach is needed. At first glance, one may instead wish to consider the mean square approach, based on the minimization problem,

$$\min_{z\in\mathbb{R}^d} \|z\|_2 \quad \text{subject to} \quad \Phi z = u. \tag{2.1.2}$$

Since the minimizer x^* must satisfy $\Phi x^* = u = \Phi x$, the minimizer must be in the subspace $K \stackrel{\text{def}}{=} x + \ker \Phi$. In fact, the minimizer x^* to (2.1.2) is the contact point where the smallest Euclidean ball centered at the origin meets the subspace K. As is illustrated in Figure 2.1.1, this contact point need not coincide with the actual signal x. This is because the geometry of the Euclidean ball does not lend itself well to detecting sparsity.

We may then wish to consider the ℓ_1-minimization problem (2.1.1). In this case, the minimizer x^* to (2.1.1) is the contact point where the smallest octahedron centered at the origin meets the subspace K. Since x is sparse, it lies in a low-dimensional coordinate subspace. Thus the octahedron has a wedge at x (see Figure 2.1.1), which forces the minimizer x^* to coincide with x for many subspaces K.

Since the ℓ_1-ball works well because of its geometry, one might think to then use an ℓ_p ball for some $0 < p < 1$. The geometry of such a ball would of course lend itself even better to sparsity. Indeed, some work in compressed sensing has used this approach (see e.g. [28, 17]), however, recovery using such a method has not yet provided optimal results. The program (2.1.1) has the advantage over those

with $p < 1$ because linear programming can be used to solve it. See Section 2.1.5 for a discussion on linear programming. Basis Pursuit utilizes the geometry of the octahedron to recover the sparse signal x using measurement matrices Φ that satisfy a deterministic property.

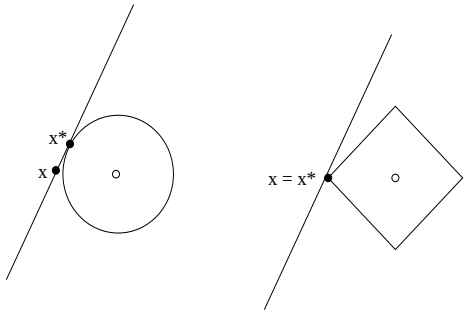


Figure 2.1.1: The minimizers to the mean square (left) and ℓ_1 (right) approaches.

2.1.2 Restricted Isometry Condition

As discussed above, to guarantee exact recovery of every s-sparse signal, the measurement matrix Φ needs to be one-to-one on all $2s$-sparse vectors. Candès and Tao [6] showed that under a slightly stronger condition, Basis Pursuit can recover every s-sparse signal by solving (2.1.1). To this end, we say that the restricted isometry condition (RIC) holds with parameters (r, δ) if

$$(1 - \delta)\|x\|_2 \leq \|\Phi x\|_2 \leq (1 + \delta)\|x\|_2 \tag{2.1.3}$$

holds for all r-sparse vectors x. Often, the quadratic form

$$(1 - \delta)\|x\|_2^2 \leq \|\Phi x\|_2^2 \leq (1 + \delta)\|x\|_2^2 \tag{2.1.4}$$

is used for simplicity. Often the notation δ_r is used to denote the smallest δ for which the above holds for all r-sparse signals. Now if we require δ to be small, this condition essentially means that every subset of r or fewer columns of Φ is approximately an orthonormal system. Note that if the restricted isometry condition holds with parameters $(2s, 1)$, then Φ must be one-to-one on all s-sparse signals. Indeed, if $\Phi x = \Phi z$ for two s-sparse vectors x and z, then $\Phi(x-z) = 0$, so by the left inequality, $\|x - z\|_2 = 0$. To use this restricted isometry condition in practice, we of course need to determine what kinds of matrices have small restricted isometry constants, and how many measurements are needed. Although it is quite difficult to check whether a given matrix satisfies this condition, it has been shown that many matrices satisfy the restricted isometry condition with high probability and few measurements. In particular, it has been show that with exponentially high probability, random Gaussian, Bernoulli, and partial Fourier matrices satisfy the restricted isometry condition with number of measurements nearly linear in the sparsity level.

Subgaussian matrices: A random variable X is subgaussian if $\mathbb{P}(|X| > t) \leq Ce^{-ct^2}$ for all $t > 0$ and some positive constants C, c. Thus subgaussian random variables have tail distributions that are dominated by that of the standard Gaussian random variable. Choosing $C = c = 1$, we trivially have that standard Gaussian matrices (those whose entries are Gaussian) are subgaussian. Choosing $C = \frac{1}{e}$ and $c = 1$, we see that Bernoulli matrices (those whose entries are uniform ± 1) are also subgaussian. More generally, any bounded random variable is subgaussian. The following theorem proven in [48] shows that any subgaussian measurement matrix satisfies the restricted isometry condition with number of measurements m nearly linear in the sparsity s.

Theorem 2.1.1 (Subgaussian measurement matrices)**.** *Let* Φ *be a* $m \times d$ *sub-*

gaussian measurement matrix, and let $s \geq 1$, $0 < \delta < 1$, *and* $0 < \varepsilon < 0.5$. *Then with probability* $1-\alpha$ *the matrix* $\frac{1}{\sqrt{m}}\Phi$ *satisfies the restricted isometry condition with parameters* (s, ε) *provided that the number of measurements* m *satisfies*

$$m \geq \frac{Cs}{\varepsilon^2} \log\big(\frac{d}{\varepsilon^2 s}\big),$$

where C *depends only on* α *and other constants in the definition of subgaussian (for details on the dependence, see [48]).*

Partial bounded orthogonal matrices: Let Ψ be an orthogonal $d \times d$ matrix whose entries are bounded by $C/\sqrt{d}$ for some constant C. A $m \times d$ partial bounded orthogonal matrix is a matrix Φ formed by choosing m rows of such a matrix Ψ uniformly at random. Since the $d \times d$ discrete Fourier transform matrix is orthogonal with entries bounded by $1/\sqrt{d}$, the $m \times d$ random partial Fourier matrix is a partial bounded orthogonal matrix. The following theorem proved in [59] shows that such matrices satisfy the restricted isometry condition with number of measurements m nearly linear in the sparsity s.

Theorem 2.1.2 (Partial bounded orthogonal measurement matrices)**.** *Let* Φ *be a* $m \times d$ *partial bounded orthogonal measurement matrix, and let* $s \geq 1$, $0 < \delta < 1$, *and* $0 < \varepsilon < 0.5$. *Then with probability* $1 - \alpha$ *the matrix* $\frac{d}{\sqrt{m}}\Phi$ *satisfies the restricted isometry condition with parameters* (s, ε) *provided that the number of measurements* m *satisfies*

$$m \geq C\big(\frac{s \log d}{\varepsilon^2}\big) \log\big(\frac{s \log d}{\varepsilon^2}\big) \log^2 d,$$

where C *depends only on the confidence level* α *and other constants in the*

definition of partial bounded orthogonal matrix (for details on the dependence, see [59]).

2.1.3 Main Theorems

Candès and Tao showed in [6] that for measurement matrices that satisfy the restricted isometry condition, Basis Pursuit recovers all sparse signals exactly. This is summarized in the following theorem.

Theorem 2.1.3 (Sparse recovery under RIC [6])**.** *Assume that the measurement matrix* Φ *satisfies the restricted isometry condition with parameters* $(3s, 0.2)$*. Then every* s*-sparse vector* x *can be exactly recovered from its measurements* $u = \Phi x$ *as a unique solution to the linear optimization problem* (2.1.1)*.*

Note that these guarantees are *uniform*. Once the measurement matrix Φ satisfies the restricted isometry condition, Basis Pursuit correctly recovers *all* sparse vectors.

As discussed earlier, exactly sparse vectors are not encountered in practice, but rather nearly sparse signals. The signals and measurements are also noisy in practice, so practitioners seek algorithms that perform well under these conditions. Candès, Romberg and Tao showed in [5] that a version of Basis Pursuit indeed approximately recovers signals contaminated with noise. It is clear that in the noisy case, (2.1.1) is not a suitable method since the exact equality in the measurements would be most likely unattainable. Thus the method is modified slightly to allow for small perturbations, searching over all signals consistent with the measurement data. Instead of (2.1.1), we consider the formulation

$$\min \|y\|_1 \qquad \text{subject to} \qquad \|\Phi y - u\|_2 \leq \varepsilon. \tag{2.1.5}$$

Candès, Romberg and Tao showed that the program (2.1.5) reconstructs the signal with error at most proportional to the noise level. First we consider exactly sparse signals whose measurements are corrupted with noise. In this case, we have the following results from [5].

Theorem 2.1.4 (Stability of BP [5])**.** *Let Φ be a measurement matrix satisfying the restricted isometry condition with parameters $(3s, 0.2)$. Then for any s-sparse signal x and corrupted measurements $u = \Phi x + e$ with $\|e\|_2 \leq \varepsilon$, the solution $\hat{x}$ to* (2.1.5) *satisfies*

$$\|\hat{x} - x\|_2 \leq C_s \cdot \varepsilon,$$

where C_s depends only on the RIC constant δ.

Note that in the noiseless case, this theorem is consistent with Theorem 2.1.3. Theorem 2.1.4 is quite surprising given the fact that the matrix Φ is a wide rectangular matrix. Since it has far more columns than rows, most of the singular values of Φ are zero. Thus this theorem states that even though the problem is very ill-posed, Basis Pursuit still controls the error. It is important to point out that Theorem 2.1.4 is fundamentally optimal. This means that the error level ε is in a strong sense *unrecoverable*. Indeed, suppose that the support S of x was known a priori. The best way to reconstruct x from the measurements $u = \Phi x + e$ in this case would be to apply the pseudoinverse $\Phi_S^\dagger \stackrel{\text{def}}{=} (\Phi_S^* \Phi_S)^{-1} \Phi_S^*$ on the support, and set the remaining coordinates to zero. That is, one would reconstruct x as

$$\hat{x} = \begin{cases} \Phi_S^\dagger u & \text{on } S \\ 0 & \text{elsewhere} \end{cases}$$

Since the singular values of Φ_S are controlled, the error on the support is approxi-

mately $\|e\|_2 \leq \varepsilon$, and the error off the support is of course zero. This is also the error guaranteed by Theorem 2.1.4. Thus no recovery algorithm can hope to recover with less error than the original error introduced to the measurements.

Thus Basis Pursuit is stable to perturbations in the measurements of exactly sparse vectors. This extends naturally to the approximate recovery of nearly sparse signals, which is summarized in the companion theorem from [5].

Theorem 2.1.5 (Stability of BP II [5])**.** *Let Φ be a measurement matrix satisfying the restricted isometry condition with parameters $(3s, 0.2)$. Then for any arbitrary signal and corrupted measurements $u = \Phi x + e$ with $\|e\|_2 \leq \varepsilon$, the solution $\hat{x}$ to* (2.1.5) *satisfies*

$$\|\hat{x} - x\|_2 \leq C_s \cdot \varepsilon + C'_s \cdot \frac{\|x - x_s\|_1}{\sqrt{s}},$$

where x_s denotes the vector of the largest coefficients in magnitude of x.

Remark 2.1.6. *In [8], Candès sharpened Theorems 2.1.3, 2.1.4, and 2.1.5 to work under the restricted isometry condition with parameters $(2s, \sqrt{2} - 1)$.*

Theorem 2.1.5 says that for an arbitrary signal x, Basis Pursuit approximately recovers its largest s coefficients. In the particularly useful case of compressible signals, we have that for signals x obeying (1.1.1), the reconstruction satisfies

$$\|\hat{x} - x\|_2 \leq C_s \cdot \varepsilon + C' s^{-q+1/2}, \tag{2.1.6}$$

where C' depends on the RIC constant and the constant R in the compressibility definition eqrefcomp. We notice that for such signals we also have

$$\|x_s - x\|_2 \leq C_R s^{-q+1/2},$$

where C_R depends on R. Thus the error in the approximation guaranteed by Theorem 2.1.5 is comparable to the error obtained by simply selecting the s largest coefficients in magnitude of a compressible signal. So at least in the case of compressible signals, the error guarantees are again optimal. See Section 2.1.6 for a discussion of advantages and disadvantages to this approach.

2.1.4 Numerical Results

Many empirical studies have been conducted using Basis Pursuit. Several are included here, other results can be found in [5, 22, 6]. The studies discussed here were performed in Matlab, with the help of ℓ_1-Magic code by Romberg [45]. The code is given in Appendix A.1. In all cases here, the measurement matrix is a Gaussian matrix and the ambient dimension d is 256. In the first study, for each trial we generated binary signals with support uniformly selected at random as well as an independent Gaussian matrix for many values of the sparsity s and number of measurements m. Then we ran Basis Pursuit on the measurements of that signal and counted the number of times the signal was recovered correctly out of 500 trials. The results are displayed in Figure 2.1.2. The 99% recovery trend is depicted in Figure 2.1.3. This curve shows the relationship between the number of measurements m and the sparsity level s to guarantee that correct recovery occurs 99% of the time. Note that by *recovery*, we mean that the estimation error falls below the threshold of 10^{-5}. Figure 2.1.4 depicts the recovery error of Basis Pursuit when the measurements were perturbed. For this simulation, the signals were again binary (flat) signals, but Gaussian noise was added to the measurements. The norm of the noise was chosen to be the constant 1/2. The last figure, Figure 2.1.5 displays the recovery error when Basis Pursuit is run on compressible signals. For this study, the Basis Pursuit was run

on signals whose coefficients obeyed the power law (1.1.1). This simulation was run with sparsity $s = 12$, dimension $d = 256$, and for various values of the compressibility constant q. Note that the smaller q is, the more quickly the coefficients decay.

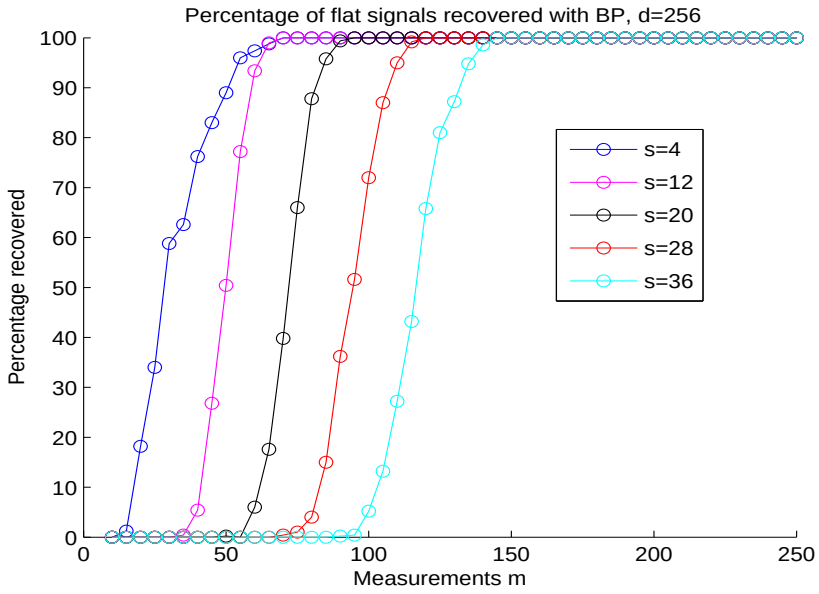


Figure 2.1.2: The percentage of sparse flat signals exactly recovered by Basis Pursuit as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

2.1.5 Linear Programming

Linear programming is a technique for optimization of a linear objective function under equality and inequality constraints, all of which are linear [15]. The problem (2.1.1) can be recast as a linear program whose objective function (to be minimized) is

$$\sum_{i=1}^{d} t_i,$$

with constraints

$$-t_i \leq z_i \leq t_i, \quad \Phi z = u.$$

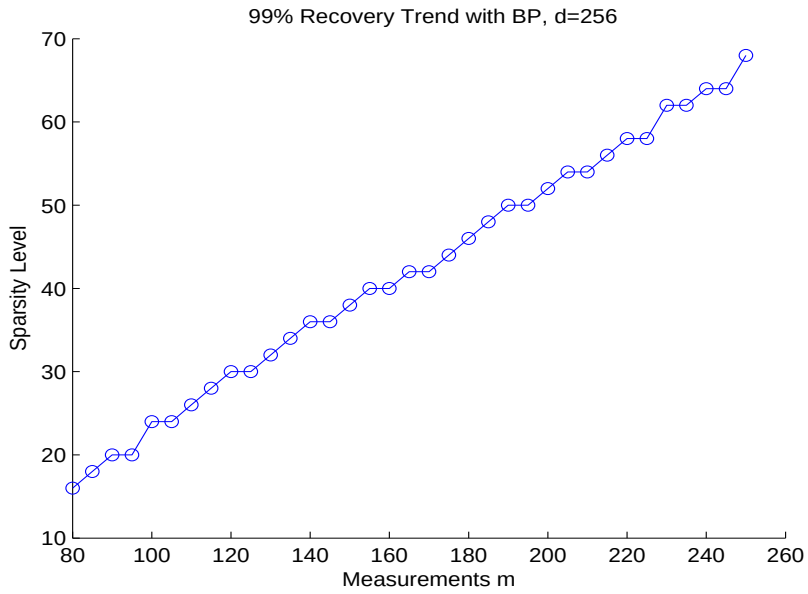


Figure 2.1.3: The 99% recovery trend of Basis Pursuit as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

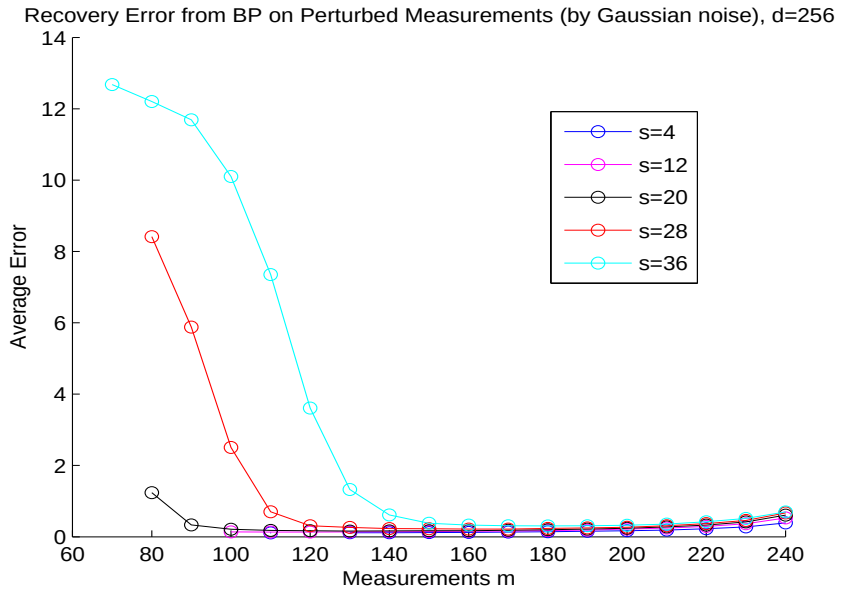


Figure 2.1.4: The recovery error of Basis Pursuit under perturbed measurements as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

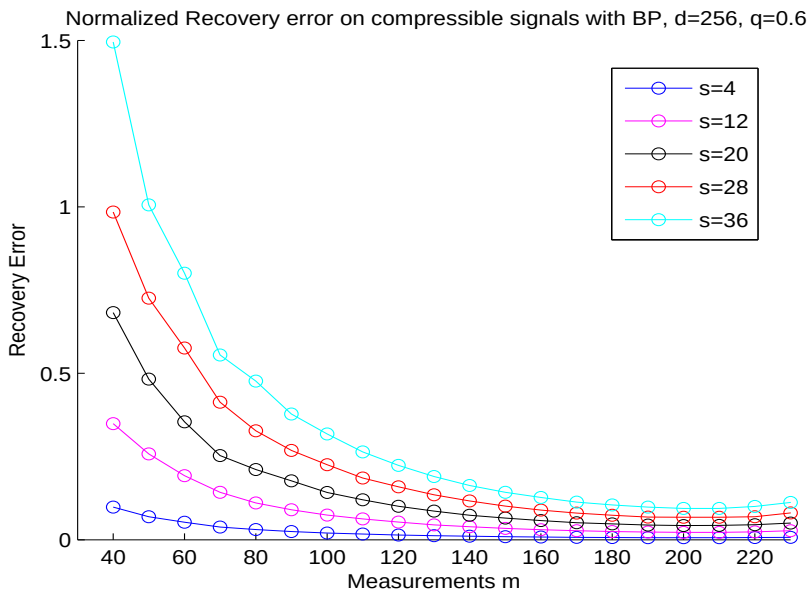


Figure 2.1.5: The normalized recovery error $\frac{\|x-\hat{x}\|_2}{\sqrt{s}\|x-x_s\|_1}$ of Basis Pursuit for compressible signals as a function of the number of measurements m in dimension $d = 256$ with compressibility $q = 0.6$ for various levels of sparsity s.

Viewed geometrically, the set of linear constraints, making up the feasible region, forms a convex polyhedron. By the Karush-Kuhn-Tucker conditions [44], all local optima are also global optima. If an optimum exists, it will be attained at a vertex of the polyhedron. There are several methods to search for this optimal vertex.

One of the most popular algorithms in linear programming is the simplex algorithm, developed by George Dantzig [15]. The simplex method begins with some admissible starting solution. If such a point is not known, a different linear program (with an obvious admissible solution) can be solved via the simplex method to find such a point. The simplex method then traverses the edges of the polytope via a sequence of pivot steps. The algorithm moves along the polytope, at each step choosing the optimal direction, until the optimum is found. Assuming that precautions against cycling are taken, the algorithm is guaranteed to find the optimum. Although

it's worst-case behavior is exponential in the problem size, it is much more efficient in practice. Smoothed analysis has explained this efficiency in practice [65], but it is still unknown whether there is a strongly polynomial bound on the runtime.

The simplex algorithm traverses the polytope's edges, but an alternative method called the interior point method traverses the interior of the polytope [56]. One such method was proposed by Karmarkar and is an interior point projective method. Recently, barrier function or path-following methods are being used for practical purposes. The best bound currently attained on the runtime of an interior point method is $O(m^2 d^{1.5})$. Other methods have been proposed, including the ellipsoid method by Khachiyan which has a polynomial worst case runtime, but as of yet none have provided strongly polynomial bounds.

2.1.6 Summary

Basis Pursuit presents many advantages over other algorithms in compressed sensing. Once a measurement matrix satisfies the restricted isometry condition, Basis Pursuit reconstructs *all* sparse signals. The guarantees it provides are thus uniform, meaning the algorithm will not fail for any sparse signal. Theorem 2.1.5 shows that Basis Pursuit is also stable, which is a necessity in practice. Its ability to handle noise and non-exactness of sparse signals makes the algorithm applicable to real world problems. The requirements on the restricted isometry constant shown in Theorem 2.1.5 along with the known results about random matrices discussed in Section 2.1.2 mean that Basis Pursuit only requires $O(s \log d)$ measurements to reconstruct d-dimensional s-sparse signals. It is thought by many that this is the optimal number of measurements.

Although Basis Pursuit provides these strong guarantees, its disadvantage is of

course speed. It relies on Linear Programming which although is often quite efficient in practice, has a polynomial runtime. For this reason, much work in compressed sensing has been done using faster methods. This approach uses greedy algorithms, which are discussed next.

2.2 Greedy Methods

An alternative approach to compressed sensing is the use of greedy algorithms. Greedy algorithms compute the support of the sparse signal x iteratively. Once the support of the signal is compute correctly, the pseudo-inverse of the measurement matrix restricted to the corresponding columns can be used to reconstruct the actual signal x. The clear advantage to this approach is speed, but the approach also presents new challenges.

2.2.1 Orthogonal Matching Pursuit

One such greedy algorithm is Orthogonal Matching Pursuit (OMP), put forth by Mallat and his collaborators (see e.g. [47]) and analyzed by Gilbert and Tropp [62]. OMP uses subGaussian measurement matrices to reconstruct sparse signals. If Φ is such a measurement matrix, then $\Phi^*\Phi$ is in a loose sense close to the identity. Therefore one would expect the largest coordinate of the observation vector $y = \Phi^*\Phi x$ to correspond to a non-zero entry of x. Thus one coordinate for the support of the signal x is estimated. Subtracting off that contribution from the observation vector y and repeating eventually yields the entire support of the signal x. OMP is quite fast, both in theory and in practice, but its guarantees are not as strong as those of Basis Pursuit.

Description

The OMP algorithm can thus be described as follows:

ORTHOGONAL MATCHING PURSUIT (OMP)

INPUT: Measurement matrix Φ, measurement vector $u = \Phi x$, sparsity level s

OUTPUT: Index set $I \subset \{1, \ldots, d\}$

PROCEDURE:

Initialize Let the index set $I = \emptyset$ and the residual $r = u$.

Repeat the following s times:

Identify Select the largest coordinate λ of $y = \Phi^* r$ in absolute value. Break ties lexicographically.

Update Add the coordinate λ to the index set: $I \leftarrow I \cup \{\lambda\}$, and update the residual:

$$\hat{x} = \underset{z}{\operatorname{argmin}} \|u - \Phi|_I z\|_2; \qquad r = u - \Phi\hat{x}.$$

Once the support I of the signal x is found, the estimate can be reconstructed as $\hat{x} = \Phi_I^\dagger u$, where recall we define the pseudoinverse by $\Phi_I^\dagger \stackrel{\text{def}}{=} (\Phi_I^* \Phi_I)^{-1} \Phi_I^*$.

The algorithm's simplicity enables a fast runtime. The algorithm iterates s times, and each iteration does a selection through d elements, multiplies by Φ^*, and solves a least squares problem. The selection can easily be done in $O(d)$ time, and the multiplication of Φ^* in the general case takes $O(md)$. When Φ is an unstructured matrix, the cost of solving the least squares problem is $O(s^2 d)$. However, maintaining a QR-Factorization of $\Phi|_I$ and using the modified Gram-Schmidt algorithm reduces this time to $O(|I|d)$ at each iteration. Using this method, the overall cost of OMP becomes $O(smd)$. In the case where the measurement matrix Φ is structured with a

fast-multiply, this can clearly be improved.

Main Theorems and Results

Gilbert and Tropp showed that OMP correctly recovers a *fixed* sparse signal with high probability. Indeed, in [62] they prove the following.

Theorem 2.2.1 (OMP Signal Recovery [62])**.** *Fix $\delta \in (0, 0.36)$ and let Φ be an $m \times d$ Gaussian measurement matrix with $m \geq Cm \log(d/\delta)$. Let x be an s-sparse signal in $\mathbb{R}^d$. Then with probability exceeding $1 - 2\delta$, OMP correctly reconstructs the signal x from its measurements Φx.*

Similar results hold when Φ is a subgaussian matrix. We note here that although the measurement requirements are similar to those of Basis Pursuit, the guarantees are not uniform. The probability is for a fixed signal rather than for all signals. The type of measurement matrix here is also more restrictive, and it is unknown whether OMP works for the important case of random Fourier matrices.

Numerical Experiments

Many empirical studies have been conducted to study the success of OMP. One study is described here that demonstrates the relationship between the sparsity level s and the number of measurements m. Other results can be found in [62]. The study discussed here was performed in Matlab, and is given in Appendix A.2.. In the study, for each trial I generated binary signals with support uniformly selected at random as well as an independent Gaussian measurement matrix, for many values of the sparsity s and number of measurements m. Then I ran OMP on the measurements of that signal and counted the number of times the signal was recovered correctly out of 500 trials. The results are displayed in Figure 2.2.1. The 99% recovery trend is

depicted in Figure 2.2.2. This curve shows the relationship between the number of measurements m and the sparsity level s to guarantee that correct recovery occurs 99% of the time. In comparison with Figures 2.1.2 and 2.1.3 we see that Basis Pursuit appears to provide stronger results empirically as well.

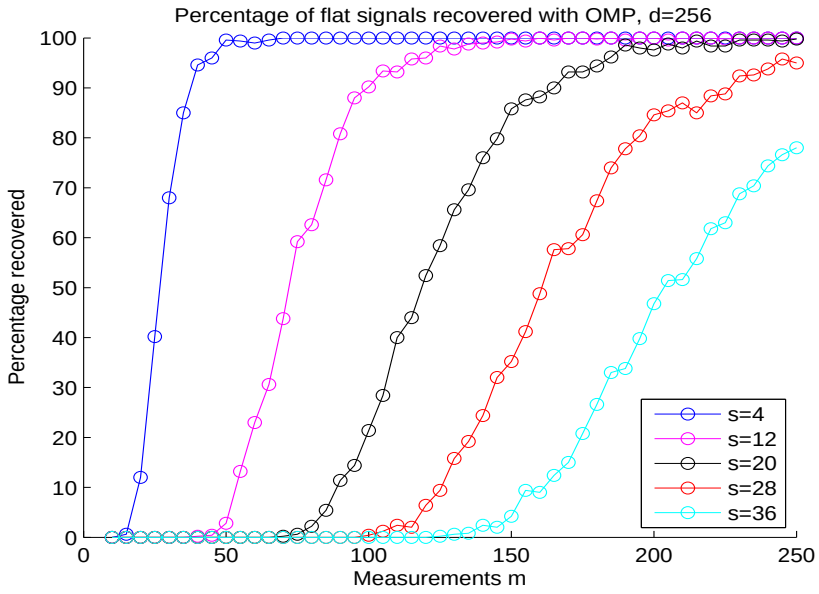


Figure 2.2.1: The percentage of sparse flat signals exactly recovered by OMP as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

Summary

It is important to note the distinctions between this theorem for OMP and Theorem 2.1.3 for Basis Pursuit. The first important difference is that Theorem 2.2.1 shows that OMP works only for the case when Φ is a Gaussian (or subgaussian) matrices, whereas Theorem 2.1.3 holds for a more general class of matrices (those which satisfy the RIC). Also, Theorem 2.1.3 demonstrates that Basis Pursuit works correctly for *all* signals, once the measurement matrix satisfies the restricted isometry

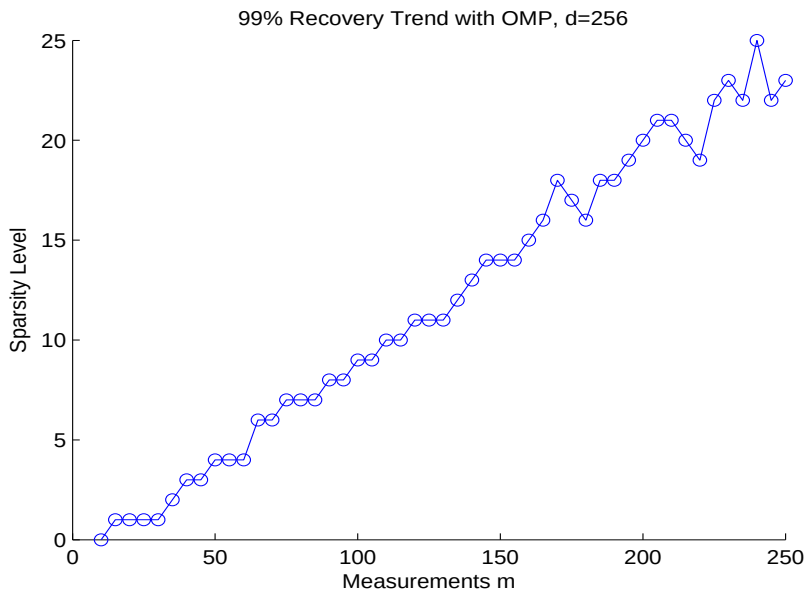


Figure 2.2.2: The 99% recovery trend of OMP as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

condition. Theorem 2.2.1 shows only that OMP works with high probability for each fixed signal. The advantage to OMP however, is that its runtime has a much faster bound than that of Basis Pursuit and Linear Programming.

2.2.2 Stagewise Orthogonal Matching Pursuit

An alternative greedy approach, Stagewise Orthogonal Matching Pursuit (StOMP) developed and analyzed by Donoho and his collaborators [23], uses ideas inspired by wireless communications. As in OMP, StOMP utilizes the observation vector $y = \Phi^* u$ where $u = \Phi x$ is the measurement vector. However, instead of simply selecting the largest component of the vector y, it selects all of the coordinates whose values are above a specified threshold. It then solves a least-squares problem to update the residual. The algorithm iterates through only a fixed number of stages and then terminates, whereas OMP requires s iterations where s is the sparsity level.

Description

The pseudo-code for StOMP can thus be described by the following.

STAGEWISE ORTHOGONAL MATCHING PURSUIT (STOMP)

INPUT: Measurement matrix Φ, measurement vector $u = \Phi x$,

OUTPUT: Estimate $\hat{x}$ to the signal x

PROCEDURE:

Initialize Let the index set $I = \emptyset$, the estimate $\hat{x} = 0$, and the residual $r = u$.

Repeat the following until stopping condition holds:

Identify Using the observation vector $y = \Phi^* r$, set

$$J = \{j : |y_j| > t_k \sigma_k\},$$

where σ_k is a formal noise level and t_k is a threshold parameter for iteration k.

Update Add the set J to the index set: $I \leftarrow I \cup J$, and update the residual and estimate:

$$\hat{x}|_I = (\Phi_I^* \Phi_I)^{-1} \Phi_I^* u, \quad r = u - \Phi \hat{x}.$$

The thresholding strategy is designed so that many terms enter at each stage, and so that algorithm halts after a fixed number of iterations. The formal noise level σ_k is proportional the Euclidean norm of the residual at that iteration. See [23] for more information on the thresholding strategy.

