Iterative Algorithms in Compressive Sensing

Kristoffer Ulvik Høisæther

Master's Thesis, Spring 2018

Abstract

The field of compressive sensing is a modern field in applied mathematics which receives a lot of attention. In this thesis, we will give some insight into the iterative algorithms used in compressive sensing. We will study in particular the primal-dual algorithm, as proposed by Chambolle and Pock, and Nesterov's algorithm, NESTA. In general, the primal-dual algorithm is a more traditional algorithm than NESTA. Nesterov proved that for general convex functions, the primal-dual algorithm cannot achieve a better convergence rate than $\mathcal{O}(1/k)$, where k is the number of iterations, whereas Nesterov's algorithm with general convex functions achieves a convergence rate of $\mathcal{O}(1/k^2)$.

Acknowledgements

First and foremost, I would like to thank my supervisors Øyvind Ryan and Anders Hansen. A special thanks to Øyvind, who has guided me through the field of compressive sensing and helped me with all the challenges the thesis has thrown at me. I am truly grateful for all your help.

During the last semester, I have shared study hall with Anders Lindstrøm, Edvard Aksnes and Tale Bakken Ulfsby. I owe them all a huge “thank you” for enduring me in my ups and downs and motivating me while writing this thesis. In addition, I would like to give special thanks to Anders Lindstrøm and Henrik Aasen Kjeldsberg, my two closest friends at the university. Thank you for all your help with studying and all the fun we’ve had along the way.

I would also like to thank my girlfriend, Anine Gulestø, for her support and encouragement to keep working and especially for the countless times she has made me dinner after a late night at the study hall. Thank you for standing by me this last year. Last but not least, I need to thank my parents for their continuous support throughout my studies. Thank you for always believing in me.

My five years at the University of Oslo have probably been the most challenging years of my life, but thanks to my friends and fellow students, they have also been the best.

Kristoffer Ulvik Høisæther
Oslo, May 2018.

Contents

CHAPTER 1

Introduction

In many different fields, we need to reconstruct a signal from measured data. For instance, when working with image processing, we would like to reconstruct an image to its original form, from some measurement points. The measurements, or samples, are often done linearly, and we reconstruct the signal by formulating the linear system of the form,

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}.$$

Here, $\boldsymbol{y} \in \mathbb{C}^m$ is the measured data, $\boldsymbol{x} \in \mathbb{C}^N$ is the vector we wish to reconstruct and $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ is the measurement matrix that we need to construct. Shannon and Nyquist's sampling theorem states that the sampling rate must be at least twice the highest frequency of a continuous-time signal, to ensure reconstruction and avoid aliasing. Similarly, for the linear system, we need at least as many measurements, m, as the length of the vector $\boldsymbol{x}$, N. If however $m < N$, the system becomes underdetermined and unsolvable with infinite possible solutions, making it impossible to reconstruct $\boldsymbol{x}$ from $\boldsymbol{y}$. So how do we proceed when the number of measurements does not match the length of our original signal? This is where we meet the idea in which the entire field of compressive sensing is built on, namely *sparsity*. It shows, that under the assumption of sparsity we can reconstruct signals, even when the system is underdetermined. A vector or matrix is sparse if most of its components are zero. If at most s of the components are non-zero, we say that it is s-sparse. In mathematical terms: the vector $\boldsymbol{x} \in \mathbb{C}^N$ is s-sparse if,

$$\|\boldsymbol{x}\|_0 := \operatorname{card}(\operatorname{supp}(\boldsymbol{x})) \leqslant s.$$

We use the notation of the ℓ_0-"norm". $\|\boldsymbol{x}\|_0$ is not a true norm, but commonly used for counting the number of non-zero elements in a vector $\boldsymbol{x}$.

The reconstruction is performed by minimizing the vector $\boldsymbol{x}$ with respect to some constraints. The constraints are usually the measurements, that is, the linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}$. We formulate this as the ℓ_0-minimization,

$$\min \|\boldsymbol{x}\|_0 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}.$$

What we are trying to accomplish with the ℓ_0-minimization, is to find the vector $\boldsymbol{x}$ with the least amount of non-zero elements. In other words, we are trying to find the sparsest solution of the linear system $\boldsymbol{Ax} = \boldsymbol{y}$. In fact, we can construct the measurement matrix $\boldsymbol{A}$ such that ℓ_0-minimization always finds the unique, exact solution. However, the general challenge here is that the location of the non-zero elements are often unknown. Therefore, the algorithm must try for all possible locations of the non-zero elements, which becomes $\binom{N}{s}$ for an s-sparse vector of length N.

Let us consider an example for a quite standard image size of 1024×1024 pixels, and a 10-sparse vector $\boldsymbol{x}$. The algorithms would have to solve the system $\boldsymbol{Ax} = \boldsymbol{y}$, $\binom{1024}{10} \geqslant \left(\frac{1024}{10}\right)^{10} > 10^{20}$ times. With the 10-sparse vector $\boldsymbol{x}$, the system will require $10 \cdot 10 = 100$ iterations. Consider running this minimization on UiO's supercomputer, Abel, with a computational power of just above 200 teraFLOPS (number of floating point operations per second). The minimization would still use nearly two years of computations. The author's MacBook Pro, on the other hand, with less than 200 gigaFLOPS, would need more than 1500 years. We can safely say that this is impractical and NP-hardness is proven in Chapter 2. The approximation we rather perform is the ℓ_1-minimization, which is the convex relaxation of hte ℓ_0-minimization, and can be solved efficiently. We can often reformulate the minimization to a linear program and use interior point methods or the traditional simplex method, which we will introduce in Chapter 4. Since these methods and algorithms are designed for general linear programs, they may be slower than algorithms explicitly developed for ℓ_1-minimization. Algorithms such as the primal-dual algorithm and Nesterov's algorithm are specifically developed for ℓ_1-minimization.

1.1 Outline

The thesis is organized as follows:

Chapter 2 derives the ℓ_1-minimizer to be the best choice for optimization. Here we also present most of the background theory that constitutes the foundation of the remainder of the thesis.

Chapter 3 introduces necessary conditions for the algorithms in compressive sensing, such as the null space property, coherence and the restricted isometry property.

Chapter 4 gives an overview of the general algorithms associated with compressive sensing and some results from Chapter 3.

Chapter 5 introduces the concept of duality before deriving the primal-dual algorithm. Next, we prove convergence of the sequences computed in the algorithm.

Chapter 6 follows Nesterov's development of the algorithm known as NESTA and show that its general form works with both ℓ_1- and TV-minimization. Lastly, we introduce continuation, a method of speeding up the algorithm for some problems.

Chapter 7 summarizes the thesis.

CHAPTER 2

Optimization

When working with optimization, the problems we face require a high number of numerical computations. Some of the problems need too many numerical computations to compute within a reasonable time, and therefore, we start by introducing the topic of computational complexity, before moving on and showing why ℓ_1-minimization is such a widely used approach in minimization.

2.1 Computational Complexity

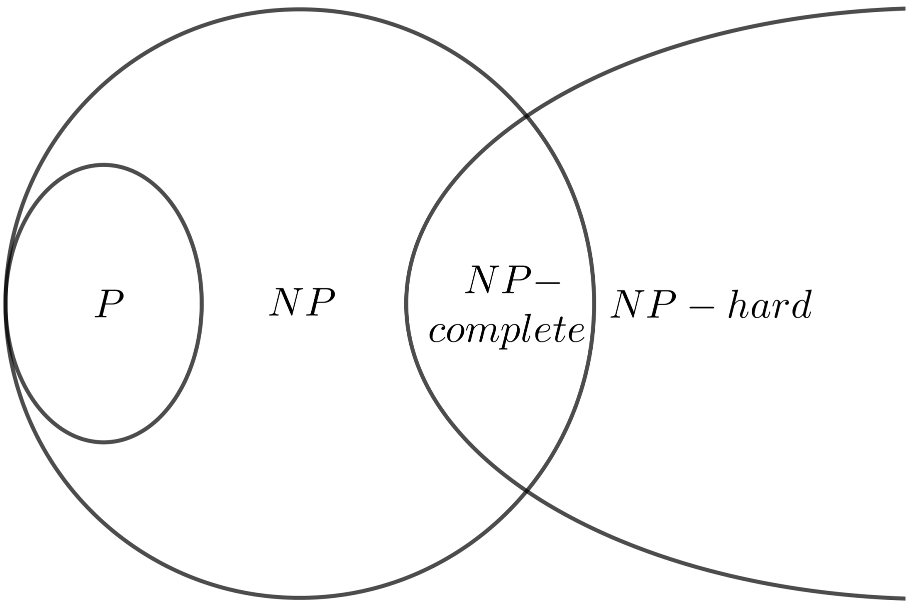

Figure 2.1: Representation of complexity classes.

First of all, a *polynomial-time algorithm* is an algorithm whose number of steps is bounded by a polynomial expression. The size of the polynomial is shortened to its dominating part and denoted $\mathcal{O}(n^k)$, where n^k is the dominating part. The classes of problems we are interested in are: P-problems (polynomial time), NP-problems (non-deterministic polynomial time), NP-hard and NP-complete.

P-problems are decision problems where there exists a polynomial-time algorithm.
NP-problems are all decision problems where there exists a polynomial-time algorithm that certifies a solution.
NP-hard problems are all problems (note: not only decision problems) that are at least as hard as any NP-problem.
NP-complete problems are all problems that are both NP and NP-hard. Thus, all NP-problems that are at least as hard as any other NP-problem.

2.2 ℓ_1-Minimization

When working with optimization, there are several possible approaches, but we will mainly consider ℓ_1-minimization as this is the main problem considered. But why ℓ_1? Why not ℓ_0 or ℓ_∞? Well, it turns out that ℓ_p-minimization with $0 \leqslant p < 1$ is NP-hard in general, and for $p > 1$, even 1-sparse vectors are not guaranteed to be a minimizer. We are now left with the problem $p = 1$, that is, ℓ_1-minimization which is a convex problem of the form:

$$\min \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}. \tag{P_1}$$

This problem is also known as (P_1), *ℓ_1-minimization* or *basis pursuit*. In the following propositions we prove NP-hardness of ℓ_p-minimization for $0 \leqslant p < 1$ and that we are not guaranteed a solution for $p > 1$. We start by proving NP-hardness of ℓ_0-minimization.

Proposition 2.1 ([11]). *ℓ_0-minimization is NP-hard.*

Proof. First, we introduce the 3-sets problem that is known to be NP-complete. $[N]$ denotes the set of natural number $\{1, 2, \dots, N\}$.
Given a collection $\{\mathcal{C}_i, i \in [N]\}$ of 3-element subsets of $[m]$, does there exist a partition of $[m]$, a set $J \subset [N]$ such that $\cup_{j \in J} \mathcal{C}_j = [m]$ and $\mathcal{C}_j \cap \mathcal{C}_{j'} = \varnothing$ for all $j, j' \in J$ with $j \neq j'$?
We wish to reduce this problem to ℓ_0-minimization in polynomial time and thereby showing NP-hardness of ℓ_0-minimization. Let the collection $\{\mathcal{C}_i, i \in [N]\}$, be defined as in the problem. We now define vectors $\boldsymbol{a}_1, \dots, \boldsymbol{a}_N \in \mathbb{C}^m$ as,

$$(\boldsymbol{a}_i)_j = \begin{cases} 1, & j \in \mathcal{C}_i, \\ 0, & j \notin \mathcal{C}_i. \end{cases}$$

The vectors make up the matrix $\boldsymbol{A} \in \mathbb{C}^{m \times N}$,

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_1 & | & \dots & | & \boldsymbol{a}_N \end{bmatrix}, \qquad \boldsymbol{y} = \begin{bmatrix} 1, & \dots, & 1 \end{bmatrix}^\top.$$

Since $N \leqslant \binom{m}{3}$, we can construct the matrix in polynomial time. Now, if the vector $\boldsymbol{z} \in \mathbb{C}^m$ obeys $\|\boldsymbol{A}\boldsymbol{z} - \boldsymbol{y}\|_2 \leqslant \eta$, then all m components of $\boldsymbol{z}$ are at most

η distant to 1 and non-zero, making $\|\boldsymbol{Az}\|_0 = m$. As each of the vectors $\boldsymbol{a}_i$ has 3 non-zero components, the product $\boldsymbol{Az} = \sum_{i=1}^N z_j \boldsymbol{a}_i$ has at most $3\|\boldsymbol{z}\|_0$ non-zero components, meaning that $\|\boldsymbol{Az}\|_0 \leqslant 3\|\boldsymbol{z}\|_0$, which again implies that a vector $\boldsymbol{z}$ satisfying the constraint must also satisfy $\|\boldsymbol{z}\|_0 \geqslant m/3$. If we now take the ℓ_0-minimization, we can separate the results into two cases: Let $\boldsymbol{x} \in \mathbb{C}^N$ be the output.

1. $\|\boldsymbol{x}\|_0 = m/3$. The collection $\{\mathcal{C}_i, i \in \operatorname{supp}(\boldsymbol{x})\}$ forms an exact cover of $[m]$.
2. $\|\boldsymbol{x}\|_0 > m/3$. No exact cover can exist, as $\boldsymbol{z}$ would satisfy $\boldsymbol{Az} = \boldsymbol{y}$ and $\|\boldsymbol{z}\|_0 = m/3$ which contradicts the minimization of $\boldsymbol{x}$.

