PRIMAL DUAL PURSUIT
A HOMOTOPY BASED ALGORITHM FOR THE DANTZIG SELECTOR

A Thesis
Presented to
The Academic Faculty

by

Muhammad Salman Asif

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2008

PRIMAL DUAL PURSUIT
A HOMOTOPY BASED ALGORITHM FOR THE DANTZIG SELECTOR

Approved by:

Justin K. Romberg, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

James H. McClellan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Russell M. Mersereau
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: June 20, 2008

To

My parents with utmost respect,

Haiqa and Dayan with best dreams

and Haleema with my deepest love.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Justin Romberg, for all the inspiration, motivation and guidance. Without his invaluable insight and constant mentoring this thesis would have not been possible. I will always be grateful to him for introducing me to this research area with so many new and exciting problems and helping me all along the way. I cannot thank him enough for the long hours of discussion on any problem I brought to him; anytime, anywhere. I also want to thank him for reading all the drafts of this thesis, his suggestions helped a lot in improving its content and presentation. I am grateful to him for being so friendly, patient and kind to me all the time (not to mention all the squash games he beats me in, ruthlessly!).

I want to thank my thesis committee members Prof. James McClellan and Prof. Russell Mersereau for their encouraging remarks about this work. I would like to thank my teachers here at Georgia Tech., all of whom influenced me a lot. I would like to thank Profs. William Green and Michael Westdickenberg who taught me about mathematical analysis. I would also like to thank Profs. John Barry and Faramarz Fekri for their exciting classes in my first semester here.

I would like to extend my gratitude towards my undergraduate advisor, Amjad Luna, whose guidance has been instrumental in every possible way, Thankyou! I would also like to thank all my teachers (and later colleagues) at UET Lahore, without whom I would not have been here.

Many thanks to all my friends who made my time here a lot more enjoyable than I had anticipated. First of all, I must thank Farasat Munir and Mohammad Omer for being a huge support to me whenever I needed them. I cherish their friendship

a lot. I especially want to thank Omer for his help and consideration at all those times when I have nobody else to talk to. I also want to thank my roommate Umair Bin Altaf ("patti") for all the great time so far, for forcing me to learn LaTeX (along with many other things) and carefully reading the initial drafts of my thesis. William Mantzel, with whom I discuss almost all of my research problems and he never gets tired of it, Thanks!. Also I wish to thank Aaron Hatch with whom I have taken most of my classes here and it has always been fun to spend time with him.

Last but not the least, to those who mean the most to me, my family. I am wholeheartedly thankful to my parents, my sisters Hana and Madiha, and my brother Nauman for their love and help throughout my life. Without their support and confidence in me, I could never have done this or anything!

Salman,

June 2008

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

The Dantzig selector is a near ideal estimator for recovery of sparse signals from linear measurements. Consider the following measurement model

$$y = Ax + e,$$

where x is the n-dimensional sparse signal, y is the measurement vector in a much lower dimension $m \ll n$, A is the $m \times n$ measurement/sensing matrix and e is the error in our measurements. The Dantzig selector estimates x by solving the following optimization program

$$\text{minimize } \|\tilde{x}\|_1 \quad \text{subject to} \quad \|A^T(A\tilde{x} - y)\|_\infty \leq \epsilon, \tag{DS}$$

for some $\epsilon > 0$. This is a convex program, can be recast as a linear program (LP) and solved using any modern optimization method e.g., interior point methods.

In this thesis we propose a fast and efficient scheme for solving the *Dantzig Selector* (DS), which we call "Primal-Dual pursuit". This algorithm can be thought of as a *primal-dual homotopy* approach to solve the Dantzig selector. It computes the solution to (DS) for a range of ϵ values, by starting with a large ϵ and moving towards the desired value. Our algorithm successively updates the primal and dual supports as ϵ reduces to the desired value, which gives the final solution. The homotopy path taken by solution of (DS) with varying ϵ is piecewise linear. At the critical values of ϵ in this path, either some new elements enter the support of the signal or some existing elements leave the support. We derive the optimality and feasibility conditions which are used to update the solutions at these critical points. We also present a detailed analysis of primal-dual pursuit for sparse signals in noiseless case. We show that if our signal is S-sparse, then we can find all its S elements in exactly S steps using about

$S^2 \cdot \log n$ random measurements with very high probability (although in practice we observe this property with only $S \cdot \log n$ measurements).

The step-wise structure of this scheme helps us to have a very fast implementation in which we do not need to solve a large system of equations at each step (as we do in conventional optimization routines). Instead, we just update our existing solution whenever the supports change, which we can do very efficiently by low rank update of inverse matrices.

CHAPTER I

INTRODUCTION

In recent few years theory of compressed sensing has captured a lot of interest in applied mathematics and statistics community. This theory attempts to answer one fundamental question in signal processing and harmonic analysis: how much information about a signal would be sufficient for its exact reconstruction?

In signal processing, we usually answer this question using Shannon-Nyquist Sampling theorem, which says

> If a function $f(t)$ contains no frequencies higher than W cycles per second, it is completely determined by giving its ordinates at a series of points spaced $1/(2W)$ seconds apart.

Sampling theorem tells us that in order to reconstruct a bandlimited signal from its discrete samples we need to take samples at atleast twice the maximum frequency in our signal (known as Nyquist rate). In fact this is the underlying principle in nearly all data acquisition schemes used in practice. However, in many applications Nyquist rate can be so high that we need to compress our sampled data before we can store or transmit it, e.g., images and videos. Even with modern-day technology no system can handle image or video signals without compression. In many other applications e.g., medical imaging and high frequency analog-to-digital converters, increasing the sampling rate beyond the current state-of-the art is very expensive.

It is well known that most signals of our interest can be represented using very few coefficients in some appropriate basis [19, 38]. For example pure sinusoids in Fourier basis and piecewise smooth signals in wavelet basis. In addition to Fourier and wavelet basis we have local cosine basis, wavelet packets [5, 21, 38] and curvelets

[12], along with numerous other representations in the expanding family of '-lets'. In many modern compression schemes e.g., transform coding, we take advantage of this fact that a small number of coefficients in some suitable basis will be sufficient to represent a signal without causing much perceptual loss. We call such basis as "sparsity inducing" and the signal as being "compressible" in that basis. The entries of a compressible signal decay very rapidly when sorted with their magnitude, usually following some power law. And we can form a good approximation of the signal by coding only few largest coefficients. For example images tend to be compressible with discrete cosine transform (DCT) and wavelets on which JPEG [45] and JPEG2000 [50] compression standards[1] are based respectively.

In short, if a signal is known to be compressible in some basis, we transform it from natural basis (time or space) to its respective sparsity inducing basis. Then we encode only a small number of the most significant coefficients (depending on the desired compression ratio and acceptable distortion) and throw away the rest. This approach is known as "non linear approximation" or "best k-term approximation" [38], where k denotes the number of elements which are encoded. It is an adaptive scheme (i.e., signal dependent) and we need to compute all the transform coefficients before choosing the best k terms, as the location of important coefficients is not known in advance.

This scheme of compression, where we acquire a complete signal, encode a small portion out of it and throw away everything else, seems to be very wasteful. This raises a fundamental question: since we know that signals/images which we acquire are sparse or compressible in some known basis, so why should we spend energy on acquiring all the data we will throw eventually? Is it possible to combine the data acquisition (sensing) and compression into one step, so that we capture only as much

[1] In these standards weighted quantization and adaptive bit allocation are used to achieve compression instead of truncating signal to only few largest coefficients.

data as we will be keeping at the end? The answer is yes, and this is what compressed sensing is all about.

Compressed sensing offers us an alternative to the traditional sampling schemes. It provides us framework for a novel sampling or "sensing" mechanism which combines compression and sampling. In a way it brings down the acquisition rate to a much lower value than what is dictated by the sampling theorem. The underlying principles to compressed sensing are sparsity of the observed signal in some basis and incoherence between the measurement and representation bases. For example, it suggests that if a signal $x \in \mathbb{R}^n$ has only S non-zero entries, then we do not need all n samples, instead we can *almost always* recover x from about $S \cdot \log n$ linear measurements. And that is what gives us compression along with acquisition. However, an added cost to this procedure is that our reconstruction phenomenon becomes highly non linear. For reconstruction we have to solve an optimization program which involves minimizing some sparsity measure under certain data fidelity constraints. There have been many approaches discussed in literature for sparse signal recovery from linear measurements, for example, relaxed convex optimization based methods [9, 13, 24, 53], greedy approximation algorithms [41, 51, 54], gradient projection algorithms [29], iterative shrinkage methods [20], homotopy based methods [28, 43] and iterative re-weighted norm based algorithms [32, 46].

In compressed sensing we are typically interested in minimizing ℓ_1 norm of the sparse coefficients under some data fidelity constraints. In this thesis our main focus will be on the Dantzig selector [10], which is a near ideal estimator for recovery of sparse signals from linear measurements. Dantzig selector is convex in nature and can be recast as a linear program (LP). Our main contribution here is a new homotopy based algorithm for Dantzig selector, which can potentially replace the optimization routines currently being used for that purpose. We call this new algorithm "primal-dual pursuit" or "PD-pursuit".

The organization of thesis is as follows. In chapter 2 we will give a brief introduction to compressed sensing. In chapter 3 we will present the primal-dual pursuit algorithm. In chapter 4 we will give detailed analysis about the S-step solution property of our algorithm and discuss the similarities of Dantzig selector with LASSO. We will also discuss there some challenges we faced to establish this property. In chapter 5 we will discuss the numerical implementation and experimental results for our proposed algorithm.

CHAPTER II

COMPRESSED SENSING

Compressed sensing [23] or compressive sampling [1, 6] (CS) is a novel sensing or sampling paradigm which suggests that under certain conditions one can almost always recover a signal or image from far fewer samples or *linear measurements* than required by traditional data acquisition schemes. Moreover these measurements are completely *non-adaptive*; we do not assume any knowledge about the locations or amplitude of signal coefficients. The reconstruction phenomenon however is non linear and we have to solve a convex program to reconstruct the signal.

The conditions for exact reconstruction involve "sparsity" of signal in some representation basis and "incoherence" between the representation and measurement bases.

Sparsity of a signal is defined as the number of non-zero elements in signal under some representation. We will say a signal is S-sparse if its support is of cardinality less than or equal to S. *Support* of a signal $x \in \mathbb{R}^n$ is defined as

$$\text{supp}(x) \stackrel{\text{def}}{=} \{i : x_i \neq 0\}.$$

If our signal is not perfectly sparse but compressible (i.e., decay rapidly), then we can extend the ideas of CS to best S-term approximation.

Incoherence gives a measure of signal concentration/sparsity in two different bases. It deals with the idea that if our signal is concentrated or sparse in one basis then we should take measurements in some basis where it is well spread out.

2.1 Measurement model for compressed sensing

Consider a general problem of reconstructing a signal $x \in \mathbb{R}^n$ from its m linear measurements y of the form

$$y_k = \langle x, \varphi_k \rangle, \quad k = 1, \ldots, m \quad \text{or} \quad y = \Phi x,$$

where each measurement y_k is given by an inner product of signal x with some "sensing" function $\varphi_k \in \mathbb{R}^n$, and Φ will be called a *sensing* or *measurement matrix*. This is a generalized methodology for "sampling" a signal which we will call "sensing". The choice of φ_k gives us flexibility in the design as how to gather information about x. If we take φ_k as Dirac delta functions (spikes), then y is a vector of samples in time or space domain, so our sensing methodology becomes conventional sampling. If the φ_k are block indicator functions of pixels, then y is the image data typically collected by sensor array in a digital camera. If the φ_k are complex sinusoids at different frequencies then y is a vector of Fourier coefficients; as in the case of magnetic resonance imaging (MRI). The choice of sensing functions φ_k has some strong implications in signal recovery as well. As indicated earlier, our signal must be well spread out in the basis where we take measurements, this ensures that we get enough information from the under-sampled data such that reconstruction is possible.

In case of compressed sensing we are interested in taking m linear and non-adaptive measurements y, where $m \ll n$, and we want to recover x from such "compressed measurements". In order to reconstruct x from y, we need to solve the following system of linear equations

$$\Phi x = y. \tag{2.1}$$

This is an under-determined system, where the dimension of observation vector is significantly smaller than the dimension of original signal. In general, it is impossible to find the exact solution from an under-determined system of equations. Since $m \ll n$ in (2.1), there exist infinitely many possible solutions, and they all lie on

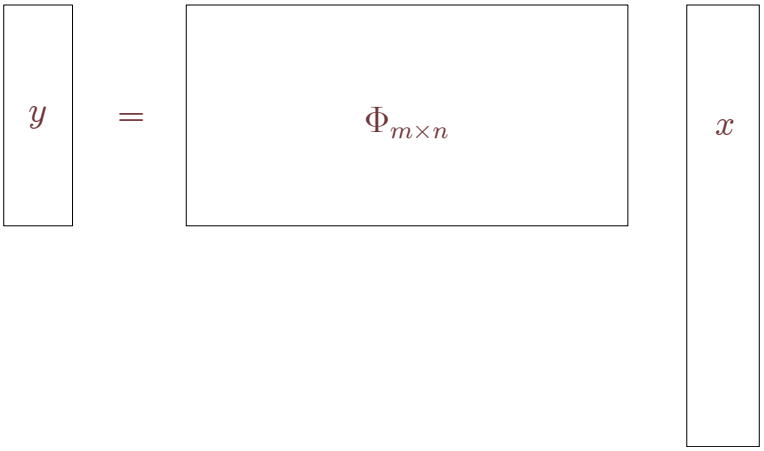

Figure 1: Compressed Sensing model: Matrix form of under-determined system of linear equations. Here Φ is an $m \times n$ measurement matrix, x is an n dimensional and y is an m dimensional vector, where $m \ll n$.

the $(n-m)$-dimensional hyperplane $\mathcal{H} := \{\hat{x} : \Phi\hat{x} = y\} = \mathcal{N}(\Phi) + x$ in $\mathbb{R}^n$, which corresponds to the null space $\mathcal{N}(\Phi)$ of Φ translated by the true solution x.

2.2 ℓ_1 norm minimization

We usually solve such inverse problems (2.1) by using *least norm* procedure, given by

$$\text{minimize } \|\tilde{x}\| \qquad \text{subject to} \qquad \Phi\tilde{x} = y, \tag{2.2}$$

where $\|\cdot\|$ is some norm defined on $\mathbb{R}^n$. The most common approach to solve (2.1) involves Euclidean or ℓ_2 norm in (2.2), which we will call *minimum ℓ_2 norm reconstruction*, given as following optimization problem,

$$\text{minimize } \|\tilde{x}\|_2^2 \qquad \text{subject to} \qquad \Phi\tilde{x} = y, \tag{P$_2$}$$

for which an analytical solution exists and is given by

$$\hat{x} = \Phi^*(\Phi\Phi^*)^{-1}y,$$

where Φ^* is adjoint of the measurement matrix Φ. In most cases least-norm solution with Euclidean norm gives very poor results. However, if we know a priori that our signal is *sparse*, then we can do much much better and that is the main tenet of

CS theory. It tells us about how and when will exact recovery of a sparse signal be possible from its compressed measurements!

If we know a priori that our original signal x is sparse, then a natural choice would be to find a vector in $\mathcal{H}$ with least non-zero entries. This leads to the following optimization problem

$$\text{minimize } \|\tilde{x}\|_0 \qquad \text{subject to} \qquad \Phi\tilde{x} = y, \tag{P$_0$}$$

where $\|\tilde{x}\|_0 := |\text{supp}(\tilde{x})|$, $\|\cdot\|_0$ is not a norm by definition, but is known as ℓ_0 *quasi-norm*. Unfortunately, this problem requires a search through all possible subsets of Φ, looking for a sparse subset which satisfies exact solution property. In general, it is not feasible to solve such a combinatorial optimization problem and it is known to be NP-hard [40]. Fortunately we can do *almost* as well with a relaxed convex problem which minimizes ℓ_1 norm of x, which we will call *minimum ℓ_1 norm reconstruction*, given as following optimization problem,

$$\text{minimize } \|\tilde{x}\|_1 \qquad \text{subject to} \qquad \Phi\tilde{x} = y \tag{P$_1$}$$

where $\|\tilde{x}\|_1 = \sum |\tilde{x}_i|$ is the ℓ_1 norm of $\tilde{x}$. (P$_1$) is also known as Basis Pursuit (BP) in literature [17]. This is a convex program and can be recast as a linear program [17] and solved using any modern optimization technique [4].

Why ℓ_1 and not ℓ_2? We can get some intuition for why minimum ℓ_1 norm reconstruction performs better than minimum ℓ_2 norm reconstruction in the case of sparse signals by looking at the geometry of ℓ_1 and ℓ_2 balls, as depicted in Figure 2. The figure demonstrates an oversimplified situation in $\mathbb{R}^2$ but we will have to, somehow, visualize a similar structure in higher dimensions [2]. Part(a) illustrates an ℓ_1 ball of radius r in $\mathbb{R}^2$. First note that it is anisotropic; it is "pointy" along the axes (compared to the standard Euclidean or ℓ_2 ball, which is spherical and thus completely isotropic). Let us assume our original signal $x \in \mathbb{R}^2$ has only one non-zero element

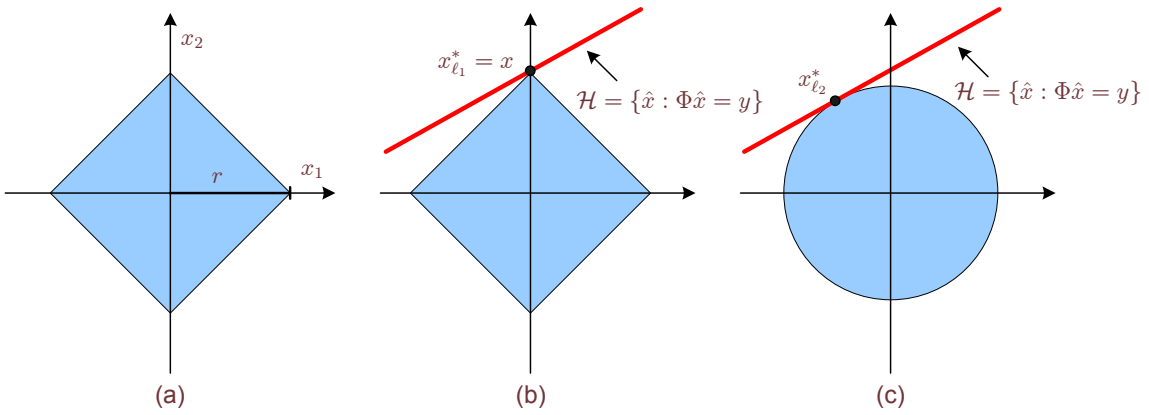

Figure 2: Geometry of ℓ_1 minimization. (a) an ℓ_1 ball of radius r. (b) $\mathcal{H}$ represents a hyperplane where all possible solutions to $\Phi x = y$, and ℓ_1 ball meets the hyperplane at some vertex. (c) an ℓ_2 ball touching $\mathcal{H}$ at point closest to the origin.

in its second coordinate, say $x = [0 \;\; r]^T$, and we take only one measurement of that. In part(b), $\mathcal{H}$ denotes the line in $\mathbb{R}^2$ (will be a hyperplane in higher dimensions) on which all possible solutions $\hat{x}$ lie which satisfy $\Phi\hat{x} = y$. Orientation of this hyperplane will depend on the measurement(s) and original signal x. To visualize how ℓ_1 reconstruction works, imagine that we start to inflate ℓ_1 ball from origin until it touches the hyperplane $\mathcal{H}$ at some point, vector at that point will be the solution to (P_1). In part(b) it is denoted by $x^*_{\ell_1}$, which is the exact solution in our case. Now if we replace ℓ_1 norm with ℓ_2 norm, and do the same thing as inflate ℓ_2 ball from origin until it hits $\mathcal{H}$ at some point. In part(c) it is denoted as $x^*_{\ell_2}$, which is the point on $\mathcal{H}$ closest to the origin, not even close to the sparse solution. In higher dimensions this difference will be more pronounced. Since ℓ_2 ball is perfectly uniform along all dimensions, with very high probability ℓ_2 norm solution will live away from the coordinate axes. Whereas the pointy structure of ℓ_1 ball will enable it to touch hyperplane $\mathcal{H}$ at a point near the coordinate axes, which is precisely where sparse vector is located.

2.3 Uncertainty principles

There is still one piece missing here; what conditions must be obeyed by the measurements? Here kicks in the incoherence! As we discussed earlier that if our signal

is sparse in one basis, we should take measurements in some basis where it is well spread out. Equivalently our sensing function should be spread out in the basis where signal is sparse, which means representation and measurement bases should be highly incoherent. That's how compressed sensing works: *sample a sparse signal in some incoherent basis and reconstruct it using ℓ_1 minimization.*

To see this let's consider an example in time and frequency domain. Suppose our signal $x \in \mathbb{R}^n$ is supported on an unknown subset T in time domain, where $|T| \ll n$. It is shown in [13] that x can be reconstructed from $m \gtrsim |T| \cdot \log n$ Fourier coefficients chosen independently at random by solving (P_1), where rows of Φ would correspond to the complex sinusoids at selected frequencies. This tells us that if our signal is S-sparse in time we can recover it from about $S \cdot \log n$ Fourier measurements at randomly chosen frequencies. Note that we cannot expect to recover x from less than n *time* samples, as the support set T is unknown and we will hit zero locations most of the time. But as it is well known that a signal which is concentrated in time will be well spread out in frequency; "time-frequency uncertainty principle" [25, 27], so we can under-sample a time domain sparse signal in frequency and still get enough information to recover the original signal perfectly. And similarly if a signal is S-sparse in frequency (i.e., composed of just S sinusoids), we can recover it from about $S \cdot \log n$ time samples at random locations.