Main Results

Donoho and his collaborators studied StOMP empirically and have heuristically derived results. Figure 6 of [23] shows results of StOMP when the thresholding strategy is to control the false alarm rate and the measurement matrix Φ is sampled from the uniform spherical ensemble. The false alarm rate is the number of incorrectly selected coordinates (ie. those that are not in the actual support, but are chosen in the estimate) divided by the number of coordinates not in the support of the signal x. The figure shows that for very sparse signals, the algorithm recovers a good approximation to the signal. For less sparse signals the algorithm does not. The red curve in this figure is the graph of a heuristically derived function which they call the Predicted Phase transition. The simulation results and the predicted transition coincide reasonably well. This thresholding method requires knowledge about the actual sparsity level s of the signal x. Figure 7 of [23] shows similar results for a thresholding strategy that instead tries to control the false discovery rate. The false discovery rate is the fraction of incorrectly selected coordinates within the estimated support. This method appears to provide slightly weaker results. It appears however, that StOMP outperforms OMP and Basis Pursuit in some cases.

Although the structure of StOMP is similar to that of OMP, because StOMP selects many coordinates at each state, the runtime is quite improved. Indeed, using iterative methods to solve the least-squares problem yields a runtime bound of $CNsd + O(d)$, where N is the fixed number of iterations run by StOMP, and C is a constant that depends only on the accuracy level of the least-squares problem.

Numerical Experiments

A thorough empirical study of StOMP is provided in [23]. An additional study on StOMP was conducted here using a thresholding strategy with constant threshold parameter. The noise level σ was proportional to the norm of the residual, as [23] suggests. StOMP was run with various sparsity levels and measurement numbers, with Gaussian measurement matrices for 500 trials. Figure 2.2.3 depicts the results, and Figure 2.2.4 depicts the 99% recovery trend. Next StOMP was run in this same way but noise was added to the measurements. Figure 2.2.5 displays the results of this study. Since the reconstructed signal is always sparse, it is not surprising that StOMP is able to overcome the noise level. Note that these empirical results are not optimal because of the basic thresholding strategy. See [23] for empirical results using an improved thresholding strategy.

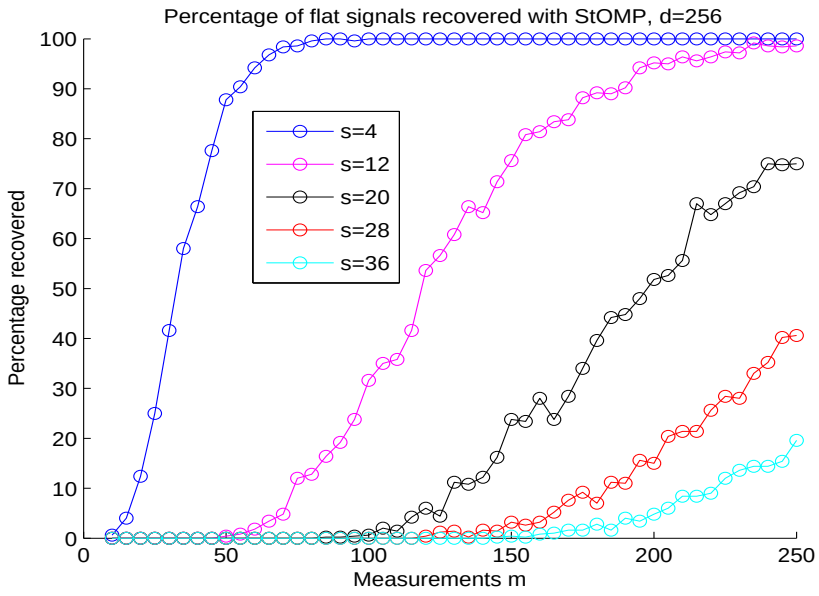


Figure 2.2.3: The percentage of sparse flat signals exactly recovered by StOMP as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

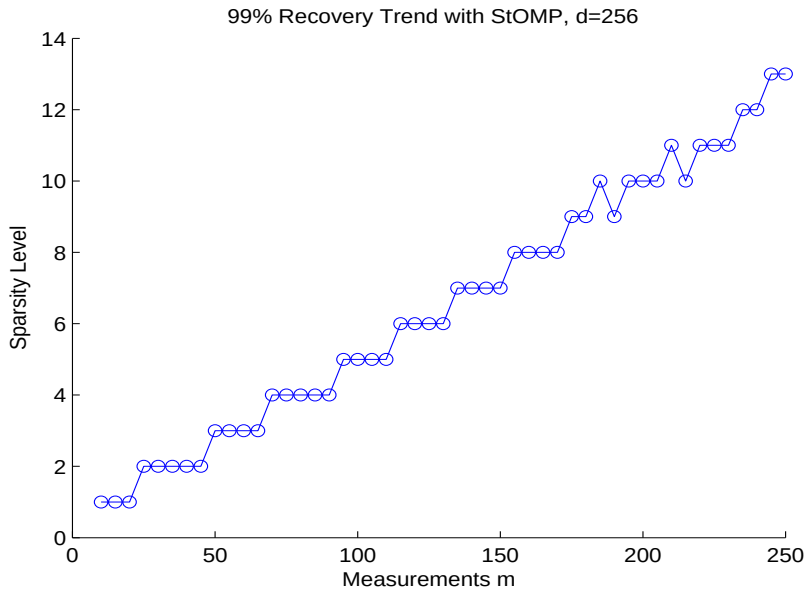


Figure 2.2.4: The 99% recovery trend of StOMP as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

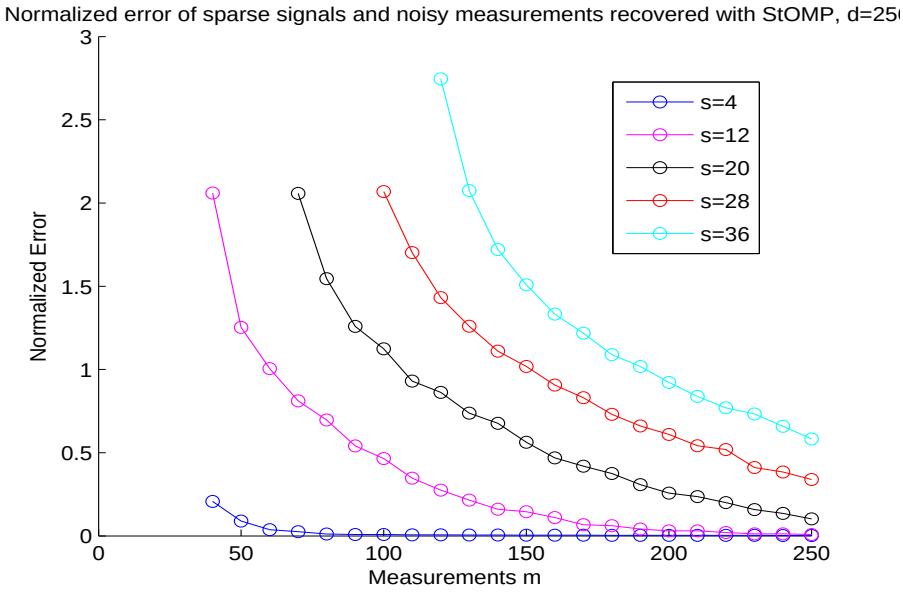


Figure 2.2.5: The normalized recovery error $\|x-\hat{x}\|_2/\|e\|_2$ of StOMP on sparse signals with noisy measurements $\Phi x + e$.

Summary

The empirical results of StOMP in [23] are quite promising, and suggest its improvement over OMP. However, in practice, the thresholding strategy may be difficult and complicated to implement well. More importantly, there are no rigorous results for StOMP available. In the next subsection other greedy methods are discussed with rigorous results, but that require highly structured measurement matrices.

2.2.3 Combinatorial Methods

The major benefit of the greedy approach is its speed, both empirically and theoretically. There is a group of combinatorial algorithms that provide even faster speed, but that impose very strict requirements on the measurement matrix. These methods use highly structured measurement matrices that support very fast reconstruction through group testing. The work in this area includes HHS pursuit [32], chaining pursuit [31], Sudocodes [60], Fourier sampling [33, 35] and some others by Cormode–Muthukrishnan [12] and Iwen [40].

Descriptions and Results

Many of the sublinear algorithms such as HHS pursuit, chaining pursuit and Sudocodes employ the idea of group testing. Group testing is a method which originated in the Selective Service during World War II to test soldiers for Syphilis [24], and now it appears in many experimental designs and other algorithms. During this time, the Wassermann test [66] was used to detect the Syphilis antigen in a blood sample. Since this test was expensive, the method was to sample a group of men together and test the entire pool of blood samples. If the pool did not contain the antigen, then one test replaced many. If it was found, then the process could either

be repeated with that group, or each individual in the group could then be tested.

These sublinear algorithms in compressed sensing use this same idea to test for elements of the support of the signal x. Chaining pursuit, for example, uses a measurement matrix consisting of a row tensor product of a bit test matrix and an isolation matrix, both of which are 0-1 matrices. Chaining pursuit first uses bit tests to locate the positions of the large components of the signal x and estimate those values. Then the algorithm retains a portion of the coordinates that are largest magnitude and repeats. In the end, those coordinates which appeared throughout a large portion of the iterations are kept, and the signal is estimated using these. Pseudo-code is available in [31], where the following result is proved.

Theorem 2.2.2 (Chaining pursuit [31])**.** *With probability at least* $1 - O(d^{-3})$*, the* $O(s \log^2 d) \times d$ *random measurement operator* Φ *has the following property. For* $x \in \mathbb{R}^d$ *and its measurements* $u = \Phi x$*, the Chaining Pursuit algorithm produces a signal* $\hat{x}$ *with at most* s *nonzero entries. The output* $\hat{x}$ *satisfies*

$$\|x - \hat{x}\|_1 \leq C(1 + \log s)\|x - x_s\|_1.$$

The time cost of the algorithm is $O(s \log^2 s \log^2 d)$*.*

HHS Pursuit, a similar algorithm but with improved guarantees, uses a measurement matrix that consists again of two parts. The first part is an identification matrix, and the second is an estimation matrix. As the names suggest, the identification matrix is used to identify the location of the large components of the signal, whereas the estimation matrix is used to estimate the values at those locations. Each of these matrices consist of smaller parts, some deterministic and some random. Using this measurement matrix to locate large components and estimate their values,

HHS Pursuit then adds the new estimate to the previous, and prunes it relative to the sparsity level. This estimation is itself then sampled, and the residual of the signal is updated. See [32] for the pseudo-code of the algorithm. Although the measurement matrix is highly structured, again a disadvantage in practice, the results for the algorithm are quite strong. Indeed, in [32] the following result is proved.

Theorem 2.2.3 (HHS Pursuit [32])**.** *Fix an integer s and a number $\varepsilon \in (0,1)$. With probability at least 0.99, the random measurement matrix Φ (as described above) has the following property. Let $x \in \mathbb{R}^d$ and let $u = \Phi x$ be the measurement vector. The HHS Pursuit algorithm produces a signal approximation $\hat{x}$ with $O(s/\varepsilon^2)$ nonzero entries. The approximation satisfies*

$$\|x - \hat{x}\|_2 \leq \frac{\varepsilon}{\sqrt{s}} \|x - x_s\|_1,$$

where again x_s denotes the vector consisting of the s largest entries in magnitude of x. The number of measurements m is proportional to (s/ε^2) polylog(d/ε), and HHS Pursuit runs in time (s^2/ε^4)polylog(d/ε). The algorithm uses working space (s/ε^2)polylog(d/ε), including storage of the matrix Φ.

Remark 2.2.4. *This theorem presents guarantees that are stronger than those of chaining pursuit. Chaining pursuit, however, still provides a faster runtime.*

There are other algorithms such as the Sudocodes algorithm that as of now only work in the noiseless, strictly sparse case. However, these are still interesting because of the simplicity of the algorithm. The Sudocodes algorithm is a simple two-phase algorithm. In the first phase, an easily implemented avalanche bit testing scheme is applied iteratively to recover most of the coordinates of the signal x. At this point, it remains to reconstruct an extremely low dimensional signal (one whose

coordinates are only those that remain). In the second phase, this part of the signal is reconstructed, which completes the reconstruction. Since the recovery is two-phase, the measurement matrix is as well. For the first phase, it must contain a sparse submatrix, one consisting of many zeros and few ones in each row. For the second phase, it also contains a matrix whose small submatrices are invertible. The following result for strictly sparse signals is proved in [60].

Theorem 2.2.5 (Sudocodes [60])**.** *Let x be an s-sparse signal in $\mathbb{R}^d$, and let the $m \times d$ measurement matrix Φ be as described above. Then with $m = O(s \log d)$, the Sudocodes algorithm exactly reconstructs the signal x with computational complexity just $O(s \log s \log d)$.*

The Sudocodes algorithm cannot reconstruct noisy signals because of the lack of robustness in the second phase. However, work on modifying this phase to handle noise is currently being done. If this task is accomplished Sudocodes would be an attractive algorithm because of its sublinear runtime and simple implementation.

Summary

Combinatorial algorithms such as HHS pursuit provide sublinear time recovery with optimal error bounds and optimal number of measurements. Some of these are straightforward and easy to implement, and others require complicated structures. The major disadvantage however is the structural requirement on the measurement matrices. Not only do these methods only work with one particular kind of measurement matrix, but that matrix is highly structured which limits its use in practice. There are no known sublinear methods in compressed sensing that allow for unstructured or generic measurement matrices.

Chapter 3

Contributions

3.1 Regularized Orthogonal Matching Pursuit

As is now evident, the two approaches to compressed sensing each presented disjoint advantages and challenges. While the optimization method provides robustness and uniform guarantees, it lacks the speed of the greedy approach. The greedy methods on the other hand had not been able to provide the strong guarantees of Basis Pursuit. This changed when we developed a new greedy algorithm, Regularized Orthogonal Matching Pursuit [55], that provided the strong guarantees of the optimization method. This work bridged the gap between the two approaches, and provided the first algorithm possessing the advantages of both approaches.

3.1.1 Description

Regularized Orthogonal Matching Pursuit (ROMP) is a greedy algorithm, but will correctly recover any sparse signal using any measurement matrix that satisfies the Restricted Isometry Condition (2.1.3). Again as in the case of OMP, we will use

the observation vector $\Phi^*\Phi x$ as a good local approximation to the s-sparse signal x. Since the Restricted Isometry Condition guarantees that every s columns of Φ are close to an orthonormal system, we will choose at each iteration not just one coordinate as in OMP, but up to s coordinates using the observation vector. It will then be okay to choose some incorrect coordinates, so long as the number of those is limited. To ensure that we do not select too many incorrect coordinates at each iteration, we include a regularization step which will guarantee that each coordinate selected contains an even share of the information about the signal. The ROMP algorithm can thus be described as follows:

Regularized Orthogonal Matching Pursuit (ROMP) [55]

Input: Measurement matrix Φ, measurement vector $u = \Phi x$, sparsity level s

Output: Index set $I \subset \{1, \dots, d\}$, reconstructed vector $\hat{x} = w$

Procedure:

Initialize Let the index set $I = \emptyset$ and the residual $r = u$.

Repeat the following steps until $r = 0$:

Identify Choose a set J of the s biggest coordinates in magnitude of the observation vector $y = \Phi^* r$, or all of its nonzero coordinates, whichever set is smaller.

Regularize Among all subsets $J_0 \subset J$ with comparable coordinates:

$$|y(i)| \leq 2|y(j)| \quad \text{for all } i, j \in J_0,$$

choose J_0 with the maximal energy $\|y|_{J_0}\|_2$.

Update Add the set J_0 to the index set: $I \leftarrow I \cup J_0$, and update the residual:

$$w = \operatorname*{argmin}_{z \in \mathbb{R}^I} \|u - \Phi z\|_2; \qquad r = u - \Phi w.$$

Remarks.

1. We remark here that knowledge about the sparsity level s is required in ROMP, as in OMP. There are several ways this information may be obtained. Since the number of measurements m is usually chosen to be $O(s \log d)$, one may then estimate the sparsity level s to be roughly $m/\log d$. An alternative approach would be to run ROMP using various sparsity levels and choose the one which yields the

least error $\|\Phi\hat{x} - \Phi x\|$ for outputs $\hat{x}$. Choosing testing levels out of a geometric progression, for example, would not contribute significantly to the overall runtime.

2. Clearly in the case where the signal is not exactly sparse and the signal and measurements are corrupted with noise, the algorithm as described above will never halt. Thus in the noisy case, we simply change the halting criteria by allowing the algorithm iterate at most s times, or until $|I| \geq s$. We show below that with this modification ROMP approximately reconstructs arbitrary signals.

3.1.2 Main Theorems

In this section we present the main theorems for ROMP. We prove these theorems in Section 3.1.3. When the measurement matrix Φ satisfied the Restricted Isometry Condition, ROMP exactly recovers all sparse signals. This is summarized in the following theorem from [55].

Theorem 3.1.1 (Exact sparse recovery via ROMP [55])**.** *Assume a measurement matrix Φ satisfies the Restricted Isometry Condition with parameters $(2s, \varepsilon)$ for $\varepsilon = 0.03/\sqrt{\log s}$. Let x be an s-sparse vector in $\mathbb{R}^d$ with measurements $u = \Phi x$. Then ROMP in at most s iterations outputs a set I such that*

$$\mathrm{supp}(x) \subset I \quad \textit{and} \quad |I| \leq 2s.$$

Remarks. 1. Theorem 3.1.1 shows that ROMP provides *exact recovery* of sparse signals. Using the index set I, one can compute the signal x from its measurements $u = \Phi x$ as $x = (\Phi_I)^{-1}u$, where Φ_I denotes the measurement matrix Φ restricted to the columns indexed by I.

2. Theorem 3.1.1 provides *uniform guarantees* of sparse recovery, meaning that

once the measurement matrix Φ satisfies the Restricted Isometry Condition, ROMP recovers *every* sparse signal from its measurements. Uniform guarantees such as this are now known to be impossible for OMP [57], and finding a version of OMP providing uniform guarantees was previously an open problem [62]. Theorem 3.1.1 shows that ROMP solves this problem.

3. Recall from Section 2.1.2 that random Gaussian, Bernoulli and partial Fourier matrices with number of measurements m almost linear in the sparsity s, satisfy the Restricted Isometry Condition. It is still unknown whether OMP works at all with partial Fourier measurements, but ROMP gives sparse recovery for these measurements, and with uniform guarantees.

4. In Section 3.1.4 we explain how the identification and regularization steps of ROMP can easily be performed efficiently. In Section 3.1.4 we show that the running time of ROMP is comparable to that of OMP in theory, and is better in practice.

Theorem 3.1.1 shows ROMP works correctly for signals which are exactly sparse. However, as mentioned before, ROMP also performs well for signals and measurements which are corrupted with noise. This is an essential property for an algorithm to be realistically used in practice. The following theorem from [54] shows that ROMP approximately reconstructs sparse signals with noisy measurements. Corollary 3.1.3 shows that ROMP also approximately reconstructs *arbitrary* signals with noisy measurements.

Theorem 3.1.2 (Stability of ROMP under measurement perturbations [54])**.** *Let Φ be a measurement matrix satisfying the Restricted Isometry Condition with parameters $(4s, \varepsilon)$ for $\varepsilon = 0.01/\sqrt{\log s}$. Let $x \in \mathbb{R}^d$ be an s-sparse vector. Suppose that the measurement vector Φx becomes corrupted, so that we consider $u = \Phi x + e$ where e*

is some error vector. Then ROMP produces a good approximation $\hat{x}$ to x:

$$\|x - \hat{x}\|_2 \leq 104\sqrt{\log s}\|e\|_2.$$

Corollary 3.1.3 (Stability of ROMP under signal perturbations [54])**.** *Let Φ be a measurement matrix satisfying the Restricted Isometry Condition with parameters $(8s, \varepsilon)$ for $\varepsilon = 0.01/\sqrt{\log s}$. Consider an arbitrary vector x in $\mathbb{R}^d$. Suppose that the measurement vector Φx becomes corrupted, so we consider $u = \Phi x + e$ where e is some error vector. Then ROMP produces a good approximation $\hat{x}$ to x_{2s}:*

$$\|\hat{x} - x_{2s}\|_2 \leq 159\sqrt{\log 2s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big). \tag{3.1.1}$$

Remarks.

1. In the noiseless case, Theorem 3.1.2 coincides with Theorem 3.1.1 in showing exact recovery.

2. Corollary 3.1.3 still holds (with only the constants changed) when the term x_{2s} is replaced by $x_{(1+\delta)s}$ for any $\delta > 0$. This is evident by the proof of the corollary given below.

2. Corollary 3.1.3 also implies the following bound on the entire signal x:

$$\|\hat{x} - x\|_2 \leq 160\sqrt{\log 2s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big). \tag{3.1.2}$$

Indeed, we have

$$\begin{aligned}\|\hat{x} - x\|_2 &\leq \|\hat{x} - x_{2s}\|_2 + \|x - x_{2s}\|_2 \\ &\leq 159\sqrt{\log 2s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big) + \|(x - x_s) - (x - x_s)_s\| \\ &\leq 159\sqrt{\log 2s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big) + \frac{\|x - x_s\|_1}{\sqrt{s}} \\ &\leq 160\sqrt{\log 2s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big),\end{aligned}$$

where the first inequality is the triangle inequality, the second uses Corollary 3.1.3 and the identity $x - x_{2s} = (x - x_s) - (x - x_s)_s$, and third uses Lemma 3.1.19 below.

3. The error bound for Basis Pursuit given in Theorem 2.1.5, is similar except for the logarithmic factor. We again believe this to be an artifact of our proof, and our empirical results in Section 3.1.5 show that ROMP indeed provides much better results than the corollary suggests.

4. In the case of noise with Basis Pursuit, the problem (2.1.5) needs to be solved, which requires knowledge about the noise vector e. ROMP requires no such knowledge.

5. If instead one wished to compute a $2s$-sparse approximation to the signal, one may just retain the $2s$ largest coordinates of the reconstructed vector $\hat{x}$. In this case Corollary 3.1.3 implies the following:

Corollary 3.1.4. *Assume a measurement matrix Φ satisfies the Restricted Isometry Condition with parameters $(8s, \varepsilon)$ for $\varepsilon = 0.01/\sqrt{\log s}$. Then for an arbitrary vector x in $\mathbb{R}^d$,*

$$\|x_{2s} - \hat{x}_{2s}\|_2 \leq 477\sqrt{\log 2s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big).$$

This corollary is proved in Section 3.1.3.

6. As noted earlier, a special class of signals are compressible signals (1.1.1), and for these it is straightforward to see that (3.1.2) gives us the following error bound for ROMP:

$$\|x - \hat{x}\|_2 \leq C'_q \frac{\sqrt{\log s}}{s^{q-1/2}} + C'' \sqrt{\log s} \|e\|_2.$$

As observed in [5], this bound is optimal (within the logarithmic factor), meaning no algorithm can perform fundamentally better.

3.1.3 Proofs of Theorems

In this section we include the proofs of Theorems 3.1.1 and 3.1.2 and Corollaries 3.1.3 and 3.1.4. The proofs presented here originally appeared in [55] and [54].

Proof of Theorem 3.1.1

We shall prove a stronger version of Theorem 3.1.1, which states that *at every iteration* of ROMP, at least 50% of the newly selected coordinates are from the support of the signal x.

Theorem 3.1.5 (Iteration Invariant of ROMP)**.** *Assume Φ satisfies the Restricted Isometry Condition with parameters $(2s, \varepsilon)$ for $\varepsilon = 0.03/\sqrt{\log s}$. Let $x \neq 0$ be a s-sparse vector with measurements $u = \Phi x$. Then at any iteration of ROMP, after the regularization step, we have $J_0 \neq \emptyset$, $J_0 \cap I = \emptyset$ and*

$$|J_0 \cap \mathrm{supp}(x)| \geq \frac{1}{2}|J_0|. \tag{3.1.3}$$

In other words, at least 50% of the coordinates in the newly selected set J_0 belong to the support of x.

In particular, at every iteration ROMP finds at least one new coordinate in the support of the signal x. Coordinates outside the support can also be found, but (3.1.3) guarantees that the number of such "false" coordinates is always smaller than those in the support. This clearly implies Theorem 3.1.1.

Before proving Theorem 3.1.5 we explain how the Restricted Isometry Condition will be used in our argument. RIC is necessarily a local principle, which concerns not the measurement matrix Φ as a whole, but its submatrices of s columns. All such submatrices Φ_I, $I \subset \{1, \dots, d\}$, $|I| \leq s$ are almost isometries. Therefore, for every s-sparse signal x, the observation vector $y = \Phi^* \Phi x$ approximates x locally, when restricted to a set of cardinality s. The following proposition formalizes these local properties of Φ on which our argument is based.

Proposition 3.1.6 (Consequences of Restricted Isometry Condition)**.** *Assume a measurement matrix Φ satisfies the Restricted Isometry Condition with parameters $(2s, \varepsilon)$. Then the following holds.*

1. (Local approximation) *For every s-sparse vector $x \in \mathbb{R}^d$ and every set $I \subset \{1, \dots, d\}$, $|I| \leq s$, the observation vector $y = \Phi^* \Phi x$ satisfies*

$$\|y|_I - x|_I\|_2 \leq 2.03\varepsilon \|x\|_2.$$

2. (Spectral norm) *For any vector $z \in \mathbb{R}^m$ and every set $I \subset \{1, \dots, d\}$, $|I| \leq 2s$, we have*

$$\|(\Phi^* z)|_I\|_2 \leq (1 + \varepsilon)\|z\|_2.$$

3. (Almost orthogonality of columns) *Consider two disjoint sets $I, J \subset \{1, \dots, d\}$, $|I \cup J| \leq 2s$. Let P_I, P_J denote the orthogonal projections in $\mathbb{R}^m$ onto* range(Φ_I)

and range(Φ_J)*, respectively. Then*

$$\|P_I P_J\|_{2\to 2} \leq 2.2\varepsilon.$$

Proof. PART 1. Let $\Gamma = I \cup \operatorname{supp}(x)$, so that $|\Gamma| \leq 2s$. Let Id_Γ denote the identity operator on $\mathbb{R}^\Gamma$. By the Restricted Isometry Condition,

$$\|\Phi_\Gamma^* \Phi_\Gamma - \mathrm{Id}_\Gamma\|_{2\to 2} = \sup_{w\in\mathbb{R}^\Gamma,\, \|w\|_2=1} \left| \|\Phi_\Gamma w\|_2^2 - \|w\|_2^2 \right| \leq (1+\varepsilon)^2 - 1 \leq 2.03\varepsilon.$$

Since $\operatorname{supp}(x) \subset \Gamma$, we have

$$\|y|_\Gamma - x|_\Gamma\|_2 = \|\Phi_\Gamma^* \Phi_\Gamma x - \operatorname*{Id}_\Gamma x\|_2 \leq 2.03\varepsilon \|x\|_2.$$

The conclusion of Part 1 follows since $I \subset \Gamma$.

PART 2. Denote by Q_I the orthogonal projection in $\mathbb{R}^d$ onto $\mathbb{R}^I$. Since $|I| \leq 2s$, the Restricted Isometry Condition yields

$$\|Q_I \Phi^*\|_{2\to 2} = \|\Phi Q_I\|_{2\to 2} \leq 1 + \varepsilon.$$

This yields the inequality in Part 2.

PART 3. The desired inequality is equivalent to:

$$\frac{|\langle x, y\rangle|}{\|x\|_2 \|y\|_2} \leq 2.2\varepsilon \qquad \text{for all } x \in \text{range}(\Phi_I),\ y \in \text{range}(\Phi_J).$$

Let $K = I \cup J$ so that $|K| \leq 2s$. For any $x \in \text{range}(\Phi_I), y \in \text{range}(\Phi_J)$, there are

a, b so that

$$x = \Phi_K a,\ y = \Phi_K b, \qquad a \in \mathbb{R}^I,\ b \in \mathbb{R}^J.$$

By the Restricted Isometry Condition,

$$\|x\|_2 \geq (1-\varepsilon)\|a\|_2,\ \|y\|_2 \geq (1-\varepsilon)\|b\|_2.$$

By the proof of Part 2 above and since $\langle ab \rangle = 0$, we have

$$|\langle x, y \rangle| = |\langle (\Phi_K^* \Phi_K - \mathrm{Id}_\Gamma) a, b \rangle| \leq 2.03\varepsilon \|a\|_2 \|b\|_2.$$

This yields

$$\frac{|\langle x, y \rangle|}{\|x\|_2 \|y\|_2} \leq \frac{2.03\varepsilon}{(1-\varepsilon)^2} \leq 2.2\varepsilon,$$

which completes the proof. □

We are now ready to prove Theorem 3.1.5.

The proof is by induction on the iteration of ROMP. The induction claim is that for all previous iterations, the set of newly chosen indices J_0 is nonempty, disjoint from the set of previously chosen indices I, and (3.1.3) holds.

Let I be the set of previously chosen indices at the start of a given iteration. The induction claim easily implies that

$$|\mathrm{supp}(x) \cup I| \leq 2s. \tag{3.1.4}$$

Let J_0, J, be the sets found by ROMP in the current iteration. By the definition of the set J_0, it is nonempty.

Let $r \neq 0$ be the residual at the start of this iteration. We shall approximate r by a vector in range$(\Phi_{\text{supp}(x)\setminus I})$. That is, we want to approximately realize the residual r as measurements of some signal which lives on the still unfound coordinates of the the support of x. To that end, we consider the subspace

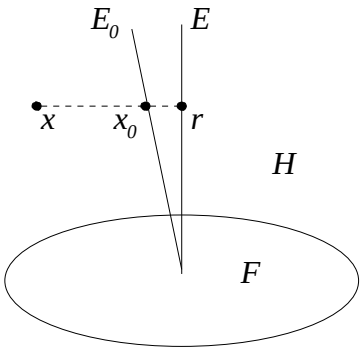
$$H := \text{range}(\Phi_{\text{supp}(x)\cup I})$$

and its complementary subspaces

$$F := \text{range}(\Phi_I), \quad E_0 := \text{range}(\Phi_{\text{supp}(x)\setminus I}).$$

The Restricted Isometry Condition in the form of Part 3 of Proposition 3.1.6 ensures that F and E_0 are almost orthogonal. Thus E_0 is close to the orthogonal complement of F in H,

$$E := F^{\perp} \cap H.$$



We will also consider the signal we seek to identify at the current iteration, its

measurements, and its observation vector:

$$x_0 := x|_{\mathrm{supp}(x)\setminus I}, \quad u_0 := \Phi x_0 \in E_0, \quad y_0 := \Phi^* u_0. \tag{3.1.5}$$

Lemma 3.1.9 will show that $\|(y-y_0)|_T\|_2$ for any small enough subset T is small, and Lemma 3.1.12 will show that $\|y|_{J_0}\|_2$ is not too small. First, we show that the residual r has a simple description:

Lemma 3.1.7 (Residual)**.** *Here and thereafter, let P_L denote the orthogonal projection in $\mathbb{R}^m$ onto a linear subspace L. Then*

$$r = P_E u.$$

Proof. By definition of the residual in the algorithm, $r = P_{F^\perp} u$. Since $u \in H$, we conclude from the orthogonal decomposition $H = F + E$ that $u = P_F u + P_E u$. Thus $r = u - P_F u = P_E u$. □

To guarantee a correct identification of x_0, we first state two approximation lemmas that reflect in two different ways the fact that subspaces E_0 and E are close to each other. This will allow us to carry over information from E_0 to E.

Lemma 3.1.8 (Approximation of the residual)**.** *We have*

$$\|u_0 - r\|_2 \le 2.2\varepsilon \|u_0\|_2.$$

Proof. By definition of F, we have $u - u_0 = \Phi(x - x_0) \in F$. Therefore, by Lemma 3.1.7, $r = P_E u = P_E u_0$, and so

$$u_0 - r = u_0 - P_E u_0 = P_F u_0 = P_F P_{E_0} u_0.$$

Now we use Part 3 of Proposition 3.1.6 for the sets I and $\mathrm{supp}(x) \setminus I$ whose union has cardinality at most $2s$ by (3.1.4). It follows that $\|P_F P_{E_0} u_0\|_2 \leq 2.2\varepsilon \|u_0\|_2$ as desired. □

Lemma 3.1.9 (Approximation of the observation)**.** *Consider the observation vectors $y_0 = \Phi^* u_0$ and $y = \Phi^* r$. Then for any set $T \subset \{1, \ldots, d\}$ with $|T| \leq 2s$, we have*

$$\|(y_0 - y)|_T\|_2 \leq 2.4\varepsilon \|x_0\|_2.$$

Proof. Since $u_0 = \Phi x_0$, we have by Lemma 3.1.8 and the Restricted Isometry Condition that

$$\|u_0 - r\|_2 \leq 2.2\varepsilon \|\Phi x_0\|_2 \leq 2.2\varepsilon(1+\varepsilon)\|x_0\|_2 \leq 2.3\varepsilon \|x_0\|_2.$$

To complete the proof, it remains to apply Part 2 of Proposition 3.1.6, which yields $\|(y_0 - y)|_T\|_2 \leq (1+\varepsilon)\|u_0 - r\|_2$. □

We next show that the energy (norm) of y when restricted to J, and furthermore to J_0, is not too small. By the approximation lemmas, this will yield that ROMP selects at least a fixed percentage of energy of the still unidentified part of the signal. By the regularization step of ROMP, since all selected coefficients have comparable magnitudes, we will conclude that not only a portion of energy but also of the *support* is selected correctly. This will be the desired conclusion.

Lemma 3.1.10 (Localizing the energy)**.** *We have $\|y|_J\|_2 \geq 0.8\|x_0\|_2$.*

Proof. Let $S = \mathrm{supp}(x) \setminus I$. Since $|S| \leq s$, the maximality property of J in the algorithm implies that

$$\|y_0|_J\|_2 \geq \|y_0|_S\|_2.$$

Furthermore, since $x_0|_S = x_0$, by Part 1 of Proposition 3.1.6 we have

$$\|y_0|_S\|_2 \geq (1 - 2.03\varepsilon)\|x_0\|_2.$$

Putting these two inequalities together and using Lemma 3.1.9, we conclude that

$$\|y|_J\|_2 \geq (1 - 2.03\varepsilon)\|x_0\|_2 - 2.4\varepsilon\|x_0\|_2 \geq 0.8\|x_0\|_2.$$

This proves the lemma. □

We next bound the norm of y restricted to the smaller set J_0. We do this by first noticing a general property of regularization:

Lemma 3.1.11 (Regularization)**.** *Let v be any vector in $\mathbb{R}^m$, $m > 1$. Then there exists a subset $A \subset \{1, \dots, m\}$ with comparable coordinates:*

$$|v(i)| \leq 2|v(j)| \quad \textit{for all } i, j \in A, \tag{3.1.6}$$

and with big energy:

$$\|v|_A\|_2 \geq \frac{1}{2.5\sqrt{\log m}}\|v\|_2. \tag{3.1.7}$$

Proof. We will construct at most $O(\log m)$ subsets A_k with comparable coordinates as in (3.1.6), and such that at least one of these sets will have large energy as in (3.1.7).

