Cardinality Optimization Problems

by

Mohammad Javad Abdi

A thesis submitted to
The University of Birmingham
for the degree of
Doctor of Philosophy

School of Mathematics
The University of Birmingham
May 2013

Abstract

In this thesis, we discuss the cardinality minimization problem(CMP) and the cardinality constraint problem, which have attracted plenty of recent attention across various disciplines. These problems have a wide range of applications in such areas like signal processing, control theory, finance, economics, statistics and principal component analysis. Due to the NP-hardness of these problems, we discuss Lagrangian and SDP relaxations, reweighted l_1-techniques, smoothing, linearization, and d.c. programming techniques for finding an approximate solution to these problems. We also discuss the l_1-minimization as one of the most efficient methods for solving CMPs, and we demonstrate that the l_1-minimization uses a kind of weighted l_2-minimization. Through theoretical, geometrical and numerical methods, we show that the reweighted l_j-minimization ($j \geq 1$) is very effective to locate a sparse solution to a linear system. Next, we show how to introduce different merit functions for sparsity, and how proper weights may reduce the gap between the performances of these functions for finding a sparse solution to an undetermined linear system. Furthermore, we introduce some effective computational approaches to locate a sparse solution for an underdetermined linear system. These approaches are based on reweighted l_j-minimization ($j \geq 1$) algorithms. As a special case of reweighted approaches, we focus on the reweighted l_1-minimization. We introduce several new concave approximations to the l_0-norm function. These approximations can be employed to define new weights for reweighted l_1-minimization algorithms. We show how the change of parameters in reweighted algorithms may affect the performance of the algorithms for finding the solution of the cardinality minimization problem. In our experiments, the problem data were generated according to different statistical distributions, and we test the algorithms on different sparsity level of the solution of the problem. As a special case of cardinality constrained problems, we also discuss compressed sensing and restricted isometry property(RIP). We illustrate how the problem of finding the restricted isometry constant(RIC) is related to an optimization problem with cardinality constraints, and we discuss some techniques for an approximate solution to these problems.

Acknowledgements

First and foremost, I would like to express my special thanks to my supervisor Dr. Yunbin Zhao, who awarded me his thoughts and experiences. I feel very proud to have him as my supervisor. This thesis would not have been possible without his support.

I am thankful to Dr. Sandor Zoltan Nemeth as my co-supervisor for his support and comments.

I am really thankful to Professor Michal Kocvara for his great lectures on nonlinear and semidefinite programming, and also for being in my defense and reading committee, for many helpful discussions, comments and suggestions. I thank Dr. Houdou Qi for kindly accepting to be on my reading committee, and for providing valuable feedbacks. I also thank Dr. Alex Bespalov as the chair of my defense committee.

I am grateful to Professor Stephen Decent, who gifted me the opportunity to be a researcher at the University of Birmingham.

I am thankful to those who helped me to complete this thesis, including the lecturers and the support staff.

I owe my deepest gratitude to my parents Ehsan and Manijeh, and my sister Elahe, for their support during all of my life.

And finally, I would like to dedicate this thesis to my wife, Niloofar.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

We study the following optimization problems:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ Card(x) \\ &\quad \text{s.t. } x \in \mathcal{F}, \end{aligned} \tag{1.1}$$

and

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ f(x) \\ &\quad \text{s.t. } Card(x) \leq \tau \\ &\qquad x \in \mathcal{F}, \end{aligned} \tag{1.2}$$

where $f(x)$ is the objective function, x is a vector, τ is a constant, and $\mathcal{F}$ is the feasible set. We may suppose that $\mathcal{F}$ is a linear or nonlinear convex set or even a nonlinear and non-convex set.

We refer to the problem (1.1) as the cardinality minimization problem(CMP), and we refer to (1.2) as the cardinality constrained problem(CCP). These problems are known as NP-hard problems [62, 135, 31, 103, 51], i.e., they are computationally intractable in general.

Cardinality minimization problems(CMPs), and cardinality constrained problems(CCPs) have many applications in finance [97, 35, 126], where the cardinality constrained defines some bounds on the number of the assets, or sets the portfolio proportion. CMPs and CCPs have also lots of applications in signal and image processing [22], and compressed sensing [11, 50]. Compressed sensing tries to recover the vector (signal) $x \in \mathbb{R}^n$ using an observed vector (signal) $b \in \mathbb{R}^m$ by solving $A_{m\times n}x = b$ $(m < n)$. CMPs and CCPs have found many applications in statistics and principal component analysis(PCA) as well [47, 154]. PCA is often used to reduce the dimension of a model, or in other words, PCA tries to compress the data without losing much information. Usually PCA is reformulated as an optimization problem on eigenvalues. We will discuss some special cases of PCAs in chapter 6.

In general, cardinality minimization problem(CMP) (1.1) is looking for the sparsest solution (i.e, a vector with the maximum number of zeros) to a mathematical program. A special case of the CMP is the problem of finding the sparsest solution to an underdetermined linear system of equations, which have found so many applications across various disciplines, including signal recovery and image processing. Excellent surveys about compressed sensing can be found in [50, 33, 22].

As normally used in the literature, we use $\|x\|_0$ or $card(x)$ or l_0-norm to denote the cardinality of the vector x. Clearly, the sparsest solution of $Ax = b$ is the solution to the following problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_0 \\ &\text{s.t.}\ \ Ax = b, \end{aligned} \tag{1.3}$$

where $A \in \mathbb{R}^{m\times n}$ $(m < n)$ and $b \in \mathbb{R}^m$. Obviously, in this case, $Ax = b$ defines an underdetermined linear system of equations, which has infinitely many solutions.

The old methods for solving the above problem are mostly based on least squares, or structured total least norm [85, 68, 134, 118]. The least square problem can be written as follows:

$$\underset{x}{\text{Minimize}}\ \|Ax-b\|_2 \ \text{ s.t. } x \in \mathbb{R}^n. \tag{1.4}$$

This problem is an unconstrained convex problem, and its optimal solution minimizes the residuals, $r = Ax - b$. The most common application of the least square method is in data fitting and linear and non-linear regression problems [98, 139].

The most well known heuristic approach for solving the problem (1.3) is to replace $\|x\|_0$ by $\|x\|_1$, and solve the following convex problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_1 \\ &\qquad \text{s.t. } Ax = b, \end{aligned} \tag{1.5}$$

which is called the basis pursuit problem. This problem has lots of applications, especially in signal processing [41, 58, 30, 22, 123, 26]. Except for solving the problem (1.5), sometimes greedy methods are also used in the literature. These methods include the orthogonal matching pursuit(OMP) [132, 55]. OMP algorithm starts with a sparse solution and then searches for a sparser solution iteratively. This procedure finds an approximate solution to the problem by linear combination of some selected columns of matrix A. To find a sparser solution, at each iteration some new column is added to the existing columns to build a possibly sparser solution. OMP is a relatively strong algorithm to solve CMPs, however the l_1-minimization is more successful than OMP algorithms in finding a sparse solution, in many situations [51, 131]. Therefore, we discuss the l_1-minimization with more details in chapter 4.

It is worth mentioning that the combination of the problem (1.4) with l_1-norm constraints has been considered by Tibshirani in 1995 [129]. This problem is called LASSO-type problem, which can be cast as

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|Ax-b\|_2 \\ &\text{s.t. } \|x\|_1 \le \tau. \end{aligned} \tag{1.6}$$

This problem has found many applications in regression selections [137, 145]. The unconstrained version of the LASSO-type problem can be written as follows:

$$\underset{x}{\text{Minimize}}\ \|Ax-b\|_2 + \lambda\|x\|_1, \quad x \in \mathbb{R}^n, \tag{1.7}$$

where $\lambda \gg 0$ is a penalty parameter.

As mentioned, the l_1-minimization is considered as one of the most successful methods to locate a sparse solution to a system of linear equations. So, seeking for even more effective algorithms than the l_1-minimization is one of the developing and ongoing researches in this area. Numerical experiments show that the reweighted l_1-minimization outperform the l_1-minimization in many situations [152, 82, 34, 140, 37, 43].

Candes, Wakin, and Boyd [34] proposed the following reweighted l_1-algorithm (CWB-algorithm):

$$\begin{aligned} x^{l+1} &= \text{argmin} \sum_{i=1}^{n} \frac{1}{|x_i^l|+\epsilon}|x_i| \\ &\text{s.t. } Ax = b, \end{aligned} \tag{1.8}$$

where l is the number of the iteration, and x_i^l is the i^{th} component of the solution from the l^{th} iteration. The term $\frac{1}{|x_i^l|+\epsilon}$ can be interpreted as the penalty, which encourages the small component to tend to zero quickly if possible. We will discuss the CWB-algorithm with more details in the chapters 4 and 5. It is shown in [61] that after enough number of iterations, the problem (1.8) may solve the following problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \sum_i log(|x_i| + \epsilon) \\ &\text{s.t.} \ \ Ax = b. \end{aligned} \tag{1.9}$$

In [140], it is demonstrated that if ϵ is set to be zero in the problem (1.9), then the iterative reweighted l_2-method for solving this problem can be interpreted as the FOCal Underdetermined System Solver algorithm (FOCUSS algorithm). The FOCUSS algorithm was first introduced by Gorodnitsky, Rao et al [115, 71, 70].

Here we should mention that the cardinality minimization problem over polyhedral sets was considered by Mangasarian in 1995 [96]. He approximated the cardinality function ($\|x\|_0$) with a smooth concave exponential approximation, and used a finite linear-programming-based iterative method (successive linearization algorithm) to solve the problem [21, 94]. Also, he proved that this algorithm converges to a stationary point (vertex point) of the approximation problem after a finite number of iterations.

Recently, Zhao and Li [152] also used concave approximations to the cardinality function and proposed a unified framework to construct reweighted l_1-algorithms. They illustrated how to define merit functions for sparsity. These merit functions are some proper concave approximations to the function $\|x\|_0$, and minimizing such functions may lead us to a sparse solution of the problem. Also, they defined range space property(RSP) for the matrix A, under which their algorithm may converge to a sparse solution. In chapter 5, we will focus on the method proposed by Zhao and Li [152], and we will construct more new concave approximations to the function $\|x\|_0$ and we will explain how to define more new reweighted l_1-algorithms along the line in [152].

The reweighted l_1-minimization is a new topic in the field of optimization and applied mathematics, but reweighted least squares methods(RLS) have been introduced by Lawson in 1960s [84]. The RLS problem can be written as follows:

$$x^{l+1} = \text{argmin} \sum_{i=1}^{n} w_i^l (Ax^l - b)_i^2, \tag{1.10}$$

where $(Ax^l - b)_i$ is the i^{th} component of the residual at the iteration l, and $w_i^1 \in \mathbb{R}_+^n$ is the weight. Reweighted least square problems have been extended to l_p-minimization $(0 < p < 1)$, which is non-convex problem [42, 43]. RLS has many applications in maximum likelihood estimates problems [5], and robust regression [111]. Note that RLS can also be considered as the reweighted l_2-algorithm [70, 38, 63], which will be discussed in chapter 4. We show that the l_1-minimization itself is a kind of weighted l_2-minimization, which is hidden inside the l_1-norm function. We also give some theoretical, geometrical, and numerical explanations for promoting weighted approaches. Hence, we will consider the following general form of weighted l_j-minimization problems:

$$\begin{aligned} x^{l+1} &= \text{argmin } \|W^l x\|_j \\ &\text{s.t. } Ax = b, \end{aligned} \tag{1.11}$$

where $j \geq 1$, and W^l is a diagonal matrix at the iteration l, with the weights on its diagonal. Note that in this thesis, the iterative weighted problems are called reweighted problems, i,e. weights are updated in each iteration. In chapter 4, we mainly focuss on the case when $j \geq 2$, and we will focuss on reweighted l_1-minimization in the chapter 5.

1.1 Outline of the thesis

In chapter 2, we review some basic concepts of convex optimization problems, duality and semidefinite programming. Besides, we discuss rank minimization problems(RMPs) briefly, as these problems are closely related to the cardinality minimization problem.

In chapter 3, we start with the optimization problems with cardinality constraints. Different SDP relaxations are provided based on Shor's lemma for these problems. Next, we proceed with the cardinality minimization problem(CMP) under non-convex quadratic constraints. We apply reformulation techniques combined with Lagrange duality methods, and Shor's lemma to relax the CMP to an SDP form. Furthermore, we show how the CMP can be cast as a bilevel optimization problem. Also, we discuss the l_1-minimization and reweighted l_1-methods for finding an approximate solution to the CMP. We introduce a continuous approximation to the $\|x\|_0$-function, and we explain how to apply linearization methods combined with the reweighted l_1-minimization method to get an approximate solution to the CMP. We continue to develop the reweighted minimization approaches in chapters 4 and 5. Finally, in chapter 3, we review branch and bound algorithms and subgradient methods. We also provide some numerical experiments based on these methods.

In chapter 4, we discuss the l_1-minimization and reweighted l_j-minimizations ($j \geq 1$) in details. To motivate the weighted approaches, we first show that a certain weighted l_2-minimization is hidden inside the l_1-minimization, and through theoretical, geometrical and numerical studies, we prove that weighted approaches are very effective to locate a sparse solution to a linear system. Also, we illustrate that choosing proper weights may reduce the gap between different merit functions for sparsity. In addition, we demonstrate how to construct different new merit functions for sparsity, and how to introduce new effective weights to reweighted algorithms by applying different merit functions. In this chapter, the choice of the parameter ϵ in the weights will be also discussed, and the performances of different algorithms will be compared through the numerical experiments.

In chapter 5, we focuss on reweighted l_1-algorithms. We introduce different new concave approximations to the $\|x\|_0$-function, and we show how to generate more approximations to the $\|x\|_0$-function, and how to introduce new reweighted l_1-algorithms based on these approximation functions. Besides, we show how the change of parameters in reweighted algorithms may affect the performances of the algorithms to find the sparsest solution of the cardinality minimization problem. In our experiments, the problem data will be generated according to different statistical distributions (as in the literatures usually normally distributed matrices are discussed), and we test the algorithms on different sparsity levels of the solution of the problem.

In chapter 6, we discuss some topics on compressed sensing and restricted isometry property(RIP). Following that, we explain how the problem of finding the restricted isometry constant(RIC) is related to a sparse eigenvalue problem, which itself is a cardinality constrained problem. Next, to find an approximate solution to RIC as a special case of the cardinality constrained problem, we study such methods as relaxation, smoothing, and d.c. programming techniques.

Finally, in chapter 7, we summarize the main results of this thesis.

Chapter 2

Preliminaries

In this chapter, we review some basic definitions and concepts of conic and semidefinite programming, [12, 20, 110, 75]. Also, we discuss rank minimization problems briefly, as these problems are closely related to the cardinality minimization problem.

2.1 General optimization problems, duality, and KKT conditions

An optimization problem in standard form is defined as follows:

$$
\begin{aligned}
\underset{x}{\text{Minimize}}\ & f_0(x) \\
\text{s.t.}\ & f_i(x) \leq 0, \quad i = 1, ..., p, \\
& h_i(x) = 0, \quad i = 1, ..., q\ ,
\end{aligned} \tag{2.1}
$$

where $x \in \mathbb{R}^n$ is the optimization variable. $f_0 : \mathbb{R}^n \to \mathbb{R}$ is the objective function, and the functions $f_i : \mathbb{R}^n \to \mathbb{R}$ and $h_i : \mathbb{R}^n \to \mathbb{R}$ are called inequality constraint functions, and equality constraint functions, respectively.

The Lagrangian $L : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$ associated with the problem above is

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{p} \lambda_i f_i(x) + \sum_{i=1}^{q} \nu_i h_i(x),$$

where the vector $\lambda = (\lambda_1, ..., \lambda_p)$ and $\nu = (\nu_1, ..., \nu_q)$ are the dual variables or Lagrange multiplier vectors associated with the problem (2.1).

The Lagrangian dual function, $g : \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$, is defined as the minimum value of the Lagrangian over x, i.e., for $\lambda \in \mathbb{R}^p$ and $\nu \in \mathbb{R}^q$, we have

$$g(\lambda, \nu) = \inf_{x \in \mathcal{F}} L(x, \lambda, \nu),$$

where $L(x, \lambda, \nu)$ is the Lagrangian defined above, and

$$\mathcal{F} = \left(\bigcap_{i=0}^{p} dom(f_i) \right) \cap \left(\bigcap_{i=1}^{q} dom(h_i) \right).$$

For each pair (λ, ν), where $\lambda \geq 0$, the Lagrangian dual function provides a lower bound for the optimal value of the problem (2.1). Hence, looking for the best lower bound leads us to the following optimization problem:

$$\begin{aligned} &\underset{\lambda,\nu}{\text{Maximize}}\ g(\lambda, \nu) \\ &\text{s.t. } \lambda \geq 0. \end{aligned} \tag{2.2}$$

The above problem is called the dual problem associated with the primal problem (2.1).

Assume that f_i, $\forall i = 0, ..., p$, and h_i, $\forall i = 1, ..., q$ are differentiable in the primal problem, and let x^* and (λ^*, ν^*) be any primal dual optimal points with zero duality gaps, i.e, there is no gap between the optimal solutions of the primal problem and the dual problem. Since x^* minimizes $L(x, \lambda^*, \nu^*)$ with respect to x, then at x^* we have

$$\nabla f_0(x^*) + \sum_{i=1}^{p} \lambda_i{}^* \nabla f_i(x^*) + \sum_{i=1}^{q} \nu_i{}^* \nabla h_i(x^*) = 0.$$

Therefore, (x^*, λ^*, ν^*) satisfies

$$\left\{ \begin{array}{c} f_i(x^*) \leq 0, \quad i = 1, ..., p, \\ h_i(x^*) = 0, \quad i = 1, ..., q, \\ \lambda_i{}^* \geq 0, \\ \lambda_i{}^* f_i(x^*) = 0, \\ \nabla f_0(x^*) + \sum_{i=1}^{p} \lambda_i{}^* \nabla f_i(x^*) + \sum_{i=1}^{q} \nu_i{}^* \nabla h_i(x^*) = 0. \end{array} \right. \tag{2.3}$$

The above conditions are called Karush-Kuhn-Tucker(KKT) conditions. For any optimization problem with differentiable objective functions and constraint functions for which the constraint qualification holds, the optimal points must satisfy the KKT conditions. So, KKT conditions are necessary for the point (x^*, λ^*, ν^*) to be optimal. If the primal problem is convex, then KKT conditions are also sufficient conditions for the points to be optimal.

In the next section, we simply review convex optimization problems. More discussions about duality theory can be found in the conic optimization section.

2.2 Convex optimization

Definition 2.1. *(Convex set)*

A subset C in $\mathbb{R}^n$ is convex if

$$\alpha x + (1 - \alpha) y \in C, \quad \forall x, y \in C, \quad 0 \leq \alpha \leq 1. \tag{2.4}$$

The set $\{\alpha x + (1-\alpha)y : \alpha \in [0,1]\}$ represents the closed line segment between x and y.

We call a point of the form $\sum_{i=1}^{k} \alpha_i x_i$, where $\sum_{i=1}^{k} \alpha_i = 1$, and $\alpha_i \geq 0,\ \forall i = 1, ..., k$ a convex combination of the points $x_1, ..., x_k$. Clearly, a set is convex if and only if it contains all convex combinations of its points.

Example 2.1.

- *An empty set, a singleton and the whole space $\mathbb{R}^n$ are the simple examples of convex sets.*

- *Affine subspaces of $\mathbb{R}^n$ are convex. In particular, a polyhedral is a convex set, since it is the solution set of a finite system defined as $Ax \leq b$.*

- *Euclidean balls and ellipsoids are convex. An Euclidean ball centered at a with a radius r is*

$$\{x : ||x-a||_2 \leq r\},$$

where $||\cdot||_2$ is the standard Euclidean norm, $||x||_2 = \sqrt{x^T x}$. An ellipsoids centered at a, with a given matrix $Q \in \mathbb{R}^{m\times n}$ is

$$\{x \in R^n : ||Q(x-a)||_2^2 \leq 1\}.$$

Definition 2.2. *(Convex hull)*

Let $K \subseteq \mathbb{R}^n$ be a nonempty arbitrary set. The intersection of all convex sets containing K, is the convex hull of K. In other words the convex hull of K, denoted by $Conv(K)$, is the smallest convex set containing K.

The convex hull of $m+1$ affinely independent points $x_0, x_1, ...x_m \in \mathbb{R}^n$, (i.e., $x_1 - x_0, ..., x_m - x_0$ are affinely independent) is called a m-dimensional simplex, defined as follows,

$$Conv(x_0, ..., x_m) = \left\{ \sum_{i=0}^{m} \alpha_i x_i : \sum_{i=0}^{m} \alpha_i = 1, \alpha_i \geq 0 \right\}.$$

Definition 2.3. *(Convex function)*

$f : \mathbb{R}^n \to \mathbb{R}$ is a convex function, if $dom(f)$ is a convex set, and

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y), \quad \forall x, y \in dom(f),\ 0 \leq \lambda \leq 1.$$

Example 2.2. *If $\| \cdot \| : \mathbb{R}^n \to \mathbb{R}$ is a norm, and $0 \leq \alpha \leq 1$, then from the triangle inequality and homogeneity of the norm we have*

$$\|\alpha x + (1-\alpha)y\| \leq \|\alpha x\| + \|(1-\alpha)y\| = \alpha \|x\| + (1-\alpha)\|y\|,$$

so, the norm is a convex function.

Here we review the first and second order convexity conditions. First order convexity condition for a differentiable function f is as follows:

f is convex, if and only if $dom(f)$ is convex, and

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \ \ \forall x, y \in dom(f).$$

If f is twice differentiable, the second order convexity condition is as follows:

f is convex, if and only if $dom(f)$ is convex, and

$$\nabla^2 f(x) \succeq 0 \ \ \forall x \in dom(f).$$

Example 2.3. *The function*

$$f : \mathbb{R}^n \to \mathbb{R}$$

$$f(x) = \frac{1}{2}x^T Bx + a^T x + c,$$

where $B \in \boldsymbol{S}^n$, $a \in \mathbb{R}^n, c \in \mathbb{R}$, is convex if and only if $B \succeq 0$, since $\nabla^2 f(x) = B$.

Definition 2.4. *(Quasiconvex functions)*

$f : \mathbb{R}^n \to \mathbb{R}$ is quasiconvex if the domain of f and its sublevel sets $\{x \in dom\ f : f(x) \leq \beta\}$, for $\beta \in \mathbb{R}$ are convex.

Quasiconvex functions can also be defined by the following expressions:

A function f is quasiconvex if and only if its domain is convex, and

$$f(\alpha x + (1-\alpha)y) \leq \max\{f(x), f(y)\}, \ \ \forall x, y \in dom(f), \ 0 \leq \alpha \leq 1,$$

and f is quasiconcave if

$$f(\alpha x + (1-\alpha)y) \geq \min\{f(x), f(y)\}, \ \ \forall x, y \in dom(f), \ 0 \leq \alpha \leq 1.$$

Example 2.4. *Cardinality function on $\mathbb{R}^n_+$, and rank function on $\boldsymbol{S}^n_+$, are quasiconcave. In fact,*

$$Card(x+y) \geq \min\{Card(x), Card(y)\}, \ x, y \geq 0,$$

$$Rank(X+Y) \geq \min\{Rank(X), Rank(Y)\}, X, Y \in \boldsymbol{S}^n_+,$$

where $\boldsymbol{S}^n_+$ denotes the cone of the positive semidefinite matrices.

In general, a convex optimization problem is of the form

$$\begin{aligned} \underset{x}{\text{Minimize}} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0, \end{aligned}$$

where f, g are convex functions. Roughly speaking, minimizing a convex function over a convex set is called convex programming.

If one can reformulate a problem as a differentiable convex optimization problem, then it can be solved efficiently, in general. However, reformulating a problem as a convex optimization can be challenging. Heuristics based on convex optimization have an important role for solving non-convex problems. As we will see in chapter 3, one of the goals of this thesis is to study the convex optimization problems based on heuristic methods for cardinality minimization and cardinality constrained problems. For example, one of the methods for solving the CMP problem is to apply l_1-norm heuristics.

Next, we are going to introduce conic optimization. Before doing so, we need some basic definition and facts.

2.3 Conic optimization and duality

Definition 2.5. *(Cone)*

A cone $\boldsymbol{K} \subseteq \mathbb{R}^n$ is a proper cone, if it satisfies

1. *$\boldsymbol{K}$ is nonempty and closed under addition, i.e.,*

$$a, b \in \boldsymbol{K} \Rightarrow a + b \in \boldsymbol{K},$$

   and $\boldsymbol{K}$ is a conic set, i.e.,

$$a \in \boldsymbol{K}, \gamma \geq 0 \Rightarrow \gamma a \in \boldsymbol{K},$$

   (if these two conditions hold then $\boldsymbol{K}$ is convex),

2. *$x \in \boldsymbol{K}$, and $-x \in \boldsymbol{K} \Rightarrow x = 0$, which means $\boldsymbol{K}$ is* pointed*,*

3. *$\boldsymbol{K}$ has nonempty interior, int($\boldsymbol{K}$) $\neq \phi$,*

4. ***K*** *is closed.*

If 1,2 above hold then $\mathbf{K}$ is a pointed convex cone.

The partial ordering using a pointed convex cone K, denoted by $\geq_{\mathbf{K}}$ or $\succeq$, can be defined as:

$$a \succeq b \Leftrightarrow a - b \succeq 0 \Leftrightarrow a - b \in \mathbf{K}.$$

Example 2.5. *(Some examples of cones)*

- *The nonnegative orthant* $\mathbb{R}_+^m = \{x = (x_1, ..., x_m) \in \mathbb{R}^m : x_i \geq 0, i = 1, ..., m\}$ *in* $\mathbb{R}^n$.
- *Lorentz or second order cone or ice cream, defined as,*

$$\mathbb{L}^m := \left\{ x = (x_1, x_2, ..., x_{m-1}, x_m) \in \mathbb{R}^m ; x_m \geq \sqrt{\sum_{i=1}^{m-1} x_i^2} \right\}.$$

- *The semidefinite cone* S_+^m*, equipped with Frobenius inner product, and defined in the space of* $m \times m$ *symmetric matrices.*

For more explanation about the last example above, we need to introduce Frobenius norm, which is based on the concept of inner product. Frobenius norm is used in semidefinite optimization.

Definition 2.6. *(Inner product)*

Define the linear vector inner product as follows,

$$\langle x, y \rangle = x^T y.$$

where the x, y *are the vectors with the same dimension. Matrix inner product can be computed using vectorization of the matrices. For an arbitrary* $m \times n$ *matrix* $Y =$

$\begin{pmatrix} y_1, & \dots & , y_n \end{pmatrix}$, *where y_i's denote the columns of Y, we may define,*

$$vecY := \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

Then the inner product of two matrices X and Y with the appropriate dimensions will be defined as follows:

$$\langle Y, X\rangle := tr(Y^T X) = (vecY)^T vecX = \mathbf{1}^T (Y \circ X)\mathbf{1},$$

where $\mathbf{1}$ *is a vector with all its components equal to 1, and $\circ$ represents the Hadamard Product of the matrices, [69].*

Definition 2.7. *(Frobenius norm)*

Assume that $X \in \mathbb{R}^{m\times n}$ is a normal matrix, i.e., it has a complete normal set of eigenvectors [146]. Then Frobenius norm is defined using vector inner product as follows:

$$||X||_F^2 = ||vecX||_2^2 = \langle X, X\rangle = tr(X^T X) = \sum_i \sigma(X)_i^2,$$

where $\sigma(X)_i$ is the ith singular value of X.

Now, we are ready to represent the general conic optimization problem. Denote $\mathbf{E}$ as an Euclidean space with finite dimension, equipped with both inner product $\langle .,.\rangle$ and partial ordering ($\succeq$ or $\geq_{\mathbf{K}}$), and let $\mathbf{K}$ be a proper cone in $\mathbf{E}$. $c \in \mathbb{R}^n$ is the objective, $b \in \mathbf{E}$, and $x \to Ax$ is a linear mapping defined with a $A_{m\times n}$ matrix. Then the conic optimization problem(CP) is in the form of

$$\min_x \{c^T x : Ax \succeq b\}. \tag{2.5}$$

Obviously, the linear programming(LP),

$$\min_{x}\{c^T x; Ax \geq b\}, \tag{2.6}$$

is a special case of (2.5), in which $\mathbf{E}$ and $\mathbf{K}$ are replaced with the whole real space and its nonnegative orthant respectively.

One of the important systematic ways for finding a lower bound for the optimal value of a linear program is to use LP duality [15]. The main idea in LP duality is to add a weighted sum of the constraint functions to the objective function to achieve the Lagrangian, then the minimum value of the Lagrangian over x is called the Lagrange dual function. Applying the above method to the linear programming (2.6) and denoting the weighted vectors with $\omega \geq 0$, we have

$$\omega^T Ax \geq \omega^T b,$$

Obviously, if $A^T\omega = c$, then $\omega^T b$ is a lower bound for $c^T x$. Hence, to achieve the best lower bound one should solve

$$\max_{A^T\omega=c,\omega\geq 0} b^T\omega.$$

One can use the same methodology to build a dual for a conic problem (2.5). Note that the weight vectors, in this case, should be chosen from the *dual cone*, which is defined as follows:

$$\mathbf{K}^* = \{\omega \in \mathbf{E} : \langle \omega, k\rangle \geq 0, \forall k \in \mathbf{K}\}. \tag{2.7}$$

Note that in the LP dual problem the weight vectors have been chosen from the nonnegative orthant, which is a convex set.

Proposition 2.1. *Dual cone is always convex even if the cone itself is not convex.*

Proof. As we have seen, the dual cone is defined as

$$\mathbf{K}^* = \{\omega \in \mathbf{E} : \omega^T k \geq 0, \forall k \in \mathbf{K}\}. \tag{2.8}$$

Assume that $\omega, z \in \mathbf{K}^*$, then for any $k \in \mathbf{K}$, and $0 \leq \alpha \leq 1$, we have

$$(\alpha\omega + (1-\alpha)z)^T k = \alpha\omega^T k + (1-\alpha)z^T k \geq 0,$$

so $\alpha\omega + (1-\alpha)z \in \mathbf{K}^*$. Hence, $\mathbf{K}^*$ is convex.

□

As an example, note that nonnegative orthant is self dual, i.e., $\mathbb{R}^m_+ = (\mathbb{R}^m_+)^*$, so is the cone of semidefinite positive matrices, which is a nice property for semidefinite programming. To derive a dual problem for the conic problem (2.5), the weight vectors should be chosen from $\mathbf{K}^*$ and repeat the same procedure as for LP dual to get:

$$\max\{\langle b, \omega\rangle : A^T\omega = c, \omega \in \mathbb{K}^*\}. \tag{2.9}$$

Recall that, $\omega \in \mathbb{K}^*$, means $\omega \geq_{\mathbb{K}^*} 0$.

The following theorem illustrates the importance of Lagrange duality.

Theorem 2.1. *(Conic duality theorem) [12]*

For the conic problem (2.5) and its dual (2.9) the following statements hold.

- *The duality is symmetric.*
- $c^T x - \langle b, \omega\rangle \geq 0$ *for all feasible points.*
- *If the primal conic problem(CP) is bounded below and Slater's condition holds (i.e., the problem is strictly feasible), then the dual is solvable and the optimal values are equal.*

- *If the dual problem is bounded above and Slater's condition holds (i.e., the dual problem is strictly feasible), then the primal conic problem is solvable and the optimal values are equal.*
- *If one of the problems (primal or dual) is bounded and strictly feasible, then the pair (x, ω) is the optimal solution to the primal and dual respectively, if and only if*

  $$cx - \langle b, \omega \rangle = 0,$$

  which means the duality gap is zero, and

  if and only if

  $$\langle \omega, Ax - b \rangle = 0,$$

  which means that the complementary slackness holds.

Before we proceed to semidefinite optimization, let us discuss a bit of conic quadratic representable sets.

Definition 2.8. *(Conic quadratic representable sets)*

We say a set $X \subset \mathbb{R}^n$ is Conic Quadratic representable(CQr), or it can be represented using conic quadratic inequalities, if there exists a finite system S with the form of

$$A_j \begin{pmatrix} x \\ u \end{pmatrix} - b_j \geq_{L^{m_j}} 0,$$

where $x \in \mathbb{R}^n$ and additional variable u, such that X is the projection of the solution set of S onto the x-space.

Or in a shorter form one can write,

$$x \in X \Leftrightarrow \exists u : A_j \begin{pmatrix} x \\ u \end{pmatrix} - b_j \geq_{L^{m_j}} 0, \quad j = 1, ..., N.$$

Every such a system S is called a Conic Quadratic Representation(CQR) of the set X. If a function f is Conic Quadratic(CQ) representable, then so all of its level sets, and every Conic Quadratic(CQ) representable (of the epigraph of f) explicitly induces CQ-representation of the level sets. Recall that the epigraph of a function f is defined as

$$Epi(f) = \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : f(x) \leq t\}.$$

Suppose that we have a CQr of the epigraph of f. Then

$$f(x) \leq t \Leftrightarrow \exists u : \|\xi_j(x,t,u)\|_2 \leq \zeta_j(x,t,u), \quad j = 1, ..., N,$$

where ξ_j is a vector valued affine function, and ζ_j is a scalar valued affine function. The inequality on the right hand side above has been achieved directly from the definition of the second order cone.

In the next section, we review semidefinite optimization concepts, which is a special case of conic optimization.

2.4 Semidefinite optimization

Definition 2.9. *(Positive semidefinite matrices)*

The matrix $A \in \boldsymbol{S}^n$ is positive semidefinite if $x^T A x \geq 0$ for all $x \in \mathbb{R}^n$,where $\boldsymbol{S}^n$ is the space of symmetric matrices.

Replacing $\geq$ with $>$ above leads to the definition of a positive definite matrix.

It is not hard to see that if $A \in \mathbf{S}^n$, then the following statements are equivalent.

- $A \succeq 0$,
- $\lambda_i(A) \geq 0 \ \ \forall i = 1, ..., Rank(A)$,
- $A = CC^T$ for certain rectangular matrix C, for such C, $Rank(C) = Rank(A)$,
- $A = LL^T$ for certain lower triangular L, $Rank(L) = Rank(A)$,
- $\langle A, B \rangle \geq 0$, $\forall B \in \mathbf{S}_+^n$.

Semidefinite programming is a special case of conic programming, which can be defined as optimizing a linear objective over the cone of positive semidefinite matrices with linear matrix inequalities(LMIs) [109]. Replacing $\mathbf{K}$ with the cone of positive semidefinite $m \times m$ matrices $\mathbf{S}_+^m$ in (2.5) and defining a linear mapping $\boldsymbol{A} : \mathbb{R}^n \to \mathbf{S}^m$ yields the following generic semidefinite optimization problem:

$$\min_x \{c^T x : \boldsymbol{A}x - B \succeq 0\}, \tag{2.10}$$

where " $\succeq$ " means " $\geq_{\mathbf{S}_+^m}$ ", and

$$\boldsymbol{A}x = \sum_{i=1}^{n} x_i A_i, \ \ A_i \in \mathbf{S}^m, i = 1, ..., n.$$

As we have seen, a semidefinite program for a variable $x \in \mathbb{R}^m$ can be also cast as

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ c^T x \\ &\quad \text{s.t.} \ \ A(x) \succeq 0, \end{aligned} \tag{2.11}$$

where $A(x)$ can be defined as follows:

$$A(x) = A_0 + \sum_{i=1}^{m} x_i A_i, \tag{2.12}$$

with $c \in \mathbb{R}^m$, and $m+1$ symmetric matrices $A_0, A_1, ..., A_m \in \mathbb{R}^{n\times n}$.

The inequality $A(x) \succeq 0$ is called linear matrix inequality(LMI). Note that the optimal solution x^* of the problem (2.11) is on the boundary of the feasible region, so, $A(x^*)$ is singular.

Semidefinite programming(SDP) can be considered as an extension of linear programming(LP). The following example illustrates this.