This shows that by solving ℓ_0-minimization, one can also solve the 3-sets problem, and thus concluding the proof. □

The technique of *reduction* used in this proof is a transformation between problems. Consider a problem A that can be solved by an algorithm for solving another problem B, then A is no harder than B, and we can say that A reduces to B. For a simple example, let us consider the problem of multiplying two numbers. As should be well known, multiplication can be solved by adding, and we can reduce the problem of multiplication to the problem of adding. Also, note that the reduction must be done in polynomial time in order for this to be possible.

Proposition 2.2. *Every ℓ_p-minimization is NP-hard when* $0 < p < 1$.

Proof. To help us prove this, we introduce the *partition problem*. The partition problem is deciding whether, with given integers $a_1, \dots, a_n$, there exists two sets I and J such that $I \cap J = \varnothing$, $I \cup J = [N]$ and $\sum_{i \in I} a_i = \sum_{j \in J} a_j$. Then, by assuming NP-completeness of the partition problem, we prove NP-hardness of ℓ_p-minimization by reducing the partition problem to ℓ_1-minimization. Let $\boldsymbol{x}$ and $\boldsymbol{z}$ be the first and second half of the vector $\boldsymbol{w}$ to $\boldsymbol{A}$. We want to show that the partition problem has a solution, if and only if, the minimum of $\|\boldsymbol{w}\|_p$ subject to $\boldsymbol{Aw} = \boldsymbol{y}$ is N. First, we need to formulate the partition problem.

$\boldsymbol{Aw} = \boldsymbol{y} \Rightarrow \sum_{i=1}^N a_i x_i = \sum_{i=1}^N a_i z_i$ and $x_i + z_i = 1 \ \forall\ i$. So, what we seek to minimize is $\sum_{i=1}^N (x_i^p + z_i^p)$, under these constraints. The function $|x|^p + |y|^p$ is concave in the first quadrant, so the minimum must be in one of the end points, $(1, 0)$ and $(0, 1)$. As we are interested in the minimum, we will only use zeros and ones. Then, if the minimum of the partition problem, $\|\boldsymbol{w}\|_p$ subject to $\boldsymbol{Aw} = \boldsymbol{y}$ is N, it will also be N when only $x_i + z_i = 1$ are considered, as either x_i or z_i must be equal to 1, for all i. If we let I be the set of indices where $x_i = 1$ and J for $z_i = 1$, we have formulated the partition problem on the form we seek, $I \cap J = \varnothing$, $I \cup J = [N]$ and the constraints

$$\sum_{i=1}^N a_i x_i = \sum_{i \in I} a_i, \qquad \sum_{j=1}^N a_j z_j = \sum_{j \in J} a_j.$$

So, we have $\sum_{i \in I} a_i = \sum_{i=1}^N a_i z_i$ and the partition problem has a solution with minimum N. By defining the vector $\boldsymbol{x}$ as $x_i = 1$ when $i \in I$ and similar for $\boldsymbol{z}$, we obtain the vector $\boldsymbol{w}$ where all the constraints are fulfilled and the minimum is N. We can now conclude that ℓ_p-minimization with $0 < p < 1$ is NP-hard. $\square$

Now that we have proved NP-hardness of ℓ_p-minimization for $0 \leqslant p < 1$, we prove that we are not guaranteed a solution for $p > 1$.

Proposition 2.3 ([11])**.** *Let $\boldsymbol{A} \in \mathbb{R}^{m\times N}$ with $m < N$ and $p > 1$. Then there exists a 1-sparse vector which is not a minimizer of*

$$\min \|\boldsymbol{x}\|_p \quad \textit{subject to} \quad \boldsymbol{Ax} = \boldsymbol{y}. \tag{P_p}$$

Proof. Assume by contradiction that every standard basis vectors, $\boldsymbol{e}_j$, are minimizers. Since $m < N$ the kernel of $\boldsymbol{A}$ is not trivial, so $\exists\, \boldsymbol{v} \neq \boldsymbol{0}$ with $\boldsymbol{Av} = \boldsymbol{0}$, we have,

$$\|\boldsymbol{e}_j + t\boldsymbol{v}\|_p^p = |1 + tv_j|^p + \sum_{k\neq j} |tv_k|^p = |1 + tv_j|^p + |t|^p \sum_{k\neq j} |v_k|^p$$

Consider the two functions, $g_+(t)$ and $g_-(t)$:

$$g_+(t) = (1 + tv_j)^p + t^p \sum_{k\neq j} |v_k|^p$$
$$g_-(t) = (1 + tv_j)^p + (-t)^p \sum_{k\neq j} |v_k|^p.$$

For $|t| < \frac{1}{v_j}$, $\|\boldsymbol{e}_j + t\boldsymbol{v}\|_p^p$ coincides with g_+ when $t \geqslant 0$ and g_- for $t \leqslant 0$. The derivatives of g_+ and g_-:

$$g_+'(t) = pqv_j(1 + tv_j)^{p-1} + pt^{p-1} \sum_{k\neq j} |v_k|^p$$
$$g_-'(t) = pv_j(1 + tv_j)^{p-1} - p(-t)^{p-1} \sum_{k\neq j} |v_k|^p.$$

For $p > 1$:

$$\lim_{t\to 0^+} g_+'(t) = \lim_{t\to 0^-} g_-'(t) = pv_j$$

The last part means that near 0, $\|\boldsymbol{e}_j + t\boldsymbol{v}\|_p^p$ has derivative $pv_j \neq 0$, and since $\boldsymbol{e}_j$ corresponds to $t = 0$, it cannot be a minimizer of (P_p). A linear function with derivative pv_j has no minimum near 0.

$\square$

Based on Propositions 2.1 to 2.3, we see that we are left with the ℓ_1-norm as the natural choice of minimizer. The ℓ_1-norm is known as the convex relaxation of the ℓ_0-norm.

The following theorem from [11, thm. B.26] states a strong duality property that we use when proving Proposition 2.5.

Theorem 2.4. *Assume that $F_0, F_1, \dots, F_M$ are convex functions with dom$(F_0) = \mathbb{R}^N$. If there exists $\boldsymbol{x} \in \mathbb{R}^N$ such that $\boldsymbol{Ax} = \boldsymbol{y}$ and $F_\ell(\boldsymbol{x}) < b_\ell$ for all $\ell \in [M]$, then strong duality holds for the optimization problem (P_1). In the absence of inequality constraints, strong duality holds if there exists $\boldsymbol{x} \in \mathbb{R}^N$ where $\boldsymbol{Ax} = \boldsymbol{y}$ is feasible.*

Proof. This is known as Slater's constraint or Slater's condition. See [10, Sect. 5.3.2] for proof. □

We will now introduce some optimization methods where their connection is showed in Proposition 2.5. The following optimization methods are called *quadratically constrained basis pursuit* (2.1), *basis pursuit denoising* (2.2) and *least absolute shrinkage and selection operator*, or *LASSO* for short, (2.3).

$$\min_{\boldsymbol{x}\in\mathbb{C}^n} \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \|\boldsymbol{Ax}-\boldsymbol{y}\|_2 \leqslant \eta \tag{2.1}$$

$$\min_{\boldsymbol{x}\in\mathbb{C}^n} \lambda\|\boldsymbol{x}\|_1 + \|\boldsymbol{Ax}-\boldsymbol{y}\|_2^2 \tag{2.2}$$

$$\min_{\boldsymbol{x}\in\mathbb{C}^n} \|\boldsymbol{Ax}-\boldsymbol{y}\|_2 \quad \text{subject to} \quad \|\boldsymbol{x}\|_1 \leqslant \tau \tag{2.3}$$

Proposition 2.5. *The following holds:*

(a) *if x is a minimizer of (2.2) with $\lambda > 0$, then $\exists\, \eta \geqslant 0$ such that x is a minimizer of (2.1).*

(b) *if x is a unique minimizer of (2.1) with $\eta \geqslant 0$, then $\exists\, \tau \geqslant 0$ such that x is a unique minimizer of (2.3).*

(c) *if x is a minimizer of (2.3) with $\tau > 0$, then $\exists\, \lambda \geqslant 0$ such that x is a minimizer of (2.2).*

Proof.

(a) Set $\eta := \|\boldsymbol{Ax}-\boldsymbol{y}\|_2$ and consider $\boldsymbol{z}\in\mathbb{C}^n$ such that $\|\boldsymbol{Az}-\boldsymbol{y}\|_2 \leqslant \eta$. Since $\boldsymbol{x}$ is a minimizer for (2.2), we have

$$\begin{aligned}\lambda\|\boldsymbol{x}\|_1 + \|\boldsymbol{Ax}-\boldsymbol{y}\|_2^2 \leqslant \lambda\|\boldsymbol{z}\|_1 + \|\boldsymbol{Az}-\boldsymbol{y}\|_2^2 \leqslant \lambda\|\boldsymbol{z}\|_1 + \|\boldsymbol{Ax}-\boldsymbol{y}\|_2^2 \\ \Rightarrow \|\boldsymbol{x}\|_1 \leqslant \|\boldsymbol{z}\|_1,\end{aligned}$$

hence, $\boldsymbol{x}$ is a minimizer for (2.1).

(b) Set $\tau := \|\boldsymbol{x}\|_1$ and consider $\boldsymbol{z}\in\mathbb{C}^n$ with $\boldsymbol{z}\neq\boldsymbol{x}$ such that $\|\boldsymbol{z}\|_1 \leqslant \tau$. Since $\boldsymbol{x}$ is the unique minimizer of (2.1), $\boldsymbol{z}$ cannot satisfy the constraint $\|\boldsymbol{Az}-\boldsymbol{y}\|_2 \leqslant \eta$. So, $\|\boldsymbol{Az}-\boldsymbol{y}\|_2 > \eta \geqslant \|\boldsymbol{Ax}-\boldsymbol{y}\|_2$. This implies that $\boldsymbol{x}$ is a unique minimizer of (2.3).

(c) (2.3) is equivalent to

$$\min_{\boldsymbol{z}\in\mathbb{C}^n} \|\boldsymbol{Az}-\boldsymbol{y}\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{z}\|_1 \leqslant \tau$$

The Lagrangian of this problem is

$$\mathcal{L}(\boldsymbol{x},\boldsymbol{\xi}) = \|\boldsymbol{Ax}-\boldsymbol{y}\|_2^2 + \boldsymbol{\xi}(\|\boldsymbol{x}\|_1 - \tau)$$

Since $\tau > 0$, there exists a vector $\boldsymbol{x}$ with $\|\boldsymbol{x}\|_1 < \tau$, and by Theorem 2.4, we have strong duality. From the strong duality we know that there exists a dual optimal $\boldsymbol{\xi}^\sharp \geqslant 0$, and the saddle point property implies that $\mathcal{L}(\boldsymbol{x}^\sharp,\boldsymbol{\xi}^\sharp) \leqslant \mathcal{L}(\boldsymbol{x},\boldsymbol{\xi}^\sharp)$ for all $\boldsymbol{x}\in\mathbb{R}^N$. Then, $\boldsymbol{x}^\sharp$ is also a minimizer of $\mathcal{L}(\boldsymbol{x},\boldsymbol{\xi}^\sharp)$. Since the term $-\boldsymbol{\xi}^\sharp\tau$ does not affect the minimizer, λ must be equal to $\boldsymbol{\xi}^\sharp$.

□

While possible, it is important to note that finding λ, τ and η that satisfies the conditions above is very difficult. In Section 4.3 we see how the SPGL1 algorithm uses the equivalence of the LASSO and the quadratically constrained pursuit to calculate τ and η.

Lipschitz continuity

Lipschitz continuity is a form of continuity that is stronger than continuous, but not as strong as continuously differentiable. The changes in the function are bounded by a real constant. A typical visualization of this property is a double cone being translated along the graph by its vertex without overlapping, as illustrated in Figure 2.2.

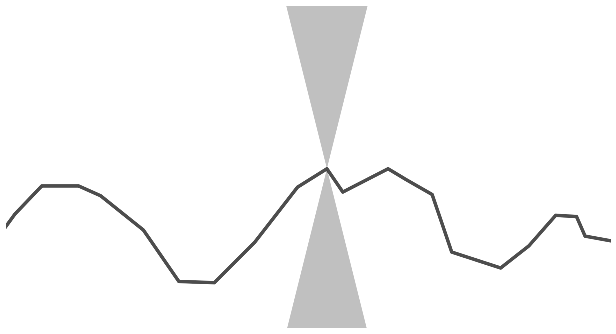
Figure 2.2: Lipschitz continuity.

Definition 2.6. A function is said to be a *Lipschitz function*, if

$$\|f(\boldsymbol{x}) - f(\boldsymbol{y})\|_2 \leqslant L\|\boldsymbol{x} - \boldsymbol{y}\|_2.$$

The constant L is called the *Lipschitz constant*.

We will revisit Lipschitz continuity when examining Nesterov's algorithm, where the general function f is assumed to be Lipschitz continuous.

2.3 Convexity

In optimization, we prefer to work with convex problems and functions. This is because a convex optimization problem has only one optimal solution. By this, we mean that a local optimal solution is also the global optimal solution. For non-convex optimization problems, there may be several local optimal solutions, and therefore, requires many more computations to find the global optimal solution. The following definition defines the convex set, and the next defines convex functions.