Let $v = (v_1, \dots, v_m)$, and consider a partition of $\{1, \dots, m\}$ using sets with comparable coordinates:

$$A_k := \{i : 2^{-k}\|v\|_2 < |v_i| \leq 2^{-k+1}\|v\|_2\}, \qquad k = 1, 2, \dots$$

Let $k_0 = \lceil \log m \rceil + 1$, so that $|v_i| \leq \frac{1}{m}\|v\|_2$ for all $i \in A_k$, $k > k_0$. Then the set $U = \bigcup_{k \leq k_0} A_k$ contains most of the energy of v:

$$\|v|_{U^c}\|_2 \leq \big(m(\frac{1}{m}\|v\|_2)^2\big)^{1/2} = \frac{1}{\sqrt{m}}\|y\|_2 \leq \frac{1}{\sqrt{2}}\|y\|_2.$$

Thus

$$\big(\sum_{k \leq k_0} \|v|_{A_k}\|_2^2\big)^{1/2} = \|v|_U\|_2 = \big(\|v\|_2^2 - \|v|_{U^c}\|_2^2\big)^{1/2} \geq \frac{1}{\sqrt{2}}\|v\|_2.$$

Therefore there exists $k \leq k_0$ such that

$$\|v|_{A_k}\|_2 \geq \frac{1}{\sqrt{2k_0}}\|v\|_2 \geq \frac{1}{2.5\sqrt{\log m}}\|v\|_2,$$

which completes the proof. □

In our context, Lemma 3.1.11 applied to the vector $y|_J$ along with Lemma 3.1.10 directly implies:

Lemma 3.1.12 (Regularizing the energy)**.** *We have*

$$\|y|_{J_0}\|_2 \geq \frac{0.32}{\sqrt{\log s}}\|x_0\|_2.$$

We now finish the proof of Theorem 3.1.5.

To show the first claim, that J_0 is nonempty, we note that $x_0 \neq 0$. Indeed, otherwise by (3.1.5) we have $I \subset \mathrm{supp}(x)$, so by the definition of the residual in the algorithm, we would have $r = 0$ at the start of the current iteration, which is a contradiction. Then $J_0 \neq \emptyset$ by Lemma 3.1.12.

The second claim, that $J_0 \cap I = \emptyset$, is also simple. Indeed, recall that by the definition of the algorithm, $r = P_{F^\perp} \in F^\perp = (\mathrm{range}(\Phi_I))^\perp$. It follows that the

observation vector $y = \Phi^* r$ satisfies $y|_I = 0$. Since by its definition the set J contains only nonzero coordinates of y we have $J \cap I = \emptyset$. Since $J_0 \subset J$, the second claim $J_0 \cap I = \emptyset$ follows.

The nontrivial part of the theorem is its last claim, inequality (3.1.3). Suppose it fails. Namely, suppose that $|J_0 \cap \mathrm{supp}(x)| < \frac{1}{2}|J_0|$, and thus

$$|J_0 \backslash \mathrm{supp}(x)| > \frac{1}{2}|J_0|.$$

Set $\Lambda = J_0 \backslash \mathrm{supp}(x)$. By the comparability property of the coordinates in J_0 and since $|\Lambda| > \frac{1}{2}|J_0|$, there is a fraction of energy in Λ:

$$\|y|_\Lambda\|_2 > \frac{1}{\sqrt{5}}\|y|_{J_0}\|_2 \geq \frac{1}{7\sqrt{\log s}}\|x_0\|_2, \tag{3.1.8}$$

where the last inequality holds by Lemma 3.1.12.

On the other hand, we can approximate y by y_0 as

$$\|y|_\Lambda\|_2 \leq \|y|_\Lambda - y_0|_\Lambda\|_2 + \|y_0|_\Lambda\|_2. \tag{3.1.9}$$

Since $\Lambda \subset J$ and using Lemma 3.1.9, we have

$$\|y|_\Lambda - y_0|_\Lambda\|_2 \leq 2.4\varepsilon\|x_0\|_2$$

Furthermore, by definition (3.1.5) of x_0, we have $x_0|_\Lambda = 0$. So, by Part 1 of Proposition 3.1.6,

$$\|y_0|_\Lambda\|_2 \leq 2.03\varepsilon\|x_0\|_2.$$

Using the last two inequalities in (3.1.9), we conclude that

$$\|y|_\Lambda\|_2 \leq 4.43\varepsilon\|x_0\|_2.$$

This is a contradiction to (3.1.8) so long as $\varepsilon \leq 0.03/\sqrt{\log s}$. This proves Theorem 3.1.5. □

Proof of Theorem 3.1.2

The proof of Theorem 3.1.2 parallels that of Theorem 3.1.1. We begin by showing that at every iteration of ROMP, either at least 50% of the selected coordinates from that iteration are from the support of the actual signal v, or the error bound already holds. This directly implies Theorem 3.1.2.

Theorem 3.1.13 (Stable Iteration Invariant of ROMP)**.** *Let Φ be a measurement matrix satisfying the Restricted Isometry Condition with parameters $(4s, \varepsilon)$ for $\varepsilon = 0.01/\sqrt{\log s}$. Let x be a non-zero s-sparse vector with measurements $u = \Phi x + e$. Then at any iteration of ROMP, after the regularization step where I is the current chosen index set, we have $J_0 \cap I = \emptyset$ and (at least) one of the following:*

(i) $|J_0 \cap \mathrm{supp}(v)| \geq \frac{1}{2}|J_0|$;

(ii) $\|x|_{\mathrm{supp}(x)\setminus I}\|_2 \leq 100\sqrt{\log s}\|e\|_2$.

We show that the Iteration Invariant implies Theorem 3.1.2 by examining the three possible cases:

Case 1: (ii) occurs at some iteration. We first note that since $|I|$ is nondecreasing, if (ii) occurs at some iteration, then it holds for all subsequent iterations. To show that this would then imply Theorem 3.1.2, we observe that by the Restricted

Isometry Condition and since $|\text{supp}(\hat{x})| \leq |I| \leq 3s$,

$$(1-\varepsilon)\|\hat{x}-x\|_2 - \|e\|_2 \leq \|\Phi\hat{x} - \Phi x - e\|_2.$$

Then again by the Restricted Isometry Condition and definition of $\hat{x}$,

$$\|\Phi\hat{x} - \Phi x - e\|_2 \leq \|\Phi(x|_I) - \Phi x - e\|_2 \leq (1+\varepsilon)\|x|_{\text{supp}(x)\setminus I}\|_2 + \|e\|_2.$$

Thus we have that

$$\|\hat{x} - x\|_2 \leq \frac{1+\varepsilon}{1-\varepsilon}\|x|_{\text{supp}(x)\setminus I}\|_2 + \frac{2}{1-\varepsilon}\|e\|_2.$$

Thus (ii) of the Iteration Invariant would imply Theorem 3.1.2.

Case 2: (i) occurs at every iteration and J_0 is always non-empty. In this case, by (i) and the fact that J_0 is always non-empty, the algorithm identifies at least one element of the support in every iteration. Thus if the algorithm runs s iterations or until $|I| \geq 2s$, it must be that $\text{supp}(x) \subset I$, meaning that $x|_{\text{supp}(x)\setminus I} = 0$. Then by the argument above for Case 1, this implies Theorem 3.1.2.

Case 3: (i) occurs at each iteration and $J_0 = \emptyset$ for some iteration. By the definition of J_0, if $J_0 = \emptyset$ then $y = \Phi^* r = 0$ for that iteration. By definition of r, this must mean that

$$\Phi^*\Phi(x-w) + \Phi^* e = 0.$$

This combined with Part 1 of Proposition 3.1.6 below (and its proof, see [55]) applied with the set $I' = \text{supp}(x) \cup I$ yields

$$\|x - w + (\Phi^* e)|_{I'}\|_2 \leq 2.03\varepsilon\|x-w\|_2.$$

Then combinining this with Part 2 of the same Proposition, we have

$$\|x - w\|_2 \leq 1.1\|e\|_2.$$

Since $x|_{\mathrm{supp}(x)\setminus I} = (x - w)|_{\mathrm{supp}(x)\setminus I}$, this means that the error bound (ii) must hold, so by Case 1 this implies Theorem 3.1.2.

We now turn to the proof of the Iteration Invariant, Theorem 3.1.13. We prove Theorem 3.1.13 by inducting on each iteration of ROMP. We will show that at each iteration the set of chosen indices is disjoint from the current set I of indices, and that either (i) or (ii) holds. Clearly if (ii) held in a previous iteration, it would hold in all future iterations. Thus we may assume that (ii) has not yet held. Since (i) has held at each previous iteration, we must have

$$|I| \leq 2s. \tag{3.1.10}$$

Consider an iteration of ROMP, and let $r \neq 0$ be the residual at the start of that iteration. Let J_0 and J be the sets found by ROMP in this iteration. As in [55], we consider the subspace

$$H := \mathrm{range}(\Phi_{\mathrm{supp}(v)\cup I})$$

and its complementary subspaces

$$F := \mathrm{range}(\Phi_I), \quad E_0 := \mathrm{range}(\Phi_{\mathrm{supp}(v)\setminus I}).$$

Part 3 of Proposition 3.1.6 states that the subspaces F and E_0 are nearly orthogonal.

For this reason we consider the subspace:

$$E := F^{\perp} \cap H.$$

First we write the residual r in terms of projections onto these subspaces.

Lemma 3.1.14 (Residual)**.** *Here and onward, denote by P_L the orthogonal projection in $\mathbb{R}^m$ onto a linear subspace L. Then the residual r has the following form:*

$$r = P_E \Phi x + P_{F^{\perp}} e.$$

Proof. By definition of the residual r in the ROMP algorithm, $r = P_{F^{\perp}} u = P_{F^{\perp}}(\Phi x + e)$. To complete the proof we need that $P_{F^{\perp}} \Phi x = P_E \Phi x$. This follows from the orthogonal decomposition $H = F + E$ and the fact that $\Phi x \in H$. □

Next we examine the missing portion of the signal as well as its measurements:

$$x_0 := x|_{\mathrm{supp}(x) \setminus I}, \quad u_0 := \Phi x_0 \in E_0. \tag{3.1.11}$$

In the next two lemmas we show that the subspaces E and E_0 are indeed close.

Lemma 3.1.15 (Approximation of the residual)**.** *Let r be the residual vector and u_0 as in* (3.1.11)*. Then*

$$\|u_0 - r\|_2 \leq 2.2\varepsilon \|u_0\|_2 + \|e\|_2.$$

Proof. Since $x - x_0$ has support in I, we have $\Phi x - u_0 = \Phi(x - x_0) \in F$. Then by Lemma 3.1.7, $r = P_E \Phi x + P_{F^{\perp}} e = P_E u_0 + P_{F^{\perp}} e$. Therefore,

$$\|x_0 - r\|_2 = \|x_0 - P_E x_0 - P_{F^{\perp}} e\|_2 \leq \|P_F x_0\|_2 + \|e\|_2.$$

Note that by (3.1.10), the union of the sets I and $\text{supp}(x) \setminus I$ has cardinality no greater than $3s$. Thus by Part 3 of Proposition 3.1.6, we have

$$\|P_F u_0\|_2 + \|e\|_2 = \|P_F P_{E_0} u_0\|_2 + \|e\|_2 \leq 2.2\varepsilon \|u_0\|_2 + \|e\|_2.$$

□

Lemma 3.1.16 (Approximation of the observation)**.** *Let $y_0 = \Phi^* u_0$ and $y = \Phi^* r$. Then for any set $T \subset \{1, \ldots, d\}$ with $|T| \leq 3s$,*

$$\|(y_0 - y)|_T\|_2 \leq 2.4\varepsilon \|x_0\|_2 + (1 + \varepsilon)\|e\|_2.$$

Proof. By Lemma 3.1.15 and the Restricted Isometry Condition we have

$$\|u_0 - r\|_2 \leq 2.2\varepsilon \|\Phi x_0\|_2 + \|e\|_2 \leq 2.2\varepsilon(1 + \varepsilon)\|x_0\|_2 + \|e\|_2 \leq 2.3\varepsilon \|x_0\|_2 + \|e\|_2.$$

Then by Part 2 of Proposition 3.1.6 we have the desired result,

$$\|(y_0 - y)|_T\|_2 \leq (1 + \varepsilon)\|u_0 - r\|_2.$$

□

The result of the theorem requires us to show that we correctly gain a portion of the support of the signal x. To this end, we first show that ROMP correctly chooses a portion of the energy. The regularization step will then imply that the support is also selected correctly. We thus next show that the energy of y when restricted to the sets J and J_0 is sufficiently large.

Lemma 3.1.17 (Localizing the energy)**.** *Let y be the observation vector and x_0 be as in* (3.1.11)*. Then $\|y|_J\|_2 \geq 0.8\|x_0\|_2 - (1+\varepsilon)\|e\|_2$.*

Proof. Let $S = \operatorname{supp}(x) \setminus I$ be the missing support. Since $|S| \leq s$, by definition of J in the algorithm, we have

$$\|y|_J\|_2 \geq \|y|_S\|_2.$$

By Lemma 3.1.16,

$$\|y|_S\|_2 \geq \|y_0|_S\|_2 - 2.4\varepsilon\|x_0\|_2 - (1+\varepsilon)\|e\|_2.$$

Since $x_0|_S = x_0$, Part 1 of Proposition 3.1.6 implies

$$\|y_0|_S\|_2 \geq (1 - 2.03\varepsilon)\|x_0\|_2.$$

These three inequalities yield

$$\|y|_J\|_2 \geq (1 - 2.03\varepsilon)\|x_0\|_2 - 2.4\varepsilon\|x_0\|_2 - (1+\varepsilon)\|e\|_2 \geq 0.8\|x_0\|_2 - (1+\varepsilon)\|e\|_2.$$

This completes the proof. □

Lemma 3.1.18 (Regularizing the energy)**.** *Again let y be the observation vector and x_0 be as in* (3.1.11)*. Then*

$$\|y|_{J_0}\|_2 \geq \frac{1}{4\sqrt{\log s}}\|x_0\|_2 - \frac{\|e\|_2}{2\sqrt{\log s}}.$$

Proof. By Lemma 3.1.11 applied to the vector $y|_J$, we have

$$\|y|_{J_0}\|_2 \geq \frac{1}{2.5\sqrt{\log s}}\|y|_J\|_2.$$

Along with Lemma 3.1.17 this implies the claim. □

We now conclude the proof of Theorem 3.1.13. The claim that $J_0 \cap I = \emptyset$ follows by the same arguments as in [55].

It remains to show its last claim, that either (i) or (ii) holds. Suppose (i) in the theorem fails. That is, suppose $|J_0 \cap \mathrm{supp}(x)| < \frac{1}{2}|J_0|$, which means

$$|J_0 \backslash \mathrm{supp}(x)| > \frac{1}{2}|J_0|.$$

Set $\Lambda = J_0 \backslash \mathrm{supp}(x)$. By the definition of J_0 in the algorithm and since $|\Lambda| > \frac{1}{2}|J_0|$, we have by Lemma 3.1.18,

$$\|y|_\Lambda\|_2 > \frac{1}{\sqrt{5}}\|y|_{J_0}\|_2 \geq \frac{1}{4\sqrt{5\log s}}\|x_0\|_2 - \frac{\|e\|_2}{2\sqrt{5\log s}}. \tag{3.1.12}$$

Next, we also have

$$\|y|_\Lambda\|_2 \leq \|y|_\Lambda - y_0|_\Lambda\|_2 + \|y_0|_\Lambda\|_2. \tag{3.1.13}$$

Since $\Lambda \subset J$ and $|J| \leq s$, by Lemma 3.1.16 we have

$$\|y|_\Lambda - y_0|_\Lambda\|_2 \leq 2.4\varepsilon\|x_0\|_2 + (1+\varepsilon)\|e\|_2.$$

By the definition of x_0 in (3.1.11), it must be that $x_0|_\Lambda = 0$. Thus by Part 1 of Proposition 3.1.6,

$$\|y_0|_\Lambda\|_2 \leq 2.03\varepsilon\|x_0\|_2.$$

Using the previous inequalities along with (3.1.13), we deduce that

$$\|y|_\Lambda\|_2 \leq 4.43\varepsilon\|x_0\|_2 + (1+\varepsilon)\|e\|_2.$$

This is a contradiction to (3.1.12) whenever

$$\varepsilon \leq \frac{0.02}{\sqrt{\log s}} - \frac{\|e\|_2}{\|x_0\|_2}.$$

If this is true, then indeed (i) in the theorem must hold. If it is not true, then by the choice of ε, this implies that

$$\|x_0\|_2 \leq 100\|e\|_2\sqrt{\log s}.$$

This proves Theorem 3.1.13. □

Proof of Corollary 3.1.3

Proof. We first partition x so that $u = \Phi x_{2s} + \Phi(x - x_{2s}) + e$. Then since Φ satisfies the Restricted Isometry Condition with parameters $(8s, \varepsilon)$, by Theorem 3.1.2 and the triangle inequality,

$$\|x_{2s} - \hat{x}\|_2 \leq 104\sqrt{\log 2s}(\|\Phi(x - x_{2s})\|_2 + \|e\|_2), \tag{3.1.14}$$

The following lemma as in [32] relates the 2-norm of a vector's tail to its 1-norm. An application of this lemma combined with (3.1.14) will prove Corollary 3.1.3.

Lemma 3.1.19 (Comparing the norms)**.** *Let $v \in \mathbb{R}^d$, and let v_T be the vector of the*

T largest coordinates in absolute value from v. Then

$$\|v - v_T\|_2 \leq \frac{\|v\|_1}{2\sqrt{T}}.$$

Proof. By linearity, we may assume $\|v\|_1 = d$. Since v_T consists of the largest T coordinates of v in absolute value, we must have that $\|v - v_T\|_2 \leq \sqrt{d-T}$. (This is because the term $\|v - v_T\|_2$ is greatest when the vector v has constant entries.) Then by the AM-GM inequality,

$$\|v - v_T\|_2\sqrt{T} \leq \sqrt{d-T}\sqrt{T} \leq (d - T + T)/2 = d/2 = \|v\|_1/2.$$

This completes the proof. □

By Lemma 29 of [32], we have

$$\|\Phi(x - x_{2s})\|_2 \leq (1+\varepsilon)\Big(\|x - x_{2s}\|_2 + \frac{\|x - x_{2s}\|_1}{\sqrt{s}}\Big).$$

Applying Lemma 3.1.19 to the vector $v = x - x_s$ we then have

$$\|\Phi(x - x_{2s})\|_2 \leq 1.5(1+\varepsilon)\frac{\|x - x_s\|_1}{\sqrt{s}}.$$

Combined with (3.1.14), this proves the corollary.

□

Proof of Corollary 3.1.4

Often one wishes to find a *sparse* approximation to a signal. We now show that by simply truncating the reconstructed vector, we obtain a $2s$-sparse vector very close

to the original signal.

Proof. Let $x_S := x_{2s}$ and $\hat{x_T} := \hat{x}_{2s}$, and let S and T denote the supports of x_S and $\hat{x_T}$ respectively. By Corollary 3.1.3, it suffices to show that $\|x_S - \hat{x}_T\|_2 \leq 3\|x_S - \hat{x}\|_2$.

Applying the triangle inequality, we have

$$\|x_S - \hat{x}_T\|_2 \leq \|(x_S - \hat{x}_T)|_T\|_2 + \|x_S|_{S\backslash T}\|_2 =: a + b.$$

We then have

$$a = \|(x_S - \hat{x}_T)|_T\|_2 = \|(x_S - \hat{x})|_T\|_2 \leq \|x_S - \hat{x}\|_2$$

and

$$b \leq \|\hat{x}|_{S\backslash T}\|_2 + \|(x_S - \hat{x})|_{S\backslash T}\|_2.$$

Since $|S| = |T|$, we have $|S\backslash T| = |T\backslash S|$. By the definition of T, every coordinate of $\hat{x}$ in T is greater than or equal to every coordinate of $\hat{x}$ in T^c in absolute value. Thus we have,

$$\|\hat{x}|_{S\backslash T}\|_2 \leq \|\hat{x}|_{T\backslash S}\|_2 = \|(x_S - \hat{x})|_{T\backslash S}\|_2.$$

Thus $b \leq 2\|x_S - \hat{x}\|_2$, and so

$$a + b \leq 3\|x_S - \hat{x}\|_2.$$

This completes the proof. □

Remark. Corollary 3.1.4 combined with Corollary 3.1.3 and (3.1.2) implies that we

can also estimate a bound on the whole signal v:

$$\|x - \hat{x}_{2s}\|_2 \leq C\sqrt{\log 2s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big).$$

3.1.4 Implementation and Runtime

The Identification step of ROMP, i.e. selection of the subset J, can be done by *sorting* the coordinates of y in the nonincreasing order and selecting s biggest. Many sorting algorithms such as Mergesort or Heapsort provide running times of $O(d \log d)$.

The Regularization step of ROMP, i.e. selecting $J_0 \subset J$, can be done fast by observing that J_0 is an *interval* in the decreasing rearrangement of coefficients. Moreover, the analysis of the algorithm shows that instead of searching over all intervals J_0, it suffices to look for J_0 among $O(\log s)$ *consecutive intervals* with endpoints where the magnitude of coefficients decreases by a factor of 2. (these are the sets A_k in the proof of Lemma 3.1.11). Therefore, the Regularization step can be done in time $O(s)$.

In addition to these costs, the k-th iteration step of ROMP involves *multiplication* of the $d \times m$ matrix Φ^* by a vector, and solving the *least squares problem* with the $d \times |I|$ matrix Φ_I, where $|I| \leq 2s$. For unstructured matrices, these tasks can be done in time dm and $O(s^2 m)$ respectively [2]. Since the submatrix of Φ when restricted to the index set I is near an isometry, using an iterative method such as the Conjugate Gradient Method allows us to solve the least squares method in a constant number of iterations (up to a specific accuracy) [2, Sec. 7.4]. Using such a method then reduces the time of solving the least squares problem to just $O(sm)$. Thus in the cases where ROMP terminates after a fixed number of iterations, the total time to solve

all required least squares problems would be just $O(sm)$. For structured matrices, such as partial Fourier, these times can be improved even more using fast multiply techniques.

In other cases, however, ROMP may need more than a constant number of iterations before terminating, say the full $O(s)$ iterations. In this case, it may be more efficient to maintain the QR factorization of Φ_I and use the Modified Gram-Schmidt algorithm. With this method, solving all the least squares problems takes total time just $O(s^2m)$. However, storing the QR factorization is quite costly, so in situations where storage is limited it may be best to use the iterative methods mentioned above.

ROMP terminates in at most $2s$ iterations. Therefore, for unstructured matrices using the methods mentioned above and in the interesting regime $m \geq \log d$, *the total running time of ROMP is O(dNn).* This is the same bound as for OMP [62].

3.1.5 Numerical Results

Noiseless Numerical Studies

This section describes our experiments that illustrate the signal recovery power of ROMP, as shown in [55]. See Section A.3 for the Matlab code used in these studies. We experimentally examine how many measurements m are necessary to recover various kinds of s-sparse signals in $\mathbb{R}^d$ using ROMP. We also demonstrate that the number of iterations ROMP needs to recover a sparse signal is in practice at most linear the sparsity.

First we describe the setup of our experiments. For many values of the ambient dimension d, the number of measurements m, and the sparsity s, we reconstruct random signals using ROMP. For each set of values, we generate an $m \times d$ Gaussian measurement matrix Φ and then perform 500 independent trials. The results we

obtained using Bernoulli measurement matrices were very similar. In a given trial, we generate an s-sparse signal x in one of two ways. In either case, we first select the support of the signal by choosing s components uniformly at random (independent from the measurement matrix Φ). In the cases where we wish to generate flat signals, we then set these components to one. Our work as well as the analysis of Gilbert and Tropp [62] show that this is a challenging case for ROMP (and OMP). In the cases where we wish to generate sparse compressible signals, we set the i^{th} component of the support to plus or minus $i^{-1/p}$ for a specified value of $0 < p < 1$. We then execute ROMP with the measurement vector $u = \Phi x$.

Figure 3.1.1 depicts the percentage (from the 500 trials) of sparse flat signals that were reconstructed exactly. This plot was generated with $d = 256$ for various levels of sparsity s. The horizontal axis represents the number of measurements m, and the vertical axis represents the exact recovery percentage. We also performed this same test for sparse compressible signals and found the results very similar to those in Figure 3.1.1. Our results show that performance of ROMP is very similar to that of OMP which can be found in [62].

Figure 3.1.2 depicts a plot of the values for m and s at which 99% of sparse flat signals are recovered exactly. This plot was generated with $d = 256$. The horizontal axis represents the number of measurements m, and the vertical axis the sparsity level s.

Theorem 3.1.1 guarantees that ROMP runs with at most $O(s)$ iterations. Figure 3.1.3 depicts the number of iterations executed by ROMP for $d = 10,000$ and $m = 200$. ROMP was executed under the same setting as described above for sparse flat signals as well as sparse compressible signals for various values of p, and the number of iterations in each scenario was averaged over the 500 trials. These aver-

ages were plotted against the sparsity of the signal. As the plot illustrates, only 2 iterations were needed for flat signals even for sparsity s as high as 40. The plot also demonstrates that the number of iterations needed for sparse compressible is higher than the number needed for sparse flat signals, as one would expect. The plot suggests that for smaller values of p (meaning signals that decay more rapidly) ROMP needs more iterations. However it shows that even in the case of $p = 0.5$, only 6 iterations are needed even for sparsity s as high as 20.

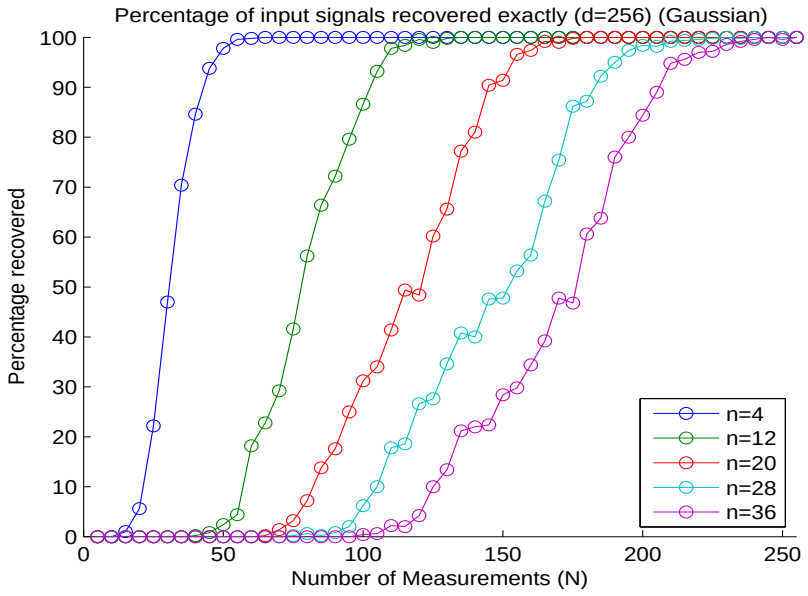


Figure 3.1.1: The percentage of sparse flat signals exactly recovered by ROMP as a function of the number of measurements in dimension $d = 256$ for various levels of sparsity.

Noisy Numerical Studies

This section describes our numerical experiments that illustrate the stability of ROMP as shown in [54]. We study the recovery error using ROMP for both perturbed measurements and signals. The empirical recovery error is actually much better than

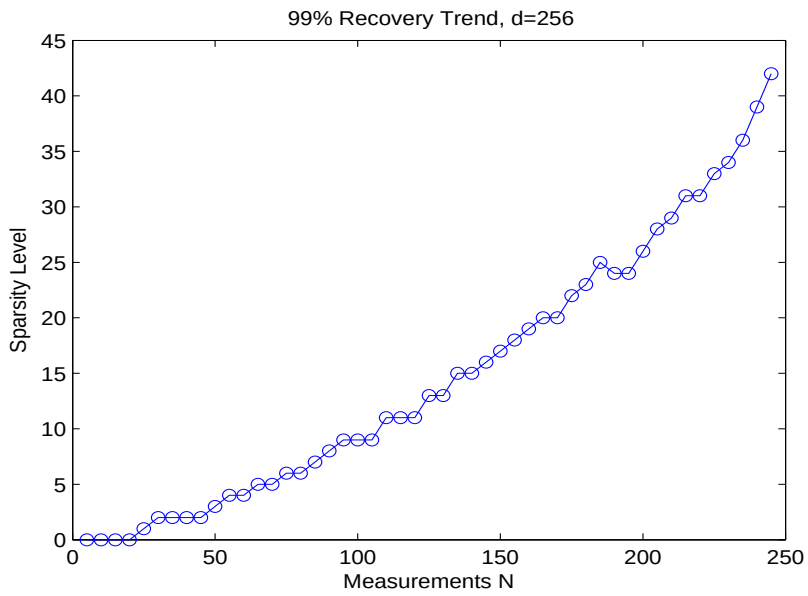


Figure 3.1.2: The 99% recovery limit as a function of the sparsity and the number of measurements for sparse flat signals.

that given in the theorems.

First we describe the setup to our experimental studies. We run ROMP on various values of the ambient dimension d, the number of measurements m, and the sparsity level s, and attempt to reconstruct random signals. For each set of parameters, we perform 500 trials. Initially, we generate an $m \times d$ Gaussian measurement matrix Φ. For each trial, independent of the matrix, we generate an s-sparse signal x by choosing s components uniformly at random and setting them to one. In the case of perturbed signals, we add to the signal a d-dimensional error vector with Gaussian entries. In the case of perturbed measurements, we add an m-dimensional error vector with Gaussian entries to the measurement vector Φx. We then execute ROMP with the measurement vector $u = \Phi x$ or $u + e$ in the perturbed measurement case. After ROMP terminates, we output the reconstructed vector $\hat{x}$ obtained from the least squares calculation and calculate its distance from the original signal.

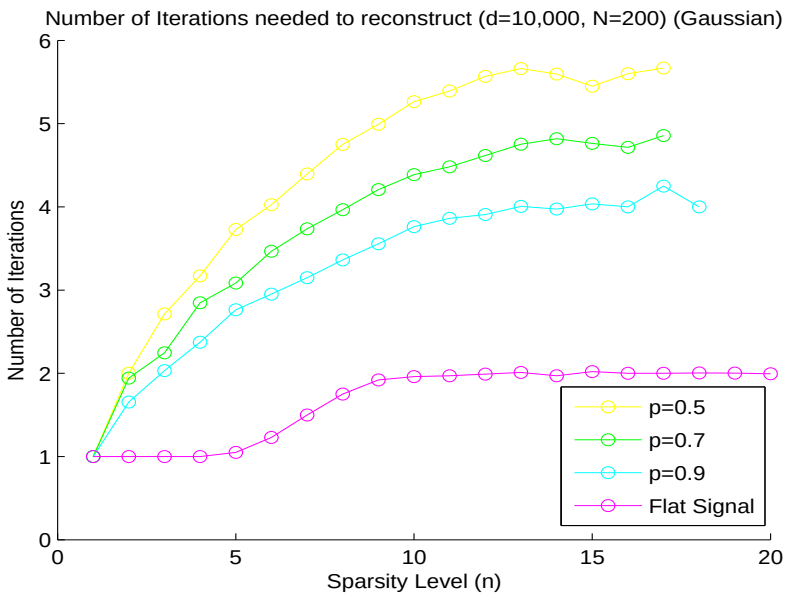


Figure 3.1.3: The number of iterations executed by ROMP as a function of the sparsity in dimension $d = 10,000$ with 200 measurements.

Figure 3.1.4 depicts the recovery error $\|x - \hat{x}\|_2$ when ROMP was run with perturbed measurements. This plot was generated with $d = 256$ for various levels of sparsity s. The horizontal axis represents the number of measurements m, and the vertical axis represents the average normalized recovery error. Figure 3.1.4 confirms the results of Theorem 3.1.2, while also suggesting the bound may be improved by removing the $\sqrt{\log s}$ factor.

Figure 3.1.5 depicts the normalized recovery error when the signal was perturbed by a Gaussian vector. The figure confirms the results of Corollary 3.1.3 while also suggesting again that the logarithmic factor in the corollary is unnecessary.