In [11, 14] Candés and Tao quantified the conditions which any measurement matrix should obey. For that purpose they introduced the notion of uniform uncertainty principle (UUP). The UUP essentially requires that the $m \times n$ sensing matrix Φ obeys "restricted isometry property" (RIP). Let $T \subset \{1, \ldots, n\}$ and Φ_T be the $m \times |T|$ matrix obtained by extracting the columns of Φ indexed by elements in T, then S-restricted isometry constant δ_S of Φ is defined as the smallest quantity such that

$$(1 - \delta_S)\|c\|_2^2 \leq \|\Phi_T c\|_2^2 \leq (1 + \delta_S)\|c\|_2^2 \tag{2.3}$$

for all subsets T with $|T| \leq S$ and coefficient sequence $(c_j)_{j\in T}$. The UUP or RIP essentially requires δ_S not to be very close to 1. In which case, every subset of columns with cardinality less than S will approximately behave like an orthonormal system. Hence, the name "near isometry" or "restricted isometry". Exact reconstruction of any sparse signal is closely tied to the UUP condition (2.3). Loosely speaking, if our signal x is supported on a set T, then we want sub-matrix Φ_T to be very well-conditioned, otherwise reconstruction will be impossible. This becomes obvious if we write (2.3) in the following equivalent form

$$(1 - \delta_S) \leq \lambda_{\min}(\Phi_T^* \Phi_T) \leq \lambda_{\max}(\Phi_T^* \Phi_T) \leq (1 + \delta_S). \tag{2.4}$$

2.4 Sparse signal recovery from compressed measurements

Here we will mention some theoretical results about recovery of sparse and compressible signals from compressed measurements. In addition to that we will discuss some optimization models which are used for signal recovery from noisy measurements.

2.4.1 Recovery from noiseless measurements

Let us first consider the simplest case where our signal x is S-sparse and our measurements are noise free, i.e., $y = \Phi x$. It is shown in [14] that if Φ obeys restricted isometry at level S such that[1] $\delta_{2S} + \delta_{3S} < 1$, then the solution x^* to (P_1) is unique and it is exact, i.e., $x^* = x$. For details see [9, 14, 15].

The result about exactly sparse signals can be extended to compressible signals as well. Although in that case we cannot reconstruct original signal but we can get something very close to the best S-term approximation. Let x_S be the best S-term approximation of a signal x. It turns out that if Φ obeys the same conditions of restricted isometry i.e., $\delta_{2S} + \delta_{3S} < 1$, then the solution x^* to (P_1) obeys

$$\|x^* - x\|_2 \leq C \cdot \frac{\|x - x_S\|_1}{\sqrt{S}},$$

[1] (P_0) will have unique solution whenever $\delta_{2S} < 1$.

where C is a small constant. For details see [9, 15].

2.4.2 Stable recovery from noisy measurements

In any practical system we cannot discount the presence of noise. This noise can be due to imperfections in the measuring devices or it can be due to quantization of measurements. So to be more realistic we need to assume that we have some inaccurate measurements and our task is to recover signal from them. Let us consider the following model for that purpose

$$y = \Phi x + z, \tag{2.5}$$

where z is some unknown stochastic or deterministic error term. Since our matrix Φ is rank deficient and our system is an underdetermined affine system, here perturbations can easily blow up the inversion process. For stable recovery, we want that small perturbations in measurements should not blow up our actual solutions, i.e., small changes in the measurements should result in small changes in the solution. For that purpose we need to modify our recovery mechanism as well. In order to take noise into account, we need to add some data fidelity constraint along with ℓ_1 norm minimization in our recovery algorithm. One possible formulation is given as

$$\text{minimize } \|\tilde{x}\|_1 \qquad \text{subject to} \qquad \|\Phi\tilde{x} - y\|_2 \leq \epsilon. \tag{$\mathrm{P_{QC}}$}$$

The program ($\mathrm{P_{QC}}$) can be recast as a second order cone program (SOCP) [4] and solved using some convex optimization package [16, 29, 33]. There exist some variants on ($\mathrm{P_{QC}}$) in statistics community, especially in the areas of model/variable selection and regression analysis, e.g., "basis pursuit denoising" (BPDN) [17] and "least absolute shrinkage and selection operator" (LASSO) [52].

It is shown in [9] that solving ($\mathrm{P_{QC}}$) will recover sparse signals with an error at most proportional to the noise level. If the matrix Φ obeys $\delta_{3S} + 3\delta_{4S} < 2$, and noise

term z has bounded energy $\|z\|_2 \leq \epsilon$. The solution x^* to ($\mathrm{P_{QC}}$) obeys

$$\|x^* - x\|_2 \leq C_1 \cdot \epsilon + C_2 \cdot \frac{\|x - x_S\|_1}{\sqrt{S}}. \tag{2.6}$$

This result is not limited to exactly sparse signals but also gives error estimate for the case of compressible signals; for S-sparse signals second term on right side of (2.6) will become zero.

In order to see how well we do with this, consider the same model in (2.5) with S-sparse signal $x \in \mathbb{R}^n$ and Gaussian noise $z \in \mathbb{R}^m$, where $z_1, \ldots, z_m$ are i.i.d. Gaussian with mean zero and variance σ^2. Since z is a vector with i.i.d. Gaussian entries, $\|z\|_2^2$ will be distributed as a chi-squared with m degrees of freedom. If x is an S-sparse vector, then from (2.6) the solution x^* to ($\mathrm{P_{QC}}$) will obey

$$\|x - x^*\|_2^2 \leq C \cdot m\, \sigma^2 \tag{2.7}$$

with high probability.

Here we have m measurement variables, each contaminated with noise of variance σ^2, so $m\sigma^2$ is not bad. But since we know that only S elements in our unknown vector x are non-zero, so can we reduce error term to near something like $S\sigma^2$? It is important to note here that if we know the support of signal beforehand then we can achieve an error estimate of about $S\sigma^2$, so in a sense this is what we expect from an *ideal estimator* [6, 10].

2.4.3 Dantzig selector

In [10] Candès and Tao showed that we can do almost as good as an ideal estimator by using a new type of estimator they called Dantzig Selector (DS). The Dantzig selector estimates x by solving the following optimization program

$$\text{minimize } \|\tilde{x}\|_1 \quad \text{subject to} \quad \|\Phi^*(\Phi\tilde{x} - y)\|_\infty \leq \lambda \cdot \sigma \tag{DS}$$

for some $\lambda > 0$, where $\|\cdot\|_\infty$ is the ℓ_∞ or supremum norm defined as $\|\tilde{x}\|_\infty = \max_i |\tilde{x}_i|$. The program (DS) is convex and can be recast as a linear program [16]. The main

result for accuracy of Dantzig selector is as follows: Assume columns of Φ are unit norm, and choose $\lambda = (1+t^{-1})\sqrt{2\log n}$ with some $t > 0$ in (DS). If x is highly sparse and Φ obeys UUP, then with high probability, the solution x^* to (DS) obeys

$$\|x - x^*\|_2^2 \leq C \cdot 2\log n \cdot \left(\sigma^2 + \sum_{i=1}^{n} \min(x_i^2, \sigma^2)\right), \tag{2.8}$$

where C is some known constant (for details see [10]). An extension of this result is available for nearly sparse signals. This result says that Dantzig selector achieves the squared error within a logarithmic factor of the mean squared error of an ideal estimator; which we could achieve only with perfect information about which coordinates are non-zero and which of them are above noise level. Here the logarithmic factor in the error term can be considered as a penalty we pay for not knowing the support of x in advance. In the same spirit, recently some other results are proposed as well which promise near ideal estimation with Lasso, for details see [3, 7, 39]

As a final remark, note that Dantzig selector is analogous to soft thresholding [26]. If Φ is an orthonormal $n \times n$ matrix, then (DS) is equivalent to soft thresholding (in the Φ domain), which can be written as

$$\text{minimize } \|x\|_1 \quad \text{subject to} \quad \|\tilde{y} - x\|_\infty \leq \lambda \cdot \sigma \quad \text{where} \quad (\tilde{y} = \Phi y).$$

So loosely speaking, Dantzig selector is equivalent to soft thresholding or shrinkage for "incomplete observations".

2.5 Random Sensing

In previous sections all the results we have seen depend on some conditions related to UUP. In this section we will discuss some measurement matrices which obey UUP.

Ideally we would like to design the $m \times n$ measurement matrix Φ which would obey UUP at sparsity level S with as fewer measurements, m as possible. The design of matrix Φ involves choice of n vectors in m dimensional space such that any subset of S such vectors be nearly orthogonal. All the known constructions of measurement

matrices which provably obey UUP are based on principle of random sensing i.e., their entries are either instances of i.i.d. random variable or independently picked at random from some basis which is incoherent to the representation basis. Here are some examples of matrices which obey UUP with high probability.

***Gaussian measurements*:** The entries of Φ are chosen independently from the NORMAL distribution with mean zero and variance $1/m$. If

$$m \geq C \cdot S \cdot \log(n/S), \tag{2.9}$$

then Φ will obey UUP with probability at least $1 - O(e^{-\gamma n})$ for some $\gamma > 0$ [11].

***Bernoulli measurement*:** The entries of Φ are independently chosen from symmetric Bernoulli distribution: $\mathbb{P}(\Phi_{kj} = \pm 1/\sqrt{m}) = 1/2$. If condition in (2.9) holds then Φ will obey UUP with probability $1 - O(e^{-\gamma n})$ for some $\gamma > 0$ [11].

***Fourier measurements*:** Suppose Φ is a partial Fourier matrix with its rows chosen uniformly at random, re-normalize it so that the columns have unit norm (otherwise we will have an extra m/n factor in (2.3)). If

$$m \geq C \cdot S \cdot \log^{\alpha} n,$$

then with high probability, UUP holds for partial Fourier matrix, where $\alpha \geq 4$ [11, 47].

***Incoherent measurements*:** Suppose that the measurement matrix Φ is obtained by selecting m rows from an $n \times n$ orthonormal matrix U uniformly at random and re-normalized so that each column has unit norm. If our signal $x = \Psi\alpha$ is S-sparse in basis Ψ (i.e., $\|\alpha\|_0 \leq S$), where Ψ is known to be incoherent with Φ. Then we can extend the ideas from Fourier measurements to any incoherent measurements [6, 8]. If

$$m \geq C \cdot \mu^2(\Phi, \Psi) \cdot S \cdot \log^4 n,$$

then $\Phi' = \Phi\Psi$ will obey UUP, where $\mu(\Phi, \Psi) := \sqrt{n} \max_{i,j} |\langle \varphi_i, \psi_j \rangle|$ is mutual coherence. So smaller the coherence (higher the incoherence), smaller the required number of measurements.

In the case of incoherent bases, we can write our measurements as

$$y = \Phi x = \Phi\Psi\alpha = \Phi'\alpha,$$

where $\Phi' = \Phi\Psi$ is a composite $m \times n$ matrix. One way to recover x is by finding the expansion coefficients vector α with minimum ℓ_1 norm which is also consistent with measurement data, which gives us following optimization problem

$$\text{minimize } \|\tilde{\alpha}\|_1 \qquad \text{subject to} \qquad \Phi'\tilde{\alpha} = y \tag{2.10}$$

This type of reconstruction is also known as *Synthesis-based ℓ_1 minimization.*

Almost all the results about matrices obeying UUP are based on concepts related to deviation of the smallest and largest singular values of random matrices [11]. An important thing to note here is that the ambient dimension of signal, n doesn't affect the required number of measurements, m very much, instead m just depend logarithmically on n, hence dictated mainly by the sparsity level of signal, S.

CHAPTER III

PRIMAL DUAL PURSUIT ALGORITHM

3.1 Introduction

The main result of compressed sensing says that under some conditions on sparsity and incoherence, a sparse signal can be recovered exactly from a small number of linear measurements. All the convex programs, we have seen for signal recovery, solve a minimization problem where they minimize ℓ_1 norm of sparse coefficients under some data fidelity constraints. These problems can be recast as linear program (LP) or second order cone program (SOCP), and can be solved using tools from convex optimization like Interior point methods e.g., in ℓ_1-MAGIC [16]. Some excellent advancements made in the area of interior point methods have made possible the polynomial time solution of these problems [42]. Although the interior point methods are really fast compared to classical simplex methods but still for some large scale problems e.g., imaging, these methods tend to be slow. The main computational burden in interior point method comes from solving a large system of linear equations for every Newton iteration, which can be solved using conjugate gradient (CG) methods [48] (see [16] for details). Although several different methods have been developed for ($\mathrm{P_{QC}}$) over the last few years, e.g., gradient projection based signal recovery (GPSR) [29], large scale ℓ_1 regularized least squares [36] and path following or homotopy continuation based methods [28, 43], but not much work has been done yet to develop fast methods for Dantzig selector.

Here we will present a *homotopy* based algorithm to solve the Dantzig selector. This new *iterative* algorithm, which we call "Primal Dual Pursuit" or "PD-pursuit", is very fast compared to the optimization methods currently being used for the Dantzig

selector. In this algorithm we follow a *primal-dual homotopy* approach to compute the solution. We successively update the primal and dual variables at every step until we reach the solution point. And at every step our update requires just a few matrix-vector multiplications.

3.2 Dantzig selector and its dual

Let us consider the following system model

$$y = Ax + e,$$

where $x \in \mathbb{R}^n$ is our unknown signal, $y \in \mathbb{R}^m$ is the measurement vector, A is the $m \times n$ measurement/sensing matrix and $e \in \mathbb{R}^m$ is the error in our measurements. The Dantzig selector (DS) solves the following convex program

$$\underset{\tilde{x}}{\text{minimize}} \ \|\tilde{x}\|_1 \quad \text{subject to} \quad \|A^T(A\tilde{x} - y)\|_\infty \leq \epsilon. \tag{Primal-DS}$$

Dual program to (Primal-DS) is:

$$\underset{\lambda}{\text{maximize}} \ -(\epsilon\|\lambda\|_1 + \langle \lambda, A^T y \rangle) \qquad \text{subject to} \qquad \|A^T A\lambda\|_\infty \leq 1, \tag{Dual-DS}$$

where $\lambda \in \mathbb{R}^n$ is our dual vector. A detailed description for derivation of dual problem is given in Appendix A. By Slater's condition, we know that strong duality holds for this problem [4]. So in our case we have the strong duality between our primal and dual objective functionals, i.e., at any primal-dual solution pair (x^*, λ^*), the objective functionals in (Primal-DS) and (Dual-DS) will be equal:

$$\|x^*\|_1 = -(\epsilon\|\lambda^*\|_1 + \langle \lambda^*, A^T y \rangle),$$

or equivalently we can write it as

$$\|x^*\|_1 + \epsilon\|\lambda^*\|_1 = -\langle x^*, A^T A\lambda^* \rangle + \langle \lambda^*, A^T(Ax^* - y) \rangle. \tag{3.1}$$

The complementary slackness condition tells us that whenever primal constraint is active, the corresponding element in dual vector will be non-zero and vice versa,

similarly for the dual constraints and elements in primal vector. So using (3.1) and the feasibility conditions for our primal and dual problems;

$$\|A^T(Ax - y)\|_\infty \leq \epsilon \tag{3.2a}$$

$$\|A^T A\lambda\|_\infty \leq 1, \tag{3.2b}$$

we get the following optimality conditions which must be obeyed by any solution pair (x^*, λ^*) to (Primal-DS) and (Dual-DS):

K1. $A_{\Gamma_\lambda}^T(Ax^* - y) = \epsilon z_\lambda$

K2. $A_{\Gamma_x}^T A\lambda^* = -z_x$

K3. $|a_\gamma^T(Ax^* - y)| < \epsilon \quad$ for all $\quad \gamma \in \Gamma_\lambda^c$

K4. $|a_\gamma^T A\lambda^*| < 1 \quad$ for all $\quad \gamma \in \Gamma_x^c$

where Γ_x and Γ_λ is the support of x^* and λ^*, z_x and z_λ are the sign sequences of x^* and λ^* on their respective supports. Using these four conditions we will derive our "path following" or "homotopy" algorithm for Dantzig selector. These conditions can also be derived by using KKT conditions, where we will have to use subgradients, since functionals are not smooth.

3.3 Homotopy for Dantzig selector

Let us start with the description of homotopy method. The general principle in homotopy methods is to trace a complete solution path by starting from an artificial initial value and iteratively moving towards a desired solution by gradually adjusting the homotopy parameter(s). For our method this implies following a path traced by a sequence of primal-dual pair (x_k, λ_k) towards the solution point while reducing ϵ_k to the desired ϵ. We start from $x_0 = 0$ and a large $\epsilon_0 > \|A^T y\|_\infty$, find the primal and dual solution set (x_k, λ_k) at every k^{th} step for corresponding ϵ_k and terminate when $\epsilon_k \to \epsilon$, and consequently $x_k \to x^*$, where x^* is the solution to (Primal-DS). This

can be considered as solving a series of relaxed optimization problems, where we start with a large feasible domain (at very large ϵ_k) and shrink the feasibility domain (by reducing ϵ_k), while updating our solution. The exact Dantzig selector path is ensured by maintaining the optimality conditions (K1-K4) at each point along the homotopy path.

It is obvious from the optimality conditions (K1-K4) that at any point (x_k, λ_k) on the solution path, corresponding to a particular ϵ_k in (Primal-DS), active primal constraints in (3.2a) give us the sign and support of dual vector λ in (K1) and active dual constraints in (3.2b) give the sign and support of primal vector x in (K2). Whereas all the primal and dual constraints corresponding to zero elements in dual and primal vectors respectively will hold with strictly inequality. We can consider this as a consequence of the *complementary slackness* property in KKT conditions [4].

In our algorithm we will use the four optimality conditions to update the supports and signs of primal vector x and dual vector λ at every ϵ_k while reducing it towards the desired value ϵ. Now how does our method work; we start with $x_0 = 0$, $\lambda_0 = 0$ and a large $\epsilon_0 > \|A^T y\|_\infty$, then we update vectors x and λ in such a way that at every step at most one element enters or leaves the support of each x and λ. Along the homotopy path we will have some *critical* values of ϵ_k when the supports of x and/or λ will change. So we move in a particular direction until there is some change in the support of either x or λ, at which point we update the supports and find new directions for primal and dual vectors. At every step we first update the sign and support of λ using primal feasibility conditions (3.2a) and once we have that information we update the sign and support of x using the dual feasibility conditions (3.2b). As we proceed along the homotopy path, by updating primal-dual pair (x_k, λ_k), we are essentially shrinking the primal constraints i.e., $\epsilon_{k+1} < \epsilon_k$. If we update x and λ along the correct directions with proper step sizes, such that the optimality conditions (K1-K4)

are maintained at every step, we will eventually reach the solution x^* to (Primal-DS) as $\epsilon_k \to \epsilon$.

3.4 Main algorithm

Assume $k = 1, 2, \ldots$ denotes our homotopy step count and let x_k and λ_k denote x and λ vectors at k^{th} step respectively. Let us write the optimality conditions at any given ϵ_k as

$$A_{\Gamma_\lambda}^T(Ax_k - y) = \epsilon_k z_\lambda \tag{3.3a}$$

$$A_{\Gamma_x}^T A\lambda_k = -z_x \tag{3.3b}$$

$$|a_\gamma^T(Ax_k - y)| < \epsilon \quad \text{for all } \gamma \in \Gamma_\lambda^c \tag{3.3c}$$

$$|a_\gamma^T A\lambda_k| < 1 \quad \text{for all } \gamma \in \Gamma_x^c, \tag{3.3d}$$

where (x_k, λ_k) is the primal-dual solution pair for DS at ϵ_k, Γ_x and Γ_λ is the support of x_k and λ_k, z_x and z_λ are the sign sequences of x_k and λ_k on their respective supports. The algorithm works in an iterative fashion, computing x_k and λ_k at every step $k = 1, 2, \ldots$ while maintaining the optimality conditions in (3.3). A pseudocode for the algorithm is given in Algorithm 1 on page 28.

The algorithm starts with an initial solution x_1 and λ_1, and operates in an iterative way computing/updating the new supports Γ_x, Γ_λ and signs z_x, z_λ and updating x and λ on each step. We can divide our algorithm into two main parts:

- Primal Update
- Dual Update

In primal update phase we update the primal vector and primal constraints which give us support and sign of dual vector. In dual update phase we update the dual vector and dual constraints which give us support and sign of primal vector to be used in next primal update phase.