Example 2.6. *As a special example of an SDP consider the following linear program(LP):*

$$\begin{aligned} &\underset{x}{Minimize}\ c^T x \\ &\quad s.t.\ Dx + b \geq 0, \end{aligned} \tag{2.13}$$

where, $\geq$ is componentwise. Since the both sides of the inequality are vectors, obviously the following holds,

$$Dx + b \geq 0 \Leftrightarrow A(x) = diag(Dx+b) \succeq 0,$$

i.e., in (2.12) we have

$$A_0 = diag(b), \quad A_i = diag(d_i), \quad i = 1, ..., m,$$

and $D = [d_1, ..., d_m] \in \mathbb{R}^{n\times m}$. So, one can reformulate the problem (2.13) as the following SDP:

$$\begin{aligned} &\underset{x}{Minimize}\ c^T x \\ &\quad s.t.\ A(x) = diag(Dx+b) \succeq 0. \end{aligned} \tag{2.14}$$

The above example illustrates that SDP can be considered as an extension of LP, with replacing the componentwise inequalities with matrix inequalities. In other words, replacing the first orthant in LP, with the cone of semidefinite positive matrices, can be resulted in SDP problem. Another point about the relationship between SDP and LP is that the matrix inequality $A(x) \succeq 0$ means $y^T A(x) y \geq 0 \;\; \forall y \in \mathbb{R}^n$. So, SDPs can be interpreted as a semi-infinite LPs. It is not strange that the approaches to solve SDPs are closely related to LPs, while many methods and algorithm which are used to deal with LPs can be generalized for SDPs. However, there are some important differences between them. For example, one can not use the practical simplex method for LPs to solve SDPs. Also, the duality results for SDPs are weaker than those of LPs.

Remark 2.1. *Based on the definition of CQr sets in the previous section, one can also define semidefinite representable sets(SDr) as follows:*

A set $X \subset \mathbb{R}^n$ is SDr if the following holds,

$$x \in X \Leftrightarrow \exists u : \boldsymbol{A} \begin{pmatrix} x \\ u \end{pmatrix} - B \succeq 0.$$

In other words, X is SDr if there exists Linear Matrix Inequalities (LMIs), in a way that

$$\boldsymbol{A} \begin{pmatrix} x \\ u \end{pmatrix} - B \succeq 0,$$

where u is the additional variable, such that X is the projection of the solution set of the LMI onto the x-space. An LMI with this property is called the Semidefinite Representation (SDR) of the set X [12].

2.5 Rank minimization

In this section, we discuss *rank* minimization problem, and we review some facts about this problem, as we use them later in this thesis.

Definition 2.10. *(Range and Kernel of a Linear Transformation)*

The range (or image) of a linear transformation $A : \mathbb{R}^n \to \mathbb{R}^m$, $\mathcal{R}(A)$ (or $Im(A)$) is defined by

$$\mathcal{R}(A) = \{u \in \mathbb{R}^m : u = Av \text{ for some } v \in \mathbb{R}^n\}.$$

If the columns of A are written as $[a_1, ..., a_n]$, then

$$\mathcal{R}(A) = Span\{a_1, ..., a_n\}.$$

So, directly from the definition we have $\mathcal{R}(A) \subseteq \mathbb{R}^m$.

The null space (or kernel) of $A : \mathbb{R}^n \to \mathbb{R}^m$, denoted by $\mathcal{N}(A)$ (or $Ker(A)$), is defined by

$$\mathcal{N}(A) = \{v \in \mathbb{R}^n : Av = 0\}.$$

Clearly, we have $\mathcal{N}(A) \subseteq \mathbb{R}^n$.

Definition 2.11. *(Rank, and Nullity)*

Let $A : \mathbb{R}^n \to \mathbb{R}^m$. Then the column rank of A is $rank(A) = dim\mathcal{R}(A)$, i.e., $rank(A)$ is the maximum number of independent columns. In the same way the row rank of A is $dim\mathcal{R}(A^T)$. Also $dim\mathcal{N}(A)$ is the nullity of A.

For more illustration, see the following theorem:

Theorem 2.2.

$$Rank(A) = dim\mathcal{R}(A) = dim\mathcal{N}(A)^{\perp} = dim\mathcal{R}(A^T) = Rank(A^T).$$

Definition 2.12. *(Moore-Penrose Pseudoinverse)*

Let $A : \Theta \to \Gamma$ be a linear transformation, where Θ, Γ are finite dimensional vector spaces. Define a transformation $B : \mathcal{N}(A)^{\perp} \to \mathcal{R}(A)$ in the following sense

$$Bx = Ax, \quad \forall x \in \mathcal{N}(A)^{\perp}.$$

Then as it was proved in Theorem 2.2, B is both one to one and onto. Now the Moore-Penorse Pseudoinverse of A, denoted by $A^{\dagger}$ is defined as follows

$$\begin{aligned} A^{\dagger} : \Gamma \to \Theta, \qquad A^{\dagger}y &= B^{-1}y_1, \\ \textit{where } y &= y_1 + y_2, \quad y_1 \in \mathcal{R}(A), \quad y_2 \in \mathcal{R}(A)^{\perp}. \end{aligned}$$

Theorem 2.3. *(Singular Value Decomposition) Let $A_{m\times n}$ be a matrix with $rank(A) = r$. Then there exists orthogonal matrices $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$, such that*

$$A = U\Sigma V^T,$$

where

$$\Sigma = \begin{pmatrix} \sigma & 0 \\ 0 & 0 \end{pmatrix}, \quad \sigma = diag(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{r\times r}, \quad \sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0.$$

σ_i's are called singular values of the matrix A.

The above theorem also can be cast as

With

$$U_1 \in \mathbb{R}^{m\times r}, \quad U_2 \in \mathbb{R}^{m\times(m-r)}, \quad V_1 \in \mathbb{R}^{n\times r}, \quad V_2 \in \mathbb{R}^{n\times(n-r)},$$

$$A = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \sigma_{r\times r} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} = U_1 \sigma V_1^T. \tag{2.15}$$

Proof. Obviously the eigenvalues of the matrix A^TA are nonnegative, since $A^TA \succeq 0$. Define these eigenvalues as follows:

$$\{\sigma_i^2 : i = 1, ..., n\}, \text{ such that } \sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0, \quad \sigma_{r+1} = \sigma_{r+2} = ...\sigma_n = 0,$$

and define the related orthonormal eigenvectors as $\{v_i : i = 1, ..., n\}$.

And let

$$V_1 = [v_1, ..., v_r], \quad V_2 = [v_{r+1}, ..., v_n], \quad \sigma = diag(\sigma_1, ...\sigma_r).$$

Then we have

$$A^TAV_1 = V_1\sigma^2 \Rightarrow V_1^TA^TAV_1 = V_1^TV_1\sigma^2 = \sigma^2, \text{ since } V_1^TV_1 = I$$

$$\Rightarrow \underbrace{\sigma^{-1}V_1^TA^T}_{U_1^T}\underbrace{AV_1\sigma^{-1}}_{U_1} = I.$$

By the definition of $U_1,\ U_1^T$ above, we have $U_1^TU_1 = I$.

Also,

$$A^TAV_2 = V_2 0 = 0 \Rightarrow V_2^TA^TAV_2 = 0 \Rightarrow AV_2 = 0.$$

Now, construct a matrix $U_2 \in \mathbb{R}^{m\times(m-r)}$ such that $[U_1\ U_2]$ is orthogonal, i.e, $U_2^TU_1 = 0$.

Hence, one can write

$$U^T AV = \begin{pmatrix} U_1^T AV_1 & U_1^T \underbrace{AV_2}_{0} \\ U_2^T AV_1 & U_2^T \underbrace{AV_2}_{0} \end{pmatrix} = \begin{pmatrix} \sigma & 0 \\ 0 & 0 \end{pmatrix}.$$

The proof is complete. □

Singular values of a matrix A can be interpreted as the eigenvalues of the matrix $\sqrt{(AA^T)}$, i.e,

$$\sigma_i(A) = \lambda_i((AA^T)^{\frac{1}{2}}).$$

The singular value decomposition theorem for $A \in \mathbb{R}^{m\times n}$, $(m \leq n)$ can also be represented as

$$A = \sum_{i=1}^{m} \sigma_i(A) u_i v_i^T,$$

where $\{u_i\}$ and $\{v_i\}$ are orthonormal sequences in $\mathbb{R}^m$ and $\mathbb{R}^n$, respectively.

Clearly, if we know $rank(A) = r \leq \min\{m, n\}$, then

$$A = \sum_{i=1}^{r} \sigma_i(A) u_i v_i^T.$$

Note that the singular value decomposition is a general case for the eigenvalue decomposition of a symmetric matrix $A_{m\times m} = A^T_{m\times m}$, which can be written as follows

$$A = \sum_{i=1}^{m} \lambda_i(A) u_i u_i^T,$$

where $\{u_i\}_{i=1}^m$ is the orthonormal eigenbasis of the matrix A.

Theorem 2.4. *Let $U\Sigma V^T$ be the singular value decomposition of matrix $A_{m\times n}$ as in*

Theorem 2.3. *Then*

$$A^\dagger = V\Sigma^\dagger U^T,$$

where

$$\Sigma^\dagger = \begin{pmatrix} \sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m\times n}.$$

One can rewrite the above expression as

$$A^\dagger = \sum_{i=1}^{r} \frac{1}{\sigma_i} v_i u_i.$$

For more detailed discussions and examples about SVD, see [124, 138].

As a corollary of the above notes, if $A = U\Sigma V^T$ is a SVD of the matrix A, then clearly

$$Rank(A) = Rank(\Sigma).$$

The matrix Σ is a diagonal matrix with its diagonal components equal to the singular values of the matrix A, i.e,

$$\Sigma = diag(\sigma_1, ..., \sigma_n).$$

Therefore, we have

$$Rank(\Sigma) = Rank\ diag(\sigma_1, ..., \sigma_n).$$

By setting $\sigma = (\sigma_1, ..., \sigma_n)^T$, the rank of the matrix A is reduced to the cardinality of the vector of singular values, i.e,

$$Rank(A) = Card(\sigma).$$

Obviously, $rank(A)$ is equal to the number of non-zero singular values. So, by using SVD theorem, one can reduce a rank minimization problem(RMP) to a cardinality minimization problem(CMP).

In [60], it was proved that $\|X\|_*$ is the convex envelop of $rank(X)$, i.e, the convex relaxation of the RMP

$$\begin{aligned} &\underset{X}{\text{Minimize}}\ Rank(X) \\ &\text{s.t. } X \in C, \end{aligned} \tag{2.16}$$

(where C is a convex set) is

$$\begin{aligned} &\underset{X}{\text{Minimize}}\ \|X\|_* \\ &\text{s.t. } X \in C. \end{aligned} \tag{2.17}$$

Nuclear norm of a matrix $X_{m\times n}$ is defined as the summation over the singular values of the matrix, i.e,

$$\|X\|_* = \sum_{i=1}^{\min\{m,n\}} \sigma_i(X).$$

Note that the problem (2.17) can be reformulated as follows:

$$\begin{aligned} \underset{t,X}{\text{Minimize}} &\quad t \\ \text{s.t.} &\quad \|X\|_* \le t \\ &\quad X \in C. \end{aligned}$$

Based on [12, 135], one can say $\|X\|_* \le r$ is SDr. The following proposition is the general case for this.

Proposition 2.2. *[12] The sum of the β largest singular values of a matrix $X_{m\times n}$, $m \le n$*

denoted with Σ_β is SDr. Specially the operator norm of a matrix is SDr, i.e,

$$|X| \leq r \Leftrightarrow \begin{pmatrix} rI_n & -X^T \\ -X & rI_m \end{pmatrix} \succeq 0.$$

Note that the operator norm for a linear map $X : V \to W$ is defined as follows:

$$|X| = sup \left\{ \frac{\|Xv\|}{\|v\|} : v \in V, v \neq 0 \right\}$$

In the next chapter, we start discussing general cardinality constrained problems and cardinality minimization problems under non-convex quadratic constraints and linear constraints.

Chapter 3 Cardinality Minimization Problem(CMP)[1]

.

The general cardinality minimization problem(CMP) over a convex set C, and cardinality constrained problem can be cast respectively as:

$$\underset{x}{\text{Minimize}}\ \{Card(x):\ x \in C\}, \tag{3.1}$$

and

$$\underset{x}{\text{Minimize}}\ \{f(x):\ Card(x) \leq \tau,\quad x \in C\}. \tag{3.2}$$

In this thesis, we also consider CMPs over nonconvex sets. CMP is to maximize the number of zero components or equivalently to minimize the number of non-zero components of a vector satisfying certain constraints. In other words, CMP is looking for the sparsest vector in a given feasible set or looking for the simplest model for describing or fitting a certain phenomena. The card function, $card(x)$, can be expressed as l_0-norm. While l_0 is

[1]Most parts of this chapter can be found in [3]

not a norm, we can still call it l_0-'norm', due to the following fact:

$$\|x\|_0 = \lim_{p\to 0} \|x\|_p^p = \lim_{p\to 0} \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} = Card(x).$$

The l_0-norm is a non-convex, non-smooth and integer valued function, and the optimization problems with 'card' objective or constraints are known as NP-hard problems [87, 135], and thus CMPs are not computationally tractable in general.

These kinds of problems have many applications in such areas as finance [97, 35, 126], signal processing and control [133, 72, 93], statistics and principal component analysis [47, 141, 154, 101] , compressive sensing [11, 50, 23], etc. Due to the NP-hardness of CMP, the aim of this chapter is to introduce different SDP relaxations/approximations of CMPs.

This chapter is organized as follows. In section 3.1, we consider cardinality constrained problems, and discuss SDP relaxation methods for these problems based on Shor's lemma and duality methods. Also, we show how this problem can be cast as a bilevel optimization problem. In section 3.2, we review various existing methods for solving CMPs under linear constraints, and as an example of reweighted l_1-techniques, we introduce a continuous approximation of the cardinality function and then apply linearization methods to solve the problem iteratively (we focus on reweighted l_j-algorithms ($j \geq 1$) in the chapters 4 and 5). In section 3.3, we study CMPs under nonlinear non-convex constraints, and show how to find an approximate solution to these problems using reformulation techniques and Lagrangian duality methods. We also explain how the dual problem can be reduced to a semidefinite problem. In section 3.4, we discuss CMP under 0-1 vectors, and explain how to reformulate these problems by adding certain penalty instead of dropping the rank constraint. In section 3.5, we discuss the subgradient method to solve the l_1-minimization problem, and in section 3.6, through numerical experiments, we show the performance of this method. In section 3.7, we explain the branch and bound algorithm to solve CMPs and in section 3.8, we demonstrate the performance of this method through the numerical experiments.

3.1 Cardinality constrained problems

Let us first start with a general cardinality constrained problem. A general cardinality constrained problem is of the form (3.2) where $f(x)$ and C are convex. $Card(x) \leq \tau$ is not a convex constraint, so we try to relax this constraint using semidefinite relaxation. Before doing so, we first note that norms are equivalent in finite dimensional spaces in the following sense: *Suppose* $\|\cdot\|_\mu$*,* $\|\cdot\|_\nu$ *are norms on* R^n*. Then there exist scalars* $a, b \geq 0$*, such that* $a\|x\|_\mu \leq \|x\|_\nu \leq b\|x\|_\mu, \quad \forall x \in R^n$. For example, we have

$$\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\|x\|_2.$$

To be more precise note that in general case for a polytope norm defined by $\max |a_i^T x|,\ i = 1, ..., n$, we have

$$\frac{1}{\sqrt{\beta}}\sqrt{x^T A x} \leq \max_{i=1,\dots,n} |a_i^T x| \leq \sqrt{\beta}\sqrt{x^T A x},\ \forall A \succcurlyeq 0,\ x \in R^n,$$

where β is a constant.

Proposition 3.1. *[47] A convex relaxation of the nonconvex cardinality constraint,* $Card(x) \leq \tau$*, can be expressed via the following convex constraints*

$$\boldsymbol{1}^T|X|\boldsymbol{1} \leq \tau tr(X), \quad \begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0, \quad \textit{for some symmetric matrices} \ \ X,$$

where $|X|$ *denotes the element-wise absolute value of the matrix* X*.*

Proof. For any given vector $x = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix}^T \neq 0$, obviously we have

$$0 \leq \frac{|x_i|}{\sqrt{x_1^2 + ... + x_n^2}} \leq 1, \quad i = 1, ..., n,$$

and hence

$$\sum_{i=1}^{n} \frac{|x_i|}{\sqrt{x_1^2 + ... + x_n^2}} \leq Card(x) \leq \tau,$$

i.e.,

$$\frac{\sum_{i=1}^{n} |x_i|}{\sqrt{x_1^2 + ... + x_n^2}} = \frac{\|x\|_1}{\|x\|_2} \leq Card(x) \leq \tau. \tag{3.3}$$

Note that the cardinality of the vector $x = (x_1, ..., x_n)$ is equal to that of the vector $|x| = (|x_1|, ..., |x_n|)$. So by the fact that $Card(|x|) = Card(x) \leq \tau$, we define the vector $\psi = (\psi_1, ...\psi_n)$, where for every i, $\psi_i = 1$ if $x_i \neq 0$; otherwise $\psi_i = 0$. By Cauchy-Schwartz inequality, i.e, $|\langle |x|, \psi \rangle|^2 \leq \langle |x|, |x| \rangle . \langle \psi, \psi \rangle$, and noting that $Card(x) \leq \tau$, we have

$$||x_1| + ... + |x_n||^2 \leq \tau(|x_1|^2 + ... + |x_n|^2).$$

Therefore we have

$$\|x\|_1^2 \leq \tau \|x\|_2^2. \tag{3.4}$$

In what follows, we use semidefinite relaxation methods [62] to represent the above statement as a convex inequality. Consider the matrix, $X = xx^T$, i.e.,

$$X = \begin{pmatrix} x_1^2 & x_1x_2 & \dots & x_1x_n \\ x_2x_1 & x_2^2 & \dots & x_2x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_nx_1 & x_nx_2 & \dots & x_n^2 \end{pmatrix}.$$

Clearly, the left hand side of the inequality (3.4) is the summation of all of the components of the matrix X, and $\|x\|_2^2$ is the trace of the matrix X. So (3.4) can be written as the following convex inequality (see e.g. [47])

$$\mathbf{1}^T|X|\mathbf{1} \leq \tau tr(X),$$

where $|X|$ denotes the element-wise absolute value of the matrix X. While the constraint $X = xx^T$ is not convex, by applying Shor's lemma, it can be written as $X \succeq xx^T$, $Rank(X) = 1$. $Rank(X) = 1$ is a nonconvex constraint, so by dropping the rank constraint, one may achieve the following convex relaxation for the constraint $X = xx^T$,

$$\begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0,$$

and this completes the proof. $\square$

In [149], Zhao proved that under certain conditions, matrix rank minimization can be formulated as a linear bilevel optimization problem. This motivates the following result.

Proposition 3.2. *If the set C is bounded and defined by linear constraints, the cardinality constrained problem (3.2) can be written as a bilevel optimization problem.*

Proof. From the proof of Proposition 3.1, one can rewrite the problem (3.2) as of the form:

$$
\begin{aligned}
&\underset{x,X}{\text{Minimize}} && f(x) \\
&\text{s.t.} && x \in C \\
& && \mathbf{1}^T|X|\mathbf{1} \leq \tau tr(X) \\
& && X = xx^T.
\end{aligned}
\tag{3.5}
$$

Now by applying Shor's lemma, we can write the above problem as the following form:

$$
\begin{aligned}
&\underset{x,X}{\text{Minimize}} && f(x) \\
&\text{s.t.} && x \in C \\
& && \mathbf{1}^T|X|\mathbf{1} \leq \tau tr(X) \\
& && \begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0 \\
& && Rank(X) = 1.
\end{aligned}
\tag{3.6}
$$

The nonconvex constraint $Rank(X) = 1$ motivates to look for the low rank matrices X. So, instead of dropping the rank constraint, in our optimization problem, one should also minimizes the rank of the matrix X. Since X is a symmetric matrix, $rank(X)$ may be replaced by its convex envelop $tr(X)$. Hence, the problem (3.6) will be equivalent to the following bilevel optimization form (see [149]):

$$
\begin{aligned}
&\underset{x,X,\hat{X}}{\text{Minimize}} && f(x)\\
&\text{s.t.} && x\in C\\
&&& \mathbf{1}^T|\hat{X}|\mathbf{1}\le \tau tr(\hat{X})\\
&&& \hat{X}=\arg\min_{X\in\mathbf{S}^n}\left\{tr(X):\begin{pmatrix} X & x\\ x^T & 1\end{pmatrix}\succeq 0\right\}.
\end{aligned}
\tag{3.7}
$$

The proof is complete. □

The constraint $Rank(X) = 1$ is not convex. In order to get a reasonable approximation/relaxization of (3.2), a simple idea is to drop this constraint. This leads to the following problem:

$$
\begin{aligned}
&\underset{x,X}{\text{Minimize}} && f(x)\\
&\text{s.t.} && x\in C\\
&&& \mathbf{1}^T|X|\mathbf{1}\le \tau tr(X)\\
&&& \begin{pmatrix} X & x\\ x^T & 1\end{pmatrix}\succeq 0,
\end{aligned}
\tag{3.8}
$$

which can be solved more efficiently than the original problem. Dropping the rank constraint, however, may result in a large gap between the optimal values of the relaxed problem (3.8) and the original problem. In [60], it is proved that the convex envelop of $Rank(X)$ is the nuclear norm function ($\|X\|_*$), which is defined as the summation over all of the singular values of the matrix X. Therefore to achieve a convex problem, we can use the penalty method instead of dropping the rank constraint to obtain better approximation of the original problem. This yields the following convex problem (semidefinite problem):

$$
\begin{aligned}
&\underset{x,X}{\text{Minimize}} && f(x) + \xi \|X\|_* \\
&\text{s.t.} && x \in C \\
&&& \mathbf{1}^T |X| \mathbf{1} \leq \tau tr(X) \\
&&& \begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0
\end{aligned} \tag{3.9}
$$

where $\xi > 0$ is the penalty parameter which is chosen to be sufficiently large.

A special case of the problem (3.2) can be cast as

$$
\begin{aligned}
&\underset{x}{\text{Minimize}} && f(x) = \frac{1}{2} x^T P x + q^T x \\
&\text{s.t.} && Ax \leq b \\
&&& Card(x) \leq \tau \\
&&& 0 \leq x_i \leq s_i, \quad i = 1, ..., n,
\end{aligned} \tag{3.10}
$$

where P is an $n \times n$ symmetric matrix, $q \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $\tau \in \mathbb{N}$. This problem was studied by Zheng, Sun and Li [153].

As we have seen above, a common way to solve the optimization problems with a cardinality function as an objective or constraint is to relax the cardinality function. We take the specific example above to demonstrate this approach further. First, the problem (3.10) can be reformulated as the following mixed integer quadratic problem:

$$\begin{aligned} &\underset{x,u}{\text{Minimize}} && f(x) = \frac{1}{2}x^T P x + q^T x \\ &\text{s.t.} && Ax \leq b, \\ (3.11) \qquad & && \mathbf{1}^T u \leq \tau, \ u \in \{0,1\}^n, \\ & && 0 \leq x_i \leq s_i u_i, \quad i = 1, ..., n, \end{aligned}$$

where $\mathbf{1}$ still denotes the vector of ones. Note that the constraint $u_i \in \{0, 1\}$ can be written as $u_i^2 - u_i = 0$. Assuming $P \succeq 0$, the convex relaxation of the problem above can be achieved by replacing $u_i \in \{0, 1\}$ by $u_i \in [0, 1]$. So, it leads to the following problem (see [153]):

$$\begin{aligned} &\underset{x,u}{\text{Minimize}} && f(x) = \frac{1}{2}x^T P x + q^T x \\ (3.12) \qquad &\text{s.t.} && Ax \leq b \\ & && \mathbf{1}^T u \leq \tau, \ u \in [0,1]^n \\ & && 0 \leq x_i \leq s_i u_i, \quad i = 1, ..., n. \end{aligned}$$

Note that the constraint $u_i \in [0, 1]$ can be written as $u_i^2 - u_i \leq 0$. Obviously, the optimal value of the problem (3.12) is a lower bound for the problem (3.10). An SDP relaxation for the problem (3.10) can be obtained as follows: Let $X = xx^T$, and $U = uu^T$, which can be relaxed to $X \succeq xx^T$ and $U \succeq uu^T$. This yields the following relaxed problem:

$$
\begin{aligned}
\underset{x,u,X,U}{\text{Minimize}} \quad & f(x) = \frac{1}{2}x^T P x + q^T x \\
\text{s.t.} \quad & Ax \le b \\
& \mathbf{1}^T u \le \tau, u_i^2 - u_i \le 0, \ i = 1, ..., n \\
& 0 \le x_i \le s_i u_i, \quad i = 1, ..., n \\
& \begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0, \quad \begin{pmatrix} U & u \\ u^T & 1 \end{pmatrix} \succeq 0.
\end{aligned}
\tag{3.13}
$$

Relationship between (3.12) and (3.13) was characterized by the following result.

Proposition 3.3. *[153]* Suppose that the feasible set of the problem (3.12) has an interior point (or a relative interior point, if $Ax \le b$ includes equality constraint). If $P \succeq 0$, then the optimal value of the problems (3.13) and (3.12) are equal.

In the next section, we explain cardinality minimization problem under linear constraints briefly, and we will discuss these problem with full details, in chapters 4 and 5.

3.2 CMP under linear constraints

The Cardinality minimization problem(CMP) with linear constraints, i.e.,

$$
\underset{x}{\text{Minimize}} \ \{Card(x) : \ Ax = b\}, \tag{3.14}
$$

where $A \in R^{m \times n}$ is a matrix with $m < n$, has been widely discussed in the field of compressive sensing [33, 92, 130], which deals with the signal processing/recovery together with applications in such areas as image processing [91].

The most popular approach for solving (3.14) (which is NP hard in general) is to replace the function $card(x)$ by its convex envelop the $\|x\|_1$-function, we will give a proof of this fact in chapter 4. Hence a relaxation of (3.14) is as follows:

$$\underset{x}{\text{Minimize}}\ \{\|x\|_1 :\ Ax = b\}, \tag{3.15}$$

which will be discussed in chapter 4 with more details.

Another effective method for solving the problem (3.14) is to apply reweighted l_1-techniques (see e.g. [34]). The reweighted l_p-minimization ($p \geq 1$) will be discussed in chapter 4 and 5, where we introduce new weights and functions for sparsity. For now, as an example let us define the following continuous approximation of $card(x)$:

$$Card(x) = \|x\|_0 = \lim_{\epsilon \to 0} \sum_{i=1}^{n} sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right). \tag{3.16}$$

Hence for a given small $\epsilon > 0$, an approximation counterpart of (3.14) is given as follows:

$$\begin{align} &\underset{x}{\text{Minimize}} \quad \sum_{i=1}^{n} sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right) \tag{3.17}\\ &\text{s.t.} \quad Ax = b. \tag{3.18}\end{align}$$

Note that

$$\begin{aligned} sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right) &\leq sin\left(atan\left(\frac{|y_i|}{\epsilon}\right)\right) + \frac{1}{y_i^2+\epsilon^2} cos\left(atan\left(\frac{|y_i|}{\epsilon}\right)\right)(|x_i| - |y_i|) \\ &\leq sin\left(atan\left(\frac{|y_i|}{\epsilon}\right)\right) + \frac{1}{y_i^2+\epsilon^2}(|x_i| - |y_i|), \quad \forall\ x, y. \end{aligned}$$

Using linearization techniques(majorization minimization), one obtain the following iterative scheme:

$$x^{l+1} = \arg\min_x \left\{ \sum_{i=1}^{n} \frac{|x_i|}{\left(x_i^l\right)^2 + \epsilon^2} : Ax = b \right\}, \tag{3.19}$$

where l is the number of the iteration, and $\frac{1}{\left(x_i^l\right)^2+\epsilon^2}$ can be interpreted as the weight which forces the nonzero component to be zero if possible. The initial point x^0 can be chosen as the optimal solution of the l_1-minimization (chapter 5).

Before closing this section, it is worth mentioning that sometimes we are interested in finding a solution with a prescribed cardinality t. Such problems can be written as the following feasibility problem:

$$\begin{aligned} \text{Find} \quad & x \\ \text{s.t.} \quad & Ax = b \\ & card(x) \le t. \end{aligned} \tag{3.20}$$

which can be reformulated as a d.c. programming. In fact, for $x \in R^n$, the problem above is equivalent to the minimization of $(n-t)$ smallest components of x.

Now suppose $S_t(x)$ is defined as the summation over the t largest components of the vector $|x|$ (assume that $|x_1| \ge |x_2| \ge ... \ge |x_n|$)

$$S_t(x) = \sum_{i=1}^{t} |x_i|,$$

which clearly is a convex function. Hence the problem (3.20) can be reformulated as

$$\operatorname*{Minimize}_x \{\|x\|_1 - S_t(x) : \ Ax = b\}, \tag{3.21}$$

which is a d.c. programming problem, since both of the functions $\|x\|_1$ and $S_t(x)$ are convex. This problem can be solved by the cutting plane method, which is a usual approach for solving d.c. problems. However, linearization method can be still used to obtain an approximate solution to the problem. The linearized version of the problem can be written as

$$\underset{x}{\text{Minimize}} \; \{\nabla(\|x\|_1 - S_t(x))^T x : \; Ax = b\}.$$

This is equivalent to

$$\begin{aligned} \underset{x,g,u}{\text{Minimize}} \quad & (sign(x) - g)^T x \\ \text{s.t.} \quad & Ax = b, \\ & g = \text{Maximize } u^T x \\ & \text{s.t. } u \in [0,1] \\ & \mathbf{1}^T u = t, \end{aligned} \tag{3.22}$$

which can be viewed as a special linear bilevel programming problem.

3.3 CMP with nonlinear non-convex constraints

In this section, we discuss the CMP with quadratic constraints, i.e., C in (3.1) is of the form

$$C = \{x : b_i x_i^2 - a_i x_i - c_i \leq 0\}, \; i = 1, ..., n.$$

We assume that the constraint functions are not necessarily convex, i.e., b_i is not necessarily positive. In this section, we discuss some approaches for the relaxation and/or reformulation of such problems.

By adding a boolean valued slack variable $v = (v_1, ..., v_n)^T$ to the problem, the CMP

(3.1), with non-convex quadratic constraints, can be reformulated as:

$$
\begin{aligned}
& & \underset{v,x}{\text{Maximize}} \quad & \sum_{i=1}^{n} v_i = \mathbf{1}^T v \\
&(3.23) & \text{s.t.} \quad & v_i x_i = 0 \\
& & & v_i \in \{0,1\},\ i = 1,2,...,n \\
& & & b_i x_i^2 - a_i x_i - c_i \leq 0,\ i = 1,2,...,n,
\end{aligned}
$$

A similar reformulation can be found in [41]. We now give a dual formulation of this problem.

Proposition 3.4. The dual SDP form of the problem above can be written as the following SDP problem

$$\min_{\gamma,\lambda,\mu,\beta} (\gamma : \begin{pmatrix} c(\lambda,\mu,\beta)+\gamma & b(\lambda,\mu,\beta)^T \\ b(\lambda,\mu,\beta) & A(\lambda,\mu,\beta) \end{pmatrix} \succeq 0),$$

where $A(\lambda,\mu,\beta),\ c(\lambda,\mu,\beta),\ b(\lambda,\mu,\beta)$ *are defined in* (3.25), (3.26).

Proof. We make some small changes to the objective of (3.23) and rewrite the problem as follows:

$$
\begin{aligned}
& & \underset{v,x}{\text{Maximize}} \quad & \begin{pmatrix} \mathbf{1} \\ \mathbf{0} \end{pmatrix}^T \begin{pmatrix} v \\ x \end{pmatrix} \\
&(3.24) & \text{s.t.} \quad & v_i x_i = 0 \\
& & & v_i \in \{0,1\},\ i = 1,2,...,n \\
& & & b_i x_i^2 - a_i x_i - c_i \leq 0,\ i = 1,2,...,n,
\end{aligned}
$$

where $\mathbf{1} \in \mathbb{R}^n$ is a column vector with all of its components are equal to one, and $\mathbf{0} \in \mathbb{R}^n$ is a column vector with all of its components are equal to zero. The condition $v_i \in \{0, 1\}$ can be relaxed with $v_i^2 - v_i \leq 0$, which is a convex constraint. Now we may produce the following relaxed problem:

$$\begin{aligned} \underset{x,v}{\text{Minimize}} \quad & -\begin{pmatrix} \mathbf{1} \\ \mathbf{0} \end{pmatrix}^T \begin{pmatrix} v \\ x \end{pmatrix} \\ \text{s.t.} \quad & v_i x_i = 0 \\ & v_i^2 - v_i \leq 0, \ i = 1, 2, ...n \\ & b_i x_i^2 - a_i x_i - c_i \leq 0, \ i = 1, 2, ..., n, \end{aligned}$$

where

$$\begin{pmatrix} v \\ x \end{pmatrix} = (v_1, ..., v_n, x_1, ..., x_n)^T, \quad (\mathbf{1}, \mathbf{0}) = \left(\underbrace{1...1}_{n-\text{times}}, \underbrace{0...0}_{n-\text{times}} \right).$$

Applying Lagrange duality and adding some weight vectors μ, λ, β yields

$$L_{v,x}(\mu, \lambda, \beta) =$$

$$\inf_{(v,x)^T \in R^{2n}} \left(-\begin{pmatrix} \mathbf{1} \\ \mathbf{0} \end{pmatrix}^T \begin{pmatrix} v \\ x \end{pmatrix} + \sum_{i=1}^{n} \mu_i (v_i^2 - v_i) + \sum_{i=1}^{n} \lambda_i v_i x_i + \sum_{i=1}^{n} \beta_i (b_i x_i^2 - a_i x_i - c_i) \right).$$

Note that

$$
\begin{aligned}
&L_{v,x}(\mu,\lambda,\beta)\\
&= \begin{pmatrix} v_1 \\ \vdots \\ v_n \\ x_1 \\ \vdots \\ x_n \end{pmatrix}^T
\begin{pmatrix}
\mu_1 & 0 & 0 & \lambda_1 & 0 & 0 \\
0 & \ddots & 0 & 0 & \ddots & 0 \\
0 & 0 & \mu_n & 0 & 0 & \lambda_n \\
\lambda_1 & 0 & 0 & \beta_1 b_1 & 0 & 0 \\
0 & \ddots & 0 & 0 & \ddots & 0 \\
0 & 0 & \lambda_n & 0 & 0 & \beta_n b_n
\end{pmatrix}_{2n\times 2n}
\begin{pmatrix} v_1 \\ \vdots \\ v_n \\ x_1 \\ \vdots \\ x_n \end{pmatrix} + \\
&\begin{pmatrix}
-1-\mu_1 & 0 & 0 & 0 & 0 & 0 \\
0 & \ddots & 0 & 0 & 0 & 0 \\
0 & 0 & -1-\mu_n & 0 & 0 & 0 \\
0 & 0 & 0 & \beta_1 a_1 & 0 & 0 \\
0 & 0 & 0 & 0 & \ddots & 0 \\
0 & 0 & 0 & 0 & 0 & \beta_n a_n
\end{pmatrix}
\begin{pmatrix} v_1 \\ \vdots \\ v_n \\ x_1 \\ \vdots \\ x_n \end{pmatrix} +
\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \beta_1 \\ \vdots \\ \beta_n \end{pmatrix}^T
\begin{pmatrix} 0 \\ \vdots \\ 0 \\ -c_1 \\ \vdots \\ -c_n \end{pmatrix}.
\end{aligned}
$$

Setting

(3.25)

$$
A(\lambda,\mu,\beta) = \begin{pmatrix}
\mu_1 & 0 & 0 & \lambda_1 & 0 & 0 \\
0 & \ddots & 0 & 0 & \ddots & 0 \\
0 & 0 & \mu_n & 0 & 0 & \lambda_n \\
\lambda_1 & 0 & 0 & \beta_1 b_1 & 0 & 0 \\
0 & \ddots & 0 & 0 & \ddots & 0 \\
0 & 0 & \lambda_n & 0 & 0 & \beta_n b_n
\end{pmatrix}, \quad
c(\lambda,\mu,\beta) = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \beta_1 \\ \vdots \\ \beta_n \end{pmatrix}^T
\begin{pmatrix} 0 \\ \vdots \\ 0 \\ -c_1 \\ \vdots \\ -c_n \end{pmatrix},
$$

$$b(\lambda,\mu.\beta)=\frac{1}{2}\begin{pmatrix} -1-\mu_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & 0 & -1-\mu_n & 0 & 0 & 0 \\ 0 & 0 & 0 & \beta_1 a_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & \beta_n a_n \end{pmatrix}, \tag{3.26}$$

and introducing a new variable y for (v, x), the function $L_{v,x}(\mu,\lambda,\beta)$ can be written as

$$L_y(\lambda,\mu,\beta)=y^T A(\lambda,\mu,\beta)y+2b(\lambda,\mu,\beta)^T y+c(\lambda,\mu,\beta).$$

Assume that λ,μ,β and θ are chosen such that

$$L_y(\lambda,\mu,\beta)-\theta\geq 0,\quad \forall y\in R^{2n}.$$

Then θ is an upper bound for the optimal value of (3.23). Also from chapter 3 of [12], we have

$$g(y)=y^T Ay+2b^T y+c-\theta\geq 0\Leftrightarrow G(y,t)=y^T Ay+2bt^T y+(c-\theta)t^2\geq 0.$$

So,

$$G(y,t)\geq 0\Leftrightarrow\begin{pmatrix} c-\theta & b^T \\ b & A\end{pmatrix}\succeq 0.$$

Then looking for the best upper bound for the main problem above becomes

$$\max_{\theta,\lambda,\mu,\beta} \left\{ \theta : \begin{pmatrix} c-\theta & b^T \\ b & A \end{pmatrix} \succeq 0 \right\}.$$

Setting $\theta = -\gamma$ yields a relaxation for the original problem

$$\min_{\gamma,\lambda,\mu,\beta} \left\{ \gamma : \begin{pmatrix} c+\gamma & b^T \\ b & A \end{pmatrix} \succeq 0 \right\},$$

which is an SDP and can be solved efficiently. $\square$

3.4 CMP with 0-1 variables

In some situations, we are interested in minimizing the cardinality of a boolean vector $x \in R^n$, i.e $x_i \in \{0,1\},\ i = 1, ..., n.$ So, we may consider the CMP with 0-1 variables and quadratic constraints:

$$\begin{aligned} &\underset{x}{\text{Minimize}} && Card(x) \\ &\text{s.t.} && x^T B_i x - A_i x - b_i \leq 0,\ i = 1, ..., m \\ &&& x \in \{0,1\}, \end{aligned} \tag{3.27}$$

where $B_i \succeq 0$, A_i is a vector with appropriate dimension, b_i is a constant. This problem is also discussed in [61] in which the feasible set is defined by a linear system.