Definition 2.7. A subset $C \subseteq \mathbb{R}^n$ is called convex, if $\forall \boldsymbol{x}, \boldsymbol{y} \in C$, the line segment connecting $\boldsymbol{x}$ and $\boldsymbol{y}$ is entirely contained in C, that is,

$$\lambda \boldsymbol{x} + (1 - \lambda)\boldsymbol{y} \in C \quad \forall \lambda \in [0, 1].$$

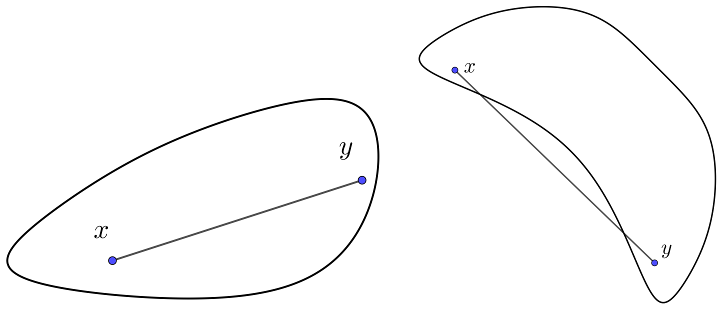

Figure 2.3: Illustration of convex set (left) and concave set (right).

Definition 2.8. A function $f\colon C \to \mathbb{R}$, where $C \subseteq \mathbb{R}^n$ is convex if $\boldsymbol{x}, \boldsymbol{y} \in C$, $0 \leqslant \lambda \leqslant 1$ and,

$$f((1-\lambda)\boldsymbol{x} + \lambda\boldsymbol{y}) \leqslant (1-\lambda)f(\boldsymbol{x}) + \lambda f(\boldsymbol{y})$$

f is called *strictly* convex if $\boldsymbol{x} \neq \boldsymbol{y}$ and $0 < \lambda < 1$ and,

$$f((1-\lambda)\boldsymbol{x} + \lambda\boldsymbol{y}) < (1-\lambda)f(\boldsymbol{x}) + \lambda f(\boldsymbol{y})$$

f is called *strongly* convex with parameter $\gamma > 0$ if, $\forall\ \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and $t \in [0,1]$

$$f((1-\lambda)\boldsymbol{x} + \lambda\boldsymbol{y}) \leqslant (1-\lambda)f(\boldsymbol{x}) + \lambda f(\boldsymbol{y}) - \frac{\gamma}{2}\lambda(1-\lambda)\|\boldsymbol{x}-\boldsymbol{y}\|_2^2$$

An alternate definition of strong convexity is, if $f(\boldsymbol{x}) - \frac{\gamma}{2}\|\boldsymbol{x}\|_2^2$ is convex, then the function $f(\boldsymbol{x})$ is strongly convex.

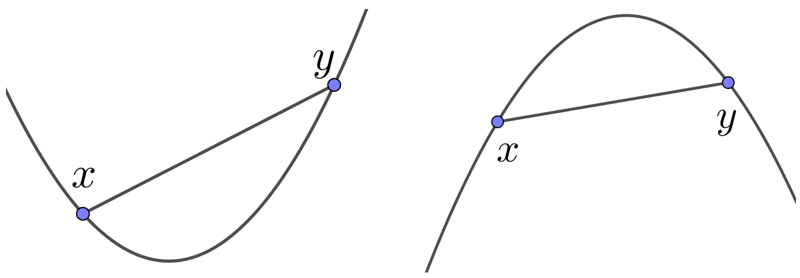

Figure 2.4: Illustration of convex function (left) and concave function (right).

Strong convexity $\Rightarrow$ strict convexity $\Rightarrow$ convexity. Strict convexity to convexity is evident from the definition, as the main difference is strict inequality. For the other implication, strong to strict, we have to notice that the last term in the strongly convex definition is always positive which means that by removing this term, strict convexity still holds while the converse does not. The following theorem from [7, thm. 12.10] shows some equivalent results regarding convex functions, which are useful when exploring how Nesterov's algorithm works in Section 6.2.

Theorem 2.9 (Convex Functions). *Let $f : C \to \mathbb{R}$ be a differentiable function defined on an open convex set $C \subseteq \mathbb{R}^n$. Then the following conditions are equivalent:*

1. *f is convex*
2. $f(\boldsymbol{x}) \geqslant f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x})^\top (\boldsymbol{x} - \boldsymbol{x}_0) \quad \forall \boldsymbol{x}, \boldsymbol{x}_0 \in C$
3. $(\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}_0))^\top (\boldsymbol{x} - \boldsymbol{x}_0) \geqslant 0 \quad \forall \boldsymbol{x}, \boldsymbol{x}_0 \in C$

Proof. First, lets assume $n = 1$ and f is convex. By the definition of convex functions we can rewrite as follows by letting $\lambda \to 0$:

$$\begin{aligned} f(x) &\geqslant f(x_0) + \frac{f(x_0 + \lambda(x - x_0)) - f(x_0)}{\lambda} \\ &= f(x_0) + \frac{f(x_0 + \lambda(x - x_0)) - f(x_0)}{\lambda(x - x_0)}(x - x_0) \\ &= f(x_0) + \nabla f(x_0)^\top (x - x_0). \end{aligned}$$

We continue to show that 2. $\Rightarrow$ 3. by adding the following equations (which also hold for $n > 1$):

$$\begin{aligned} f(\boldsymbol{x}) &\leqslant f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^\top (\boldsymbol{x} - \boldsymbol{x}_0) \\ f(\boldsymbol{x}_0) &\leqslant f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\top (\boldsymbol{x}_0 - \boldsymbol{x}) \\ \\ \Rightarrow \quad & (\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}_0))^\top (\boldsymbol{x} - \boldsymbol{x}_0) \geqslant 0. \end{aligned}$$

Now, 3. $\Rightarrow$ 1. Consider $x_1 < x_2 < x_3$, then the mean value theorem states that there exists two numbers a and b such that, $x_1 \leqslant a \leqslant x_2$, $x_2 \leqslant b \leqslant x_3$, so

$$\frac{f(x_2) - f(x_1)}{x_2 - x_1} = f'(a), \quad \text{and} \quad \frac{f(x_3) - f(x_2)}{x_3 - x_2} = f'(b).$$

By 3. we know that $f'(a) \leqslant f'(b)$, which means that the slope is increasing and the function must be convex. This concludes the proof for $n = 1$.

For $n > 1$, we define the general function $g(t)$ and its derivative

$$\begin{aligned} g(t) &= f(t\boldsymbol{x} + (1 - t)\boldsymbol{x}_0) \\ g'(t) &= \nabla f(t\boldsymbol{x} + (1 - t)\boldsymbol{x}_0)^\top (\boldsymbol{x} - \boldsymbol{x}_0). \end{aligned}$$

If f is convex, then g is also convex, and this can be shown by the definition of convexity. From 2. we have $g(1) \geqslant g(0) + g'(0)$, which holds for $n > 1$ by replacing f and g, $f(\boldsymbol{x}) \geqslant f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^\top (\boldsymbol{x} - \boldsymbol{x}_0)$. Final part of the proof, 3. $\Rightarrow$ 1., we let $0 \leqslant t_1 \leqslant t_2 < 1$ and $\boldsymbol{y}_1 = t_1\boldsymbol{x} + (1 - t_1)\boldsymbol{x}_0$ with $\boldsymbol{y}_2$ defined in the same way, and we notice that $\boldsymbol{y}_2 - \boldsymbol{y}_1 = (t_2 - t_1)(\boldsymbol{x} - \boldsymbol{x}_0)$. Rewriting 3., we see that it can take the form,

$$\begin{aligned} &(\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}_0))^\top (\boldsymbol{x} - \boldsymbol{x}_0) \geqslant 0 \\ &\qquad \Rightarrow \nabla f(\boldsymbol{x}_0)^\top (\boldsymbol{x} - \boldsymbol{x}_0) \geqslant \nabla f(\boldsymbol{x})^\top (\boldsymbol{x} - \boldsymbol{x}_0). \end{aligned}$$

From this we find,

$$g'(t_i) = \nabla f(\boldsymbol{y}_i)^\top(\boldsymbol{x}-\boldsymbol{x}_0) = \nabla f(\boldsymbol{y}_i)^\top \frac{(\boldsymbol{y}_2-\boldsymbol{y}_1)}{(t_2-t_1)}.$$

This will also hold if we use $\boldsymbol{x}$ and $\boldsymbol{x}_0$ instead of $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$, and it follows that $g'(t_1) \leqslant g'(t_2)$. From this 1. follows,

$$\begin{aligned} g(t) = f(t\boldsymbol{x}+(1-t)\boldsymbol{x}_0) &= g(t\cdot 1+(1-t)\cdot 0) \\ &\leqslant tg(1)+(1-t)g(0) \\ &= tf(\boldsymbol{x})+(1-t)f(\boldsymbol{x}_0), \end{aligned}$$

completing the proof. □

Later in the thesis we will go through a proof that requires that the maximum of convex function is convex as well, and this is shown in the following proposition.

Proposition 2.10. *If f and g are convex functions, then $\max\{f(\boldsymbol{x}), g(\boldsymbol{x})\}$ are convex as well.*

Proof. Let $h(\boldsymbol{x}) := \max\{f(\boldsymbol{x}), g(\boldsymbol{x})\}$. We want to show:

$$h((1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y}) \leqslant (1-\lambda)h(\boldsymbol{x})+\lambda h(\boldsymbol{y}).$$

Thus,

$$\begin{aligned} h((1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y}) &= \max\{f((1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y}), g((1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y})\} \\ &\leqslant \max\{(1-\lambda)f(\boldsymbol{x})+\lambda f(\boldsymbol{y}), (1-\lambda)g(\boldsymbol{x})+\lambda g(\boldsymbol{y})\} \\ &\leqslant \max\{(1-\lambda)f(\boldsymbol{x}), (1-\lambda)g(\boldsymbol{x})\} + \max\{\lambda f(\boldsymbol{y}), \lambda g(\boldsymbol{y})\} \\ &= (1-\lambda)\max\{f(\boldsymbol{x}), g(\boldsymbol{x})\} + \lambda\max\{f(\boldsymbol{y}), g(\boldsymbol{y})\} \\ &= (1-\lambda)h(\boldsymbol{x})+\lambda h(\boldsymbol{y}), \end{aligned}$$

as we wanted to show. □

In Definition 2.8, we defined strongly convex functions and in this proposition, we prove strong convexity of a given function that is widely used in optimization, as we'll notice when we introduce the proximal mapping and prox-functions.

Proposition 2.11. *$f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{x}_c\|^2$, where $\boldsymbol{x}, \boldsymbol{x}_c \in \mathbb{R}^N$, is strongly convex with parameter $\gamma = 1$.*

Proof. We want to show:

$$f((1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y}) - (1-\lambda)f(\boldsymbol{x}) - \lambda f(\boldsymbol{y}) \leqslant -\frac{\gamma}{2}\lambda(1-\lambda)\|\boldsymbol{x}-\boldsymbol{y}\|_2^2.$$

First, lets rewrite $f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{x}_c\|^2 = \frac{1}{2}\|\boldsymbol{x}\|^2 + \|\boldsymbol{x}_c\|^2 - \langle \boldsymbol{x}, \boldsymbol{x}_c\rangle$. We start by writing out the different parts of the left hand side of the inequality:

$$\begin{aligned}
f((1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y}) =& \frac{1}{2}\|(1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y}\|^2+\frac{1}{2}\|\boldsymbol{x}_c\|^2-\langle(1-\lambda)\boldsymbol{x}+\lambda\boldsymbol{y},\boldsymbol{x}_c\rangle\\
=& \frac{1}{2}(1-\lambda)^2\|\boldsymbol{x}\|^2+\frac{1}{2}\lambda^2\|\boldsymbol{y}\|^2+\lambda(1-\lambda)\langle\boldsymbol{x},\boldsymbol{y}\rangle+\frac{1}{2}\|\boldsymbol{x}_c\|^2\\
&-(1-\lambda)\langle\boldsymbol{x},\boldsymbol{x}_c\rangle-\lambda\langle\boldsymbol{y},\boldsymbol{x}_c\rangle,\\
(1-\lambda)f(\boldsymbol{x}) =& \frac{1}{2}(1-\lambda)\|\boldsymbol{x}\|^2+\frac{1}{2}(1-\lambda)\|\boldsymbol{x}_c\|^2-(1-\lambda)\langle\boldsymbol{x},\boldsymbol{x}_c\rangle,\\
\lambda f(\boldsymbol{y}) =& \frac{1}{2}\lambda\|\boldsymbol{y}\|^2+\frac{1}{2}\lambda\|\boldsymbol{x}_c\|^2-\lambda\langle\boldsymbol{y},\boldsymbol{x}_c\rangle.
\end{aligned}$$

Setting these back together and rewriting we find,

$$-\frac{1}{2}\lambda(1-\lambda)\|\boldsymbol{x}-\boldsymbol{y}\|^2\leqslant-\frac{\gamma}{2}\lambda(1-\lambda)\|\boldsymbol{x}-\boldsymbol{y}\|_2.$$

Where the inequality holds when $0\leqslant\gamma\leqslant 1$.

□

The subdifferential is a set of subgradients of a function at a given point. The subgradients are gradients for non-differential functions. For example, lets consider the function $f(x)=|x|$. The subdifferential of $f(x)$ at any point $x<0$ is the set $\{-1\}$ and for $x>0$ it is $\{1\}$. For the origin, the gradient is not defined for this function. The subdifferential is therefore all possible subgradients, making it the set $[-1,1]$. In mathematical terms, the subdifferential is defined as follows.