3.4.1 Primal update

In primal update step we update the primal vector x and primal constraints which give us new support Γ_λ and sign sequence z_λ for dual vector λ. First we compute the update direction for primal vector ∂x as defined in (3.6) and set $x_{k+1} = x_k + \delta\partial x$ with some step size $\delta > 0$ as described in (3.4). Now as we move in the direction ∂x primal constraints will change, with all the active constraints shrinking by a factor δ in magnitude. So by increasing step size δ we can encounter two scenarios; either a new element can enter the support of λ (i.e., a new primal constraint can become active) or an element from within the support of x can shrink to zero. So the path taken by our primal vector will be continuous except at certain critical values of ϵ_k where supports of primal and/or dual vectors change. So x will follow a piecewise linear path w.r.t. changing ϵ_k. We choose our step size δ depending on which occurs first, as described in (3.4). The value of active constraints at the new point gives us $\epsilon_{k+1} = \epsilon_k - \delta$.

$$|a_\gamma^T(Ax_{k+1} - y)| = \epsilon_{k+1} \qquad \text{for all } \gamma \in \Gamma_\lambda \tag{3.4a}$$

$$|a_\gamma^T(Ax_{k+1} - y)| \leq \epsilon_{k+1} \qquad \text{for all } \gamma \in \Gamma_\lambda^c \tag{3.4b}$$

$$|\underbrace{a_\gamma^T(Ax - y)}_{p_k(\gamma)} + \delta \underbrace{a_\gamma^T A\partial x}_{d_k(\gamma)}| \leq \epsilon_k - \delta \qquad \text{for all } \gamma \in \Gamma_\lambda^c \tag{3.4c}$$

$$|p_k(\gamma) + \delta d_k(\gamma)| \leq \epsilon_k - \delta \qquad \text{for all } \gamma \in \Gamma_\lambda^c \tag{3.4d}$$

$$\delta^+ = \min_{i \in \Gamma_\lambda^c} \left(\frac{\epsilon_k - p_k(i)}{1 + d_k(i)}, \frac{\epsilon_k + p_k(i)}{1 - d_k(i)} \right) \tag{3.4e}$$

$$i^+ = \arg\min_{i \in \Gamma_\lambda^c} \left(\frac{\epsilon_k - p_k(i)}{1 + d_k(i)}, \frac{\epsilon_k + p_k(i)}{1 - d_k(i)} \right) \tag{3.4f}$$

$$\delta^- = \min_{i \in \Gamma_x} \left(\frac{-x_k(i)}{\partial x(i)} \right) \tag{3.4g}$$

$$i^- = \arg\min_{i \in \Gamma_x} \left(\frac{-x_k(i)}{\partial x(i)} \right) \tag{3.4h}$$

$$\delta = \min(\delta^+, \delta^-). \tag{3.4i}$$

Here minimum is taken over positive arguments only. Let us call the index corresponding to δ^+ as i^+ and index corresponding to δ^- as i^-. So either i^+ enters the support of λ (if $\delta^+ < \delta^-$) or i^- leaves the support of x (if $\delta^+ > \delta^-$) and we update the supports and signs accordingly.

3.4.2 Dual update

In dual update we follow exactly the same procedure as done in primal update with an extra check that we have to make on sign of update direction vector $\partial\lambda$ if an element leaves support of x during primal update. So similar to primal update, here also we compute the update direction as defined in (3.7) and set $\lambda_{k+1} = \lambda_k + \theta\partial\lambda$, where $\theta > 0$ is the step size as described in (3.5). Now here dual constraints do not shrink like primal constraints, instead at some critical value of ϵ_k, either a new constraint just becomes active which gives us new support of x, or an element from within the support of λ goes to zero. So we accordingly select the largest step size θ as given in (3.5).

$$|a_\nu^T A\lambda_{k+1}| = 1 \qquad \text{for all } \nu \in \Gamma_x \tag{3.5a}$$

$$|a_\nu^T A\lambda_{k+1}| \leq 1 \qquad \text{for all } \nu \in \Gamma_x^c \tag{3.5b}$$

$$|\underbrace{a_\nu^T A\lambda_k}_{a_k(\nu)} + \theta \underbrace{a_\nu^T A\partial\lambda}_{b_k(\nu)}| \leq 1 \qquad \text{for all } \nu \in \Gamma_x^c \tag{3.5c}$$

$$|a_k(\nu) + \theta b_k(\nu))| \leq 1 \qquad \text{for all } \nu \in \Gamma_x^c \tag{3.5d}$$

$$\theta^+ = \min_{j \in \Gamma_x^c} \left(\frac{1 - a_k(j)}{b_k(j)}, \frac{1 + a_k(j)}{-b_k(j)} \right) \tag{3.5e}$$

$$j^+ = \arg\min_{j \in \Gamma_x^c} \left(\frac{1 - a_k(j)}{b_k(j)}, \frac{1 + a_k(j)}{-b_k(j)} \right) \tag{3.5f}$$

$$\theta^- = \min_{j \in \Gamma_\lambda} \left(\frac{-\lambda(j)}{\partial\lambda(j)} \right) \tag{3.5g}$$

$$j^- = \arg\min_{j \in \Gamma_\lambda} \left(\frac{-\lambda(j)}{\partial\lambda(j)} \right) \tag{3.5h}$$

$$\theta = \min(\theta^+, \theta^-). \tag{3.5i}$$

Here again the minimum is taken over positive arguments only. Let us call the index corresponding to θ^+ as j^+ and index corresponding to θ^- as j^-. So either j^+ enters the support of x (if $\theta^+ < \theta^-$) or j^- leaves the support of λ (if $\theta^+ > \theta^-$) and we update the supports and signs accordingly.

3.4.3 Geometry of primal and dual constraints

The primal constraints exhibit the "shrinkage" behavior as ϵ is reduced. At some facet determined by primal and dual supports $(\Gamma_x, \Gamma_\lambda)$, if we increase the step size δ from zero to some small value $\Delta\delta$, the active constraints shrink uniformly by the same amount becoming $\epsilon - \Delta\delta$. And as we continue to increase δ, either a new constraint gets active or an element from x shrinks to zero, and that is where we need to update the supports. In contrast dual constraints will stay almost same throughout the facet determined by $(\Gamma_x, \Gamma_\lambda)$ except at end points. So dual constraints will change only when a new element enters the support of x or an existing element leaves the support of x. Therefore, in a sense dual constraints only determine the direction or facet to move on, and in order to move along that facet we need to change the step size δ in primal update (up or down). And as we move along some facet, we hit some vertex at a critical value of ϵ_k, and going beyond that point needs some change in the supports $(\Gamma_x, \Gamma_\lambda)$, so dual vector tells us which direction or facet to take next. This is also an indication that the path taken by primal vector x is piecewise linear whereas path taken by dual vector λ is piecewise constant (i.e., only indicator of a facet).

3.4.4 Update directions

At every k^{th} step, we need to compute new directions ∂x and $\partial\lambda$ for primal and dual vectors x and λ respectively. We can compute the update directions using the optimality conditions in (3.3). Let us assume we are at some point (x_k, λ_k) corresponding to ϵ_k, with primal dual supports Γ_x, Γ_λ, and sign sequences z_x, z_λ. First for primal vector; we want to change x in a direction ∂x which causes maximum

change in ϵ_k. So from (3.3a) we get our primal update direction ∂x as

$$\partial x = \begin{cases} -(A_{\Gamma_\lambda}^T A_{\Gamma_x})^{-1} z_\lambda & \text{on } \Gamma_x \\ 0 & \text{elsewhere} \end{cases} \tag{3.6}$$

Using ∂x as defined above we update the support and sign of the dual vector as described in (3.4). After that we update the support and sign of primal vector as described in (3.5) by changing our dual vector in the dual update direction $\partial\lambda$ defined as

$$\partial\lambda = \begin{cases} -z_\gamma (A_{\Gamma_x}^T A_{\Gamma_\lambda})^{-1} A_{\Gamma_x}^T a_\gamma & \text{on } \Gamma_\lambda \\ z_\gamma & \text{on } \gamma \\ 0 & \text{elsewhere} \end{cases} \tag{3.7}$$

where a_γ is the γth column of A corresponding to index of element last entered in the support of λ, z_γ is the sign of γth primal active constraint which infact is the sign of the new element in λ. We derived (3.7) using (3.3b). To see this, assume that the new support of λ at $(k+1)$th step will be $\Gamma'_\lambda = [\Gamma_\lambda \;\; \gamma]$ and for dual vector update we pick an arbitrary direction vector $\partial\lambda'$ which is supported on set Γ'_λ. In order for this to be a valid direction for λ, it must obey optimality conditions (3.3a) at k^{th} step with $\lambda' = \lambda_k + \theta\partial\lambda'$ for a small step size $\theta > 0$. We can write (3.3b) with λ' as

$$\begin{aligned} A_{\Gamma_x}^T A\lambda' &= -z_x \\ A_{\Gamma_x}^T A\lambda_k + \theta A_{\Gamma_x}^T A\partial\lambda' &= -z_x \\ A_{\Gamma_x}^T A_{\Gamma_\lambda} u + A_{\Gamma_x}^T a_\gamma v &= 0, \end{aligned}$$

where u is the restriction of $\partial\lambda'$ on Γ_λ and v is the value of $\partial\lambda'$ on γth index. Since we already know that the sign of γth element in λ_{k+1} will be z_γ, so we can write $v = cz_\lambda$, where c is some positive number. This gives us

$$A_{\Gamma_x}^T A_{\Gamma_\lambda} u + cz_\gamma A_{\Gamma_x}^T a_\gamma = 0,$$

or equivalently

$$u = -cz_\gamma (A_{\Gamma_x}^T A_{\Gamma_\lambda})^{-1} A_{\Gamma_x}^T a_\gamma, \quad v = cz_\gamma, \tag{3.8}$$

which is precisely what is given in (3.7) with $c = 1$ (and it doesn't make a difference because the factor c will be adjusted in the final step size θ). This also implies that we can pick any γ from within the supp(λ) as long as the new Gram matrix $A_{\Gamma_x}^T A_{\Gamma_\lambda}$ is invertible.

3.4.5 More on primal-dual update

One additional check we need in the dual update is related to the situation when an element is removed from support of x in the primal update step. Let Γ_x and Γ_λ be the support of x and λ respectively. If an element corresponding to index γ_x is removed from the support of x in the primal step, our new support becomes $\Gamma_{x_1} = \Gamma_x \backslash \gamma_x$. We can rewrite $\Gamma_\lambda = [\Gamma_{\lambda_1} \ \gamma_\lambda]$, where γ_λ is an element picked from the current support of λ for which the new matrix $A_{\Gamma_{x_1}}^T A_{\Gamma_{\lambda_1}}$ does not become singular. This can be easily checked by looking at inverse of the Schur complement for $A_{\Gamma_{x_1}}^T A_{\Gamma_{\lambda_1}}$ in $A_{\Gamma_x}^T A_{\Gamma_\lambda}$, which should not become zero. Let us write inverse of $A_{\Gamma_x}^T A_{\Gamma_\lambda}$ as

$$(A_{\Gamma_x}^T A_{\Gamma_\lambda})^{-1} := \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}.$$

This can be seen in (5.3) that for $(A_{\Gamma_{x_1}}^T A_{\Gamma_{\lambda_1}})^{-1}$ to exist we need Q_{22} to be non-zero (see section 3.4.7 and section 5.2.2 for further details). For stability purpose, a better way is to choose γ_λ in the following way. Pick the column in matrix $(A_{\Gamma_x}^T A_{\Gamma_\lambda})^{-1}$ corresponding to index of γ_x in Γ_x. Find the entry in that column with largest absolute value, and select the element at respective row index in Γ_λ as γ_λ. This will ensure that Q_{22} will not become zero.

In order to find the update direction $\partial\lambda$ we will use $\Gamma_{x_1}, \Gamma_{\lambda_1}$ and γ_λ in place of Γ_x, Γ_λ and γ respectively in (3.7). Since in this case $\lambda_k(\gamma_\lambda) \neq 0$ and we do not know what will be the sign of $\lambda_{k+1}(\gamma_\lambda)$, therefore, we will have uncertainty in sign of $\partial\lambda$

(as c is not necessarily positive in (3.8)). In order to resolve this we will check if $\{\text{sign}(a_k(\gamma_x)) = \text{sign}(b_k(\gamma_x))\}$ (which means dual constraint corresponding to γ_x will be violated for any value of $\theta > 0$ (3.5d)), then we first flip the sign of $\partial\lambda$ in (3.7) and then compute θ.

3.4.6 Initialization

We start with $x_0 = 0$, $\lambda_0 = 0$, $\Gamma_x = [\ ]$, $\Gamma_\lambda = [\ ]$, $z_x = [\ ]$, $z_\lambda = [\ ]$. Choose ϵ_1 large enough such that there is only one primal constraint active i.e., pick $\epsilon_1 = \|A^T y\|_\infty$. This gives us the support of λ for ϵ_1; $\Gamma_\lambda = \{\gamma\}$, where γ corresponds to the index of the only active primal constraint (3.3a) i.e., $\gamma = \{i : a_i^T y = \epsilon_1\}$, where a_i denotes the i^{th} column of A and $z_\gamma = \text{sign}(a_\gamma^T y)$. Using this information we find the support[1] and sign of x as $\Gamma_x = \{\nu\}$ and z_x by using the update direction (3.7), which will be $\partial\lambda = z_\gamma \delta_\gamma$, where δ_γ is a vector with all zero entries except at index γ where it is 1. After the first step we will have $x_1 = 0$ and $\lambda_1 = \theta_1 \partial\lambda$ (θ_1 as defined in (3.4)). $\Gamma_x = [\nu]$, $\Gamma_\lambda = [\gamma]$, $z_x = -\text{sign}(A_{\Gamma_x}^T A\lambda_1)$, $z_\lambda = \text{sign}(-A_{\Gamma_\lambda}^T y)$. For next steps we follow the same procedure of primal and dual update as described before.

3.4.7 Degenerate cases

As a side note, it is important to know that in some extreme cases this scheme will not work. For example, it is possible that at some critical value of ϵ_k more than one inactive constraints become active, or more than one active constraints become inactive or some constraints become active and some become inactive. In short it is possible under some controlled settings (e.g., if we take very few Bernoulli measurements of a signal whose entries are $\{\pm q\}$ for some constant q), it can happen that the primal and/or dual vectors change at more than one locations simultaneously. This means that we have some degeneracy at that particular vertex, and more than one constraints are touching it. So one way to resolve this problem is to store all the

[1] support of x and λ will be same for first step

Algorithm 1 Primal Dual Pursuit Algorithm for Dantzig Selector

Initialize $x_k, \lambda_k, \Gamma_x, \Gamma_\lambda, z_x, z_\lambda$ and ϵ_k for $k = 1$ as described in section 3.4.6
repeat
  $k \leftarrow k + 1$
  Primal update:
  compute the primal update direction ∂x as in (3.6)
  compute p_k, d_k and δ as in (3.4)
  $x_{k+1} = x_k + \delta \partial x$
  $\epsilon_{k+1} = \epsilon_k - \delta$
  if $\delta = \delta^-$ **then**
    $\Gamma_x \leftarrow \Gamma_x \setminus i^-$ {remove i^- from supp(x) and update Γ_x}
    $\tilde{\Gamma}_\lambda = \Gamma_\lambda$ {store the current Γ_λ in a dummy variable}
    $\Gamma_\lambda \leftarrow \Gamma_\lambda \setminus \gamma$ {select an index γ from supp(λ) and remove it from Γ_λ}
    $z_\gamma = z_\lambda(\gamma)$ {treat γ as the new element to supp(λ)}
    update z_x, z_λ {update sign sequences on updated supports}
  else
    $\tilde{\Gamma}_\lambda = \Gamma_\lambda \cup \{i^+\}$ {store i^+ but do not update Γ_λ}
    $z_\lambda = \text{sign}[A^T_{\tilde{\Gamma}_\lambda}(Ax_{k+1} - y)]$ {update z_λ}
    $\gamma = i^+$
    $z_\gamma = z_\lambda(\gamma)$
  end if
  Dual update:
  compute the dual update direction $\partial\lambda$ as in (3.7)
  compute a_k and b_k as in (3.5)
  if $\delta = \delta^-$ && $\text{sign}[a_k(i^-)] = \text{sign}[b_k(i^-)]$ **then**
    $\partial\lambda \leftarrow -\partial\lambda$ {a check needed due to uncertainty in sign}
    $b_k \leftarrow -b_k$ {flip the sign of $\partial\lambda$ and in turn b_k}
  end if
  compute θ as in (3.5)
  $\lambda_{k+1} = \lambda_k + \theta \partial\lambda$
  if $\theta = \theta^-$ **then**
    $\Gamma_\lambda \leftarrow \tilde{\Gamma}_\lambda \setminus j^-$ {remove j^- from supp(λ) and update Γ_λ}
    update z_λ {update sign sequence on updated support}
  else
    $\Gamma_x \leftarrow \Gamma_x \cup \{j^+\}$ {add j^+ to supp(x) and update Γ_x}
    $\Gamma_\lambda \leftarrow \tilde{\Gamma}_\lambda$ {set Γ_λ to supp(λ) determined in Primal update}
    $z_x = \text{sign}[A^T_{\Gamma_x} A\lambda_{k+1}]$ {update z_x}
  end if
until $\epsilon_{k+1} \leq \epsilon$

candidates in some stack and work with one of them at a time, and then perform primal or dual updates in cycle. We have tried it and this works in many cases.

In some cases it can happen that some of the incoming or outgoing columns make the gram matrix $A_{\Gamma_x}^T A_{\Gamma_\lambda}$ or $A_{\Gamma_\lambda}^T A_{\Gamma_x}$ singular, so under such situations we need to verify that our Gram matrices do not become singular by adding or removing the chosen column, and if they are becoming singular we need to choose some other column from the stack. If we are adding new columns, we can check the singularity of the updated Gram matrix easily by checking Schur complement. Let us consider the following $K+1 \times K+1$ matrix

$$D = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

where A_{11} is a $K \times K$ matrix, A_{12} is a K-dimensional column vector, A_{21} is a K-dimensional row vector and A_{22} is a scalar. $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ is known as Schur complement of A_{11} in D. A well known result from linear algebra tells us that $\det(D) = \det(A_{11})\ \det(S)$ [4]. So if A_{11} is full rank then we just need to check that Schur complement doesn't become zero by update. On the other hand if we are removing some columns, we just need to check that inverse of the Schur complement for the remaining matrix does not become zero. This can also be seen in (5.3), where we want $Q_{22} = 1/S$ to be non-zero for inverse to exist.

In case all the columns in our stack make matrices singular, we cannot go further; so take it easy, just `break` from the loop and `return` the most recent results. Although we expect that this degeneracy will not appear in any practical situation.

3.5 Comparison with DASSO

While working on primal-dual pursuit algorithm we learnt about another homotopy based algorithm for the Dantzig selector; DASSO [35]. Although both methods are designed in the same spirit, i.e., to trace the entire homotopy path towards the solution

of the Dantzig selector, but formulation and update schemes of the two algorithms follow different approaches. Here we will highlight some advantages we think our scheme has over DASSO.

One major advantage of our scheme over DASSO is the *direct* computation of update directions ∂x and $\partial \lambda$. In DASSO, the authors solve a simple optimization routine to find the update direction ∂x, for which one needs at least $2n$ matrix-vector multiplications with a K dimensional square matrix at every step, where K denotes size of the supports at that step. In contrast to this, with our primal dual formulation we get direct formulae for the update directions, each of which can be computed with a single matrix-vector multiplication with a K dimensional square matrix.

As we discussed in section 3.4.7, we can have some degenerate cases under some extreme conditions. Although degenerate cases will not appear very often in any practical situation, but in case we have one, our proposed algorithm behaves very efficiently. Using the direct formulae for the update directions we can find the primal and dual supports much easily and can be updated as discussed in section 3.4.7. Whereas in DASSO, the proposed optimization routine with $d > 1$ new elements can become highly cumbersome, where one needs to find a pair of d columns from $K \times 2n$ matrix which maximize the cost, for details see [35].

In addition to this, our primal-dual formulation helps in getting some further insights into the connections between Lasso and Dantzig Selector as discussed in section 4.4.

CHAPTER IV

ANALYSIS OF PRIMAL DUAL PURSUIT

Here we will discuss the *S-step solution property* of our primal dual pursuit algorithm under certain sparsity conditions on the signal. By the *S-step solution property* we mean that we can recover an S sparse signal from compressed measurements in at most S steps of primal dual update as described in Algorithm 1 on page 28. We will first discuss the basic conditions required for this property to hold. Then we will establish those conditions for Gaussian and Bernoulli measurement matrices. Later we will discuss the S-step solution property for general incoherent ensembles. In the end we will draw some connections between the homotopy methods for Lasso and Dantzig selector, and give conditions under which they both follow identical path.