Define a new variable $z = 2x - \mathbf{1}$. Hence the problem above can be reformulated as:

$$
\begin{array}{lll}
 & \underset{z}{\text{Minimize}} & Card(z+\mathbf{1}) \\
(3.28) & \text{subject to} & 0.25(z+\mathbf{1})^T B_i(z+\mathbf{1}) - 0.5(A_i(z+\mathbf{1})) - b_i \leq 0,\ \ i=1,...,m \\
 & & z \in \{-1,1\}.
\end{array}
$$

Now let $Z = zz^T$. By shor's lemma, $Z = zz^T$ is equivalent to $Z \succeq zz^T$ and $rank(Z) = 1$. Also note that $tr(Z) = n$, since all the elements on the main diagonal of the matrix Z are equal to one. Hence the problem (3.28) is equivalent to the following problem:

$$
\begin{array}{lll}
 & \underset{z,Z}{\text{Minimize}} & \mathbf{1}^T z \\
 & \text{s.t.} & 0.25(z+\mathbf{1})^T B_i(z+\mathbf{1}) - 0.5(A_i(z+\mathbf{1})) - b_i \leq 0,\ \ i=1,...,m \\
(3.29) & & Z \succeq zz^T,\ \ Z \succeq 0 \\
 & & Tr(Z) = n,\ \ Rank(Z) = 1.
\end{array}
$$

Since $z \in \{-1,1\}$, we can remove the constraint $tr(Z) = n$ and replace it by $diag(Z) = \mathbf{1}$.

To relax the constraint $Rank(Z) = 1$, one can replace the *rank* function by the nuclear norm which is certain convex relaxation of the rank function. So, we get the following SDP:

$$
\begin{array}{lll}
 & \underset{z,Z}{\text{Minimize}} & \mathbf{1}^T z \\
(3.30) & \text{s.t.} & 0.25(z+\mathbf{1})^T B_i(z+\mathbf{1}) - 0.5(A_i(z+\mathbf{1})) - b_i \leq 0,\ \ i=1,...,m \\
 & & Z \succeq zz^T,\ \ Z \succeq 0 \\
 & & Diag(Z) = \mathbf{1}, \|Z\|_* \leq \gamma,
\end{array}
$$

where γ is a constant depends on the bound of $\|Z\|_*$.

As we mentioned in the previous sections, another simple approach to construct an approximation of the problem (3.29) is using penalty function as proposed by Zhao [149]. So, we obtain the following approximation counterpart of (3.29):

$$\begin{array}{ll} \underset{z,Z}{\text{Minimize}} & \mathbf{1}^T z + \xi\|Z\|_* \\ \text{s.t.} & 0.25(z+\mathbf{1})^T B_i(z+\mathbf{1}) - 0.5(A_i(z+\mathbf{1})) - b_i \leq 0, \; i = 1,...,m \\ & Z \succeq zz^T, \; Z \succeq 0, \; Diag(Z) = \mathbf{1}, \end{array} \tag{3.31}$$

where $\xi > 0$ is the penalty parameter.

In the next sections, we discuss subgradient method and branch and bound method, and we test these methods through some numerical experiments.

3.5 Subgradient method

More detailed discussion about subgradient method can be found in [121, 120, 100]. As we discuss the l_1-norm methods in chapter 4 and 5, it is worth mentioning some of the old methods to solve l_1-minimization problems. l_1-norm is a convex and non-smooth function, so here we give some explanation about subgradient method to solve l_1-minimization problems.

Definition 3.1. *(Subgradient)*

Subgradient of a convex function $f : \mathbb{R}^n \to \mathbb{R}$, at a point $x^{(k)}$, is every vector $\xi^{(k)} = (\xi_1^{(k)}, \xi_2^{(k)}, ..., \xi_n^{(k)})^T$, which satisfies the following inequality,

$$f(x) \geq f(x^{(k)}) + {\xi^{(k)}}^T (x - x^{(k)}). \tag{3.32}$$

Definition 3.2. *(Subdifferential) Subdifferential of a function f, at a point $x^{(k)}$, is the set of all subgradients of f at $x^{(k)}$, denoted by $\partial f(x^{(k)})$.*

Theorem 3.1. *Let f be a convex function with dom$(f) \neq \phi$ and $f(x) \geq -\infty, \quad \forall x \in$ dom(f). Then $x^\star$ is a minimizer of f if and only if $0 \in \partial f(x^\star)$.*

Proof.

$$0 \in \partial f(x^\star) \Leftrightarrow f(x) \geq f(x^\star) + 0(x - x^\star) \ (\forall x \in \mathbb{R}^n) \Leftrightarrow x^\star \text{ minimizes } f.$$

□

Subgradient methods are used for non-differentiable optimization problems. A usual method, which is used to minimize a convex(or even concave) function f is approximating the function f by tangential linearization and then improving the approximation iteratively. This method is so called generalized linear programming(GLP), however for the problems with large number of variables, it might be better to use subgradient algorithm [100]. For a differentiable function, we know the subgradient at an arbitrary point is exactly the gradient at that point, in this case, the subgradient method reduces to the gradient method.

For the unconstrained case, the subgradient method uses the following iterative scheme to minimize a convex non-smooth objective f.

$$x^{(k+1)} = x^{(k)} - \beta_k \xi^{(k)}, \tag{3.33}$$

where β_k is the step size.

Subgradient method is not a descent method, since the negative subgradient direction, $-\beta^{(k)}$, is not necessary a descent direction. So, the iterations above do not necessarily decrease the objective at each step. Hence, in this method we should store the best point which has been found so far, in other words in (3.33) at every iteration k, we have

$$f_*^{(k)} = \min\{f_*^{(k-1)}, f(x^{(k)})\} = \min_{j=1,\ldots,k} f(x^{(j)}), \tag{3.34}$$

where f_* denotes the smallest value of the objective which has been found so far.

For constrained optimization of the form

$$\begin{aligned} \underset{x}{\text{Minimize}} \quad & f(x) \\ \text{s.t.} \quad & x \in C, \end{aligned}$$

one could rewrite the iteration (3.33) as follows

$$x^{(k+1)} = P_C\left(x^{(k)} - \beta_k \xi^{(k)}\right), \tag{3.35}$$

where β_k is the step size. $P_C(x) = argmin_{y \in C}\left(\|x - y\|_2^2\right)$ is the standard projection operator on C. The above method is called the subgradient projection algorithm(see Algorithm 3.1). Now based on [105, 20], we prove the convergence of the subgradient projection algorithm as follows.

First, observe that from (3.32) we have

$$f(x^{(k)}) - f(x^\star) \geq -{\xi^{(k)}}^T (x^\star - x^{(k)}). \tag{3.36}$$

In (3.35) let $u^{(k+1)} = \left(x^{(k)} - \beta_k \xi^{(k)}\right)$. Then

$$\begin{aligned}
\|u^{(k+1)} - x^\star\|_2^2 &= \|x^{(k)} - \beta_k \xi^{(k)} - x^\star\|_2^2 \\
&= \|x^{(k)} - x^\star\|_2^2 + \beta_k^2 \|\xi^{(k)}\|_2^2 - 2\beta_k {\xi^{(k)}}^T \left(x^{(k)} - x^\star\right) \\
&\leq \|x^{(k)} - x^\star\|_2^2 + \beta_k^2 \|\xi^{(k)}\|_2^2 - 2\beta_k \left(f(x^{(k)}) - f(x^\star)\right),
\end{aligned}$$

where the last inequality follows from (3.36). Now by the definition of projection mapping we have

$$\|P_C(u^{(k+1)}) - x^\star\|_2 \leq \|u^{(k+1)} - x^\star\|_2.$$

Note that

$$\|x^{(k+1)} - x^\star\|_2 = \|P_C(u^{(k+1)}) - x^\star\|_2.$$

Therefore,

$$\|x^{(k+1)} - x^\star\|_2 \leq \|u^{(k+1)} - x^\star\|_2,$$

which results in the following inequality,

$$\|x^{(k+1)} - x^\star\|_2^2 \leq \|x^{(k)} - x^\star\|_2^2 + \beta_k^2 \|\xi^{(k)}\|_2^2 - 2\beta_k \left(f(x^{(k)}) - f(x^\star)\right).$$

If we apply the inequality above for $\|x^{(k)} - x^\star\|_2^2$, we have

$$\|x^{(k)} - x^\star\|_2^2 \leq \|x^{(k-1)} - x^\star\|_2^2 + \beta_{k-1}^2 \|\xi^{(k-1)}\|_2^2 - 2\beta_{k-1} \left(f(x^{(k-1)}) - f(x^\star)\right).$$

Repeating this procedure yields

$$0 \leq \|x^{(k+1)} - x^\star\|_2^2 \leq \|x^{(1)} - x^\star\|_2^2 + \sum_{j=1}^{k} \beta_j^2 \|\xi^{(j)}\|_2^2 - 2\sum_{j=1}^{k} \beta_j \left(f(x^{(j)} - f(x^\star)\right).$$

Now assume that the Euclidean distance of the starting point $x^{(1)}$ of the algorithm to the optimal set, $C^{\star}$, is known, and denoted by $d(x^{(1)}, C^{\star})$. We have

$$d(x^{(1)}, C^{\star})^2 + \sum_{j=1}^{k} \beta_j^2 \|\xi^{(j)}\|_2^2 \geq 2 \sum_{j=1}^{k} \beta_j \left(f(x^{(j)} - f(x^{\star}) \right).$$

In addition, from (3.34), we see that

$$2 \sum_{j=1}^{k} \beta_j \left(f(x^{(j)} - f(x^{\star}) \right) \geq 2 \left(\sum_{j=1}^{k} \beta_j \right) \left(f_*^{(k)} - f(x^{\star}) \right),$$

which implies that

$$\left(f_*^{(k)} - f(x^{\star}) \right) \leq \frac{d(x^{(1)}, C^{\star})^2 + \sum_{j=1}^{k} \beta_j^2 \|\xi^{(j)}\|_2^2}{2 \sum_{j=1}^{k} \beta_j}. \tag{3.37}$$

So, the convergence of the subgradient algorithm depends essentially on the way how the step sizes β_k, at each step k, are chosen.

The simplest case is to choose $\beta_k = \mathit{constant}$, or in the case of known or estimated optimal value f^* one can choose the step size as [113],

$$\beta_{(k)} = \frac{f(x^{(k)}) - f^*}{||\xi^{(k)}||_2^2}.$$

Another option, for example, is to choose the step size using the divergent series rules,

$$\beta_k \to 0 (k \to \infty), \quad \textstyle\sum_{k=1}^{\infty} \beta_k = \infty. \tag{3.38}$$

As proved in [112, 59], if the step size is chosen as above then the sequence generated by the constrained subgardient algorithm satisfies

$$\lim\inf{}_{k \to \infty} f(x^{(k)}) = f(x^{\star}) = \min_{x \in C} f(x).$$

As an example about the convergency analysis using the inequality (3.37), see the following example and its proof.

Example 3.1. *If we choose the step size by divergent series rule, assuming* $|\xi^{(j)}\|_2^2$ *is bounded with* $\xi > 0$*, we have*

$$\lim_{k\to\infty}\left(f_*^{(k)} - f(x^\star)\right) \to 0.$$

Proof. Notice that

$$\beta_k \to 0(k \to \infty) \Rightarrow \forall\epsilon \geq 0, \exists M_1 \in \mathbb{N} \text{ s.t. } \forall k > M_1, \beta_k \leq \frac{\epsilon}{\xi}.$$

Since $\sum_{k=1}^{\infty}\beta_k \to \infty$, we have

$$\exists M_2 \in \mathbb{N} \text{ s.t. } \sum_{k=1}^{M_2}\beta_k \geq \frac{1}{\epsilon}\left(d(x^{(1)}, C^*)^2 + \xi\sum_{k=1}^{M_1}\beta_k^2\right).$$

Now for $M \geq \max\{M_1, M_2\}$, we have

$$\frac{d(x^{(1)}, C^\star) + \sum_{k=1}^{M}\beta_k^2\|\xi^{(k)}\|_2^2}{2\sum_{k=1}^{M}\beta_k} \leq \frac{d(x^{(1)}, C^\star)^2 + \xi\sum_{k=1}^{M}\beta_k^2}{2\sum_{k=1}^{M}\beta_k}$$

$$\leq \frac{d(x^{(1)}, C^\star)^2 + \xi\sum_{k=1}^{M_1}\beta_k^2}{2\sum_{k=1}^{M}\beta_k} + \frac{\xi\sum_{k=M_1+1}^{M}\beta_k^2}{2\sum_{k=1}^{M_1}\beta_k + 2\sum_{k=M_1+1}^{M}\beta_k}$$

$$\leq \frac{d(x^{(1)}, C^\star)^2 + \xi\sum_{k=1}^{M_1}\beta_k^2}{\frac{2}{\epsilon}\left(d(x^{(1)}, C^\star)^2 + \xi\sum_{k=1}^{M_1}\beta_k^2\right)} + \frac{\xi\sum_{k=M_1+1}^{M}\frac{\epsilon\beta_k}{\xi}}{2\sum_{k=M_1+1}^{M}\beta_k} = \epsilon.$$

□

For other step sizes the convergency proof is quite the same.

Now we are going to apply subgradient method to the convex envelop of CMP under some convex sets.

Recall the following CMP,

$$\underset{x}{\text{Minimize}}\ Card(x)$$
$$\text{s.t. } x \in C.$$

As we have seen, CMP can be relaxed by replacing the card function with its convex envelop, the l_1-norm function to minimize the summation of absolute values of the vector components. So, CMP can be relaxed as

$$\underset{x}{\text{Minimize}}\ \|x\|_1 = |x_1| + ... + |x_n|$$
$$\text{s.t. } x \in C.$$

Obviously, the l_1-norm function is a convex and non-smooth function, besides we assume that C is a convex, closed and bounded set, so by Weierstrass theorem the existence of the optimal solution is guaranteed.

It is not hard to see that, the problem above is equivalent to

$$\underset{x}{\text{Minimize}}\ f(x) = \max\{\alpha^T x; \alpha_i \in \{-1, 1\}\}$$
$$\text{s.t. } x \in C.$$

Because of the non differentiability of objective function, we use the subgradient direction to decrease the objective in the feasible set. Using the following algorithm one can generate a sequence $\{x^{(k)}\}_{k=0}^{\infty}$ which converges to the optimal solution.

Algorithm 3.1. *1. At step 0, choose an starting point x^0 for the algorithm.*

2. At step k, compute subgradient $\xi^{(k)}$ of the objective at $x^{(k)}$, if $\xi^{(k)} = 0$, then $x^{(k)}$ is *optimal,(as a result of the Theorem* 3.1*, if* $\xi^{(k)} \neq 0$ *then compute*

$$x^{(k+1)} = P_C\left(x^{(k)} - \beta_k \frac{\xi^{(k)}}{||\xi^{(k)}||}\right),$$

where $P_C(x) = x$*, if and only if,* $x \in C$*.*

3. *If the stopping criteria is satisfied, the optimum is achieved.*

To compute the subgradients of the l_1-norm function, the following lemma from chapter 4 of [14] is applied.

Lemma 3.1. *Let* $g(x) = \max_{i=1,2,...,n}(g_i(x))$*, where* g_i*'s are convex and subdifferentiable, and let* $j \in \{i; i = 1, 2, ..., n\}$*, such that* $g_j(x) = g(x)$*. If* $\xi \in \partial g_j(x)$ *then* $\xi \in \partial g(x)$*.*

By using the above lemma, assume there exists $\alpha \in \{-1, 1\}^n$, such that $\alpha^T x = f(x)$, hence, subgradients of the function are $\xi = (\xi_1, \xi_2, ..., \xi_n)$ such that

$$\xi_i = \begin{cases} 1, & x_i > 0 \\ -1, & x_i < 0 \\ \text{[-1,1]}, & x_i = 0. \end{cases}$$

Now, one can apply the subgradient method described above. Suppose that in the above CMP the constraints set C is the set of convex inequity constraints $f_i(x) \leq 0,\ i = 1, ..., n$. We take the subgradients $\xi^{(k)}$ as defined above, if $x^{(k)}$ is a feasible point. If $x^{(k)}$ does not satisfy one (or more than one) of the constraints, (for example, if $f_j(x^{(k)}) \nleq 0$), then we use the subgradient of the violated constraint. If all of the $x^{(k)}$ are infeasible, then the objective function will go to infinity.

In the next section, we present our numerical experiments to solve an l_1-norm minimization problem by applying subgradient method.

3.6 Numerical experiments

Consider the following problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_1 \\ &\qquad \text{s.t.}\ Ax \leq b, \end{aligned}$$

where $A \in \mathbb{R}^{m\times n}(m < n)$. Obviously, one of the subgradients of the $\|x\|_1$-function at x^0 is $sign(x^0)$. Also, the projection operator is

$$P_C(y) = y - A^T(AA^T)^{-1}(Ay - b).$$

So, the subgradient iteration for the problem (3.39) will be

$$x^{(k+1)} = x^{(k)} - \beta_k(I - A^T(AA^T)^{-1}A\ \text{sign}(x^{(k)}).$$

As an example, we consider the problem (3.39), with a randomly generated matrix of $A_{100\times 2000}$ from the normal distribution with mean zero and variance one, i.e, $N(0, 1)$. Let $g^{(k)}$ be the optimal value of the problem (3.39), at each iteration. Choose the step size $\beta_k = \frac{0.1}{k}$. Note that this step size satisfies the divergent series rules. The graph below shows how the subgradient method works for this choice of step size. We used Matlab programming, and the procedure took 25 minutes for 4000 iterations on a laptop with dual core CPU-2GHz.

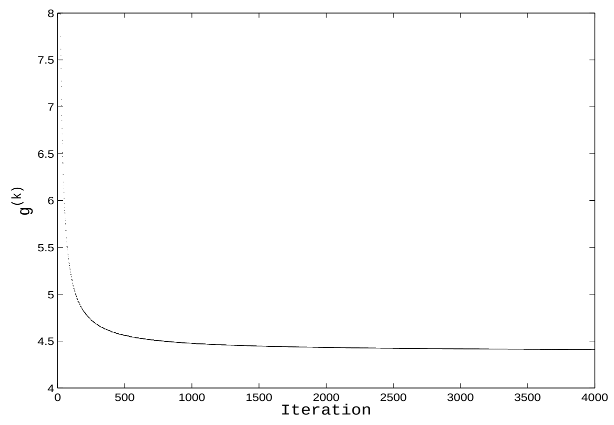

Figure 3.1: The figure shows the value of $g^{(k)}$ at each iteration. The value achieved after 4000 iterations is equal to 4.4095.

The optimal value of the problem (3.39) for our defined matrix A is equal to $g^* = 4.3679$. For more illustration, we present the error graph of $g^{(k)} - g^*$ at each iteration.

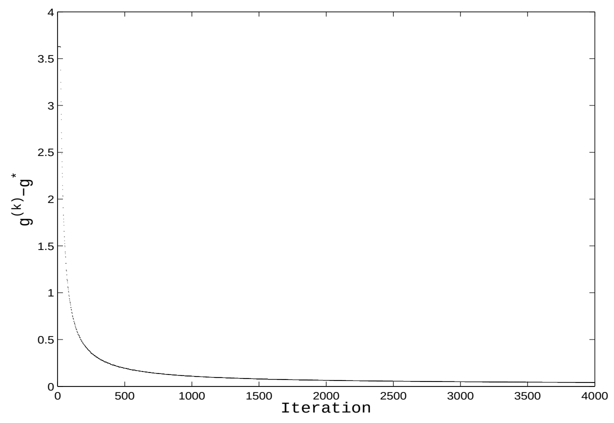

Figure 3.2: The figure shows the error, $(g^{(k)} - g^*)$, at each iteration.

3.7 Branch and bound method

One may refer to [25, 13] for full discussion about the branch and bound method. Branch and bound(BB) algorithm is one of the global optimization methods for solving NP-hard combinatorial problems. BB algorithm searches the whole space of the feasible solutions set, and find an optimal solution to the problem. Therefore it usually needs lots of computational efforts, and the convergence of the algorithm can be very slow.

To introduce the method, suppose that we want to solve the following problem by applying BB method,

$$f^{\star} = \underset{x}{\text{Minimize}}\ f(x)$$
$$\text{s.t. } x \in S.$$

Branch and bound method has three steps: Branch, Bound, and Prune. Branch procedure splits the feasible set S into 2 or more subsets S_1,S_2,...,S_n, and minimize $f(x)$ over those sets one by one. Therefore the main problem can be cast as the set of the following optimization problems

$$\nu_i = \underset{x}{\text{Minimize}}\ f(x)$$
$$\text{s.t. } x \in S_i,$$

where $i = 1, ..., n$.

Then clearly $f^{\star} = \min\{\nu_i : i = 1, ..., n\}$. This algorithm can be represented as a tree graph with its root node as the original feasible set S, and the other nodes as the sets which result from branching procedure. Next step is bounding. This procedure uses two functions, f_l, and f_u, in the following sense,

$$\forall i = 1, ..., n, \quad f_l(S_i) \leq minf(S_i) \leq f_u(S_i).$$

f_l and f_u should be defined in a way that the bounds in the inequality above become tighter. When S_i gets smaller, specially when the size of the set S_i,($Size(S_i)$) goes to a singleton, $f_u(S_i) - f_l(S_i)$ converges to zero. This implies that for enough small α, if $Size(S_i) \leq \alpha$, we have

$$f_u(S_i) - f_l(S_i) \leq \epsilon, \quad \forall \epsilon \geq 0.$$

The efficiency of BB algorithm is strongly related to the splitting procedure, and also to the upper and lower bound defined functions. The main idea in BB algorithm is that if the lower bound for some $S_j \subset S$, (one of the tree nodes) is greater than the upper bound for some others $S_k \subset S$, (some of the other tree nodes) then S_j can be removed from the search, since every point in that subset is worse than the current upper bound on $f^\star$. This step is so called prune step. Removing these kinds of subsets does not affect the algorithm, since they will never be chosen later to split. For more illustrations, we review the algorithm.

Suppose we have already estimated f_l and f_u satisfying the above conditions. For a given sufficiently small ϵ, if $f_u(S) - f_l(S) \leq \epsilon$, then we are done and the algorithm terminates. Otherwise, we split S into two subsets S_1 and S_2, without intersection. Then computing the lower and upper bounds on these sets yields

$$\min\{f_l(S_1), f_l(S_2)\} \leq min f(S) \leq \min\{f_u(S_1), f_u(S_2)\}.$$

Again if the difference between the achieved bounds is less than ϵ, then the algorithm terminates. Otherwise we split the S_i, which satisfies $f_l(S_i) = \min\{f_l(S_1), f_l(S_2)\}$ into two subsets, and we repeat the procedure above for the three subsets achieved so far. Clearly, after k iterations, we have k subsets of S, in a way that $\cup_{i=1}^k S_i = S$. As we mentioned, this procedure can be represented as a binary tree (since we split S_i into two subsets at each iteration).

Hence, after k iterations, we have

$$\min_{i=1,\ldots,k} f_l(S_i) \leq f^\star = \min \ f(x) \leq \min_{i=1,\ldots,k} f_u(S_i).$$

We again check the termination criteria. If it is not satisfied, we proceed to the $k+1$ iteration. At each iteration, one could prune the useless subsets to reduce the storage(memory) requirement.

As for more illustration, we apply the method above to the following CMP:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ Card(x) \\ &\text{s.t. } Ax \preceq b. \end{aligned} \tag{3.39}$$

A simple reformulation of the problem above is as follows [19]:

$$\begin{aligned} v^\star = &\underset{v,x}{\text{Minimize}}\ \mathbf{1}^T v \\ &\text{s.t. } L_i v_i \le x_i \le U_i v_i \\ &\quad Ax \preceq b \\ &\quad v_i \in \{0,1\}, \quad i = 1, ..., n, \end{aligned} \tag{3.40}$$

where $\mathbf{1} = (1...1)^T$

This is a special case of mixed integer linear programming(MILP)[65], called mixed 0-1 linear programming [48], (as we have seen in the previous sections), which can be solved for each of 2^n linear (convex) programs for different values of v_i. So, this method is not practical for large problems.

Obviously, the problem (3.40) can be relaxed by replacing $v_i \in \{0, 1\}$ with $v_i \in [0, 1]$, so, the relaxed problem is

$$
\begin{aligned}
\beta_1 = \underset{v,x}{\text{Minimize}}\ & \mathbf{1}^T v \\
\text{s.t.}\ & L_i v_i \leq x_i \leq U_i v_i \\
& Ax \preceq b \\
& 0 \leq v_i \leq 1 \quad i = 1, ..., n,
\end{aligned}
\tag{3.41}
$$

U_i and L_i can be easily computed by minimizing and maximizing the variable x_i on the feasible set $Ax \preceq b$. Let $L_i \leq 0$ and $U_i \geq 0$ (otherwise the solution is trivial). The problem (3.41) is a convex problem, which can be solved easily. The optimal value of this problem β_1 is a lower bound for (3.40), consequently $\lceil \beta_1 \rceil$ is a better lower bound for (3.39), since $Card(x)$ is an integer. One can also achieve an upper bound, denoted by α_1, for the problem (3.40), using the l_1-norm relaxation. Or by decoding the relaxed problem (3.41). Now using the branch and bound algorithm described above, we are going ro provide an upper and lower bound for the problem (3.40).

If $\alpha_1 - \beta_1 \leq \epsilon$, then the algorithm terminates. Otherwise in (3.41), one can fix the variable v_j, which is equal to 0 or 1 alternatively, and solve the following relaxed problems:

$$
\begin{aligned}
\underset{v,x}{\text{Minimize}}\ & \mathbf{1}^T v \\
\text{s.t.}\ & L_i v_i \leq x_i \leq U_i v_i \\
& Ax \preceq b \\
& 0 \leq v_i \leq 1, \quad i = 1, ..., n \\
& v_j = 1,
\end{aligned}
\tag{3.42}
$$

and

$$
\begin{aligned}
&\underset{v,x}{\text{Minimize}}\ \mathbf{1}^T v\\
&\text{Subject to } L_i v_i \le x_i \le U_i v_i\\
&\qquad Ax \preceq b\\
&\qquad 0 \le v_i \le 1,\quad i = 1, ..., n\\
&\qquad v_j = 0.
\end{aligned}
\tag{3.43}
$$

Solving the convex problem (3.42), and (3.43), results in achieving lower and upper bounds for the following (MILP) branched problems:

$$
\begin{aligned}
&\underset{v,x}{\text{Minimize}}\ \mathbf{1}^T v\\
&\text{Subject to } L_i v_i \le x_i \le U_i v_i\\
&\qquad Ax \preceq b\\
&\qquad v_i \in \{0, 1\},\quad i = 1, ..., n\\
&\qquad v_j = 1,
\end{aligned}
\tag{3.44}
$$

and

$$
\begin{aligned}
&\underset{v,x}{\text{Minimize}}\ \mathbf{1}^T v\\
&\text{Subject to } L_i v_i \le x_i \le U_i v_i\\
&\qquad Ax \preceq b\\
&\qquad v_i \in \{0, 1\},\quad i = 1, ..., n\\
&\qquad v_j = 0.
\end{aligned}
\tag{3.45}
$$

By using the relaxed problems, suppose β' , β'' are the lower bounds and α' , α'' are the upper bounds for the problems (3.44), and (3.45) respectively. Clearly we have

$$\beta_1 \leq \beta_2 = \min\{\beta', \beta''\} \leq v^\star \leq \alpha_2 = \min\{\alpha', \alpha''\} \leq \alpha_1,$$

where $v^\star$ is the optimal value for problem (3.40). Note that since cardinality is an integer valued function one could replace β_2 with $\lceil \beta_2 \rceil$ to get a better(tighter) lower bound. Now we can choose one of the problems (3.44), or (3.45). Its lower bound is equal to $\min\{\beta', \beta''\}$, to branch again by fixing another $v_\gamma = 0$, or 1, $\gamma \neq j$, and continue the same procedure explained above, to get new bounds for $v^\star$ satisfying

$$\beta_1 \leq \beta_2 \leq \beta_3 \leq v^\star \leq \alpha_3 \leq \alpha_2 \leq \alpha_1.$$

By continuing this procedure, the algorithm will be terminated after k iterations if $\alpha_k - \beta_k \leq \epsilon$. At the iteration k, we have an upper bound α_k. Now to prune the search area, if any of the achieved lower bounds for the branched problem is more than α_k, then one could remove the the related area to reduce the search efforts.

The worst case scenario for the BB algorithm is fixing all of the variables $v_i,\ i = 1, ..., n$, and solving 2^n relaxed problems to get an equal upper and lower bound for each problem.

3.8 Numerical experiments

Here we represents a graph for the CMP, with a randomly generated matrix of A from the normal distribution with mean 0 and variance 1, i.e, $N(0, 1)$. $x \in \mathbb{R}^{50}$, and we have 110 constraints. The graph shows the lower and upper bound on $card(x)$ for each iteration.

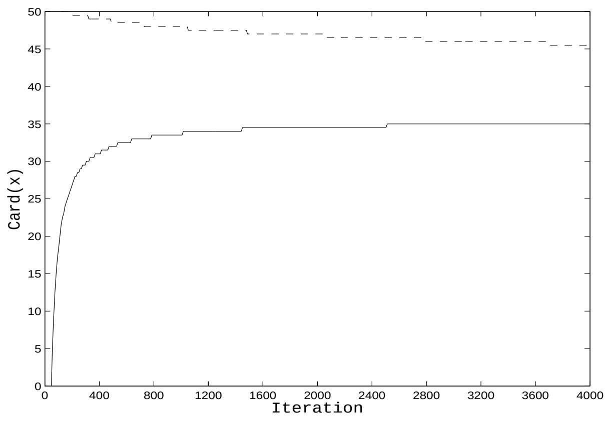

Figure 3.3: $x \in \mathbb{R}^{50}$. The solid line represents the lower bound and the dashed line represents the upper bound on $card(x)$ at each iteration.

Chapter 4

The l_1-Method and Reweighted Algorithms[1]

In this chapter, we discuss cardinality minimization problem(CMP) under some linear constraints denoted by

$$C = \{x; Ax = b, \},\ A \in \mathbb{R}^{m\times n},\ b \in \mathbb{R}^m,$$

where $m < n$, which means we have more variables than equations. Clearly, these systems have infinitely many solutions. The CMP in this case, can be interpreted as finding the sparsest solution to a linear system of equations. In this chapter, we first start with the l_1-minimization method, which is one of the common methods for solving CMPs. Then to motivate the reweighted algorithms, we show that the l_1-minimization uses a kind of weighted l_2-minimization, which is hidden inside the l_1-norm function. In this thesis, a reweighted algorithm in general refers to an iterative weighted algorithm, i.e, the weights are updated in each iteration. Next, we discuss other different convex norm-functions to locate a sparse solution to a linear system, and then we proceed to different reweighted

[1]Special thanks to my supervisor, Dr. Yun-bin Zhao, for his stimulating suggestions for the section 4.2 and 4.4 of this chapter.

l_j-algorithms ($j \geq 1$). The main idea of the reweighted l_j-minimization ($\forall j \geq 1$) is to define the weights based on the current solution of each iteration x^l. The weights should be defined in a way that penalize the small components, and force them to tend to zero. These weights may improve the sparsity measurement property of a merit function. Also, we introduce different merit functions for sparsity, and we show how minimizing such functions may lead us to a sparse solution to a linear system. In addition, we illustrate that different weights, if chosen properly, may reduce the gap between different merit functions. For example, we define some special weights for the reweighted l_2-algorithm and the reweighted l_3-algorithm, and through the numerical experiments, we show how these weights may reduce the gap between the performances of these algorithms and the l_1-minimization algorithm to find a sparse solution to a linear system.

4.1 The l_1-minimization

The l_1-minimization has lots of applications in basis pursuit problems [41, 39, 40], noise deconvolution [128], sparse signal recovery [41, 54], sparse model selection [144, 9]. Some more applications of the l_1-minimization can be found in [150, 67, 127, 136, 8]. Also, a short survey of the l_1-minimization can be found in [34]. The l_1-minimization has been generalized to the nuclear-norm minimization, which is used in matrix rank minimization [60, 61] and low-rank matrix recovery [28, 32].

A usual approach to solve cardinality minimization problems is to relax the problem by the l_1-norm, and to solve the relaxed problem efficiently [20]. The l_1-norm is known as the convex envelop of the function $card(x)$, over $\{x : \|x\|_\infty \leq 1\}$.

Here we give a proof of this fact. Before proving this, let's first define the conjugate function, and review some of the important properties of the conjugate functions.

Definition 4.1. *(Conjugate function)*

The conjugate of a function $f : \mathbb{R}^n \to \mathbb{R}$, denoted by $f^* : \mathbb{R}^n \to \mathbb{R}$, is defined as follows

$$f^*(y) = \sup_{x \in dom(f)} \{y^T x - f(x)\}.$$

Proposition 4.1. *f^* is always a convex function, [20].*

Proof. The proof is straight forward, since the conjugate function is the pointwise supremum of a family of affine functions of y. □

The conjugate of conjugate or the biconjugate of the function f, denoted by f^{**}, is the convex envelop of the function f.

Proof. The conjugate of $card(x)$ on the set $\|x\|_\infty \leq 1$, is defined by

$$f^*(y) = \sup_{\|x\|_\infty \leq 1} \{y^T x - Card(x)\},$$

where $y = (y_1, ..., y_n)^T \in \mathbb{R}^n$.