3.1.6 Summary

There are several critical properties that an ideal algorithm in compressed sensing should possess. One such property is stability, guaranteeing that under small pertur-

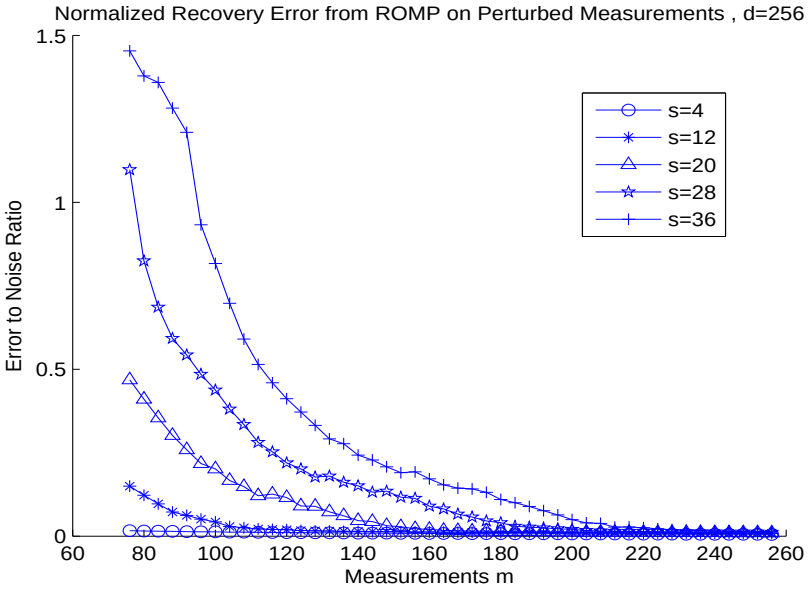


Figure 3.1.4: The error to noise ratio $\frac{\|\hat{x}-x\|_2}{\|e\|_2}$ as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

bations of the inputs, the algorithm still performs approximately correct. Secondly, the algorithm needs to provide uniform guarantees, meaning that with high probability the algorithm works correctly for *all* inputs. Finally, to be ideal in practice, the algorithm would need to have a fast runtime. The ℓ_1-minimization approach to compressed sensing is stable and provides uniform guarantees, but since it relies on the use of Linear Programming, lacks a strong bound on its runtime. The greedy approach is quite fast both in theory and in practice, but had lacked both stability and uniform guarantees. We analyzed the restricted isometry property in a unique way and found consequences that could be used in a greedy fashion. Our breakthrough algorithm ROMP is the first to provide all these benefits (stability, uniform guarantees, and speed), and essentially bridges the gap between the two major approaches in compressed sensing.

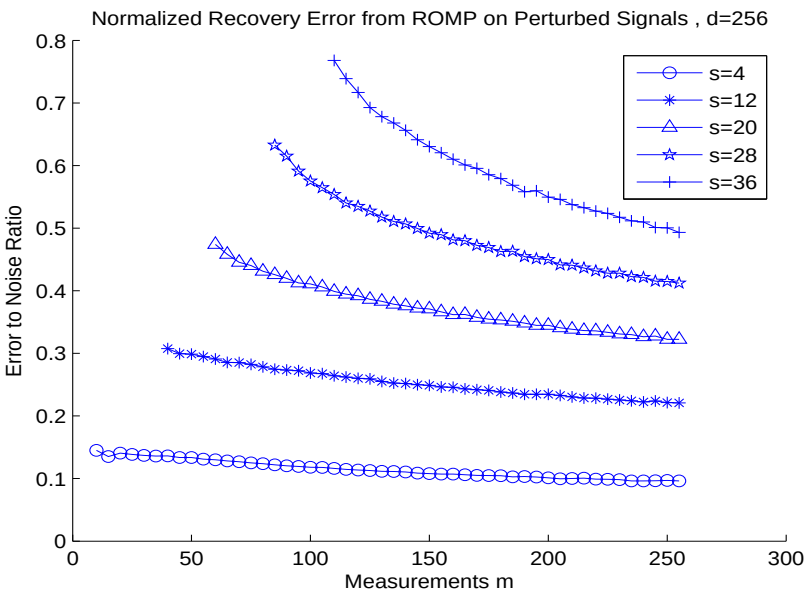


Figure 3.1.5: The error to noise ratio $\frac{\|\hat{x}-x_{2s}\|_2}{\|x-x_s\|_1/\sqrt{s}}$ using a perturbed signal, as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

3.2 Compressive Sampling Matching Pursuit

Regularized Orthogonal Matching Pursuit bridged a critical gap between the major approaches in compressed sensing. It provided the speed of the greedy approach and the strong guarantees of the convex optimization approach. Although its contributions were significant, it still left room for improvement. The requirements imposed by ROMP on the restricted isometry condition were slightly stronger than those imposed by the convex optimization approach. This then in turn weakened the error bounds provided by ROMP in the case of noisy signals and measurements. These issues were resolved by our algorithm Compressive Sampling Matching Pursuit (CoSaMP). A similar algorithm, Subspace Matching Pursuit was also developed around this time, which provides similar benefits to those of CoSaMP. See [14] for details.

3.2.1 Description

One of the key differences between ROMP and OMP is that at each iteration ROMP selects more than one coordinate to be in the support set. Because of this, ROMP is able to make mistakes in the support set, while still correctly reconstructing the original signal. This is accomplished because we bound the number of incorrect choices the algorithm can make. Once the algorithm chooses an incorrect coordinate, however, there is no way for it to be removed from the support set. An alternative approach would be to allow the algorithm to choose incorrectly as well as fix its mistakes in later iterations. In this case, at each iteration we select a slightly larger support set, reconstruct the signal using that support, and use that estimation to calculate the residual.

Tropp and Needell developed a new variant of OMP, Compressive Sampling Matching Pursuit (CoSaMP) [53, 52]. This new algorithm has the same uniform guarantees as ROMP, but does not impose the logarithmic term for the Restricted Isometry Property or in the error bounds. Since the sampling operator Φ satisfies the Restricted Isometry Property, every s entries of the signal proxy $y = \Phi^*\Phi x$ are close in the Euclidean norm to the s corresponding entries of the signal x. Thus as in ROMP, the algorithm first selects the largest coordinates of the signal proxy y and adds these coordinates to the running support set. Next however, the algorithm performs a least squares step to get an estimate b of the signal, and prunes the signal to make it sparse. The algorithm's major steps are described as follows:

1. **Identification.** The algorithm forms a proxy of the residual from the current samples and locates the largest components of the proxy.

2. **Support Merger.** The set of newly identified components is united with the

set of components that appear in the current approximation.

3. **Estimation.** The algorithm solves a least-squares problem to approximate the target signal on the merged set of components.

4. **Pruning.** The algorithm produces a new approximation by retaining only the largest entries in this least-squares signal approximation.

5. **Sample Update.** Finally, the samples are updated so that they reflect the residual, the part of the signal that has not been approximated.

These steps are repeated until the halting criterion is triggered. Initially, we concentrate on methods that use a fixed number of iterations. Section 3.2.4 discusses some other simple stopping rules that may also be useful in practice. Using these ideas, the pseudo-code for CoSaMP can thus be described as follows.

Compressive Matching Pursuit (CoSaMP) [53]

Input: Measurement matrix Φ, measurement vector $u = \Phi x$, sparsity level s

Output: s-sparse reconstructed vector $\hat{x} = a$

Procedure:

Initialize Set $a^0 = 0$, $v = u$, $k = 0$. Repeat the following steps and increment k until the halting criterion is true.

Signal Proxy Set $y = \Phi^* v$, $\Omega = \text{supp} y_{2s}$ and merge the supports: $T = \Omega \cup \text{supp} a^{k-1}$.

Signal Estimation Using least-squares, set $b|_T = \Phi_T^\dagger u$ and $b|_{T^c} = 0$.

Prune To obtain the next approximation, set $a^k = b_s$.

Sample Update Update the current samples: $v = u - \Phi a^k$.

There are a few major concepts of which the algorithm CoSaMP takes advantage. Unlike some other greedy algorithms, CoSaMP selects many components at each iteration. This idea can be found in theoretical work on greedy algorithms by Temlyakov as well as some early work of Gilbert, Muthukrishnan, Strauss and Tropp [34],[63]. It is also the key idea of recent work on the Fourier sampling algorithm [35]. The ROMP and StOMP algorithms also incorporate this notion [55], [23].

The application of the Restricted Isometry Property to compare the norms of vectors under the action of the sampling operator and its adjoint is also key in this algorithm and its analysis. The Restricted Isometry Property is due to Candès and Tao [9]. The application of the property to greedy algorithms is relatively new, and appears in [32] and [55].

Another key idea present in the algorithm is the pruning step to maintain sparsity

of the approximation. This also has significant ramifications in other parts of the analysis and the running time. Since the Restricted Isometry Property only holds for sparse vectors, it is vital in the analysis that the approximation remain sparse. This idea also appears in [32].

Our analysis focuses on mixed-norm error bounds. This idea appears in the work of Candès, Romberg, Tao [5] as well as [32] and [10]. In our analysis, we focus on the fact that if the error is large, the algorithm must make progress. This idea appears in work by Gilbert and Strauss, for example [32].

The L_1-minimization method and the ROMP algorithm provide the strongest known guarantees of sparse recovery. These guarantees are *uniform* in that once the sampling operator satisfies the Restricted Isometry Property, both methods work correctly for all sparse signals. L_1-minimization is based on linear programming, which provides only a polynomial runtime. Greedy algorithms such as OMP and ROMP on the other hand, are much faster both in theory and empirically. Our algorithm CoSaMP provides both uniform guarantees as well as fast runtime, while improving upon the error bounds and Restricted Isometry requirements of ROMP. We describe these results next as we state the main theorems.

3.2.2 Main Theorem

Next we state the main theorem which guarantees exact reconstruction of sparse signals and approximate reconstruction of arbitrary signals. The proof of the theorem is presented in Section 3.2.3.

Theorem 3.2.1 (CoSaMP [53])**.** *Suppose that Φ is an $m \times d$ sampling matrix with restricted isometry constant $\delta_{2s} \leq \mathrm{c}$, as in* (2.1.4)*. Let $u = \Phi x + e$ be a vector of samples of an arbitrary signal, contaminated with arbitrary noise. For a given*

precision parameter η, the algorithm CoSaMP produces an s-sparse approximation $\hat{x}$ that satisfies

$$\|x - \hat{x}\|_2 \leq \mathrm{C} \cdot \max\left\{\eta, \frac{1}{\sqrt{s}} \|x - x_{s/2}\|_1 + \|e\|_2\right\}$$

where $x_{s/2}$ is a best $(s/2)$-sparse approximation to x. The running time is $\mathrm{O}(\mathscr{L} \cdot \log(\|x\|_2 / \eta))$, where $\mathscr{L}$ bounds the cost of a matrix–vector multiply with Φ or Φ^*. Working storage is $\mathrm{O}(d)$.

Remarks. 1. We note that as in the case of ROMP, CoSaMP requires knowledge of the sparsity level s. As described in Section 3.1.1, there are several strategies to estimate this level.

2. In the hypotheses, a bound on the restricted isometry constant δ_{2s} also suffices. Indeed, Corollary 3.2.7 of the sequel implies that $\delta_{4s} \leq 0.1$ holds whenever $\delta_{2s} \leq 0.025$.

3. Theorem 3.2.1 is a result of running CoSaMP using an iterative algorithm to solve the least-squares step. We analyze this step in detail below. In the case of exact arithmetic, we again analyze CoSaMP and provide an iteration count for this case:

Theorem 3.2.2 (Iteration Count)**.** *Suppose that CoSaMP is implemented with exact arithmetic. After at most $6(s + 1)$ iterations, CoSaMP produces an s-sparse approximation $\hat{x}$ that satisfies*

$$\|x - a\|_2 \leq 20\nu,$$

where ν is the unrecoverable energy (3.2.1)*.*

See Theorem 3.2.22 in Section 3.2.3 below for more details.

The algorithm produces an s-sparse approximation whose ℓ_2 error is comparable with the scaled ℓ_1 error of the best $(s/2)$-sparse approximation to the signal. Of

course, the algorithm cannot resolve the uncertainty due to the additive noise, so we also pay for the energy in the noise. This type of error bound is structurally optimal, as discuss when describing the unrecoverable energy below. Some disparity in the sparsity levels (here, s versus $s/2$) seems to be necessary when the recovery algorithm is computationally efficient [58].

To prove our theorem, we show that CoSaMP makes significant progress during each iteration where the approximation error is large relative to *unrecoverable energy* ν in the signal. This quantity measures the baseline error in our approximation that occurs because of noise in the samples or because the signal is not sparse. For our purposes, we define the unrecoverable energy by the following.

$$\nu = \|x - x_s\|_2 + \frac{1}{\sqrt{s}} \|x - x_s\|_1 + \|e\|_2 . \tag{3.2.1}$$

The expression (3.2.1) for the unrecoverable energy can be simplified using Lemma 7 from [32], which states that, for every signal $y \in \mathbb{C}^N$ and every positive integer t, we have

$$\|y - y_t\|_2 \leq \frac{1}{2\sqrt{t}} \|y\|_1 .$$

Choosing $y = x - x_{s/2}$ and $t = s/2$, we reach

$$\nu \leq \frac{1.71}{\sqrt{s}} \left\|x - x_{s/2}\right\|_1 + \|e\|_2 . \tag{3.2.2}$$

In words, the unrecoverable energy is controlled by the scaled ℓ_1 norm of the signal tail.

The term "unrecoverable energy" is justified by several facts. First, we must pay for the ℓ_2 error contaminating the samples. To check this point, define $S = \text{supp} x_s$. The matrix Φ_S is nearly an isometry from ℓ_2^S to ℓ_2^m, so an error in the large components

of the signal induces an error of equivalent size in the samples. Clearly, we can never resolve this uncertainty.

The term $s^{-1/2} \|x - x_s\|_1$ is also required on account of classical results about the Gel'fand widths of the ℓ_1^d ball in ℓ_2^d, due to Kashin [42] and Garnaev–Gluskin [30]. In the language of compressive sampling, their work has the following interpretation. Let Φ be a fixed $m \times d$ sampling matrix. Suppose that, for every signal $x \in \mathbb{C}^d$, there is an algorithm that uses the samples $u = \Phi x$ to construct an approximation a that achieves the error bound

$$\|x - a\|_2 \leq \frac{\mathrm{C}}{\sqrt{s}} \|x\|_1 .$$

Then the number m of measurements must satisfy $m \geq \mathrm{c} s \log(d/s)$.

3.2.3 Proofs of Theorems

Theorem 3.2.1 will be shown by demonstrating that the following iteration invariant holds. These results can be found in [53].

Theorem 3.2.3 (Iteration Invariant)**.** *For each iteration $k \geq 0$, the signal approximation a^k is s-sparse and*

$$\left\|x - a^{k+1}\right\|_2 \leq 0.5 \left\|x - a^k\right\|_2 + 10\nu.$$

In particular,

$$\left\|x - a^k\right\|_2 \leq 2^{-k} \|x\|_2 + 20\nu.$$

We will first show this holds for sparse input signals, and then derive the general case.

When the sampling matrix satisfies the restricted isometry inequalities (2.1.4), it

has several other properties that we require repeatedly in the proof that the CoSaMP algorithm is correct. Our first observation is a simple translation of (2.1.4) into other terms, in the same light as Proposition 3.1.6 used in the proof of ROMP.

Proposition 3.2.4. *Suppose Φ has restricted isometry constant δ_r. Let T be a set of r indices or fewer. Then*

$$\begin{aligned} \left\|\Phi_T^* u\right\|_2 &\leq \sqrt{1+\delta_r}\,\|u\|_2 \\ \left\|\Phi_T^\dagger u\right\|_2 &\leq \frac{1}{\sqrt{1-\delta_r}}\,\|u\|_2 \\ \left\|\Phi_T^*\Phi_T x\right\|_2 &\lessgtr (1\pm\delta_r)\,\|x\|_2 \\ \left\|(\Phi_T^*\Phi_T)^{-1} x\right\|_2 &\lessgtr \frac{1}{1\pm\delta_r}\,\|x\|_2 . \end{aligned}$$

where the last two statements contain an upper and lower bound, depending on the sign chosen.

Proof. The restricted isometry inequalities (2.1.4) imply that the singular values of Φ_T lie between $\sqrt{1-\delta_r}$ and $\sqrt{1+\delta_r}$. The bounds follow from standard relationships between the singular values of Φ_T and the singular values of basic functions of Φ_T. □

A second consequence is that disjoint sets of columns from the sampling matrix span nearly orthogonal subspaces. The following result quantifies this observation.

Proposition 3.2.5 (Approximate Orthogonality)**.** *Suppose Φ has restricted isometry constant δ_r. Let S and T be disjoint sets of indices whose combined cardinality does not exceed r. Then*

$$\|\Phi_S^*\Phi_T\| \leq \delta_r.$$

Proof. Abbreviate $R = S \cup T$, and observe that $\Phi_S^*\Phi_T$ is a submatrix of $\Phi_R^*\Phi_R - \mathrm{Id}$.

The spectral norm of a submatrix never exceeds the norm of the entire matrix. We discern that

$$\|\Phi_S^*\Phi_T\| \leq \|\Phi_R^*\Phi_R - \mathrm{Id}\| \leq \max\{(1+\delta_r) - 1, 1 - (1-\delta_r)\} = \delta_r$$

because the eigenvalues of $\Phi_R^*\Phi_R$ lie between $1-\delta_r$ and $1+\delta_r$. □

This result will be applied through the following corollary.

Corollary 3.2.6. *Suppose Φ has restricted isometry constant δ_r. Let T be a set of indices, and let x be a vector. Provided that $r \geq |T \cup \mathrm{supp}x|$,*

$$\|\Phi_T^*\Phi \cdot x|_{T^c}\|_2 \leq \delta_r \|x|_{T^c}\|_2 .$$

Proof. Define $S = \mathrm{supp}x \setminus T$, so we have $x|_S = x|_{T^c}$. Thus,

$$\|\Phi_T^*\Phi \cdot x|_{T^c}\|_2 = \|\Phi_T^*\Phi \cdot x|_S\|_2 \leq \|\Phi_T^*\Phi_S\| \, \|x|_S\|_2 \leq \delta_r \|x|_{T^c}\|_2 ,$$

owing to Proposition 3.2.5. □

As a second corollary, we show that δ_{2r} gives weak control over the higher restricted isometry constants.

Corollary 3.2.7. *Let c and r be positive integers. Then $\delta_{cr} \leq c \cdot \delta_{2r}$.*

Proof. The result is clearly true for $c = 1, 2$, so we assume $c \geq 3$. Let S be an arbitrary index set of size cr, and let $\mathbf{M} = \mathbf{\Phi_S^*\Phi_S} - \mathrm{Id}$. It suffices to check that $\|\mathbf{M}\| \leq c \cdot \delta_{2r}$. To that end, we break the matrix $\mathbf{M}$ into $r \times r$ blocks, which we denote $\mathbf{M_{ij}}$. A block version of Gershgorin's theorem states that $\|\mathbf{M}\|$ satisfies at

least one of the inequalities

$$|\|\mathbf{M}\| - \|\mathbf{M_{ii}}\|| \le \sum\nolimits_{j \neq i} \|\mathbf{M_{ij}}\| \qquad \text{where } i = 1, 2, \dots, c.$$

The derivation is entirely analogous with the usual proof of Gershgorin's theorem, so we omit the details. For each diagonal block, we have $\|\mathbf{M_{ii}}\| \le \delta_r$ because of the restricted isometry inequalities (2.1.4). For each off-diagonal block, we have $\|\mathbf{M_{ij}}\| \le \delta_{2r}$ because of Proposition 3.2.5. Substitute these bounds into the block Gershgorin theorem and rearrange to complete the proof. □

Finally, we present a result that measures how much the sampling matrix inflates nonsparse vectors. This bound permits us to establish the major results for sparse signals and then transfer the conclusions to the general case.

Proposition 3.2.8 (Energy Bound)**.** *Suppose that* Φ *verifies the upper inequality of* (2.1.4)*, viz.*

$$\|\Phi x\|_2 \le \sqrt{1 + \delta_r}\, \|x\|_2 \qquad \textit{when} \qquad \|x\|_0 \le r.$$

Then, for every signal x*,*

$$\|\Phi x\|_2 \le \sqrt{1 + \delta_r} \left[\|x\|_2 + \frac{1}{\sqrt{r}} \|x\|_1 \right].$$

Proof. First, observe that the hypothesis of the proposition can be regarded as a statement about the operator norm of Φ as a map between two Banach spaces. For a set $I \subset \{1, 2, \dots, N\}$, write B_2^I for the unit ball in $\ell_2(I)$. Define the convex body

$$S = \operatorname{conv} \left\{ \bigcup\nolimits_{|I| \le r} B_2^I \right\} \subset \mathbb{C}^N,$$

and notice that, by hypothesis, the operator norm

$$\|\Phi\|_{S\to 2} = \max_{x\in S} \|\Phi x\|_2 \leq \sqrt{1+\delta_r}.$$

Define a second convex body

$$K = \left\{ x : \|x\|_2 + \frac{1}{\sqrt{r}} \|x\|_1 \leq 1 \right\} \subset \mathbb{C}^N,$$

and consider the operator norm

$$\|\Phi\|_{K\to 2} = \max_{x\in K} \|\Phi x\|_2 .$$

The content of the proposition is the claim that

$$\|\Phi\|_{K\to 2} \leq \|\Phi\|_{S\to 2} .$$

To establish this point, it suffices to check that $K \subset S$.

Choose a vector $x \in K$. We partition the support of x into sets of size r. Let I_0 index the r largest-magnitude components of x, breaking ties lexicographically. Let I_1 index the next largest r components, and so forth. Note that the final block I_J may have fewer than r components. We may assume that $x|_{I_j}$ is nonzero for each j.

This partition induces a decomposition

$$x = x|_{I_0} + \sum\nolimits_{j=0}^{J} x|_{I_j} = \lambda_0 y_0 + \sum\nolimits_{j=0}^{J} \lambda_j y_j$$

where

$$\lambda_j = \left\| x|_{I_j} \right\|_2 \quad \text{and} \quad y_j = \lambda_j^{-1} x|_{I_j}.$$

By construction, each vector y_j belongs to S because it is r-sparse and has unit ℓ_2 norm. We will prove that $\sum_j \lambda_j \leq 1$, which implies that x can be written as a convex combination of vectors from the set S. As a consequence, $x \in S$. It emerges that $K \subset S$.

Fix j in the range $\{1, 2, \ldots, J\}$. It follows that I_j contains at most r elements and I_{j-1} contains exactly r elements. Therefore,

$$\lambda_j = \left\| x|_{I_j} \right\|_2 \leq \sqrt{r} \left\| x|_{I_j} \right\|_\infty \leq \sqrt{r} \cdot \frac{1}{r} \left\| x|_{I_{j-1}} \right\|_1$$

where the last inequality holds because the magnitude of x on the set I_{j-1} dominates its largest entry in I_j. Summing these relations, we obtain

$$\sum\nolimits_{j=1}^{J} \lambda_j \leq \frac{1}{\sqrt{r}} \sum\nolimits_{j=1}^{J} \left\| x|_{I_{j-1}} \right\|_1 = \frac{1}{\sqrt{r}} \left\| x \right\|_1 .$$

It is clear that $\lambda_0 = \left\| x|_{I_0} \right\|_2 \leq \left\| x \right\|_2$. We may conclude that

$$\sum\nolimits_{j=0}^{J} \lambda_j \leq \left\| x \right\|_2 + \frac{1}{\sqrt{r}} \left\| x \right\|_1 \leq 1$$

because $x \in K$. □

Iteration Invariant: Sparse Case

We now commence the proof of Theorem 3.2.3. For the moment, let us assume that the signal is actually sparse. We will remove this assumption later.

The result states that each iteration of the algorithm reduces the approximation error by a constant factor, while adding a small multiple of the noise. As a consequence, when the approximation error is large in comparison with the noise, the

algorithm makes substantial progress in identifying the unknown signal.

Theorem 3.2.9 (Iteration Invariant: Sparse Case)**.** *Assume that x is s-sparse. For each $k \geq 0$, the signal approximation a^k is s-sparse, and*

$$\left\|x - a^{k+1}\right\|_2 \leq 0.5\left\|x - a^k\right\|_2 + 7.5 \left\|e\right\|_2 .$$

In particular,

$$\left\|x - a^k\right\|_2 \leq 2^{-k} \left\|x\right\|_2 + 15 \left\|e\right\|_2 .$$

The argument proceeds in a sequence of short lemmas, each corresponding to one step in the algorithm. Throughout this section, we retain the assumption that x is s-sparse.

Fix an iteration $k \geq 1$. We write $a = a^{k-1}$ for the signal approximation at the beginning of the iteration. Define the residual $r = x - a$, which we interpret as the part of the signal we have not yet recovered. Since the approximation a is always s-sparse, the residual r must be $2s$-sparse. Notice that the vector v of updated samples can be viewed as noisy samples of the residual:

$$v \stackrel{\text{def}}{=} u - \Phi a = \Phi(x - a) + e = \Phi r + e.$$

The identification phase produces a set of components where the residual signal still has a lot of energy.

Lemma 3.2.10 (Identification)**.** *The set $\Omega = \operatorname{supp} y_{2s}$, where $y = \Phi^* v$ is the signal proxy, contains at most $2s$ indices, and*

$$\left\|r|_{\Omega^c}\right\|_2 \leq 0.2223 \left\|r\right\|_2 + 2.34 \left\|e\right\|_2 .$$

Proof. The identification phase forms a proxy $y = \Phi^* v$ for the residual signal. The algorithm then selects a set Ω of $2s$ components from y that have the largest magnitudes. The goal of the proof is to show that the energy in the residual on the set Ω^c is small in comparison with the total energy in the residual.

Define the set $R = \operatorname{supp} r$. Since R contains at most $2s$ elements, our choice of Ω ensures that $\|y|_R\|_2 \leq \|y|_\Omega\|_2$. By squaring this inequality and canceling the terms in $R \cap \Omega$, we discover that

$$\left\|y|_{R\setminus\Omega}\right\|_2 \leq \left\|y|_{\Omega\setminus R}\right\|_2 .$$

Since the coordinate subsets here contain few elements, we can use the restricted isometry constants to provide bounds on both sides.

First, observe that the set $\Omega \setminus R$ contains at most $2s$ elements. Therefore, we may apply Proposition 3.2.4 and Corollary 3.2.6 to obtain

$$\begin{aligned}
\left\|y|_{\Omega\setminus R}\right\|_2 &= \left\|\Phi^*_{\Omega\setminus R}(\Phi r + e)\right\|_2 \\
&\leq \left\|\Phi^*_{\Omega\setminus R}\Phi r\right\|_2 + \left\|\Phi^*_{\Omega\setminus R} e\right\|_2 \\
&\leq \delta_{4s}\|r\|_2 + \sqrt{1+\delta_{2s}}\,\|e\|_2 .
\end{aligned}$$

Likewise, the set $R \setminus \Omega$ contains $2s$ elements or fewer, so Proposition 3.2.4 and Corollary 3.2.6 yield

$$\begin{aligned}
\left\|y|_{R\setminus\Omega}\right\|_2 &= \left\|\Phi^*_{R\setminus\Omega}(\Phi r + e)\right\|_2 \\
&\geq \left\|\Phi^*_{R\setminus\Omega}\Phi\cdot r|_{R\setminus\Omega}\right\|_2 - \left\|\Phi^*_{R\setminus\Omega}\Phi\cdot r|_{\Omega}\right\|_2 - \left\|\Phi^*_{R\setminus\Omega} e\right\|_2 \\
&\geq (1-\delta_{2s})\left\|r|_{R\setminus\Omega}\right\|_2 - \delta_{2s}\|r\|_2 - \sqrt{1+\delta_{2s}}\,\|e\|_2 .
\end{aligned}$$

Since the residual is supported on R, we can rewrite $r|_{R\setminus\Omega} = r|_{\Omega^c}$. Finally, combine

the last three inequalities and rearrange to obtain

$$\|r|_{\Omega^c}\|_2 \leq \frac{(\delta_{2s} + \delta_{4s}) \|r\|_2 + 2\sqrt{1+\delta_{2s}} \|e\|_2}{1 - \delta_{2s}}.$$

Invoke the numerical hypothesis that $\delta_{2s} \leq \delta_{4s} \leq 0.1$ to complete the argument. □

The next step of the algorithm merges the support of the current signal approximation a with the newly identified set of components. The following result shows that components of the signal x outside this set have very little energy.

Lemma 3.2.11 (Support Merger)**.** *Let Ω be a set of at most $2s$ indices. The set $T = \Omega \cup \operatorname{supp} a$ contains at most $3s$ indices, and*

$$\|x|_{T^c}\|_2 \leq \|r|_{\Omega^c}\|_2 .$$

Proof. Since $\operatorname{supp} a \subset T$, we find that

$$\|x|_{T^c}\|_2 = \|(x-a)|_{T^c}\|_2 = \|r|_{T^c}\|_2 \leq \|r|_{\Omega^c}\|_2 ,$$

where the inequality follows from the containment $T^c \subset \Omega^c$. □

The estimation step of the algorithm solves a least-squares problem to obtain values for the coefficients in the set T. We need a bound on the error of this approximation.

Lemma 3.2.12 (Estimation)**.** *Let T be a set of at most $3s$ indices, and define the least-squares signal estimate b by the formulae*

$$b|_T = \Phi_T^\dagger u \qquad \text{and} \qquad b|_{T^c} = 0,$$

where $u = \Phi x + e$. *Then*

$$\|x - b\|_2 \leq 1.112 \, \|x|_{T^c}\|_2 + 1.06 \, \|e\|_2 .$$

This result assumes that we solve the least-squares problem in infinite precision. In practice, the right-hand side of the bound contains an extra term owing to the error from the iterative least-squares solver. Below, we study how many iterations of the least-squares solver are required to make the least-squares error negligible in the present argument.

Proof. Note first that

$$\|x - b\|_2 \leq \|x|_{T^c}\|_2 + \|x|_T - b|_T\|_2 .$$

Using the expression $u = \Phi x + e$ and the fact $\Phi_T^\dagger \Phi_T = \mathrm{Id}_T$, we calculate that

$$\begin{aligned} \|x|_T - b|_T\|_2 &= \big\|x|_T - \Phi_T^\dagger(\Phi \cdot x|_T + \Phi \cdot x|_{T^c} + e)\big\|_2 \\ &= \big\|\Phi_T^\dagger(\Phi \cdot x|_{T^c} + e)\big\|_2 \\ &\leq \big\|(\Phi_T^* \Phi_T)^{-1} \Phi_T^* \Phi \cdot x|_{T^c}\big\|_2 + \big\|\Phi_T^\dagger e\big\|_2. \end{aligned}$$

The cardinality of T is at most $3s$, and x is s-sparse, so Proposition 3.2.4 and Corollary 3.2.6 imply that

$$\begin{aligned} \|x|_T - b|_T\|_2 &\leq \frac{1}{1 - \delta_{3s}} \, \|\Phi_T^* \Phi \cdot x|_{T^c}\|_2 + \frac{1}{\sqrt{1 - \delta_{3s}}} \, \|e\|_2 \\ &\leq \frac{\delta_{4s}}{1 - \delta_{3s}} \, \|x|_{T^c}\|_2 + \frac{\|e\|_2}{\sqrt{1 - \delta_{3s}}}. \end{aligned}$$

Combine the bounds to reach

$$\|x - b\|_2 \leq \left[1 + \frac{\delta_{4s}}{1 - \delta_{3s}}\right] \|x|_{T^c}\|_2 + \frac{\|e\|_2}{\sqrt{1 - \delta_{3s}}}.$$

Finally, invoke the hypothesis that $\delta_{3s} \leq \delta_{4s} \leq 0.1$. □

The final step of each iteration is to prune the intermediate approximation to its largest s terms. The following lemma provides a bound on the error in the pruned approximation.

Lemma 3.2.13 (Pruning)**.** *The pruned approximation b_s satisfies*

$$\|x - b_s\|_2 \leq 2 \|x - b\|_2 .$$

Proof. The intuition is that b_s is close to b, which is close to x. Rigorously,

$$\|x - b_s\|_2 \leq \|x - b\|_2 + \|b - b_s\|_2 \leq 2 \|x - b\|_2 .$$

The second inequality holds because b_s is the best s-sparse approximation to b. In particular, the s-sparse vector x is a worse approximation. □

We now complete the proof of the iteration invariant for sparse signals, Theorem 3.2.9. At the end of an iteration, the algorithm forms a new approximation $a^k = b_s$, which is evidently s-sparse. Applying the lemmas we have established, we

easily bound the error:

$$\begin{aligned}
\left\|x - a^k\right\|_2 &= \|x - b_s\|_2 \\
&\leq 2\,\|x - b\|_2 && \text{Pruning (Lemma 3.2.13)} \\
&\leq 2 \cdot (1.112\,\|x|_{T^c}\|_2 + 1.06\,\|e\|_2) && \text{Estimation (Lemma 3.2.12)} \\
&\leq 2.224\,\|r|_{\Omega^c}\|_2 + 2.12\,\|e\|_2 && \text{Support merger (Lemma 3.2.11)} \\
&\leq 2.224 \cdot (0.2223\,\|r\|_2 + 2.34\,\|e\|_2) + 2.12\,\|e\|_2 && \text{Identification (Lemma 3.2.10)} \\
&< 0.5\,\|r\|_2 + 7.5\,\|e\|_2 \\
&= 0.5\left\|x - a^{k-1}\right\|_2 + 7.5\,\|e\|_2 .
\end{aligned}$$

To obtain the second bound in Theorem 3.2.9, simply solve the error recursion and note that

$$(1 + 0.5 + 0.25 + \dots) \cdot 7.5\,\|e\|_2 \leq 15\,\|e\|_2 .$$

This point completes the argument.

Before extending the iteration invariant to the sparse case, we first analyze in detail the least-sqaures step. This will allow us to completely prove our main result, Theorem 3.2.1.

Least Squares Analysis

To develop an efficient implementation of CoSaMP, it is critical to use an iterative method when we solve the least-squares problem in the estimation step. Here we analyze this step for the noise-free case. The two natural choices are Richardson's iteration and conjugate gradient. The efficacy of these methods rests on the assumption that the sampling operator has small restricted isometry constants. Indeed, since

the set T constructed in the support merger step contains at most $3s$ components, the hypothesis $\delta_{4s} \leq 0.1$ ensures that the condition number

$$\kappa(\Phi_T^* \Phi_T) = \frac{\lambda_{\max}(\Phi_T^* \Phi_T)}{\lambda_{\min}(\Phi_T^* \Phi_T)} \leq \frac{1 + \delta_{3s}}{1 - \delta_{3s}} < 1.223.$$

This condition number is closely connected with the performance of Richardson's iteration and conjugate gradient. In this section, we show that Theorem 3.2.9 holds if we perform a constant number of iterations of either least-squares algorithm.