4.1 S-step solution property

Let $x \in \mathbb{R}^n$ be an S-sparse signal supported on index set Γ with sign sequence z, A be the $m \times n$ measurement matrix, $y = Ax$ be the noiseless measurement vector in a much lower dimensional space ($m \ll n$). For reference we will state the Dantzig selector optimization problem, its dual and optimality conditions discussed in section 3.2 again.

$$\underset{\tilde{x}}{\text{minimize}} \ \|\tilde{x}\|_1 \quad \text{subject to} \quad \|A^T(A\tilde{x} - y)\|_\infty \leq \epsilon. \tag{Primal-DS}$$

$$\underset{\lambda}{\text{maximize}} \ -(\epsilon\|\lambda\|_1 + \langle \lambda, A^T y \rangle) \qquad \text{subject to} \qquad \|A^T A\lambda\|_\infty \leq 1, \tag{Dual-DS}$$

A pair (x^*, λ^*) is a solution set to the problem (Primal-DS) and (Dual-DS) if and only if the following four conditions hold.

K1. $A_{\Gamma_\lambda}^T(Ax^* - y) = \epsilon z_\lambda$

K2. $A_{\Gamma_x}^T A\lambda^* = -z_x$

K3. $\|A_{\Gamma_\lambda^c}^T(Ax^* - y)\|_\infty < \epsilon$

K4. $\|A_{\Gamma_x^c}^T A\lambda^*\|_\infty < 1$

The following lemma gives us sufficient conditions for the existence of an optimal set (x^*, λ^*) which satisfies the above mentioned optimality conditions (K1-K4) and hence solve (Primal-DS) for any given value of ϵ. Not only that but it also gives an explicit value of dual vector λ^* corresponding to x^* at any facet corresponding to Γ_x.

Lemma 4.1. *Let $x_0 \in \mathbb{R}^n$ be supported on a set Γ with sign sequence $z := sign\,(x_\Gamma)$, and $y = Ax_0 \in \mathbb{R}^m$ be the measurement vector with $m \ll n$. Suppose A satisfies the following three conditions with Γ and z:*

H1. *A_Γ is full rank*

H2. $\|A_{\Gamma^c}^T A_\Gamma (A_\Gamma^T A_\Gamma)^{-1} z\|_\infty < 1$

H3. $sign[(A_\Gamma^T A_\Gamma)^{-1} z] = z$

Take $\lambda^* \in \mathbb{R}^n$ as

$$\lambda^* = \begin{cases} -(A_\Gamma^T A_\Gamma)^{-1} z & on\ \Gamma \\ 0 & on\ \Gamma^c \end{cases}$$

and

$$x_\epsilon^* = x_0 + \epsilon\lambda^*. \tag{4.1}$$

Then for all ϵ in the range

$$0 \le \epsilon \le \epsilon_{\text{crit}} = \min_{\gamma \in \Gamma} \left(\frac{x_0(\gamma)}{-\lambda(\gamma)} \right),$$

$(x_\epsilon^*, \lambda^*)$ will be a solution set to (Primal-DS) *and* (Dual-DS).

Proof. If A_Γ is full rank (H1), then λ^* is well-defined. We will show that this proposed pair $(x_\epsilon^*, \lambda^*)$ meet the criteria (K1-K4) above. First, note that $\Gamma_\lambda = \Gamma_x = \Gamma$.

If (H3) holds, then also $z_\lambda = -z$, $z_x = z$ and (K2) is satisfied. In addition, this makes (H2) the same as (K4). Finally with x_ϵ^* as in (4.1),

$$A^T(Ax_\epsilon^* - y) = \epsilon A^T A\lambda^* = -\epsilon A^T A_\Gamma (A_\Gamma^T A_\Gamma)^{-1} z,$$

and so (H2) implies (K1) and (K3). □

4.1.1 Dantzig shrinkability

Under the conditions (H1-H3), we can interpret the solution of Dantzig Selector x_ϵ^* as a "shrinkage" of the original signal x_0. As ϵ increases, the magnitudes of all the non-zero entries in x_ϵ^* will decrease. But instead of decreasing at the same rate (as in soft thresholding), the decrease at component γ is proportional to $\lambda(\gamma)$. ϵ_{crit} is the value of ϵ for which one of the components shrinks to zero and leaves the support.

It is natural to ask then, if this "shrinkage" property holds for this $x^*_{\epsilon_{\text{crit}}}$ supported on $\Gamma_1 \subset \Gamma$. If so, can we continue the process until $x^* = 0$. To make this more precise, we will call x_0 "Dantzig shrinkable" with respect to A if the following procedure terminates in `Success`:

1. Set $k = 0$, $\Gamma_0 = \text{supp}(x_0)$, and $z_0 = \text{sign}(x_0)$ restricted to Γ_0.

2. If $x_k = 0$, return `Success`.

3. Check that

$$\|A_{\Gamma_k^c}^T A_{\Gamma_k}(A_{\Gamma_k}^T A_{\Gamma_k})^{-1} z\|_\infty < 1$$
$$\text{sign}[(A_{\Gamma_k}^T A_{\Gamma_k})^{-1} z] = z$$

   If either condition fails, break and return `Failure`.

4. Set

$$\lambda_k = \begin{cases} -(A_{\Gamma_k}^T A_{\Gamma_k})^{-1} z_k & \text{on} \quad \Gamma_k \\ 0 & \text{on} \quad \Gamma_k^c \end{cases},$$

$$\epsilon_{k+1} = \min_{\gamma \in \Gamma_k} \left(\frac{x_k(\gamma)}{-\lambda_k(\gamma)} \right),$$

$$x_{k+1} = x_k + \epsilon_{k+1} \lambda_k,$$

$$\gamma'_{k+1} = \arg\min_{\gamma \in \Gamma_k} \left(\frac{x_k(\gamma)}{-\lambda_k(\gamma)} \right),$$

$$\Gamma_{k+1} = \Gamma_k \backslash \gamma'_{k+1},$$

$$z_{k+1} = z_k \text{ restricted to } \Gamma_{k+1}.$$

5. Set $k \leftarrow k+1$, and return to step 2.

As x_0 is being shrunk to zero in the procedure above, it is following the solution path to (Primal-DS) for increasing value of ϵ, as demonstrated in the next lemma.

Lemma 4.2. *Suppose x_0 is Dantzig shrinkable, and define $x_k, \lambda_k, \epsilon_k$ as above. Given any $0 \leq \epsilon \leq \|A^T y\|_\infty$, let K be the largest integer such that*

$$\epsilon \geq \sum_{k=0}^{K-1} \epsilon_{k+1} =: E_K.$$

Then the solution to (Primal-DS) *will be exactly*

$$x_\epsilon^* = x_K + (\epsilon - E_K)\lambda_K.$$

Proof. Set $\delta = \epsilon - E_K$. Since x_0 is Dantzig shrinkable, we have that

$$x_\epsilon^* = x_0 + \sum_{k=0}^{K-1} \epsilon_{k+1} \lambda_k + \delta \lambda_K$$

and so

$$A^T(Ax_\epsilon^* - y) = A^T A \left(\sum_{k=0}^{K-1} \epsilon_{k+1} \lambda_k + \delta \lambda_K \right).$$

Since

$$(a_\gamma^T A \lambda_k)(\gamma) = -z(\gamma) \quad \text{for } \gamma \in \Gamma_K \text{ and for all } \quad k = 1, \ldots, K,$$

and

$$|a_\gamma^T A\lambda_k| < 1 \quad \text{for } \gamma \in \Gamma_K^c \text{ and for all} \quad k = 1, \ldots, K-1,$$

we have

$$\left| a_\gamma^T A \left(\sum_{k=0}^{K-1} \epsilon_{k+1}\lambda_k + \delta\lambda_K \right) \right| < (E_K + \delta) = \epsilon \quad \text{for all} \gamma \in \Gamma_K^c.$$

Hence (K1-K4) are satisfied. □

Given these definition, it is not hard to see that if x_0 is S-sparse and Dantzig shrinkable with respect to A, then the Primal-Dual Pursuit algorithm will terminate in S steps. To see this, note that (H2) at every step ensures that only true elements enter the support of signal, and (H3) ensures that any element which has entered the support does not leave it.

4.1.2 Sufficient conditions for Dantzig shrinkability

Let us assume that A_Γ is full rank i.e., condition (H1) is satisfied. Now let us define $G = I - A_\Gamma^T A_\Gamma$. Condition (H2) and (H3) will be satisfied if $\|G\| < 1$ and

$$\max_{\gamma \in \{1,\ldots,n\}} |\langle (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma, z\rangle| < 1, \tag{4.2}$$

with

$$Y_\gamma = \begin{cases} A_\Gamma^T a_\gamma & \gamma \in \Gamma^c \\ A_\Gamma^T a_\gamma - \mathbf{1}_\gamma & \gamma \in \Gamma \end{cases}, \tag{4.3}$$

where a_γ is the column of A indexed by γ, and $\mathbf{1}_\gamma$ is a vector which is equal to 1 at γ and zero elsewhere. To see this, first note that (H2) is same as

$$\max_{\gamma \in \Gamma^c} |\langle a_\gamma, A_\Gamma (A_\Gamma^T A_\Gamma)^{-1} z\rangle| < 1.$$

Whenever $\|G\| < 1$, the Neumann series $\sum_{\ell=0}^{\infty} G^l$ converges to the inverse $(I-G)^{-1}$. So we can write $(A_\Gamma^T A_\Gamma)^{-1} z$ in the following way

$$(A_\Gamma^T A_\Gamma)^{-1} z = (I-G)^{-1} z = \sum_{\ell=0}^{\infty} G^\ell z = \left(z + \sum_{\ell=1}^{\infty} G^\ell z \right),$$

and condition (H3): $\text{sign}[(A_\Gamma^T A_\Gamma)^{-1} z] = z$, will be satisfied if

$$\left\| \sum_{\ell=1}^{\infty} G^\ell z \right\|_\infty < 1. \tag{4.4}$$

We can rewrite (4.4) as

$$\begin{aligned}
\max_{\gamma \in \Gamma} \left| \langle \mathbf{1}_\gamma, \sum_{\ell=1}^{\infty} G^\ell z \rangle \right| &= \max_{\gamma \in \Gamma} \left| \langle \sum_{\ell=1}^{\infty} G^\ell \mathbf{1}_\gamma, z \rangle \right| \\
&= \max_{\gamma \in \Gamma} \left| \langle \sum_{\ell=1}^{\infty} G^{\ell-1} g_\gamma, z \rangle \right| \\
&= \max_{\gamma \in \Gamma} \left| \langle (A_\Gamma^T A_\Gamma)^{-1} g_\gamma, z \rangle \right|
\end{aligned}$$

where g_γ is the column of G indexed by γ, $g_\gamma = \mathbf{1}_\gamma - A_\Gamma^T a_\gamma$. The first equality above comes from the self-adjointness of G, the second comes from simple fact that $g_\gamma = G\mathbf{1}_\gamma$, and the third because $\sum_{\ell \geq 1} G^{\ell-1} = \sum_{\ell \geq 0} G^\ell = (A_\Gamma^T A_\Gamma)^{-1}$.

In order to show that an S-sparse signal x supported on Γ with sign sequence z is Dantzig shrinkable w.r.t. the measurement matrix A, we need to show that conditions (H1-H3) are obeyed at every k^{th} step for $k = 1, 2, \ldots, S$. So we need A_{Γ_k} to be full rank, its respective matrix $G = I - A_{\Gamma_k}^T A_{\Gamma_k}$ to have $\|G\| < 1$ at every k^{th} step and we need to ensure that (4.2) is obeyed at every step with appropriate Y_γ.

Now we will establish the Dantzig shrinkability conditions (H1-H3) for Gaussian, Bernoulli and Incoherent ensembles.

4.2 Dantzig shrinkability with random matrices

Here we will establish Dantzig shrinkability conditions (H1-H3) for two special types of measurement ensembles 1) Gaussian and 2) Bernoulli distribution.

1. Independently select each entry of the measurement matrix A to be i.i.d. Gaussian with zero mean and variance $1/m$.

2. Independently select each entry of A to be $\pm 1/\sqrt{m}$ with equal probability.

The uniform uncertainty principle (UUP) or restricted isometry property (RIP) as discussed in section 2.3 tells us that any sub-matrix of A (as described above), consisting of S columns indexed by set Γ, will be well conditioned (2.4). In [11] Candès and Tao established that random matrices having Gaussian or Bernoulli distribution as defined above obey (2.3) with overwhelming large probability if $m \gtrsim S \cdot \log n$. This establishes (H1) in Lemma 4.1 with appropriate support size. Theorem 4.3 shows that for "most" S-sparse signals supported on an *arbitrary* set Γ, condition (H2) and (H3) will hold.

Theorem 4.3. *Generate an $m \times n$ matrix A whose entries are chosen independently at random from any of the two distributions 1) Gaussian or 2) Bernoulli as described above with*

$$m \geq C \cdot S^2 \cdot \log n \tag{4.5}$$

where $C > 0$ is a known constant. Given an arbitrary signal $x \in \mathbb{R}^n$ supported on subset Γ where $|\Gamma| \leq S$. Let z be the sign sequence of x on coordinates given by Γ. Then with probability exceeding $1 - O(n^{-\beta})$, we will have

$$\max_{\gamma \in \{1,\ldots,n\}} |\langle (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma, z \rangle| < 1$$

with Y_γ as defined in (4.3).

The strategy for proving Theorem 4.3 is as follows. We will bound the size (ℓ_2 norm) of Y_γ, which when combined with the UUP will give us bound on the size of $w_\gamma := (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma$. Then we will use Cauchy-Schwarz inequality to show that $|\langle w_\gamma, z \rangle|$ will be less than 1. Without loss of generality we will assume that Γ corresponds to the first S columns of A.

For the proof of Theorem 4.3 we will first consider the Gaussian case, the argument for Bernoulli distribution is very similar, as will be shown later.

4.2.1 The Gaussian ensemble

Consider the $m \times n$ measurement matrix A with i.i.d NORMAL$(0, 1/m)$ entries. We will first consider the case for

$$Y_\gamma = A_\Gamma^T a_\gamma \qquad \text{for all} \qquad \gamma \in \Gamma^c.$$

We can write this vector Y_γ as a sequence of inner products $\{\langle a_j, a_\gamma \rangle\}_{j=1}^S$ as

$$Y_\gamma = \begin{bmatrix} \langle a_1, a_\gamma \rangle \\ \langle a_2, a_\gamma \rangle \\ \vdots \\ \langle a_S, a_\gamma \rangle \end{bmatrix} =: \begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_S \end{bmatrix}.$$

Since we know from the construction of matrix A that all its entries are i.i.d. Gaussian with zero mean and variance $1/m$. Here we will use the fact that all the elements of sequence $\{Q_j\}_{j=1}^S$ will also be random variables following Gaussian distribution with zero mean and variance $\|a_\gamma\|^2/m$ *for the fixed vector* a_γ as shown below.

$$\mathbb{E}(Q_j) = \mathbb{E}\left(\sum_{i=1}^m a_{ij} a_{i\gamma}\right) = \sum_{i=1}^m \mathbb{E}(a_{ij}) a_{i\gamma} = 0$$

and

$$\begin{aligned} \mathbb{E}(Q_j^2) &= \mathbb{E}\left(\left(\sum_{i=1}^m a_{ij} a_{i\gamma}\right)^2\right) \\ &= \mathbb{E}\left(\sum_{i=1}^m (a_{ij} a_{i\gamma})^2 + \sum_{l=1}^m \sum_{k=1}^m 2 a_{lj} a_{l\gamma} a_{kj} a_{k\gamma}\right) \\ &= \sum_{i=1}^m \mathbb{E}(a_{ij}^2) a_{i\gamma}^2 + \sum_{l=1}^m \sum_{k=1}^m 2\mathbb{E}(a_{lj})\mathbb{E}(a_{kj}) a_{l\gamma} a_{k\gamma} \\ &= \frac{\|a_\gamma\|^2}{m} \end{aligned}$$

A well-known Gaussian tail bound (using Chernoff bound) [e.g., 49, pg. 215] gives us the following inequality for a Gaussian random variable X with zero mean and variance σ^2

$$\mathbb{P}[\ |X| > \epsilon\ ] \leq \exp(-\epsilon^2/2\sigma^2). \tag{4.6}$$

Applying the Gaussian tail bound (4.6) on the inner product between any two independent columns, a_j, a_γ where $j \neq \gamma$, we get the following inequality

$$\mathbb{P}[|\langle a_j, a_\gamma \rangle| > \epsilon] \leq 2\exp(-\epsilon^2 m/2\|a_\gamma\|^2). \tag{4.7}$$

Here we need some bound for the value of $\|a_\gamma\|^2$, where we know that $\|a_\gamma\|^2$ will be a random variable following chi-square distribution. Suppose that the random variable Z is defined as

$$Z = \sum_{i=1}^{m} X_i^2,$$

where $X_i, i = 1, 2, \ldots, m$, are statistically independent and identically distributed Gaussian random variables with zero mean and variance $1/m$. The Chernoff bound for concentration of any such chi-square distributed random variable Z around its mean is given as [18]

$$\mathbb{P}\left[|Z - \mathbb{E}[Z]| > \delta\right] \leq 2\exp(-m\delta^2/8). \tag{4.8}$$

So for any given column vector a_γ from matrix A, where $\mathbb{E}[\|a_\gamma\|^2] = 1$, we get the following result about concentration of its squared norm around 1,

$$\mathbb{P}\left[\|a_\gamma\|^2 > 1 + \delta\right] \leq 2\exp(-m\delta^2/8),$$

and we have a slightly better bound for $\mathbb{P}[\|a_\gamma\|^2 < 1 - \delta]$. Now in order to bound the probability for squared norm of *all* columns of A we can use the union bound to get the following result

$$\mathbb{P}\left[\max_{\gamma \in \{1,\ldots,n\}} \|a_\gamma\|^2 > 1 + \delta\right] \leq 2n\exp(-m\delta^2/8), \tag{4.9}$$

using this bound for any given $\|a_\gamma\|^2$ together with the Gaussian tail bound on the inner product $|\langle a_j, a_\gamma \rangle|$ in (4.7) we get the following inequality

$$\mathbb{P}[|\langle a_j, a_\gamma \rangle| > \epsilon \;\big|\; \|a_\gamma\|^2 \leq 1 + \delta] \leq 2\exp(-\epsilon^2 m/2(1+\delta)).$$

In order to establish such bound for inner products between *all* the pairs of two columns from matrix A we will use the union bound on $\binom{n}{2} \leq n^2$ possible pairs.

$$\mathbb{P}\left[\max_{\substack{j,\gamma\in\{1,\dots,n\}\\ j\neq\gamma}} |\langle a_j, a_\gamma\rangle| > \epsilon\right] \leq 2\binom{n}{2}\exp(-\epsilon^2 m/2(1+\delta)) + \mathbb{P}\left[\max \|a_\gamma\|^2 > 1+\delta\right],$$

If we take $\delta = \epsilon$ and if $\epsilon < 1$, we get

$$\begin{aligned}\mathbb{P}\left[\max_{\substack{j,\gamma\in\{1,\dots,n\}\\ j\neq\gamma}} |\langle a_j, a_\gamma\rangle| > \epsilon\right] &\leq 2\binom{n}{2}\exp(-\epsilon^2 m/2(1+\epsilon)) + 2n\exp(-m\epsilon^2/8)\\ &< 2n^2\exp(-\epsilon^2 m/4) + 2n\exp(-m\epsilon^2/8).\end{aligned}$$

In order to satisfy condition (4.2) we will use $\epsilon \sim \sqrt{(8(\beta+1)\log n)/m}$. This is detailed as follows:

$$\begin{aligned}\mathbb{P}\left[\max_{\substack{j,\gamma\in\{1,\dots,n\}\\ j\neq\gamma}} |\langle a_j, a_\gamma\rangle| > \sqrt{\frac{8(\beta+1)\log n}{m}}\right] &\leq 2n^2\exp(-2(\beta+1)\log n)\\ &\quad + n\exp(-(\beta+1)\log n)\\ &= 2/n^{2\beta} + 2/n^\beta\\ &\leq 3/n^\beta. \qquad (4.10)\end{aligned}$$

whenever $n^\beta > 2$.

Now let's consider the case for

$$Y_\gamma = A_\Gamma^T a_\gamma - \mathbf{1}_\gamma \qquad \text{for all} \qquad \gamma \in \Gamma.$$

We can write this vector Y_γ as

$$Y_\gamma = \begin{bmatrix}\langle a_1, a_\gamma\rangle\\ \langle a_2, a_\gamma\rangle\\ \vdots\\ \langle a_\gamma, a_\gamma\rangle - 1\\ \vdots\\ \langle a_S, a_\gamma\rangle\end{bmatrix} =: \begin{bmatrix}Q_1\\ Q_2\\ \vdots\\ Q_\gamma\\ \vdots\\ Q_S\end{bmatrix}.$$

Here we can use the already established bound (4.10) (for the inner product between any two *distinct* columns of matrix A) for any element of sequence $\{Q_j\}_{j=1}^S$ except when $j = \gamma$. For $Q_\gamma = \langle a_\gamma, a_\gamma \rangle - 1$, we can use the chi-square concentration as in (4.8). And using similar argument we will get the same bound in (4.10).

To summarize, we have the following result for *all* elements $\{Q_j\}_{j=1}^S$ of Y_γ for all $\gamma \in \{1, \ldots, n\}$,

$$\mathbb{P}\left[\max_{j,\gamma} |Q_j| > \sqrt{\frac{8(\beta+1)\log n}{m}} \right] \leq 3/n^\beta \tag{4.11}$$

whenever $n^\beta > 2$.