If $x = 0$, then $f^*(y) = 0$. If $Card(x) = m, \quad 1 \leq m \leq n$, then

$$y^T x - Card(x) \leq \sum_{i=1}^{m} |y_i||x_i| - m \leq \sum_{i=1}^{m} |y_i| - m, \quad \text{over } \{x : \|x\|_\infty \leq 1\},$$

so,

$$f^*(y) = \sup \left\{ 0, |y_1| - 1, ..., \sum_{i=1}^{m} |y_i| - m, ..., \sum_{i=1}^{n} |y_i| - n \right\},$$

and hence if $\|y\|_\infty \leq 1$, then $\sum_{i=1}^{j} |y_i| - j \leq 0, \ \forall j = 1, ..., n$, which means $f^*(y) = 0$.

Now if $\|y\|_\infty \geq 1$, without loss of generality suppose that $|y_i| \geq 1$ for $i = 1, ..., m$, and $|y_i| \leq 1$ for $i = m, m+1, ..., n$. Then $f^*(y) = \sum_{i=1}^{m} |y_i| - m$, and therefore

$$f^*(y) = \begin{cases} 0, & \|y\|_\infty \leq 1, \\ \sum_{i=1}^{m} |y_i| - m, & |y_i| \geq 1, \forall i = 1, ..., m, \quad |y_i| \leq 1, \forall i = m+1, ..., n. \end{cases}$$

Now we compute the conjugate of f^*, i.e.,

$$f^{**}(u) = \sup_u \{u^T y - f^*(y)\}.$$

$$f^{**}(u) = \sup_u \left\{ u^T y - (\sum_{i=1}^{m} |y_i| - m) \right\}. \tag{4.1}$$

If $\|u\|_\infty \geq 1$, there exists at least one i such that $|u_i| \geq 1$, so, one can increase that $|u_i|$ in a way that $f^{**}(u)$ goes to infinity. If $\|u\|_\infty \leq 1$, and $\|y\|_\infty \leq 1$, then $f^*(y) = 0$, and

$$f^{**}(u) = \sum_{i=1}^{n} |u_i| = \|u\|_1.$$

Now if $\|u\|_\infty \leq 1$ and $\|y\|_\infty \geq 1$, then

$$\begin{aligned} u^T y - (\textstyle\sum_{i=1}^{m} |y_i| - m) = & \sum_{i=1}^{n} |u_i||y_i| - (\sum_{i=1}^{n} |y_i| - m) + \sum_{i=1}^{n} |u_i| - \sum_{i=1}^{n} |u_i| \\ & = \underbrace{\sum_{i=1}^{m} (|y_i| - 1)(|u_i| - 1)}_{\leq 0} + \underbrace{\sum_{i=m+1}^{n} (|y_i| - 1)|u_i|}_{\leq 0} + \sum_{i=1}^{n} |u_i| \\ & \leq \sum_{i=1}^{n} |u_i| = \|u\|_1. \end{aligned}$$

hence,

$$f^{**}(u) = \sup_u \{u^T y - f^*(y)\} = \|u\|_1.$$

$\square$

Note that for $x \in \mathbb{R}^n$, the convex envelop of $card(x)$ on $\{x : \|x\|_\infty \leq \tau\}$ is $\frac{1}{\tau}\|x\|_1$.

Hence the following CMP:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_0 = Card(x) \\ &\text{s.t.}\ \ Ax = b, \end{aligned} \tag{4.2}$$

is relaxed by replacing $card(x)$ with the l_1-norm. The relaxed problem is as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_1 = \sum_{i=1}^{n} |x_i| \\ &\text{s.t.}\ \ Ax = b. \end{aligned} \tag{4.3}$$

The l_1-minimization problem can be written as

$$\underset{x}{\text{Minimize}}\ \{s :\ \|x\|_1 \leq s,\ Ax = b\}. \tag{4.4}$$

Clearly, (4.3) and (4.4) are linear programming problems. For instance, (4.4) can be written as the following linear program:

$$\begin{aligned} \underset{s,x}{\text{Minimize}} \quad & \mathbf{1}^T s \\ \text{s.t.} \quad & -s \leq x \leq s \\ & Ax = b. \end{aligned} \tag{4.5}$$

The l_1-norm is the most well known merit function for sparsity. Note that in this chapter, a function is said to be a merit function for sparsity if it promotes sparsity, i.e, minimizing such a function can drive the variable to be sparse.

In the next section, we show that weighted l_2-minimization is hidden inside the l_1-minimization. This gives a motivation to focuss on reweighted algorithms.

4.2 Weighted l_2-minimization is hidden inside the l_1-minimization

In this section, using the Huber function, we prove that weighted l_2-minimization is hidden inside the l_1-minimization. This motivates to focuss on reweighted algorithms.

Proposition 4.2. $\lim_{\delta\to 0}\sum_{i=1}^{n}\phi_\delta(x_i) = \|x\|_1$, *where* $\phi_\delta : R \to R_+$ *is the Huber function. This shows a kind of weighted* l_2*-minimization is already used by* l_1*-minimization.*

Proof. Suppose $x \in \mathbb{R}^n$, and let $\delta \in (0,1)$, and $\phi_\delta : R \to R_+$ be the Huber function, i.e.,

$$\phi_\delta(t) = \begin{cases} |t| - \frac{\delta}{2} & \text{if } |t| \geq \delta \\ \frac{t^2}{2\delta}, & \text{if } |t| < \delta \end{cases}$$

Consider the function $F_\delta : R^n \to R$, which is given by

$$F_\delta(x) = \sum_{i=1}^{n}\phi_\delta(x_i) = \sum_{x_i\in(-\delta,\delta)}\frac{x_i^2}{2\delta} + \sum_{|x_i|\geq\delta}\left(|x_i| - \frac{\delta}{2}\right).$$

The function F_δ is a uniform approximation of the l_1-norm function norm. Clearly,

$$\begin{aligned} |F_\delta(x) - \|x\|_1| &= \left|\sum_{|x_i|<\delta}\left(\frac{x_i^2}{2\delta} - |x_i|\right) + \sum_{|x_i|\geq\delta}\left((|x_i| - \frac{\delta}{2}) - |x_i|\right)\right| \\ &= \left|\frac{\delta}{2}\sum_{|x_i|<\delta}\left(\frac{x_i^2}{\delta^2} - \frac{2|x_i|}{\delta}\right) + \sum_{|x_i|\geq\delta}(-\frac{\delta}{2})\right| \\ &\leq \frac{\delta}{2}(card(\Gamma) + card(\Lambda)) = \frac{\delta n}{2}, \end{aligned}$$

where $\Gamma = \{i : |x_i| < \delta\}$, and $\Lambda = \{i : |x_i| \geq \delta\}$. Notice that

$$\lim_{\delta \to 0} F_\delta(x) = \lim_{\delta \to 0} \sum_{i=1}^{n} \phi_\delta(x_i) = \|x\|_1.$$

Because of the above fact, when δ is sufficient small, minimizing the l_1-norm over a feasible set is almost identical to minimizing the Huber function over the same feasible set. Notice that the first term of $F_\delta(x)$ can be written as

$$\sum_{|x_i| \in \delta} \frac{x_i^2}{2\delta} = \|Wx\|_2^2, \quad \text{for} \quad -\delta < x < \delta,$$

where is $W = diag(w)$, and $w = (w_1, ..., w_n)$, $\quad w_i = \frac{1}{2\delta}$, $\forall i \in \Gamma$. $\square$

So, the l_1-minimization already uses weighted minimization. As seen, when x_i is small enough i.e, $|x_i| < \delta$, then a large weight $(\frac{1}{2\delta})$ is assigned to the component. This weight acts like a large penalty and forces the component to tend to zero. The l_1-minimization ignores the difference between the large components of x i.e, $|x_i| > \delta$, by simply allocating them the same weight. This fact motivates that the performance of the l_1-minimization might be improved. For example, a natural way to improve the performance of the l_1-minimization is to define an iterative algorithm and to allocate the weights to the components according to the value of them at each iteration. In the next section, we give some natural and geometrical explanation for promoting weighted approaches.

4.3 How weights may perform to find the sparsest solution of an underdetermined linear system of equations

In this section, we demonstrate several graphical examples of different norm-functions and weighted norm-functions for finding the sparsest solution to a simple linear system. The aim of these examples is to illustrate how these different methods work, and how the weighted functions may perform to find the sparsest solution to a linear system.

Suppose in the problem (4.2) the matrix A, and the vector b are as follows:

$$A = \begin{pmatrix} \frac{1}{2} & 1 \end{pmatrix}, \; b = \begin{pmatrix} \frac{3}{2} \end{pmatrix}. \tag{4.6}$$

Then the cardinality minimization problem (4.2) can be written as

$$\begin{aligned} \underset{x}{\text{Minimize}} \; & \|x\|_0 \\ \text{s.t. } & \frac{1}{2}x_1 + x_2 = \frac{3}{2}. \end{aligned} \tag{4.7}$$

Clearly, the above problem has two optimal (sparse) solutions, which are:

$$x_1 = 0, \; x_2 = \frac{3}{2}, \text{ and } \; x_1 = 3, \; x_2 = 0.$$

As mentioned before, one of the usual approaches to find the sparsest solution of the problem (4.7) is to replace $\|x\|_0$ with $\|x\|_1$, and then to solve the following l_1-minimization problem:

$$\text{(4.8)} \qquad \begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_1 \\ &\text{s.t.}\ \frac{1}{2}x_1 + x_2 = \frac{3}{2}. \end{aligned}$$

Figure 4.1 shows how the l_1-minimization may find the sparsest solution of this linear system, i.e. $x_1 = 0,\ x_2 = \frac{3}{2}$.

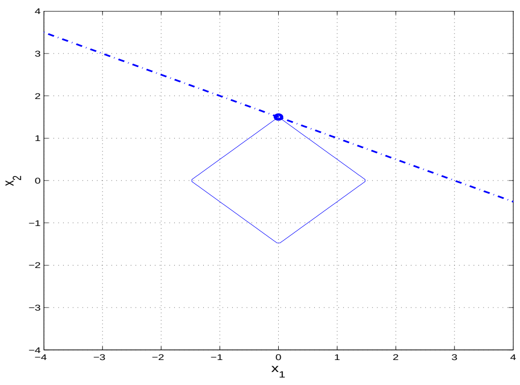

Figure 4.1: This figure shows how the l_1-minimization may locate the sparsest solution of a linear system.

Another approach to solve the problem (4.7) is to replace $\|x\|_0$ with $\|x\|_2$, and then to solve the following l_2-minimization problem:

$$\text{(4.9)} \qquad \begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_2 \\ &\text{s.t.}\ \frac{1}{2}x_1 + x_2 = \frac{3}{2}. \end{aligned}$$

Figure 4.2 shows even for a very simple linear system, the l_2-minimization may fail to locate the sparsest solution. The optimal solution found by the l_2-minimization is $x_1 = \frac{3}{5},\ x_2 = \frac{6}{5}$, which obviously is not the sparsest solution.

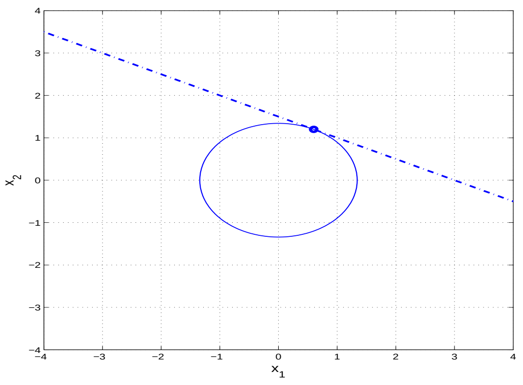

Figure 4.2: This figure shows how the l_2-minimization may fail to locate the sparsest solution of a simple linear system.

Also replacing $\|x\|_0$ with $\|x\|_\infty$ results the following optimization problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_\infty \\ &\text{s.t.}\ \frac{1}{2}x_1 + x_2 = \frac{3}{2}. \end{aligned} \tag{4.10}$$

Figure 4.3 shows how the l_∞-minimization may fail to locate the sparsest solution of a simple linear system. The optimal solution found by l_∞-minimization is $x_1 = 1,\ x_2 = 1$, which clearly is not the sparsest solution.

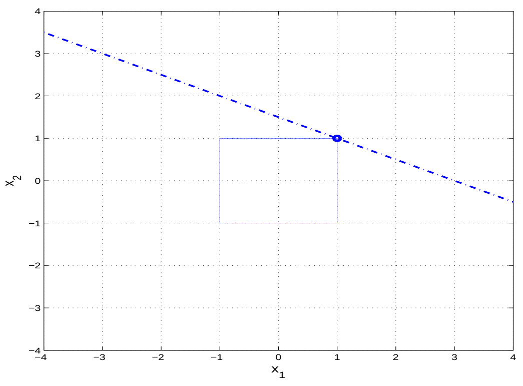

Figure 4.3: This figure shows how the l_∞-minimization may fail to locate the sparsest solution of a simple linear system.

Now lets consider the reweighted l_1-minimization. First, we allocate a large penalty like $\lambda \gg 0$ to the first component of the vector x. Hence the related weighted l_1-minimization problem can be written as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \lambda|x_1| + |x_2| \\ &\text{s.t. } \frac{1}{2}x_1 + x_2 = \frac{3}{2}. \end{aligned} \tag{4.11}$$

Figure 4.4 shows how the weighted l_1-minimization may locate the sparsest solutions of the linear system. In this case, the optimal solution is $x_1 = 0,\ x_2 = \frac{3}{2}$.

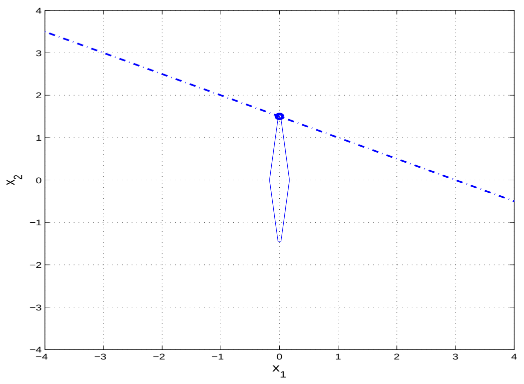

Figure 4.4: This figure shows how the weighted l_1-minimization may locate the sparsest solution of a linear system.

Note that if we use the reweighted l_1-algorithm starting from the solution of the l_1-minimization, then we find the same optimal solution as well. We will discuss the reweighted l_1-minimization in the next chapter.

Now we put the penalty ($\lambda \gg 0$) on the second component of the vector x. Hence the related weighted l_1-minimization problem can be written as follows:

$$\begin{aligned}&\underset{x}{\text{Minimize}}\ |x_1| + \lambda |x_2| \\ &\quad \text{s.t. } \frac{1}{2}x_1 + x_2 = \frac{3}{2}.\end{aligned} \tag{4.12}$$

Figure 4.5 shows how the weighted l_1-minimization may locate the sparsest solutions of the linear system. Notice that the change of the weights may enable us to find the second sparsest solution of the problem, i.e. $x_1 = 3,\ x_2 = 0$. One of the open questions in this area is how to jump from one sparse solution to another one. Note that in lots of cases the problem may have many sparse solutions.

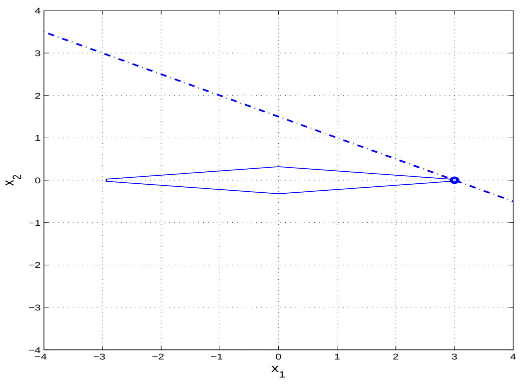

Figure 4.5: This figure shows how the weighted l_1-minimization may locate the sparsest solution of a linear system.

Now we consider the reweighted l_2-minimization, which can be written as follows:

$$\begin{aligned}&\underset{x}{\text{Minimize}}\ \lambda x_1^2 + x_2^2 \\ &\quad \text{s.t. } \frac{1}{2}x_1 + x_2 = \frac{3}{2},\end{aligned} \tag{4.13}$$

where $\lambda \gg 0$ is a large penalty for x_1^2. The optimal solution of the above problem is $x_1 = 0,\ x_2 = \frac{3}{2}$, which is one of the sparsest solutions. This is shown in Figure 4.6.

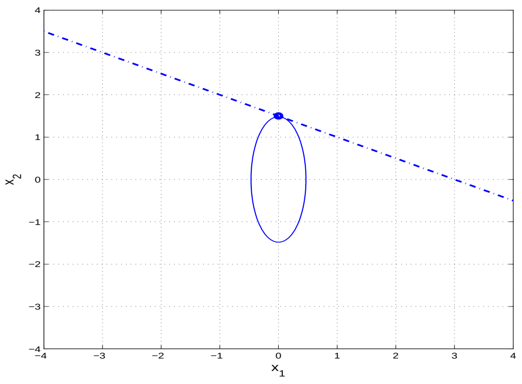

Figure 4.6: This figure shows how the weighted l_2-minimization may locate the sparsest solution of a linear system.

Another form of the weighted l_2-minimization can be written as follows:

$$\begin{aligned} \underset{x}{\text{Minimize}} \; & x_1^2 + \lambda x_2^2 \\ \text{s.t.} \; & \frac{1}{2}x_1 + x_2 = \frac{3}{2}, \end{aligned} \tag{4.14}$$

where $\lambda \gg 0$ is a large penalty for x_2^2. The optimal solution of the above problem is $x_1 = 3,\ x_2 = 0$, which is one of the sparsest solutions. This is shown in Figure 4.7.

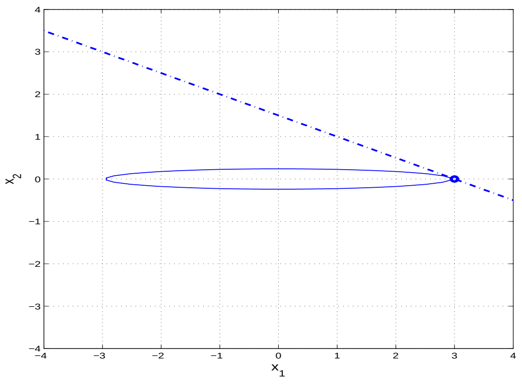

Figure 4.7: This figure shows how the weighted l_2-minimization locates the sparsest solution of a linear system.

The weighted l_∞-minimization problems can be written as the follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \;\; \max\{\lambda|x_1|, |x_2|\} \\ &\text{s.t.} \;\; \frac{1}{2}x_1 + x_2 = \frac{3}{2}, \end{aligned} \tag{4.15}$$

and

$$\begin{aligned} &\underset{x}{\text{Minimize}} \;\; \max\{|x_1|, \lambda|x_2|\} \\ &\text{s.t.} \;\; \frac{1}{2}x_1 + x_2 = \frac{3}{2}, \end{aligned} \tag{4.16}$$

where $\lambda \gg 0$ is a large penalty. Note that in this case, we should choose a larger penalty than what used in the weighted l_1-minimization and the weighted l_2-minimization. If a proper weight is not found, then the weighted l_∞-minimization may fail to find the sparsest solution. This is shown in Figures 4.8 and 4.9. But a very big penalty makes the weighted l_∞-minimization successful to find the the solutions $x_1 = 0$, $x_2 = \frac{3}{2}$ and $x_1 = 3$, $x_2 = 0$ for the problems (4.15) and (4.16), respectively.

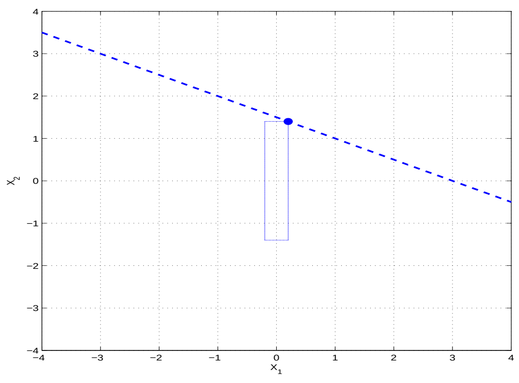

Figure 4.8: This figure shows how the weighted l_∞-minimization may locate the sparsest solution of a linear system.

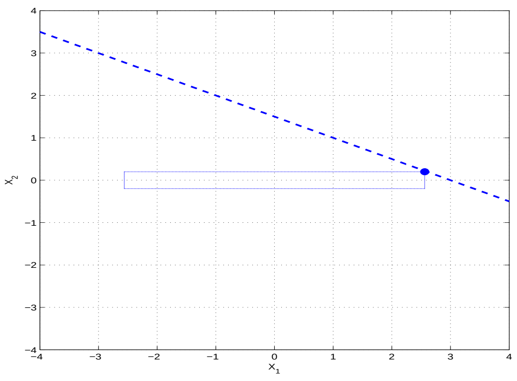

Figure 4.9: This figure shows how the weighted l_∞-minimization may locate the sparsest solution of a linear system.

Now we explain another example in higher dimension.

Example 4.1. *Suppose the matrix A, and the vector b are as follows:*

$$A = \begin{pmatrix} 3 & -2 & 4 & 0 & -1 & 1 \\ 5 & -2 & -3 & 5 & 0 & 6 \\ 1 & 2 & -1 & -5 & -6 & 7 \\ 2 & -3 & 0 & 4 & -6 & 0 \end{pmatrix}, \; b = \begin{pmatrix} 0 \\ 5 \\ -5 \\ 4 \end{pmatrix}.$$

We would like to solve the following CMP:

$$\begin{aligned} &\underset{x}{Minimize} \; \|x\|_0 \\ &\quad s.t. \;\; Ax = b. \end{aligned} \tag{4.17}$$

The optimal solution (the sparsest solution) of the above problem is $\hat{x} = (0, 0, 0, 1, 0, 0)^T$, clearly, $\|x\|_0 = 1$. The l_1-minimization finds $x^* = (0, 0, -1.0078, 0.8532, -0.0978, -0.3327)^T$, which is not the sparsest solution.

The l_2-minimization finds $x^* = (0.0813, -0.4678, -0.9394, 0.5778, -0.0205, -0.3313)^T$, and the l_∞-minimization finds $x^* = (-0.8459, -0.8459, -0.8459, 0.8234, 0.0233, 0.1355)^T$. None of the solutions above are the optimal solution to the problem (4.17). So, for this example, the l_1, l_2, l_∞ fail to find the sparsest solution. However, weighted algorithms have more capability to locate the optimal solution of the problem. As we see in the numerical experiments section, most of the reweighted l_j-minimizations, where $j = 1, 2, ...15, \infty$, (if the weights are chosen properly), are more successful to find the optimal solution of the problem, i.e. $\hat{x} = (0, 0, 0, 1, 0, 0)$. Without using weights, the different norms may yield totally different solutions, which may be far from a sparse solution of the system. However, once we introduce the weights, such differences are significantly reduced. Even by applying suitable weights, it may be observed that the performance the algorithms can be comparable.

In the following section, we explain how to define different merit functions for sparsity, i.e. how to construct new concave approximations for $\|x\|_0$. Also, we show that even for some functions which are not approximating $\|x\|_0$, we may improve their sparsity measurement by adding suitable weights.

4.4 Weights may improve sparsity enhancing property of merit functions

As mentioned before, the merit functions should have strong sparsity measurement property. Obviously, if a function approximates the $\|x\|_0$-function, then it has a strong sparsity measurement, or in other words, it has strong sparsity enhancing property. Under certain conditions, minimizing such functions may hopefully lead us to a sparse solution to the linear system. Notice that adding suitable weights to a merit function may enhance its sparsity measurement property. We will discuss this in this section, and we will explain more about merit functions and their properties in the next chapter. However, in this chapter, let's define the general merit function for sparsity, $F(x)$, as the function which leads us to a sparse solution to the system, i.e., minimizing such function may result in finding a sparse solution to the system. Additionally, some other properties will be defined for the function $F(x)$ as follows:

$$F(x) = \sum_{i=1}^{n} \psi_i(|x_i|),$$

where $\psi_i : R_{++} \to R,\ i = 1, ..., n$ *are twice continuously differentiable and nondecreasing, functions.*

Thus, a general cardinality minimization problem can be written as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \sum_{i=1}^{n} \psi_i(|x_i|) \\ &\text{s.t. } Ax = b. \end{aligned} \tag{4.18}$$

To avoid the division by zero and the non-differentiability (in some cases) of these functions, we may perturb the function by introducing a parameter $\epsilon > 0$. In different cases depending on the merit function, there are different ways to perturb these function. We perturb these function as follows; Replace $|x_i|$ by the functions $\sqrt{x_i^2 + \epsilon}$ or $|x_i| + \epsilon$. Notice that

$$\lim_{\epsilon \to 0^+} \psi_i(\sqrt{x_i^2 + \epsilon}) = \psi_i(|x_i|),$$

$$\lim_{\epsilon \to 0^+} \psi_i(|x_i| + \epsilon) = \psi_i(|x_i|),$$

Usually, we use the first perturbation to avoid the non-differentiability, and the second one to avoid the division by zero when we differentiate these functions. So, the cardinality minimization problem can be approximated by the following problem:

$$\begin{aligned}\underset{x}{\text{Minimize}}\ & F_\epsilon(x)\\ \text{s.t.}\ & Ax=b,\end{aligned} \tag{4.19}$$

where

$$F_\epsilon(x)=\sum_{i=1}^{n}\psi_i(\sqrt{x_i^2+\epsilon}), \quad \text{or} \tag{4.20}$$

$$F_\epsilon(x)=\sum_{i=1}^{n}\psi_i(|x_i|+\epsilon). \tag{4.21}$$

As we will see in the next chapter, $F_\epsilon(x)$ should be concave to ensure that the function is an approximation to $\|x\|_0$.

It can be guaranteed that F_ϵ is concave if we choose all ψ_is to be concave. In this thesis, it is assumed that all of the functions ψ_is are concave. Note that we always seek for concave merit functions for sparsity, because of the nature of the $\|x\|_0$-function. Since the cardinality minimization problem is NP-hard, it seems that there is no possibilities to approximate $\|x\|_0$ to any level of accuracy by a convex function. But there are many concave functions, which can approximate $\|x\|_0$ to any level of accuracy. More explanation can be found in the next chapter.

For more illustration, here we present some examples about different choices of ψ_is.

Example 4.2. *Suppose all ψ_is are identical mappings, i.e, $\psi_i(\tau)=\tau,\ i=1,...,n.$*

We apply $\psi_i(\tau)=\tau$, then we have

$$F(x)=\sum_{i=1}^{n}\psi_i(|x_i|)=\sum_{i=1}^{n}|x_i|=\|x\|_1.$$

Note that

$$F(x) = \lim_{\epsilon \to 0} F_\epsilon(x) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} (|x_i| + \epsilon) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \sqrt{x_i^2 + \epsilon}.$$

Therefore, the problem (4.18) reduces to the l_1-minimization problem, and a smooth approximation of the problem (4.18) can be written as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \sum_{i=1}^{n} \sqrt{x_i^2 + \epsilon} \\ &\text{s.t.} \ \ Ax = b. \end{aligned} \tag{4.22}$$

Example 4.3. *Let $\psi_i(\tau) = log(\tau), \ i = 1, ..., n$.*

First define $J = \{i \in \mathbb{N}; \ x_i \neq 0\}$. If we apply $\psi_i(\tau) = \log(\tau)$, then

$$F(x) = \sum_{J} \psi(|x_i|) = \sum_{J} \log |x_i|.$$

Hence

$$F(x) = \lim_{\epsilon \to 0} F_\epsilon(x) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \log(|x_i| + \epsilon) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \log \sqrt{x_i^2 + \epsilon}.$$

Therefore, the approximations of the problem (4.18) can be cast as

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \sum_{i=1}^{n} \log \sqrt{x_i^2 + \epsilon} \\ &\text{s.t.} \ \ Ax = b, \end{aligned} \tag{4.23}$$

and

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \sum_{i=1}^{n} (|x_i| + \epsilon) \\ &\text{s.t.} \ \ Ax = b. \end{aligned} \tag{4.24}$$

Example 4.4. *Suppose* $\psi_i(\tau) = \tau^p, \ i = 1, ..., n.$

If we replace $\psi_i(\tau) = \tau^p$, then

$$F(x) = \sum_{i=1}^{n} \psi_i(|x_i|) = \sum_{i=1}^{n} |x_i|^p = \|x\|_p^p.$$

So,

$$F(x) = \lim_{\epsilon \to 0} F_\epsilon(x) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} (|x_i| + \epsilon)^p = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \left(\sqrt{x_i^2 + \epsilon} \right)^p .$$

Thus, in this case, the problem (4.18) reduces to the well-known l_p-minimization ($0 < p < 1$), so an approximations of the problem can be cast as

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \sum_{i=1}^{n} \left(\sqrt{x_i^2 + \epsilon} \right)^p \\ &\text{s.t.} \ \ Ax = b, \end{aligned} \tag{4.25}$$

Example 4.5. *Set* $\psi_i(\tau) = \mathit{log}(1 + \tau), \ i = 1, ..., n.$

$$F(x) = \sum_{i=1}^{n} \log(1 + |x_i|) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \log(1 + (|x_i| + \epsilon)) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \log \left(1 + \sqrt{x_i^2 + \epsilon} \right) .$$

The same as the above examples, these functions can be also used to define new approximations for the cardinality minimization problem.

There are infinitely many functions, which can be used to define new merit functions F. For instance, we may also use some proper mixture of the functions ψ_is to construct some more complicated merit functions for sparsity.

Let Ψ be the cone generated by univariate twice differentiable, nondecreasing concave functions. Then any function in Ψ can be used to construct a new sparsity merit function. Notice that any constant function, $\psi_i(\tau) = \textit{constant}$, is in the cone. So, we may also generate a new merit function from this fact. Here we give another example to illustrate how to construct new merit functions for sparsity.

Example 4.6. *Set* $\psi_i(\tau) = -\frac{1}{\tau} \in \Psi,\ i = 1, ...n.$

This yields the merit function

$$F_\epsilon(x) = \sum_{i=1}^{n} \psi_i(|x_i| + \epsilon) = \sum_{i=1}^{n} -\frac{1}{|x_i| + \epsilon}.$$

Now, the combination of the constant function, i.e, $\phi_i(\tau) = 1 \in \Psi$ and ψ_i yields the following function:

$$\xi_i = \phi_i(\tau) + \epsilon\psi_i(\tau) = 1 - \frac{\epsilon}{\tau}.$$

Clearly, $\xi_i \in \Psi$, so, we can construct the following merit function:

$$F_\epsilon(x) = \sum_{i=1}^{n} \xi_i(|x_i| + \epsilon) = \sum_{i=1}^{n} (1 - \frac{\epsilon}{|x_i| + \epsilon}).$$

This function is an approximation to the $\|x\|_0$-function, since

$$\lim_{\epsilon \to 0} \sum_{i=1}^{n} (1 - \frac{\epsilon}{|x_i| + \epsilon}) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \frac{|x_i|}{|x_i| + \epsilon} = \|x\|_0.$$

For more explanation about the above formulation, note that

$$\lim_{\epsilon \to 0} \frac{|x_i|}{|x_i| + \epsilon} = 0 \text{ if } x_i = 0,\ i = 1, ..., n,$$

and

$$\lim_{\epsilon \to 0} \frac{|x_i|}{|x_i| + \epsilon} = 1 \text{ if } x_i \neq 0,\ i = 1, ..., n.$$

In the next chapter, we introduce new merit functions, which lead us to new reweighted l_1-algorithms.

Here we explain how we can apply weights to improve the sparsity measurement of a function. Suitable weights may improve the approximation level to the $\|x\|_0$-function, i.e, adding good weights to some merit functions may lead us to a good approximation to the $\|x\|_0$-function. If we add weights to the problem (4.18), the achieved weighted problem can be written as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \sum_{i=1}^{n} w_i \psi_i(|x_i|) \\ &\text{s.t. } Ax = b, \end{aligned} \tag{4.26}$$

and its approximation can be cast as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \sum_{i=1}^{n} w_i \psi_i \left(\sqrt{x_i^2 + \epsilon} \right) \\ &\text{s.t. } Ax = b. \end{aligned} \tag{4.27}$$

There are many different options to define weights. In most of the cases, weights depend on the parameter ϵ, or they depend on the both parameters ϵ and x. Note that If we use a mixture of the log funtion and the l_p-norm $(0 < p < 1)$ to construct a merit function, then the weights may also rely on the parameters ϵ, x, and p (see chapter 5).

In the following example, we show that by choosing proper weights, the objective function of the problem (4.27) can approximate the $\|x\|_0$-function to any level of accuracy.

Example 4.7. *The weights depend on both the current point x and ϵ.*

Let $\psi_i = \log(1+\tau)$, $i = 1, ..., n$, and let

$$w_i = \frac{1}{\log(1+\sqrt{x_i^2+\epsilon^p})}, \; i = 1, ..., n,$$

where $0 < p < 1$ is any fixed constant (e.g., $p = \frac{1}{2}, \frac{1}{3}, ...$), then

$$\lim_{\epsilon \to 0} \sum_{i=1}^{n} w_i \psi_i \left(\sqrt{x_i^2+\epsilon}\right) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \frac{\log\left(1+\sqrt{x_i^2+\epsilon}\right)}{\log(1+\sqrt{x_i^2+\epsilon^p})} = \|x\|_0$$

Similarly, we may choose

$$w_i = \frac{1}{\log(1+|x_i|+\epsilon^p)},$$

such that the above limitation remains valid.

As we see in the numerical experiment section, the choice of the initial point for the reweighted algorithm plays a vital rule. This means, by choosing a proper starting point, most of the weighted l_j-minimizations ($j \geq 1$) are successful in locating the sparsest solution, Tables 4.1, 4.2 and 4.4 .

In the next section, we continue discussing the weights, and we show that choosing proper weights may reduce the gap between the performances of different merit functions to locate a sparse solution to a linear system.

4.5 Weights may reduce the gap between some merit functions

In this section, we show that suitable weights may significantly reduce the gap between some of the merit functions. We suppose that the initial points in the wl_j-algorithms ($j \geq 2$) are set to be the solution of the l_1-minimization. Also, we explain how to define different kinds of weights for weighted norm-functions in order to have a function with high sparsity measurement property. Besides, we discuss which choices of weights may reduce the gap between different merit functions. For example, in what follows we illustrate what choices of weights may reduce the gap between the performances of the $\|Wx\|_2$-minimization and the $\|x\|_1$-minimization to locate a sparse solution of a linear system.

One approach to find such weights is to define the weights in a way that each $w_i^2 x_i^2$ becomes equivalent to $|x_i|,\ i = 1, ..., n$. Suppose that $W = \text{diag}(w_1, w_2, ..., w_n)$, then

$$\|Wx\|_2 = \sqrt{\sum_{i=1}^{n} w_i^2 x_i^2}.$$

Now let's choose

$$w_i = \frac{1}{(x_i^2 + \epsilon)^{\frac{1}{4}}}. \tag{4.28}$$

If we replace the above weights to the weighted l_2-norm, then

$$\|Wx\|_2 = \sqrt{\sum_{i=1}^{n} \frac{1}{(x_i^2 + \epsilon)^{\frac{1}{2}}} x_i^2}.$$

Note that $\sqrt{x_i^2+\epsilon} \approx |x_i|$, when ϵ is small enough. This yields:

$$\|Wx\|_2 \approx \sqrt{\sum_{i=1}^{n} |x_i|} = \sqrt{\|x\|_1},$$

and therefore,

$$\|Wx\|_2^2 \approx \|x\|_1.$$

In general, choosing proper weights in $\|Wx\|_j$-minimization ($j \geq 2$) can make it equivalent to the $\|x\|_1$-minimization. Set

$$w_i = (|x_i|^j + \epsilon)^{\frac{1-j}{j^2}}. \tag{4.29}$$

Clearly, this choice of the weight will result in the following:

$$\|Wx\|_j^j \approx \|x\|_1.$$

It is worth mentioning that the weighted l_2-norm function can also be an approximation to the $\|x\|_0$-function. This can be proved as follows:

Set the weights as

$$w_i = \frac{1}{\sqrt{x_i^2+\epsilon}}, \; i = 1, ..., n, \tag{4.30}$$

and let $W = \mathrm{diag}(w_1, w_2, ..., w_n)$. Therefore

$$\|Wx\|_2 = \sqrt{\sum_{i=1}^{n} (w_i x_i)^2} = \sqrt{\sum_{i=1}^{n} \frac{x_i^2}{x_i^2+\epsilon}}.$$

Now taking the limit when $\epsilon \to 0$, yields: For $x_i \neq 0$, we have

$$\lim_{\epsilon \to 0} \frac{x_i^2}{x_i^2 + \epsilon} = 1,$$

and for $x_i = 0$, we have

$$\lim_{\epsilon \to 0} \frac{x_i^2}{x_i^2 + \epsilon} = 0.$$

Hence

$$\lim_{\epsilon \to 0} \sum_{i=1}^{n} \frac{x_i^2}{x_i^2 + \epsilon} = \|x\|_0.$$

This motivates us to test another reweighted l_2-algorithm with the weights defined in (4.30), see the numerical experiment section.