Definition 2.12. The subdifferential of a convex function $F\colon\mathbb{R}^N\to(-\infty,\infty]$ at a point $\boldsymbol{x}\in\mathbb{R}^N$ is defined by,

$$\partial F(\boldsymbol{x})=\{\boldsymbol{v}\in\mathbb{R}^N\mid F(\boldsymbol{z})\geqslant F(\boldsymbol{x})+\langle\boldsymbol{v},\boldsymbol{z}-\boldsymbol{x}\rangle\ \forall\boldsymbol{z}\in\mathbb{R}^N\}.$$

From the definition, we can immediately derive the following result which we will use when proving convergence of the primal-dual algorithm in Theorem 5.3.

Proposition 2.13 ([11]). *$\boldsymbol{x}\in\mathbb{R}^N$ is a minimum of a convex function F if and only if $\mathbf{0}\in\partial F(\boldsymbol{x})$.*

Proof. If $\boldsymbol{x}$ is a minimum of F, then the term $\langle\boldsymbol{v},\boldsymbol{z}-\boldsymbol{x}\rangle$ from the definition of the subdifferential must be $\mathbf{0}$, which can only happen when $\boldsymbol{v}=\mathbf{0}$. The proof the other way is equally easy to notice, if $\boldsymbol{v}=\mathbf{0}$, $\boldsymbol{x}$ must be a minimizer of F. □

Next, we introduce the proximal mapping and the correlated prox-function. The proximal mapping is an approximation of a function, and is mostly used in optimization when the function at hand is not convex. The mapping is defined as presented below.

Definition 2.14. The proximal mapping associated with a function F is defined as,

$$P_F(\boldsymbol{x}) := \operatorname*{argmin}_{\boldsymbol{z}\in\mathbb{R}^N} F(\boldsymbol{z}) + \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{x}\|_2^2.$$

The function we use for smoothing in the proximal mapping is a prox-function. In short, a prox-function is a continuous and strongly convex function on a closed set. As we showed in Proposition 2.11, the function we use for the proximal mapping is a strongly convex function.

Definition 2.15. p is a prox-function for a closed convex set Q if

- p is continuous on Q
- p is strongly convex on Q

If the prox-function vanishes at the prox-center, $\boldsymbol{x}_c$, then we get the boundary

$$p(\boldsymbol{x}) \geqslant \frac{\sigma}{2}\|\boldsymbol{x} - \boldsymbol{x}_c\|_2^2,$$

where σ is a convexity parameter. We may refer to a prox-function as $p_p(\boldsymbol{x})$, when it's associated with, for instance, a primal set Q_p. The convexity parameter, σ, also takes the subscript, in this case, σ_p. The primal-dual algorithm and Nesterov's algorithm both uses proximal mappings and prox-functions.

The following proposition shows a useful connection for how to rewrite the proximal mapping as the subdifferential. The proposition will help us proving convergence of the primal-dual algorithm.

Proposition 2.16 ([11]). *Let $F\colon\mathbb{R}^N \to (-\infty, \infty]$ be a convex function, then $\boldsymbol{x} = P_F(\boldsymbol{z})$ if and only if $\boldsymbol{z} \in \boldsymbol{x} + \partial F(\boldsymbol{x})$.*

Proof. From Proposition 2.13 we know that if $\boldsymbol{x}$ is a minimum of a convex function F, then $\boldsymbol{0} \in \partial F(\boldsymbol{x})$. Based on this, we can find that $\boldsymbol{x}$ is the proximal mapping $P_F(\boldsymbol{z})$ if and only if,

$$\boldsymbol{0} \in \partial(\frac{1}{2}\|\cdot - \boldsymbol{z}\|_2^2 + F)(\boldsymbol{x}).$$

From the proximal mapping we have the function $\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2$ with gradient $\nabla(\frac{1}{2}\|\cdot - \boldsymbol{z}\|_2^2)(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{z}$. We can now insert this in the expression above such that,

$$\boldsymbol{0} \in \boldsymbol{x} - \boldsymbol{z} + \partial F(\boldsymbol{x}) \Leftrightarrow \boldsymbol{z} \in \boldsymbol{x} + \partial F(\boldsymbol{x}),$$

as we wanted to show. □

Another useful connection regarding the proximal mapping is Moreau's identity. This identity shows the relation between the proximal mapping of F and its convex conjugate F^*. Before presenting the identity, we need to define the convex conjugate.

Definition 2.17. Given a function $F\colon \mathbb{R}^N \to (-\infty,\infty]$, the convex conjugate function of F is, $F^*\colon \mathbb{R}^N \to (-\infty,\infty]$ defined as,

$$F^*(\boldsymbol{y}) := \sup_{\boldsymbol{x}\in\mathbb{R}^N} \{\langle \boldsymbol{x},\boldsymbol{y}\rangle - F(\boldsymbol{x})\}.$$

The convex conjugate function is widely used in the primal-dual algorithm, where we will also revisit Moreaus's identity which reads as follows.

$$P_F(\boldsymbol{z}) + P_{F^*}(\boldsymbol{z}) = \boldsymbol{z}. \tag{2.4}$$

For proximal mappings where the function takes the form τF, the identity reads,

$$P_{\tau F^*}(\boldsymbol{z}) + \tau P_{\tau^{-1}F}(\boldsymbol{z}/\tau) = \boldsymbol{z}. \tag{2.5}$$

Both identities are proved in [11, thm. B.24], by using the relation between the subdifferential and the proximal mapping and some more identities for the mapping.

2.4 Lagrangian and KKT

The Lagrangian is a widely used strategy in mathematical optimization for finding the maximum and minimum of functions with constraints. Consider the following optimization problem,

$$\min f(\boldsymbol{x}) \quad \text{subject to} \quad h_i(\boldsymbol{x}) = \boldsymbol{0} \quad (i \leqslant m), \tag{2.6}$$

where f and h_i ($\forall\, i$) are continuously differentiable functions from $\mathbb{R}^N$ to $\mathbb{R}$. We let $\boldsymbol{h}$ be the vector containing each h_i, $\boldsymbol{h} = \{h_1,\dots,h_m\}$, and before we formulate the Lagrangian theorem we show how the gradient of a linear system is computed.

Proposition 2.18. *Let $\boldsymbol{A}$ be a positive definite matrix, then the gradient of the function f becomes,*

$$\begin{aligned} f(\boldsymbol{x}) &= \tfrac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x},\\ \nabla f(\boldsymbol{x}) &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}. \end{aligned}$$

Proof. For easier notation we will let $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$.

$$\begin{aligned} \nabla(\tfrac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^\top\boldsymbol{x}) &= \nabla(\tfrac{1}{2}\boldsymbol{x}^\top\boldsymbol{y}) + \nabla(\boldsymbol{b}^\top\boldsymbol{x})\\ &= \frac{1}{2}\frac{\delta(\boldsymbol{x}^\top\boldsymbol{y})}{\delta\boldsymbol{x}} + \frac{\delta(\boldsymbol{b}^\top\boldsymbol{x})}{\delta\boldsymbol{x}}\\ &= \frac{1}{2}\frac{\delta(\boldsymbol{x}^\top\boldsymbol{y})}{\delta\boldsymbol{x}} + \frac{1}{2}\frac{\delta\boldsymbol{y}^\top}{\delta\boldsymbol{x}}\frac{\delta(\boldsymbol{x}^\top\boldsymbol{y})}{\delta\boldsymbol{y}} + \frac{\delta(\boldsymbol{b}^\top\boldsymbol{x})}{\delta\boldsymbol{x}}\\ &= \tfrac{1}{2}\boldsymbol{A}\boldsymbol{x} + \tfrac{1}{2}\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}\\ &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}. \end{aligned}$$

□

Theorem 2.19 (Lagrange [7]). *Let x^* be a local minimum of the optimization problem (2.6) and assume it is a regular point. Then there is a unique vector $\boldsymbol{\lambda}^* = (\lambda_1^*, \lambda_2^*, \dots, \lambda_m^*) \in \mathbb{R}^m$ such that,*

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \boldsymbol{\lambda}_i^* \nabla h_i(\boldsymbol{x}^*) = \mathbf{0}.$$

If f and each h_i are twice continuously differentiable, then the following also holds,

$$\boldsymbol{h}^\top (\nabla^2 f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i^* \nabla^2 h_i(\boldsymbol{x}^*))\boldsymbol{h} \geqslant \mathbf{0},$$

for all $\boldsymbol{h} \in T(\boldsymbol{x}^*)$, where $T(\boldsymbol{x}^*) = \{\boldsymbol{h} \in \mathbb{R}^m \mid \nabla h_i(\boldsymbol{x}^*) \cdot \boldsymbol{h} = \mathbf{0}\ (i \leqslant m)\}$.

The λ_i's in this theorem are called the *Lagrangian multipliers*. From the theorem we can formulate the *Lagrangian function*, $\mathcal{L}: \mathbb{R}^N \times \mathbb{R}^m \to \mathbb{R}$, with $\boldsymbol{x} \in \mathbb{R}^N$ and $\boldsymbol{\lambda} \in \mathbb{R}^m$,

$$\begin{aligned}\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) &= f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i h_i(\boldsymbol{x}) \\ &= f(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{h}(\boldsymbol{x}).\end{aligned}$$

We can choose to either add or subtract the term $\boldsymbol{\lambda}^\top \boldsymbol{h}(\boldsymbol{x})$. Now we have,

$$\begin{aligned}\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) &= \nabla f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}) \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) &= \boldsymbol{h}(\boldsymbol{x}).\end{aligned}$$

Karush-Kuhn-Tucker Conditions

Karush-Kuhn-Tucker conditions are often shortened as KKT-conditions. The KKT-conditions are an essential result in non-linear optimization that gives optimality conditions. We will not go in depth on how these results were achieved, but to present them accurately, we need the Lagrangian function. The Lagrangian will be in three variables, because of the additional constraints. First, consider the optimization problem,

$$\begin{aligned}\min f(\boldsymbol{x}) \quad \text{subject to} \quad & h_i(\boldsymbol{x}) = \mathbf{0} \quad (i \leqslant m), \\ & g_j(\boldsymbol{x}) \leqslant \mathbf{0} \quad (j \leqslant r).\end{aligned} \tag{2.7}$$

The Lagrangian function where we also include the inequality constraints, $\boldsymbol{g} = \{g_1, \dots, g_r\}$, becomes $\mathcal{L}: \mathbb{R}^N \times \mathbb{R}^m \times \mathbb{R}^r \to \mathbb{R}$,

$$\begin{aligned}\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i + h_i(\boldsymbol{x}) \sum_{j=1}^{r} \mu_i g_j(\boldsymbol{x}) \\ &= f(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{\mu}^\top \boldsymbol{g}(\boldsymbol{x}).\end{aligned}$$

And we also have,

$$\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \nabla f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}) + \sum_{j=1}^{r} \mu_j \nabla g_j(\boldsymbol{x}).$$

The indices of the active inequalities at $\boldsymbol{x}$ is denoted $A(\boldsymbol{x}) = \{j \leqslant r \mid g_j(\boldsymbol{x}) = \mathbf{0}\}$. A point $\boldsymbol{x}$ is called *regular* if $\{\nabla h_1(\boldsymbol{x}), \ldots, \nabla h_m(\boldsymbol{x})\} \cup \{\nabla g_j(\boldsymbol{x}) \mid i \in A(\boldsymbol{x})\}$ are linearly independent.

Theorem 2.20 (KKT). *Consider the optimization problem (2.7). Let $\boldsymbol{x}^*$ be a local minimum and assume it is a regular point. Then there are Lagrange multiplier vectors $\boldsymbol{\lambda}^* = (\lambda_1^*, \lambda_2^*, \ldots, \lambda_m^*)$ and $\boldsymbol{\mu}^* = (\mu_1^*, \mu_2^*, \ldots, \mu_r^*)$ such that*

$$\begin{aligned} \nabla_x L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) &= \mathbf{0}, \\ \mu_j^* &\geqslant \mathbf{0}, \qquad (j \leqslant r), \\ \mu_j^* &= \mathbf{0}, \qquad (j \notin A(\boldsymbol{x}^*)). \end{aligned}$$

In this chapter, we have presented necessary background theory for understanding and justifying the remainder of the thesis. We have derived ℓ_1-minimization to be the best choice for the algorithms that will be further investigated. We have introduced convexity and showed the advantages it have in optimization and derived the Lagrangian of a function. In the following chapter, we will discuss some necessary conditions for algorithms performing ℓ_1-minimization.

CHAPTER 3

Recovery Guarantees and Quality Measures

In this chapter, we present a recovery guarantee called the null space property, and two different quality measures: coherence and restricted isometry property. The guarantee and quality measures are both found from the measurement matrix, $\boldsymbol{A}$. These are necessary conditions for most of the algorithms in compressive sensing.

3.1 Null Space Property

The null space property is a condition that needs to be fulfilled to achieve exact recovery of sparse vectors and is defined as follows.

Definition 3.1. A matrix $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ satisfies the null space property relative to a set $S \subset [N]$ if

$$\|\boldsymbol{v}_S\|_1 < \|\boldsymbol{v}_{\bar{S}}\|_1 \quad \forall\, \boldsymbol{v} \in \ker \boldsymbol{A} \backslash \{\mathbf{0}\}.$$

$\boldsymbol{A}$ satisfies the null space property of order s if $\operatorname{card}(S) \leqslant s$, or, equivalently, if the vector supported on S is s-sparse.