From the discussion above we can immediately bound the ℓ_2 norm of Y_γ as defined in (4.3) for the Gaussian case by the following inequality,

$$\mathbb{P}\left[\max_{\gamma \in \{1,\ldots,n\}} \|Y_\gamma\|_2 > \sqrt{\frac{8S(\beta+1)\log n}{m}} \right] \leq 3/n^\beta.$$

Uniform uncertainty principle tells us that for Gaussian matrix A if $m \gtrsim S \cdot \log n$, then with probability $O(1 - n^{-\beta})$

$$1/2 \leq \lambda_{\min}(A_\Gamma^T A_\Gamma) \leq \lambda_{\max}(A_\Gamma^T A_\Gamma) \leq 3/2,$$

which is equivalent to the following condition about spectral norm of $(A_\Gamma^T A_\Gamma)^{-1}$,

$$2/3 \leq \lambda_{\min}(A_\Gamma^T A_\Gamma)^{-1} \leq \lambda_{\max}(A_\Gamma^T A_\Gamma)^{-1} \leq 2. \tag{4.12}$$

Define $w_\gamma = (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma$. Using the UUP condition (4.12) we get the following inequality

$$\mathbb{P}\left[\sup_{\substack{\gamma \in \{1,\ldots,n\} \\ |\Gamma| \leq S}} \|w_\gamma\|_2 > \sqrt{\frac{16S(\beta+1)\log n}{m}} \;\middle|\; \lambda_{\max}(A_\Gamma^T A_\Gamma)^{-1} < 2 \right] \leq 3/n^\beta$$

or equivalently

$$\mathbb{P}\left[\sup_{\gamma \in \{1,\ldots,n\}} \|w_\gamma\|_2 > \sqrt{\frac{16S(\beta+1)\log n}{m}} \right] \leq O(1/n^\beta). \tag{4.13}$$

So we can write (4.3) in the following way,

$$\begin{aligned} |\langle (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma, z\rangle| &= |\langle w_\gamma, z\rangle| \\ &\leq \|w_\gamma\|_2 \|z\|_2 \\ &\leq \sqrt{\frac{16S(\beta+1)\log n}{m}}.\sqrt{S}, \end{aligned}$$

where the first inequality comes from Cauchy-Schwarz inequality and second inequality comes from (4.13) under consideration that $\|z\|_2 \leq S$. So in order to satisfy (4.2) in Theorem 4.3 we need the following constraint on m

$$\boxed{m > S^2 16(\beta+1)\log n} \tag{4.14}$$

This proves Theorem 4.3 for Gaussian case with $C \sim 16(\beta+1)$.

4.2.2 The Bernoulli ensemble

Now we will analyze the case for Bernoulli distributed measurement matrix. The argument is very similar to the Gaussian case. Let A be the $m \times n$ matrix with its entries independently chosen from $\{\pm 1/\sqrt{m}\}$ We will first consider the case for

$$Y_\gamma = A_\Gamma^T a_\gamma \qquad \text{for all} \qquad \gamma \in \Gamma^c,$$

We can write this vector Y_γ as a sequence of inner products $\{\langle a_j, a_\gamma\rangle\}_{j=1}^S$

$$Y_\gamma = \begin{bmatrix} \langle a_1, a_\gamma\rangle \\ \langle a_2, a_\gamma\rangle \\ \vdots \\ \langle a_S, a_\gamma\rangle \end{bmatrix} =: \begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_S \end{bmatrix}$$

We can write every element vector Y_γ as

$$Q_j = \langle a_j, a_\gamma\rangle = \sum_{i=1}^m a_{ij} a_{i\gamma} = \sum_{i=1}^m q_i$$

Since all the entries of matrix A are chosen independently, each element of the sequence $\{q_i\}_{i=1}^m$ will be a random variable with its value $\pm 1/m$ with equal probability. And its clear that $\mathbb{E}[Q_j] = \sum_{i=1}^m \mathbb{E}[a_{ij}]a_{i\gamma} = 0$ for all $j \in \{1, \ldots, S\}$

Here we will use the well known Hoeffding's inequality to bound the value of inner product between any two independent columns a_j, a_γ. Hoeffding's inequality [37] is given in the following lemma,

Lemma 4.4 (Hoeffding's tail inequality)**.** *Let $X_1, \ldots, X_m$ be independent bounded random variables such that X_i falls in the interval $[a_i, b_i]$ with probability one. If we define $S_m = \sum_{i=1}^m X_i$, then for any $t > 0$ we have*

$$\mathbb{P}[|S_m - \mathbb{E}S_m| \geq t] \leq e^{-2t^2/\sum_{i=1}^m (b_i - a_i)^2}.$$

Using Hoeffding's inequality we get the following bound on the value of any given element of vector Y_γ

$$\mathbb{P}[|\langle a_j, a_\gamma \rangle| \geq t] \leq 2e^{-mt^2/2}. \tag{4.15}$$

Now using the same procedure we will use the union bound on right hand side to justify this bound for values of inner products between all $\binom{n}{2}$ possible pairs. To satisfy condition (4.2) we need $t \sim \sqrt{2(\beta+2)\log n}$. This is detailed as follows:

$$\mathbb{P}\left[|\langle a_j, a_\gamma \rangle| > \sqrt{\frac{2(\beta+2)\log n}{m}} \right] \leq 2\exp(-(\beta+2)\log n) \tag{4.16}$$

and after using the union bound we get the following bound for *all* inner product pairs

$$\mathbb{P}\left[\max_{\substack{j,\gamma \in \{1,\ldots,n\} \\ j \neq \gamma}} |\langle a_j, a_\gamma \rangle| > \sqrt{\frac{2(\beta+2)\log n}{m}} \right] \leq 2\binom{n}{2}.1/n^{\beta+2} \tag{4.17}$$

$$\leq 2/n^\beta. \tag{4.18}$$

For Bernoulli case we don't need to consider separately the case for

$$Y_\gamma = A_\Gamma^T a_\gamma - \mathbf{1}_\gamma \qquad \text{for all} \qquad \gamma \in \Gamma.$$

The already established bound in (4.17) will work for all elements $Q_j = \langle a_j, a_\gamma \rangle$ whenever $j \neq \gamma$. And for the case when $j = \gamma$, $Q_\gamma = \langle a_\gamma, a_\gamma \rangle - 1 = 0$. From the discussion above we can bound the ℓ_2 norm of Y_γ for any γ as follows:

$$\mathbb{P}\left[\max_{\gamma\in\{1,\ldots,n\}} \|Y_\gamma\|_2 > \sqrt{\frac{2S(\beta+2)\log n}{m}}\right] \leq 2/n^\beta. \tag{4.19}$$

Setting $w_\gamma = (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma$ and using the UUP argument we get

$$\mathbb{P}\left[\sup_{\gamma\in\{1,\ldots,n\}} \|w_\gamma\|_2 > \sqrt{\frac{4S(\beta+2)\log n}{m}}\right] \leq O(1/n^\beta). \tag{4.20}$$

Again we can write (4.3) in the following way,

$$\begin{aligned} |\langle (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma, z\rangle| &= |\langle w_\gamma, z\rangle| \\ &\leq \|w_\gamma\|_2 \|z\|_2 \\ &\leq \sqrt{\frac{4S(\beta+2)\log n}{m}}.\sqrt{S}, \end{aligned}$$

where the first inequality comes from Cauchy-Schwarz inequality and second inequality comes from (4.20) under consideration that $\|z\|_2 \leq S$. So in order to satisfy (4.2) in Theorem 4.3 we need the following constraint on m

$$\boxed{m > S^2 4(\beta+2)\log n} \tag{4.21}$$

This proves Theorem 4.3 for Bernoulli case with $C \sim 4(\beta+2)$.

4.3 Dantzig shrinkability with incoherent ensemble

Here we will extend the S-step solution property to any general incoherent ensemble. Let A be an $m \times n$ measurement matrix with unit length columns a_j, $\|a_j\|_2 = 1$. Let the coherence or mutual coherence of matrix A be given as

$$M(A) = \max_{i\neq j} |\langle a_i, a_j \rangle|.$$

Theorem 4.5 gives relation between the sparsity of a signal and mutual coherence of measurement matrix in order to recover an S-sparse signal in S steps. This result is a

more generalized version of similar earlier results on signal recovery using orthogonal matching pursuit (orthogonal matching pursuit (OMP)) [54] and ℓ_1 minimization as in basis pursuit [30]. Also similar result holds for recovery of S-sparse signal with Lasso [22]. And this should be no surprise because the optimality conditions and respective solution paths for Lasso and Dantzig will be exactly identical in some settings, and S-step convergence is one of them. We will discuss it further in section 4.4.

Theorem 4.5. *Let A be the incoherent matrix with mutual coherence M. Let x be an S-sparse signal supported on Γ with sign sequence z, and Y_γ be as defined in* (4.3). *If*

$$S \leq \frac{1}{2}\left(1 + \frac{1}{M}\right) \tag{4.22}$$

then

$$\max_{\gamma \in \{1,\ldots,n\}} |\langle (A_\Gamma^T A_\Gamma)^{-1} Y_\gamma, z\rangle| < 1.$$

Proof. This can be proved easily by following the arguments in [30, Theorem 3]. We need a slight modification due to our different definition of Y_γ. It will be exactly same for the case where $\gamma \notin \Gamma$. For the case where $\gamma \in \Gamma$ we can still use similar argument because the γth component of Y_γ will become zero (since $a_\gamma^T a_\gamma = 1$). □

The result given in Theorem 4.5 can be considered as an extension of result given in Theorem 4.3, where we bound each entry of Y_γ to be less than constant M.

In our analysis for all three cases, we have established the required conditions for the complete support Γ and sign sequence z on it. But it is obvious that the argument can be extended to any subset $\Gamma_k \subset \Gamma$. Since $A_{\Gamma_k}^T A_{\Gamma_k}$ will be a submatrix of $A_\Gamma^T A_\Gamma$, from Cauchy's interlacing theorem [34] we know that

$$\|G^k\| \leq \|G\| \quad \text{where } G^k := I - A_{\Gamma_k}^T A_{\Gamma_k}.$$

So A_{Γ_k} will be full rank and $A_{\Gamma_k}^T A_{\Gamma_k}$ will be well-conditioned. Similarly $\|Y_\gamma\|$ and $\|z\|$ corresponding to any subset $\Gamma_k \subset \Gamma$ will get smaller as we reduce size of the

support Γ_k. Since we are using Cauchy-Schwarz inequality, so conditions (H1-H3) will be obeyed for all the steps.

4.4 Connections between Dantzig selector and Lasso

Dantzig selector has captured some decent attention in statistics community in last few years. Recently there have been many discussions about the efficiency of Dantzig selector and its comparison to other well established model selection methods especially Lasso (for details see [10] and related discussions in the same volume). One important area researchers have been interested in is to find a fast algorithm to solve Dantzig selector with path following features. We hope that our primal-dual pursuit algorithm fulfills this purpose.

Another question being raised very often now is to find conditions under which Dantzig selector and Lasso behave similarly. Here we will try to answer some parts of this question; we will give some conditions under which Lasso and Dantzig selector follow identical path for the model, hence same performance. Lasso minimizes ℓ_1 norm of the parameters subject to a quadratic constraint on error, as given in ($\mathrm{P_{QC}}$). There is an equivalent penalized version of Lasso also known as basis pursuit denoising problem, given as

$$\underset{\tilde{x}}{\text{minimize}}\ \frac{1}{2}\|y - A\tilde{x}\|_2^2 + \epsilon\|\tilde{x}\|_1 \qquad \text{(Lasso)}$$

It is well known that dual to (Lasso) is given by [44]

$$\underset{\tilde{x}}{\text{minimize}}\ \frac{1}{2}\|A\tilde{x}\|_2^2 \quad \text{subject to} \quad \|A^T(y - A\tilde{x})\|_\infty \leq \epsilon. \qquad \text{(Dual-Lasso)}$$

As it is clear that dual problem to Lasso (Dual-Lasso) looks very similar to the Dantzig selector (Primal-DS). But we will not exploit this similarity, instead we will confine ourselves to the homotopy algorithm for Lasso as described in [22, 43]. We can easily derive optimality conditions for Lasso [30], which require that any valid solution x^* to (Lasso) for a given ϵ must obey

L1. $A_\Gamma^T(Ax^* - y) = -\epsilon z$

L2. $|a_\gamma^T(Ax^* - y)| < \epsilon \quad$ for all $\quad \gamma \in \Gamma^c$

where Γ is the support of x^* and z is its sign sequence on Γ . This gives us the update direction

$$\partial x = \begin{cases} (A_\Gamma^T A_\Gamma)^{-1} z & \text{on } \Gamma \\ 0 & \text{on } \Gamma^c \end{cases} \tag{4.23}$$

Note that Lasso optimality conditions (L1-L2) are exactly same as the Dantzig selector optimality conditions (K1-K4) when $\Gamma_\lambda = \Gamma_x$ and $z_\lambda = -z_x$. This gives us the following result about equivalence between Lasso and Dantzig selector.

Corrollary 4.6. *If the conditions presented in (H1-H4) are obeyed, then* (Lasso) *and* (Primal-DS) *will recover x in exactly S steps and the paths they take will be identical.*

Proof. As discussed in section 4.1, in order to recover an S-sparse signal in S primal-dual steps we need the supports of x and λ to match i.e., $\Gamma_x = \Gamma_\lambda$, and their signs on respective supports to mismatch i.e., $z_x = -z_\lambda$. Let $\Gamma := \Gamma_x = \Gamma_\lambda$ and $z = z_x$. Then at every step the non-zero portion of update direction in (3.6) for Dantzig selector is given as $\partial x\big|_{\Gamma_x} = -(A_{\Gamma_\lambda}^T A_{\Gamma_x})^{-1} z_\lambda = (A_\Gamma^T A_\Gamma)^{-1} z$, which is precisely the update direction for Lasso (4.23). □

Similarly, the difference between the paths taken by Lasso and Dantzig selector can be easily seen by looking at their respective update directions

$$\partial x^{\text{Lasso}}\big|_\Gamma = (A_\Gamma^T A_\Gamma)^{-1} z \qquad \text{(Lasso update)}$$

$$\partial x^{\text{DS}}\big|_{\Gamma_x} = -(A_{\Gamma_\lambda}^T A_{\Gamma_x})^{-1} z_\lambda \qquad \text{(DS update)}$$

So as long as support of primal and dual vector in Dantzig selector stays the same, Lasso and Dantzig selector paths will be identical. Their paths start to diverge when support of dual vector Γ_λ differs from primal support Γ_x in the Dantzig selector. If

we modify our homotopy algorithm for Dantzig selector such that it does not update dual direction and use same support for both primal and dual vectors, we will get homotopy path for Lasso.

4.5 Discussion and future work

In previous sections we have seen that if our sensing matrix obeys Dantzig shrinkability conditions, then we can recover an S-sparse signal in exactly S-steps. Although we have proved the S-step solution property for random matrices with $m \gtrsim S^2 \cdot \log n$, but our ultimate goal was to establish this property with something like $m \gtrsim S \cdot \log^{\alpha} n$, for some small $\alpha > 0$ (what we observe in practice!).

In order to show that S-step solution property holds for a given signal x supported on a subset Γ with sign sequence z, we need to show that the optimality conditions (H1-H3) hold at *every* step, with respective support $\Gamma_k \subset \Gamma$. As shown in section 4.1 this is equivalent to say that matrix $G := 1 - A_{\Gamma_k}^T A_{\Gamma_k}$ obeys $\|G\| < 1$ and condition in (4.2) is obeyed at every step. The condition about G is satisfied by any matrix which obeys uniform or weak uncertainty principle, and we can extend the result to sub-matrices of G using Cauchy's interlacing theorem. The main difficulty arises with establishing condition in (4.2) for all subsets of Γ. We can establish (4.2) for a *fixed* subset $\Gamma' \subset \Gamma$ with $m \gtrsim S \cdot \log n$ [8]. In which case we will use some concentration inequality e.g., Hoeffding's inequality to bound the inner product inner product $|\langle w_\gamma, z \rangle|$, instead of Cauchy-Schwarz inequality. But these concentration inequalities require w_γ and z to be statistically independent, and after the first step new w_γ will not be independent of z anymore. However, we can use some concentration inequality in the first step to bound the inner product $|\langle w_\gamma, z \rangle|$ with support set Γ. Although we know that we need to establish (4.2) for some S subsets of Γ, for next S steps, but we do not know in advance which subsets would they be. So an obvious choice to establish (4.2) at *every* step can be to use the union bound for all 2^S possible subsets

of Γ. This will essentially give us the same result of $S^2 \cdot \log n$ measurements.

As a final remark, we expect that S-step solution property holds for $m \gtrsim S \cdot \log^{\alpha} n$, for some $\alpha > 0$. But we have not been able to establish it yet, unless we somehow get rid of that dependence issue. In future, we intend to investigate this further and see if it is possible to push the measurement bound from $S^2 \cdot \log n$ to something like $S \cdot \log^{\alpha} n$.

CHAPTER V

NUMERICAL IMPLEMENTATION AND EXPERIMENTAL RESULTS

5.1 Numerical implementation

The main computational burden in the interior point methods for Dantzig selector comes from solving an n dimensional system of linear equations multiple times for each Newton iteration, which can be solved using conjugate gradient (CG) methods (see [16] for details). Computational cost for each Newton iteration is about $O(n^3)$. This is a bottleneck for large scale problems e.g., imaging.

In contrast our method is iterative and main cost comes from finding update directions ∂x and $\partial\lambda$ and respective step sizes δ and θ. Under normal situation we are adding or removing one column to or from each A_{Γ_x} and A_{Γ_λ} at any step. Therefore, it won't be sensible to solve a complete system of equations at every step, each of which will cost $O(k^3+km)$ flops, where k is the size of our current primal-dual support. Instead we can update the inverse of our Gram matrices at every step with k elements in the support with only $O(km)$ flops. The cost associated to compute step sizes is about $O(mn)$. This is essentially the cost of few matrix-vector multiplications, and it cannot be reduced if we use explicit matrix representation.

5.2 Block matrix update

In our implementation we continuously update the Gram matrix $A_{\Gamma_x}^T A_{\Gamma_\lambda}$ and $A_{\Gamma_\lambda}^T A_{\Gamma_x}$, and their inverses whenever the Γ_x and/or Γ_λ are changed. As we see that these two matrices are transpose of each other, so update for any one would be sufficient (since for any invertible matrix A, $(A^T)^{-1} = (A^{-1})^T$). In our algorithm, at every step

usually one element is added or removed from each support, so we can easily update the inverse using matrix inversion lemma (or Sherman-Woodbury-Morrison formula) for low rank update [4, 31]. For a single primal-dual step, we need to update Gram matrix and its inverse before computing ∂x. And if any element is removed from the primal support during primal update step, then we will have to update inverse before computing $\partial\lambda$ as well.

Under the assumption that at every step primal-dual support changes at one location at most, we can encounter the following scenarios where we need to update the Gram matrix and its inverse

1. A new element enters Γ_x and Γ_λ.

2. An existing element leaves Γ_x during primal update.

3. A new element enters Γ_λ and an existing element leaves Γ_λ during dual update.

In order to update the Gram matrices for these case we can use the following simple matrix inverse update schemes.

5.2.1 Adding new columns

Assume that Γ_x and Γ_λ is the support of our primal and dual vectors respectively and we already have $A^T_{\Gamma_x} A_{\Gamma_\lambda}$ and its inverse. Let us assume that one new element enters in the support of each vector, and our new supports become $\tilde{\Gamma}_x := \Gamma_x \cup \{\gamma_x\}$ and $\tilde{\Gamma}_\lambda := \Gamma_\lambda \cup \{\gamma_\lambda\}$. Then we need to update our Gram matrix $A^T_{\tilde{\Gamma}_x} A_{\tilde{\Gamma}_\lambda}$ and its inverse.