In general, for weighted l_j-norms ($j \geq 2$), if the weights are chosen as

$$w_i = (x_i^j + \epsilon)^{-\frac{1}{j^2}}, \; i = 1, ..., n,$$

then we can conclude that

$$\lim_{\epsilon \to 0} \|Wx\|_j^j \approx \|x\|_0,$$

where $W = \text{diag}(w_1, ..., w_n)$.

4.6 General reweighted l_j ($j \geq 1$) algorithms

In chapter 5, we will discuss the reweighted l_1 algorithms in details, however in this section we explain a general reweighted l_j ($j \geq 1$) algorithm.

First, we should choose a starting point x^1, and an initial weight $\omega^1 = (\omega_1^1, ..., \omega_n^1)$ for the algorithm. Usually the weights at the iteration $l+1$ are chosen using the solution vector, x^l, from the iteration l. Note that at each iteration l the vector $\omega^l = (\omega_1^l, ..., \omega_n^l)$ is the penalty vector, where ω_is can be interpreted as penalties that discourage the existence of nonzero elements of the vector $x \in \mathbb{R}^n$. Also a parameter ϵ is used in our penalty vector, ω^l, to avoid the division by zero. Sometimes we may use some updating rule for ϵ as well, e.g. set $\epsilon^{l+1} = 0.5\epsilon^l$. Hence, a general reweighted l_j algorithm can be defined as follows:

Algorithm 4.1. *Set l as an index which counts the iterations, and choose a small enough ϵ,*

Step 0: Choose a starting(initial) point x^1 for the algorithm,

Step 1: Set $l = 1$ and $\omega^l = (\omega_1, ..., \omega_n)$,

Step 2: The iterative scheme; Solve

$$x^{l+1} = argmin\{\|diag(\omega^l)x\|_j : Ax = b\}, where\ j \geq 1, \tag{4.31}$$

Step 3: Update the weights; update the penalty vector ω^{l+1} using the optimal solution, x^l, achieved in the last step,

Step 4: If some stopping criteria holds, stop. Otherwise, go to step 2.

We may use the weights introduced in (4.28) and (4.30) to define the following reweighted l_2-minimization problems:

$$\begin{aligned} x^{l+1} = \text{argmin } & \sum_{i=1}^{n} \left((x_i^2)^l + \epsilon\right)^{-0.25} x_i^2 \\ \text{s.t. } & Ax = b, \end{aligned} \tag{4.32}$$

and

$$\begin{aligned} x^{l+1} = \text{argmin } & \sum_{i=1}^{n} \left((x_i^2)^l + \epsilon\right)^{-0.5} x_i^2 \\ \text{s.t. } & Ax = b, \end{aligned} \tag{4.33}$$

where l is the number of the iteration.

In our numerical experiments, we compared the performances of the above reweighted l_2-minimization problems with the l_1-minimization, see Figures 4.10 and 4.11. The numerical experiments show that the choice of ϵ is very important to improve the performances of the algorithms. In addition, by using the above discussion, we introduced some weights for the reweighted l_3-algorithm, and we also compared the performances of the reweighted l_3-minimization with the l_1-minimization, via different choices of ϵ, see Figures 4.12 and 4.13.

Note that after adding the weights, the resulting function may lose its simplicity or convexity. The procedure of adding weights normally generates a concave function with a strong sparsity measurement. If the function is a good continuous approximation to the $\|x\|_0$-function, then it means the function is concave. One of the common approaches to solve concave optimization problems is the linearization method, which will be explained in the next chapter.

In the next section, we demonstrate some numerical examples to see how weights may reduce the gap between merit functions for sparsity.

4.7 Numerical experiments

In this section, we report some results from our numerical experiments to support the points made in this chapter. These experiments show that weights (if chosen properly) may significantly reduce the gap between merit functions in finding a sparse solution to a linear system.

We start with the simple Example 4.1. As seen, the optimal solution of the problem is $\hat{x} = (0,0,0,1,0,0)^T$, and $\|\hat{x}\|_0 = 1$. In this example, numerical tests show that the l_1-minimization finds the solution with cardinality 4, and all the l_j-minimizations for $j = 2, ..., \infty$ find the solutions with cardinality 6. However, reweighted l_j-minimizations

$(j \geq 1)$ are much better in general. In the Table 4.1, we use the weight $w_i^l = \frac{1}{|x_i^l|+\epsilon_l}$ in the reweighted l_j-minimization with $j = 1, ..., 5, 15, \infty$. And in the Table 4.2, we use the weight $w_i^l = \frac{1}{(|x_i^l|+\epsilon_l)^{1.5}}$ in the reweighted l_j-minimization with $j = 1, ..., 5, 15, \infty$. Also, the updating rule for ϵ is $\epsilon_{l+1} = 0.25\epsilon_l$, with $\epsilon_0 = 0.0001$. The number of the iterations for all of the reweighted algorithms is equal to 4, i.e, $l = 4$, and we run the algorithms from 10 different initial points. After presenting these numerical experiments in the following tables, we continue to improve the results by changing the number of iterations.

Table 4.1: Comparison of different reweighted l_j-minimizations with $w_i^l = \frac{1}{|x_i^l|+\epsilon_l}$. The number of iterations is $l = 4$. The numbers in the columns show the cardinality of the solution.

Initial Point x^0	wl_1	wl_2	wl_3	wl_4	wl_5	wl_{15}	wl_∞
$(0,0,0,0,0,0)^T$	1	1	6	6	6	6	6
$(1,1,1,1,1,1)^T$	1	1	6	6	6	6	6
$(0,1,1,1,1,1)^T$	1	1	5	5	6	6	6
$(0,0,1,1,1,1)^T$	1	1	1	1	1	5	6
$(0,1,1,1,0,0)^T$	1	1	1	1	1	6	6
$(0,1,0,1,0,1)^T$	1	1	1	1	1	6	6
$(1,1,1,0,0,0)^T$	4	6	6	6	6	6	6
$(0,100,100,100,100,100)^T$	1	1	5	5	6	6	6
$(0,0,100,100,100,100)^T$	1	1	1	1	1	5	5
$(0,0,0,100,100,100)^T$	1	1	1	1	1	5	6

Table 4.2: Comparison of different reweighted l_j-minimizations with $w_i^l = \frac{1}{(|x_i^l|+\epsilon_l)^{1.5}}$. The number of iterations is $l = 4$. The numbers in the columns show the cardinality of the solution.

Initial Point x^0	wl_1	wl_2	wl_3	wl_4	wl_5	wl_{15}	wl_∞
$(0,0,0,0,0,0)^T$	1	1	1	5	5	5	6
$(1,1,1,1,1,1)^T$	1	1	1	5	5	6	6
$(0,1,1,1,1,1)^T$	1	1	1	1	1	5	5
$(0,0,1,1,1,1)^T$	1	1	1	1	1	1	1
$(0,1,1,1,0,0)^T$	1	1	1	1	1	1	6
$(0,1,0,1,0,1)^T$	1	1	1	1	1	1	6
$(1,1,1,0,0,0)^T$	1	1	5	5	5	6	6
$(0,100,100,100,100,100)^T$	1	1	1	1	1	5	5
$(0,0,100,100,100,100)^T$	1	1	1	1	6	6	1
$(0,0,0,100,100,100)^T$	1	1	1	1	6	6	6

Now, we generate randomly distributed matrices A and the sparse vectors x. The matrices A are normally distributed with mean zero and variance 1, i.e, $N(0,1)$. We generate 500 such matrices A, and sparse vectors x, and compare the frequency of success of the l_j-minimization and the reweighted l_j-minimization ($j = 1,2,3,\infty$) in finding 5-sparse, 7-sparse, 9-sparse, 11-sparse and 12-sparse solutions of the linear system. We use the weights $w_i^l = \left(|x_i^l| + \epsilon_l\right)^{-1}$ for the reweighted l_1-minimization and the reweighted l_2-minimization, and $w_i^l = \left(|x_i^l| + \epsilon_l\right)^{-1.5}$ for the reweighted l_3-minimization and the reweighted l_∞-minimization. The initial point for all of the algorithm are set to be $x^0 = \mathbf{1} \in \mathbb{R}^{120}$, where $\mathbf{1}$ is a vector with all entries are equal to 1. Also $\epsilon_0 = 0.01$, and the updating rule for ϵ_l is $\epsilon_{l+1} = 0.5\epsilon_l$. The results are presented in Table 4.3. As seen, the l_1-minimization is efficient in finding the sparsest solution of a linear system. The l_2, l_3 and l_∞-minimization fail to locate any sparse solution when $\|x\|_0 \le 12$. However, the gap between them was significantly reduced when we add the weights to the functions. As shown, the reweighted l_1-minimization and the reweighted l_2-minimization perform similarly in many cases. Also, note that the frequencies of success in the reweighted

l_3-minimization and the reweighted l_∞-minimization are significantly improved.

Table 4.3: Comparison of l_j-minimizations and reweighted l_j-minimizations for $j = 1, 2, 3, \infty$. The numbers in the columns show the frequencies of success in finding the sparse solutions.

Sparsity	l_1	l_2	l_3	l_∞	wl_1	wl_2	wl_3	wl_∞
$\|x\|_0 \leq 5$	98.6%	0%	0%	0%	99.2%	92%	70.2%	0.6%
$\|x\|_0 \leq 7$	98.6%	0%	0%	0%	99.2%	93%	74.8%	0.8%
$\|x\|_0 \leq 9$	98.6%	0%	0%	0%	99.2%	93.8%	76.8%	1%
$\|x\|_0 \leq 11$	98.6%	0%	0%	0%	99.2%	94.2%	78%	1%
$\|x\|_0 \leq 12$	98.6%	0%	0%	0%	99.2%	94.6%	79.6%	1%

Now, we test different reweighted l_j-minimizations with a much higher number of iteration for finding the sparsest solution of the Example 4.1. The numerical experiments show that for the reweighted l_1-minimization usually 4 iterations are enough to find the optimal solution. However, for the reweighted l_j-minimization ($j \geq 2$), we should increase the number of iterations to locate the solutions with higher sparsity. We set the weights as $w_i^l = \frac{1}{|x_i^l|+\epsilon_l}$, and iterate the algorithm 40 times, i.e, $l = 40$. Based on our experiments, after 40 iterations there will not be any improvements to the optimal solution of the problem. This means one may stop the algorithm at the iteration k if $x^k = x^{k-1} = x^{k-2}$.

Table 4.4: Comparison of different reweighted l_j-minimizations with $w_i^l = \frac{1}{|x_i^l|+\epsilon_l}$. The number of iterations is $l = 40$. The numbers in the columns show the cardinality of the solution.

Initial Point x^0	wl_1	wl_2	wl_3	wl_4	wl_5	wl_{15}	wl_{20}	wl_{200}	wl_∞
$(0,0,0,0,0,0)^T$	1	1	1	1	1	1	5	6	6
$(1,1,1,1,1,1)^T$	1	1	1	1	1	1	5	6	6
$(0,1,1,1,1,1)^T$	1	1	1	1	1	1	1	6	6
$(0,0,1,1,1,1)^T$	1	1	1	1	1	1	1	2	6
$(0,1,1,1,0,0)^T$	1	1	1	1	1	1	1	5	6
$(0,1,0,1,0,1)^T$	1	1	1	1	1	1	1	1	6
$(1,1,1,0,0,0)^T$	4	4	4	4	4	5	5	6	6
$(0,100,100,100,100,100)^T$	1	1	1	1	1	1	1	5	6
$(0,0,100,100,100,100)^T$	1	1	1	1	1	1	1	1	5
$(0,0,0,100,100,100)^T$	1	1	1	1	1	1	1	1	5

We continue our numerical experiments by focusing on the following reweighted l_2-minimization and reweighted l_3-minimization problems:

$$x^{l+1} = \text{argmin} \sum_{i=1}^{n} \left((x_i^2)^l + \epsilon\right)^{-0.25} x_i^2 \quad \text{s.t. } Ax = b, \tag{4.34}$$

$$x^{l+1} = \text{argmin} \sum_{i=1}^{n} \left((x_i^2)^l + \epsilon\right)^{-0.5} x_i^2 \quad \text{s.t. } Ax = b, \tag{4.35}$$

$$x^{l+1} = \text{argmin} \sum_{i=1}^{n} \left((x_i^3)^l + \epsilon\right)^{-\frac{1}{3}} x_i^3 \quad \text{s.t. } Ax = b, \tag{4.36}$$

$$x^{l+1} = \text{argmin} \sum_{i=1}^{n} \left((x_i^3)^l + \epsilon\right)^{-\frac{2}{3}} x_i^3 \tag{4.37}$$
$$\text{s.t. } Ax = b,$$

As mentioned, a suitable weight may vanish the gap between the reweighted l_j-minimization ($j \geq 2$), and the l_1-minimization. We compared the performance of the 4 algorithms above with the l_1-minimization. 100 randomly generated matrices have been tested. The matrices A are normally distributed with mean zero and variance 1, i.e, $N(0,1)$, and we run the algorithms for different sparsity levels of the solution vector x, (i.e, $\|x\|_0 = 1, \ldots 16$). Figure 4.10 shows the performance of the first reweighted l_2-minimization (4.34) for finding the sparsest solution to the linear system $Ax = b$. Different choices of ϵ have been tested.

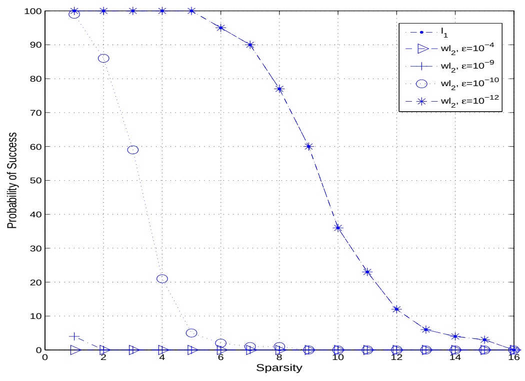

Figure 4.10: The performance of the reweighted l_2-minimization with $w_i = (x_i^2 + \epsilon)^{-0.25}$ for different choices of ϵ.

Figure 4.11 shows the performance of the second reweighted l_2-minimization (4.35) for finding the sparsest solution of the linear system $Ax = b$ for different choices of ϵ. See that if a suitable ϵ is chosen, then the performance of the algorithms will be quite similar to the l_1-minimization.

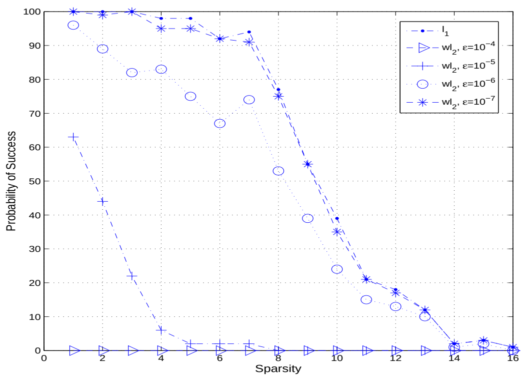

Figure 4.11: The performance of the reweighted l_2-minimization with $w_i = (x_i^2 + \epsilon)^{-0.5}$ for different choices of ϵ.

Figure 4.12 shows the performance of the first reweighted l_3-minimization (4.36) for finding the sparsest solution of a linear system. Clearly, if a proper ϵ is chosen, then the gap between the reweighted l_3-minimization and the l_1-minimization will be reduced significantly.

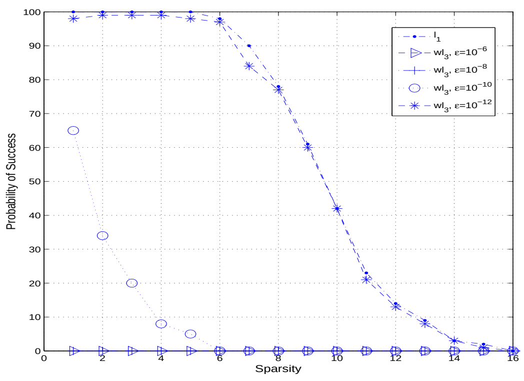

Figure 4.12: The performance of the reweighted l_3-minimization with $w_i = (x_i^3 + \epsilon)^{-\frac{1}{3}}$ for different choices of ϵ.

Figure 4.13 shows the performance of the second reweighted l_3-minimization (4.37) for finding the sparsest solution of linear system. See that if a suitable ϵ is chosen, then the gap between the reweighted l_3-minimization and the l_1-minimization will be reduced.

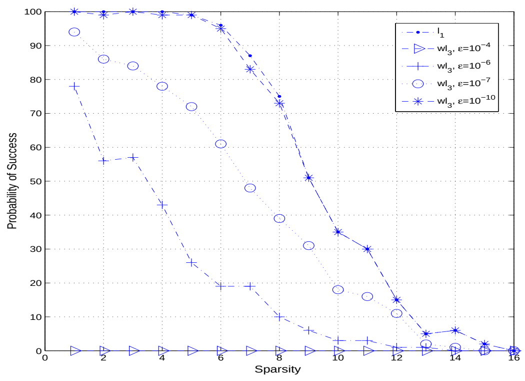

Figure 4.13: The performance of the reweighted l_3-minimization with $w_i = (x_i^3 + \epsilon)^{-\frac{2}{3}}$ for different choices of ϵ.

In the next section, we discuss the reweighted l_1-minimization, which is one of the most successful approaches to find a sparse solution of a linear system.

Chapter 5

Reweighted l_1-Algorithms[1]

We discussed different reweighted algorithms and the l_1-minimization method in the last chapter. Because of the effectiveness of l_1-minimization, it is natural to ask if there are other methods which can be comparable or outperforms the l_1-minimization. Numerical experiments show that proper reweighted l_1-minimizations outperform the l_1-minimization in many situations [34, 37, 43]. This means reweighted minimization methods may find the sparsest solution when the l_1-minimization fails. The reweighted l_1-minimization can be written as

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \|Wx\|_1 \\ &\quad \text{s.t.} \ \ Ax = b, \end{aligned} \tag{5.1}$$

where W is a diagonal matrix $W = \text{diag}(w)$ with $w = (w_1, w_2, ..., w_n) \in R^n_+$. The main idea of the reweighted l_1-minimization is to define the weights based on the current iterate x^l. These weights can be interpreted as large penalties for small component of the vector x. This means the weights force the small components to be zero in every iteration. Reweighted l_1-minimization is a new topic in the field of optimization and applied

[1]Most parts of this chapter can be found in [2].

mathematics, however reweighted least square (RLS) method has a relatively long history. RLS was proposed by Lawson [84] in 1960s, who introduced the weighted l_p-norm for solving a class of uniform approximation problems. This method has been extended to l_p-minimization ($0 < p < 1$) later. RLS is used in many areas such as robust statistical estimation [142, 78, 111] and FOCUSS algorithms [72]. The recent interest is now focused on reweighted l_1-minimizations which have shown powerfulness in finding the sparsest solution to an underdetermined linear system of equations.

A unified framework of reweighted l_1-algorithms has been recently proposed by Zhao and Li [152], where they illustrated how to define different merit functions for sparsity. These functions are certain concave approximations to the l_0-norm function. In this chapter, we continue to introduce more new merit functions, which can be employed to propose new weights for the reweighted l_1-algorithms. Especially, we focus on the numerical comparison between some new and existing reweighted l_1-algorithms. We show how the change of parameters in reweighted algorithms may affect the performance of the algorithms for finding the solution of the cardinality minimization problem. In our experiments, the problem data were generated according to different statistical distributions, and we test the algorithms on different sparsity level of the solution of the problem. Our numerical results demonstrate that these reweighted l_1-methods are very efficient for locating the solution of the cardinality minimization problem.

In the next section, we discuss some of the existing weighted l_1-algorithms, and the majorization-minimization(MM) method. Also, we apply the MM-method to the function which we defined in chapter 3 (3.16) to define a reweighted l_1-algorithm, and we continue to develop more approximations to the function $\|x\|_0$ in the other sections.

5.1 Weighted l_1-algorithms

Weighted l_1-minimization problem can be written as

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \omega^T|x| = \langle \omega, |x| \rangle \\ &\text{s.t.}\ \ Ax = b, \end{aligned} \tag{5.2}$$

where the vector $|x|$ is the componentwise absolute value of the vector x, and ω is the weight vector. The weight vector should be chosen in a way that they encourage the small entries of the vector x to be zero, so this method is sometimes called l_1 penalty method.

Most of weighted l_1-minimization techniques are based on majorization-minimization methods, so we introduce this method and we discuss about its applications to achieve different weights to our reweighted l_1-minimization problems.

The majorization-minimization(MM) method is one of the well-known optimization techniques which has lots of applications in different optimization problems [66, 64]. MM-method is an iterative technique using a majorization function, which is easier to work with, for minimizing a given function $f(x)$. Here we introduce majorization function, and then we apply the MM-method to our cardinality problem.

Definition 5.1. *(Majorization function)*

Consider the function $f : \mathbb{R} \to \mathbb{R}$, and suppose in the i-th iterations one can find a function $g_i(x)$ with the following properties,

- $g_i(x^i) = f(x^i)$. *Note that the parameter vector at the iteration $i+1$ is obtained as follows: $x^{i+1} = arg\min_x g_i(x)$.*
- $g_i(x) \geq f(x)$,
- *The minimization of $g_i(x)$ is easier than minimization of $f(x)$.*

This $g_i(x)$ is called a majorization function for $f(x)$.

The common feature of MM-algorithm, like expectation-maximization(EM) [99], and cyclic-minimization [125], is that it monotonically decreases the function's value at each iteration. Based on the definition of $g_i(x)$ above, to verify this feature for MM-method, let's see that

$$f(x^i) = g_i(x^i) \geq g_i(x^{i+1}) \geq f(x^{i+1}).$$

To apply MM-algorithm to CMP, first one should reformulate the cardinality function to a differentiable one, and then relax the concave problem with replacing its linear majorized function. To see how this procedure works, we consider the following CMP:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ Card(x) \\ &\text{s.t. } Ax = b. \end{aligned} \tag{5.3}$$

An approximation of $card(x)$ can be written as [122]

$$Card(x) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \frac{log(1 + \frac{|x_i|}{\epsilon})}{log(1 + \frac{1}{\epsilon})}, \tag{5.4}$$

with $\epsilon \geq 0$ and $x = (x_1, ..., x_n)^T$. Without loss of generality we can assume that $x \geq 0$, since if not one can write $x = x^+ - x^-$ with $x^+, x^- \geq 0$, and $card(x) = card(x^+) + card(x^-)$. Now one could reformulate the problem (5.3) as follows [34]:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \sum_{i=1}^{n} log\left(1 + \frac{x_i}{\epsilon}\right) \\ &\text{s.t. } Ax = b,\ x \geq 0. \end{aligned} \tag{5.5}$$

Note that $log(1 + \frac{x_i}{\epsilon})$ is not equivalent to the l_0-norm function, but it encourages the

x_is to be set to zero. As we will see on this chapter later, we call these kinds of functions as merit functions.

The objective function of the problem (5.5) is concave, so one can use the linear majorization of the function as a new objective which is convex. For a a given concave function f we know

$$f(x) \leq f(y) + (x-y)^T \nabla f(y), \quad \forall x, y \in dom(f).$$

The right hand side of the inequality above is a linear function with respect to x. Linearization is a famous method to minimize a concave function over convex sets [77]. The convergence of these kinds of algorithms can be shown using the global convergence theorem [90].

One could reformulate the problem (5.5) by its linearized majorization function. To do so, we write the first order approximation of the objective function, at the point x^l, as follows

$$\sum_{i=1}^{n} log\left(1+\frac{x_i}{\epsilon}\right) \approx \sum_{i=1}^{n} log\left(1+\frac{x_i^l}{\epsilon}\right) + \sum_{i=1}^{n} \frac{x_i - x_i^l}{\epsilon + x_i^l} \tag{5.6}$$

Then one could solve the achieved following convex problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \sum_{i=1}^{n} \frac{x_i - x_i^l}{\epsilon + x_i^l} \\ &\text{s.t. } Ax = b, \; x \geq 0. \end{aligned} \tag{5.7}$$

By setting $\frac{1}{\epsilon + x_i^l} = \omega_i$, we have the following weighted l_1-minimization problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \sum_{i=1}^{n} \omega_i x_i \\ &\text{s.t. } Ax = b,\ x \geq 0. \end{aligned} \tag{5.8}$$

In [34], a simple iterative algorithm is proposed to solve reweighted l_1-minimization problems. In what follows, we briefly introduce this algorithm.

Define a vector $\omega = (\omega_1, ..., \omega_n)$, where ω_is can be interpreted as penalties that discourage the existence of nonzero elements of the vector x.

Algorithm 5.1. *Set l as an index which counts the iterations, and choose a small enough ϵ.*

*Step 0: Choose a starting(initial) point x^0 for the algorithm, solving the convex optimization problem (*4.3*).*

Step 1: Set $l = 0$ and $\omega^l = (\omega_1, ..., \omega_n) = \mathbf{1}^T$.

Step 2: Solve

$$x^l = argmin\{\langle \omega^{(l)}, |x| \rangle : Ax = b\}. \tag{5.9}$$

Step 3:

$$\omega_i^{l+1} = \frac{1}{|x_i^l| + \epsilon}, \quad i = 1, ..., n.$$

Step 4: If some stopping criteria holds, stop. Otherwise ($l + 1 \mapsto l$), and go to step 2.

There are different stopping criteria for this algorithm. For example, one may stop the algorithm when there is no improvements in the optimal solution of the algorithm. Hence the algorithm may be terminated at iteration k if $x^k = x^{k-1} = x^{k-2}$. Or one may define some updating rules for the parameter ϵ, e.g. $\epsilon^{l+1} = 0.5\epsilon^l$, and terminates the algorithm when ϵ is small enough.

The above algorithm is sometimes called reweighted l_1-heuristic.

One of the most recommended choices for the starting point, x^0, is the optimal solution of the relaxed l_1-minimization problem.

Another more general reformulations of the CMP (5.3) are as follows [34]:

$$\begin{aligned} &\underset{x,R}{\text{Minimize}} \ \sum_{i=}^{n} R_i \\ &\text{s.t.} \ Ax = b \\ &\qquad |x_i| \leq R_i, \end{aligned} \tag{5.10}$$

or

$$\begin{aligned} &\underset{x,R}{\text{Minimize}} \ \sum_{i=1}^{n} log(R_i + \epsilon) \\ &\text{s.t.} \ Ax = b \\ &\qquad |x_i| \leq R_i, \end{aligned} \tag{5.11}$$

which is equivalent to

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \sum_{i=1}^{n} log(|x_i| + \epsilon) \\ &\text{s.t.} \ Ax = b, \end{aligned} \tag{5.12}$$

The problem (5.12) and (5.11) are equivalent in the following sense.

If x^* is a solution to (5.12) then $(x^*, |x^*|)$ is a solution to (5.11), and if (x^*, R^*) is a solution to (5.11), then x^* is a solution to (5.12).

Based on the above technique, the problem (5.11) can be solved by using its linearized relaxation function.

$$R^{l+1} = \text{argmin} \sum_{i=1}^{n} \frac{R_i}{R_i^l + \epsilon} \quad \text{s.t. } Ax = b, \quad |x_i| \leq R_i,\ i = 1, ..., n, \tag{5.13}$$

which is equivalent to

$$x^{l+1} = \text{argmin} \sum_{i=1}^{n} \frac{|x_i|}{|x_i^l| + \epsilon} \quad \text{s.t. } Ax = b. \tag{5.14}$$

Then the Algorithm 5.1 can be applied to the above problem, by setting

$$\omega_i^{l+1} = \frac{1}{|x_i^l| + \epsilon}, \quad i = 1, ..., n.$$

Here, we use the following reformulation for the cardinality function (as seen in chapter 3),

$$Card(x) = \|x\|_0 = \lim_{\epsilon \to 0} \sum_{i=1}^{n} sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right). \tag{5.15}$$

If we repeat the previous MM-procedure for the above function, the following weights can be obtained:

$$\omega_i = \frac{1}{x_i^2 + \epsilon^2}.$$

Now, one can easily apply the Algorithm 5.1. It seems, in this case, the algorithm should converge faster since the above weights propose larger penalties for small x_is. See Figures 5.1 and 5.2.

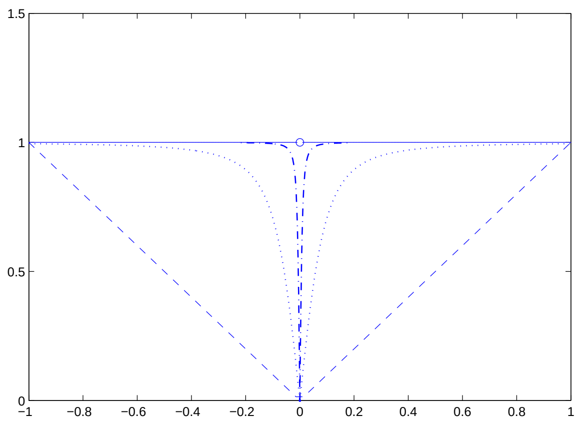

Figure 5.1: The solid graph represents $card(x)$, the dash graph represents $\|x\|_1$, as the convex envelop of $card(x)$. The dot graph represents the function $sin\left(atan\left(\frac{|x|}{\epsilon}\right)\right)$ for $\epsilon = 0.1$, and the dash-dot graph represents $sin\left(atan\left(\frac{|x|}{\epsilon}\right)\right)$ for $\epsilon = 0.01$.

For more illustration, here we present Figure 5.2, based on the l_1-minimization and the reweighted l_1-minimization for the following cardinality problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ Card(x) \\ &\text{s.t. } Ax = b, \end{aligned} \tag{5.16}$$

where $A \in \mathbb{R}^{100\times 200}$.

A, b are randomly generated from the normal distribution with mean of 0 and variance of 1, $N(0,1)$.

The dot line is presenting $card(x)$ achieved by the l_1-minimization. The solid line is presenting $card(x)$ achieved from the reweighted l_1-minimization at each iteration. The weights, in this case, are obtained by applying the linearization method to the following approximation of $card(x)$, denoted by φ_1,

$$Card(x) = \lim_{\epsilon \to 0} \sum_{i=1}^{n} \frac{log(1 + \frac{|x_i|}{\epsilon})}{log(1 + \frac{1}{\epsilon})} = \varphi_1. \tag{5.17}$$

And the dashed line is presenting $card(x)$ achieved from the reweighted l_1-minimization method at each iteration, the weights in this cased are achieved by applying linearization method based on our approximation of $card(x)$, denoted by φ_2,

$$Card(x) = \|x\|_0 = \lim_{\epsilon \to 0} \sum_{i=1}^{n} sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right) = \varphi_2. \tag{5.18}$$

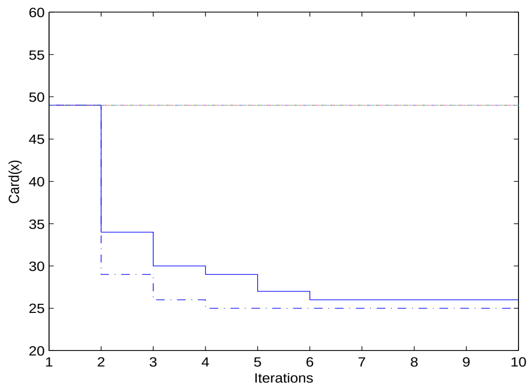

Figure 5.2: The l_1-minimization and the reweighted l_1-minimization with a matrix $A \in \mathbb{R}^{100\times 200}$. The dots line represents $card(x)$ based on the l_1-minimization. The solid line represents $card(x)$ achieved by the reweighted l_1-minimization based on φ_1 approximation, at each iteration. And the dashed line represents $card(x)$ achieved by the reweighted l_1-minimization based on φ_2 approximation, at each iteration.

In the next section, we focus on some new concave merit approximations to the function $\|x\|_0$, which lead us to new reweighted l_1-algorithms.

5.2 Concave approximations to $\|x\|_0$ and reweighted l_1-minimization

As we have seen, there are some alternatives to improve the l_1-minimization, called reweighted l_1-minimization techniques. In this section, we continue to develop the reweighted l_1-minimization algorithms.

Remind that the problem of finding the sparsest solution to the linear system, can be stated as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_0 \\ &\quad \text{s.t.}\ \ Ax = b, \end{aligned} \tag{5.19}$$

where $Ax = b$ is an undetermined linear system of equations.

The l_0-norm function is discontinuous, so the main idea for solving the problem (5.19) is to approximate the l_0-norm function by some other continuous functions which are easier to deal with. For example, the l_p-norm function$(0 < p < 1)$ is one of the approximations to the l_0-norm function. The l_p-minimization $(0 < p < 1)$ has been studied in [119, 42, 43]. Figure 5.3 represents the graph of $\sum_{i=1}^{n} |x_i|^p$, for $p = 1$, $p = 0.6$, and $p = 0.2$. Note that as p goes to zero, $\sum_{i=1}^{n} |x_i|^p$ approaches to the l_0-norm function.

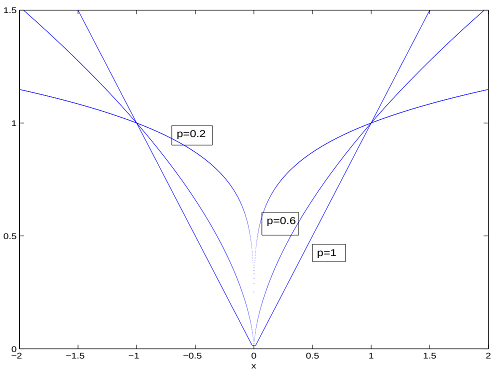

Figure 5.3: The graph of $\sum_{i=1}^{n} |x_i|^p$ for different values of $0 < p \leq 1$.

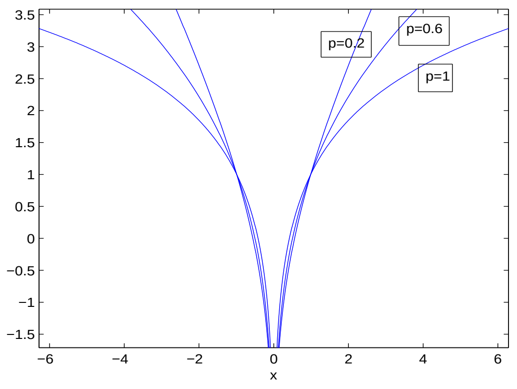

Figure 5.4: The graph of $\sum_{i=1}^{n} \log(|x_i|+\epsilon)+\sum_{i=1}^{n}(|x_i|+\epsilon)^p$ for different values of $0 < p \leq 1$.

As seen in Figure 5.3, the closest convex approximation to the $\|x\|_0$-function is the well known l_1-norm function. So it is unavoidable to use non-convex functions, especially concave functions, in order to have a better approximation to the $\|x\|_0$-function. As we will see in this chapter, these approximations will lead us to construct different reweighted l_1 algorithms through the linearization method. These reweighted algorithms perform much better than the original l_j-minimization ($j \geq 1$) techniques to find a sparse solution to a system of linear equations. In [152] Zhao and Li introduced the following function which is a combination of the l_p-norm ($0 < p < 1$) and the log function to approximate the

$\|x\|_0$-function, $x \in \mathbb{R}^n$, (see Figure 5.4):

$$F_\epsilon(x) = \sum_{i=1}^{n} \log(|x_i| + \epsilon) + \sum_{i=1}^{n} (|x_i| + \epsilon)^p, \ 0 < p < 1.$$

We will discuss these kinds of functions later. Finite successive linear approximation algorithms have also been used and are still used to get an approximate solution of the concave approximation problems [96, 117, 21, 94].

l_1-minimization methods have been used to solve the problem (5.19). This is motivated by the main idea of replacing the $\|x\|_0$-function with its convex envelop, the l_1-norm function, and then solve the resulting linear program [31, 52, 27]. Under certain conditions the l_1-minimization method is able to obtain the exact solution of the problem (5.19) for very sparse solutions of the system $Ax = b$. In the literature, several conditions have been introduced and discussed for the equivalence between the l_1-minimization and the l_0-minimization. The outstanding ones are spark [52], mutual coherence [53, 29], restricted isometry property(RIP)[31, 25, 16], and null space property(NSP) [28, 11]. For large optimization problems, the unconstrained version of the problem has been investigated in the literature, that may be referred as Lasso-type problems [137].