The following theorem from [11, thm. 4.4] proves evidence that the null space property guarantees exact recovery of sparse vectors.

Theorem 3.2 (Null Space Property)**.** *Given a matrix $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ every vector $\boldsymbol{x} \in \mathbb{C}^N$ supported on a set S is the unique solution of (P_1) with $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ if and only if $\boldsymbol{A}$ satisfies the null space property relative to S.*

Proof. Let us begin with a fixed index set S, and assume that every vector $\boldsymbol{x} \in \mathbb{C}^N$ supported on S is the unique minimizer of $\|\boldsymbol{z}\|_1$ subject to $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$. Then we know that for every $\boldsymbol{v} \in \ker \boldsymbol{A} \backslash \{\mathbf{0}\}$, $\boldsymbol{v}_S$ is the unique minimizer of $\|\boldsymbol{z}\|_1$ subject to $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{A}\boldsymbol{v}_S$. Since $\boldsymbol{v} = \boldsymbol{v}_S + \boldsymbol{v}_{\bar{S}}$, $\boldsymbol{A}\boldsymbol{v} = \mathbf{0}$ and $\boldsymbol{v} \neq \mathbf{0}$ we have that $\boldsymbol{v}_S \neq -\boldsymbol{v}_{\bar{S}}$ and $\boldsymbol{A}(-\boldsymbol{v}_{\bar{S}}) = \boldsymbol{A}\boldsymbol{v}_S$. We can therefore conclude that the null space property relative to S holds, $\|\boldsymbol{v}_S\|_1 < \|\boldsymbol{v}_{\bar{S}}\|_1$.
For the other way, we assume the null space property relative to S holds, and consider the vectors $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{C}^N$ where $\boldsymbol{x}$ is supported on S, satisfying

$\boldsymbol{Az} = \boldsymbol{Ax}$. We define the vector $\boldsymbol{v} := \boldsymbol{x} - \boldsymbol{z} \in \ker \boldsymbol{A} \backslash \{\boldsymbol{0}\}$, which gives us $\boldsymbol{v}_S = \boldsymbol{x} - \boldsymbol{z}_S$ and $\boldsymbol{v}_{\bar{S}} = -\boldsymbol{z}_{\bar{S}}$. From this, we have:

$$\begin{aligned}\|\boldsymbol{x}\|_1 &\leqslant \|\boldsymbol{x} - \boldsymbol{z}_S\|_1 + \|\boldsymbol{z}_S\|_1\\ &= \|\boldsymbol{v}_S\|_1 + \|\boldsymbol{z}_S\|_1\\ &< \|\boldsymbol{v}_{\bar{S}}\|_1 + \|\boldsymbol{z}_S\|_1\\ &= \|-\boldsymbol{z}_{\bar{S}}\|_1 + \|\boldsymbol{z}_S\|_1\\ &= \|\boldsymbol{z}\|_1\end{aligned}$$

which establishes the minimality of $\|\boldsymbol{x}\|_1$. □

Proposition 3.3. *If the vector x in Theorem 3.2 is s-sparse, then $\boldsymbol{A}$ is said to satisfy the null space property of order s.*

Moreover, we have two more definitions: the stable null space property and robust null space property, which are both extensions of Definition 3.1. The stable null space property takes into consideration that for real-world signals, the vectors we are working with are not guaranteed to be sparse, only close to sparse. The definition are as presented:

Definition 3.4. A matrix $\boldsymbol{A} \in \mathbb{C}^{m\times N}$ satisfies the stable null space property with constant $0 < \rho < 1$ relative to a set $S \subset [N]$ if

$$\|\boldsymbol{v}_S\|_1 \leqslant \rho\|\boldsymbol{v}_{\bar{S}}\|_1 \quad \forall\, \boldsymbol{v} \in \ker \boldsymbol{A}\backslash\{\boldsymbol{0}\}.$$

With this definition, we are not guaranteed a unique solution of (P_1), but otherwise the same result as Theorem 3.2 holds. The robust null space property is a further strengthening of the null space property and allows for the measurement vector $\boldsymbol{y}$ to be an approximation of $\boldsymbol{Ax}$. The new definition is as follows:

Definition 3.5. A matrix $\boldsymbol{A} \in \mathbb{C}^{m\times N}$ satisfies the robust null space property with constant $0 < \rho < 1$ and $\tau > 0$ relative to a set $S \subset [N]$ if

$$\|\boldsymbol{v}_S\|_1 \leqslant \rho\|\boldsymbol{v}_{\bar{S}}\|_1 + \tau\|\boldsymbol{Av}\| \quad \forall\, \boldsymbol{v} \in \mathbb{C}^N$$

For the robust null space property, we are not guaranteed a solution for (P_1), but rather the similar convex problem with noise,

$$\min \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \|\boldsymbol{Ax} - \boldsymbol{b}\|_2 \leqslant \eta \qquad (P_{1,\eta})$$

Unfortunately, the computations required to see if a matrix $\boldsymbol{A}$ satisfies the null space property are NP-hard.

3.2 Coherence

When working with recovery algorithms, it is natural to include some measure of quality. Coherence is a measure of this, where smaller is better. Coherence is a property we find in the measurement matrices, and it is defined as presented below.

Definition 3.6. $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ is a matrix with ℓ_2-normalized columns. The coherence, $\mu = \mu(\boldsymbol{A})$, of the matrix is defined as

$$\mu := \max_{i \neq j} |\langle \boldsymbol{a}_i, \boldsymbol{a}_j \rangle|,$$

where the $\boldsymbol{a}_i$'s are the columns of $\boldsymbol{A}$.

A more general way of using the coherence is the ℓ_1-coherence function. This function also considers the sparsity, and is defined in the following definition.

Definition 3.7. $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ is a matrix with ℓ_2-normalized columns. The ℓ_1-coherence function μ_1 of the matrix $\boldsymbol{A}$ is defined for $s \in [N-1]$ by

$$\mu_1(s) := \max_{i \in [N]} \max \Big\{ \sum_{j \in S} |\langle \boldsymbol{a}_i, \boldsymbol{a}_j \rangle|, S \subset [N], \operatorname{card}(S) = s, i \notin S \Big\}.$$

Notice that $\mu_1(1) = \mu$, and

$$\mu \leqslant \mu_1(s) \leqslant s\mu.$$

This theorem from [11, Sect. 5.3] for orthogonal matching pursuit, guarantees exact recovery of s-sparse vectors when $\mu < 1/(2s-1)$

Theorem 3.8. $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ *is a matrix with* ℓ_2*-normalized columns. If,*

$$\mu_1(s) + \mu_1(s-1) < 1,$$

then every s*-sparse vector* $\boldsymbol{x} \in \mathbb{C}^N$ *is exactly recovered from the measurement vector* $\boldsymbol{y} = \boldsymbol{Ax}$ *after at most* s *iterations of orthogonal matching pursuit.*

Previously we saw that the coherence is a measure of the quality of the measurement matrix. The coherence is often sufficient, but when a more delicate measure is needed, we have the restricted isometry property that we will introduce next.

3.3 Restricted Isometry Property

Similar to coherence, the restricted isometry property is a measure of the quality of the measurement matrix. The measure is given by the restricted isometry constant, δ_s. δ_s is said to be of order s and involves all s-tuples of the columns of the measurement matrix, whereas coherence is easier to compute as it only uses pairs of columns. The restricted isometry property is therefore a finer measure of quality and is defined as follows.

Definition 3.9. The s'th restricted isometry constant $\delta_s = \delta_s(\boldsymbol{A})$ of a matrix $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ is the smallest $\delta \geqslant 0$ such that

$$(1-\delta)\|\boldsymbol{x}\|_2^2 \leqslant \|\boldsymbol{Ax}\|_2^2 \leqslant (1+\delta)\|\boldsymbol{x}\|_2^2$$

for all s-sparse vectors $\boldsymbol{x} \in \mathbb{C}^N$. Equivalently, it is given by

$$\delta_s = \max_{S \subset [N], \operatorname{card}(S) \leqslant s} \|\boldsymbol{A}_S^* \boldsymbol{A}_S - \boldsymbol{I}\|.$$

The restricted isometry property provides a theorem that guarantees exact s-sparse recovery in $12s$ iterations. The theorem reads as follows, from [11, thm. 6.25].

Theorem 3.10. *Suppose $\boldsymbol{A} \in \mathbb{C}^{m \times N}$ has restricted isometry constant,*

$$\delta_{13s} < \frac{1}{6}.$$

Then there exists a constant C, depending only on δ_{13s} such that for any $\boldsymbol{x} \in \mathbb{C}^N$, the target vector generated by the algorithm for orthonormal matching pursuit with $\boldsymbol{y} = \boldsymbol{Ax}$ satisfies,

$$\|\boldsymbol{y} - \boldsymbol{Ax}^{12s}\|_2 \leqslant C\|\boldsymbol{Ax}_{\bar{S}}\|_2$$

for any $S \subset [N]$ with $\operatorname{card}(S) \leqslant s$.

The null space property, coherence and restricted isometry property are all helpful tools in compressive sensing. These conditions are not applicable for the specialized ℓ_1-algorithms, but rather for the algorithms in compressive sensing for general linear programs. In the following chapter, we will take a closer look at these general algorithms.

CHAPTER 4

Algorithms for ℓ_1-Minimization

Among the algorithms in the field of compressive sensing, we find three main categories: optimization methods, greedy methods and thresholding-based methods. In this chapter we will give a short introduction to these categories and some algorithms related to compressive sensing and optimization.

Optimization Methods

For the optimization methods, we have algorithms known as basis pursuit and quadratically constrained basis pursuit, which we know from Chapter 2. These are the problems we wish to solve using the primal-dual algorithm and the NESTA algorithm, witch we will inspect in Chapters 5 and 6.

Greedy Methods

What greedy methods have in common is that they find the local optimal solution at each step and approximate a global optimal solution. The greedy algorithms work iteratively by first updating the support set with a new index and updating the target vector on the new support set. Among the greedy algorithms, we find: orthogonal matching pursuit (OMP), which we will examine later on, and compressive sampling matching pursuit.

Thresholding-Based Methods

Further, we have the thresholding-based methods. These algorithms have in common that they rely on the finding the s largest entries of the adjoint of the measurement matrix, $\boldsymbol{A}$, namely $\boldsymbol{A}^*$. After the largest entries have been located, it proceeds to update the target vector so that it best fits of the measurements. Among the thresholding algorithms, we find: basic thresholding, iterative hard thresholding and hard thresholding pursuit.

4.1 Simplex Method

The simplex method is an older algorithm for optimizing linear programs. We can visualize the algorithm geometrically, by considering a convex polytope generated by the constraints of the problem, and move along its edges in search of the optimal solution, which is a vertex in the polytope. A more formal presentation of the simplex method can be viewed as follows, by [12, sect. 6.1].

$$\max \boldsymbol{c}^\top \boldsymbol{x} \quad \text{subject to} \quad \begin{aligned} \boldsymbol{A}\boldsymbol{x} &= \boldsymbol{y}, \\ \boldsymbol{x} &\geqslant \boldsymbol{0}. \end{aligned}$$

Where $\boldsymbol{c}$ is the vector containing the coefficients of the objective function, x. The vector x contains both the *basic* variables of the function and the *non-basic* variables, also called *slack* variables. The matrix $\boldsymbol{A}$ and the vector $\boldsymbol{y}$ make up the constraints of the program.

The algorithm starts with a feasible solution and works iteratively to find a better solution until the optimal solution is found. In the initialization, we set the basic variables to zero and the non-basic are chosen, so that the equalities in the constraints are met. Then, iteratively interchange basic and non-basic variables to improve the objective function.

4.2 Orthogonal Matching Pursuit

Algorithm 1 Orthogonal Matching Pursuit

Input: measurement matrix $\boldsymbol{A}$ and vector $\boldsymbol{y}$
Initialization: $S^0 = \varnothing$, $\boldsymbol{x}^0 = \boldsymbol{0}$
Iteration: repeat until stopping criterion is met at $n = \bar{n}$:

$$S^{n+1} = S^n \cup \{j_{n+1}\}, \quad j_{n+1} := \underset{j \in [N]}{\operatorname{argmax}}\{|(\boldsymbol{A}^*(\boldsymbol{A}\boldsymbol{x}^n - \boldsymbol{y}))_j|\}, \tag{4.1}$$

$$\boldsymbol{x}^{n+1} = \underset{\boldsymbol{z} \in \mathbb{C}^N}{\operatorname{argmin}}\{\|\boldsymbol{A}\boldsymbol{z} - \boldsymbol{y}\|_2, \operatorname{supp}(\boldsymbol{z}) \subset S^{n+1}\} \tag{4.2}$$

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

The algorithm for orthogonal matching pursuit is given as above from [11, sect. 3.2].

The first step (4.1) finds the new index, j, that reduces the residual $\boldsymbol{A}\boldsymbol{x}^n - \boldsymbol{y}$. This is done greedily, that is, as much as possible at each iteration. With some clever calculations we can find that minimizing $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2$ is the same as maximizing $|(\boldsymbol{A}^*(\boldsymbol{A}\boldsymbol{x}^n - \boldsymbol{y}))_j|$. Once the index is found, it is included in the support set.

For the second step (4.2), the target vector, x, is updated as the best fit of the measurements with the new support set. This is the step that requires the most calculations as it performs a projection. We can increase the speed of this step, by using a QR-decomposition of $\boldsymbol{A}$.