Let us consider we have two $m \times k$ matrices A and B and we already have $k \times k$ inverse matrix $(A^T B)^{-1}$. Now if we add a new column to each of the two matrices such that our new matrices become $\tilde{A} = [A \quad a]$ and $\tilde{B} = [B \quad b]$, we need to find the inverse of the updated $(k+1) \times (k+1)$ Gram matrix $(\tilde{A}^T \tilde{B})$, which we can write as:

$$\tilde{A}^T \tilde{B} = \begin{bmatrix} A^T & a^T \end{bmatrix} \begin{bmatrix} B \\ b \end{bmatrix} = \begin{bmatrix} A^T B & A^T b \\ a^T B & a^T b \end{bmatrix}.$$

In order to find this updated matrix we need one vector-vector and two matrix-vector multiplications, which requires nearly $(2k+1)m$ flop counts. For the update of inverse we will use block matrix inverse update as described below. Let us write $D := \tilde{A}^T\tilde{B}$ as

$$D = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} A^TB & A^Tb \\ a^TB & a^Tb \end{bmatrix}, \tag{5.1}$$

where $A_{11} = A^TB$ is a $k \times k$ block matrix whose inverse we have already computed and stored. Using matrix inversion lemma [31], the inverse of our square matrix $\tilde{A}^T\tilde{B}$ can be written as

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} A_{11}^{-1} + A_{11}^{-1}A_{12}S^{-1}A_{21}A_{11}^{-1} & -A_{11}^{-1}A_{12}S^{-1} \\ -S^{-1}A_{21}A_{11}^{-1} & S^{-1} \end{bmatrix}, \tag{5.2}$$

where $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ is the Schur complement of A_{11} in the square matrix D and I is the k dimensional identity matrix. Under the assumption that one element enters at a time S will always be scalar. In order to update the inverse matrix we need to first compute $A_{11}^{-1}A_{12}$ and $A_{21}A_{11}^{-1}$, each of which take about k^2 flops, after that S can be computed in just k flops. For the main $k \times k$ block, we need a rank one update $A_{11}^{-1} + (A_{11}^{-1}A_{12})(A_{21}A_{11}^{-1})/S$ which will cost about $2k^2$ flops. So we can find the inverse any modified Gram matrix in just about $(2k+1)m + 4k^2$ flops which is bounded above by $O(km)$.

5.2.2 Deleting columns

Similarly if we remove an element from Γ_x during primal update, we can find the inverse of a $(k-1) \times (k-1)$ block of a $k \times k$ matrix whose inverse we have computed previously and stored. To see this let's consider an example where we have two matrices $\tilde{A}$ and $\tilde{B}$ as defined earlier, and this time we remove columns a and b from $\tilde{A}$ and $\tilde{B}$ respectively. Here our task is to compute inverse of A^TB from $\tilde{A}^T\tilde{B}$. Let us

write the inverse of bigger matrix in the following form

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} =: \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}.$$

Since we are interested in finding $A_{11}^{-1} =: Q_{11}$, we can write Q_{11} as in (5.2)

$$\begin{aligned} Q_{11} &= A_{11}^{-1} + A_{11}^{-1} A_{12} S^{-1} A_{21} A_{11}^{-1} \\ &= A_{11}^{-1} + (Q_{12} S) S^{-1} (S Q_{21}), \end{aligned}$$

which gives us

$$A_{11}^{-1} = Q_{11} - Q_{12} Q_{22}^{-1} Q_{21}. \tag{5.3}$$

This is again just a rank one update, so if we want to update inverse of our Gram matrices after removing a column from the support, it can be done in just about k^2 flops.

In our discussion above, we assumed that last row and column from the bigger matrix is being removed, which is equivalent to say that last element from Γ_x and Γ_λ are removed. If some other row and column is to be removed, we can modify this method easily. For example, we can first change Γ_x and Γ_λ such that outgoing indices are swapped with last indices and then modify Gram matrix and its inverse accordingly. Let us assume we need to remove i^{th} row and j^{th} column. So to modify Gram matrix we just need to swap its i^{th} row and j^{th} column with last row and column. Similarly to get the modified inverse we need to swap the j^{th} row and i^{th} column of the actual inverse of Gram matrix with its last row and column respectively. To see this let's consider $Q := \tilde{A}^T \tilde{B}$ and Q_{ij} as the modified matrix whose i^{th} row and j^{th} column is swapped with last ones. Then we can write

$$Q_{ij} = P_{in} Q P_{jn}$$

and in turn

$$Q_{ij}^{-1} = P_{jn} Q^{-1} P_{in},$$

where P_{in} and P_{jn} denote the permutation matrices. After this we can apply the same procedure as discussed in (5.3).

5.2.3 Replacement of columns

If an existing element in supp(λ) is replaced by a new element, we can update the Gram matrices and their inverses by similar rank update. Consider for example that we already have the Gram matrix $A_{\Gamma_x}^T A_{\Gamma_\lambda}$ and its inverse. An element γ^- in Γ_λ is replaced by a new element γ^+, let us denote the new support as $\tilde{\Gamma}_\lambda$. So we need to compute the new matrix $A_{\Gamma_x}^T A_{\tilde{\Gamma}_\lambda}$ and its inverse. We can conveniently write the updated matrix as rank one update of the old matrix as

$$A_{\Gamma_x}^T A_{\tilde{\Gamma}_\lambda} = A_{\Gamma_x}^T A_{\Gamma_\lambda} + (A_{\Gamma_x}^T a_{\gamma^+} - A_{\Gamma_x}^T a_{\gamma^-})\mathbf{1}_\gamma^T, \tag{5.4}$$

where $\mathbf{1}_\gamma$ is a vector which is equal to 1 at index corresponding to the location of γ^- in Γ_λ and zero elsewhere. In order to find the inverse of updated matrix in (5.4) we can again use Sherman-Woodbury-Morrison formula [31] given as

$$(F + UV)^{-1} = F^{-1} - F^{-1}U(I + VF^{-1}U)^{-1}VF^{-1},$$

where F is $k \times k$ nonsingular matrix, U is $k \times q$ matrix and V is $q \times k$ matrix. In our implementation q will usually be equal to 1.

To summarize, in our proposed algorithm any primal-dual step can be performed at a cost $O(mn)$. In section 4.1 we presented conditions under which our algorithm takes at most S primal-dual steps to recover an S-sparse, in which case our total cost will be $O(Smn)$. In general, our computational cost will be bounded by $O(dmn)$, where d is the total number of primal-dual steps taken, which is same as updating a least squares solution d times.

5.3 Experimental results

In this section we will present some simulation results for the primal-dual pursuit algorithm. Our results can be interpreted into two main categories for signal recovery:

- S-step recovery: when S-sparse signal is exactly recovered in at most S primal-dual steps of PD-pursuit.

- Exact recovery: when S-sparse signal is exactly recovered by PD-pursuit (without considering the number of steps taken).

The results presented here are aimed at finding, empirically, how many measurement, m do we need for S-step recovery. On the other hand, we want to see how many primal-dual steps does our algorithm take on average whenever an S-sparse signal is exactly recovered from m measurements. In practice, we usually get exact signal recovery with nearly all ℓ_1 minimization schemes if $m \geq 4S$.

In the performance plots, Figures 3 to Figures 12, we will give performance results for S-step recovery by PD-pursuit and compare our results with S-sparse signal recovery using orthogonal matching pursuit (OMP) [55], which is a greedy algorithm and terminates after exactly S iterations. Along with that we will also give results regarding exact recovery of sparse signals by PD-pursuit and the number of primal-dual steps taken on average by our algorithm whenever signal is recovered exactly.

In our simulations we used the following different distributions for our $m \times n$ measurement matrix A.

Gaussian: Choose each entry as i.i.d. Gaussian with zero mean and variance $1/m$.

Bernoulli: Choose each entry independently to be $\pm 1/\sqrt{m}$ with equal probability.

Orthogonalized Gaussian: Generate Gaussian matrix and orthogonalize its rows.

Partial Hadamard: Select m rows of an $n \times n$ Hadamard matrix uniformly at random.

In all simulations we performed 1000 independent trials for every set of measurements (m) at different sparsity levels (S). In each trial a new measurement matrix

is generated, along with a new sparse n dimensional vector x, whose S entries at randomly chosen locations are set either $+1$ or -1 with equal probability[1].

Figure 3 and Figure 4 show the results for Gaussian matrix with ($n = 256$). Figure 5 and Figure 6 show the results for Bernoulli matrix with ($n = 256$). Figure 7 and Figure 8 show the results for orthogonalized Gaussian matrix with ($n = 256$). It is interesting to note here that exact recovery performance of Gaussian matrix and orthogonalized Gaussian matrix is exactly same but S-step recovery with orthogonalized Gaussian matrix is much superior. Similar results can be seen in Figure 9 and Figure 10 for partial Hadamard matrix, whose rows are also orthogonal to each other with ($n = 256$). Figure 11 and Figure 12 show the results for partial Hadamard matrix with ($n = 1024$).

We observe that, if the measurement matrix has orthogonal rows, we get exact recovery with about $4S$ measurements in nearly $4S$ primal-dual steps, whereas for S-step recovery we need about $8S$ measurements. In case of Gaussian and Bernoulli matrices we get exact recovery with about $4S$ measurements in about $10S$ primal-dual steps. As can be seen from the simulation plots that required number of primal-dual steps for exact recovery decrease sharply with increased number of measurements.

[1]We have seen similar performance with other distributions for signal values, e.g., Gaussian and binary.

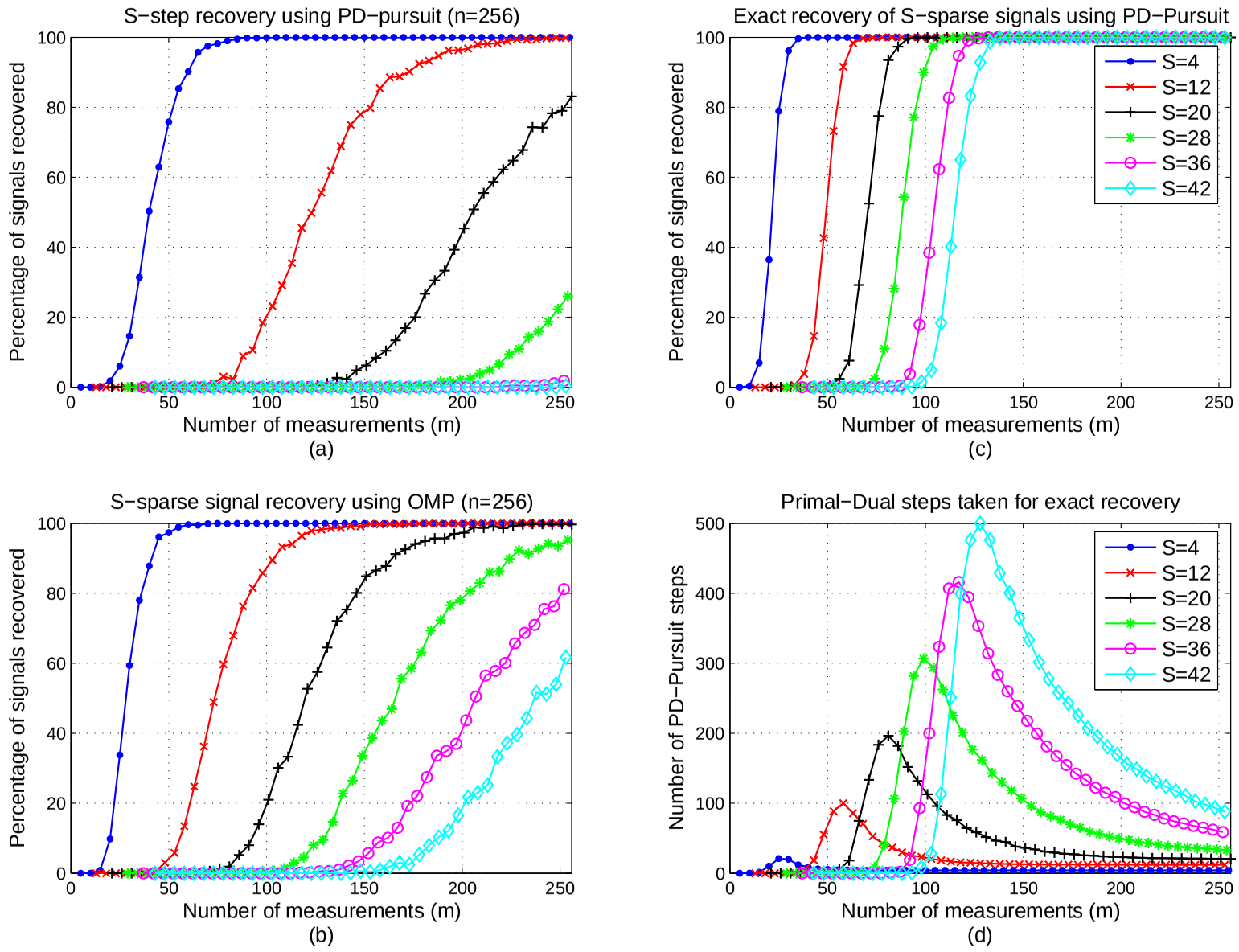

Figure 3: Gaussian matrix (n=256). (a) Percentage of S-sparse signals recovered in S-steps of PD-pursuit using m measurements. (b) Percentage of S-sparse signals recovered by orthogonal matching pursuit (OMP) using m measurements. (c) Percentage of S-sparse signals exactly recovered by PD-pursuit. (d) Average number of primal-dual steps taken for exact recovery of S-sparse signals.

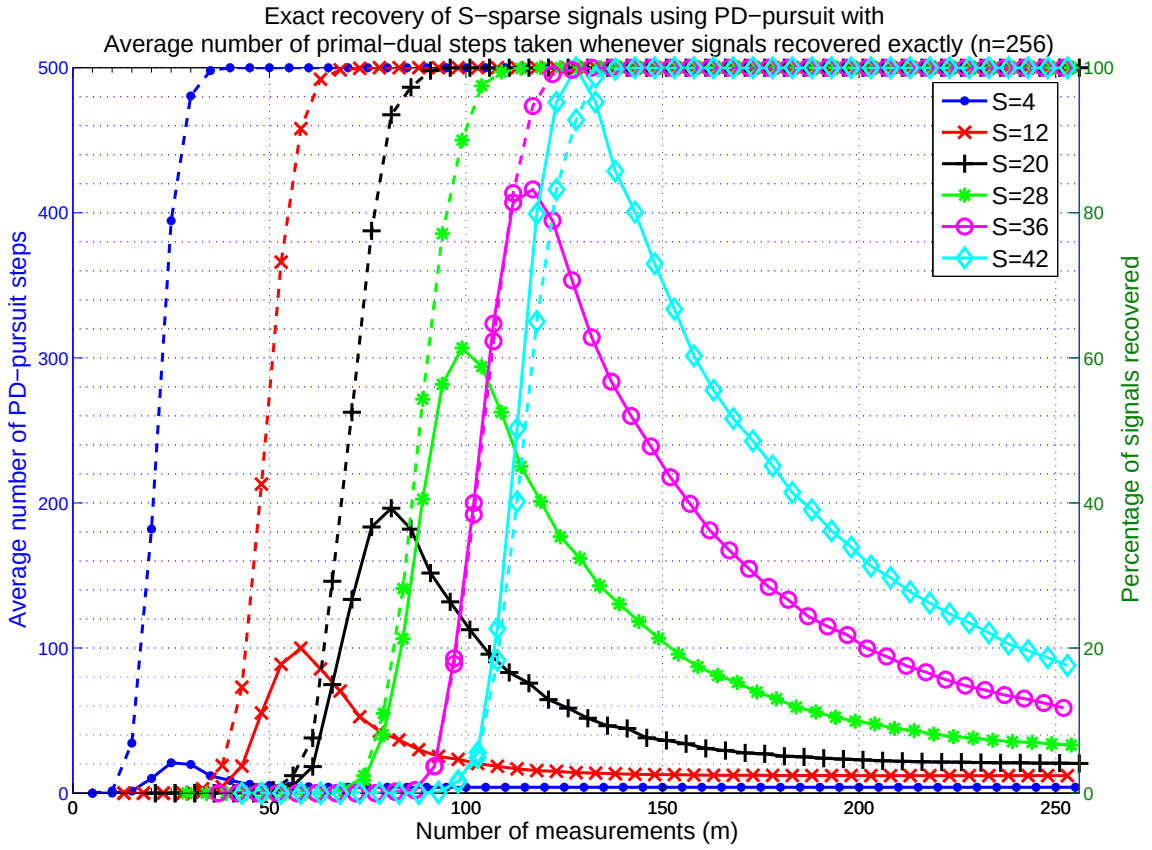

Figure 4: Gaussian matrix (n=256). Superimposed Plots for exact recovery of S-sparse signal with the average number of primal-dual steps taken by primal-dual pursuit. Dashed lines in the plot and vertical axis on the right gives percentage of signals recovered using m measurements. Solid lines in the plot and vertical axis on the left gives the average number of primal-dual steps taken whenever signal is recovered exactly with m measurements.

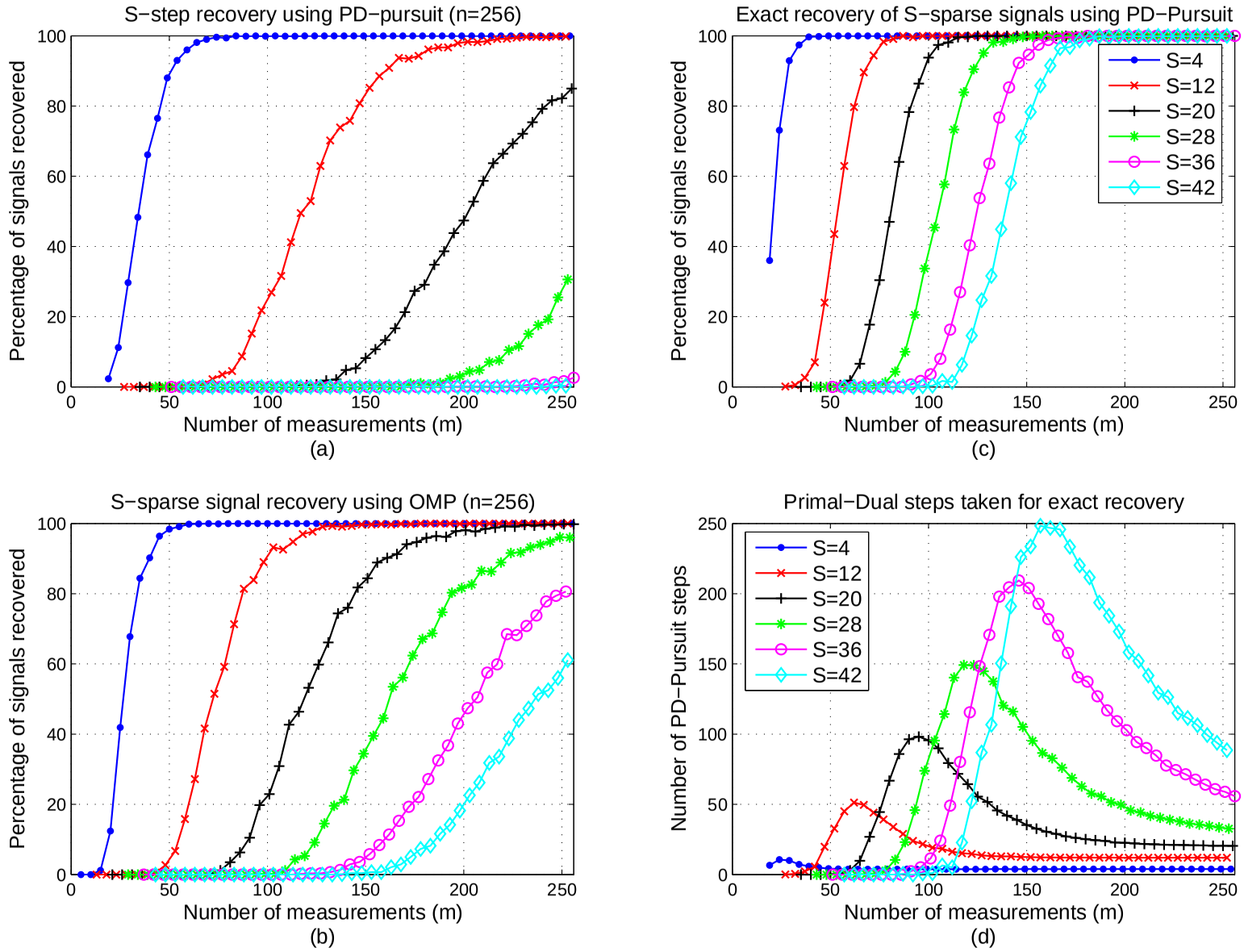

Figure 5: Bernoulli matrix (n=256). (a) Percentage of S-sparse signals recovered in S-steps of PD-pursuit using m measurements. (b) Percentage of S-sparse signals recovered by orthogonal matching pursuit (OMP) using m measurements. (c) Percentage of S-sparse signals exactly recovered by PD-pursuit. (d) Average number of primal-dual steps taken for exact recovery of S-sparse signals.