Numerical experiments show that weighted approaches are very affective in locating an exact solution of the problem (5.19), and it can outperform other methods, in many situations [152, 82, 34, 140, 37, 43].

Candes, Wakin, and Boyd [34] proposed a reweighted l_1-algorithm as follows:

$$\begin{aligned} &\underset{x}{\text{Minimize}} \ \sum_{i=1}^{n} \omega_i |x_i| \\ &\text{s.t. } Ax = b. \end{aligned} \tag{5.20}$$

By introducing a diagonal matrix $W = \text{diag}(\omega_1, \omega_2, ..., \omega_n)$, the problem above can be written as

$$\begin{aligned} &\underset{x}{\text{Minimize}} \; \|Wx\|_1 \\ &\text{s.t. } Ax = b. \end{aligned} \tag{5.21}$$

The weight can be interpreted as penalties for the components of the vector x. Larger penalties, (ω_i)s, apply to smaller component of the vector x, for example one may choose the weights as $\frac{1}{|x_i|}$, $i = 1, ..., n$. However to avoid having infinity penalties, one may add a parameter like $\epsilon > 0$ to define the following weight [34]

$$\omega_i = \frac{1}{|x_i| + \epsilon}, \; i = 1, ..., n.$$

Choosing the proper ϵ to have a more efficient algorithm is one of the challenges. Very small or very large ϵ might lead to improper weights which may cause the failure of the algorithms. Since very small ϵ might result in infinity penalties for small component of x, and with very big ϵ the penalty might not recognize the difference between the small components of x and the large ones. We discuss the choice of ϵ for our algorithms in the numerical experiment later.

In an iterative reweighted l_1-algorithm, weights can be defined from the iteration in the previous step. Suppose the solution at the step l is x^l, then the weight at the next step $l+1$, can be given as $\omega^{l+1} = \frac{1}{|x^l|+\epsilon}$. This was introduced by Candes, Wakin, and Boyd, and we refer to their algorithm as CWB, in this chapter.

In [152], Zhao and Li introduced a unified framework of reweighted l_1-minimization.

The main idea is to define a merit function which is a certain concave approximation of the cardinality function, and to construct different types of weights through the linearization techniques. Based on the class of merit functions defined by Zhao and Li [152], we identify several new specific merit functions, which are used to define the weights of the reweighted l_1-algorithms. The main purpose of this chapter is to study these merit functions, and to test the success probability of the reweighted algorithms associated with these merit functions for locating the sparsest solution to linear systems, where the matrices, A, are generated based on different statistical distributions. Note that most of the previous experiments in the literature use normally distributed matrices. Also, we demonstrate how the parameters used in the algorithms may affect the performance of these methods. Furthermore, we evaluate what choice of ϵ may make our algorithms work better. In section 5.3, we discuss different types of merit functions and the associated reweighted l_1-algorithms. In section 5.4, we present and discuss our numerical results, and provide comparison between these algorithms.

5.3 Merit functions and reweighted algorithms

Merit functions have been used frequently in the field of optimization. Recently Zhao and Li [152] have used merit functions to approximate the l_0-norm function. The merit function is defined as follows.

Definition 5.2. *(Merit Function)*

For any $\epsilon > 0$, a merit function $F_\epsilon(x) : \mathbb{R}^n \to \mathbb{R}$ is separable and coercive of the form $F_\epsilon(x) = \sum_{i=1}^n \psi_i(|x_i| + \epsilon)$ for approximating the l_0-norm, where the functions ψ_i are strictly concave, strictly increasing and twice differentiable. Also the function $F_\epsilon(x)$ should satisfy the following properties:

1. $\lim_{\epsilon\to 0} \frac{F_\epsilon(x)}{g(\epsilon)} = \|x\|_0 + C$, *$g(\epsilon) > 0$ is a function of ϵ, C is a constant,*

2. $F_\epsilon(x) = F_\epsilon(|x|), \quad \forall x \in \mathbb{R}^n,$

3. $\lim_{(x_i,\epsilon)\to(0,0)}[\nabla F_\epsilon(x)]_i = \infty, \quad \forall x \geq 0, \ \forall \epsilon > 0,$

4. $\lim_{\epsilon\to 0}[\nabla F_\epsilon(x)]_i = c_i, \quad \forall x_i > 0,$ *where each c_i is a positive constant.*

After replacing the $\|x\|_0$ by a merit function the problem (5.19) can be written as

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ F_\epsilon(x) \\ &\qquad \text{s.t.}\ \ Ax = b. \end{aligned} \tag{5.22}$$

Note that $F_\epsilon(x)$ is a concave function. One of the usual methods to solve concave optimization problems is to apply the linearization method, which in this case is a special type of Majorization-Minimization(MM) method. For more illustration see that by applying the Taylor expansion of $F_\epsilon(x)$ around a point u, we conclude

$$F_\epsilon(x) \leq F_\epsilon(u) + \langle \nabla F_\epsilon(u), x - u \rangle.$$

The right hand side of the inequality above is a linear function. Hence the problem (5.22) would be reduced to the following linear program:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \langle \nabla F_\epsilon(u), x \rangle \\ &\qquad \text{s.t.}\ \ Ax = b. \end{aligned} \tag{5.23}$$

So in our iterative reweighted algorithm, the iteration l should solve the following optimization problem:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \langle \nabla F_\epsilon(x^l), x\rangle \\ &\text{s.t. } Ax = b, \end{aligned} \tag{5.24}$$

where x^l is the solution of the previous iteration, and $\nabla F_\epsilon(x^l)$ are the weights. The reweighted l_1-algorithm can be defined as follows:

Algorithm 5.2.

- *Set l as an index which counts the iterations, and choose a small enough $\epsilon > 0$.*
- *Step 0: Choose a starting point x^1. This can be obtained by solving the l_1-minimization problem.*
- *Step l: Set $\omega^l = \nabla F_\epsilon(x^l)$, and Solve*

$$x^{l+1} = argmin\{\langle \omega^l, x\rangle : Ax = b\}. \tag{5.25}$$

- *Step $l+1$: If some termination criteria holds, stop. Otherwise ($l \leftarrow l+1$), and go to step l.*

An additional step can be added to the above algorithm concerning the choice of ϵ. In this chapter, our updating rule is $\epsilon_{l+1} = 0.5\epsilon_l$. In CWB algorithm, ϵ is updated as $\epsilon_{l+1} = \max\{|x^l|_{(i_0)}, 0.001\}$, where $i_0 = \frac{m}{[4\log(\frac{n}{m})]}$, and $|x|_{i_0}$ is the biggest i_0 elements of x.

It is quit challenging to prove that under a mild condition, the reweighted l_1-algorithm converges to the sparsest solution of problem (5.19). This is still an open questions in this field. However some progress have been made in this area [152, 95, 43, 140]. Mangasarian [95] introduced a successive linearization algorithm(SLA) to find the solution of general complementarity problems, and proved that SLA algorithm terminates in finite number of iterations, and creates decreasing objective function values at each iteration. Furthermore he proved these values converge to a stationary point. Chen and Zhou in [43] proved that the sequence generated by reweighted l_1-algorithm converges to a stationary point of a kind of truncated l_p-minimization problem ($0 < p < 1$). Similar results can also be found in [82]. Recently, Zhao and Li [152] defined a range space property(RSP) for matrices, under which he proved that the reweighted l_1-algorithm converges to certain sparse solution of the problem.

Following the framework of the reweighted l_1-algorithm in [152], we discuss some new merit functions. Before we go ahead, let's consider the following merit function

$$F_\epsilon(x) = \sum_{i=1}^{n} \log(|x_i| + \epsilon) + \sum_{i=1}^{n} (|x_i| + \epsilon)^p, \tag{5.26}$$

where $0 < p < 1$, which is mentioned in [152], based on which we will construct new merit functions. To verify that the above function is a merit function, one should check all the defined properties are satisfied. First, let's verify that this function is an approximation of l_0-norm function.

Indeed, it is easy to check that

$$\lim_{\epsilon \to 0} \left(n - \frac{\sum_{i=1}^{n} \log(|x_i| + \epsilon) + \sum_{i=1}^{n} (|x_i| + \epsilon)^p}{\log \epsilon} \right) = \|x\|_0.$$

Note that

$$\lim_{x \to \infty} F_\epsilon(x) = \infty,$$

which means the function is coercive. It is clear that $F_\epsilon(x) = F_\epsilon(|x|)$, and the function is increasing. In R_+^n, we have

$$\nabla F_\epsilon(x) = \left(\frac{1 + (x_1 + \epsilon)^p p}{x_1 + \epsilon}, ..., \frac{1 + (x_n + \epsilon)^p p}{x_n + \epsilon}\right)^T .$$

Also, for every $i = 1, ..., n$, we have

$$\lim_{(x_i,\epsilon)\to(0,0)} [\nabla F_\epsilon(|x|)]_i = \lim_{(x_i,\epsilon)\to(0,0)} \frac{1 + (|x_i| + \epsilon)^p p}{|x_i| + \epsilon} = \infty, \ i = 1, ..., n,$$

and $\nabla F_\epsilon(x)$ is bounded when $\epsilon \to 0$. Since $0 < p < 1$ and $x_i > 0$, we have

$$p(x_i + \epsilon)^p > p^2(x_i + \epsilon)^p, \ i = 1, ..., n,$$

so

$$\frac{-1 + (x_i + \epsilon)^p p^2 - (x_i + \epsilon)^p p}{x_i + \epsilon} < 0, \ i = 1, ..., n,$$

and hence

$$\nabla^2 F_\epsilon(x) = \text{diag}\left(\frac{-1 + (x_i + \epsilon)^p p^2 - (x_i + \epsilon)^p p}{x_i + \epsilon}\right) \prec 0, \ i = 1, ..., n.$$

As seen, in R_+^n the Hessian of the above merit function is negative definite, so the function $F_\epsilon(x)$ is strictly concave. From the above discussion one can define the following weights for the reweighted l_1-algorithm:

$$\omega_i = [\nabla F_\epsilon(|x|)]_i = \frac{1 + (|x_i| + \epsilon)^p p}{|x_i| + \epsilon}, \ i = 1, ..., n.$$

Note that the item (2) of the definition of a merit function implies that $[\nabla F_\epsilon(x^l)]_i \to \infty$ as $(x_i^l, \epsilon) \to (0, 0)$, which means larger penalties(weights) for the smaller elements of x, at each iteration.

5.3.1 New Merit Functions

Now, we start to define a new merit function as follows

$$F_\epsilon(x) = \sum_{i=1}^{n} \log(\log(|x_i| + \epsilon + (|x_i| + \epsilon)^p)). \tag{5.27}$$

We verify this function is a merit function. Clearly, this function is an approximation of l_0-norm function. Because

$$\lim_{x\to 0} \frac{\log(\log(x_i + \epsilon + (x_i + \epsilon)^p))}{\log(\log(\epsilon))} = 0, \text{ for } x_i \neq 0,$$

and

$$\lim_{x\to 0} \frac{\log(\log(x_i + \epsilon + (x_i + \epsilon)^p))}{\log(\log(\epsilon))} = 1, \text{ for } x_i = 0,$$

we conclude that

$$\lim_{x\to 0} \frac{\sum_{i=1}^{n} \log(\log(|x_i| + \epsilon + (|x_i| + \epsilon)^p))}{\log(\log(\epsilon))} = n - \|x\|_0,$$

In R_+^n, the gradient of $F_\epsilon(x)$ is given by

$$[\nabla F_\epsilon(x)]_i = \frac{1 + \frac{(x_i+\epsilon)^p p}{x_i+\epsilon}}{(x_i + \epsilon + (x_i + \epsilon)^p)(\log(x_i + \epsilon + (x_i + \epsilon)^p))},$$

and since $\lim_{x_i\to 0} x_i \log(x_i) = 0$, we have

$$\lim_{(x_i,\epsilon)\to(0,0)} [\nabla F_\epsilon(x)]_i = \infty.$$

Note that for every $x_i > 0$,

$$\lim_{\epsilon \to 0} [\nabla F_\epsilon(x)]_i = \frac{1 + p x_i^{p-1}}{(x_i + x_i^p) \log(x_i + x_i^p)} = c_i, \;\; i = 1, ..., n,$$

where c_i is positive and bounded for every $i = 1, ..., n$. Also in R_+^n, $\nabla^2 F_\epsilon(x)$ is a diagonal matrix with the following entries on its diagonal,

$$[\nabla^2 F_\epsilon(x)]_{ii} = \frac{\frac{(x_i+\epsilon)^p p^2}{(x_i+\epsilon)^2} - \frac{(x_i+\epsilon)^p p}{(x_i+\epsilon)^2}}{(x_i + \epsilon + (x_i + \epsilon)^p) \log(x_i + \epsilon + (x_i + \epsilon)^p)}$$

$$- \frac{\left(1 + \frac{(x_i+\epsilon)^p p}{x_i+\epsilon}\right)^2}{(x_i + \epsilon + (x_i + \epsilon)^p)^2 \log(x_i + \epsilon + (x_i + \epsilon)^p)}$$

$$- \frac{\left(1 + \frac{(x_i+\epsilon)^p p}{x_i+\epsilon}\right)^2}{(x_i + \epsilon + (x_i + \epsilon)^p)^2 \log (x_i + \epsilon + (x_i + \epsilon)^p)^2}, \;\; i = 1, ..., n.$$

Since, for every $i = 1, ..., n$, $[\nabla^2 F_\epsilon(x)]_{ii} < 0$, we have

$$\nabla^2 F_\epsilon(x) \prec 0.$$

So the function (5.26) is strictly concave, and it is a merit function.

Based on (5.26), the reweighted l_1-algorithm choose the following weights:

$$\omega_i = \frac{1 + \frac{(|x_i|+\epsilon)^p p}{|x_i|+\epsilon}}{(|x_i| + \epsilon + (|x_i| + \epsilon)^p)(\log(|x_i| + \epsilon + (|x_i| + \epsilon)^p))}, \;\; i = 1, ..., n.$$

In this chapter, we refer to W_1 as the reweighted algorithm with the above weights. Figure 5.17 shows the probability of success of W_1 algorithm via different choices of ϵ. This figure demonstrates that $\epsilon = 0.01$ works very good to locate the exact solution of the problem (5.19), when the sparsity is 15. Clearly the above weights are related to the parameter p, so we tested the performance of W_1 algorithm for different sparsity of the solution, i.e, $k = 5, 10, 15, 20$, via different choices of p. Thirteen different values of p have been tested (matrix A has been normally distributed), and the result is summarized in Figure 5.16. Obviously the probability of success is higher when the sparsity of the solution is lower. This can be seen in Figure 5.16.

Another new merit function can be defined as follows

$$F_\epsilon(x) = \frac{1}{p}\sum_{i=1}^{n} \left(\log(|x_i| + \epsilon + (|x_i| + \epsilon)^q)\right)^p, \tag{5.28}$$

where $0 < p, q < 1$, which is an approximation of $\|x\|_0$. In fact

$$n - \lim_{\epsilon\to 0} \frac{\frac{1}{p}\sum_{i=1}^{n} \left(\log(|x_i| + \epsilon + (|x_i| + \epsilon)^q)\right)^p}{\frac{1}{p}\left(\log(\epsilon + \epsilon^q)\right)^p} = \|x\|_0.$$

In R^n_+, the gradient of $F_\epsilon(x)$ is given by

$$[\nabla F_\epsilon(x)]_i = \frac{\log\left(x_i + \epsilon + (x_i + \epsilon)^q\right)^p \left(1 + \frac{(x_i+\epsilon)^q q}{x_i+\epsilon}\right)}{\left(x_i + \epsilon + (x_i + \epsilon)^q\right) \log\left(x_i + \epsilon + (x_i + \epsilon)^q\right)}.$$

Note that $\lim_{(x_i,\epsilon)\to(0,0)}[\nabla F_\epsilon(x)]_i = \infty$, and $\lim_{\epsilon\to 0}[\nabla F_\epsilon(x)]_i$ is bounded, for every fixed $x > 0$. In R^n_+, the Hessian is a diagonal matrix with the following diagonal elements

$$[\nabla^2 F_\epsilon(x)]_{ii} = \frac{\log\left(x_i + \epsilon + (x_i + \epsilon)^p\right)^q}{\left(x_i + \epsilon + (x_i + \epsilon)^q\right) \log\left(x_i + \epsilon + (x_i + \epsilon)^q\right)}$$

$$\cdot \left(\frac{(p-1)\left(1 + \frac{(x_i+\epsilon)^q q}{x_i+\epsilon}\right)^2}{\left(x_i + \epsilon + (x_i + \epsilon)^q\right) \log\left(x_i + \epsilon + (x_i + \epsilon)^q\right)} + \right.$$

$$\left. \frac{(x_i + \epsilon)^q q^2 - (x_i + \epsilon)^q q}{(x_i + \epsilon)^2} - \frac{\left(1 + \frac{(x_i+\epsilon)^q q}{x_i+\epsilon}\right)^2}{x_i + \epsilon + (x_i + \epsilon)^q} \right), \quad i = 1, ..., n.$$

where $0 < p, q < 1$. Clearly, $\nabla^2 F_\epsilon(x) \prec 0$, which implies the function is strictly concave. Thus, the function (5.28) is a merit function, so we may choose the following weights in our algorithm:

$$\omega_i = \frac{\log\left(|x_i| + \epsilon + (|x_i| + \epsilon)^q\right)^p \left(1 + \frac{(|x_i|+\epsilon)^q q}{|x_i|+\epsilon}\right)}{(|x_i| + \epsilon + (|x_i| + \epsilon)^q) \log\left(|x_i| + \epsilon + (|x_i| + \epsilon)^q\right)}, \quad i = 1, ..., n.$$

We refer to W_2 as the reweighted algorithm with the weights above. Figures 5.13, 5.14, 5.15 show the performance of W_2 algorithm for finding the exact solution of the problem (5.19) for different choices of the weights parameters, p and q, and for different fixed sparsity of the solution, i.e., $k = 5, 10, 15, 20$.

Remark 5.1. *We see from above that the* log *function plays a vital rule in constructing a merit function. As pointed in [152, 151], the* log *function can enhance the concavity of a given function without affecting its coercivity and monotonicity. For the convergency analysis of the reweighted l_1-algorithms based on the class of merit functions that defined at the beginning of this chapter, one may refer to the Theorems (3.9) and (3.11) in [152], where it has been shown that under the so-called RSP condition, the algorithm may converge to a solution of problem (*5.19*) with certain level of sparsity.*

5.4 Numerical experiments

In this section, we compare the performance of the l_1-minimization and the above reweighted algorithms for finding the exact solution of the problem (5.19) through the numerical tests. We compare the following problems/algorithms in our numerical experiments.
l_1-min:

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_1 \\ &\quad \text{s.t. } Ax = b, \end{aligned} \tag{5.29}$$

CWB(Candes, Wakin, Boyd):

$$\begin{aligned} x^{l+1} = \text{argmin} \ &\sum_{i=1}^{n} \frac{1}{|x_i^l| + \epsilon^l} |x_i| \\ &\text{s.t. } Ax = b, \end{aligned} \tag{5.30}$$

W_1:

$$\begin{aligned} x^{l+1} = \text{argmin} \ &\sum_{i=1}^{n} \frac{1 + \frac{(|x_i^l|+\epsilon^l)^p p}{|x_i^l|+\epsilon^l}}{(|x_i^l| + \epsilon^l + (|x_i^l| + \epsilon^l)^p)(\log(|x_i^l| + \epsilon^l + (|x_i^l| + \epsilon^l)^p))} |x_i| \\ &\text{s.t. } Ax = b, \end{aligned} \tag{5.31}$$

W_2:

$$\begin{aligned} x^{l+1} = \text{argmin} \ &\sum_{i=1}^{n} \frac{\log\left(|x_i^l| + \epsilon^l + (|x_i^l| + \epsilon^l)^q\right)^p \left(1 + \frac{(|x_i^l|+\epsilon^l)^q q}{|x_i^l|+\epsilon^l}\right)}{\left(|x_i^l| + \epsilon + (|x_i^l| + \epsilon^l)^q\right) \log\left(|x_i^l| + \epsilon^l + (|x_i^l| + \epsilon^l)^q\right)} |x_i| \\ &\text{s.t. } Ax = b, \end{aligned} \tag{5.32}$$

where l is the l^{th} iteration, $0 < p, q < 1$, $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, and $x \in \mathbb{R}^{200}$.

In our numerical works, we randomly generated the matrix $A \in \mathbb{R}^{50\times 200}$, and for a fixed sparsity, we randomly generated the solution vector $x \in \mathbb{R}^{200}$. We tested 100 randomly generated matrices, A, for different level of k-sparsity of the solution, i.e., $k = 1, 2, ..., 26$. The matrix A (the problem data) was randomly generated based on different statistical distributions. Most of the previous numerical experiments in the literature usually use normally distributed matrices.

The distributions that we considered were Normal, $(N(\mu, \sigma))$ with the parameters $\mu = 0$ and $\sigma = 1$, Poisson, $(Pois(\lambda))$ with the parameter $\lambda = 2$, Exponential, $(Exp(\mu))$ with the parameter $\mu = 5$, F-distribution, $(F(\alpha, \beta))$ with the parameters $\alpha = 1$ and $\beta = 6$, Gamma distribution, $(Gam(a, b))$ with parameters $a = 5$ and $b = 10$, and Uniform distribution, $(U(N))$ with the parameter $N = 10$. The probability of success of the 4 algorithms mentioned above, i.e, l_1-min, CWB, W_1, W_2 have been compared via different sparsity of the solution, and through all the above differently distributed matrices A. On a laptop with a Core 2 Duo CPU (2.00 GHz, 2.00GHz) and 4.00 GB of RAM memory, each comparing figure took approximately 14-hours time (in average).

The updating rule $\epsilon^{l+1} = \frac{1}{2}\epsilon^{l}$ was used, at each iteration l. The choice of ϵ is crucial for reweighted l_1-algorithms. Hence, we have also tested the algorithms by applying Candes, Wakin, Boyd(CWB) updating rule for ϵ, and also a fixed $\epsilon = 0.01$. These figures demonstrate how these choices of ϵ may affect the performance of the algorithms.

As seen, the weights in W_1 and W_2 vary for different values of p and p, q. Therefore, we have tried different choices of p and q to find out how they may affect the success probability for W_1 and W_2 algorithms.

In the Figure 5.5, the matrix A has been generated from $Exp(\mu)$, with $\mu = 5$. We set $p = 0.05$ in W_1, and $p = q = 0.05$ in W_2. As shown, all of the algorithms are very successful when $\|x\|_0 < 7$. When $7 < \|x\|_0 < 11$, CWB, W_1 and W_2 almost perform the same as each other, but when $\|x\|_0 > 11$, W_1 and W_2 outperform the CWB algorithm. All of the algorithms fail when the cardinality of the solution is above 25, i.e, $\|x\|_0 > 25$.

In Figure 5.6, the matrix A has been generated from $Exp(\mu)$, with $\mu = 5$, as in Figure 5.5. However, in this case we used different values for p and q. We chose $p = q = 0.4$, which is much larger than 0.05. As expected, both W_1 and W_2 perform significantly worse than the case of $p = q = 0.05$. Even for lower sparsity, both algorithms fail to locate the exact sparse solution with a high probability.

In Figure 5.7, the matrix A has been generated from $F(\alpha, \beta)$, with $\alpha = 1$ and $\beta = 6$, and we set $p = q = 0.05$. As shown, all of the algorithms start failing when the cardinality of the solution is higher than 4, i.e, $\|x\|_0 > 4$. W_1 and W_2 perform better than CWB for higher cardinality of the solution, and W_2 is slightly better than W_1 in general.

In Figure 5.8, the matrix A has been generated from $Gam(a, b)$, with $a = 5$ and $b = 10$, and we set $p = q = 0.05$. For lower cardinality of the solution, CWB and W_1 perform slightly better than W_2 when $\|x\|_0 < 8$. Also CWB, W_1, and l_1-min are completely successful for locating the exact solution, when $\|x\|_0 < 8$. But for $8 < \|x\|_0 < 10$, only CWB and W_1 are successful. l_1-min, CWB, W_1 and W_2 fail when $\|x\|_0 > 17$, $\|x\|_0 > 21$, $\|x\|_0 > 24$, $\|x\|_0 > 26$, respectively. Therefore W_1 and W_2 perform significantly better for higher cardinality of the solution.

In Figure 5.9, the matrix A has been generated from $N(\mu, \sigma)$, with $\mu = 0$ and $\sigma = 1$, and we set $p = q = 0.05$. As shown, l_1-min, CWB, and W_1 are very successful for finding the sparsest solution of the system when $\|x\|_0 < 8$. W_1 and W_2 perform better than the other two algorithms for higher cardinality of the solution.

In Figure 5.10 , the matrix A has been generated from $N(\mu, \sigma)$, with $\mu = 0$ and $\sigma = 1$ as in Figure 5.9. However, we chose bigger values for p and q, i.e, $p = q = 0.4$. For large values of p and q, W_2 starts failing for $\|x\|_0 > 4$, and performs much worst than W_1, CWB, and l_1-min. Also, for higher cardinality of the solution CWB performs better than W_1 and W_2. Hence, from this figure and Figure 5.6, one may conclude that smaller values for p and q should be chosen in order to achieve better results. Note that for large values of p and q the merit functions in W_1 and W_2 are not good concave approximations of l_0-norm.

In Figure 5.11, the matrix A has been generated form $U(N)$, with $N = 10$, and we set $p = q = 0.05$. All of the algorithms except W_2 are successful for finding the sparsest solution of the system when $\|x\|_0 < 9$. For $9 < \|x\|_0 < 12$, CWB performs slightly better than W_1 and W_2. But for higher cardinality of the solution, W_1 and W_2 outperform l_1-min and CWB.

In Figure 5.12, the matrix A has been generated from $Pois(\lambda)$, with $\lambda = 5$, and we set $p = q = 0.05$. All of the algorithms except W_2 are successful for finding the sparsest solution of the system when $\|x\|_0 < 9$. For higher cardinality of the solution, W_1 and W_2 outperform the other algorithms.

Clearly, for small values of p, the best algorithm is W_1 in general, i.e. for different cardinality of the solution and for different tested distributions. For all of the different tested distributions, both W_1 and W_2 (for small choices of p and q) outperform CWB when the cardinality of the solution is higher.

In Figures 5.13, 5.14, 5.15, we focused on the performance of W_2 algorithms for different values of p and q via different fixed cardinality of the solution. In Figure 5.13, we fixed $p = 0.08$ and set different values of q. We examined the probability of success of W_2 for different fixed sparsity of 5,10,15,20. As expected, when cardinality of the solution is lower the success probability of W_2 is higher. As seen, the probability of success for fixed sparsity of 5 is the highest, and the probability of success for fixed sparsity of 20 is the lowest. Figures 5.14 and 5.15 show the same results for fixed $p = 0.4$ and $p = 0.8$, respectively.

In Figure 5.16, the performance of W_1 has been tested using different choices of p and different fixed sparsity of the solution. As seen, in Figure 5.16, when p increases from 0.04 to 1, the probability of success of the algorithm becomes lower(except some jumps). As shown, for different fixed sparsity of 5,10,15,20 the highest probability of success was achieved when $p = 0.04$. Looking back to the merit function defined for the W_1 algorithm, one may see that for smaller values of p the function is a better concave approximation of l_0-norm.

As we have discussed before, the choice of ϵ for the reweighted l_1-algorithm is important. Either very small or very big ϵ may result in improper weights, which may cause the failure of the algorithms. In Figure 5.17, we fixed the sparsity of the solution ($k = 15$) and set $p = 0.05$. Different choices of ϵ have been tested to suggest what ϵ might be the good one for which W_1 performs better. The matrix A has been generated from $N(0, 1)$. As shown, when ϵ tends to zero (e.g. $\epsilon \approx 0.0001$), or when ϵ is big (e.g. $\epsilon \approx 0.1$), the probability of success decreases. Our numerical experiments, in Figure 5.17, show that $\epsilon = 0.01$ is a good choice for the weights in W_1 algorithm.

In Figure 5.18, we fixed $\epsilon = 0.01$(with no updating rule) and compared the performance of l_1-min, CWB, W_1, W_2. Like Figure 5.9, the matrix has been generated from $N(0,1)$, and we set $p = q = 0.05$. Our numerical experiment show that W_1 and W_2 significantly outperform CWB especially for higher cardinality of the solution. Comparing Figure 5.18 and Figure 5.9, one may conclude that even for a fixed ϵ, if chosen correctly, both W_1 and W_2 algorithms may perform very well to find a sparse solution.

Again to show that how important the choice of ϵ is, we compare the performance of l_1-min, CWB, W_1, W_2 based on the Candes updating rule. As seen, CWB outperforms both W_1 and W_2 for lower cardinality of the solution, i.e, when $\|x\|_0 < 12$, in our numerical experiments.

In the next chapter, we discuss different methods for finding the restricted isometry constant(RIC). Since the problem of finding the RIC is a problem with cardinality constraints, we will use different approximation techniques to solve the problem.

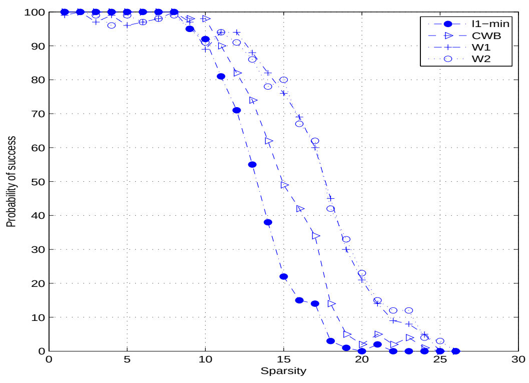

Figure 5.5: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from Exponential distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

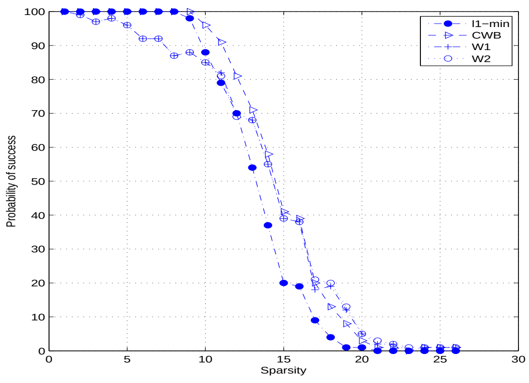

Figure 5.6: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.4$. Matrix A has been generated from Exponential distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

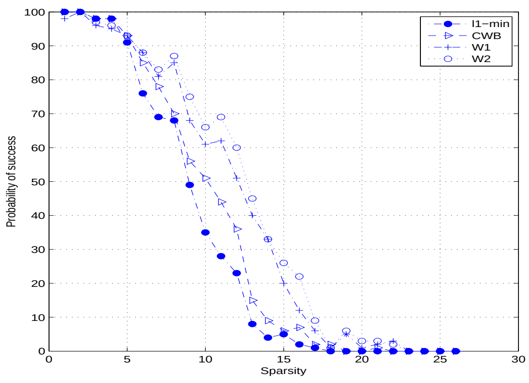

Figure 5.7: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from F-distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

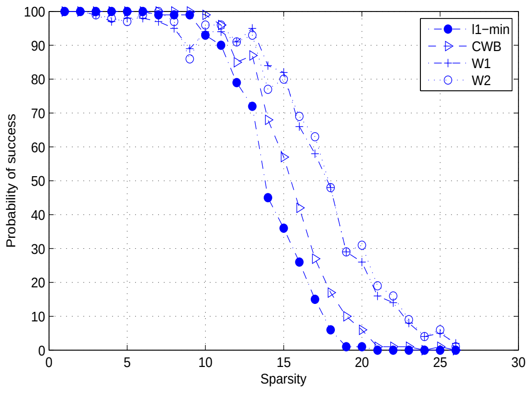

Figure 5.8: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from Gamma distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

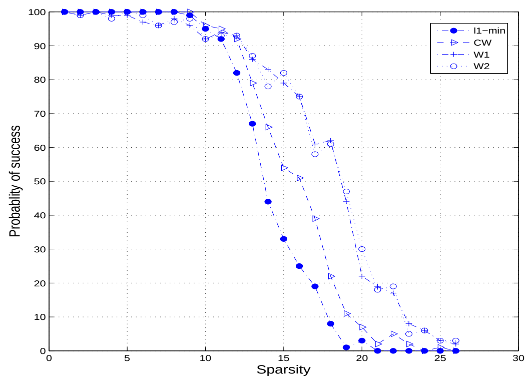

Figure 5.9: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from Normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

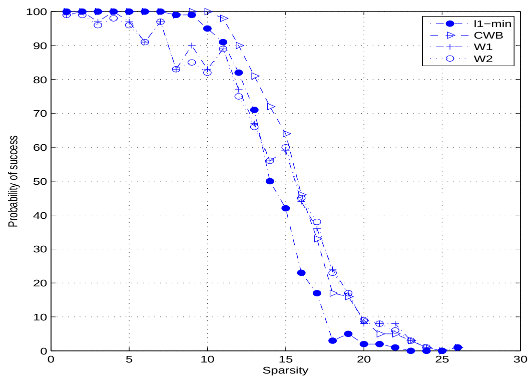

Figure 5.10: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.4$. Matrix A has been generated from Normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

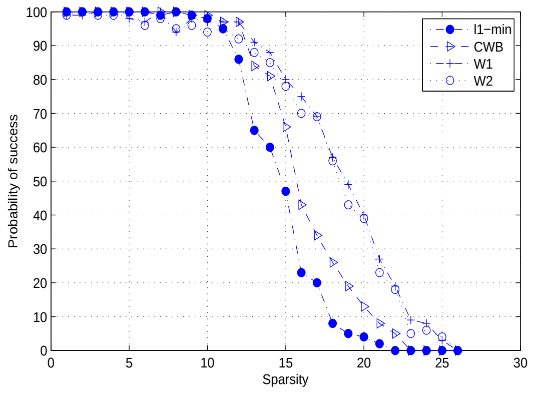

Figure 5.11: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from Uniform distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

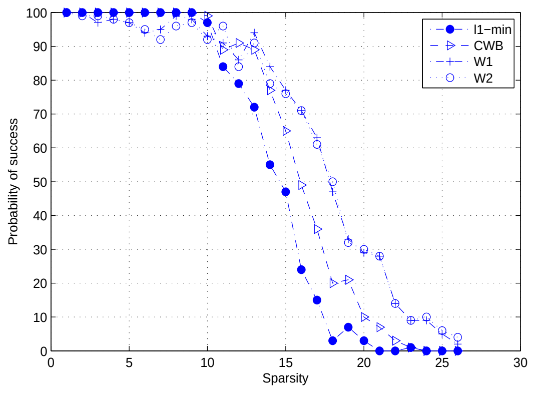

Figure 5.12: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from Poisson distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

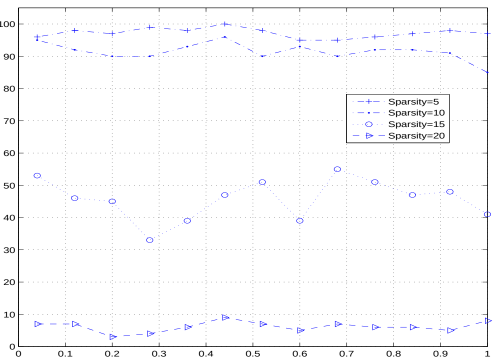

Figure 5.13: Comparing the performance of W_2 minimization for different $q = 0.04 : 0.08 : 1$, $p = 0.08$ via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$. Matrix A has been generated from normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 5, 10, 15, 20$.