For completeness, let us explain how Richardson's iteration can be applied to solve the least-squares problems that arise in CoSaMP. Suppose we wish to compute $\mathbf{A}^\dagger \mathbf{u}$ where $\mathbf{A}$ is a tall, full-rank matrix. Recalling the definition of the pseudoinverse, we realize that this amounts to solving a linear system of the form

$$(\mathbf{A}^* \mathbf{A})\mathbf{b} = \mathbf{A}^* \mathbf{u}.$$

This problem can be approached by *splitting* the Gram matrix:

$$\mathbf{A}^* \mathbf{A} = \mathrm{Id} + \mathbf{M}$$

where $\mathbf{M} = \mathbf{A}^* \mathbf{A} - \mathrm{Id}$. Given an initial iterate z^0, Richardon's method produces the subsequent iterates via the formula

$$z^{\ell+1} = \mathbf{A}^* \mathbf{u} - \mathbf{M} \mathbf{z}^\ell.$$

Evidently, this iteration requires only matrix–vector multiplies with $\mathbf{A}$ and $\mathbf{A}^*$. It is worth noting that Richardson's method can be accelerated [2, Sec. 7.2.5], but we omit the details.

It is quite easy to analyze Richardson's iteration [2, Sec. 7.2.1]. Observe that

$$\left\|z^{\ell+1} - \mathbf{A}^{\dagger}\mathbf{u}\right\|_2 = \left\|\mathbf{M}(\mathbf{z}^{\ell} - \mathbf{A}^{\dagger}\mathbf{u})\right\|_2 \leq \|\mathbf{M}\| \left\|z^{\ell} - \mathbf{A}^{\dagger}\mathbf{u}\right\|_2.$$

This recursion delivers

$$\left\|z^{\ell} - \mathbf{A}^{\dagger}\mathbf{u}\right\|_2 \leq \|\mathbf{M}\|^{\ell} \left\|z^{0} - \mathbf{A}^{\dagger}\mathbf{u}\right\|_2 \qquad \text{for } \ell = 0, 1, 2, \ldots.$$

In words, the iteration converges linearly.

In our setting, $\mathbf{A} = \mathbf{\Phi_T}$ where T is a set of at most $3s$ indices. Therefore, the restricted isometry inequalities (2.1.4) imply that

$$\|\mathbf{M}\| = \|\Phi_T^* \Phi_T - \mathrm{Id}\| \leq \delta_{3s}.$$

We have assumed that $\delta_{3s} \leq \delta_{4s} \leq 0.1$, which means that the iteration converges quite fast. Once again, the restricted isometry behavior of the sampling matrix plays an essential role in the performance of the CoSaMP algorithm.

Conjugate gradient provides even better guarantees for solving the least-squares problem, but it is somewhat more complicated to describe and rather more difficult to analyze. We refer the reader to [2, Sec. 7.4] for more information. The following lemma summarizes the behavior of both Richardson's iteration and conjugate gradient in our setting.

Lemma 3.2.14 (Error Bound for LS)**.** *Richardson's iteration produces a sequence* $\{z^{\ell}\}$ *of iterates that satisfy*

$$\left\|z^{\ell} - \Phi_T^{\dagger} u\right\|_2 \leq 0.1^{\ell} \cdot \left\|z^{0} - \Phi_T^{\dagger} u\right\|_2 \qquad \textit{for } \ell = 0, 1, 2, \ldots.$$

Conjugate gradient produces a sequence of iterates that satisfy

$$\left\|z^\ell - \Phi_T^\dagger u\right\|_2 \leq 2 \cdot \rho^\ell \cdot \left\|z^0 - \Phi_T^\dagger u\right\|_2 \qquad \text{for } \ell = 0, 1, 2, \ldots.$$

where

$$\rho = \frac{\sqrt{\kappa(\Phi_T^* \Phi_T)} - 1}{\sqrt{\kappa(\Phi_T^* \Phi_T)} + 1} \leq 0.072.$$

This can even be improved further if the eigenvalues of $\Phi_T^* \Phi_T$ are clustered [64]. Iterative least-squares algorithms must be seeded with an initial iterate, and their performance depends heavily on a wise selection thereof. CoSaMP offers a natural choice for the initializer: the current signal approximation. As the algorithm progresses, the current signal approximation provides an increasingly good starting point for solving the least-squares problem. The following shows that the error in the initial iterate is controlled by the current approximation error.

Lemma 3.2.15 (Initial Iterate for LS)**.** *Let x be an s-sparse signal with noisy samples $u = \Phi x + e$. Let a^{k-1} be the signal approximation at the end of the $(k-1)$th iteration, and let T be the set of components identified by the support merger. Then*

$$\left\|a^{k-1} - \Phi_T^\dagger u\right\|_2 \leq 2.112 \left\|x - a^{k-1}\right\|_2 + 1.06 \left\|e\right\|_2$$

Proof. By construction of T, the approximation a^{k-1} is supported inside T, so

$$\left\|x|_{T^c}\right\|_2 = \left\|(x - a^{k-1})|_{T^c}\right\|_2 \leq \left\|x - a^{k-1}\right\|_2.$$

Using Lemma 3.2.12, we may calculate how far a^{k-1} lies from the solution to the

least-squares problem.

$$\begin{aligned}\left\|a^{k-1} - \Phi_T^\dagger u\right\|_2 &\leq \left\|x - a^{k-1}\right\|_2 + \left\|x - \Phi_T^\dagger u\right\|_2 \\ &\leq \left\|x - a^{k-1}\right\|_2 + 1.112\left\|x|_{T^c}\right\|_2 + 1.06\left\|e\right\|_2 \\ &\leq 2.112\left\|x - a^{k-1}\right\|_2 + 1.06\left\|e\right\|_2 .\end{aligned}$$

□

We need to determine how many iterations of the least-squares algorithm are required to ensure that the approximation produced is sufficiently good to support the performance of CoSaMP.

Corollary 3.2.16 (Estimation by Iterative LS)**.** *Suppose that we initialize the LS algorithm with $z^0 = a^{k-1}$. After at most three iterations, both Richardson's iteration and conjugate gradient produce a signal estimate b that satisfies*

$$\left\|x - b\right\|_2 \leq 1.112\left\|x|_{T^c}\right\|_2 + 0.0022\left\|x - a^{k-1}\right\|_2 + 1.062\left\|e\right\|_2 .$$

Proof. Combine Lemma 3.2.14 and Lemma 3.2.15 to see that three iterations of Richardson's method yield

$$\left\|z^3 - \Phi_T^\dagger u\right\|_2 \leq 0.002112\left\|x - a^{k-1}\right\|_2 + 0.00106\left\|e\right\|_2 .$$

The bound for conjugate gradient is slightly better. Let $b|_T = z^3$. According to the estimation result, Lemma 3.2.12, we have

$$\left\|x - \Phi_T^\dagger u\right\|_2 \leq 1.112\left\|x|_{T^c}\right\|_2 + 1.06\left\|e\right\|_2 .$$

An application of the triangle inequality completes the argument. □

Finally, we need to check that the sparse iteration invariant, Theorem 3.2.9 still holds when we use an iterative least-squares algorithm.

Theorem 3.2.17 (Sparse Iteration Invariant with Iterative LS)**.** *Suppose that we use Richardson's iteration or conjugate gradient for the estimation step, initializing the LS algorithm with the current approximation a^{k-1} and performing three LS iterations. Then Theorem 3.2.9 still holds.*

Proof. We repeat the calculation from the above case using Corollary 3.2.16 instead of the simple estimation lemma. To that end, recall that the residual $r = x - a^{k-1}$. Then

$$\begin{aligned}
\left\|x - a^k\right\|_2 &\leq 2 \left\|x - b\right\|_2 \\
&\leq 2 \cdot (1.112 \left\|x|_{T^c}\right\|_2 + 0.0022 \left\|r\right\|_2 + 1.062 \left\|e\right\|_2) \\
&\leq 2.224 \left\|r|_{\Omega^c}\right\|_2 + 0.0044 \left\|r\right\|_2 + 2.124 \left\|e\right\|_2 \\
&\leq 2.224 \cdot (0.2223 \left\|r\right\|_2 + 2.34 \left\|e\right\|_2) + 0.0044 \left\|r\right\|_2 + 2.124 \left\|e\right\|_2 \\
&< 0.5 \left\|r\right\|_2 + 7.5 \left\|e\right\|_2 \\
&= 0.5 \left\|x - a^{k-1}\right\|_2 + 7.5 \left\|e\right\|_2 .
\end{aligned}$$

This bound is precisely what is required for the theorem to hold. □

We are now prepared to extend the iteration invariant to the general case, and prove Theorem 3.2.1.

Extension to General Signals

In this section, we finally complete the proof of the main result for CoSaMP, Theorem 3.2.3. The remaining challenge is to remove the hypothesis that the target signal is sparse, which we framed in Theorems 3.2.9 and 3.2.17. Although this difficulty might seem large, the solution is simple and elegant. It turns out that we can view the noisy samples of a general signal as samples of a sparse signal contaminated with a different noise vector that implicitly reflects the tail of the original signal.

Lemma 3.2.18 (Reduction to Sparse Case)**.** *Let x be an arbitrary vector in $\mathbb{C}^N$. The sample vector $u = \Phi x + e$ can also be expressed as $u = \Phi x_s + \widetilde{e}$ where*

$$\|\widetilde{e}\|_2 \leq 1.05 \left[\|x - x_s\|_2 + \frac{1}{\sqrt{s}} \|x - x_s\|_1 \right] + \|e\|_2 .$$

Proof. Decompose $x = x_s + (x - x_s)$ to obtain $u = \Phi x_s + \widetilde{e}$ where $\widetilde{e} = \Phi(x - x_s) + e$. To compute the size of the error term, we simply apply the triangle inequality and Proposition 3.2.8:

$$\|\widetilde{e}\|_2 \leq \sqrt{1 + \delta_s} \left[\|x - x_s\|_2 + \frac{1}{\sqrt{s}} \|x - x_s\|_1 \right] + \|e\|_2 .$$

Finally, invoke the fact that $\delta_s \leq \delta_{4s} \leq 0.1$ to obtain $\sqrt{1 + \delta_s} \leq 1.05$. □

This lemma is just the tool we require to complete the proof of Theorem 3.2.3.

Proof of Theorem 3.2.3. Let x be a general signal, and use Lemma 3.2.18 to write the noisy vector of samples $u = \Phi x_s + \widetilde{e}$. Apply the sparse iteration invariant, Theorem 3.2.9, or the analog for iterative least-squares, Theorem 3.2.17. We obtain

$$\left\| x_s - a^{k+1} \right\|_2 \leq 0.5 \left\| x_s - a^k \right\|_2 + 7.5 \|\widetilde{e}\|_2 .$$

Invoke the lower and upper triangle inequalities to obtain

$$\left\|x - a^{k+1}\right\|_2 \le 0.5\left\|x - a^k\right\|_2 + 7.5\left\|\widetilde{e}\right\|_2 + 1.5\left\|x - x_s\right\|_2.$$

Finally, recall the estimate for $\left\|\widetilde{e}\right\|_2$ from Lemma 3.2.18, and simplify to reach

$$\begin{aligned}\left\|x - a^{k+1}\right\|_2 &\le 0.5\left\|x - a^k\right\|_2 + 9.375\left\|x - x_s\right\|_2 + \frac{7.875}{\sqrt{s}}\left\|x - x_s\right\|_1 + 7.5\left\|e\right\|_2 \\ &< 0.5\left\|x - a^k\right\|_2 + 10\nu.\end{aligned}$$

where ν is the unrecoverable energy (3.2.1). □

We have now collected all the material we need to establish the main result. Fix a precision parameter η. After at most $\mathrm{O}(\log(\left\|x\right\|_2/\eta))$ iterations, CoSaMP produces an s-sparse approximation a that satisfies

$$\left\|x - a\right\|_2 \le \mathrm{C} \cdot (\eta + \nu)$$

in consequence of Theorem 3.2.3. Apply inequality (3.2.2) to bound the unrecoverable energy ν in terms of the ℓ_1 norm. We see that the approximation error satisfies

$$\left\|x - a\right\|_2 \le \mathrm{C} \cdot \max\left\{\eta, \frac{1}{\sqrt{s}}\left\|x - x_{s/2}\right\|_1 + \left\|e\right\|_2\right\}.$$

According to Theorem 3.2.23, each iteration of CoSaMP is completed in time $\mathrm{O}(\mathscr{L})$, where $\mathscr{L}$ bounds the cost of a matrix–vector multiplication with Φ or Φ^*. The total runtime, therefore, is $\mathrm{O}(\mathscr{L}\log(\left\|x\right\|_2/\eta))$. The total storage is $\mathrm{O}(N)$.

Finally, in the statement of the theorem, we replace δ_{4s} with δ_{2s} by means of Corollary 3.2.7, which states that $\delta_{cr} \le c \cdot \delta_{2r}$ for any positive integers c and r.

Iteration Count for Exact Arithmetic

As promised, we now provide an iteration count for CoSaMP in the case of exact arithmetic. These results can be found in [52].

We obtain an estimate on the number of iterations of the CoSaMP algorithm necessary to identify the recoverable energy in a sparse signal, assuming exact arithmetic. Except where stated explicitly, we assume that x is s-sparse. It turns out that the number of iterations depends heavily on the signal structure. Let us explain the intuition behind this fact.

When the entries in the signal decay rapidly, the algorithm must identify and remove the largest remaining entry from the residual before it can make further progress on the smaller entries. Indeed, the large component in the residual contaminates each component of the signal proxy. In this case, the algorithm may require an iteration or more to find each component in the signal.

On the other hand, when the s entries of the signal are comparable, the algorithm can simultaneously locate many entries just by reducing the norm of the residual below the magnitude of the smallest entry. Since the largest entry of the signal has magnitude at least $s^{-1/2}$ times the ℓ_2 norm of the signal, the algorithm can find all s components of the signal after about $\log s$ iterations.

To quantify these intuitions, we want to collect the components of the signal into groups that are comparable with each other. To that end, define the *component bands* of a signal x by the formulae

$$B_j \stackrel{\text{def}}{=} \left\{ i : 2^{-(j+1)} \|x\|_2^2 < |x_i|^2 \leq 2^{-j} \|x\|_2^2 \right\} \qquad \text{for } j = 0, 1, 2, \ldots. \tag{3.2.3}$$

The *profile* of the signal is the number of bands that are nonempty:

$$\text{profile}(x) \stackrel{\text{def}}{=} \#\{j : B_j \neq \emptyset\}.$$

In words, the profile counts how many orders of magnitude at which the signal has coefficients. It is clear that the profile of an s-sparse signal is at most s.

See Figure 3.2.1 for images of stylized signals with different profiles.

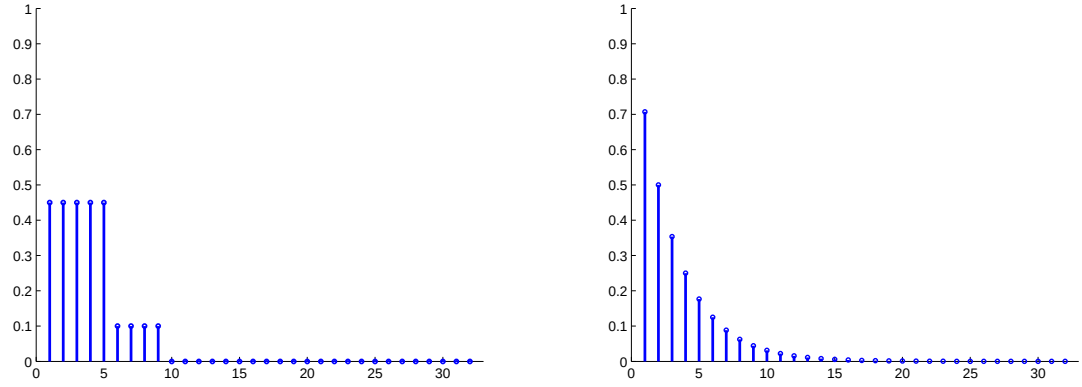


Figure 3.2.1: Illustration of two unit-norm signals with sharply different profiles (left: low, right: high).

First, we prove a result on the number of iterations needed to acquire an s-sparse signal. At the end of the section, we extend this result to general signals, which yields Theorem 3.2.2.

Theorem 3.2.19 (Iteration Count: Sparse Case)**.** *Let x be an s-sparse signal, and define $p = \text{profile}(x)$. After at most*

$$p \log_{4/3}(1 + 4.6\sqrt{s/p}) + 6$$

iterations, CoSaMP produces an approximation a that satisfies

$$\|x - a\|_2 \leq 17 \|e\|_2 .$$

For a fixed s, the bound on the number of iterations achieves its maximum value at $p = s$. Since $\log_{4/3} 5.6 < 6$, the number of iterations never exceeds $6(s + 1)$.

Let us instate some notation that will be valuable in the proof of the theorem. We write $p = \text{profile}(x)$. For each $k = 0, 1, 2, \ldots$, the signal a^k is the approximation after the kth iteration. We abbreviate $S_k = \text{supp} a^k$, and we define the residual signal $r^k = x - a^k$. The norm of the residual can be viewed as the approximation error.

For a nonnegative integer j, we may define an auxiliary signal

$$y^j \stackrel{\text{def}}{=} x|_{\bigcup_{i \geq j} B_i}$$

In other words, y^j is the part of x contained in the bands B_j, B_{j+1}, B_{j+2}, ... For each $j \in J$, we have the estimate

$$\left\|y^j\right\|_2^2 \leq \sum\nolimits_{i \geq j} 2^{-i} \|x\|_2^2 \cdot |B_i| \tag{3.2.4}$$

by definition of the bands. These auxiliary signals play a key role in the analysis.

The proof of the theorem involves a sequence of lemmas. The first object is to establish an alternative that holds in each iteration. One possibility is that the approximation error is small, which means that the algorithm is effectively finished. Otherwise, the approximation error is dominated by the energy in the unidentified part of the signal, and the subsequent approximation error is a constant factor smaller.

Lemma 3.2.20. *For each iteration $k = 0, 1, 2, \ldots$, at least one of the following*

alternatives holds. Either

$$\left\|r^k\right\|_2 \leq 70 \left\|e\right\|_2 \tag{3.2.5}$$

or else

$$\left\|r^k\right\|_2 \leq 2.3\left\|x|_{S_k^c}\right\|_2 \qquad \textit{and} \tag{3.2.6}$$

$$\left\|r^{k+1}\right\|_2 \leq 0.75\left\|r^k\right\|_2. \tag{3.2.7}$$

Proof. Define T_k as the merged support that occurs during iteration k. The pruning step ensures that the support S_k of the approximation at the end of the iteration is a subset of the merged support, so

$$\left\|x|_{T_k^c}\right\|_2 \leq \left\|x|_{S_k^c}\right\|_2 \qquad \text{for } k = 1, 2, 3, \ldots.$$

At the end of the kth iteration, the pruned vector b_s becomes the next approximation a^k, so the estimation and pruning results, Lemmas 3.2.12 and 3.2.13, together imply that

$$\begin{aligned}\left\|r^k\right\|_2 &\leq 2 \cdot (1.112\left\|x|_{T_k^c}\right\|_2 + 1.06 \left\|e\right\|_2) \\ &\leq 2.224\left\|x|_{S_k^c}\right\|_2 + 2.12 \left\|e\right\|_2 \qquad \text{for } k = 1, 2, 3, \ldots.\end{aligned} \tag{3.2.8}$$

Note that the same relation holds trivially for iteration $k = 0$ because $r^0 = x$ and $S_0 = \emptyset$.

Suppose that there is an iteration $k \geq 0$ where

$$\left\|x|_{S_k^c}\right\|_2 < 30 \left\|e\right\|_2.$$

We can introduce this bound directly into the inequality (3.2.8) to obtain the first conclusion (3.2.5).

Suppose on the contrary that in iteration k we have

$$\left\|x|_{S_k^c}\right\|_2 \geq 30 \left\|e\right\|_2 .$$

Introducing this relation into the inequality (3.2.8) leads quickly to the conclusion (3.2.6). We also have the chain of relations

$$\left\|r^k\right\|_2 \geq \left\|r^k|_{S_k^c}\right\|_2 = \left\|(x - a^k)|_{S_k^c}\right\|_2 = \left\|x|_{S_k^c}\right\|_2 \geq 30 \left\|e\right\|_2 .$$

Therefore, the sparse iteration invariant, Theorem 3.2.9 ensures that (3.2.7) holds.

□

The next lemma contains the critical part of the argument. Under the second alternative in the previous lemma, we show that the algorithm completely identifies the support of the signal, and we bound the number of iterations required to do so.

Lemma 3.2.21. *Fix* $K = \lfloor p \log_{4/3}(1 + 4.6\sqrt{s/p}) \rfloor$. *Assume that* (3.2.6) *and* (3.2.7) *are in force for each iteration* $k = 0, 1, 2, \ldots, K$. *Then* $\operatorname{supp} a^K = \operatorname{supp} x$.

Proof. First, we check that, once the norm of the residual is smaller than each element of a band, the components in that band persist in the support of each subsequent approximation. Define J to be the set of nonempty bands, and fix a band $j \in J$. Suppose that, for some iteration k, the norm of the residual satisfies

$$\left\|r^k\right\|_2 \leq 2^{-(j+1)/2} \left\|x\right\|_2 . \tag{3.2.9}$$

Then it must be the case that $B_j \subset \operatorname{supp} a^k$. If not, then some component $i \in B_j$

appears in the residual: $r_i^k = x_i$. This supposition implies that

$$\left\|r^k\right\|_2 \geq |x_i| > 2^{-(j+1)/2} \left\|x\right\|_2,$$

an evident contradiction. Since (3.2.7) guarantees that the norm of the residual declines in each iteration, (3.2.9) ensures that the support of each subsequent approximation contains B_j.

Next, we bound the number of iterations required to find the next nonempty band B_j, given that we have already identified the bands B_i where $i < j$. Formally, assume that the support S_k of the current approximation contains B_i for each $i < j$. In particular, the set of missing components $S_k^c \subset \operatorname{supp} y^j$. It follows from relation (3.2.6) that

$$\left\|r^k\right\|_2 \leq 2.3 \left\|y^j\right\|_2.$$

We can conclude that we have identified the band B_j in iteration $k + \ell$ if

$$\left\|r^{k+\ell}\right\|_2 \leq 2^{-(j+1)/2} \left\|x\right\|_2.$$

According to (3.2.7), we reduce the error by a factor of $\beta^{-1} = 0.75$ during each iteration. Therefore, the number ℓ of iterations required to identify B_j is at most

$$\log_\beta \left\lceil \frac{2.3 \left\|y^j\right\|_2}{2^{-(j+1)/2} \left\|x\right\|_2} \right\rceil$$

We discover that the total number of iterations required to identify all the (nonempty) bands is at most

$$k_\star \stackrel{\text{def}}{=} \sum\nolimits_{j \in J} \log_\beta \left\lceil 2.3 \cdot \frac{2^{(j+1)/2} \left\|y^j\right\|_2}{\left\|x\right\|_2} \right\rceil.$$

For each iteration $k \geq \lfloor k_\star \rfloor$, it follows that $\operatorname{supp} a^k = \operatorname{supp} x$.

It remains to bound $k_\star$ in terms of the profile p of the signal. For convenience, we focus on a slightly different quantity. First, observe that $p = |J|$. Using the geometric mean–arithmetic mean inequality, we discover that

$$\begin{aligned}\exp\left\{\frac{1}{p}\sum\nolimits_{j\in J}\log\left\lceil 2.3\cdot\frac{2^{(j+1)/2}\,\|y^j\|_2}{\|x\|_2}\right\rceil\right\} &\leq \exp\left\{\frac{1}{p}\sum\nolimits_{j\in J}\log\left(1+2.3\cdot\frac{2^{(j+1)/2}\,\|y^j\|_2}{\|x\|_2}\right)\right\}\\ &\leq \frac{1}{p}\sum\nolimits_{j\in J}\left(1+2.3\cdot\frac{2^{(j+1)/2}\,\|y^j\|_2}{\|x\|_2}\right)\\ &= 1+\frac{2.3}{p}\sum\nolimits_{j\in J}\left(\frac{2^{j+1}\,\|y^j\|_2^2}{\|x\|_2^2}\right)^{1/2}.\end{aligned}$$

To bound the remaining sum, we recall the relation (3.2.4). Then we invoke Jensen's inequality and simplify the result.

$$\begin{aligned}\frac{1}{p}\sum_{j\in J}\left(\frac{2^{j+1}\,\|y^j\|_2^2}{\|x\|_2^2}\right)^{1/2} &\leq \frac{1}{p}\sum\nolimits_{j\in J}\left(2^{j+1}\sum\nolimits_{i\geq j}2^{-i}\,|B_i|\right)^{1/2} \leq \left(\frac{1}{p}\sum\nolimits_{j\in J}2^{j+1}\sum_{i\geq j}2^{-i}\,|B_i|\right)^{1/2}\\ &\leq \left(\frac{1}{p}\sum\nolimits_{i\geq 0}|B_i|\sum\nolimits_{j\leq i}2^{j-i+1}\right)^{1/2} \leq \left(\frac{4}{p}\sum\nolimits_{i\geq 0}|B_i|\right)^{1/2}\\ &= 2\sqrt{s/p}.\end{aligned}$$

The final equality holds because the total number of elements in all the bands equals the signal sparsity s. Combining these bounds, we reach

$$\exp\left\{\frac{1}{p}\sum\nolimits_{j\in J}\log\left\lceil 2.3\cdot\frac{2^{(j+1)/2}\,\|y^j\|_2}{\|x\|_2}\right\rceil\right\} \leq 1+4.6\sqrt{s/p}.$$

Take logarithms, multiply by p, and divide through by $\log\beta$. We conclude that the

required number of iterations $k_\star$ is bounded as

$$k_\star \leq p \log_\beta(1 + 4.6\sqrt{s/p}).$$

This is the advertised conclusion. □

Finally, we check that the algorithm produces a small approximation error within a reasonable number of iterations.

Proof of Theorem 3.2.19. Abbreviate $K = \lfloor p \log(1+4.6\sqrt{s/p}) \rfloor$. Suppose that (3.2.5) never holds during the first K iterations of the algorithm. Under this circumstance, Lemma 3.2.20 implies that both (3.2.6) and (3.2.7) hold during each of these K iterations. It follows from Lemma 3.2.21 that the support S_K of the Kth approximation equals the support of x. Since S_K is contained in the merged support T_K, we see that the vector $x|_{T_K^c} = 0$. Therefore, the estimation and pruning results, Lemmas 3.2.12 and 3.2.13, show that

$$\left\|r^K\right\|_2 \leq 2 \cdot \left(1.112 \left\|x|_{T_K^c}\right\|_2 + 1.06 \left\|e\right\|_2\right) = 2.12 \left\|e\right\|_2.$$

This estimate contradicts (3.2.5).

It follows that there is an iteration $k \leq K$ where (3.2.5) is in force. Repeated applications of the iteration invariant, Theorem 3.2.9, allow us to conclude that

$$\left\|r^{K+6}\right\|_2 < 17 \left\|e\right\|_2.$$

This point completes the argument. □

Finally, we extend the sparse iteration count result to the general case.

Theorem 3.2.22 (Iteration Count)**.** *Let x be an arbitrary signal, and define $p =$* profile(x_s)*. After at most*

$$p \log_{4/3}(1 + 4.6\sqrt{s/p}) + 6$$

iterations, CoSaMP produces an approximation a that satisfies

$$\|x - a\|_2 \leq 20 \|e\|_2 .$$

Proof. Let x be a general signal, and let $p =$ profile(x_s). Lemma 3.2.18 allows us to write the noisy vector of samples $u = \Phi x_s + \widetilde{e}$. The sparse iteration count result, Theorem 3.2.19, states that after at most

$$p \log_{4/3}(1 + 4.6\sqrt{s/p}) + 6$$

iterations, the algorithm produces an approximation a that satisfies

$$\|x_s - a\|_2 \leq 17 \|\widetilde{e}\|_2 .$$

Apply the lower triangle inequality to the left-hand side. Then recall the estimate for the noise in Lemma 3.2.18, and simplify to reach

$$\begin{aligned} \|x - a\|_2 &\leq 17 \|\widetilde{e}\|_2 + \|x - x_s\|_2 \\ &\leq 18.9 \|x - x_s\|_2 + \frac{17.9}{\sqrt{s}} \|x - x_s\|_1 + 17 \|e\|_2 \\ &< 20\nu, \end{aligned}$$

where ν is the unrecoverable energy. ∎

Proof of Theorem 3.2.2. Invoke Theorem 3.2.22. Recall that the estimate for the number of iterations is maximized with $p = s$, which gives an upper bound of $6(s+1)$ iterations, independent of the signal. □

3.2.4 Implementation and Runtime

CoSaMP was designed to be a practical method for signal recovery. An efficient implementation of the algorithm requires some ideas from numerical linear algebra, as well as some basic techniques from the theory of algorithms. This section discusses the key issues and develops an analysis of the running time for the two most common scenarios.

We focus on the least-squares problem in the estimation step because it is the major obstacle to a fast implementation of the algorithm. The algorithm guarantees that the matrix Φ_T never has more than $3s$ columns, so our assumption $\delta_{4s} \leq 0.1$ implies that the matrix Φ_T is extremely well conditioned. As a result, we can apply the pseudoinverse $\Phi_T^\dagger = (\Phi_T^* \Phi_T)^{-1} \Phi_T^*$ very quickly using an iterative method, such as Richardson's iteration [2, Sec. 7.2.3] or conjugate gradient [2, Sec. 7.4]. These techniques have the additional advantage that they only interact with the matrix Φ_T through its action on vectors. It follows that the algorithm performs better when the sampling matrix has a fast matrix–vector multiply.

Let us stress that direct methods for least squares are likely to be extremely inefficient in this setting. The first reason is that each least-squares problem may contain substantially different sets of columns from Φ. As a result, it becomes necessary to perform a completely new QR or SVD factorization during each iteration at a cost of $\mathrm{O}(s^2 m)$. The second problem is that computing these factorizations typically requires direct access to the columns of the matrix, which is problematic when the

matrix is accessed through its action on vectors. Third, direct methods have storage costs $\mathrm{O}(sm)$, which may be deadly for large-scale problems.

The remaining steps of the algorithm involve standard techniques. Let us estimate the operation counts.

Proxy Forming the proxy is dominated by the cost of the matrix–vector multiply $\Phi^* v$.

Identification We can locate the largest $2s$ entries of a vector in time $\mathrm{O}(N)$ using the approach in [11, Ch. 9]. In practice, it may be faster to sort the entries of the signal in decreasing order of magnitude at cost $\mathrm{O}(N \log N)$ and then select the first $2s$ of them. The latter procedure can be accomplished with quicksort, mergesort, or heapsort [11, Sec. II]. To implement the algorithm to the letter, the sorting method needs to be stable because we stipulate that ties are broken lexicographically. This point is not important in practice.

Support Merger We can merge two sets of size $\mathrm{O}(s)$ in expected time $\mathrm{O}(s)$ using randomized hashing methods [11, Ch. 11]. One can also sort both sets first and use the elementary merge procedure [11, p. 29] for a total cost $\mathrm{O}(s \log s)$.

LS estimation We use Richardson's iteration or conjugate gradient to compute $\Phi_T^\dagger u$. Initializing the least-squares algorithm requires a matrix–vector multiply with Φ_T^*. Each iteration of the least-squares method requires one matrix–vector multiply each with Φ_T and Φ_T^*. Since Φ_T is a submatrix of Φ, the matrix–vector multiplies can also be obtained from multiplication with the full matrix. We proved above that a constant number of least-squares iterations suffices for Theorem 3.2.3 to hold.

Pruning This step is similar to identification. Pruning can be implemented in time

$\mathrm{O}(s)$, but it may be preferable to sort the components of the vector by magnitude and then select the first s at a cost of $\mathrm{O}(s \log s)$.

Sample Update This step is dominated by the cost of the multiplication of Φ with the s-sparse vector a^k.

Table 3.1 summarizes this discussion in two particular cases. The first column shows what happens when the sampling matrix Φ is applied to vectors in the standard way, but we have random access to submatrices. The second column shows what happens when the sampling matrix Φ and its adjoint Φ^* both have a fast multiply with cost $\mathscr{L}$, where we assume that $\mathscr{L} \geq N$. A typical value is $\mathscr{L} = \mathrm{O}(N \log N)$. In particular, a partial Fourier matrix satisfies this bound.

Table 3.1: Operation count for CoSaMP. Big-O notation is omitted for legibility. The dimensions of the sampling matrix Φ are $m \times N$; the sparsity level is s. The number $\mathscr{L}$ bounds the cost of a matrix–vector multiply with Φ or Φ^*.

Step	Standard multiply	Fast multiply
Form proxy	mN	$\mathscr{L}$
Identification	N	N
Support merger	s	s
LS estimation	sm	$\mathscr{L}$
Pruning	s	s
Sample update	sm	$\mathscr{L}$
Total per iteration	$\mathrm{O}(mN)$	$\mathrm{O}(\mathscr{L})$

Finally, we note that the storage requirements of the algorithm are also favorable. Aside from the storage required by the sampling matrix, the algorithm constructs only one vector of length N, the signal proxy. The sample vectors u and v have length m, so they require $\mathrm{O}(m)$ storage. The signal approximations can be stored

using sparse data structures, so they require at most $\mathrm{O}(s \log N)$ storage. Similarly, the index sets that appear require only $\mathrm{O}(s \log N)$ storage. The total storage is at worst $\mathrm{O}(N)$.

The following result summarizes this discussion.

Theorem 3.2.23 (Resource Requirements)**.** *Each iteration of CoSaMP requires* $\mathrm{O}(\mathscr{L})$ *time, where* $\mathscr{L}$ *bounds the cost of a multiplication with the matrix* Φ *or* Φ^**. The algorithm uses storage* $\mathrm{O}(N)$*.*

Algorithmic Variations

This section describes other possible halting criteria and their consequences. It also proposes some other variations on the algorithm.