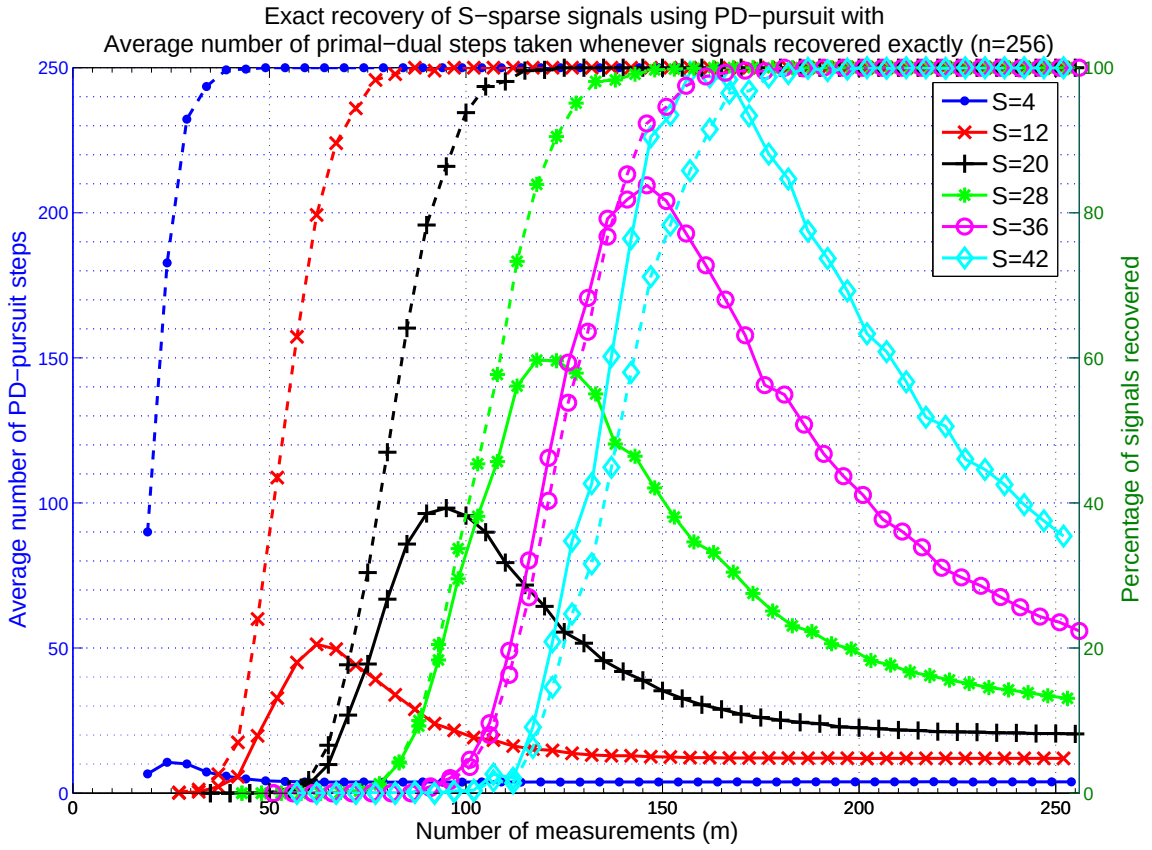

Figure 6: Bernoulli matrix (n=256). Superimposed Plots for exact recovery of S-sparse signal with the average number of primal-dual steps taken by primal-dual pursuit. Dashed lines in the plot and vertical axis on the right gives percentage of signals recovered using m measurements. Solid lines in the plot and vertical axis on the left gives the average number of primal-dual steps taken whenever signal is recovered exactly with m measurements.

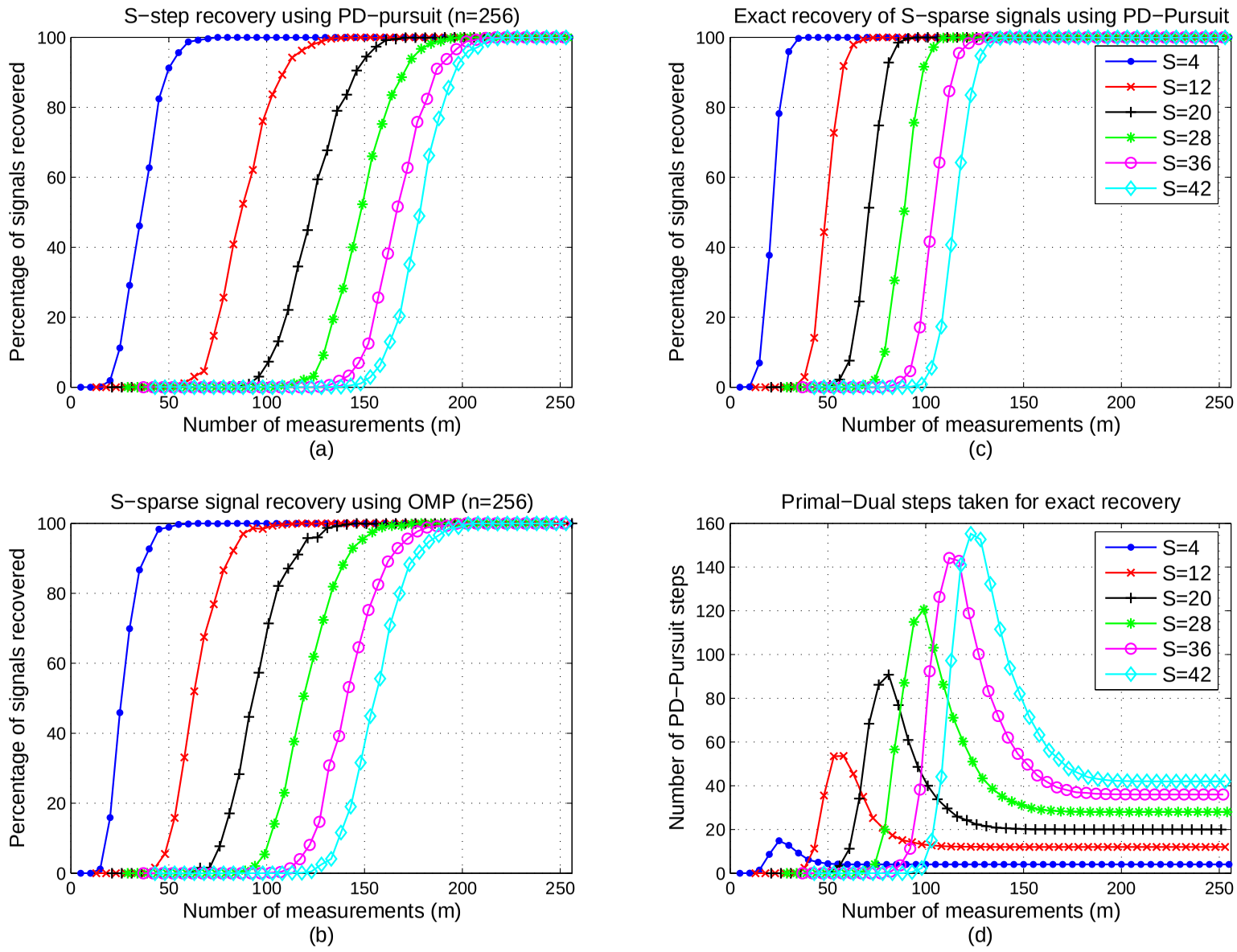

Figure 7: Orthogonalized Gaussian matrix (n=256). (a) Percentage of S-sparse signals recovered in S-steps of PD-pursuit using m measurements. (b) Percentage of S-sparse signals recovered by orthogonal matching pursuit (OMP) using m measurements. (c) Percentage of S-sparse signals exactly recovered by PD-pursuit. (d) Average number of primal-dual steps taken for exact recovery of S-sparse signals.

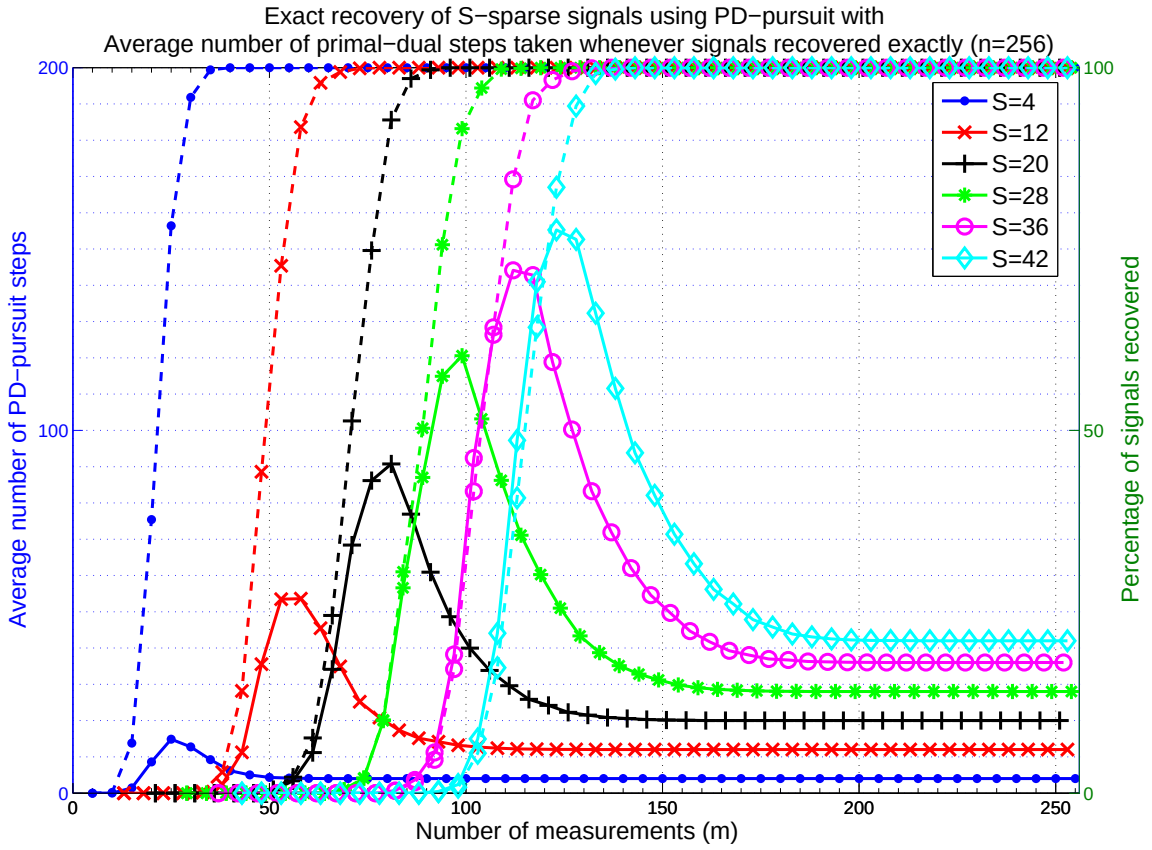

Figure 8: Orthogonalized Gaussian matrix (n=256). Superimposed Plots for exact recovery of S-sparse signal with the average number of primal-dual steps taken by primal-dual pursuit. Dashed lines in the plot and vertical axis on the right gives percentage of signals recovered using m measurements. Solid lines in the plot and vertical axis on the left gives the average number of primal-dual steps taken whenever signal is recovered exactly with m measurements.

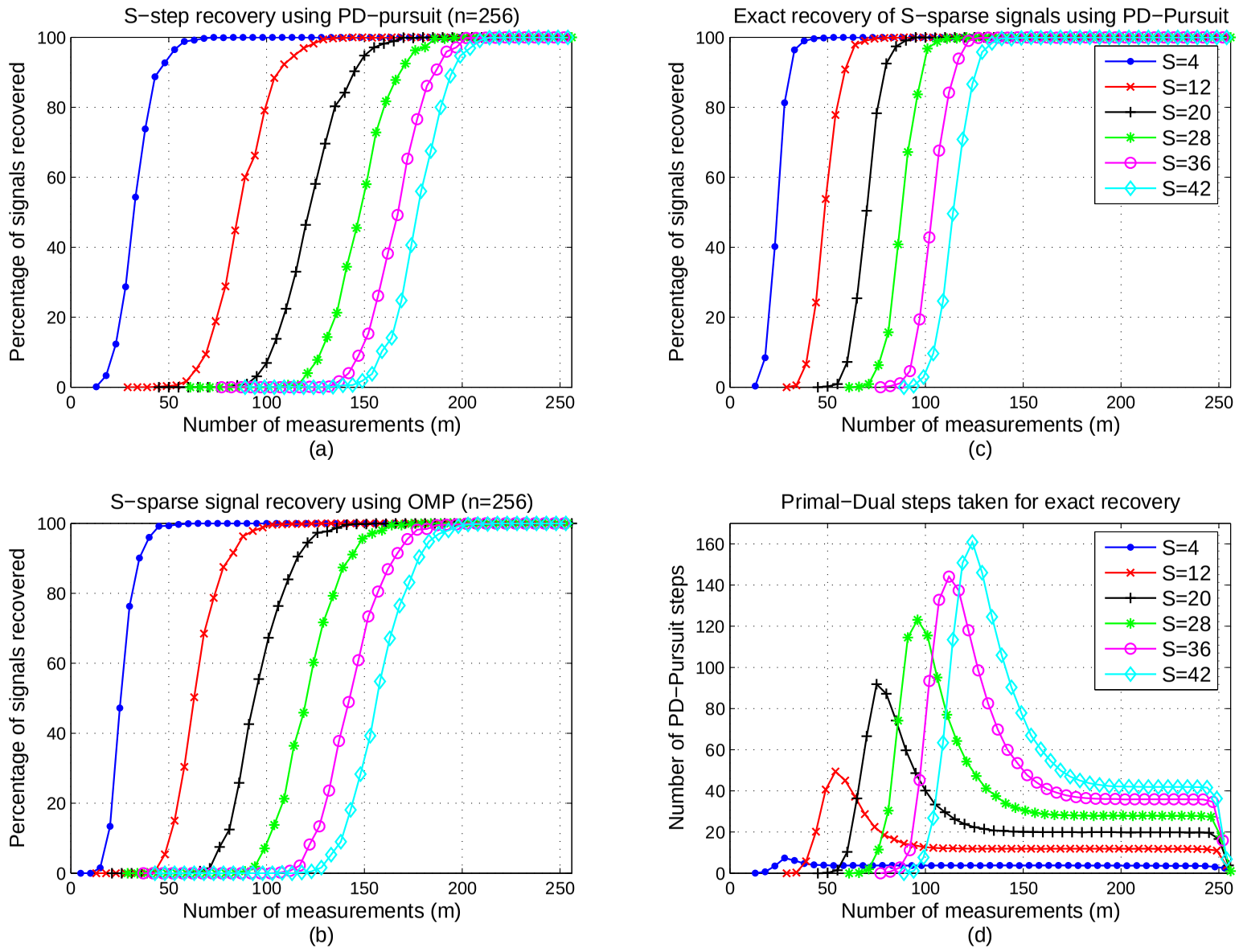

Figure 9: Partial Hadamard matrix (n=256). (a) Percentage of S-sparse signals recovered in S-steps of PD-pursuit using m measurements. (b) Percentage of S-sparse signals recovered by orthogonal matching pursuit (OMP) using m measurements. (c) Percentage of S-sparse signals exactly recovered by PD-pursuit. (d) Average number of primal-dual steps taken for exact recovery of S-sparse signals.

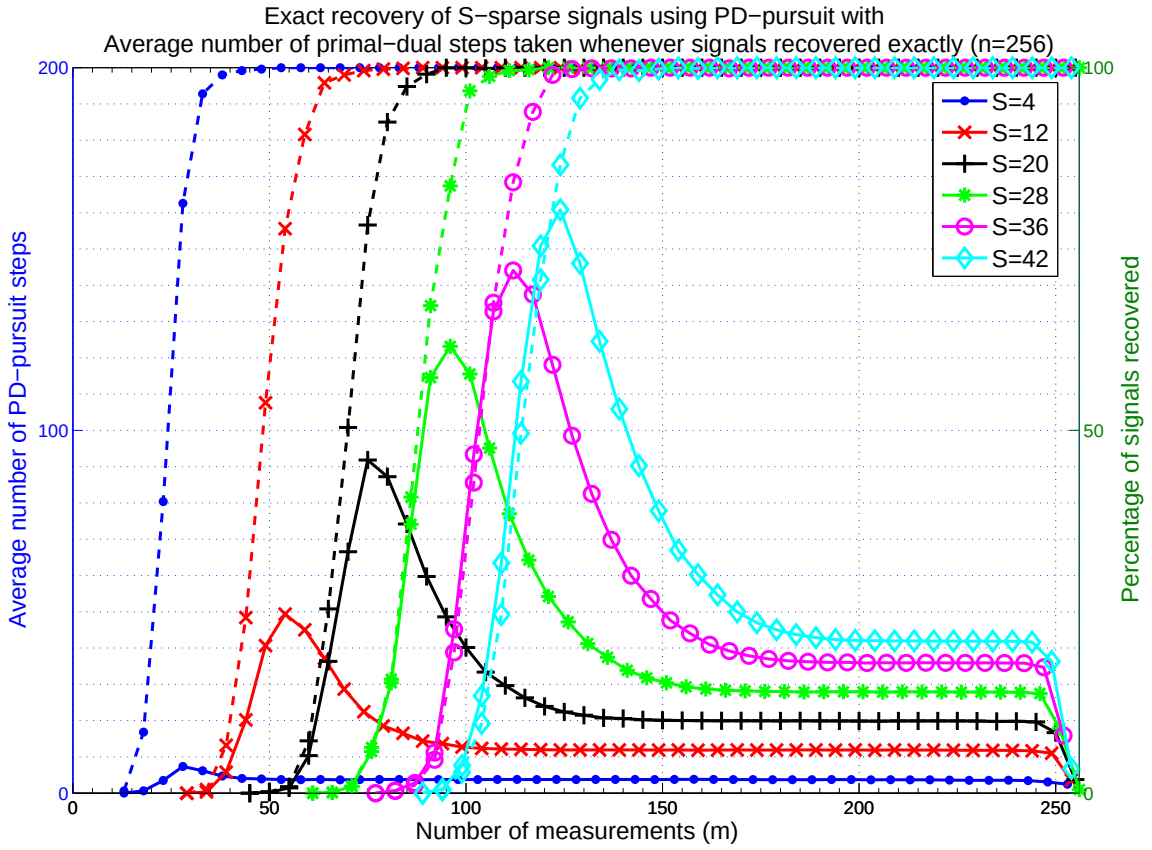

Figure 10: Partial Hadamard matrix (n=256). Superimposed Plots for exact recovery of S-sparse signal with the average number of primal-dual steps taken by primal-dual pursuit. Dashed lines in the plot and vertical axis on the right gives percentage of signals recovered using m measurements. Solid lines in the plot and vertical axis on the left gives the average number of primal-dual steps taken whenever signal is recovered exactly with m measurements.

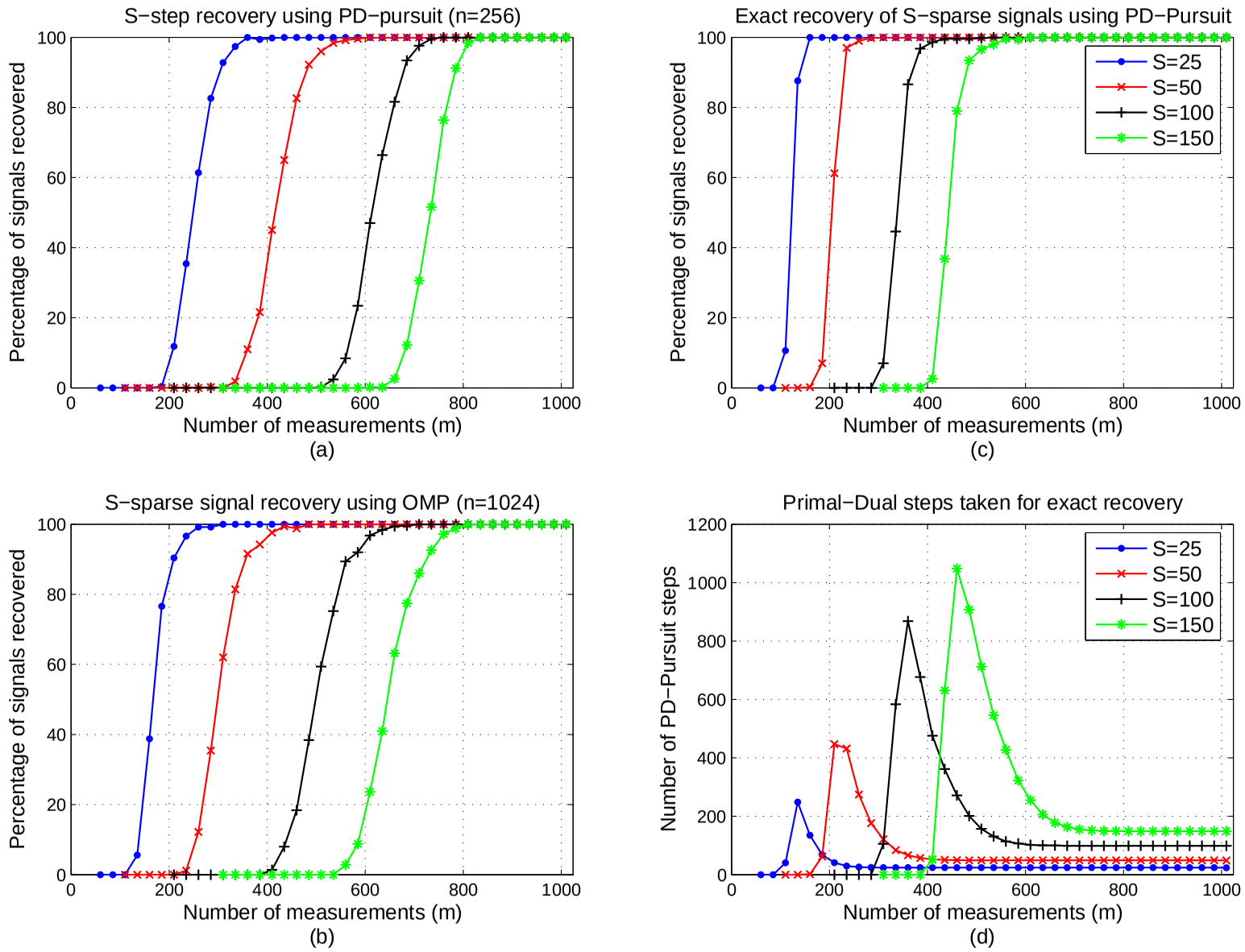

Figure 11: Partial Hadamard matrix (n=1024). (a) Percentage of S-sparse signals recovered in S-steps of PD-pursuit using m measurements. (b) Percentage of S-sparse signals recovered by orthogonal matching pursuit (OMP) using m measurements. (c) Percentage of S-sparse signals exactly recovered by PD-pursuit. (d) Average number of primal-dual steps taken for exact recovery of S-sparse signals.