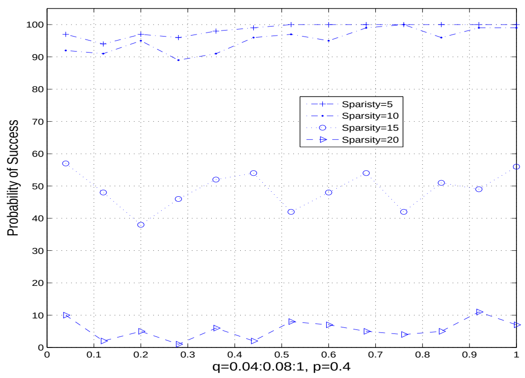

Figure 5.14: Comparing the performance of W_2 minimization for different $q = 0.04 : 0.08 : 1$, $p = 0.4$ via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$. Matrix A has been generated from normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 5, 10, 15, 20$.

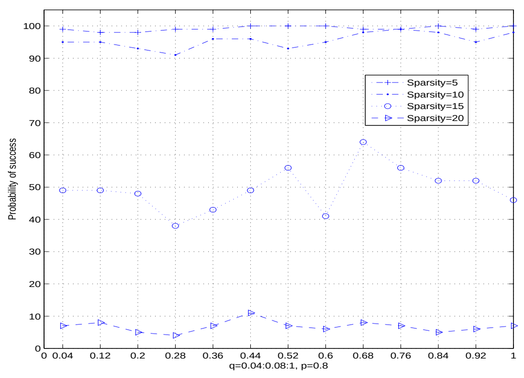

Figure 5.15: Comparing the performance of W_2 minimization for different $q = 0.04 : 0.08 : 1$, $p = 0.8$ via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$. Matrix A has been generated from normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 5, 10, 15, 20$.

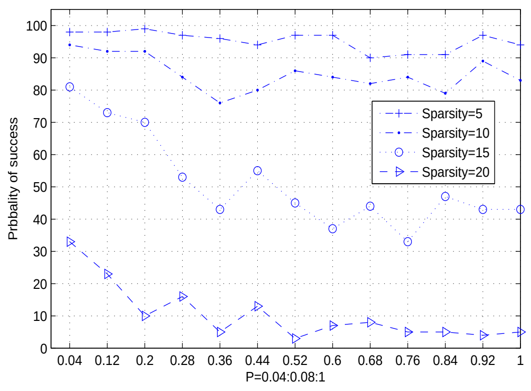

Figure 5.16: Comparing the performance of W_1 minimization for different $p = 0.04 : 0.08 : 1$ via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$. Matrix A has been generated from normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 5, 10, 15, 20$.

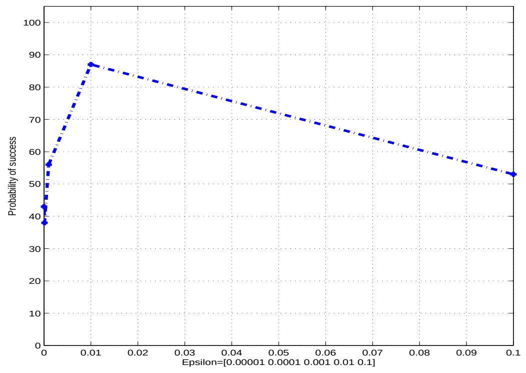

Figure 5.17: Comparing the performance of W_1 minimization using different $\epsilon = 0.00001, 0.0001, 0.001, 0.01, 0.1$ via the probability of success for finding the exact $k = 15$-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = 0.05$. Matrix A has been generated from Normal distribution. 100 randomly generated matrices have been tested for different chosen epsilons.

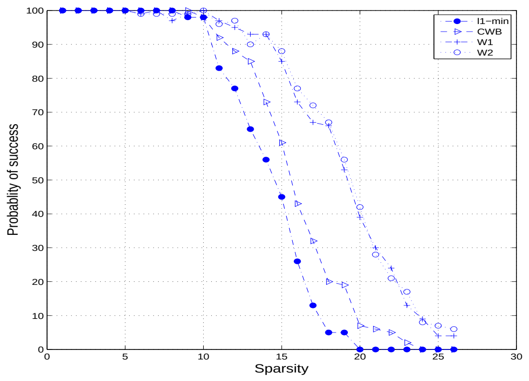

Figure 5.18: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization using fixed $\epsilon = 0.01$ via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from Normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

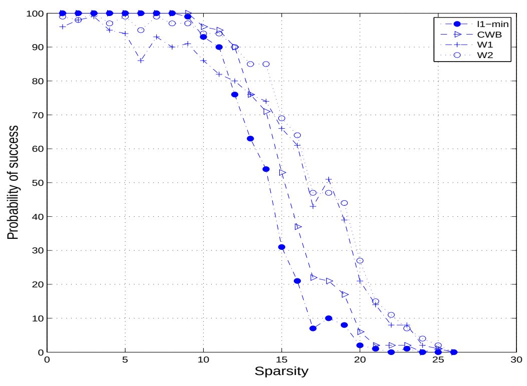

Figure 5.19: Comparing the performance of l_1-min, CWB, W_1, W_2 minimization using Candes updating rule via the probability of success for finding the exact k-sparse solution of $Ax = b$, where $A \in \mathbb{R}^{50\times 200}$, $b \in \mathbb{R}^{50}$, $p = q = 0.05$. Matrix A has been generated from Normal distribution. 100 randomly generated matrices have been tested for different sparsity of $k = 1, ..., 26$.

Chapter 6

RIP Constant via Cardinality Constrained Problems

6.1 An introduction to compressed sensing

The main focus of compressed sensing is on signal and image processing.The related background for compressed sensing(CS) can be found in [33, 92, 130]. When a signal is transmitted, some information should be neglected in order to compress the signal efficiently. One of the main problems handled by compressed sensing is how to recover an unknown signal from much fewer of measurements. In other words, it tries to recover a vector with a sparser vector using an observed vector. More specifically, the sparse vector x can be recovered by solving $A_{m\times n}x = b$ $(m < n)$, where $b \in \mathbb{R}^m$ is an observed vector, and $A_{m\times n}$ is a measurement matrix.

To guarantee recovering a k sparse vector x, which satisfies the system $Ax = b$, some restrictions are applied on the matrix A, like Spark [56, 86], Mutual Coherence [57, 114], Restricted Isometry Property(RIP) [25, 10, 36], Null Space Property(NSP) [44, 81, 46], Restricted Orthogonality Constant(ROC) [102], and Range Space Property(RSP), which recently was introduced by Zhao and Li [152].

In this section, we explain Spark, Mutual Coherence and Restricted Isometry Property(RIP). Especially, we focus on RIP and the computation of Restricted Isometry Constant(RIC) [24, 17], as a problem with cardinality constraints.

We consider the case that the solution to the following cardinality problem is unique,

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ Card(x) \\ &\quad \text{s.t. } Ax = b. \end{aligned} \tag{6.1}$$

We know a convex relaxation of the problem above is

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_1 \\ &\quad \text{s.t. } Ax = b. \end{aligned} \tag{6.2}$$

One of the questions concerned in compressed sensing is finding some conditions on A, such that (6.1), and (6.2) have the same solution. Before answering this question, we review some basic concepts.

Definition 6.1. *($Spark(A)$)*

For an arbitrary matrix A with nonzero columns, $Spark(A)$ is the smallest number n such that there exists a set of n columns of A which are linearly dependent, or alternatively

$$Spark(A) = \min \left\{ \|x\|_0 : Ax = 0, x \neq 0 \right\}.$$

The following theorem illustrates that $spark(A)$ provides an upper bound for the cardinality minimization problem.

Theorem 6.1. *[53] Vector x is the unique solution of the cardinality problem (6.1) if*

$Ax = b$, and

$$\|x\|_0 \leq \frac{Spark(A)}{2}.$$

Proof. Assume that there exist a vector x, such that $Ax = b$, and satisfies $\|x\|_0 \leq \frac{Spark(A)}{2}$, and suppose y is another solution to the system $Ay = b$, such that $\|y\|_0 \leq \|x\|_0$. Clearly, we have $A(x - y) = 0$, so $(x - y) \in null(A)$, which implies that

$$\|x - y\|_0 \geq Spark(A). \tag{6.3}$$

But from the assumption $\|y\|_0 \leq \|x\|_0$, we have

$$2\|x\|_0 \geq \|x\|_0 + \|y\|_0.$$

Since $\|x\|_0 + \|y\|_0 \geq \|x - y\|_0$, and $2\|x\|_0 \leq Spark(A)$, we have

$$Spark(A) > \|x - y\|_0,$$

which contradicts the inequality (6.3), so x is the only (sparsest) solution to the equation $Ax = b$. □

Note that $\frac{Spark(A)}{2}$ can be considered as an upper bound for the cardinality problem(6.1). However, $Spark(A)$ is hard to compute. Instead one could calculate an upper bound for $\|x\|_0$, using the incoherence of a matrix that is easy to compute.

Definition 6.2. *(Incoherence of a matrix)*

Suppose $A = [a_1, ..., a_n]$, where $a_i \in \mathbb{R}^m$ for $i = 1, ..., n$, are the column vectors of the matrix A, and $\|a_i\|_2 = 1$. Then The incoherence of the matrix A is defined as follows

$$M(A) = \max_{i \neq j, 1 \leq i,j \leq n} |\langle a_i, a_j \rangle|.$$

So, for a matrix with normalized columns we always have $0 \leq M(A) \leq 1$. Obviously, $M(A) = 0$ if and only if the columns of A are orthogonal, and $M(A) = 1$ if and only if there exists $i, j,\ (i \neq j)$ such that $a_i = \alpha a_j$ for some $|\alpha| = 1$. Now, we start reviewing some theorems and their proofs, since they are needed to understand the RIP.

Theorem 6.2. *[53] If $A = [a_1, ..., a_n]$ is a matrix with normalized columns, then*

$$Spark(A) \geq \frac{1}{M(A)} + 1.$$

Proof. Suppose $x^\star = (x_1^\star, ..., x_n^\star) = \text{argmin}\{\|x\|_0 : Ax = 0, \|x\|_\infty = 1\}$, hence, $Spark(A) = \|x^\star\|_0$. On the other hand, we have, $A^T A x^\star = 0$, i.e., $\sum_{j=1}^n \langle a_i, a_j \rangle x_j^\star = 0, i = 1, ..., n$.

Now, assume that $\max\{|x_i| : i = 1, ..., n\} = x_\alpha^\star = \|x^\star\|_\infty$. Therefore, we have

$$x_\alpha^\star = -\sum_{j=1, j\neq\alpha}^{n} \langle a_\alpha, a_j \rangle x_j^\star,$$

which implies that

$$1 = \|x^\star\|_\infty = |x_\alpha^\star| \leq \sum_{j=1, j\neq\alpha}^{n} |\langle a_\alpha, a_j \rangle| |x_j^\star| \leq M(A) \sum_{j=1, j\neq\alpha} |x_\alpha^\star| = M(A)(\|x^\star\|_1 - 1).$$

As a result, we obtain

$$\|x^\star\|_1 \geq \frac{1}{M(A)} + 1,$$

so,

$$\|x^\star\|_0 = Spark(A) \geq \frac{1}{M(A)} + 1, \text{ since } \|x^\star\|_0 \geq \|x^\star\|_1.$$

□

Theorem 6.3. *[53] If $x^\star$ is a vector that satisfies $Ax = b$, and*

$$\|x^\star\|_0 \leq \frac{1}{2}\left(1 + \frac{1}{M(A)}\right),$$

then $x^\star$ is the unique solution to the problem (6.1).

Proof. This follows form the Theorem 6.1, and the fact that $\frac{Spark(A)}{2} \geq \frac{1}{2}\left(1 + \frac{1}{M(A)}\right)$. $\square$

In the numerical experiment in section 6.4, we showed that l_1-minimization may recover the exact solution for much higher sparsity level than that indicated by mutual coherence.

The following theorem provides a sufficient condition for sparse recovery.

Theorem 6.4. *[116] Suppose $\Omega = supp(x^*)$, and the matrix A_Ω is defined as $A_\Omega = [a_i]_{i\in\Omega}$. Also suppose $x^\star$ is the unique solution to the problem* (6.1). *If*

$$\|A_\Omega^\dagger A_{\Omega^c}\|_1 \leq 1,$$

*then x^* is also a unique solution to the problem (*6.2*).*

Note that the support of a vector x is defined as: $supp(x) = \{i; x_i \neq 0\}$. Recall that for a matrix $A_{m\times n}$,

$$\|A\|_1 := \max_{1\leq j\leq n} \sum_{i=1}^{m} |a_{ij}| = \left\{\max \frac{\|Ax\|_1}{\|x\|_1} : x \neq 0\right\} = \max_{1\leq i\leq n} \|a_i\|_1.$$

Proof. Assume

$$\exists\ x = (x_\Omega, x_{\Omega^c})^T \text{ s.t. } b = Ax = A_\Omega x_\Omega + A_{\Omega^c} x_{\Omega^c}.$$

So,

$$x_\Omega = A_\Omega^\dagger b - A_\Omega^\dagger A_{\Omega^c} x_{\Omega^c}$$

$$\Rightarrow x_\Omega = (A_\Omega^T A_\Omega)^{-1} A_\Omega^T b - (A_\Omega^T A_\Omega)^{-1} A_\Omega^T A_{\Omega^c} x_{\Omega^c}.$$

Since $A_\Omega x^\star_\Omega = b$, we have

$$x^\star_\Omega = A^\dagger_\Omega b = (A^T_\Omega A_\Omega)^{-1} A^T_\Omega b,$$

and

$$x^\star_\Omega - x_\Omega = (A^T_\Omega A_\Omega)^{-1} A^T_\Omega b - (A^T_\Omega A_\Omega)^{-1} A^T_\Omega b + (A^T_\Omega A_\Omega)^{-1} A^T_\Omega A_{\Omega^c} x_{\Omega^c} = (A^T_\Omega A_\Omega)^{-1} A^T_\Omega A_{\Omega^c} x_{\Omega^c}.$$

Applying the assumption in the theorem, i.e., $\|A^\dagger_\Omega A_{\Omega^c}\|_1 \leq 1$, we have

$$\|x^\star_\Omega - x_\Omega\|_1 = \|(A^T_\Omega A_\Omega)^{-1} A^T_\Omega A_{\Omega^c}\|_1 \|x_{\Omega^c}\|_1 \leq \|x_{\Omega^c}\|_1.$$

Therefore,

$$\|x^\star\|_1 = \|x^\star_\Omega\|_1 \leq \|x\|_1.$$

□

6.2 Restricted isometry property(RIP)

As we mentioned in the previous section, with a given matrix A, and a vector b, compressed sensing methods try to recover a k-sparse optimal vector x, which is a solution to

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|x\|_0 \\ &\quad \text{s.t.}\ \ Ax = b. \end{aligned} \tag{6.4}$$

It is not hard to see that the problem above is closely related to

$$\begin{aligned} &\underset{x}{\text{Minimize}}\ \|Ax - b\|_2 \\ &\text{s.t. } \|x\|_0 \le k, \end{aligned} \tag{6.5}$$

where k is a constant, see Figure 6.1. The problem (6.5) is called regressor selection problem [6, 147, 89].

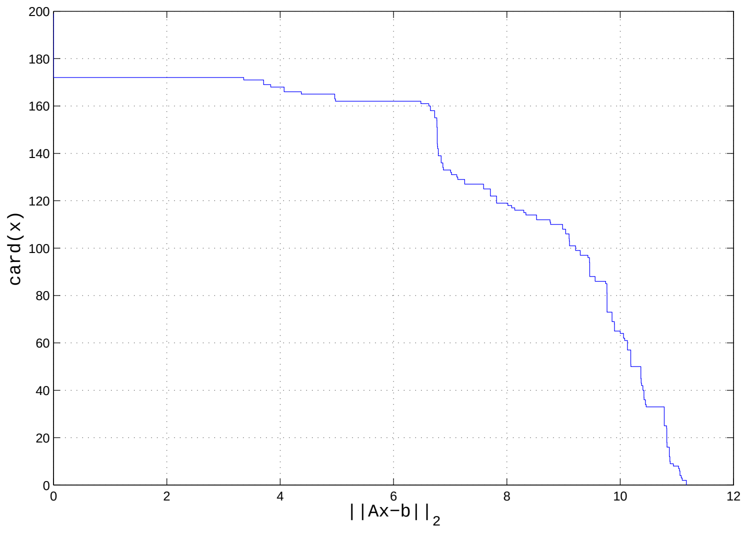

Figure 6.1: Trade-off graph between $\|Ax - b\|_2$, and $card(x)$, $A \in \mathbb{R}^{100\times 200}$.

The above reformulation is a motivation to proceed with Restricted Isometry Property(RIP).

Definition 6.3. *(Restricted Isometry Constant) The restricted isometry constant(RIC) δ_k of a matrix A is the smallest nonnegative number such that [143],*

$$(1 - \delta_k)\|x\|_2^2 \le \|Ax\|_2^2 \le (1 + \delta_k)\|x\|_2^2, \quad \forall x \in \Sigma_k, \tag{6.6}$$

where $\Sigma_k := \{x \in \mathbb{R}^n : \|x\|_0 = |supp(x)| \le k\}$.

If the inequity above holds, we say that the matrix A satisfies the restricted isometry property(RIP). As in [16], for a given matrix $A_{m\times n}$, $x \in \mathbb{R}^n$, and an observed vector $b \in \mathbb{R}^m$, one may rewrite the RIP inequity in the form of

$$R(k, m, n) := \min_{\delta \geq 0} \delta \text{ subject to } (1-\delta)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1+\delta)\|x\|_2^2,$$

$$\forall x \in \mathbb{R}^n \ s.t. \ \|x\|_0 \leq k.$$

In the following section, we show how the problem of finding the RIC is closely related to the problems with cardinality constraints. Furthermore, we provide different methods to solve these problems.

6.3 Computing RIC

The following lemma illustrates how to calculate an RIC, for a given matrix.

Lemma 6.1. *δ_k in the inequality (6.6) is the maximum of $|\lambda - 1|$ over all λ, eigenvalues of $A_\beta^T A_\beta$, $|\beta| \leq k$, where $A_\beta := [a_i]_{i\in\beta}$, and a_i is the i^{th} column of the matrix A [16].*

Proof. For any β, such that $|\beta| \leq k$ we have

$$(1-\delta_k)x_\beta^T x_\beta \leq x_\beta^T A_\beta^T A_\beta x_\beta \leq (1+\delta_k)x_\beta^T x_\beta,$$

which implies that

$$(1-\delta_k) \leq \frac{x_\beta^T A_\beta^T A_\beta x_\beta}{x_\beta^T x_\beta} \leq (1+\delta_k).$$

Applying variational characterization of eigenvalues [12], we have

$$(1-\delta_k) \leq \lambda_{min}(A_\beta^T A_\beta) \leq \lambda_{max}(A_\beta^T A_\beta) \leq (1+\delta_k),$$

equivalently,

$$-\delta_k \leq \frac{x_\beta^T(A_\beta^T A_\beta - I)x_\beta}{x_\beta^T x_\beta} \leq \delta_k,$$

which implies that

$$\begin{aligned}\delta_k &= \max_{\beta:\beta=|k|} \sup_{x_\beta} \frac{|x_\beta^T(A_\beta^T A_\beta - I)x_\beta|}{|x_\beta^T x_\beta|} \\ &= \max_{\beta:|\beta|=k} \gamma(A_\beta^T A_\beta - I).\end{aligned}$$

□

Problem of finding RIC, i.e., δ_k above, can be solved by applying principal component analysis(PCA) techniques [7, 1]. Principal Component Analysis(PCA) has a wide range of applications in data analysis and dimensionality reduction. It is a way for identifying pattern in data and expressing data such that one can highlight their similarities and differences. In other words, it compresses the data by reducing the dimensions of data, without loss of much information. As we know, PCA is an optimization problem on eigenvalues. The general optimization problem to find a maximum eigenvalue for a matrix ψ can be cast as

$$(6.7)\qquad \begin{aligned}\lambda_{max}(\psi) = \underset{x}{\text{Maximize}}\ & x^T(\psi)x \\ \text{s.t. }\ & \|x\|_2 = 1.\end{aligned}$$

A special case of the problem above is the sparse eigenvalue problem as follows:

$$(6.8)\qquad \begin{aligned}(1 + \delta_k^{max}) = \underset{x}{\text{Maximize}}\ & x^T(A_\beta^T A_\beta)x \\ \text{s.t. }\ & card(x) \leq \beta \\ & \|x\|_2 = 1.\end{aligned}$$

One can recognize that the problem above is directly related to the problem of finding RIC. This problem is NP hard, since it has a cardinality constraint. Due to the NP-hardness of the problem above, one of the main challenges in compressed sensing is to find RIC constant.

Sparse eigenvalue problem (6.8) is used frequently in Principle Component Analysis(PCA) [80]. There are lots of free software which can be used to solve a sparse eigenvalue problem [74]. Here we give more details to solve the problem (6.8). The first approach is to relax the problem, and the second one is to apply smoothing techniques, and the third is to apply d.c. programming methods.

6.3.1 Relaxation techniques

In fact, (6.8) is a PCA problem in the following sense.

Given a covariance matrix $A_\beta^T A_\beta = B \in S^n$, PCA is the problem of finding a sparse factor which describes the maximum amount of the data variance.

We apply Shor's complement for the problem (6.8) by replacing $X = xx^T$. Also with $B = A_\beta^T A_\beta$, the problem (6.8) will be [47]:

$$\begin{aligned} \underset{X}{\text{Maximize}}\ & tr(BX) \\ \text{s.t.}\ & tr(X) = 1 \\ & Card(X) \leq \beta^2 \\ & X \succeq 0 \\ & Rank(X) = 1. \end{aligned} \tag{6.9}$$

The objective $tr(BX)$ is an affine function, which is convex. The non-convex constraint $\|x\|_2 = 1$ was relaxed by replacing $tr(X) = 1$, which is linear and so a convex constraint. The problem above is still an NP-hard problem because of the non-convex constraints

$card(X) \leq \beta^2$ and $Rank(X) = 1$.

$Card(x) \leq \beta^2$ can be relaxed by a weaker but convex constraint, $\mathbf{1}^T|X|\mathbf{1} \leq \beta tr(X)$. Now, if we drop the rank constraint, the problem will be a convex one, as follows:

$$\begin{aligned} \underset{X}{\text{Maximize}}\ & tr(BX) \\ \text{s.t. } & tr(X) = 1 \\ & \mathbf{1}^T|X|\mathbf{1} \leq \beta tr(X) \\ & X \succeq 0. \end{aligned} \tag{6.10}$$

Dropping the rank constraint may result in a large gap between the optimal solution of (6.8) and (6.10). As we have seen before, the convex envelop of $rank(X)$ on the set $\{X \in \mathbb{R}^{m\times n} : \|X\| \leq M\}$ is $\frac{1}{M}\|X\|_*$, for large M enough. Hence one can relax the rank constraint by replacing it by its convex envelop, to get the following SDP:

$$\begin{aligned} \underset{X}{\text{Maximize}}\ & tr(BX) \\ \text{s.t. } & tr(X) = 1 \\ & \mathbf{1}^T|X|\mathbf{1} \leq \beta tr(X) \\ & \|X\|_* \leq M \\ & X \succeq 0. \end{aligned} \tag{6.11}$$

The above problem is a semidefinite program (linear objective with LMI constraints) which can be solved efficiently using interior point methods with the solvers such as CVX, which has two packages in it, Sedumi, and SDPT3 [73].

The optimal value of the semidefinite program (6.10) is an upper bound for the problem of sparse eigenvalue (6.9) and (6.8). Similarly we can get a lower bound on sparse minimum eigenvectors. A lower bound on δ_k can be computed using approximate sparse eigenvectors (block squares). Next, we discuss a smoothing technique to solve the problem (6.8).

6.3.2 Smoothing techniques

As we have seen, finding the RIP constant (δ_k) is reduced to a PCA problem, for which provided the semidefinite relaxation (6.10). Note that the problem (6.10) can also be cast as follows [148]:

$$\begin{aligned} \underset{X}{\text{Maximize}}\ & tr(BX) \\ \text{s.t.}\ & tr(X) = 1 \\ & \mathbf{1}^T|X|\mathbf{1} \leq \beta \\ & X \succeq 0. \end{aligned} \tag{6.12}$$

Using penalty methods, the above problem can be written as

$$\begin{aligned} \underset{X}{\text{Maximize}}\ & tr(BX) - \kappa\mathbf{1}^T|X|\mathbf{1} \\ \text{s.t.}\ & tr(X) = 1 \\ & X \succeq 0, \end{aligned} \tag{6.13}$$

where $\kappa > 0$ is the penalty parameter.

The dual of the above problem is equal to the following maximum eigenvalue problem:

$$\begin{aligned} \underset{Y}{\text{Minimize}}\ & \lambda_{max}(B+Y) \\ \text{s.t.}\ & |Y_{ij}| \leq \kappa\ i,j = 1,...,n, \end{aligned} \tag{6.14}$$

where $Y \in \mathbf{S}^n$ is the dual variable. To obtain the dual problem above, we note that the problem (6.13) can be cast as

$$
\begin{aligned}
\text{Max Min} \;\; & tr(X(B+Y)) \\
\text{s.t.} \;\; & tr(X) = 1 \\
& X \succeq 0 \\
& |Y_{ij}| \leq \kappa \;\; i,j = 1, ..., n.
\end{aligned} \tag{6.15}
$$

For the min-max structure one can use prox function algorithms in [107, 104]. We briefly explain this special problem.

If the problem has min-max model, the problem can be solved with two main steps as follows:

- Regularization: Add strongly convex penalty inside the min-max representation to produce an ϵ-approximation of the objective function with Lipschitz continuous gradient. It can be considered as generalized Moreau-Yosida regularization step [88].
- Optimal first order minimization: Using optimal first order scheme for Lipschitz continuous functions as in [106], to solve the regularized problem.

Note that the KKT conditions for the problems above are as follows [45]:

$$
\begin{cases}
\lambda_{max}(B+Y)X = (B+Y)X, \\
Y \circ X = -\kappa |X|, \\
tr(X) = 1, \\
X \succeq 0, \\
|Y_{ij}| \leq \kappa \;\; i,j = 1, ..., n.
\end{cases} \tag{6.16}
$$

If $\lambda_{max}(B+Y)$ has the multiplicity equal to one, then X is a rank one matrix. If $\lambda_{max}(B+Y)$ has the multiplicity greater than one then one can truncate the matrix X to get a dominant eigenvector as an approximate solution. For more details about this procedure one can refer to [87, 4].

To solve problem (6.14), note that λ_{max} is a non-smooth function. So it can be solved using interior point methods or by general methods of convex optimization, (see chapter 7 of [110]).

In [108] the author considered a smoothing technique which is similar to the method has been used in [107]. Here we give an outline to solve the problem (6.14) using smoothing techniques.

Put

$$f(Y) = \lambda_{max}(Y+B),$$

and replace $f(Y)$ with its smooth approximation as in [108],

$$f_\gamma(Y) = \gamma E(\frac{1}{\gamma}(Y+B)), \quad \gamma > 0,$$

where

$$E(X) = ln \sum_{i=1}^{n} e^{\lambda_i(X)}.$$

So,

$$f_\gamma(Y) = \gamma ln \left(\sum_{i=1}^{n} e^{\frac{\lambda_i(B+Y)}{\gamma}} \right).$$

Note that

$$f_\gamma(Y) \geq \lambda_{max}(B+Y),$$

and

$$f_\gamma(Y) \leq \lambda_{max}(Y+B) + \gamma ln(n).$$

Combining these inequalities gives

$$\lambda_{max}(Y+B) \leq f_\gamma(Y) \leq \lambda_{max}(Y+B) + \gamma ln(n).$$

Since $(B+Y) \in \mathbf{S}^n$, the eigenvalue decomposition of $(B+Y)$ is

$$B+Y = Udiag(\lambda)U^T, \text{ with } \lambda \in \mathbb{R}^n, \text{ and } UU^T = I.$$

The gradients of the smooth approximate function $f_\gamma(Y)$ is

$$\nabla f_\gamma(Y) = \left(\sum_{i=1}^{n} exp\left(\frac{\lambda_i(B+Y)}{\gamma}\right)\right)^{-1} \cdot \left(\sum_{i=1}^{n} exp\left(\frac{\lambda_i(B+Y)}{\gamma}\right) u_i u_i^T\right),$$

where u_i's are the components of the eigenvector U. Since $exp(\lambda_i\left(\frac{(B+Y)}{\gamma}\right)$ decreases very fast, the gradient above depends on the few largest eigenvalues.

So, the problem (6.14) will be reduced to the following problem:

$$\text{(6.17)} \qquad \begin{aligned} &\underset{Y}{\text{Minimize}}\ f_\gamma(Y) \\ &\qquad \text{s.t. } Y \in Q = \{Y \in \mathbf{S}^n : |Y_{ij}| \leq \kappa\}. \end{aligned}$$

Now, referring to [106], choose

$$\gamma = \frac{\epsilon}{2ln(n)}.$$

This choice of γ gives an ϵ-approximate solution to problem (6.14), i.e., with this γ, $f_\gamma(Y)$ becomes a uniform ϵ approximation of $\lambda_{max}(B+Y)$, with a Lipschitz continuous gradient with the constant $\frac{2ln(n)}{\epsilon}$.

Note that we are looking for Y^* such that

$$\lambda_{max}(B+Y^*) - \min_{Y\in Q} \lambda_{max}(B+Y) \le \epsilon.$$

Now, one should find a $\frac{1}{2}$ ϵ-approximate solution to the smooth problem. To see this, note that

$$f_\gamma(Y^*) - \min_{Y\in Q} f_\gamma(Y) \le \frac{1}{2}\,\epsilon \Rightarrow \lambda_{max}(B+Y^*) - \min_{Y\in Q} \lambda_{max}(B+Y)$$

$$\le f_\gamma(Y^*) - \min_{Y\in Q} f_\gamma(Y) + \gamma ln(n) \le \epsilon.$$

For more details about the complexity of the procedure one can refer to [108].

The algorithm to find an approximate solution to our problem using the smoothing technique above has four steps [148] as follows.

Given a matrix $B \in \mathbb{R}^n$, ϵ, and a parameter which controls sparsity, κ.

Algorithm 6.1. *For $j = 1, ..., N$:*

- *Compute $\nabla f_\gamma(Y_j)$. This step is the most expensive step.*
- *Euclidean projection steps. Both are the projections on $Q = \{Y \in \boldsymbol{S}^n : |Y_{ij} \le \kappa\}$: Find*

  $$V_j = arg \min_{V\in Q} \langle \nabla f_\gamma(Y_j), V\rangle + \frac{1}{2} L \|Y_j - V\|_F{}^2,$$

  where L is the Lipcshitz constant.

  This problem can be cast as

  $$arg \min_{\|V\|_\infty \le 1} \|V - W\|_F,$$

  with $W = Y - L^{-1}\nabla f_\gamma(Y)$ being given.

Find

$$Z_j = arg\min_{Z\in Q}\left\{\frac{L\|Z\|_F{}^2}{2} + \sum_{i=0}^{N}\frac{i+1}{2}\left(f_\gamma(Y_i) + \langle\nabla f_\gamma(Y_i), Z - U_i\rangle\right)\right\}.$$

- $$Y_{j+1} = \frac{2}{j+3}Z_j + \frac{j+1}{j+3}V_j.$$

- *When the duality gap is less than ϵ, the algorithm terminates, i.e., when*

$$\lambda_{max}(B + Y_j) - trBX_i + \mathbf{1}^T|X_i|\mathbf{1} \leq \epsilon.$$

6.3.3 D.C programming approaches

For full detailed discussion about d.c. function, and d.c. programming one may refer to chapters 3 and 5 of [76].

Consider the problem (6.8) which, by setting $A_\beta^T A_\beta = B$, can be cast as

$$\begin{aligned} &\underset{x}{\text{Maximize}}\ x^T Bx \\ &\quad \text{s.t. } x^T Cx \leq 1 \\ &\qquad \|x\|_0 \leq \beta, \end{aligned} \tag{6.18}$$

where $C = diag(1, ..., 1)$.

By adding the $\|x\|_0$-function to the objective, the problem above can be written as

$$\begin{aligned} &\underset{x}{\text{Maximize}}\ x^T Bx - \lambda\|x\|_0 \\ &\quad \text{s.t. } x^T Cx \leq 1, \end{aligned} \tag{6.19}$$

where $\lambda \gg 0$ is the penalty parameter.

Now, we may use the following merit function to replace the $\|x\|_0$-function,

$$F_\epsilon(x) = \sum_{i=1}^{n} \log(\log(|x_i| + \epsilon + (|x_i| + \epsilon)^p)).$$

Referring to [76, 122], we apply a d.c. programming approach to solve the following problem:

$$\begin{aligned} &\underset{x}{\text{Maximize}}\ x^T Bx - \lambda \sum_{i=1}^{n} \log(\log(|x_i| + \epsilon + (|x_i| + \epsilon)^p)) \\ &\text{s.t. } x^T Cx \leq 1. \end{aligned} \tag{6.20}$$

Here, we introduce a d.c. function.

Definition 6.4. *(d.c function)*

Let Ω be a convex set in $\mathbb{R}^n$. We say that a function is d.c. on Ω if it can be expressed as the difference of two convex functions on Ω, i.e., if $f(x) = f_1(x) - f_2(x)$, where f_1, f_2 are convex on Ω.

For solving a d.c. optimization problem, cutting plane method [79], or branch and bound method can be applied.

Now, in the problem (6.20) suppose that B is indefinite, i.e., it has negative eigenvalues. Assume there exists a scalar $\gamma \in \mathbb{R}$, such that $\gamma \geq -\lambda_{min}(B)$. So, the matrix $B + \gamma I_n$ is positive semidefinite. Hence, we have

$$\begin{aligned} x^T Bx &= x^T Bx + x^T \gamma I_n x - x^T \gamma I_n x \\ &= x^T (B + \gamma I_n)x - \gamma \|x\|_2^2. \end{aligned}$$

Note that $\gamma\|x\|_2^2$, and $x^T(B + \gamma I_n)x$ are convex. The problem (6.20) can be written as follows:

$$\begin{aligned}&\underset{x}{\text{Maximize}}\ x^T(B+\gamma I_n)x-\gamma\|x\|_2^2-\lambda\sum_{i=1}^{n}\log(\log(|x_i|+\epsilon+(|x_i|+\epsilon)^p))\\&\quad\text{s.t. } x^TCx\le 1,\end{aligned}\tag{6.21}$$

or

$$\begin{aligned}&\underset{x}{\text{Minimize}}\ \gamma\|x\|_2^2-x^T(B+\gamma I_n)x+\sum_{i=1}^{n}\log(\log(|x_i|+\epsilon+(|x_i|+\epsilon)^p))\\&\quad\text{s.t. } x^TCx\le 1,\end{aligned}\tag{6.22}$$

which is equivalent to the d.c. optimization problem

$$\begin{aligned}&\underset{x,y}{\text{Minimize}}\ \gamma\|x\|_2^2-\left(x^T(B+\gamma I_n)x-\lambda\sum_{i=1}^{n}\log(\log(|y_i|+\epsilon+(|y_i|+\epsilon)^p))\right)\\&\quad\text{s.t. } x^TCx\le 1\\&\qquad -y\le x\le y.\end{aligned}\tag{6.23}$$

See that the function $x^T(B+\gamma I_n)x-\lambda\sum_{i=1}^{n}\log(\log(|y_i|+\epsilon+(|y_i|+\epsilon)^p))$ is jointly convex in x and y. The problem above can be solved using augmented lagrangian method or reduced gradient method [49, 18].