There are three natural approaches to halting the algorithm. The first, which we have discussed in the body of the paper, is to stop after a fixed number of iterations. Another possibility is to use the norm $\|v\|_2$ of the current samples as evidence about the norm $\|r\|_2$ of the residual. A third possibility is to use the magnitude $\|y\|_\infty$ of the entries of the proxy to bound the magnitude $\|r\|_\infty$ of the entries of the residual.

It suffices to discuss halting criteria for sparse signals because Lemma 3.2.18 shows that the general case can be viewed in terms of sampling a sparse signal. Let x be an s-sparse signal, and let a be an s-sparse approximation. The residual $r = x - a$. We write $v = \Phi r + e$ for the induced noisy samples of the residual and $y = \Phi^* v$ for the signal proxy.

The discussion proceeds in two steps. First, we argue that an *a priori* halting criterion will result in a guarantee about the quality of the final signal approximation.

Theorem 3.2.24 (Halting I)**.** *The halting criterion* $\|v\|_2 \leq \varepsilon$ *ensures that*

$$\|x - a\|_2 \leq 1.06 \cdot (\varepsilon + \|e\|_2).$$

The halting criterion $\|y\|_\infty \leq \eta/\sqrt{2s}$ *ensures that*

$$\|x - a\|_\infty \leq 1.12\eta + 1.17 \|e\|_2 .$$

Proof. Since r is $2s$-sparse, Proposition 3.2.4 ensures that

$$\sqrt{1 - \delta_{2s}} \|r\|_2 - \|e\|_2 \leq \|v\|_2 .$$

If $\|v\|_2 \leq \varepsilon$, it is immediate that

$$\|r\|_2 \leq \frac{\varepsilon + \|e\|_2}{\sqrt{1 - \delta_{2s}}}.$$

The definition $r = x - a$ and the numerical bound $\delta_{2s} \leq \delta_{4s} \leq 0.1$ dispatch the first claim.

Let $R = \text{supp} r$, and note that $|R| \leq 2s$. Proposition 3.2.4 results in

$$(1 - \delta_{2s}) \|r\|_2 - \sqrt{1 + \delta_{2s}} \|e\|_2 \leq \|y|_R\|_2 .$$

Since

$$\|y|_R\|_2 \leq \sqrt{2s} \|y|_R\|_\infty \leq \sqrt{2s} \|y\|_\infty ,$$

we find that the requirement $\|y\|_\infty \leq \eta/\sqrt{2s}$ ensures that

$$\|r\|_\infty \leq \frac{\eta + \sqrt{1 + \delta_{2s}} \|e\|_2}{1 - \delta_{2s}}.$$

The numerical bound $\delta_{2s} \leq 0.1$ completes the proof. □

Second, we check that each halting criterion is triggered when the residual has the desired property.

Theorem 3.2.25 (Halting II)**.** *The halting criterion* $\|v\|_2 \leq \varepsilon$ *is triggered as soon as*

$$\|x - a\|_2 \leq 0.95 \cdot (\varepsilon - \|e\|_2).$$

The halting criterion $\|y\|_\infty \leq \eta/\sqrt{2s}$ *is triggered as soon as*

$$\|x - a\|_\infty \leq \frac{0.45\eta}{s} - \frac{0.68\,\|e\|_2}{\sqrt{s}}.$$

Proof. Proposition 3.2.4 shows that

$$\|v\|_2 \leq \sqrt{1 + \delta_{2s}}\,\|r\|_2 + \|e\|_2\,.$$

Therefore, the condition

$$\|r\|_2 \leq \frac{\varepsilon - \|e\|_2}{\sqrt{1 + \delta_{2s}}}$$

ensures that $\|v\|_2 \leq \varepsilon$. Note that $\delta_{2s} \leq 0.1$ to complete the first part of the argument.

Now let R be the singleton containing the index of a largest-magnitude coefficient of y. Proposition 3.2.4 implies that

$$\|y\|_\infty = \|y|_R\|_2 \leq \sqrt{1 + \delta_1}\,\|v\|_2\,.$$

By the first part of this theorem, the halting criterion $\|y\|_\infty \leq \eta/\sqrt{2s}$ is triggered as

soon as

$$\|x - a\|_2 \leq 0.95 \cdot \left(\frac{\eta}{\sqrt{2s}\sqrt{1+\delta_1}} - \|e\|_2 \right).$$

Since $x - a$ is $2s$-sparse, we have the bound $\|x - a\|_2 \leq \sqrt{2s}\,\|x - a\|_\infty$. To wrap up, recall that $\delta_1 \leq \delta_{2s} \leq 0.1$. □

Next we discuss other variations of the algorithm.

Here is a version of the algorithm that is, perhaps, simpler than that described above. At each iteration, we approximate the current residual rather than the entire signal. This approach is similar to HHS pursuit [32]. The inner loop changes in the following manner.

Identification As before, select $\Omega = \text{supp} y_{2s}$.

Estimation Solve a least-squares problem with the *current samples* instead of the original samples to obtain an approximation of the *residual signal*. Formally, $b = \Phi_\Omega^\dagger v$. In this case, one initializes the iterative least-squares algorithm with the zero vector to take advantage of the fact that the residual is becoming small.

Approximation Merger Add this approximation of the residual to the previous approximation of the signal to obtain a new approximation of the signal: $c = a^{k-1} + b$.

Pruning Construct the s-sparse signal approximation: $a^k = c_s$.

Sample Update Update the samples as before: $v = u - \Phi a^k$.

By adapting the argument in this paper, we have been able to show that this algorithm also satisfies Theorem 3.2.1. We believe this version is quite promising for applications.

An alternative variation is as follows. After the inner loop of the algorithm is complete, we can solve another least-squares problem in an effort to improve the final result. If a is the approximation at the end of the loop, we set $T = \text{supp}a$. Then solve $b = \Phi_T^\dagger u$ and output the s-sparse signal approximation b. Note that the output approximation is not guaranteed to be better than a because of the noise vector e, but it should never be much worse.

Another variation is to prune the merged support T down to s entries *before* solving the least-squares problem. One may use the values of the proxy y as surrogates for the unknown values of the new approximation on the set Ω. Since the least-squares problem is solved at the end of the iteration, the columns of Φ that are used in the least-squares approximation are orthogonal to the current samples v. As a result, the identification step always selects new components in each iteration. We have not attempted an analysis of this algorithm.

3.2.5 Numerical Results

Noiseless Numerical Studies

This section describes our experiments that illustrate the signal recovery power of CoSaMP. See Section A.4 for the Matlab code used in these studies. We experimentally examine how many measurements m are necessary to recover various kinds of s-sparse signals in $\mathbb{R}^d$ using ROMP. We also demonstrate that the number of iterations CoSaMP needs to recover a sparse signal is in practice at most linear the sparsity, and in fact this serves as a successful halting criterion.

First we describe the setup of our experiments. For many values of the ambient dimension d, the number of measurements m, and the sparsity s, we reconstruct random signals using CoSaMP. For each set of values, we generate an $m \times d$ Gaussian

measurement matrix Φ and then perform 500 independent trials. The results we obtained using Bernoulli measurement matrices were very similar. In a given trial, we generate an s-sparse signal x in one of two ways. In either case, we first select the support of the signal by choosing s components uniformly at random (independent from the measurement matrix Φ). In the cases where we wish to generate flat signals, we then set these components to one. In the cases where we wish to generate sparse compressible signals, we set the i^{th} component of the support to plus or minus $i^{-1/p}$ for a specified value of $0 < p < 1$. We then execute CoSaMP with the measurement vector $u = \Phi x$.

Figure 3.2.2 depicts the percentage (from the 500 trials) of sparse flat signals that were reconstructed exactly. This plot was generated with $d = 256$ for various levels of sparsity s. The horizontal axis represents the number of measurements m, and the vertical axis represents the exact recovery percentage.

Figure 3.2.3 depicts a plot of the values for m and s at which 99% of sparse flat signals are recovered exactly. This plot was generated with $d = 256$. The horizontal axis represents the number of measurements m, and the vertical axis the sparsity level s.

Our results guarantee that CoSaMP reconstructs signals correctly with just $O(s)$ iterations. Figure 3.2.4 depicts the number of iterations needed by CoSaMP for $d = 10,000$ and $m = 200$ for perfect reconstruction. CoSaMP was executed under the same setting as described above for sparse flat signals as well as sparse compressible signals for various values of p, and the number of iterations in each scenario was averaged over the 500 trials. These averages were plotted against the sparsity of the signal. The plot demonstrates that often far fewer iterations are actually needed in some cases. This is not surprising, since as we discussed above alternative halting

criteria may be better in practice. The plot also demonstrates that the number of iterations needed for sparse compressible is higher than the number needed for sparse flat signals, as one would expect. The plot suggests that for smaller values of p (meaning signals that decay more rapidly) CoSaMP needs more iterations.

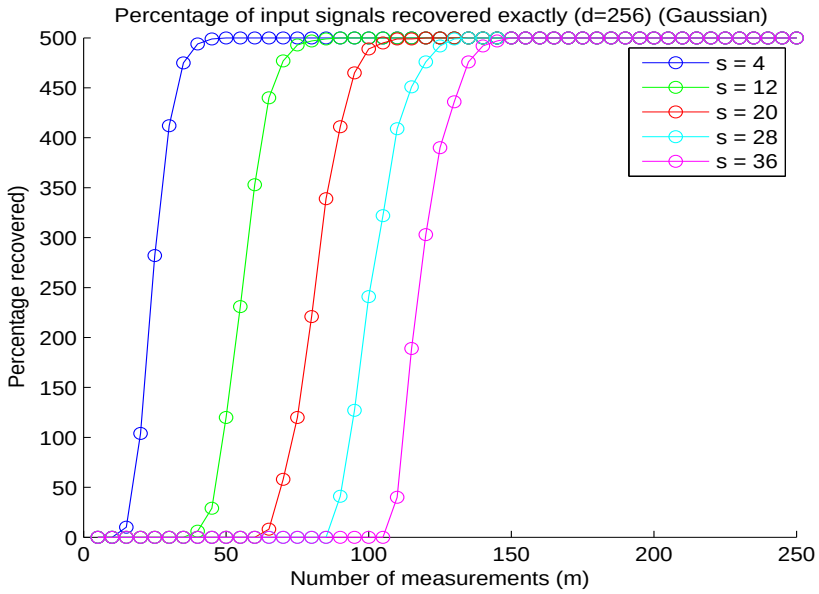


Figure 3.2.2: The percentage of sparse flat signals exactly recovered by CoSaMP as a function of the number of measurements in dimension $d = 256$ for various levels of sparsity.

Noisy Numerical Studies

This section describes our numerical experiments that illustrate the stability of CoSaMP. We study the recovery error using CoSaMP for both perturbed measurements and signals. The empirical recovery error confirms that given in the theorems.

First we describe the setup to our experimental studies. We run CoSaMP on various values of the ambient dimension d, the number of measurements m, and the sparsity level s, and attempt to reconstruct random signals. For each set of param-

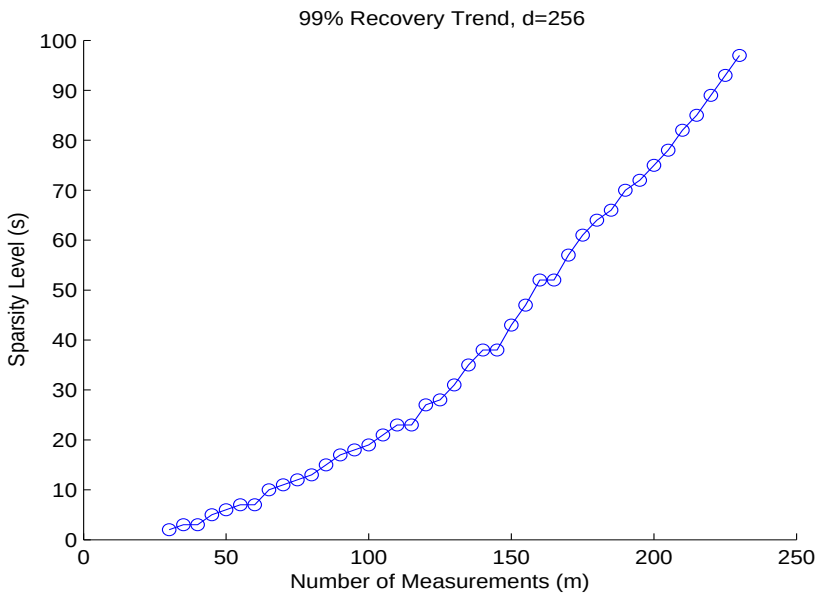


Figure 3.2.3: The 99% recovery limit as a function of the sparsity and the number of measurements for sparse flat signals.

eters, we perform 500 trials. Initially, we generate an $m \times d$ Gaussian measurement matrix Φ. For each trial, independent of the matrix, we generate an s-sparse signal x by choosing s components uniformly at random and setting them to one. In the case of perturbed signals, we add to the signal a d-dimensional error vector with Gaussian entries. In the case of perturbed measurements, we add an m-dimensional error vector with Gaussian entries to the measurement vector Φx. We then execute ROMP with the measurement vector $u = \Phi x$ or $u + e$ in the perturbed measurement case. After CoSaMP terminates (using a fixed number of iterations of $10s$), we output the reconstructed vector $\hat{x}$ obtained from the least squares calculation and calculate its distance from the original signal.

Figure 3.2.5 depicts the recovery error $\|x - \hat{x}\|_2$ when CoSaMP was run with perturbed measurements. This plot was generated with $d = 256$ for various levels of sparsity s. The horizontal axis represents the number of measurements m, and the

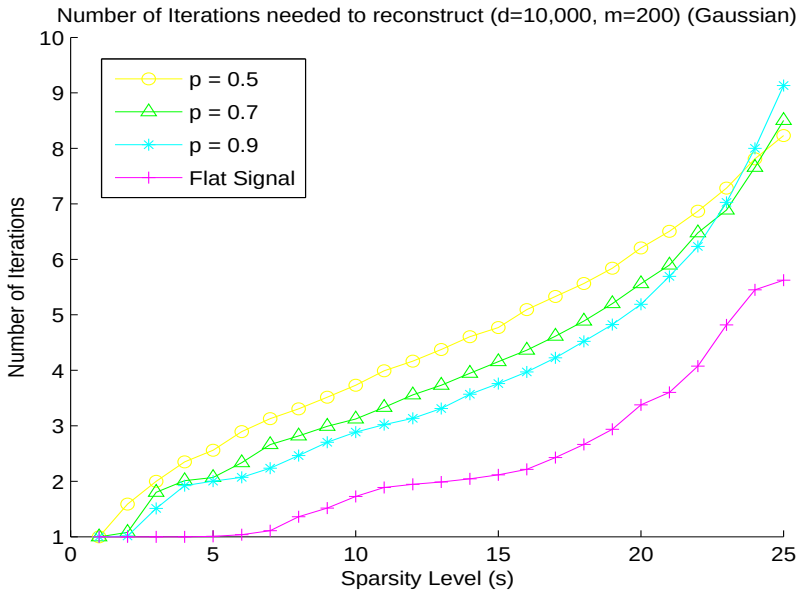


Figure 3.2.4: The number of iterations needed by CoSaMP as a function of the sparsity in dimension $d = 10,000$ with 200 measurements.

vertical axis represents the average normalized recovery error.

Figure 3.2.6 depicts the normalized recovery error when the signal was perturbed by a Gaussian vector. Again these results are consistent with our proven theorems, but notice that we normalize here by $\|x - x_s\|_1/\sqrt{s}$ rather than $\|x - x_{s/2}\|_1/\sqrt{s}$ as our theorems suggest. These plots show that perhaps the former normalization factor is actually more accurate, and the latter may be a consequence of our analysis only.

3.2.6 Summary

CoSaMP draws on both algorithmic ideas and analytic techniques that have appeared before. Here we summarize the results in the context of other work. This discussion can also be found in [53]. The initial discovery works on compressive sampling proposed to perform signal recovery by solving a convex optimization problem [4, 19] (see also Section 2.1 above). Given a sampling matrix Φ and a noisy vector of sam-

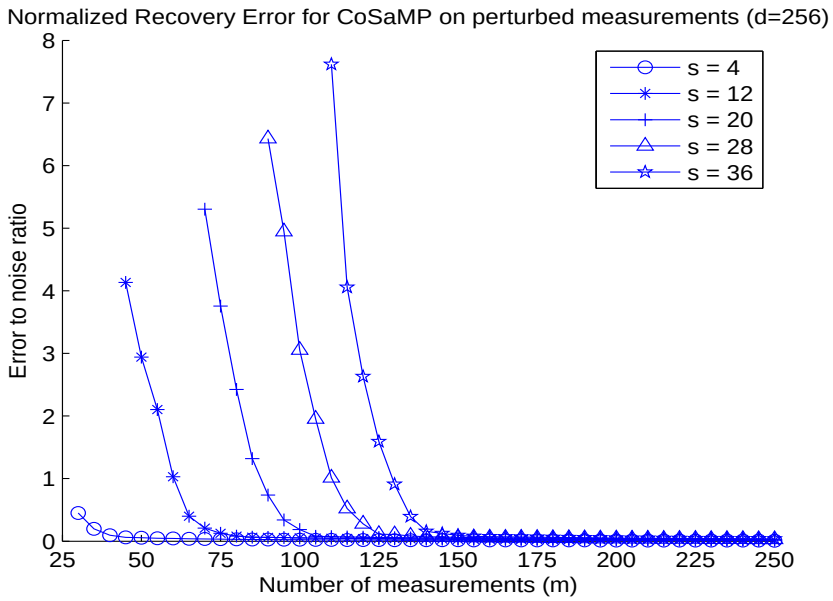


Figure 3.2.5: The error to noise ratio $\frac{\|\hat{x}-x\|_2}{\|e\|_2}$ as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

ples $u = \Phi x + e$ with $\|e\|_2 \leq \varepsilon$, we consider the mathematical program (2.1.5). In words, we look for a signal reconstruction that is consistent with the samples but has minimal ℓ_1 norm. The intuition behind this approach is that minimizing the ℓ_1 norm promotes sparsity, so allows the approximate recovery of compressible signals. Candès, Romberg, and Tao established in [5] that a minimizer a of (2.1.5) satisfies

$$\|x - a\|_2 \leq \mathrm{C} \left[\frac{1}{\sqrt{s}} \|x - x_s\|_1 + \varepsilon \right] \tag{3.2.10}$$

provided that the sampling matrix Φ has restricted isometry constant $\delta_{4s} \leq 0.2$. In [8], the hypothesis on the restricted isometry constant is sharpened to $\delta_{2s} \leq \sqrt{2} - 1$. The error bound for CoSaMP is equivalent, modulo the exact value of the constants.

The literature describes a huge variety of algorithms for solving the optimization problem (2.1.5). The most common approaches involve interior-point methods [4,

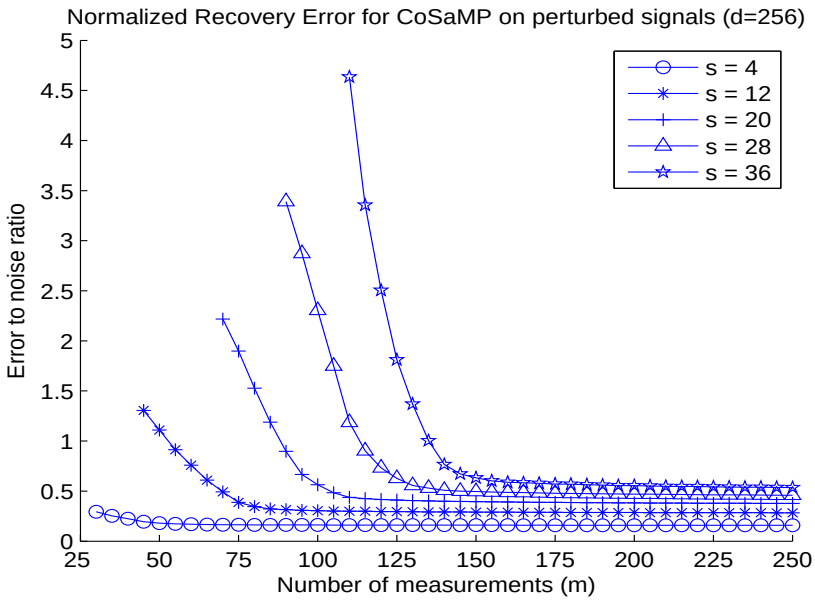


Figure 3.2.6: The error to noise ratio $\frac{\|\hat{x}-x_s\|_2}{\|x-x_s\|_1/\sqrt{s}}$ using a perturbed signal, as a function of the number of measurements m in dimension $d = 256$ for various levels of sparsity s.

43], projected gradient methods [26], or iterative thresholding [16] The interior-point methods are guaranteed to solve the problem to a fixed precision in time $\mathrm{O}(m^2 d^{1.5})$, where m is the number of measurements and d is the signal length [56]. Note that the constant in the big-O notation depends on some of the problem data. The other convex relaxation algorithms, while sometimes faster in practice, do not currently offer rigorous guarantees. CoSaMP provides rigorous bounds on the runtime that are much better than the available results for interior-point methods.

Tropp and Gilbert proposed the use of a greedy iterative algorithm called *orthogonal matching pursuit* (OMP) for signal recovery [62] (see also Section 2.2.1 above). Tropp and Gilbert were able to prove a weak result for the performance of OMP [62]. Suppose that x is a fixed, s-sparse signal, and let $m = \mathrm{C}s \log s$. Draw an $m \times s$ sampling matrix Φ whose entries are independent, zero-mean subgaussian random variables with equal variances. Given noiseless measurements $u = \Phi x$, OMP recon-

structs x after s iterations, except with probability s^{-1}. In this setting, OMP must fail for some sparse signals[57], so it does not provide the same uniform guarantees as convex relaxation. It is unknown whether OMP succeeds for compressible signals or whether it succeeds when the samples are contaminated with noise.

Donoho et al. invented another greedy iterative method called *stagewise OMP*, or StOMP [23] (see also Section 2.2.2 above). This algorithm uses the signal proxy to select multiple components at each step, using a rule inspired by ideas from wireless communications. The algorithm is faster than OMP because of the selection rule, and it sometimes provides good performance, although parameter tuning can be difficult. There are no rigorous results available for StOMP.

Needell and Vershynin developed and analyzed another greedy approach, called *regularized OMP*, or ROMP [55, 54] (see also Section 3.1 above). The work on ROMP represents an advance because the authors establish under restricted isometry hypotheses that their algorithm can approximately recover any compressible signal from noisy samples. More precisely, suppose that the sampling matrix Φ has restricted isometry constant $\delta_{8s} \leq 0.01/\sqrt{\log s}$. Given noisy samples $u = \Phi x + e$, ROMP produces a $2s$-sparse signal approximation a that satisfies

$$\|x - a\|_2 \leq \mathrm{C}\sqrt{\log s}\left[\frac{1}{\sqrt{s}}\|x - x_s\|_1 + \|e\|_2\right].$$

This result is comparable with the result for convex relaxation, aside from the extra logarithmic factor in the restricted isometry hypothesis and the error bound. The results for CoSaMP show that it does not suffer these parasitic factors, so its performance is essentially optimal.

3.3 Reweighted L1-Minimization

As discussed in Section 2.1, the ℓ_1-minimization problem (2.1.1) is equivalent to the nonconvex problem (1.1.2) when the measurement matrix Φ satisfies a certain condition. Let us first state the best known theorem for recovery using ℓ_1, provided by Candès, in more detail. We will rely on this result to prove the new theorems.

Theorem 3.3.1 (ℓ_1-minimization from [8]). *Assume Φ has $\delta_{2s} < \sqrt{2} - 1$. Let x be an arbitrary signal with noisy measurements $\Phi x + e$, where $\|e\|_2 \leq \varepsilon$. Then the approximation $\hat{x}$ to x from ℓ_1-minimization satisfies*

$$\|x - \hat{x}\|_2 \leq C\varepsilon + C' \frac{\|x - x_s\|_1}{\sqrt{s}},$$

where $C = \frac{2\alpha}{1-\rho}$, $C' = \frac{2(1+\rho)}{1-\rho}$, $\rho = \frac{\sqrt{2}\delta_{2s}}{1-\delta_{2s}}$, and $\alpha = \frac{2\sqrt{1+\delta_{2s}}}{\sqrt{1-\delta_{2s}}}$.

The key difference between the two problems of course, is that the ℓ_1 formulation depends on the magnitudes of the coefficients of a signal, whereas the ℓ_0 does not. To reconcile this imbalance, a new weighted ℓ_1-minimization algorithm was proposed by Candès, Wakin, and Boyd [7]. This algorithm solves the following weighted version of (L_1) at each iteration:

$$\min_{\hat{x} \in \mathbb{R}^d} \sum_{i=1}^{d} \delta_i \hat{x}_i \text{ subject to } \Phi x = \Phi \hat{x}. \qquad (WL_1)$$

It is clear that in this formulation, large weights δ_i will encourage small coordinates of the solution vector, and small weights will encourage larger coordinates. Indeed, suppose the s-sparse signal x was known exactly, and that the weights were set as

$$\delta_i = \frac{1}{|x_i|}. \qquad (3.3.1)$$

Notice that in this case, the weights are infinite at all locations outside of the support of x. This will force the coordinates of the solution vector $\hat{x}$ at these locations to be zero. Thus if the signal x is s-sparse with $s \leq m$, these weights would guarantee that $\hat{x} = x$. Of course, these weights could not be chosen without knowing the actual signal x itself, but this demonstrates the positive effect that the weights can have on the performance of ℓ_1-minimization.

The helpful effect of the weights can also be viewed geometrically. Recall that the solution to the problem (L_1) is the contact point where the smallest ℓ_1-ball meets the subspace $x + \ker \Phi$. When the solution to (L_1) does not coincide with the original signal x, it is because there is an ℓ_1-ball smaller than the one containing x, which meets the subspace $x + \ker \Phi$. The solution to problem (WL_1), however, is the place where the *weighted* ℓ_1-ball meets the subspace. When the weights are appropriate, this is an ℓ_1-ball that has been pinched toward the signal x (see Figure 3.3.1). This new geometry reduces the likelihood of the incorrect solution.

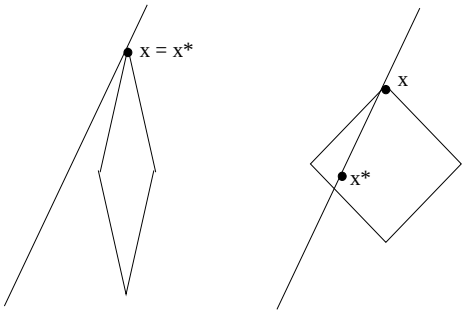


Figure 3.3.1: Weighted ℓ_1-ball geometry (right) versus standard (left).

Although the weights might not initially induce this geometry, one hopes that by solving the problem (WL_1) at each iteration, the weights will get closer to the optimal values (3.3.1), thereby improving the reconstruction of x. Of course, one

cannot actually have an infinite weight as in (3.3.1), so a stability parameter must also be used in the selection of the weight values. The reweighted ℓ_1-minimization algorithm can thus be described precisely as follows.

REWEIGHTED ℓ_1-MINIMIZATION

INPUT: Measurement vector $u \in \mathbb{R}^m$, stability parameter a

OUTPUT: Reconstructed vector $\hat{x}$

Initialize Set the weights $\delta_i = 1$ for $i = 1 \dots d$.

Repeat the following until convergence or a fixed number of times:

Approximate Solve the reweighted ℓ_1-minimization problem:

$$\hat{x} = \underset{\hat{x} \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{d} \delta_i \hat{x}_i \text{ subject to } u = \Phi\hat{x} \text{ (or } \|\Phi\hat{x} - u\|_2 \leq \varepsilon).$$

Update Reset the weights:

$$\delta_i = \frac{1}{|\hat{x}_i| + a}.$$

Remark 3.3.2. *Note that the optional set of constraints given in the algorithm is only for the case in which the signal or measurements may be corrupted with noise. It may also be advantageous to decrease the stability parameter a so that $a \to 0$ as the iterations tend to infinity. See the proof of Theorem 3.3.3 below for details.*

In [7], the reweighted ℓ_1-minimization algorithm is discussed thoroughly, and experimental results are provided to show that it often outperforms the standard method. However, no provable guarantees have yet been made for the algorithm's success. Here we analyze the algorithm when the measurements and signals are corrupted with noise. Since the reweighted method needs a weight vector that is

somewhat close to the actual signal x, it is natural to consider the noisy case since the standard ℓ_1-minimization method itself produces such a vector. We are able to prove an error bound in this noisy case that improves upon the best known bound for the standard method. We also provide numerical studies that show the bounds are improved in practice as well.

3.3.1 Main Results

The main theorem of this paper guarantees an error bound for the reconstruction using reweighted ℓ_1-minimization that improves upon the best known bound of Theorem 3.3.1 for the standard method. For initial simplicity, we consider the case where the signal x is exactly sparse, but the measurements u are corrupted with noise. Our main theorem, Theorem 3.3.3 will imply results for the case where the signal x is arbitrary.

Theorem 3.3.3 (Reweighted ℓ_1, Sparse Case)**.** *Assume Φ satisfies the restricted isometry condition with parameters $(2s, \delta)$ where $\delta < \sqrt{2} - 1$. Let x be an s-sparse vector with noisy measurements $u = \Phi x + e$ where $\|e\|_2 \leq \varepsilon$. Assume the smallest nonzero coordinate μ of x satisfies $\mu \geq \frac{4\alpha\varepsilon}{1-\rho}$. Then the limiting approximation from reweighted ℓ_1-minimization satisfies*

$$\|x - \hat{x}\|_2 \leq C''\varepsilon,$$

where $C'' = \frac{2\alpha}{1+\rho}$, $\rho = \frac{\sqrt{2}\delta}{1-\delta}$ and $\alpha = \frac{2\sqrt{1+\delta}}{1-\delta}$.

Remarks.

1. We actually show that the reconstruction error satisfies

$$\|x - \hat{x}\|_2 \leq \frac{2\alpha\varepsilon}{1 + \sqrt{1 - \frac{4\alpha\varepsilon}{\mu} - \frac{4\alpha\varepsilon\rho}{\mu}}}. \tag{3.3.2}$$

This bound is stronger than that given in Theorem 3.3.3, which is only equal to this bound when μ nears the value $\frac{4\alpha\varepsilon}{1-\rho}$. However, the form in Theorem 3.3.3 is much simpler and clearly shows the role of the parameter δ by the use of ρ.

2. For signals whose smallest non-zero coefficient μ does not satisfy the condition of the theorem, we may apply the theorem to those coefficients that do satisfy this requirement, and treat the others as noise. See Theorem 3.3.4 below.

3. Although the bound in the theorem is the *limiting* bound, we provide a recursive relation (3.3.9) in the proof which provides an exact error bound per iteration. In Section 3.3.3 we use dynamic programming to show that in many cases only a very small number of iterations are actually required to obtain the above error bound.

We now discuss the differences between Theorem 3.3.1 and our new result Theorem 3.3.3. In the case where δ nears its limit of $\sqrt{2}-1$, the constant ρ increases to 1, and so the constant C in Theorem 3.3.1 is unbounded. However, the constant C'' in Theorem 3.3.3 remains bounded even in this case. In fact, as δ approaches $\sqrt{2}-1$, the constant C'' approaches just 4.66. The tradeoff of course, is in the requirement on μ. As δ gets closer to $\sqrt{2}-1$, the bound needed on μ requires the signal to have unbounded non-zero coordinates relative to the noise level ε. However, to use this theorem efficiently, one would select the largest $\delta < \sqrt{2}-1$ that allows the requirement on μ to be satisfied, and then apply the theorem for this value of δ. Using this strategy, when the ratio $\frac{\mu}{\varepsilon} = 10$, for example, the error bound is just 3.85ε.

Theorem 3.3.3 and a short calculation will imply the following result for *arbitrary*

signals x.

Theorem 3.3.4 (Reweighted ℓ_1)**.** *Assume Φ satisfies the restricted isometry condition with parameters $(2s, \sqrt{2}-1)$. Let x be an arbitrary vector with noisy measurements $u = \Phi x + e$ where $\|e\|_2 \leq \varepsilon$. Assume the smallest nonzero coordinate μ of x_s satisfies $\mu \geq \frac{4\alpha\varepsilon_0}{1-\rho}$, where $\varepsilon_0 = 1.2(\|x - x_s\|_2 + \frac{1}{\sqrt{s}}\|x - x_s\|_1) + \varepsilon$. Then the limiting approximation from reweighted ℓ_1-minimization satisfies*

$$\|x - \hat{x}\|_2 \leq \frac{4.1\alpha}{1+\rho}\Big(\frac{\|x - x_{s/2}\|_1}{\sqrt{s}} + \varepsilon\Big),$$

and

$$\|x - \hat{x}\|_2 \leq \frac{2.4\alpha}{1+\rho}\Big(\|x - x_s\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}} + \varepsilon\Big),$$

where ρ and α are as in Theorem 3.3.3.

Again in the case where δ nears its bound of $\sqrt{2}-1$, both constants C and C' in Theorem 3.3.1 approach infinity. However, in Theorem 3.3.4, the constant remains bounded even in this case. The same strategy discussed above for Theorem 3.3.3 should also be used for Theorem 3.3.4. Next we begin proving Theorem 3.3.3 and Theorem 3.3.4.

3.3.2 Proofs

We will first utilize a lemma that bounds the ℓ_2 norm of a small portion of the difference vector $x - \hat{x}$ by the ℓ_1-norm of its remainder. This lemma is proved in [8] and essentially in [5] as part of the proofs of the main theorems of those papers. We include a proof here as well since we require the final result as well as intermediate steps for the proof of our main theorem.