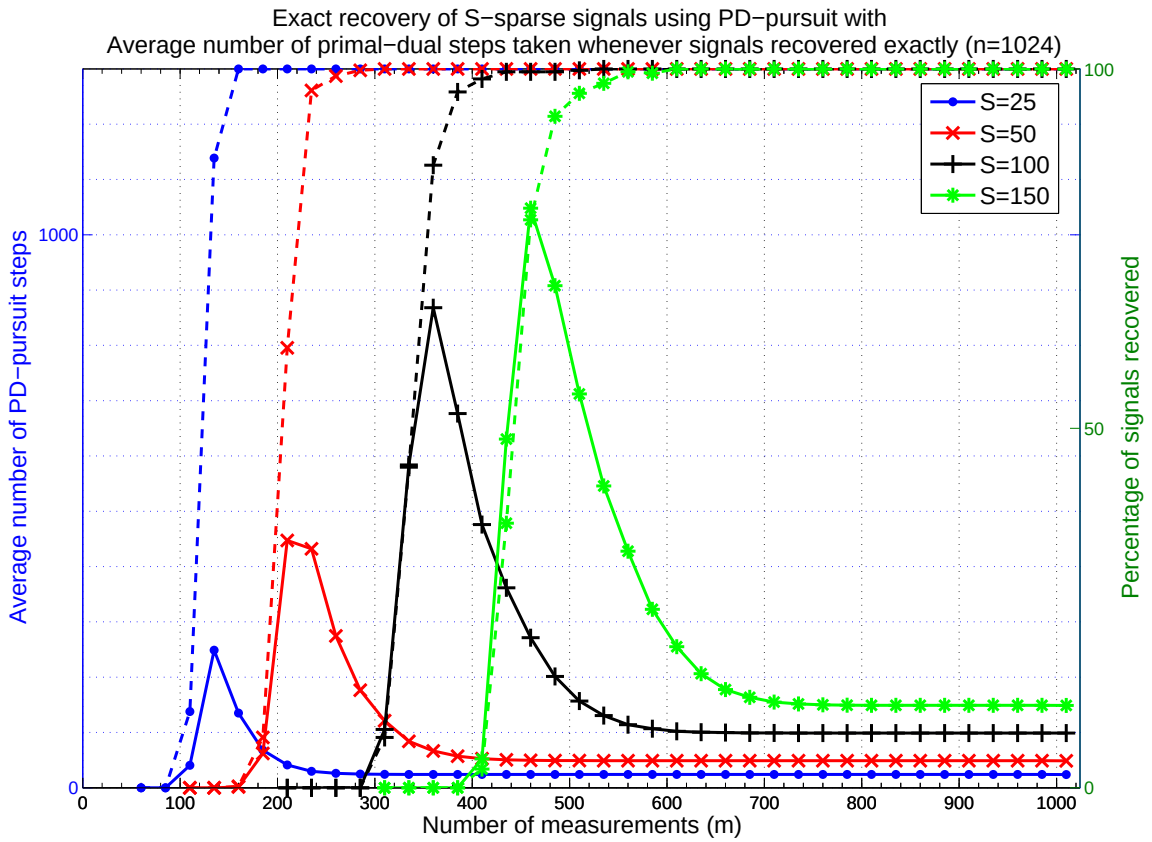

Figure 12: Partial Hadamard matrix (n=1024). Superimposed Plots for exact recovery of S-sparse signal with the average number of primal-dual steps taken by primal-dual pursuit. Dashed lines in the plot and vertical axis on the right gives percentage of signals recovered using m measurements. Solid lines in the plot and vertical axis on the left gives the average number of primal-dual steps taken whenever signal is recovered exactly with m measurements.

APPENDIX A

PRIMAL DUAL FORMULATION

A.1 Standard primal dual LP

Let us consider the following linear program under inequality constraints.

$$\underset{z}{\text{minimize}}\ c^T z \quad \text{subject to} \quad Mz \preceq b, \tag{Primal-LP}$$

where $z \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, $M \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Here $\preceq$ and $\succeq$ denote element wise inequalities. Dual to (Primal-LP) is given as

$$\underset{v}{\text{maximize}}\ -b^T v \quad \text{subject to} \quad M^T v + c = 0, \quad v \succeq 0, \tag{Dual-LP}$$

where $v \in \mathbb{R}^m$ is the dual vector [4].

A.2 ℓ_1 norm with equality constraint

Let $x \in \mathbb{R}^n, A \in \mathbb{R}^{m\times n}, y \in \mathbb{R}^m$, then our first convex program is given as

$$\underset{x}{\text{minimize}}\ \|x\|_1 \quad \text{such that} \quad Ax = y \tag{P-EQ}$$

We can recast (P-EQ) into an equivalent linear program (LP) as

$$\begin{aligned} \underset{x,u}{\text{minimize}}\ & \sum_{i=1}^{n} u_i && \text{Dual Variables} \\ \text{subject to} \quad & x - u \preceq 0 && \nu_1 \\ & -x - u \preceq 0 && \nu_2 \\ & Ax \preceq y && \lambda_1 \\ & -Ax \preceq -y && \lambda_2 \end{aligned} \tag{A.1}$$

where $u \in \mathbb{R}^n$ is a dummy vector. We have written the respective dual variables opposite to each set of inequalities, dimension of the dual vectors will match the

number of inequalities, i.e., $\nu_{1,2} \in \mathbb{R}^n$ and $\lambda_{1,2} \in \mathbb{R}^m$. Now we can relate (A.1) to the form of (Primal-LP) with following format:

$$c = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix}, \quad z = \begin{bmatrix} x \\ u \end{bmatrix}, \quad M = \begin{bmatrix} I & -I \\ -I & -I \\ A & 0 \\ -A & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ y \\ -y \end{bmatrix},$$

where $\mathbf{1}$ is a vector with all its entries as 1, I is n-dimensional identity matrix. The dual to (P-EQ) can be written as

$$\begin{aligned} \underset{\nu_1,\nu_2,\lambda_1,\lambda_2}{\text{maximize}} \quad & - \begin{bmatrix} 0 & 0 & y^T & -y^T \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} \\ \text{subject to} \quad & \begin{bmatrix} I & -I & A^T & -A^T \\ -I & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix} = 0, \\ & \nu_1, \nu_2, \lambda_1, \lambda_2 \succeq 0. \end{aligned} \tag{A.2}$$

where ν_1, ν_2 are n-dimensional and λ_1, λ_2 are m-dimensional dual vectors. The feasibility conditions are as follows:

$$\begin{aligned} \nu_1 - \nu_2 + A^T\lambda_1 - A^T\lambda_2 &= 0 \\ -\nu_1 - \nu_2 + 1 &= 0 \\ \nu_1, \nu_2, \lambda_1, \lambda_2 &\succeq 0. \end{aligned}$$

Let us set $\lambda = \lambda_1 - \lambda_2$ and $\nu = \nu_1 - \nu_2$. Now we know that $\nu_1 = 1 - \nu_2$ and $\nu_1, \nu_2 \succeq 0$, so we can write the second feasibility condition in its equivalent form as $\|\nu\|_\infty \leq 1$, where $\|\cdot\|_\infty$ is the infinity or supremum norm. Also we can write the first feasibility

condition as

$$A^T(\lambda_1 - \lambda_2) = \nu_2 - \nu_1,$$

and taking supremum norm on both sides we get the following feasibility condition

$$\|A^T\lambda\|_\infty = \|\nu\|_\infty \leq 1.$$

Since $\lambda = \lambda_1 - \lambda_2$ and $\lambda_1, \lambda_2 \succeq 0$, therefore λ is unconstrained in sign. So we can write the dual problem to (P-EQ) as:

$$\underset{\lambda}{\text{maximize}} \ -y^T\lambda \quad \text{subject to} \qquad \|A^T\lambda\|_\infty \leq 1 \tag{D-EQ}$$

A.3 ℓ_1 norm with bounded residual - Dantzig selector

Let $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $y \in \mathbb{R}^m$ and ϵ be a small non-zero constant. The convex program for Dantzig selector is given as

$$\underset{x}{\text{minimize}} \ \|x\|_1 \quad \text{subject to} \quad \|A^T(Ax - y)\|_\infty \leq \epsilon \tag{P-DS}$$

We can write its equivalent LP as:

$$\begin{aligned}
&\underset{x,u}{\text{minimize}} \ \sum_{i=1}^{n} u_i && \text{Dual Variables} \\
&\text{subject to} \quad x - u \preceq 0 && \boldsymbol{\nu_1} \\
&\qquad -x - u \preceq 0 && \boldsymbol{\nu_2} \\
&\qquad A^TAx \preceq \epsilon + A^Ty && \boldsymbol{\lambda_1} \\
&\qquad -A^TAx \preceq \epsilon - A^Ty && \boldsymbol{\lambda_2}
\end{aligned} \tag{A.3}$$

where $u \in \mathbb{R}^n$ is a dummy vector. We have written the respective dual variables opposite to each set of inequalities, dimension of the dual vectors will match the number of inequalities, and here all of them will be n-dimensional. Now we can relate

(A.3) to the form of (Primal-LP) with following format:

$$c = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix}, \quad z = \begin{bmatrix} x \\ u \end{bmatrix}, \quad M = \begin{bmatrix} I & -I \\ -I & -I \\ A^TA & 0 \\ -A^TA & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ \epsilon + A^Ty \\ \epsilon - A^Ty \end{bmatrix}$$

So the dual to (P-DS) can be written as:

$$\begin{aligned} \underset{\nu_1,\nu_2,\lambda_1,\lambda_2}{\text{maximize}} \quad & -\begin{bmatrix} 0 & 0 & (\epsilon + A^Ty)^T & (\epsilon - A^Ty)^T \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} I & -I & A^TA & -A^TA \\ -I & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix} = 0, \\ & \nu_1, \nu_2, \lambda_1, \lambda_2 \geq 0. \end{aligned} \tag{A.4}$$

The feasibility conditions are as follows:

$$\begin{aligned} \nu_1 - \nu_2 + A^TA\lambda_1 - A^TA\lambda_2 &= 0 \\ -\nu_1 - \nu_2 + 1 &= 0 \\ \nu_1, \nu_2, \lambda_1, \lambda_2 &\geq 0. \end{aligned}$$

Let us set $\lambda = \lambda_1 - \lambda_2$ and $\nu = \nu_1 - \nu_2$. Now we know $\nu_1 = 1 - \nu_2$ and $\nu_1, \nu_2 \geq 0$, so we can write the second feasibility condition in its equivalent form as $\|\nu\|_\infty \leq 1$. Also we can write the first feasibility condition as

$$A^TA(\lambda_1 - \lambda_2) = \nu_2 - \nu_1,$$

taking supremum norm on both sides gives us the following feasibility condition.

$$\|A^TA\lambda\|_\infty = \|\nu\|_\infty \leq 1.$$

This way we get rid of ν terms in the constraints. Now we can write the dual problem (A.4) as:

$$\begin{aligned} \underset{\lambda_1,\lambda_2}{\text{maximize}} \;\; & -\left(\epsilon(\lambda_1+\lambda_2) + (A^T y)^T \lambda \right) \\ \text{subject to} \quad & \|A^T A\lambda\|_\infty \leq 1 \\ & \lambda = \lambda_1 - \lambda_2 \\ & \lambda_1, \lambda_2 \succeq 0 \end{aligned} \tag{A.5}$$

Now we know $\lambda = \lambda_1 - \lambda_2$, and at any instant either λ_1 or λ_2 is zero (because they represent complementary constraints), and $\lambda_1, \lambda_2 \succeq 0$, so in order to maximize $-(\lambda_1+\lambda_2)$ or equivalently minimize $\lambda_1+\lambda_2$ we minimize ℓ_1 norm of $\lambda = \lambda_1-\lambda_2$, which is essentially $\lambda_1+\lambda_2$. Now our optimization variable will be λ (which is unconstrained in sign). So our Dual problem to (P-DS) can be written as

$$\underset{\lambda}{\text{maximize}} \;\; -(\epsilon\|\lambda\|_1 + \langle \lambda, A^T y \rangle) \quad \text{subject to} \quad \|A^T A\lambda\|_\infty \leq 1 \tag{D-DS}$$

REFERENCES

[1] "IEEE Signal Processing Magazine [Sensing, Sampling, and Compression]," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, March 2008.

[2] Abbott, E., *Flatland: A Romance of Many Dimensions*. Seeley & Co., 1884.

[3] Bickel, P. and Ya'acov Ritov, A. T., "Simultaneous analysis of Lasso and Dantzig selector," *Arxiv preprint arXiv:0801.1095*, 2008.

[4] Boyd, S. and Vandenberghe, L., *Convex Optimization*. Cambridge University Press, March 2004.

[5] Burrus, C., Gopinath, R., and Guo, H., *Introduction to Wavelets and Wavelet Transforms: A Primer*. Prentice Hall, Upper Saddle River, NJ, 1998.

[6] Candes, E., "Compressive sampling," *Proceedings of the International Congress of Mathematicians, Madrid, Spain*, vol. 3, pp. 1433–1452, 2006.

[7] Candes, E. and Plan, Y., "Near-ideal model selection by L1 minimization," *Arxiv preprint arXiv:0801.0345*, 2008.

[8] Candes, E. and Romberg, J., "Sparsity and incoherence in compressive sampling," *Inverse Problems*, vol. 23, no. 3, pp. 969–985, 2007.

[9] Candes, E., Romberg, J., and Tao, T., "Stable signal recovery from incomplete and inaccurate measurements," *Comm. Pure Appl. Math*, vol. 59, no. 8, pp. 1207–1223, 2006.

[10] Candes, E. and Tao, T., "The Dantzig selector: Statistical estimation when p is much larger than n," *Annals of Statistics*, vol. 35, no. 6, pp. 2313–2351, 2007.

[11] Candes, E. J. and Tao, T., "Near-optimal signal recovery from random projections: Universal encoding strategies?," *Information Theory, IEEE Transactions on*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.

[12] Candes, E. and Donoho, D., "New tight frames of curvelets and optimal representations of objects with piecewise C^2 singularities," *Communications on Pure and Applied Mathematics*, vol. 57, no. 2, pp. 219–266, 2004.

[13] Candes, E., Romberg, J., and Tao, T., "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[14] Candes, E. and Tao, T., "Decoding by linear programming," *Information Theory, IEEE Transactions on*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.

[15] Candes, E., "Lecture Notes; Compressive Sampling and Frontiers in Signal Processing." New direction short course, The Institute of Mathematics and its Applications, University of Minnesota, June 2007.

[16] Candes, E. and Romberg, J., "ℓ_1-magic: Recovery of Sparse Signals via Convex Programming." http://www.acm.caltech.edu/l1magic.

[17] Chen, S. S., Donoho, D. L., and Saunders, M. A., "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.

[18] Cohen, A., Dahmen, W., and DeVore, R., "Near optimal approximation of arbitrary signals from highly incomplete measurements," Preprint.

[19] Coifman, R. and Wickerhauser, M., "Entropy-based algorithms for best basis selection," *Information Theory, IEEE Transactions on*, vol. 38, no. 2, pp. 713–718, Mar 1992.

[20] Daubechies, I., Defrise, M., and De Mol, C., "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, 2004.

[21] Daubechies, I., *Ten Lectures on Wavelets*, vol. 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM), 1992.

[22] Donoho, D. L. and Tsaig, Y., "Fast solution of ℓ_1 minimization problems when solution may be sparse." Preprint, October 2006.

[23] Donoho, D., "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[24] Donoho, D., Elad, M., and Temlyakov, V., "Stable recovery of sparse overcomplete representations in the presence of noise," *Information Theory, IEEE Transactions on*, vol. 52, no. 1, pp. 6–18, 2006.

[25] Donoho, D. and Huo, X., "Uncertainty principles and ideal atomic decomposition," *Information Theory, IEEE Transactions on*, vol. 47, no. 7, pp. 2845–2862, Nov 2001.

[26] Donoho, D. and Johnstone, J., "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[27] Donoho, D. and Stark, P., "Uncertainty principles and signal recovery," *SIAM Journal on Applied Mathematics*, vol. 49, no. 3, pp. 906–931, 1989.

[28] Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R., "Least angle regression," *Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[29] Figueiredo, M., Nowak, R., and Wright, S., "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 4, pp. 586–597, 2007.

[30] Fuchs, J., "On sparse representations in arbitrary redundant bases," *Information Theory, IEEE Transactions on*, vol. 50, no. 6, pp. 1341–1344, 2004.

[31] Golub, G. and Van Loan, C., *Matrix Computations*. Johns Hopkins University Press, 1996.

[32] Gorodnitsky, I. and Rao, B., "Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm," *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]*, vol. 45, no. 3, pp. 600–616, 1997.

[33] Grant, M., Boyd, S., and Y.Ye., "CVX: Matlab software for disciplined convex programming." http://stanford.edu/ boyd/cvx.

[34] Horn, R. and Johnson, C., *Matrix Analysis*. Cambridge University Press, 1985.

[35] James, G., Radchenko, P., and Lv, J., "The DASSO algorithm for fitting the Dantzig selector and the Lasso," *Under review.(Available at www-rcf.usc.edu/∼gareth)*, 2007.

[36] Kim, S.-J., Koh, K., Lustig, M., Boyd, S., and Gorinevsky, D., "An interior-point method for large-scale ℓ_1-regularized least squares," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 4, pp. 606–617, 2007.

[37] Lugosi, Gabór, "Concentration-of-Measure Inequalities," Lecture notes, www.econ.upf.edu/ lugosi/anu.ps, February 2006.

[38] Mallat, S., *A Wavelet Tour of Signal Processing, Second Edition (Wavelet Analysis & Its Applications)*. Academic Press, September 1999.

[39] Meinshausen, N. and Yu, B., "Lasso-type recovery of sparse representations for high-dimensional data," *submitted to Annals of Statistics*, 2007.

[40] Natarajan, B. K., "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.

[41] Needell, D. and Tropp, J., "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Arxiv preprint arXiv:0803.2392*, 2008.

[42] Nesterov, Y. and Nemirovsky, A., "Interior Point Polynomial Methods in Convex Programming," *Studies in Applied Mathematics (SIAM)*, vol. 13, 1994.

[43] Osborne, M., Presnell, B., and Turlach, B., "A new approach to variable selection in least squares problems," *IMA Journal of Numerical Analysis*, vol. 20, no. 3, pp. 389–403, 2000.

[44] Osborne, M., Presnell, B., and Turlach, B., "On the lasso and its dual," *Journal of Computational and Graphical Statistics*, vol. 9, no. 2, pp. 319–337, 2000.

[45] Pennebaker, W. and Mitchell, J., *JPEG Still Image Data Compression Standard*. Kluwer Academic Publishers, 1993.

[46] Rao, B. and Kreutz-Delgado, K., "An affine scaling methodology for best basis selection," *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]*, vol. 47, pp. 187–200, Jan 1999.

[47] Rudelson, M. and Vershynin, R., "On sparse reconstruction from Fourier and Gaussian measurements," *Comm. Pure Appl. Math., to appear*.

[48] Shewchuk, J. R., "An introduction to the conjugate gradient method without the agonizing pain." http://www.cs.cmu.edu/ quake-papers/painless-conjugate-gradient.pdf, August 1994.

[49] Stark, H. and Woods, J. W., *Probability and Random Processes with Applications to Signal Processing*. Prentice Hall, 2001.

[50] Taubman, D. and Marcellin, M., *JPEG2000: Image compression fundamentals, standards and practice,*. Kluwer Academic Publishers Boston, 2002.

[51] Temlyakov, V., "Nonlinear Methods of Approximation," *Foundations of Computational Mathematics*, vol. 3, no. 1, pp. 33–107, 2002.

[52] Tibshirani, R., "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[53] Tropp, J., "Just relax: Convex programming methods for identifying sparse signals in noise," *Information Theory, IEEE Transactions on*, vol. 52, no. 3, pp. 1030–1051, 2006.

[54] Tropp, J., "Greed is good: Algorithmic results for sparse approximation," *Information Theory, IEEE Transactions on*, vol. 50, no. 10, pp. 2231–2242, 2004.

[55] Tropp, J. A. and Gilbert, A., "Signal recovery from random measurements via orthogonal matching pursuit," *Information Theory, IEEE Transactions on*, vol. 53, pp. 4655–4666, December 2007.