The two steps run until the stopping criterion, $\bar{n}$, is met. This stopping criterion can be chosen in a few different ways. For instance, the most natural choice would be $\boldsymbol{A}\boldsymbol{x}^{\bar{n}} = \boldsymbol{y}$, but since we often either cannot achieve a perfect solution or do not need one, we may use the approximation $\|\boldsymbol{A}\boldsymbol{x}^{\bar{n}} - \boldsymbol{y}\|_2 \leqslant \eta$. However, if the coherence of $\boldsymbol{A}$ is small and satisfies Theorem 3.8, we know that we have exact recovery after at most s iterations and we can set $\bar{n} = s$. Similarly, if the restricted isometry constant of $\boldsymbol{A}$ satisfies Theorem 3.10, we have recovery after $12s$ iterations and can set $\bar{n} = 12s$.

From the coherence and the restricted isometry property, we can see that OMP works very well with sparse vectors, as the sparsity, s, defines the number of iterations. As mentioned in Chapter 1, algorithms designed for ℓ_1-minimization may be much faster than general algorithms. Hence, for larger s the primal-dual algorithm can be quite a lot faster than OMP.

In Figure 4.1, we have reconstructed a signal using Algorithm 1 for two different sparsities, $10\,\%$ and $20\,\%$. As expected, the algorithm performs significantly better with smaller sparsity. We have used the sparsity as the stopping criterion in both examples.

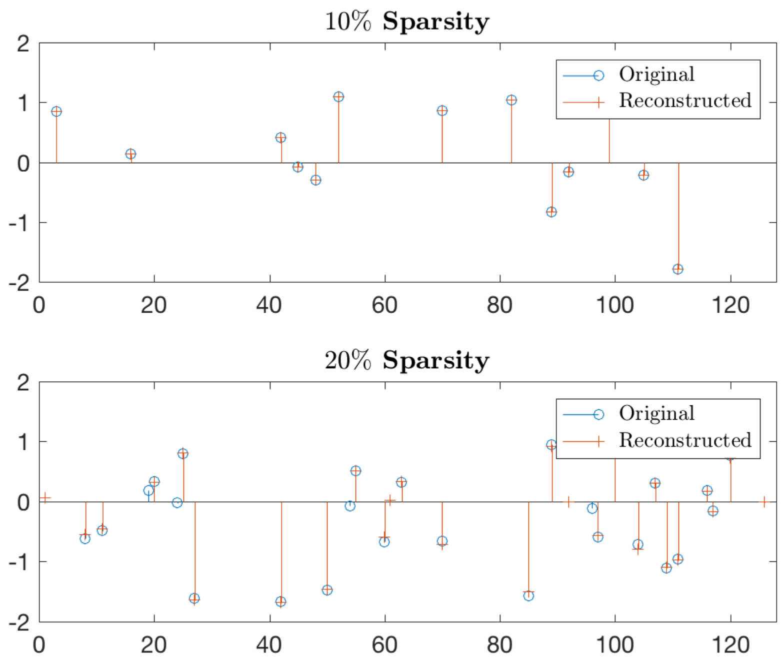

Figure 4.1: Reconstruction using OMP with sparsity of $10\,\%$ (top) and $20\,\%$ (bottom).

4.3 Spectral Projected Gradient for ℓ_1

The spectral projected gradient method for ℓ_1-minimization, or SPGL1 for short, from [3], is an iterative algorithm widely used when working with ℓ_1-minimization problems. The algorithm has shown to scale well to large problems and only depends on matrix-vector operations. SPGL1 mainly solves the quadratically constrained optimization problem (2.1),

$$\min_{\boldsymbol{x}\in\mathbb{C}^n} \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{y}\|_2 \leqslant \eta.$$

One of the reasons SPGL1 is set aside from the algorithms above is the approach. The algorithm solves the problem by solving a sequence of the LASSO problem (2.3) for varying values of τ,

$$\min_{\boldsymbol{x}\in\mathbb{C}^n} \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{y}\|_2 \quad \text{subject to} \quad \|\boldsymbol{x}\|_1 \leqslant \tau.$$

We recall from Proposition 2.5 that the two problems are equivalent. The intermediate LASSO problems are solved by using the spectral gradient (SPG) method. The SPG approach starts with an initial point $\boldsymbol{x}_0$ and iteratively updating the iterate $\boldsymbol{x}^n$ with the next point $\boldsymbol{x}^{n+1}$ found on the projected gradient path defined as,

$$\alpha \mapsto \mathcal{P}_\tau(\boldsymbol{x}^n - \alpha\nabla f(\boldsymbol{x}^n)).$$

Here, $f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{y}\|_2^2$ and its gradient becomes, $\nabla f(\boldsymbol{x}) = \boldsymbol{A}^*(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{y})$. The projection is an orthogonal projection onto the unit ℓ_1-ball,

$$\mathcal{P}_\tau(\boldsymbol{x}) = \operatorname{argmin}\{\|\boldsymbol{z}-\boldsymbol{x}\|_2 \mid \|\boldsymbol{z}\|_1 \leqslant \tau\}.$$

This projection can be computed efficiently and is closely related to the proximal mapping. This can be showed by considering the ℓ_∞-norm and an identity for the proximal mapping.

Once the intermediate LASSO problem is solved, what remains is to find the η such that we can find the optimal solution of the quadratically constrained problem. From the sequence of optimal solutions of the LASSO problem, SPGL1 provides the Pareto curve defined by the residual of LASSO,

$$\phi(\tau) = \|\boldsymbol{A}\boldsymbol{x}_\tau - \boldsymbol{y}\|_2,$$

where $\boldsymbol{x}_\tau$ is the optimal solution of LASSO with the corresponding τ. It is proved that the Pareto curve is convex and continuously differentiable at all points of interest. The algorithm uses Newton-based root finding methods for solving the equation $\phi(\tau) = \eta$ and finding the value of η and a minimizer of the quadratically constrained problem.

Now that we have presented some of the more general algorithms, in addition to SPGL1, we move on the algorithms specifically constructed to perform ℓ_1-minimization.

CHAPTER 5

Primal-Dual Algorithm

In this chapter, we will investigate the primal-dual algorithm. In Section 5.1 we introduce the concept of duality and derive the algorithm that will be further investigated in Section 5.2, where we also prove convergence.

5.1 Duality Theory

We will now introduce a concept in optimization known as *duality*. The idea is that for every optimization problem, there is a dual of the problem, and another perspective with which one can investigate the optimization problem. We will refer to the different problems as the primal and the dual. We note that the dual of the dual is the primal. Let us consider the general optimization problem,

$$\min f(\boldsymbol{x}) \quad \text{subject to} \quad \begin{aligned} h_i(\boldsymbol{x}) &= 0, \\ g_j(\boldsymbol{x}) &\leqslant 0. \end{aligned} \tag{5.1}$$

In order to derive the dual problem, we use the Lagrange function that will take the form,

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^{\mathsf{T}}\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{\mu}^{\mathsf{T}}\boldsymbol{g}(\boldsymbol{x}).$$

We define a new function d,

$$d(\boldsymbol{\lambda}, \boldsymbol{\mu}) := \inf_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}).$$

Note that this function is unbounded below and may therefore be equal to $-\infty$. d is a concave function because it is an infimum of a family of affine functions, one for each x. From this function d, we can formulate an important result, called *weak duality*,

Theorem 5.1 (Weak Duality)**.** *Assume x is feasible in the general optimization problem (5.1) and $\boldsymbol{\lambda} \in \mathbb{R}^m$, $\boldsymbol{\mu} \geqslant 0 \in \mathbb{R}^r$, then*

$$d(\boldsymbol{\lambda}, \boldsymbol{\mu}) \leqslant f(\boldsymbol{x}).$$

Proof.

$$\begin{aligned} d(\boldsymbol{\lambda}, \boldsymbol{\mu}) &\leqslant \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \\ &= f(\boldsymbol{x}) + \boldsymbol{\lambda}^{\mathsf{T}}\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{\mu}^{\mathsf{T}}\boldsymbol{g}(\boldsymbol{x}) \\ &\leqslant f(\boldsymbol{x}). \end{aligned}$$

□

What Theorem 5.1 says, is that the function $d(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is a lower bound for the function $f(\boldsymbol{x})$. In order to find the best lower bound, that is, the largest lower bound, for $f(\boldsymbol{x})$, we formulate the problem,

$$\max d(\boldsymbol{\lambda}, \boldsymbol{\mu}) \quad \text{subject to} \quad \boldsymbol{\mu} \geqslant 0.$$

This is the problem we refer to as the *dual problem*. Before the introduction of the primal-dual algorithm, we need to formulate the general form of the primal and dual problem, as the algorithm uses. Then we will show that we can formulate it from the ℓ_1-norm. We recall that the convex conjugate function is defined as,

$$F^*(\boldsymbol{y}) := \sup_{\boldsymbol{x} \in \mathbb{R}^N} \{\langle \boldsymbol{x}, \boldsymbol{y} \rangle - F(\boldsymbol{x})\}.$$

The general form of the primal problem is,

$$\min_{\boldsymbol{x} \in \mathbb{C}^N} F(\boldsymbol{A}\boldsymbol{x}) + G(\boldsymbol{x}), \tag{5.2}$$

and for the dual problem we use the form,

$$\max_{\boldsymbol{\xi} \in \mathbb{C}^m} -F^*(\boldsymbol{\xi}) - G^*(-\boldsymbol{A}^*\boldsymbol{\xi}). \tag{5.3}$$

From this, the saddle-point problem takes the form,

$$\min_{x \in \mathbb{C}^N} \max_{\boldsymbol{\xi} \in \mathbb{C}^m} \operatorname{Re}\langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{\xi} \rangle + G(\boldsymbol{x}) - F^*(\boldsymbol{\xi}). \tag{5.4}$$

We will now show that for a quadratic constrained ℓ_1-minimization problem (the procedure is similar for ℓ_1-minimization problems with $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}$), we can formulate (5.2) and (5.3).

$$\min_{\boldsymbol{x} \in \mathbb{C}^N} \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2 \leqslant \eta \tag{5.5}$$

This translates to the general primal problem by letting $G(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ and $F(\boldsymbol{x})$ as,

$$F(\boldsymbol{x}) = \chi_{B(\boldsymbol{y},\eta)}(\boldsymbol{x}) = \begin{cases} 0, & \|\boldsymbol{x} - \boldsymbol{y}\|_2 \leqslant \eta \\ \infty, & \text{else,} \end{cases}$$

where χ is the characteristic function of the set $B(\boldsymbol{y}, \eta)$, which is the closed ball defined as,

$$B(\boldsymbol{y}, \eta) = \{\boldsymbol{x} \in \mathbb{R}^N \mid d(\boldsymbol{x}, \boldsymbol{y}) \leqslant \eta\}.$$

With these functions, we see that (5.2) satisfies (5.5). For the dual, we first formulate the equivalent problem of (5.2) by substituting $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$.

$$\min_{\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{z} \in \mathbb{R}^m} F(\boldsymbol{z}) + G(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} - \boldsymbol{z} = \boldsymbol{0}.$$

With this problem, we follow the standard approach for finding the dual. This is done by finding the infimum of the Lagrangian of the primal problem.

$$\begin{aligned}
&\inf_{\boldsymbol{x},\boldsymbol{z}}\{F(\boldsymbol{z}) + G(\boldsymbol{x}) - \langle \boldsymbol{A}^*\boldsymbol{\xi}, \boldsymbol{x}\rangle - \langle \boldsymbol{\xi}, \boldsymbol{z}\rangle\} \\
&= -\sup_{\boldsymbol{x},\boldsymbol{z}}\{-F(\boldsymbol{z}) - G(\boldsymbol{x}) + \langle \boldsymbol{A}^*\boldsymbol{\xi}, \boldsymbol{x}\rangle + \langle \boldsymbol{\xi}, \boldsymbol{z}\rangle\} \\
&= -\sup_{\boldsymbol{z}}\{\langle \boldsymbol{\xi}, \boldsymbol{z}\rangle - F(\boldsymbol{z})\} - \sup_{\boldsymbol{x}}\{\langle \boldsymbol{A}^*\boldsymbol{\xi}, \boldsymbol{x}\rangle - G(\boldsymbol{x})\} \\
&= -F^*(\boldsymbol{\xi}) - G^*(-\boldsymbol{A}^*\boldsymbol{\xi}),
\end{aligned}$$

where F^* and G^* are the convex conjugates as defined in Definition 2.17, which have the form,

$$\begin{aligned}
F^*(\boldsymbol{\xi}) &= \sup_{\boldsymbol{x}\in\mathbb{R}^N}\{\langle \boldsymbol{x}, \boldsymbol{\xi}\rangle - \chi_{B(\boldsymbol{y},\eta)}(\boldsymbol{\xi})\} \\
&= \sup_{\boldsymbol{x}:\|\boldsymbol{x}-\boldsymbol{y}\|_2\leqslant\eta} \operatorname{Re}\langle \boldsymbol{x}, \boldsymbol{\xi}\rangle \\
&= \operatorname{Re}\langle \boldsymbol{y}, \boldsymbol{\xi}\rangle + \eta\|\boldsymbol{\xi}\|_2.
\end{aligned}$$

For G^*, we first notice that $\langle \boldsymbol{x}, \boldsymbol{y}\rangle \leqslant \|\boldsymbol{x}\|_1\|\boldsymbol{y}\|_\infty$, which yields,

$$\begin{aligned}
G^*(\boldsymbol{y}) &= \sup_{\boldsymbol{x}\in\mathbb{R}^N}\{\langle \boldsymbol{x}, \boldsymbol{y}\rangle - \|\boldsymbol{x}\|_1\} \\
&\leqslant \sup_{\boldsymbol{x}\in\mathbb{R}^N}\{\|\boldsymbol{x}\|_1(\|\boldsymbol{y}\|_\infty - 1)\}.
\end{aligned}$$

The supremum of the expression when $\|\boldsymbol{y}\|_\infty \leqslant 1$ is 0, as we can see by letting $x = 0$, otherwise the expression would become negative. We notice the function is unbounded above when $\|\boldsymbol{y}\|_\infty > 1$, so we can rewrite G^* to be the characteristic function,

$$G^*(\boldsymbol{y}) = \chi_{B(\boldsymbol{y},\eta)}(\boldsymbol{y}) = \begin{cases} 0, & \|\boldsymbol{y}\|_\infty \leqslant 1, \\ \infty, & \text{else.} \end{cases}$$

We have now formulated both the primal and dual general problems from the quadratically constrained problem, (5.5), and we proceed to formulate the primal-dual algorithm. For this algorithm to be efficient, it requires that the proximal mappings are easy to evaluate. We start with the following proposition that states that a fixed point of the algorithm is an optimal point of the primal and dual problem.