Lemma 3.3.5. *Set $h = \hat{x} - x$, and let α, ε, and ρ be as in Theorem 3.3.3. Let T_0 be the set of s largest coefficients in magnitude of x and T_1 be the s largest coefficients of $h_{T_0^c}$. Then*

$$\|h_{T_0 \cup T_1}\|_2 \leq \alpha\varepsilon + \frac{\rho}{\sqrt{s}}\|h_{T_0^c}\|_1.$$

Proof. Continue defining sets T_j by setting T_1 to be the s largest coefficients of $h_{T_0^c}$, T_2 the next s largest coefficients of $h_{T_0^c}$, and so on. We begin by applying the triangle inequality and using the fact that x itself is feasible in (WL_1). This yields

$$\|\Phi h\|_2 \leq \|\Phi\hat{x} - u\|_2 + \|\Phi x - u\|_2 \leq 2\varepsilon. \tag{3.3.3}$$

By the decreasing property of these sets and since each set T_j has cardinality at most s, we have for each $j \geq 2$,

$$\|h_{T_j}\|_2 \leq \sqrt{s}\|h_{T_j}\|_\infty \leq \frac{1}{\sqrt{s}}\|h_{T_{j-1}}\|_1.$$

Summing the terms, this gives

$$\sum_{j \geq 2} \|h_{T_j}\|_2 \leq \frac{1}{\sqrt{s}}\|h_{T_0^c}\|_1. \tag{3.3.4}$$

By the triangle inequality, we then also have

$$\|h_{(T_0 \cup T_1)^c}\|_2 \leq \frac{1}{\sqrt{s}}\|h_{T_0^c}\|_1. \tag{3.3.5}$$

Now by linearity we have

$$\|\Phi h_{T_0 \cup T_1}\|_2^2 = \langle \Phi h_{T_0 \cup T_1}, \Phi h \rangle - \langle \Phi h_{T_0 \cup T_1}, \sum_{j \geq 2} \Phi h_{T_j} \rangle.$$

By (3.3.3) and the restricted isometry condition, we have

$$|\langle \Phi h_{T_0 \cup T_1}, \Phi h \rangle| \leq \|\Phi h_{T_0 \cup T_1}\|_2 \|\Phi h\|_2 \leq 2\varepsilon \sqrt{1+\delta} \|h_{T_0 \cup T_1}\|_2.$$

As shown in Lemma 2.1 of [8], the restricted isometry condition and parallelogram law imply that for $j \geq 2$,

$$|\langle \Phi h_{T_0}, \Phi h_{T_j} \rangle| \leq \delta \|\Phi h_{T_0}\|_2 \|\Phi h_{T_j}\|_2 \text{ and } |\langle \Phi h_{T_1}, \Phi h_{T_j} \rangle| \leq \delta \|\Phi h_{T_1}\|_2 \|\Phi h_{T_j}\|_2.$$

Since all sets T_j are disjoint, the above three inequalities yield

$$(1-\delta)\|h_{T_0 \cup T_1}\|_2^2 \leq \|\Phi h_{T_0 \cup T_1}\|_2^2 \leq \|h_{T_0 \cup T_1}\|_2 (2\varepsilon \sqrt{1+\delta} + \sqrt{2}\delta \sum_{j \geq 2} \|h_{T_j}\|_2).$$

Therefore by (3.3.4), we have

$$\|h_{T_0 \cup T_1}\|_2 \leq \alpha \varepsilon + \frac{\rho}{\sqrt{s}} \|h_{T_0^c}\|_1.$$

□

We will next require two lemmas that give results about a single iteration of reweighted ℓ_1-minimization.

Lemma 3.3.6 (Single reweighted ℓ_1-minimization)**.** *Assume Φ satisfies the restricted isometry condition with parameters $(2s, \sqrt{2}-1)$. Let x be an arbitrary vector with noisy measurements $u = \Phi x + e$ where $\|e\|_2 \leq \varepsilon$. Let w be a vector such that $\|w - x\|_\infty \leq A$ for some constant A. Denote by x_s the vector consisting of the s (where $s \leq |\text{supp}(x)|$) largest coefficients of x in absolute value. Let μ be the smallest coordinate of x_s in absolute value, and set $b = \|x - x_s\|_\infty$. Then when*

$\mu \geq A$ and $\rho C_1 < 1$, the approximation from reweighted ℓ_1-minimization using weights $\delta_i = 1/(w_i + a)$ satisfies

$$\|x - \hat{x}\|_2 \leq D_1 \varepsilon + D_2 \frac{\|x - x_s\|_1}{a},$$

where $D_1 = \frac{(1+C_1)\alpha}{1-\rho C_1}$, $D_2 = C_2 + \frac{(1+C_1)\rho C_2}{1-\rho C_1}$, $C_1 = \frac{A+a+b}{\mu - A + a}$, $C_2 = \frac{2(A+a+b)}{\sqrt{s}}$, and ρ and α are as in Theorem 3.3.3.

Proof of Lemma 3.3.6. Set h and T_j for $j \geq 0$ as in Lemma 3.3.5. For simplicity, denote by $\|\cdot\|_w$ the weighted ℓ_1-norm:

$$\|z\|_w \stackrel{\text{def}}{=} \sum_{i=1}^{d} \frac{1}{|w_i| + a} z_i.$$

Since $\hat{x} = x + h$ is the minimizer of (WL_1), we have

$$\|x\|_w \geq \|x+h\|_w = \|(x+h)_{T_0}\|_w + \|(x+h)_{T_0^c}\|_w \geq \|x_{T_0}\|_w - \|h_{T_0}\|_w + \|h_{T_0^c}\|_w - \|x_{T_0^c}\|_w.$$

This yields

$$\|h_{T_0^c}\|_w \leq \|h_{T_0}\|_w + 2\|x_{T_0^c}\|_w.$$

Next we relate the reweighted norm to the usual ℓ_1-norm. We first have

$$\|h_{T_0^c}\|_w \geq \frac{\|h_{T_0^c}\|_1}{A + a + b},$$

by definition of the reweighted norm as well as the values of A, a, and b. Similarly we have

$$\|h_{T_0}\|_w \leq \frac{\|h_{T_0}\|_1}{\mu - A + a}.$$

Combining the above three inequalities, we have

$$\begin{aligned}\|h_{T_0^c}\|_1 \leq (A+a+b)\|h_{T_0^c}\|_w &\leq (A+a+b)(\|h_{T_0}\|_w + 2\|x_{T_0^c}\|_w) \\ &\leq \frac{A+a+b}{\mu - A + a}\|h_{T_0}\|_1 + 2(A+a+b)\|x_{T_0^c}\|_w. \qquad (3.3.6)\end{aligned}$$

Using (3.3.5) and (3.3.6) along with the fact $\|h_{T_0}\|_1 \leq \sqrt{s}\|h_{T_0}\|_2$, we have

$$\|h_{(T_0 \cup T_1)^c}\|_2 \leq C_1\|h_{T_0}\|_2 + C_2\|x_{T_0^c}\|_w, \qquad (3.3.7)$$

where $C_1 = \frac{A+a+b}{\mu - A + a}$ and $C_2 = \frac{2(A+a+b)}{\sqrt{s}}$. By Lemma 3.3.5, we have

$$\|h_{T_0 \cup T_1}\|_2 \leq \alpha\varepsilon + \frac{\rho}{\sqrt{s}}\|h_{T_0^c}\|_1,$$

where $\rho = \frac{\sqrt{2}\delta_{2s}}{1-\delta_{2s}}$ and $\alpha = \frac{2\sqrt{1+\delta_{2s}}}{\sqrt{1-\delta_{2s}}}$. Thus by (3.3.6), we have

$$\|h_{T_0 \cup T_1}\|_2 \leq \alpha\varepsilon + \frac{\rho}{\sqrt{s}}(C_1\|h_{T_0}\|_1 + 2(A+a+b)\|x_{T_0^c}\|_w) = \alpha\varepsilon + \rho C_1\|h_{T_0 \cup T_1}\|_2 + \rho C_2\|x_{T_0^c}\|_w.$$

Therefore,

$$\|h_{T_0 \cup T_1}\|_2 \leq (1 - \rho C_1)^{-1}(\alpha\varepsilon + \rho C_2\|x_{T_0^c}\|_w). \qquad (3.3.8)$$

Finally by (3.3.7) and (3.3.8),

$$\begin{aligned}\|h\|_2 &\leq \|h_{T_0 \cup T_1}\|_2 + \|h_{(T_0 \cup T_1)^c}\|_2 \\ &\leq (1 + C_1)\|h_{T_0 \cup T_1}\|_2 + C_2\|x_{T_0^c}\|_w \\ &\leq (1 + C_1)((1 - \rho C_1)^{-1}(\alpha\varepsilon + \rho C_2\|x_{T_0^c}\|_w)) + C_2\|x_{T_0^c}\|_w.\end{aligned}$$

Applying the inequality $\|x_{T_0^c}\|_w \leq (1/a)\|x_{T_0^c}\|_1$ and simplifying completes the claim.

□

Lemma 3.3.7 (Single reweighted ℓ_1-minimization, Sparse Case)**.** *Assume Φ satisfies the restricted isometry condition with parameters $(2s, \sqrt{2}-1)$. Let x be an s-sparse vector with noisy measurements $u = \Phi x + e$ where $\|e\|_2 \leq \varepsilon$. Let w be a vector such that $\|w - x\|_\infty \leq A$ for some constant A. Let μ be the smallest non-zero coordinate of x in absolute value. Then when $\mu \geq A$, the approximation from reweighted ℓ_1-minimization using weights $\delta_i = 1/(w_i + a)$ satisfies*

$$\|x - \hat{x}\|_2 \leq D_1 \varepsilon.$$

Here $D_1 = \frac{(1+C_1)\alpha}{1-\rho C_1}$, $C_1 = \frac{A+a}{\mu - A + a}$, and α and ρ are as in Theorem 3.3.3.

Proof. This is the case of Lemma 3.3.6 where $x - x_s = 0$ and $b = 0$. □

Proof of Theorem 3.3.3. The proof proceeds as follows. First, we use the error bound in Theorem 3.3.1 as the initial error, and then apply Lemma 3.3.7 repeatedly. We show that the error decreases at each iteration, and then deduce its limiting bound using the recursive relation. To this end, let $E(k)$ for $k = 1, \ldots,$ be the error bound on $\|x - \hat{x}_k\|_2$ where $\hat{x}_k$ is the reconstructed signal at the k^{th} iteration. Then by Theorem 3.3.1 and Lemma 3.3.7, we have the recursive definition

$$E(1) = \frac{2\alpha}{1-\rho}\varepsilon, \quad E(k+1) = \frac{(1 + \frac{E(k)}{\mu - E(k)})\alpha}{1 - \rho\frac{E(k)}{\mu - E(k)}}\varepsilon. \tag{3.3.9}$$

Here we have taken $a \to 0$ iteratively (or if a remains fixed, a small constant $O(a)$ will be added to the error). First, we show that the base case holds, $E(1) \leq E(2)$.

Since $\mu \geq \frac{4\alpha\varepsilon}{1-\rho}$, we have that

$$\frac{E(1)}{\mu - E(1)} = \frac{\frac{2\alpha\varepsilon}{1-\rho}}{\mu - \frac{2\alpha\varepsilon}{1-\rho}} \leq 1.$$

Therefore we have

$$E(2) = \frac{(1 + \frac{E(1)}{\mu - E(1)})\alpha}{1 - \rho\frac{E(1)}{\mu - E(1)}}\varepsilon \leq \frac{2\alpha}{1-\rho}\varepsilon = E(1).$$

Next we show the inductive step, that $E(k+1) \leq E(k)$ assuming the inequality holds for all previous k. Indeed, if $E(k) \leq E(k-1)$, then we have

$$E(k+1) = \frac{(1 + \frac{E(k)}{\mu - E(k)})\alpha}{1 - \rho\frac{E(k)}{\mu - E(k)}}\varepsilon \leq \frac{(1 + \frac{E(k-1)}{\mu - E(k-1)})\alpha}{1 - \rho\frac{E(k-1)}{\mu - E(k-1)}}\varepsilon = E(k).$$

Since $\mu \geq \frac{4\alpha\varepsilon}{1-\rho}$ and $\rho \leq 1$ we have that $\mu - E(k) \geq 0$ and $\rho\frac{E(k)}{\mu - E(k)} \leq 1$, so $E(k)$ is also bounded below by zero. Thus E(k) is a bounded decreasing sequence, so it must converge. Call its limit L. By the recursive definition of $E(k)$, we must have

$$L = \frac{(1 + \frac{L}{\mu - L})\alpha}{1 - \rho\frac{L}{\mu - L}}\varepsilon.$$

Solving this equation yields

$$L = \frac{\mu - \sqrt{\mu^2 - 4\mu\alpha\varepsilon - 4\mu\alpha\varepsilon\rho}}{2(1+\rho)},$$

where we choose the solution with the minus since $E(k)$ is decreasing and $E(1) < \mu/2$ (note also that $L = 0$ when $\varepsilon = 0$). Multiplying by the conjugate and simplifying

yields

$$L = \frac{4\mu\alpha\varepsilon + 4\mu\alpha\varepsilon\rho}{2(1+\rho)(\mu + \sqrt{\mu^2 - 4\mu\alpha\varepsilon - 4\mu\alpha\varepsilon\rho})} = \frac{2\alpha\varepsilon}{1 + \sqrt{1 - \frac{4\alpha\varepsilon}{\mu} - \frac{4\alpha\varepsilon\rho}{\mu}}}.$$

Then again since $\mu \geq \frac{4\alpha\varepsilon}{1-\rho}$, we have

$$L \leq \frac{2\alpha\varepsilon}{1+\rho}.$$

□

Proof of Theorem 3.3.4. By Lemma 6.1 of [53] and Lemma 7 of [32], we can rewrite $\Phi x + e$ as $\Phi x_s + \widetilde{e}$ where

$$\|\widetilde{e}\|_2 \leq 1.2(\|x - x_s\|_2 + \frac{1}{\sqrt{s}}\|x - x_s\|_1) + \|e\|_2 \leq 2.04\Big(\frac{\|x - x_{s/2}\|_1}{\sqrt{s}}\Big) + \|e\|_2.$$

This combined with Theorem 3.3.3 completes the claim. □

3.3.3 Numerical Results and Convergence

Our main theorems prove bounds on the reconstruction error limit. However, as is the case with many recursive relations, convergence to this threshold is often quite fast. To show this, we use dynamic programming to compute the theoretical error bound $E(k)$ given by (3.3.9) and test its convergence rate to the threshold given by eqrefactualbnd. Since the ratio between μ and ε is important, we fix $\mu = 10$ and test the convergence for various values of ε and δ. We conclude that the error bound has achieved the threshold when it is within 0.001 of it. The results are displayed in Figure 3.3.2.

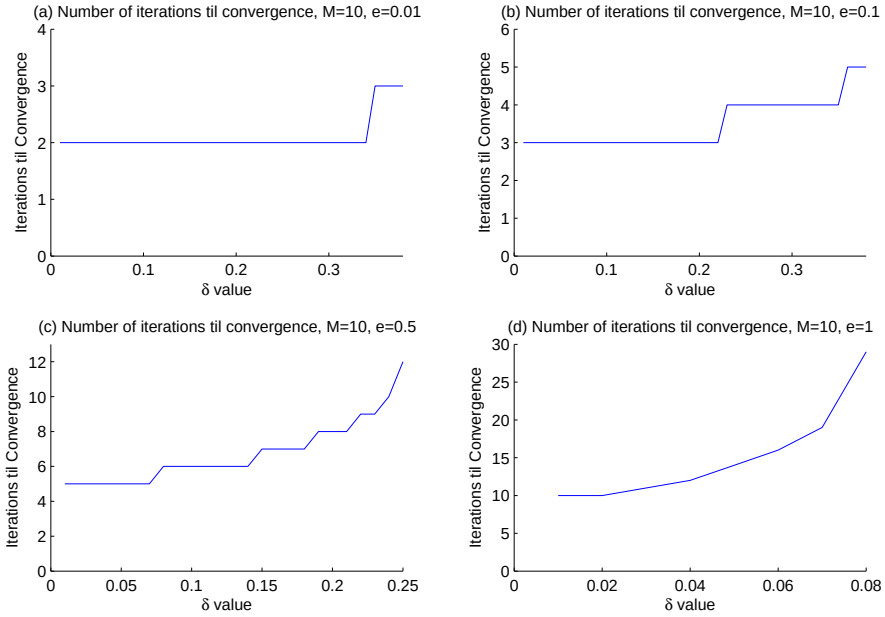


Figure 3.3.2: The number of iterations required for the theoretical error bounds eqrefEk to reach the theoretical error threshold (3.3.2) when (a) $\mu = 10$, $\varepsilon = 0.01$, (b) $\mu = 10$, $\varepsilon = 0.1$, (c) $\mu = 10$, $\varepsilon = 0.5$, (d) $\mu = 10$, $\varepsilon = 1.0$.

We observe that in each case, as δ increases we require slightly more iterations. This is not surprising since higher δ means a lower bound. We also confirm that less iterations are required when the ratio μ/ε is smaller.

Next we examine some numerical experiments conducted to test the actual error with reweighted ℓ_1-minimization versus the standard ℓ_1 method. In these experiments we consider signals of dimension $d = 256$ with $s = 30$ non-zero entries. We use a 128×256 measurement matrix Φ consisting of Gaussian entries. We note that we found similar results when the measurement matrix Φ consisted of symmetric Bernoulli entries. Sparsity, measurement, and dimension values in this range demonstrate a good example of the advantages of the reweighted method. Our results above suggest that improvements are made using the reweighted method when δ cannot be forced very small. This means that in situations with sparsity levels s much smaller

or measurement levels m much larger, the reweighted method may not offer much improvements. These levels are not the only ones that show improvements, however, see for example those experiments in [7].

For each trial in our experiments we construct an s-sparse signal x with uniformly chosen support and entries from either the Gaussian distribution or the symmetric Bernoulli distribution. All entries are chosen independently and independent of the measurement matrix Φ. We then construct the noise vector e to have independent Gaussian entries, and normalize the vector to have norm a fraction $(1/5)$ of the noiseless measurement vector Φx norm. We then run the reweighted ℓ_1-algorithm using ε such that $\varepsilon^2 = \sigma^2(m + 2\sqrt{2m})$ where σ^2 is the variance of the normalized error vectors. This value is likely to provide a good upper bound on the noise norm (see e.g. [5], [7]). The stability parameter a tends to zero with increased iterations. We run 500 trials for each parameter selection and signal type. We found similar results for non-sparse signals such as noisy sparse signals and compressible signals. This is not surprising since we can treat the signal error as measurement error after applying the the measurement matrix (see the proof of Theorem 3.3.4).

Figure 3.3.3 depicts the experiment with Gaussian signals and decreasing stability parameter a. In particular, we set $a = 1/(1000k)$ in the k^{th} iteration. The plot (left) depicts the error after a single ℓ_1-minimization and after 9 iterations using the reweighted method. The histogram (right) depicts the improvements $\|x - \hat{x}\|_2/\|x - x^*\|_2$ where $\hat{x}$ and x^* are the reconstructed vectors after 9 iterations of reweighted and a single ℓ_1-minimization, respectively.

Figure 3.3.4 depicts the same results as Figure 3.3.3 above, but for the experiments with Bernoulli signals. We see that in this case again reweighted ℓ_1 offers improvements. In this setting, the improvements in the Bernoulli signals seem to be

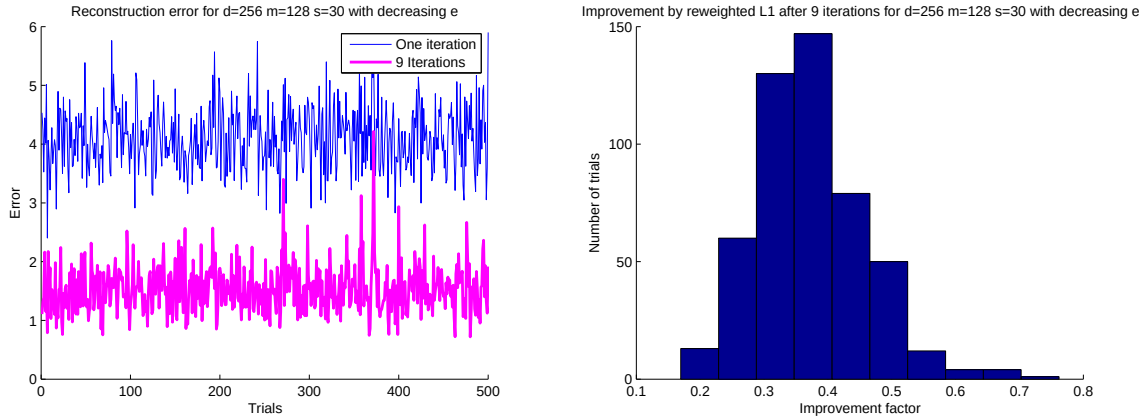


Figure 3.3.3: Improvements in the ℓ_2 reconstruction error using reweighted ℓ_1-minimization versus standard ℓ_1-minimization for Gaussian signals.

slightly less than in the Gaussian case. It is clear that in the case of flat signals like Bernoulli, the requirement on μ in Theorem 3.3.3 may be easier to satisfy, since the signal will have no small components unless they are all small. However, in the case of flat signals, if this requirement is not met, then the Theorem guarantees no improvements whatsoever. In the Gaussian case, even if the requirement on μ is not met, the Theorem still shows that improvements will be made on the larger coefficients.

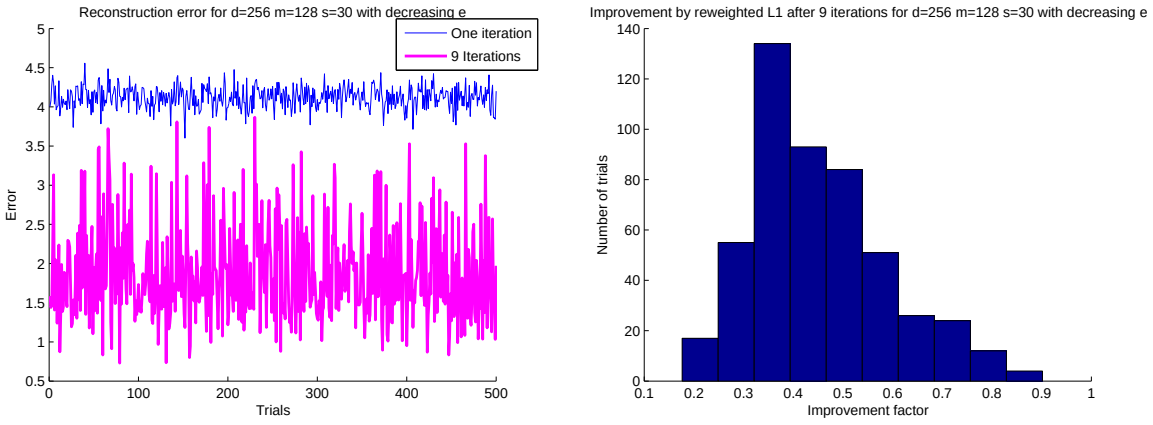


Figure 3.3.4: Improvements in the ℓ_2 reconstruction error using reweighted ℓ_1-minimization versus standard ℓ_1-minimization for Bernoulli signals.

3.4 Randomized Kaczmarz

Although the algorithms in compressed sensing themselves are not random, the measurement matrices used in the compression step are. The suggestion that randomness often makes things easier is the key to the idea of randomized algorithms. It is often the case that a deterministic algorithm's flaw can be fixed by introducing randomness. This notion is at the heart of a new randomized version of the well known Kaczmarz algorithm. Although the work on Kaczmarz is outside the realm on compressed sensing, it does bare some striking similarities.

The Kaczmarz method [41] is one of the most popular solvers of overdetermined linear systems and has numerous applications from computer tomography to image processing. The algorithm consists of a series of alternating projections, and is often considered a type of *Projection on Convex Sets* (POCS) method. Given a consistent system of linear equations of the form

$$Ax = b,$$

the Kaczmarz method iteratively projects onto the solution spaces of each equation in the system. That is, if $a_1, \ldots, a_m \in \mathbb{R}^n$ denote the rows of A, the method cyclically projects the current estimate orthogonally onto the hyperplanes consisting of solutions to $\langle a_i, x \rangle = b_i$. Each iteration consists of a single orthogonal projection. The algorithm can thus be described using the recursive relation,

$$x_{k+1} = x_k + \frac{b_i - \langle a_i, x_k \rangle}{\|a_i\|_2^2} a_i,$$

where x_k is the k^{th} iterate and $i = (k \bmod m) + 1$.

Theoretical results on the convergence rate of the Kaczmarz method have been difficult to obtain. Most known estimates depend on properties of the matrix A which may be time consuming to compute, and are not easily comparable to those of other iterative methods (see e.g. [18], [29], [37]). Since the Kaczmarz method cycles through the rows of A sequentially, its convergence rate depends on the order of the rows. Intuition tells us that the order of the rows of A does not change the difficulty level of the system as a whole, so one would hope for results independent of the ordering. One natural way to overcome this is to use the rows of A in a random order, rather than sequentially. Several observations were made on the improvements of this randomized version [50, 38], but only recently have theoretical results been obtained [61].

In designing a random version of the Kaczmarz method, it is necessary to set the probability of each row being selected. Strohmer and Vershynin propose in [61] to set the probability proportional to the Euclidean norm of the row. Their revised algorithm can then be described by the following:

$$x_{k+1} = x_k + \frac{b_{p(i)} - \langle a_{p(i)}, x_k \rangle}{\|a_{p(i)}\|_2^2} a_{p(i)},$$

where $p(i)$ takes values in $\{1, \ldots, m\}$ with probabilities $\frac{\|a_{p(i)}\|_2^2}{\|A\|_F^2}$. Here and throughout, $\|A\|_F$ denotes the Frobenius norm of A and $\|\cdot\|_2$ denotes the usual Euclidean norm or spectral norm for vectors or matrices, respectively. We note here that of course, one needs some knowledge of the norm of the rows of A in this version of the algorithm. In general, this computation takes $O(mn)$ time. However, in many cases such as the case in which A contains Gaussian entries, this may be approximately or exactly known.

In [61], Strohmer and Vershynin prove the following exponential bound on the expected rate of convergence for the randomized Kaczmarz method,

$$\mathbb{E}\|x_k - x\|_2^2 \leq \left(1 - \frac{1}{R}\right)^k \|x_0 - x\|_2^2, \tag{3.4.1}$$

where $R = \|A^{-1}\|^2 \|A\|_F^2$ and x_0 is an arbitrary initial estimate. Here and throughout, $\|A^{-1}\| \stackrel{\text{def}}{=} \inf\{M : M\|Ax\|_2 \geq \|x\|_2 \text{ for all } x\}$.

The first remarkable note about this result is that it is essentially independent of the number m of equations in the system. Indeed, by the definition of R, R is proportional to n within a square factor of the condition number of A. Also, since the algorithm needs only access to the randomly chosen rows of A, the method need not know the entire matrix A. Indeed, the bound (3.4.1) and the relationship of R to n shows that the estimate x_k converges exponentially fast to the solution in just $\mathrm{O}(n)$ iterations. Since each iteration requires $\mathrm{O}(n)$ time, the method overall has a $\mathrm{O}(n^2)$ runtime. Thus this randomized version of the algorithm provides advantages over all previous methods for extremely overdetermined linear systems. There are of course situations where other methods, such as the conjugate gradient method, outperform the randomized Kaczmarz method. However, numerical studies in [61] show that in many scenarios (for example when A is Gaussian), the randomized Kaczmarz method provides faster convergence than even the conjugate gradient method.

The remarkable benefits provided by the randomized Kaczmarz algorithm lead one to question whether the method works in the more realistic case where the system is corrupted by noise. In this paper we provide theoretical and empirical results to suggest that in this noisy case the method converges exponentially to the solution within a specified error bound. The error bound is proportional to $\sqrt{R}$, and we also provide a simple example showing this bound is sharp in the general setting.

3.4.1 Main Results

In this section we discuss the results by Needell in [51].

Theoretical and empirical studies have shown the randomized Kaczmarz algorithm to provide very promising results. Here we show that it also performs well in the case where the system is corrupted with noise. In this section we consider the system $Ax = b$ after an error vector r is added to the right side:

$$Ax \approx b + r.$$

First we present a simple example to gain intuition about how drastically the noise can affect the system. To that end, consider the $n \times n$ identity matrix A. Set $b = 0$, $x = 0$, and suppose the error is the vector whose entries are all one, $r = (1, 1, \ldots, 1)$. Then the solution to the noisy system is clearly r itself, and so by (3.4.1), the iterates x_k of randomized Kaczmarz will converge exponentially to r. Since A is the identity matrix, we have $R = n$. Then by (3.4.1) and Jensen's inequality, we have

$$\mathbb{E}\|x_k - r\|_2 \leq \left(1 - \frac{1}{R}\right)^{k/2} \|x_0 - r\|_2.$$

Then by the triangle inequality, we have

$$\mathbb{E}\|x_k - x\|_2 \geq \|r - x\|_2 - \left(1 - \frac{1}{R}\right)^{k/2} \|x_0 - r\|_2.$$

Finally by the definition of r and R, this implies

$$\mathbb{E}\|x_k - x\|_2 \geq \sqrt{R} - \left(1 - \frac{1}{R}\right)^{k/2} \|x_0 - r\|_2.$$

This means that the limiting error between the iterates x_k and the original solution x is $\sqrt{R}$. In [61] it is shown that the bound provided in (3.4.1) is optimal, so if we wish to maintain a general setting, the best error bound for the noisy case we can hope for is proportional to $\sqrt{R}$. Our main result proves this exact theoretical bound.

Theorem 3.4.1 (Noisy randomized Kaczmarz)**.** *Let x_k^* be the k^{th} iterate of the noisy Randomized Kaczmarz method run with $Ax \approx b+r$, and let $a_1, \dots a_m$ denote the rows of A. Then we have*

$$\mathbb{E}\|x_k^* - x\|_2 \leq \left(1 - \frac{1}{R}\right)^{k/2} \|x_0\|_2 + \sqrt{R}\gamma,$$

where $R = \|A^{-1}\|^2 \|A\|_F^2$ and $\gamma = \max_i \frac{|r_i|}{\|a_i\|_2}$.

Remark 3.4.2. *In the case discussed above, note that we have $\gamma = 1$, so the example indeed shows the bound is sharp.*

Before proving the theorem, it is important to first analyze what happens to the solution spaces of the original equations $Ax = b$ when the error vector is added. Letting $a_1, \dots a_m$ denote the rows of A, we have that each solution space $\langle a_i, x\rangle = b_i$ of the original system is a hyperplane whose normal is $\frac{a_i}{\|a_i\|_2}$. When noise is added, each hyperplane is translated in the direction of a_i. Thus the new geometry consists of hyperplanes parallel to those in the noiseless case. A simple computation provides the following lemma which specifies exactly how far each hyperplane is shifted.

Lemma 3.4.3. *Let H_i be the affine subspace of $\mathbb{R}^n$ consisting of the solutions to $\langle a_i, x\rangle = b_i$. Let H_i^* be the solution space of the noisy system $\langle a_i, x\rangle = b_i + r_i$. Then $H_i^* = \{w + \alpha_i a_i : w \in H_i\}$ where $\alpha_i = \frac{r_i}{\|a_i\|_2^2}$.*

Proof. First, if $w \in H_i$ then $\langle a_i, w + \alpha a_i\rangle = \langle a_i, w\rangle + \alpha\|a_i\|_2^2 = b_i + r_i$, so $w + \alpha a_i \in H_i^*$.

Next let $u \in H_i^*$. Set $w = u - \alpha a_i$. Then $\langle a_i, w\rangle = \langle a_i, u\rangle - r_i = b_i + r_i - r_i = b_i$, so $w \in H_i^*$. This completes the proof. □

We will also utilize the following lemma which is proved in the proof of Theorem 2 in [61].

Lemma 3.4.4. *Let x_{k-1}^* be any vector in $\mathbb{R}^n$ and let x_k be its orthogonal projection onto a random solution space as in the noiseless randomized Kaczmarz method run with $Ax = b$. Then we have*

$$\mathbb{E}\|x_k - x\|_2^2 \leq \Big(1 - \frac{1}{R}\Big)\|x_{k-1}^* - x\|_2^2,$$

where $R = \|A^{-1}\|^2\|A\|_F^2$.

We are now prepared to prove Theorem 3.4.1.

Proof of Theorem 3.4.1. Let x_{k-1}^* denote the $(k-1)^{th}$ iterate of noisy Randomized Kaczmarz. Using notation as in Lemma 3.4.3, let H_i^* be the solution space chosen in the k^{th} iteration. Then x_k^* is the orthogonal projection of x_{k-1}^* onto H_i^*. Let x_k denote the orthogonal projection of x_{k-1}^* onto H_i (see Figure 3.4.1).

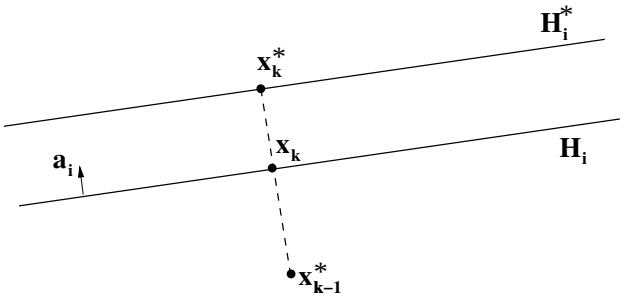


Figure 3.4.1: The parallel hyperplanes H_i and H_i^* along with the two projected vectors x_k and x_k^*.