The problem above can also be solved using the linear majorization technique as in the weighted l_1-minimization section. In this case, we may use the following reformulation for the $\|x\|_0$-function as follows

$$Card(x)=\|x\|_0=\lim_{\epsilon\to 0}\sum_{i=1}^{n}sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right).\tag{6.24}$$

Note that, we may apply other merit functions introduced in chapters 4 and 5 to approximate the $\|x\|_0$-function:

from the above reformulation, we have

$$\begin{aligned}
\lim_{\epsilon\to 0} sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right) &= sin(0) = 0, \quad x_i = 0,\\
\lim_{\epsilon\to 0} sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right) &= sin\left(\frac{\pi}{2}\right) = 1, \quad x_i \neq 0,
\end{aligned}$$

$$\begin{aligned}
sin\left(atan\left(\frac{|x_i|}{\epsilon}\right)\right) &\le sin\left(atan\left(\frac{|y_i|}{\epsilon}\right)\right) + \frac{1}{y_i^2+\epsilon^2} cos\left(atan\left(\frac{|y_i|}{\epsilon}\right)\right)(|x_i| - |y_i|)\\
&\le sin\left(atan\left(\frac{|y_i|}{\epsilon}\right)\right) + \frac{1}{y_i^2+\epsilon^2}(|x_i| - |y_i|), \ \ \forall\ x, y.
\end{aligned}$$

Also, we have

$$\gamma\|x\|_2^2 - x^T(B+\gamma I_n)x \le \gamma\|x\|_2^2 - y^T(B+\gamma I_n)y - 2(x-y)^T(B+\gamma I_n)y, \ \ \forall\ x, y.$$

So, the following similar iteration to the reweighted l_1-norm can be achieved.

$$x^{l+1} = \arg\min_x\{\gamma\|x\|_2^2 - 2x^T(B+\gamma I_n)x^l + \lambda\sum_{i=1}^n \frac{|x_i|}{\left(x_i^l\right)^2+\epsilon^2} : x^T Cx \le 1\},$$

where l is the number of iterations. For convergency proof of the algorithm above one can refer to [83, 122].

6.4 Numerical experiments

Here we present our numerical experiments about sparsity recovery by using the l_1-minimization. We tested 100 randomly generated matrices $A \in \mathbb{R}^{100\times 200}$ from a normal distribution of mean zero and variance 1.

For each matrix, the cardinality of 1 to 100 was tested.(The process takes 85 minutes on a laptop with 2GHz dual core CPUs).

The expected value of the mutual coherence of our randomly generated matrices is approximately equal to 0.292, i.e., $\mathbf{E}(M(A)) = 0.292$. According to Theorem 6.3, only for the $\|x\|_0 \leq \frac{1}{2}(1 + \frac{1}{0.292}) \approx 2.21$, the l_1 norm guaranteed to succeed.

However, according to our experiment the cardinality of the solution achieved from the l_1-minimization problem is almost exact (99 successful solutions has been found) when the cardinality of the solution is less than 20, i.e, for $\|x\|_0 \leq 20$.

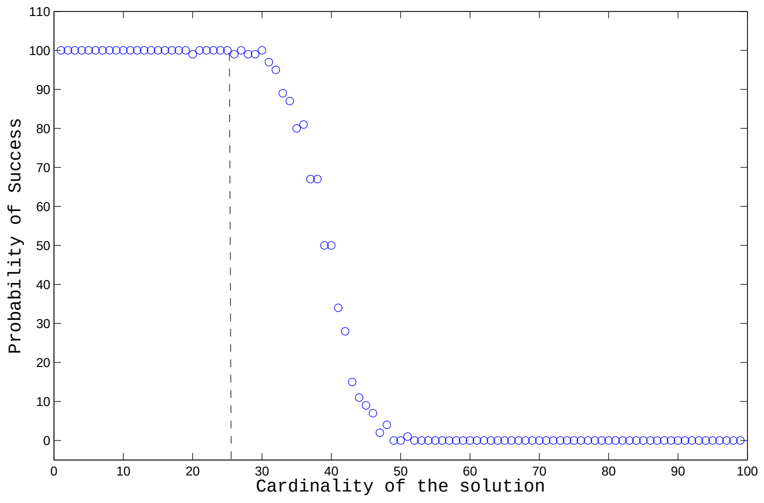

Figure 6.2: The figure shows the probability of success for the l_0 recovery using the l_1-minimization. The graph shows that the l_1-minimization is almost exact for the solutions with cardinality less than 20, i.e, 99 successful solutions has been found.

Chapter 7

Conclusions

We studied cardinality minimization problems(CMPs) and cardinality constrained problems(CCPs). Both of these problems are known as NP-hard in general, hence, we first discussed different relaxations and heuristic methods to solve these problems approximately. Different relaxation techniques were applied to reformulate the problems in different forms of semidefinite programming problems. For a general cardinality minimization problem under non-convex quadratic constraints, we relaxed the problem by applying the Lagrangian duality combined with some SDP relaxation techniques. Also, by reformulation techniques, we transformed the problem to the so-called bilevel optimization problem.

We continued our studies by focusing on a special case of the cardinality minimization problem with underdetermined linear system of equations, which is called the problem of finding the sparsest solution to a linear system. We demonstrated that reweighted l_j-algorithms ($j \geq 1$) are very efficient to find a sparse solution of such systems. We showed that the l_1-minimization already uses a hidden weighted l_2-minimization, and we presented some theoretical, geometrical and numerical results, which indicate that reweighted l_j-minimizations ($j \geq 1$) are quite successful to locate a sparse solution of the problem. Furthermore, we demonstrated that the weights may reduce the gap between different merit functions for sparsity. This means the performances of different weighted algorithms are quite similar if the weights are chosen properly.

We introduced several new concave approximations to the $\|x\|_0$-function, and showed how to construct more approximations to the $\|x\|_0$-function. We also demonstrated how these approximations can be employed to introduce new weights to reweighted l_1-algorithms. Through numerical experiments, we presented the performances of our new reweighted l_1-algorithms, and we explained how these algorithms may perform even better by changing different weights parameters. Also, we compared the new reweighted algorithms and some of the existing reweighted algorithms via different statistical distributed matrices, A, and with respect to different sparsity levels of the solution. Besides, we showed when these algorithms outperform each other in different situations. In addition, through the numerical experiments, we showed that different choices of ϵ may seriously affect the performance of the algorithms.

As a special case of cardinality constrained problems, we studied the problem of computing restricted isometry constant(RIC). We reviewed and explained how the problem of finding the restricted isometry constant(RIC) is related to a sparse eigenvalue problem, which again is a cardinality constrained problem. We studied some additional methods to solve the problem of finding the RIC approximately.

Notation

Some specific Sets

$\mathbb{R}$	Set of real matrices.
$\mathbb{R}^n$	Set of real m-vectors.
$\mathbb{R}^{m\times n}$	Set of real $m \times n$ matrices.
$\mathbb{R}_+$	Nonnegative real numbers.
$\mathbf{S}^n$	Symmetric $n \times n$ matrices.
$\mathbf{S}^n_+$	Symmetric positive semidefinite $n \times n$ matrices.
$\mathbb{L}^m$	Second order cone or ice cream.

Vectors and matrices

$\mathbf{1}$	Vector with all components 1.
I	Identity matrix.
X^T (x^T)	Transpose of a matrix X (vector x).
$X^\dagger$	Moore-Penrose or pseudo-inverse of matrix X.
$diag(x)$	Diagonal matrix with diagonal entries $x_1, ..., x_n$.
$tr(X)$	Trace of matrix X.
$\lambda_i(X)$	The i-th largest eigenvalue of symmetric matrix X.
$\sigma_i(X)$	The i-th largest singular value of matrix X.
$\lambda_{min}(X), \lambda_{max}(X)$	Minimum, maximum eigenvalue of symmetric matrix X.

$x \perp y$	Vectors x and y are orthogonal.
$U^{\perp}$	Orthogonal complement of subspace U.
$Rank(x)$	Rank of a matrix x.
$\mathcal{R}(A)$	Range of matrix A.
$\mathcal{N}(A)$	Nullspace of matrix A.
$M(A)$	Incoherence of matrix A.
$\langle x, y \rangle$	Inner products of vectors x, y.
$\langle X, Y \rangle$	Inner product of matrices X, Y.
$X \circ Y$	Hadamard product of matrices X, Y.

Norms and distances

$\|.\|$	A norm.
$\|x\|_0$ or $l_0 - norm$	Cardinality of vector x.
$\|x\|_1$	l_1-norm of vector x.
$\|x\|_2$	l_2-norm or Euclidean norm of vector x.
$\|x\|_\infty$	l_∞-norm of vector x.
$\|X\|_F$	Frobenius norm of matrix X.
$\|X\|_*$	Nuclear norm of matrix X.
$d(A, B)$	Distance between sets(or points) A and B.

Generalized inequalities

$x \preceq y$	Components wise inequalities between vectors x and y.
$X \succeq 0$	X is positive semidefinite.
$X \succ 0$	X is positive definite.
$x \preceq_{\mathbf{K}} y$	Generalized inequality induced by proper cone **K**.
$x \preceq_{\mathbf{K}^*} y$	Dual generalized inequality induced by proper cone **K**.

Topology and convex analysis

$Card(x)$	Cardinality of a vector x.
$Conv\ C$	Convex hull of set C.
$\mathbf{K}^*$	Dual cone associated with $\mathbf{K}$.
f^*	Conjugate function of f.
f^{**}	Convex envelop of function f.

Probability and Statistics

$\mathbf{E}(X)$	Expected value of a random vector X.
$N(\mu, \sigma)$	Normal distribution with mean μ and variance σ.
$Pois(\lambda)$	Poisson distribution with the parameter λ.
$Exp(\mu)$	Exponential distribution with the parameter μ.
$F(\alpha, \beta)$	F-distribution with the parameters α and β.
$Gam(a, b)$	Gamma distribution with the parameters a and b.
$U(N)$	Uniform distribution with the parameter N.

Function and derivatives

$f : A \to B$	f is a function on the set $dom(f)$ into the set B.
$dom(f)$	Domain of function f.
$epi(f)$	Epigraph of function f.
∇f	Gradient of function f.
$\nabla^2 f$	Hessian of function f.
$P_C(x)$	Standard projection operator on C.
$\partial f(x)$	Subdifferential of f at x.

Acronyms

CMP	Cardinality minimization problem.
CCP	Cardinality constrained problem.
RMP	Rank minimization problem.
CS	Compressed sensing.
PSD	Positive semidefinite cone.
SDP	Semidefinite programming.
MILP	Mixed integer linear programming.
LMI	Linear matrix inequity.
PCA	Principal component analysis.
LP	Linear Programming.
GLP	Generalized linear programming.
RIP	Restricted isometry property.
NSP	Null space property.
RIC	Restricted Isometry constant.
ROC	Restricted orthogonality constant.
RSP	Range Space Property.
SVD	Singular value decomposition.
KKT	Karush-Kuhn-Tucker.
SDr	Semidefinite representable.
MM	Majorization minimization.
BB	Branch and bound.
RLS	Reweighted Least Squares.

Bibliography

[1] H. Abdi and L. J. Williams, *Principal component analysis*, Wiley Interdisciplinary Reviews: Computational Statistics **2** (2010), no. 4, 433–459.

[2] M. J. Abdi, *Comparison of several reweighted l_1-algorithms for solving cardinality minimization problems*, University of Birmingham, Technical Report, Submitted (2013).

[3] M. J. Abdi and Y. B. Zhao, *Approximation, reformulation and convex techniques for cardinality optimization problems*, Journal of the School of Business Administration, Istanbul University **40** (2011), no. 2, 124–137.

[4] F. Alizadeh, *Interior point methods in semidefinite programming with applications to combinatorial optimization*, SIAM Journal on Optimization **5** (1995), no. 1, 13–51.

[5] T. Amemiya, *The maximum likelihood and the nonlinear three-stage least squares estimator in the general nonlinear simultaneous equation model*, Econometrica: Journal of the Econometric Society (1977), 955–968.

[6] ———, *Selection of regressors*, International Economic Review **21** (1980), no. 2, 331–354.

[7] F. Bach and L. El Ghaoui, *Optimal Solutions for Sparse Principal Component Analysis*, Journal of Machine Learning Research **9** (2008), 1269–1294.

[8] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, *Compressive wireless sensing*, Proceedings of the 5th international conference on Information processing in sensor networks, ACM, 2006, pp. 134–142.

[9] O. Banerjee, L. El Ghaoui, and A. d'Aspremont, *Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data*, The Journal of Machine Learning Research **9** (2008), 485–516.

[10] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, *A simple proof of the restricted isometry property for random matrices*, Constructive Approximation **28** (2008), no. 3, 253–263.

[11] R. G. Baraniuk, *Compressive sensing*, IEEE Signal Processing Magazine **24** (2007), no. 4, 118.

[12] A. Ben-Tal and A. S. Nemirovski, *Lectures on modern convex optimization*, (2000).

[13] D. P. Bertsekas et al., *Dynamic programming and optimal control*, (1995).

[14] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, *Convex analysis and optimization*, Athena Scientific, 2003.

[15] D.P. Bertsekas, WW Hager, and OL Mangasarian, *Nonlinear programming*, Athena Scientific Belmont, MA, 1999.

[16] J. D. Blanchard, C. Cartis, and J. Tanner, *Compressed sensing: How sharp is the restricted isometry property?*, SIAM Review **53** (2011), no. 1, 105–125.

[17] T. Blumensath and M. E. Davies, *Iterative hard thresholding for compressed sensing*, Applied and Computational Harmonic Analysis **27** (2009), no. 3, 265–274.

[18] J. F. Bonnans and C. Lemaréchal, *Numerical optimization: theoretical and practical aspects*, Springer-Verlag, 2006.

[19] S. P. Boyd, A. Ghosh, and A. Magnani, *Branch and bound methods*, Available at: http://www. stanford. edu/class/ee3920/bb. pdf, http://www. stanford. edu/class/ee3920/bb. pdf (2003).

[20] S. P. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge Univ Pr, 2004.

[21] P. S. Bradley, O. L. Mangasarian, and J. B. Rosen, *Parsimonious least norm approximation*, Computational Optimization and Applications **11** (1998), no. 1, 5–21.

[22] A. M. Bruckstein, D. L. Donoho, and M. Elad, *From sparse solutions of systems of equations to sparse modeling of signals and images*, SIAM Review **51** (2009), no. 1, 34–81.

[23] M. Bruglieri, M. Ehrgott, H. W. Hamacher, and F. Maffioli, *An annotated bibliography of combinatorial optimization problems with fixed cardinality constraints*, Discrete Applied Mathematics **154** (2006), no. 9, 1344–1357.

[24] T.T. Cai, L. Wang, and G. Xu, *New bounds for restricted isometry constants*, Information Theory, IEEE Transactions on **56** (2010), no. 9, 4388–4394.

[25] E. J. Candès, *The restricted isometry property and its implications for compressed sensing*, Comptes Rendus Mathematique **346** (2008), no. 9-10, 589–592.

[26] E. J. Candès, L. Demanet, D. Donoho, and L. Ying, *Fast discrete curvelet transforms*, Multiscale Modeling & Simulation **5** (2006), no. 3, 861–899.

[27] E. J. Candès and Y. Plan, *Near-ideal model selection by l_1 minimization*, The Annals of Statistics **37** (2009), no. 5A, 2145–2177.

[28] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Foundations of Computational mathematics **9** (2009), no. 6, 717–772.

[29] E. J. Candès and J. Romberg, *Quantitative robust uncertainty principles and optimally sparse decompositions*, Foundations of Computational Mathematics **6** (2006), no. 2, 227–254.

[30] E. J. Candès, J. K. Romberg, and T. Tao, *Stable signal recovery from incomplete and inaccurate measurements*, Communications on Pure and Applied Mathematics **59** (2006), no. 8, 1207–1223.

[31] E. J. Candès and T. Tao, *Decoding by linear programming*, Information Theory, IEEE Transactions on **51** (2005), no. 12, 4203–4215.

[32] ———, *The power of convex relaxation: Near-optimal matrix completion*, Information Theory, IEEE Transactions on **56** (2010), no. 5, 2053–2080.

[33] E. J. Candès and M. B. Wakin, *An introduction to compressive sampling*, Signal Processing Magazine, IEEE **25** (2008), no. 2, 21–30.

[34] E. J. Candès, M. B. Wakin, and S. P. Boyd, *Enhancing sparsity by reweighted 1 minimization*, Journal of Fourier Analysis and Applications **14** (2008), no. 5, 877–905.

[35] T. J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha, *Heuristics for cardinality constrained portfolio optimisation*, Computers and Operations Research **27** (2000), no. 13, 1271–1302.

[36] R. Chartrand and V. Staneva, *Restricted isometry properties and nonconvex compressive sensing*, Inverse Problems **24** (2008), no. 3, 035020.

[37] R. Chartrand and W. Yin, *Iteratively reweighted algorithms for compressive sensing*, Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on, IEEE, 2008, pp. 3869–3872.

[38] ______, *Iteratively reweighted algorithms for compressive sensing*, Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on, IEEE, 2008, pp. 3869–3872.

[39] S. Chen and D. L. Donoho, *Basis pursuit*, Signals, Systems and Computers, 1994. 1994 Conference Record of the Twenty-Eighth Asilomar Conference on, vol. 1, IEEE, 1994, pp. 41–44.

[40] ______, *Examples of basis pursuit*, SPIE's 1995 International Symposium on Optical Science, Engineering, and Instrumentation, International Society for Optics and Photonics, 1995, pp. 564–574.

[41] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM journal on scientific computing **20** (1998), no. 1, 33–61.

[42] X. Chen, F. Xu, and Y. Ye, *Lower bound theory of nonzero entries in solutions of l_2-l_p minimization*, SIAM Journal on Scientific Computing **32** (2010), no. 5, 2832–2852.

[43] X. Chen and W. Zhou, *Convergence of reweighted l_1 minimization algorithms and unique solution of truncated l_p minimization*, Technical Report, HK Polytech. Univ (2010).

[44] A. Cohen, W. Dahmen, and R. DeVore, *Compressed sensing and best k-term approximation*, American Mathematical Society **22** (2009), no. 1, 211–231.

[45] A. d'Aspremont, F. Bach, and L. E. Ghaoui, *Optimal solutions for sparse principal component analysis*, The Journal of Machine Learning Research **9** (2008), 1269–1294.

[46] A. d'Aspremont and L. El Ghaoui, *Testing the nullspace property using semidefinite programming*, Mathematical programming **127** (2011), no. 1, 123–144.

[47] A. d'Aspremont, L. E. Ghaoui, M. I. Jordan, and G. R. G. Lanckriet, *A direct formulation for sparse PCA using semidefinite programming*, Arxiv preprint cs/0406021 (2004).

[48] R. E. Davis, D. A. Kendrick, and M. Weitzman, *A Branch-and-Bound Algorithm for Zero-One Mixed Integer Programming Problems*, Operations Research **19** (1971), no. 4, 1036–1044.

[49] E. P. de Carvalho, A. dos Santos Júnior, and T. F. Ma, *Reduced gradient method combined with augmented Lagrangian and barrier for the optimal power flow problem*, Applied Mathematics and Computation **200** (2008), no. 2, 529–536.

[50] D. L. Donoho, *Compressed sensing*, Information Theory, IEEE Transactions on **52** (2006), no. 4, 1289–1306.

[51] ______, *For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution*, Communications on pure and applied mathematics **59** (2006), no. 6, 797–829.

[52] D. L. Donoho and M. Elad, *Optimally sparse representation in general (nonorthogonal) dictionaries via l_1 minimization*, Proceedings of the National Academy of Sciences **100** (2003), no. 5, 2197–2202.

[53] D. L. Donoho, M. Elad, and V. N. Temlyakov, *Stable recovery of sparse overcomplete representations in the presence of noise*, Information Theory, IEEE Transactions on **52** (2006), no. 1, 6–18.

[54] D. L. Donoho and P. B. Stark, *Uncertainty principles and signal recovery*, SIAM Journal on Applied Mathematics **49** (1989), no. 3, 906–931.

[55] D. L. Donoho, Y. Tsaig, I. Drori, and J-L Starck, *Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit*, Information Theory, IEEE Transactions on **58** (2012), no. 2, 1094–1121.

[56] M. F. Duarte and Y. C. Eldar, *Structured compressed sensing: From theory to applications*, Signal Processing, IEEE Transactions on **59** (2011), no. 9, 4053–4085.

[57] M. Elad, *Optimized projections for compressed sensing*, Signal Processing, IEEE Transactions on **55** (2007), no. 12, 5695–5702.

[58] M. Elad and A. M. Bruckstein, *A generalized uncertainty principle and sparse representation in pairs of bases*, Information Theory, IEEE Transactions on **48** (2002), no. 9, 2558–2567.

[59] Y. M. Ermol'ev, *Methods of solution of nonlinear extremal problems*, Cybernetics and Systems Analysis **2** (1966), no. 4, 1–14.

[60] M. Fazel, *Matrix rank minimization with applications*, Ph.D. thesis, Stanford. Univ, 2002.

[61] M. Fazel, H. Hindi, and S. P. Boyd, *Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices*, American Control Conference, 2003. Proceedings of the 2003, vol. 3, IEEE, 2003, pp. 2156–2162.

[62] ______, *Rank minimization and applications in system theory*, American Control Conference, 2004. Proceedings of the 2004, vol. 4, IEEE, 2004, pp. 3273–3278.

[63] M. A. Figueiredo, *Adaptive sparseness for supervised learning*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **25** (2003), no. 9, 1150–1159.

[64] M. A. T. Figueiredo, J. M. Bioucas-Dias, and R. D. Nowak, *Majorization–minimization algorithms for wavelet-based image restoration*, Image Processing, IEEE Transactions on **16** (2007), no. 12, 2980–2991.

[65] M. Fischetti and D. P. Williamson, *Integer programming and combinatorial optimization*, Springer-Verlag.

[66] C. S. Foo, C. B. Do, and A. Y. Ng, *A majorization-minimization algorithm for (multiple) hyperparameter learning*, Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 321–328.

[67] L. Gan, *Block compressed sensing of natural images*, Digital Signal Processing, 2007 15th International Conference on, IEEE, 2007, pp. 403–406.

[68] G. H. Golub and C. F. Van Loan, *An analysis of the total least squares problem*, SIAM Journal on Numerical Analysis **17** (1980), no. 6, 883–893.

[69] ———, *Matrix computations*, Johns Hopkins Univ Pr, 1996.

[70] I. F. Gorodnitsky, J. S. George, and B. D. Rao, *Neuromagnetic source imaging with focuss: a recursive weighted minimum norm algorithm*, Electroencephalography and clinical Neurophysiology **95** (1995), no. 4, 231–251.

[71] I. F. Gorodnitsky and B. D. Rao, *Sparse signal reconstruction from limited data using focuss: A re-weighted minimum norm algorithm*, Signal Processing, IEEE Transactions on **45** (1997), no. 3, 600–616.

[72] ———, *Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm*, Signal Processing, IEEE Transactions on **45** (1997), no. 3, 600–616.

[73] M. Grant, S. P. Boyd, and Y. Ye, *CVX: Matlab software for disciplined convex programming*, Web Page and Software) 2008 [Online]. Available: http://stanford.edu/˜ boyd/cvx.

[74] V. Hernandez, J. E. Roman, A. Tomas, and V. Vidal, *A survey of software for sparse eigenvalue problems*, Universidad Politécnica de Valencia, Tech. Rep. STR-6 (2006).

[75] J. B. Hiriart-Urruty and C. Lemaréchal, *Fundamentals of convex analysis*, Springer-Verlag, 2001.

[76] T. Hoang, *Convex analysis and global optimization*, Springer-Verlag, 1998.

[77] K. L. Hoffman, *A method for globally minimizing concave functions over convex sets*, Mathematical Programming **20** (1981), no. 1, 22–32.

[78] P. W. Holland and R. E. Welsch, *Robust regression using iteratively reweighted least-squares*, Communications in Statistics-Theory and Methods **6** (1977), no. 9, 813–827.

[79] R. Horst and N. V. Thoai, *DC programming: overview*, Journal of Optimization Theory and Applications **103** (1999), no. 1, 1–43.

[80] I. T. Jolliffe, *Principal component analysis*, Springer-Verlag, 2002.

[81] B. S. Kashin and V. N. Temlyakov, *A remark on compressed sensing*, Mathematical notes **82** (2007), no. 5, 748–755.

[82] M. J. Lai and J. Wang, *An unconstrained l_q minimization with $0 < q < 1$ for sparse solution of under-determined linear systems*, SIAM J. Optim **21** (2010), 82–101.

[83] Gert R Lanckriet and Bharath K Sriperumbudur, *On the convergence of the concave-convex procedure*, Advances in neural information processing systems, 2009, pp. 1759–1767.

[84] C. Lawson, *Contributions to the theory of linear least maximum approximation*, PhD Thesis (1961).

[85] C. L. Lawson and R. J. Hanson, *Solving least squares problems*, vol. 161, SIAM, 1974.

[86] K. Lee and Y. Bresler, *Computing performance guarantees for compressed sensing*, Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on, IEEE, 2008, pp. 5129–5132.

[87] Claude Lemaréchal and François Oustry, *Semidefinite relaxations and lagrangian duality with application to combinatorial optimization*, Rapports de recherche- INRIA (1999).

[88] Claude Lemaréchal and Claudia Sagastizábal, *Practical aspects of the moreau–yosida regularization: Theoretical preliminaries*, SIAM Journal on Optimization **7** (1997), no. 2, 367–385.

[89] I. Lind and L. Ljung, *Regressor selection with the analysis of variance method*, Automatica **41** (2005), no. 4, 693–700.

[90] D. G. Luenberger and Y. Ye, *Linear and nonlinear programming*, Springer-Verlag, 2008.

[91] M. Lustig, D. Donoho, and J. M. Pauly, *Sparse MRI: The application of compressed sensing for rapid MR imaging*, Magnetic Resonance in Medicine **58** (2007), no. 6, 1182–1195.

[92] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, *Compressed sensing MRI*, Signal Processing Magazine, IEEE **25** (2008), no. 2, 72–82.

[93] D. Malioutov, M. Cetin, and A. S. Willsky, *A sparse signal reconstruction perspective for source localization with sensor arrays*, Signal Processing, IEEE Transactions on **53** (2005), no. 8, 3010–3022.

[94] O. L. Mangasarian, *Machine learning via polyhedral concave minimization*, Applied Mathematics and Parallel Computing-Festschrift for Klaus Ritter (1996), 175–188.

[95] ———, *Solution of general linear complementarity problems via nondifferentiable minimization*, Acta Mathematics Vietnamical **22** (1997), no. 1, 199–205.

[96] ———, *Minimum-support solutions of polyhedral concave programs*, Optimization **45** (1999), no. 1-4, 149–162.

[97] D. Maringer and H. Kellerer, *Optimization of cardinality constrained portfolios with a hybrid local search algorithm*, OR Spectrum **25** (2003), no. 4, 481–495.

[98] I. Markovsky and S. Van Huffel, *Overview of total least-squares methods*, Signal processing **87** (2007), no. 10, 2283–2302.

[99] G. J. McLachlan and T. Krishnan, *The EM algorithm and extensions*, LibreDigital, 2008.

[100] M. Minoux and S. Vajda, *Mathematical programming: theory and algorithms*, Wiley New York, 1986.

[101] B. Moghaddam, Y. Weiss, and S. Avidan, *Spectral bounds for sparse PCA: Exact and greedy algorithms*, Advances in Neural Information Processing Systems **18** (2006), 915.

[102] K. Mohan and M. Fazel, *New Restricted Isometry results for noisy low-rank recovery*, Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on, IEEE, 2010, pp. 1573–1577.

[103] B. K. Natarajan, *Sparse approximate solutions to linear systems*, SIAM journal on computing **24** (1995), no. 2, 227–234.

[104] A. Nemirovski, *Prox-method with rate of convergence O (1/t) for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems*, SIAM Journal on Optimization **15** (2005), no. 1, 229–251.

[105] A. S. Nemirovskij and D. B. Yudin, *Problem complexity and method efficiency in optimization. Transl. from the Russian by ER Dawson*, (1983).

[106] Y. Nesterov, *A method of solving a convex programming problem with convergence rate O (1/k2)*, Soviet Mathematics Doklady, vol. 27, 1983, pp. 372–376.

[107] ———, *Smooth minimization of non-smooth functions*, Mathematical Programming **103** (2005), no. 1, 127–152.

[108] ———, *Smoothing technique and its applications in semidefinite optimization*, Mathematical Programming **110** (2007), no. 2, 245–259.

[109] Y. Nesterov and A. S. Nemirovski, *Optimization over positive semidefinite matrices: Mathematical background and user's manual*, USSR Academy Sciences. Center of Economy and Mathematics Inst, 32 Krasikova St, Moscow, 1990.

[110] Y. Nesterov and I. U. E. Nesterov, *Introductory lectures on convex optimization: A basic course*, Springer Netherlands, 2004.

[111] D. P. O'Leary, *Robust regression computation using iteratively reweighted least squares*, SIAM Journal on Matrix Analysis and Applications **11** (1990), no. 3, 466–480.

[112] B. T. Polyak, *A general method for solving extremal problems*, Sov.Math.Dokl **8** (1967).

[113] ______, *Introduction to optimization*, Optimization Software Inc. Publications Division, 1987.

[114] L. C. Potter, E. Ertin, J. T. Parker, and M. Cetin, *Sparsity and compressed sensing in radar imaging*, Proceedings of the IEEE **98** (2010), no. 6, 1006–1020.

[115] B. D. Rao and K. Kreutz-Delgado, *An affine scaling methodology for best basis selection*, Signal Processing, IEEE Transactions on **47** (1999), no. 1, 187–200.

[116] H. Rauhut, *Compressive sensing and structured random matrices*, Theoretical foundations and numerical methods for sparse recovery **9** (2010), 1–92.

[117] F. Rinaldi, F. Schoen, and M. Sciandrone, *Concave programming for minimizing the zero-norm over polyhedral sets*, Computational Optimization and Applications **46** (2010), no. 3, 467–486.

[118] J. B. Rosen, H. Park, and J. Glick, *Total least norm formulation and solution for structured problems*, SIAM Journal on Matrix Analysis and Applications **17** (1996), no. 1, 110–126.

[119] R. Saab, R. Chartrand, and O. Yilmaz, *Stable sparse approximations via nonconvex optimization*, Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on, IEEE, 2008, pp. 3885–3888.

[120] N. Z. Shor, *Nondifferentiable optimization and polynomial problems*, Kluwer Academic Publishers, 1998.

[121] N. Z. Shor, K. C. Kiwiel, and A. Ruszcayski, *Minimization methods for non-differentiable functions*, Springer-Verlag New York, Inc. New York, NY, USA, 1985.

[122] B. K. Sriperumbudur, D. A. Torres, and G. R. G. Lanckriet, *A dc programming approach to the sparse generalized eigenvalue problem*, stat **1050** (2009), 13.

[123] J-L. Starck, F. Murtagh, and J. M. Fadili, *Sparse image and signal processing: wavelets, curvelets, morphological diversity*, Cambridge University Press, 2010.

[124] Gilbert W Stewart and GW Stewart, *Introduction to matrix computations*, vol. 441, Academic press New York, 1973.

[125] P. Stoica and Y. Selén, *Cyclic minimizers, majorization techniques, and the expectation-maximization algorithm: a refresher*, Signal Processing Magazine, IEEE **21** (2004), no. 1, 112–114.

[126] F. Streichert, H. Ulmer, and A. Zell, *Evolutionary algorithms and the cardinality constrained portfolio optimization problem*, Operations research proceedings 2003: selected papers of the International Conference on Operations Research (OR 2003), Heidleberg, September 3-5, 2003, Springer Verlag, 2004, p. 253.

[127] D. Takhar, J. N. Laska, M. B. Wakin, M. F. Duarte, D. Baron, S. Sarvotham, K. F. Kelly, and R. G. Baraniuk, *A new compressive imaging camera architecture using optical-domain compression*, Electronic Imaging 2006, International Society for Optics and Photonics, 2006, pp. 606509–606509.

[128] H. L. Taylor, S. C. Banks, and J. F. McCoy, *Deconvolution with l_1 norm*, Geophysics **44** (1979), 39–52.

[129] R. Tibshirani, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society. Series B (Methodological) (1996), 267–288.

[130] M.Y.J. Ting, *Signal processing for magnetic resonance force microscopy*, Ph.D. thesis, 2006.

[131] J. A. Tropp, *Greed is good: Algorithmic results for sparse approximation*, Information Theory, IEEE Transactions on **50** (2004), no. 10, 2231–2242.

[132] J. A. Tropp and A. C. Gilbert, *Signal recovery from random measurements via orthogonal matching pursuit*, Information Theory, IEEE Transactions on **53** (2007), no. 12, 4655–4666.

[133] Y. Tsaig and D. L. Donoho, *Extensions of compressed sensing*, Signal processing **86** (2006), no. 3, 549–571.

[134] S. Van Huffel and J. Vandewalle, *The total least squares problem: computational aspects and analysis*, vol. 9, Society for Industrial and Applied Mathematics, 1987.

[135] L. Vandenberghe and S. P. Boyd, *Semidefinite programming*, SIAM Review **38** (1996), no. 1, 49–95.

[136] M. B. Wakin, J. N. Laska, M. F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. F. Kelly, and R. G. Baraniuk, *An architecture for compressive imaging*, Image Processing, 2006 IEEE International Conference on, IEEE, 2006, pp. 1273–1276.

[137] H. Wang, G. Li, and C. L. Tsai, *Regression coefficient and autoregressive order shrinkage and selection via the lasso*, Journal of the Royal Statistical Society: Series B (Statistical Methodology) **69** (2007), no. 1, 63–78.

[138] D. S. Watkins, *Fundamentals of matrix computations*, John Wiley and Sons, 2002.

[139] H. White, *Using least squares to approximate unknown regression functions*, International Economic Review **21** (1980), no. 1, 149–170.

[140] D. Wipf and S. Nagarajan, *Iterative reweighted l_1 and l_2 methods for finding sparse solutions*, IEEE Journal of Selected Topics in Signal Processing **4** (2010), no. 2, 317–329.

[141] S. Wold, K. Esbensen, and P. Geladi, *Principal component analysis*, Chemometrics and intelligent laboratory systems **2** (1987), no. 1-3, 37–52.

[142] A. Y. Yang, S. R. Rao, and Y. Ma, *Robust statistical estimation and segmentation of multiple subspaces*, Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on, IEEE, 2006, pp. 99–99.

[143] L. Ying and Y. M. Zou, *Linear Transformations and Restricted Isometry Property*, Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on, IEEE, 2009, pp. 2961–2964.

[144] G. Yu, G. Sapiro, and S. Mallat, *Image modeling and enhancement via structured sparse model selection*, Image Processing (ICIP), 2010 17th IEEE International Conference on, IEEE, 2010, pp. 1641–1644.

[145] M. Yuan and Y. Lin, *Model selection and estimation in regression with grouped variables*, Journal of the Royal Statistical Society: Series B (Statistical Methodology) **68** (2006), no. 1, 49–67.

[146] F. Zhang, *Matrix theory: basic results and techniques*, Springer-Verlag, 1999.

[147] Q. Zhang, *Regressor selection and wavelet network construction*, Decision and Control, 1993., Proceedings of the 32nd IEEE Conference on, IEEE, 1993, pp. 3688–3693.

[148] Y. Zhang, A. d'Aspremont, and L. El Ghaoui, *Sparse pca: Convex relaxations, algorithms and applications*, Handbook on Semidefinite, Conic and Polynomial Optimization, Springer, 2012, pp. 915–940.

[149] Y. B. Zhao, *An approximation theory of matrix rank minimization and its application to quadratic equations*, Linear Algebra and its Applications **437** (2012), no. 1, 77–93.

[150] Y. B. Zhao, S-C Fang, and J. E. Lavery, *Geometric dual formulation for first-derivative-based univariate cubic l_1 splines*, Journal of Global Optimization **40** (2008), no. 4, 589–621.

[151] Y. B. Zhao, S. C. Fang, and D. Li, *Constructing generalized mean functions using convex functions with regularity conditions*, SIAM Journal on Optimization **17** (2006), 37–51.

[152] Y. B. Zhao and D. Li, *Reweighted l_1-minimization for sparse solutions to underdetermined linear systems*, SIAM Journal on Optimization **22** (2012), no. 3, 1065–1088.

[153] X. Zheng, X. Sun, D. Li, and X. Cui, *Lagrangian decomposition and mixed-integer quadratic programming reformulations for probabilistically constrained quadratic programs*, European Journal of Operational Research (2012).

[154] H. Zou, T. Hastie, and R. Tibshirani, *Sparse principal component analysis*, Journal of computational and graphical statistics **15** (2006), no. 2, 265–286.