Proposition 5.2. $(x^\sharp, \xi^\sharp)$ *is a fixed point of the primal-dual algorithm if and only if it is a saddle point of (5.4), implying that it is an optimal point of (5.2) and (5.3).*

Proof. From Proposition 2.16, we see that a fixed point $(x^\sharp, \xi^\sharp)$ satisfies,

$$\begin{aligned}
\boldsymbol{\xi}^\sharp + \sigma\boldsymbol{A}\boldsymbol{x}^\sharp &\in \boldsymbol{\xi}^\sharp + \sigma\partial F^*(\boldsymbol{\xi}^\sharp) \\
\boldsymbol{x}^\sharp - \tau\boldsymbol{A}^*\boldsymbol{\xi}^\sharp &\in \boldsymbol{x}^\sharp + \tau\partial G(\boldsymbol{x}^\sharp).
\end{aligned}$$

From Proposition 2.13 we also see that this implies,

$$\boldsymbol{0} \in -\boldsymbol{A}\boldsymbol{x}^\sharp + \partial F^*(\boldsymbol{\xi}^\sharp) \quad \text{and} \quad \boldsymbol{0} \in \boldsymbol{A}^*\boldsymbol{\xi}^\sharp + \partial G(\boldsymbol{x}^\sharp),$$

which again implies that $\boldsymbol{\xi}^\sharp$ is a maximizer of (5.6) and $\boldsymbol{x}^\sharp$ is a minimizer of (5.7).

$$\operatorname{Re}\langle \boldsymbol{A}\boldsymbol{x}^\sharp, \boldsymbol{\xi}\rangle + G(\boldsymbol{x}^\sharp) - F^*(\boldsymbol{\xi}) \tag{5.6}$$

$$\operatorname{Re}\langle \boldsymbol{x}, \boldsymbol{A}^*\boldsymbol{\xi}^\sharp\rangle + G(\boldsymbol{x}) - F^*(\boldsymbol{\xi}^\sharp). \tag{5.7}$$

We notice that this is equivalent to $(\boldsymbol{x}^\sharp, \boldsymbol{\xi}^\sharp)$ being a saddle point of (5.4), that is, an optimal solution. We have now proved one way of the equivalence, the other way, saddle point $\Rightarrow$ fixed point, takes similar arguments and we will not go through the steps. $\square$

We recall that Moreau's identity, (2.5), is,

$$P_{\tau F^*}(\boldsymbol{z}) + \tau P_{\tau^{-1}F}(\boldsymbol{z}/\tau) = \boldsymbol{z}.$$

From this identity we see that the proximal mapping of F^* is easy to compute once the mapping of F is computed, and vice versa. The proximal mapping of our primal function F takes the form,

$$\begin{aligned} P_F(\sigma; \boldsymbol{\xi}) &= \operatorname*{argmin}_{\boldsymbol{z}\in\mathbb{C}^N} \sigma F(\boldsymbol{z}) + \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{\xi}\|_2^2 \\ &= \operatorname*{argmin}_{\boldsymbol{z}\in\mathbb{C}^N} \chi_{B(\boldsymbol{y},\eta)}(\boldsymbol{z}) + \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{\xi}\|_2^2 \\ &= \operatorname*{argmin}_{\boldsymbol{z}\in\mathbb{C}^N : \|\boldsymbol{z}-\boldsymbol{y}\|_2 \leqslant \eta} \|\boldsymbol{z} - \boldsymbol{\xi}\|_2. \end{aligned}$$

The final transition before we get a precise expression for the proximal mapping of F, we find graphically. We wish to find the shortest path from $\boldsymbol{z}$, contained in a ball around $\boldsymbol{y}$ with radius η, to $\boldsymbol{\xi}$. $\boldsymbol{\xi}$ is given, and we need to find $\boldsymbol{z}$. If $\boldsymbol{\xi}$ is contained in the same ball, $\boldsymbol{z} = \boldsymbol{\xi}$. If $\boldsymbol{\xi}$ is not contained in the ball, $\boldsymbol{z}$ must lie on the line connecting $\boldsymbol{\xi}$ and the center of the ball, $\boldsymbol{y}$, and on the edge of the ball. From this, we can derive the expression,

$$\operatorname*{argmin}_{\boldsymbol{z}\in\mathbb{C}^N : \|\boldsymbol{z}-\boldsymbol{y}\|_2 \leqslant \eta} \|\boldsymbol{z} - \boldsymbol{\xi}\|_2 = \begin{cases} \boldsymbol{\xi}, & \|\boldsymbol{\xi} - \boldsymbol{y}\|_2 \leqslant \eta, \\ \boldsymbol{y} + \frac{\eta}{\|\boldsymbol{\xi}-\boldsymbol{y}\|_2}(\boldsymbol{\xi} - \boldsymbol{y}), & \text{else.} \end{cases}$$

Using (2.5) we find the proximal mapping of F^* to b,

$$\begin{aligned} P_{F^*}(\sigma; \boldsymbol{\xi}) &= \boldsymbol{\xi} - \sigma P_F(\sigma^{-1}; \boldsymbol{\xi}/\sigma) \\ &= \begin{cases} \boldsymbol{0}, & \|\boldsymbol{\xi} - \sigma\boldsymbol{y}\|_2 \leqslant \sigma\eta, \\ \boldsymbol{\xi} - \left(\sigma\boldsymbol{y} + \frac{\eta\sigma}{\|\boldsymbol{\xi}/\sigma-\boldsymbol{y}\|_2}(\boldsymbol{\xi}/\sigma - \boldsymbol{y})\right), & \text{else.} \end{cases} \\ &= \begin{cases} \boldsymbol{0}, & \|\boldsymbol{\xi} - \sigma\boldsymbol{y}\|_2 \leqslant \sigma\eta, \\ \left(1 - \frac{\eta\sigma}{\|\boldsymbol{\xi}-\sigma\boldsymbol{y}\|_2}\right)(\boldsymbol{\xi} - \sigma\boldsymbol{y}), & \text{else.} \end{cases} \end{aligned}$$

Now we have an expression of the proximal mapping of the function F^*, and we move on to finding the mapping of the function, $G(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$.

$$P_G(\tau; \boldsymbol{z}) = \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{C}^N} \tau\|\boldsymbol{x}\|_1 + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2.$$

This problem is separable and we can find an expression by solving the following problem,

$$P_G'(\tau; z_i) = \operatorname*{argmin}_{x_i \in \mathbb{C}} \tau |x_i| + \frac{1}{2}(x_i - z_i)^2.$$

We will now investigate the three possibilities, $x_i > 0$, $x_i < 0$ and $x_i = 0$. By setting the derivative equal to zero we find

$$\begin{aligned} &x_i > 0, \\ &\qquad x_i = z_i - \tau, \qquad z_i > \tau. \\ &x_i < 0, \\ &\qquad x_i = z_i - \tau, \qquad z_i < \tau. \end{aligned}$$

The derivative when $x_i = 0$, the subdifferential of the absolute value at 0 are, as we recall from the example of subdifferential on Page 12, the set $[-1, 1]$. We can now piece together an alternate expression for the proximal mapping,

$$P_G'(\tau, z_i) = \begin{cases} \operatorname{sgn}(z_i)(|z_i| - \tau), & |z_i| \geqslant \tau, \\ 0, & \text{else.} \end{cases}$$

Since we started by separating the problem, we find that the proximal mapping can be written as,

$$P_G(\tau; \boldsymbol{z})_j = P_G'(\tau; z_j), \qquad j \in [N].$$

Now that we have formulated both of the proximal mappings, we find from Proposition 5.2 and Proposition 2.16 that the primal-dual algorithm reads,

$$\begin{aligned} \boldsymbol{\xi}^{n+1} &= P_{F^*}(\sigma; \boldsymbol{\xi}^n + \sigma \boldsymbol{A}\bar{\boldsymbol{x}}^n) \\ &= \begin{cases} \boldsymbol{0}, & \|\sigma^{-1}\boldsymbol{\xi}^n + \boldsymbol{A}\bar{\boldsymbol{x}}^n - \boldsymbol{y}\|_2 \leqslant \eta, \\ \left(1 - \frac{\eta\sigma}{\|\boldsymbol{\xi}^n + \sigma(\boldsymbol{A}\bar{\boldsymbol{x}}^n - \boldsymbol{y})\|_2}\right)(\boldsymbol{\xi}^n + \sigma(\boldsymbol{A}\bar{\boldsymbol{x}} - \boldsymbol{y})), & \text{else.} \end{cases} \\ \boldsymbol{x}^{n+1} &= P_G(\tau; \boldsymbol{x}^n - \tau \boldsymbol{A}^* \boldsymbol{\xi}^{n+1}). \end{aligned}$$

When updating the solutions, the best and most natural choice would be to let $\bar{\boldsymbol{y}} = \boldsymbol{y}^{n+1}$ and $\bar{\boldsymbol{x}} = \boldsymbol{x}^{n+1}$. This is not a feasible solution, as we require solutions that are not yet computed. An alternate feasible choice is to keep $\bar{\boldsymbol{y}} = \boldsymbol{y}^{n+1}$, and let $\bar{\boldsymbol{x}} = \boldsymbol{x}^n$. However, choosing $\bar{\boldsymbol{x}} = \boldsymbol{x}^n + \theta(\boldsymbol{x}^n - \boldsymbol{x}^{n-1})$, using both current and previous iterates has shown good results. The last step of the algorithm with this update becomes,

$$\bar{\boldsymbol{x}}^{n+1} = \boldsymbol{x}^{n+1} + \theta(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n)$$

5.2 Primal-Dual Algorithm

In the following section, we have the algorithm that we just derived, formulated as in [11, sect. 15.2].

Algorithm 2 Primal-Dual Algorithm

```
$A \in \mathbb{C}^{m\times n}$, $F, G$ are convex functions.
$\theta \in [0,1]$, $\tau, \sigma > 0$ such that $\sigma\tau\|\boldsymbol{A}\|^2 < 1$
Initial values: $\boldsymbol{x}^0 \in \mathbb{C}^N$, $\boldsymbol{\xi}^0 \in \mathbb{C}^m$, $\bar{\boldsymbol{x}}^0 = \boldsymbol{x}^0$.
Iterations: repeat until the stopping criterion is met, $n = \bar{n}$
    $\boldsymbol{\xi}^{n+1} := P_{F^*}(\sigma; \boldsymbol{\xi}^n + \sigma \boldsymbol{A}\bar{\boldsymbol{x}}^n)$
    $\boldsymbol{x}^{n+1} := P_G(\tau; \boldsymbol{x}^n - \tau \boldsymbol{A}^* \boldsymbol{\xi}^{n+1})$
    $\bar{\boldsymbol{x}}^{n+1} := \boldsymbol{x}^{n+1} + \theta(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n)$
Output:
Approximation, $\boldsymbol{\xi}^\sharp = \boldsymbol{\xi}^{\bar{n}}$, to a solution of the dual problem.
Approximation, $\boldsymbol{x}^\sharp = \boldsymbol{x}^{\bar{n}}$, to a solution of the primal problem.
```

The norm, $\|\boldsymbol{A}\|$, used here is the standard operator norm that finds the largest singular value of the matrix $\boldsymbol{A}$ on ℓ_2. We will now go through the theorem that guarantees convergence of the primal-dual algorithm. We will consider the algorithm with parameter $\theta = 1$ as, in this case, it is easier to find convergence estimates of the algorithm. Other values of θ have different advantages, for smooth cases, $\theta = 0$ has shown to provide good solutions. For $\theta = 0$, the algorithm is known as the Arrow-Hurwicz algorithm. The theorem from [11, thm. 15.8] reads as follows.