By Lemma 3.4.3 and the fact that a_i is orthogonal to H_i and H_i^*, we have that $x_k^* - x = x_k - x + \alpha_i a_i$. Again by orthogonality, we have $\|x_k^* - x\|_2^2 = \|x_k - x\|_2^2 + \|\alpha_i a_i\|_2^2$. Then by Lemma 3.4.4 and the definition of γ, we have

$$\mathbb{E}\|x_k^* - x\|_2^2 \leq \Big(1 - \frac{1}{R}\Big)\|x_{k-1}^* - x\|_2^2 + \gamma^2,$$

where the expectation is conditioned upon the choice of the random selections in the first $k-1$ iterations. Then applying this recursive relation iteratively and taking full expectation, we have

$$\begin{aligned}\mathbb{E}\|x_k^* - x\|_2^2 &\leq \Big(1 - \frac{1}{R}\Big)^k \|x_0 - x\|_2^2 + \sum_{j=0}^{k-1} \Big(1 - \frac{1}{R}\Big)^j \gamma^2 \\ &\leq \Big(1 - \frac{1}{R}\Big)^k \|x_0 - x\|_2^2 + R\gamma^2.\end{aligned}$$

By Jensen's inequality we then have

$$\mathbb{E}\|x_k^* - x\|_2 \leq \left(\Big(1 - \frac{1}{R}\Big)^k \|x_0 - x\|_2^2 + R\gamma^2\right)^{1/2} \leq \Big(1 - \frac{1}{R}\Big)^{k/2} \|x_0 - x\|_2 + \sqrt{R}\gamma.$$

This completes the proof. □

3.4.2 Numerical Examples

In this section we describe some of our numerical results for the randomized Kaczmarz method in the case of noisy systems. The results displayed in this section use matrices with independent Gaussian entries. Figure 3.4.2 depicts the error between the estimate by randomized Kaczmarz and the actual signal, in comparison with the predicted threshold value. This study was conducted for 100 trials using 100×50

Gaussian matrices and independent Gaussian noise. The systems were homogeneous, meaning $x = 0$ and $b = 0$. The thick line is a plot of the threshold value, $\gamma\sqrt{R}$ for each trial. The thin line is a plot of the error in the estimate after 1000 iterations for the corresponding trial. As is evident by the plots, the error is quite close to the threshold. Of course in the Gaussian case depicted, it is not surprising that often the error is below the threshold. As discussed above, the threshold is sharp for certain kinds of matrices and noise vectors.

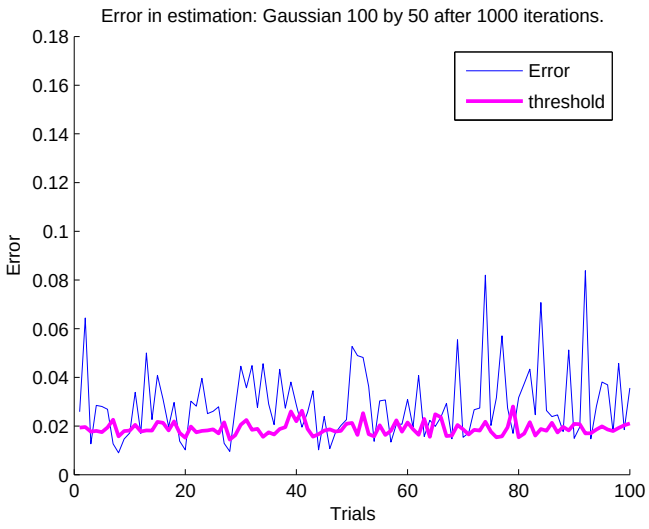


Figure 3.4.2: The comparison between the actual error in the randomized Kaczmarz estimate (thin line) and the predicted threshold (thick line).

Our next study investigated the convergence rate for the randomized Kaczmarz method with noise for homogeneous systems. Again we let our matrices A be 100×50 Gaussian matrices, and our error vector be independent Gaussian noise. Figure 3.4.3 displays a scatter plot of the results of this study over various trials. It is not surprising that the convergence appears exponential as predicted by the theorems.

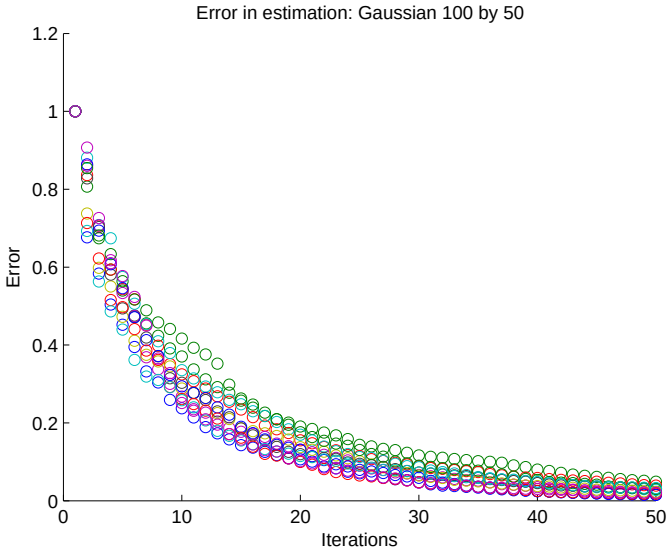


Figure 3.4.3: Convergence rate for randomized Kaczmarz over various trials.

Chapter 4

Summary

Compressed sensing is a new and fast growing field of applied mathematics that addresses the shortcomings of conventional signal compression. Given a signal with few nonzero coordinates relative to its dimension, compressed sensing seeks to reconstruct the signal from few nonadaptive linear measurements. As work in this area developed, two major approaches to the problem emerged, each with its own set of advantages and disadvantages. The first approach, L1-Minimization [6, 5], provided strong results, but lacked the speed of the second, the greedy approach. The greedy approach, while providing a fast runtime, lacked stability and uniform guarantees. This gap between the approaches has led researchers to seek an algorithm that could provide the benefits of both. In collaboration, my adviser Roman Vershynin and I have bridged this gap and provided a breakthrough algorithm, called Regularized Orthogonal Matching Pursuit (ROMP) [55, 54]. ROMP is the first algorithm to provide the stability and uniform guarantees similar to those of L1-Minimization, while providing speed as a greedy approach. After analyzing these results, my colleague Joel Tropp and I developed the algorithm Compressive Sampling Matching Pursuit (CoSaMP), which improved upon the guarantees of ROMP [53, 52]. CoSaMP is the

first algorithm to have provably optimal guarantees in every important aspect.

It was the negative opinions about conventional signal compression that spurred the development of compressed sensing. The traditional methodology is a costly process, which acquires the entire signal, compresses it, and then throws most of the information away. However, new ideas in the field combine signal acquisition and compression, significantly improving the overall cost. The problem is formulated as the realization of such a signal x from few linear measurements, of the form Φx where Φ is a (usually random) measurement matrix. Since Linear Algebra clearly shows that recovery in this fashion is not possible, the domain from which the signal is reconstructed must be restricted. Thus the domain that is considered is the domain of all *sparse* vectors. In particular, we call a signal *s-sparse* when it has s or less nonzero coordinates. It is now well known that many signals in practice are sparse in this sense or with respect to a different basis.

As discussed, there are several critical properties that an ideal algorithm in compressed sensing should possess. The algorithm clearly needs to be efficient in practice. This means it should have a fast runtime and low memory requirements. It should also provide uniform guarantees so that one measurement matrix works for all signals with high probability. Lastly, the algorithm needs to provide stability under noise in order to be of any use in practice. The second approach uses greedy algorithms and is thus quite fast both in theory and in practice, but lacks both stability and uniform guarantees. The first approach relies on a condition called the Restricted Isometry Property (RIP), which had never been usefully applied to greedy algorithms. For a measurement matrix Φ, we say that Φ satisfies the RIP with parameters (s, ε) if

$$(1-\varepsilon)\|v\|_2 \leq \|\Phi v\|_2 \leq (1+\varepsilon)\|v\|_2$$

holds for all s-sparse vectors v. We analyzed this property in a unique way and found consequences that could be used in a greedy fashion. Our breakthrough algorithm ROMP is the first to provide all these benefits (stability, uniform guarantees, and speed), while CoSaMP improves upon these significant results and provides completely optimal runtime bounds.

One of the basic greedy algorithms which inspired our work is Orthogonal Matching Pursuit (OMP), which was analyzed by Gilbert and Tropp [62]. OMP uses the observation vector $u = \Phi^* \Phi x$ to iteratively calculate the support of the signal x (which can then be used to reconstruct x). At each iteration, OMP selects the largest component of the observation vector u to be in the support, and then subtracts off its contribution. Although OMP is very fast, it does not provide uniform guarantees. Indeed, the algorithm correctly reconstructs a *fixed* signal x with high probability, rather than *all* signals. It is also unknown whether OMP is stable.

Our new algorithm ROMP is similar in spirit to OMP, in that it uses the observation vector u to calculate the support of the signal x. One of the key differences in the algorithm is that ROMP selects many coordinates of u at each iteration. The regularization step of ROMP guarantees that each of the selected coordinates have close to an equal share of the information about the signal x. This allows us to translate captured energy of the signal into captured support of the signal. We show that when the measurement matrix Φ satisfies the Restricted Isometry Property with parameters $(2s, c/\log s)$, ROMP exactly reconstructs any s-sparse signal in just s iterations. Our stability results show that for an *arbitrary* signal x with noisy measurements $\Phi x + e$, ROMP provides an approximation $\hat{x}$ to x that satisfies

$$\|\hat{x} - x\|_2 \leq C\sqrt{\log s}\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big), \tag{4.0.1}$$

where x_s is the vector of the s largest coordinates of x in magnitude. ROMP is thus the first greedy algorithm providing the strong guarantees of the L1-Minimization approach, and bridges the gap between the two approaches.

After the breakthrough of ROMP, Needell and Tropp developed a new algorithm, Compressive Sampling Matching Pursuit (CoSaMP) [53, 52]. CoSaMP maintains an estimation of the support as well as an estimation of the signal throughout each iteration. It again is a greedy algorithm that provides the uniform and stability guarantees of the L1 approach. CoSaMP improves upon the stability bounds of ROMP as well as the number of measurements required by the algorithm. Indeed we show that for any measurement matrix satisfying the Restricted Isometry Property with parameters (2s, c), CoSaMP approximately reconstructs any arbitrary signal x from its noisy measurements $u = \Phi x + e$ in at most $6s$ iterations:

$$\|\hat{x} - x\|_2 \leq C\Big(\|e\|_2 + \frac{\|x - x_s\|_1}{\sqrt{s}}\Big).$$

We also provide a rigorous analysis of the runtime, which describes how exactly the algorithm should be implemented in practice. CoSaMP thus provides optimal guarantees at every important aspect.

Appendix A

Matlab Code

This section contains original Matlab code used to produce the figures contained within this work.

A.1 Basis Pursuit

```
%NAME: Basis Pursuit Tester
%PURPOSE: Tests sparse, noisy, compressible signals on Basis Pursuit
%AUTHOR: Deanna Needell
%OUTSIDE FUNCTIONS: l1eq_pd (L1-Magic, J. Romberg)

clear all
warning off all

%Variables
N=[10:5:250];
d=[256];
n=[2:2:90];
p=[0.2:0.2:1]; %Used for compressible signals

%Number of Trials for each parameter set
numTrials = 500;
```

```
%Counters
numN = size(N, 2);
numd = size(d, 2);
numn = size(n, 2);
nump = size(p, 2);

%Data Collection
numCorr = zeros(numN, numd, numn);
minMeas = zeros(numN);
AvgerrorNormd = zeros(numN, numd, numn, nump);
Avgerror = zeros(numN, numd, numn, nump);
%for ip:=1:nump %USED FOR COMPRESSIBLE SIGNALS
  for id = 1:numd
    for iN=1:numN
     in=1;
     done=0;
     while in <= numn && ~done
       for trial = 1:numTrials
         tN = N(1, iN);
         td = d(1, id);
         tn = n(1, in);
         tp = p(1, ip);

         %Set Matrix
         Phi = randn(tN, td);
         Phi = sign(Phi);
         I = zeros(1,1);

         %Set signal
         z = randperm(td);
         v = zeros(td, 1);
         for t = 1:tn
             v(z(t))=1;
         end

         %USED IN THE CASE OF COMPRESSIBLE SIGNALS
         %y = sign(randn(td, 1));
         %noiErr=0;
         %for i=1:tn

             %v(z(i)) = i^(-1/tp)*y(i);
```

```
            %if i > tn
            %    noiErr = noiErr + abs(v(z(i)));
            %end
          %end

          %Set measurement and residual
          x = Phi * v;

          %USED IN THE CASE OF NOISE
          %e = randn(tN, 1);
          %x = x + e / norm(e, 2) / 2;

          %Set initial estimate
          x0 = Phi'*x;

          %Run Basis Pursuit (via L1-Magic)
          xp = l1eq_pd(x0, Phi, [], x, 1e-6);

          %Collect Data
          if norm(xp-v, 2) < 0.01
               numCorr(iN, id, in) = numCorr(iN, id, in) +1;
          end
          AvgerrorNormd(iN, id, in, ip) = (AvgerrorNormd(iN, id, in, ip) *
                    (counted-1) + (norm(xp-v,2)/noiErr*(tn)^0.5))/counted;
          Avgerror(iN, id, in, ip) = (Avgerror(iN, id, in, ip) *
                    (counted-1) + norm(xp-v, 2))/counted;
          end % end Trial

        if numCorr(iN, id, in) / numTrials > 0.98
           minMeas(iN) = tn;
        else
           done=1;
        end
        in = in +1;
       end % n
     end % N
  end % d
%end % p
```

A.2 Orthogonal Matching Pursuit

```
%NAME: Orthogonal Matching Pursuit Tester
%PURPOSE: Tests sparse signals on Orthogonal Matching Pursuit
%AUTHOR: Deanna Needell
%OUTSIDE FUNCTIONS: None

clear all
warning off all
%Variables
N=[10:5:250];
d=[256];
n=[2:2:90];

numTrials = 500;
numN = size(N, 2);
numd = size(d, 2);
numn = size(n, 2);

%Data Collection
numCorr = zeros(numN, numd, numn);
mostSpars = zeros(numN);

for id = 1:numd
 for iN=1:numN
   in=1;
   done=1;
   keepGo=1;
   while in <= numn && keepGo
     for trial = 1:numTrials
       tN = N(1, iN);
       td = d(1, id);
       tn = n(1, in);

       %Set Matrix
       Phi = randn(tN, td);
       Phi = sign(Phi);
       I = zeros(1,1);

       %Set signal
       z = randperm(td);
```

```
  supp=z(1:tn);
  v = zeros(td, 1);
  for t = 1:tn
      v(z(t))=1;
  end
  x = Phi * v;
  r = x;

  %Run OMP

  while length(I)-1 < tn
    u = Phi' * r;
    absu = abs(u);
    [b, ix] = sort(absu, 'descend');
    bestInd = ix(1);
    bestVal = b(1);

    %Update I
    I(length(I)+1) = bestInd;

    %Update the residual
    PhiSubI = Phi(:, I(2));
    for c=3:length(I)
        if ~isMember(I(2:c-1),I(c))
          PhiSubI(:,c-1) = Phi(:, I(c));
        end
    end
    y = lscov(PhiSubI, x);
    r = x - PhiSubI * y;
  end

if isMember(supp, I);
     numCorr(iN, id, in) = numCorr(iN, id, in) +1;
end
end % end Trial

if numCorr(iN, id, in) / numTrials > 0.98 && done
    mostSpars(iN) = tn;
else
    done=0;
end
```

```
     if numCorr(iN, id, in) <= 0.01
         keepGo=0;
     end
     in = in +1;
   end % n
 end % N
end % d
```

A.3 Regularized Orthogonal Matching Pursuit

```
function [vOut, numIts] = romp(n, Phi, x)
% [vOut] = romp(n, Phi, x)
%%% PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%%%
% d = ambient dimension of the signal v
% N = number of measurements
% n = sparsity level of n
% Phi = N by d measurement matrix
% x = measurement vector (Phi * v)
% vOut = reconstructed signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% FUNCTION DESCRIPTION %%%%%%%%%%%%%%%%
% romp takes parameters as described
% above. Given the sparsity level n and
% the N by d measurement matrix Phi, and
% the measurement vector x = Phi * v, romp
% reconstructs the original signal v.
% This reconstruction is the output.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear r I J J0 u b ix numIts Jvals
warning off all

N = size(Phi, 1);
d = size(Phi, 2);

% Set residual
r = x;
```

```
%Set index set to "empty"
I = zeros(1,1);

%Counter (to be used optionally)
numIts = 0;

%Run ROMP
while length(I)-1 < 2*n && norm(r) > 10^(-6)

   numIts = numIts + 1;

   %Find J, the biggest n coordinates of u
   u = Phi' * r;
   absu = abs(u);
   [b, ix] = sort(absu, 'descend');
   J = ix(1:n);
   Jvals = b(1:n);

   %Find J0, the set of comparable coordinates with maximal energy
   w=1;
   best = -1;
   J0 = zeros(1);
   while w <= n
       first = Jvals(w);
       firstw = w;
       energy = 0;
       while ( w <= n ) && ( Jvals(w) >= 1/2 * first )
           energy = energy + Jvals(w)^2;
           w = w+1;
       end
       if energy > best
           best = energy;
           J0 = J(firstw:w-1);
       end
   end

   %Add J0 to the index set I
   I(length(I)+1: length(I)+length(J0)) = J0;

   %Update the residual
   PhiSubI = Phi(:, I(2));
```

```
   for c=3:length(I)
       if ~isMember(I(2:c-1),I(c))
         PhiSubI(:,c-1) = Phi(:, I(c));
       end
   end
   y = lscov(PhiSubI, x);
   r = x - PhiSubI * y;

end % end Run IRA

vSmall = PhiSubI \ x;
vOut = zeros(d, 1);
for c=2:length(I)
    vOut(I(c)) = vSmall(c-1);
end
```

A.4 Compressive Sampling Matching Pursuit

```
%NAME: CoSaMP Tester
%PURPOSE: Tests sparse signals on CoSaMP
%AUTHOR: Deanna Needell
%OUTSIDE FUNCTIONS: None

%Testing Parameters
sVals=[1:1:55]; % Sparsity levels
mVals=[5:5:250]; %Measurement levels
dVals=[256]; %dimension

numTrials=500; %Number of trials per parameter set

%Set Variable lengths and Data Collection
nums=length(sVals);
numm=length(mVals);
numd=length(dVals);
numCorrect = zeros(nums, numm, numd);
trend99 = zeros(numm, 1);

for is=1:nums
```

```
for im=1:numm
    for id=1:numd
        %Set Parameters
        s = sVals(is);
        m = mVals(im);
        d = dVals(id);

        %Start a trial
        for trial=1:numTrials

            %Generate Measurement matrix
            Phi = randn(m,d);

            %Generate sparse signal
            z = randperm(d);
            x = zeros(d, 1);
            x(z(1:s)) = sign(randn(s,1));

            %Generate measurements
            u = Phi*x;

            %Begin CoSaMP

            %Initialize
            a = zeros(d,1);
            v = u;
            it=0;
            stop = 0;
            while ~stop

                %Signal Proxy
                y = Phi'*v;
                [tmp, ix] = sort(abs(y), 'descend');
                Omega = ix(1:2*s);
                [tmp, ix] = sort(abs(a), 'descend');
                T = union(Omega, ix(1:s));

                %Signal Estimation
                b = zeros(d, 1);
                b(T) = Phi(:, T) \ u;
```

```
                    %Prune
                    [tmp, ix] = sort(abs(b), 'descend');
                    a = zeros(d, 1);
                    a(ix(1:s), 1) = b(ix(1:s), 1);

                    %Sample Update
                    v = u - Phi*a;

                    %Iteration counter
                    it = it + 1;

                    %Check Halting Condition
                    if norm(a-x) <= 10^(-4) || it > max(8*s, 60)
                        stop = 1;
                    end

                end %End CoSaMP iteration

                %Collect Data
                if norm(a-x) <= 10^(-4)
                    numCorrect(is, im, id) = numCorrect(is, im, id) + 1;
                end

            end % End trial
        end %d

        if trend99(im) == 0 && numCorrect(is, im, id) >= 0.99*numTrials
            trend99(im) = s;
        end
    end %m
end %s
```

A.5 Reweighted L1 Minimization

```
%NAME: Reweighted L1-Minimization Tester
%PURPOSE: Tests sparse and noisy signals on Reweighted L1
%AUTHOR: Deanna Needell
%OUTSIDE FUNCTIONS: CVX package (Michael Grant and Stephen Boyd)

N = 256; %dimension
```

```
M = 128; %measurements
kVals = [30]; %sparsity
eepsVals = [1];
numTrials = 500;
maxIter = 9;

errorVecDecoding = zeros(length(kVals),length(eepsVals),maxIter,numTrials);
errors = zeros(numTrials,1);
for trial = 1:numTrials
   for kIndex = 1:length(kVals)
       K = kVals(kIndex);
       for eIndex = 1:length(eepsVals)
           eeps = eepsVals(eIndex);

           % Gaussian spikes in random locations
           x = zeros(N,1); q = randperm(N);
           x(q(1:K)) = sign(randn(K,1));

           % measurement matrix
           Phi = sign(randn(M,N))/sqrt(M);

           % observations
           err = randn(M, 1);
           sigma = 0.2*norm(Phi*x,2)/norm(err,2);
           err = sigma*err;
           y = Phi*x + err;
           errors(trial) = norm(err,2);

           for iter = 1:maxIter
               %Set the weights
               if iter > 1

                   weights = 1./(abs(xDecoding)+eeps/(1000*iter));
               else
                   weights = 1*ones(N,1);
               end

               %Set noise tolerance parameter
               delta=sqrt(sigma^2*(M+2*sqrt(2*M)));

               %Use CVX to perform minimization
```

```
                cvx_begin
                cvx_quiet(true)
                    variable xa(N);
                    minimize( norm(diag(weights)*xa,2) );
                    subject to
                         norm(Phi*xa - y, 2) <= delta;
                cvx_end

                %Collect results
                xDecoding = xa;
                errorVecDecoding(kIndex,eIndex,iter,trial) = norm((x-xDecoding),2);

            end
        end
    end

end
```

A.6 Randomized Kaczmarz

```
%NAME: Randomized Kaczmarz Tester
%PURPOSE: Tests RK on noisy systems
%AUTHOR: Deanna Needell
%OUTSIDE FUNCTIONS: none

clear all
warning off all

m = 100;  %rows
n=100;     %columns

numIts = 1000;
numTrials = 100;

A = zeros(numTrials, m, n);
e = zeros(numTrials, m);
x = zeros(numTrials, n);
b = zeros(numTrials, m);
```

```
est = zeros(numTrials, numIts, n);  %estimations
initErr = zeros(numTrials);
R = zeros(numTrials, 1); %Value of R (as in paper)
mu = zeros(numTrials); %Coherence
gamma = zeros(numTrials, 1); %worst error to row norm ratio
beta = zeros(numTrials); %beta is in theorem
errorsRK = zeros(numTrials, numIts);
errorsCG = zeros(numTrials, numIts);

for trial=1:numTrials
    if mod(trial, 1) == 0
        display(['Trial ', num2str(trial)]);
    end
    %Set matrix equation
    A(trial, :, :) = sign(randn(m,n));
    x = zeros(n, 1)';
    b(trial,:) = reshape(A(trial, :, :), m, n)*(x');

    %Set initial guess, ||x - x0|| = 1
    est(trial, 1, :) = randn(1, n);
    est(trial, 1, :) = est(trial, 1, :) / norm(reshape(est(trial, 1, :), 1, n),2)
    origest = reshape(est(trial, 1, :), 1, n)';

    %Add error to RHS of Ax=0
    e(trial, :) = randn(1, m)*2;
    e(trial, :) = e(trial, :) / norm(e(trial, :), 2) / 10;
    b(trial, :) = b(trial, :)+e(trial, :);  %%%%NOISY!!

    %Calculate stats
    initErr = norm(reshape(est(trial, 1, :), 1, n) - x,2);
    fronorm = norm(reshape(A(trial, :, :), m, n), 'fro');
    R(trial) = norm(pinv(reshape(A(trial, :, :), m, n)),2)*fronorm;
    temp = zeros(1, m);
    for i=1:m
        temp(i) = norm(reshape(A(trial, i, :), 1, n), 2);
    end
    gamma(trial) = max(abs( e(trial, :)./temp ));
    errorsRK(trial, 1) = norm(reshape(est(trial, 1, :), 1, n)-x,2);
    errorsCG(trial, 1) = norm(reshape(est(trial, 1, :), 1, n)-x,2);

    %Run RK
```

```
    for it=2:numIts
        %Select random hyperplane
        pick = rand * fronorm^2;
        counter = 0;
        index = 1;
        while counter + norm(reshape(A(trial, index, :),1, n), 2)^2 < pick
            counter = counter + norm(reshape(A(trial, index, :),1, n), 2)^2;
            index = index + 1;
        end

        %Modify estimation
        est(trial, it, :) = est(trial, it-1, :) + (b(index) - dot((A(trial, index,
        (est(trial, it-1, :))) )/ (norm(reshape(A(trial, index, :), 1, n),2)^2) *
        errorsRK(trial, it) = norm(reshape(est(trial, it, :), 1, n)-x,2);
    end

    %Run CG
    for it=1:numIts
        [estcg,flag] = cgs(reshape(A(trial, :, :),m, n),b(trial,:)',10^(-10), it);
        errorsCG(trial, it) = norm(estcg-x',2);
    end


end
```

References

[1] Richard Baraniuk and Philippe Steeghs. Compressive radar imaging. IEEE Radar Conference, Waltham, Massachusetts, 2007.

[2] Å. Björck. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, 1996.

[3] T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. Preprint, 2008.

[4] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete Fourier information. *IEEE Trans. Info. Theory*, 52(2):489–509, Feb. 2006.

[5] 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.

[6] E. Candès and T. Tao. Decoding by linear programming. *IEEE Trans. Inform. Theory*, 51:4203–4215, 2005.

[7] E. Candès, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted ell-1 minimization. *J. Fourier Anal. Appl.*, 14(5):877–905, Dec. 2008.

[8] E. J. Candès. The restricted isometry property and its implications for compressed sensing. *C. R. Math. Acad. Sci. Paris, Serie I*, 346:589–592, 2008.

[9] E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Trans. Info. Theory*, 51(12):4203–4215, Dec. 2005.

[10] A. Cohen, W. Dahmen, and R. DeVore. Compressed sensing and best k-term approximation. Preprint, 2006.

[11] T. Cormen, C. Lesierson, L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 2nd edition, 2001.

[12] G. Cormode and S. Muthukrishnan. Combinatorial algorithms for compressed sensing. Technical report, DIMACS, 2005.

[13] Compressed sensing webpage. `http://www.dsp.ece.rice.edu/cs/`.

[14] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing: Closing the gap between performance and complexity. Submitted for publication, Mar. 2008.

[15] G. B. Dantzig and M. N. Thapa. *Linear Programming*. Springer, New York, NY, 1997.

[16] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure Appl. Math.*, 57:1413–1457, 2004.

[17] M. E. Davies and R. Gribonval. Restricted isometry constants where ℓ_p sparse recovery can fail for $0 < p \leq 1$. Preprint, 2008.

[18] F. Deutsch and H. Hundal. The rate of convergence for the method of alternating projections. *J. Math. Anal. Appl.*, 205(2):381–405, 1997.

[19] D. L. Donoho. Compressed sensing. *IEEE Trans. Info. Theory*, 52(4):1289–1306, Apr. 2006.

[20] D. L. Donoho and M. Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1-minimization. *Proc. Natl. Acad. Sci.*, 100:2197–2202, 2003.

[21] D. L. Donoho and P. B. Stark. Uncertainty principles and signal recovery. *SIAM J. Appl. Math.*, 49(3):906–931, June 1989.

[22] D. L. Donoho and J. Tanner. Counting the faces of radomly-projected hypercubes and orthants, with applications. *J. Amer. Math. Soc.*, 22(1):1–53, 2009.

[23] D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck. Sparse solution of underdetermined linear equations by stagewise Orthogonal Matching Pursuit (StOMP). Submitted for publication, 2007.

[24] R. Dorfman. The detection of defective members of large populations. *Ann. Math. Statist.*, 14:436–440, 1943.

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

[26] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to Compressed Sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing*, 1(4):586–598, 2007.

[27] M. Fornasier and H. Rauhut. Iterative thresholding algorithms. Preprint, 2007.

[28] S. Foucart and M.-J. Lai. Sparsest solutions of undetermined linear systems via ℓ_q-minimization for $0 < q \leq 1$. Preprint, 2008.

[29] A. Galàntai. On the rate of convergence of the alternating projection method in finite dimensional spaces. *J. Math. Anal. Appl.*, 310(1):30–44, 2005.

[30] A. Garnaev and E. Gluskin. On widths of the euclidean ball. *Sov. Math. Dokl.*, 30:200–204, 1984.

[31] A. Gilbert, M. Strauss, J. Tropp, and R. Vershynin. Algorithmic linear dimension reduction in the ℓ_1 norm for sparse vectors. Submitted for publication, August 2006.

[32] A. Gilbert, M. Strauss, J. Tropp, and R. Vershynin. One sketch for all: Fast algorithms for compressed sensing. In *Proc. 39th ACM Symp. Theory of Computing*, San Diego, June 2007.

[33] A. C. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. J. Strauss. Near-optimal sparse Fourier representations via sampling. In *Proc. of the 2002 ACM Symposium on Theory of Computing STOC*, pages 152–161, 2002.

[34] A. C. Gilbert, M. Muthukrishnan, and M. J. Strauss. Approximation of functions over redundant dictionaries using coherence. In *Proc. of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, Jan. 2003.

[35] A. C. Gilbert, S. Muthukrishnan, and M. J. Strauss. Improved time bounds for near-optimal sparse fourier representation via sampling. In *Proceedings of SPIE Wavelets XI*, San Diego, CA, 2005.

[36] I.F. Gorodnitsky, J. George, and B.D. Rao. Neuromagnetic source imaging with focuss: A recursive weighted minimum norm algorithm. *Electrocephalography and Clinical Neurophysiology*, 95:231–251, 1995.

[37] M. Hanke and W. Niethammer. On the acceleration of kaczmarz's method for inconsistent linear systems. *Linear Alg. Appl.*, 130:83–98, 1990.

[38] G.T. Herman and L.B. Meyer. Algebraic reconstruction techniques can be made computationally efficient. *IEEE Transactions on Medical Imaging*, 12(3):600–609, 1993.

[39] M. Herman and T. Strohmer. High-resolution radar via compressed sensing. *IEEE Trans. Signal Processing*, 57(6), 2007.

[40] M. Iwen. A deterministic sub-linear time sparse fourier algorithm via non-adaptive compressed sensing methods. Preprint, 2007.

[41] S. Kaczmarz. Angenäherte auflösung von systemen linearer gleichungen. *Bull. Internat. Acad. Polon.Sci. Lettres A*, pages 335–357, 1937.

[42] B. Kashin. The widths of certain finite dimensional sets and classes of smooth functions. *Izvestia*, 41:334–351, 1977.

[43] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. A method for large-scale ℓ_1-regularized least-squares problems with applications in signal processing and statistics. Submitted for publication, 2007.

[44] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proc. of 2nd Berkeley Symposium*, pages 481–492, Berkeley, CA, 1951.

[45] ℓ_1 magic. `http://www.acm.caltech.edu/l1magic/`.

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

[47] S. Mallat and Z. Zhang. Matching Pursuits with time-frequency dictionaries. *IEEE Trans. Signal Process.*, 41(12):3397–3415, 1993.

[48] S. Mendelson, A. Pajor, and N. Tomczak-Jaegermann. Uniform uncertainty principle for Bernoulli and subgaussian ensembles. To appear, *Constr. Approx.*, 2009.

[49] S. Muthukrishnan. *Data Streams: Algorithms and Applications*. Now Publishers, Hanover, MA, 2005.

[50] F. Natterer. *The Mathematics of Computerized Tomography*. Wiley, New York, 1986.

[51] D. Needell. Randomized kaczmarz solver for noisy linear systems. Submitted for publication, February 2009.

[52] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. ACM Technical Report 2008-01, California Institute of Technology, Pasadena, July 2008.

[53] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from noisy samples. *Appl. Comput. Harmon. Anal.*, 2008. DOI: 10.1016/j.acha.2008.07.002.

[54] D. Needell and R. Vershynin. Signal recovery from incomplete and inaccurate measurements via Regularized Orthogonal Matching Pursuit. Submitted for publication, October 2007.

[55] D. Needell and R. Vershynin. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Found. Comput. Math.*, 2007. DOI: 10.1007/s10208-008-9031-3.

[56] Y. E. Nesterov and A. S. Nemirovski. *Interior Point Polynomial Algorithms in Convex Programming*. SIAM, Philadelphia, 1994.

[57] H. Rauhut. On the impossibility of uniform sparse reconstruction using greedy methods. *Sampl. Theory Signal Image Process.*, 7(2):197–215, 2008.

[58] G. Reeves and M. Gastpar. Sampling bounds for sparse support recovery in the presence of noise. Submitted for publication, Jan. 2008.

[59] M. Rudelson and R. Vershynin. On sparse reconstruction from Fourier and Gaussian measurements. *Comm. Pure Appl. Math.*, 61:1025–1045, 2008.

[60] S. Sarvotham, D. Baron, and R. Baraniuk. Sudocodes – fast measurement and reconstruction of sparse signals. In *IEEE Int. Symposium on Information Theory (ISIT)*, Seattle, July 2006.

[61] T. Strohmer and R. Vershynin. A randomized solver for linear systems with exponential convergence. In *RANDOM 2006 (10th International Workshop on Randomization and Computation)*, number 4110 in Lecture Notes in Computer Science, pages 499–507. Springer, 2006.

[62] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Info. Theory*, 53(12):4655–4666, 2007.

[63] J. A. Tropp, A. C. Gilbert, S. Muthukrishnan, and M. J. Strauss. Improved sparse approximation over quasi-incoherent dictionaries. In *Proc. of the 2003 IEEE International Conference on Image Processing*, Barcelona, 2003.

[64] A. van der Sluis and H.A. van der Vorst. The rate of convergence of conjugate gradients. *Numer. Math.*, 48:543–560, 1986.

[65] R. Vershynin. Beyond Hirsch conjecture: walks on random polytopes and smoothed complexity of the simplex method. *SIAM J. Comput.*, 2006. To appear.

[66] M. Wassermann, A. Neisser, and C. Bruck. Eine serodiagnostische reaktion bei syphilis. *Deutsche medicinische Wochenschrift*, 32:745–746, 1906.