Theorem 5.3. *Assume that the problem (5.4) has a saddle point. Choose $\theta = 1$ and $\sigma, \tau > 0$ such that $\sigma\tau\|\boldsymbol{A}\|^2 < 1$. Let $(\boldsymbol{x}^n, \bar{\boldsymbol{x}}^n, \boldsymbol{\xi}^n)$ be the sequence generated by the primal dual algorithm. Then the sequence $(\boldsymbol{x}^n, \boldsymbol{\xi}^n)$ converges to a saddle point $(\boldsymbol{x}^\sharp, \boldsymbol{\xi}^\sharp)$ of (5.4). In particular, $(\boldsymbol{x}^n)$ converges to a minimizer of (5.2).*

The proof of this theorem is quite long and contains many calculations. We will not cover the entire proof in detail, but merely the important parts. For starters, we will be using the Lagrangian,

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\xi}) = \mathrm{Re}\langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{\xi}\rangle + G(\boldsymbol{x}) - F^*(\boldsymbol{\xi}).$$

In order to make parts of the proof simpler, notation-wise at least, we introduce the two expressions called the *divided difference* and the *discrete derivative* respectively,

$$\Delta_\tau \boldsymbol{u}^n = \frac{\boldsymbol{u}^n - \boldsymbol{u}^{n-1}}{\tau},$$
$$\Delta_\tau \|\boldsymbol{u}^{n+1}\|_2^2 = \frac{\|\boldsymbol{u}^{n+1}\|_2^2 - \|\boldsymbol{u}^n\|_2^2}{\tau}.$$

From these expressions, we derive two identities that will be useful later in the proof. First,

$$2\mathrm{Re}\langle \Delta_\tau \boldsymbol{u}^n, \boldsymbol{u}^n - \boldsymbol{u}\rangle = \Delta_\tau \|\boldsymbol{u} - \boldsymbol{u}^n\|_2^2 + \tau \|\Delta_\tau \boldsymbol{u}^n\|_2^2. \tag{5.8}$$

This can easily be showed by using the properties of the inner product and using the divided difference and discrete derivative. The second identity is called *discrete integration by parts*,

$$\tau \sum_{n=1}^{M} \big(\langle \Delta_\tau \boldsymbol{u}^n, \boldsymbol{v}^n \rangle + \langle \boldsymbol{u}^{n-1}, \Delta_\tau \boldsymbol{v}^n \rangle \big) = \langle \boldsymbol{u}^M, \boldsymbol{v}^M \rangle - \langle \boldsymbol{u}^0, \boldsymbol{v}^0 \rangle. \tag{5.9}$$

This is proved similarly as the first identity, in addition to using the telescopic identity for proving that the sum can be written as,

$$\tau \sum_{n=1}^{M} \Delta_\tau \langle \boldsymbol{u}^n, \boldsymbol{v}^n \rangle = \langle \boldsymbol{u}^M, \boldsymbol{v}^M \rangle - \langle \boldsymbol{u}^0, \boldsymbol{v}^0 \rangle.$$

Next, we have another inequality that will further help us with the proof. If we have a sequence $(\boldsymbol{x}^n, \bar{\boldsymbol{x}}^n, \boldsymbol{\xi}^n)_{n \geqslant 0}$ generated by the primal-dual algorithm, then we have for any $n \geqslant 1$,

$$\begin{aligned} &\frac{1}{2}\Delta_\sigma \|\boldsymbol{\xi} - \boldsymbol{\xi}^n\|_2^2 + \frac{1}{2}\Delta_\tau \|\boldsymbol{x} - \boldsymbol{x}^n\|_2^2 + \frac{\sigma}{2}\|\Delta_\sigma \boldsymbol{\xi}^n\|_2^2 + \frac{\tau}{2}\|\Delta_\tau \boldsymbol{x}^n\|_2^2 \\ \leqslant &\mathcal{L}(\boldsymbol{x}, \boldsymbol{\xi}^n) - \mathcal{L}(\boldsymbol{x}^n, \boldsymbol{\xi}) + \mathrm{Re}\langle \boldsymbol{A}(\boldsymbol{x}^n - \bar{\boldsymbol{x}}^{n-1}), \boldsymbol{\xi} - \boldsymbol{\xi}^n \rangle. \end{aligned} \tag{5.10}$$

To show this, we use a property of proximal mappings shown in Proposition 2.16, the subdifferentials ∂F^* and ∂G, as defined in Definition 2.12. The use of this together with equation (5.8), we find the desired result.

Further, we use yet another inequality that we derive from summing over (5.10) and reformulating the expression we get by using (5.9), we find

$$\begin{aligned} &\sum_{n=1}^{M} (\mathcal{L}(\boldsymbol{x}, \boldsymbol{\xi}^n) - \mathcal{L}(\boldsymbol{x}^n, \boldsymbol{\xi})) + \frac{1}{2\tau}\|\boldsymbol{x} - \boldsymbol{x}^M\|_2^2 + \frac{1 - \sqrt{\sigma\tau}\|\boldsymbol{A}\|^2}{2\sigma}\|\boldsymbol{\xi} - \boldsymbol{\xi}^M\|_2^2 \\ +&\frac{1 - \sqrt{\sigma\tau}\|\boldsymbol{A}\|}{2\tau} \sum_{n=1}^{M-1} \|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|_2^2 + \frac{1 - \sqrt{\sigma\tau}\|\boldsymbol{A}\|}{2\sigma} \sum_{n=1}^{M} \|\boldsymbol{\xi}^n - \boldsymbol{\xi}^{n-1}\|_2^2 \\ \leqslant &\frac{1}{2\tau}\|\boldsymbol{x} - \boldsymbol{x}^0\|_2^2 + \frac{1}{2\sigma}\|\boldsymbol{\xi} - \boldsymbol{\xi}^0\|_2^2. \end{aligned} \tag{5.11}$$

We will include some important parts of the proof where we see that,

$$\begin{aligned} &\tau^2 \sum_{n=1}^{M} \mathrm{Re}\langle \boldsymbol{A}\Delta_\tau^2 \boldsymbol{x}^n, \boldsymbol{\xi} - \boldsymbol{\xi}^n \rangle \\ =&\sigma\tau \sum_{n=1}^{M} \mathrm{Re}\langle \boldsymbol{A}\Delta_\tau \boldsymbol{x}^{n-1}, \Delta_\sigma \boldsymbol{\xi}^n \rangle + \tau \mathrm{Re}\langle \Delta_\tau \boldsymbol{x}^M, \boldsymbol{A}^*(\boldsymbol{\xi} - \boldsymbol{\xi}^M) \rangle. \end{aligned} \tag{5.12}$$

Here we have used (5.9) and the fact that $\Delta_\tau \boldsymbol{x}^0 = 0$ (because we define $\boldsymbol{x}^{-1}$ to be equal to $\boldsymbol{x}^0$). Furthermore, we find the inequality,

$$\begin{aligned} &\sigma\tau \mathrm{Re}\langle \boldsymbol{A}\Delta_\tau \boldsymbol{x}^{n-1}, \Delta_\sigma \boldsymbol{\xi}^n \rangle \\ \leqslant &\frac{\sqrt{\sigma\tau}\|\boldsymbol{A}\|}{2\tau}\|\boldsymbol{x}^{n-1} - \boldsymbol{x}^{n-1}\|_2^2 + \frac{\sqrt{\sigma\tau}\|\boldsymbol{A}\|}{2\sigma}\|\boldsymbol{\xi}^n - \boldsymbol{\xi}^{n-1}\|_2^2. \end{aligned} \tag{5.13}$$

Finally, before we start with the actual proof, we will find a boundary for the iterates $(\boldsymbol{x}^n, \boldsymbol{\xi}^n)$. In order to do so, we need to see that, given a saddle point $(\boldsymbol{x}^\sharp, \boldsymbol{\xi}^\sharp)$, the sums $\mathcal{L}(\boldsymbol{x}^n, \boldsymbol{\xi}^\sharp) - \mathcal{L}(\boldsymbol{x}^\sharp, \boldsymbol{\xi}^n)$ are all non-negative. We can show this by writing out the Lagrangian function for each of the expressions. We find the boundary,

$$\frac{1}{2\sigma}\|\boldsymbol{\xi}^\sharp - \boldsymbol{\xi}^M\|_2^2 + \frac{1}{2\tau}\|\boldsymbol{x}^\sharp - \boldsymbol{x}^M\|_2^2 \leqslant C\Big(\frac{1}{2\sigma}\|\boldsymbol{\xi}^\sharp - \boldsymbol{\xi}^0\|_2^2 + \frac{1}{2\tau}\|\boldsymbol{x}^\sharp - \boldsymbol{x}^0\|_2^2\Big),$$

where the constant $C = \left(1 - \sigma\tau\|\boldsymbol{A}\|^2\right)^{-1}$.

We are now ready to dive into the proof of Theorem 5.3.

Proof. Since the sequence $(\boldsymbol{x}^n, \boldsymbol{\xi}^n)$ is bounded, there must exist a convergent subsequence, $(\boldsymbol{x}^{n_k}, \boldsymbol{\xi}^{n_k}) \to (\boldsymbol{x}^\circ, \boldsymbol{\xi}^\circ)$ as $k \to \infty$. From the fact that the sum of the Lagrangians are non-negative when $(\boldsymbol{x}^\sharp, \boldsymbol{\xi}^\sharp)$ is a saddle point, we can make all terms non-negative in (5.11) by choosing a saddle point. Now, since all terms are non-negative we find that,

$$\frac{1 - \sqrt{\sigma\tau}\|\boldsymbol{A}\|}{2\tau}\sum_{n=1}^{M-1}\|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|_2^2 \leqslant \frac{1}{2\tau}\|\boldsymbol{x} - \boldsymbol{x}^0\|_2^2 + \frac{1}{2\sigma}\|\boldsymbol{\xi} - \boldsymbol{\xi}^0\|_2^2.$$

Here, the right-hand side does not depend on M, and the term $\sqrt{\sigma\tau}\|\boldsymbol{A}\| < 1$, this implies that $\|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|_2 \to 0$ as $n \to \infty$. We can do similar calculations and show the same for $\|\boldsymbol{\xi}^n - \boldsymbol{\xi}^{n-1}\|_2 \to 0$. Based on this, we see that also the subsequence $(\boldsymbol{x}^{n_k}, \boldsymbol{\xi}^{n_k}) \to (\boldsymbol{x}^\circ, \boldsymbol{\xi}^\circ)$, meaning $(\boldsymbol{x}^\circ, \boldsymbol{\xi}^\circ)$ must be a fixed point of the algorithm. By Proposition 5.2, it is a saddle point and an optimal point. We choose $(\boldsymbol{x}, \boldsymbol{\xi}) = (\boldsymbol{x}^\circ, \boldsymbol{\xi}^\circ)$ for saddle point qualities, and sum over (5.10) from n_k to M. Since $\mathcal{L}(\boldsymbol{x}^n, \boldsymbol{\xi}^\circ) - \mathcal{L}(\boldsymbol{x}^\circ, \boldsymbol{\xi}^n) \geqslant 0 \Rightarrow \mathcal{L}(\boldsymbol{x}^\circ, \boldsymbol{\xi}^n) - \mathcal{L}(\boldsymbol{x}^n, \boldsymbol{\xi}^\circ) \leqslant 0$ and we find,

$$\begin{aligned}
&\frac{1}{2\sigma}(\|\boldsymbol{\xi}^\circ - \boldsymbol{\xi}^M\|_2^2 - \|\boldsymbol{\xi}^\circ - \boldsymbol{\xi}^{n_k}\|_2^2) + \frac{1}{2\tau}(\|\boldsymbol{x}^\circ - \boldsymbol{x}^M\|_2^2 - \|\boldsymbol{x}^\circ - \boldsymbol{x}^{n_k}\|_2^2)\\
&+\frac{1}{2\sigma}\sum_{n=n_k}^{M}\|\boldsymbol{\xi}^n - \boldsymbol{\xi}^{n-1}\|_2^2 + \frac{1}{2\tau}\sum_{n=n_k}^{M}\|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|_2^2\\
&\leqslant \tau^2\sum_{n=n_k}^{M}\mathrm{Re}\langle \boldsymbol{A}\Delta_\tau^2\boldsymbol{x}^n, \boldsymbol{\xi}^\circ - \boldsymbol{\xi}^n\rangle.
\end{aligned}$$

Similar to (5.12), we use (5.9) and find,

$$\begin{aligned}
&\tau^2\sum_{n=n_k}^{M}\mathrm{Re}\langle \boldsymbol{A}\Delta_\tau^2\boldsymbol{x}^n, \boldsymbol{\xi}^\circ - \boldsymbol{\xi}^n\rangle\\
&=\sigma\tau\sum_{n=n_k}^{M}\mathrm{Re}\langle \boldsymbol{A}\Delta_\tau\boldsymbol{x}^{n-1}, \Delta_\sigma\boldsymbol{\xi}^n\rangle + \tau\mathrm{Re}\langle \boldsymbol{A}\Delta_\tau\boldsymbol{x}^M, \boldsymbol{\xi}^\circ - \boldsymbol{\xi}^M\rangle\\
&\quad - \tau\mathrm{Re}\langle \boldsymbol{A}\Delta_\tau\boldsymbol{x}^{n_k-1}, \boldsymbol{\xi}^\circ - \boldsymbol{\xi}^{n_k}\rangle.
\end{aligned}$$

Finally, with all our findings we can rewrite (5.13) and get,

$$\begin{aligned}
&\frac{1}{2\sigma}\|\boldsymbol{\xi}^\circ - \boldsymbol{\xi}^M\|_2^2 + \frac{1}{2\tau}\|\boldsymbol{x}^\circ - \boldsymbol{x}^M\|_2^2 + \frac{1-\sqrt{\sigma\tau}\|\boldsymbol{A}\|}{2\sigma}\sum_{n=n_k}^{M}\|\boldsymbol{\xi}^n - \boldsymbol{\xi}^{n-1}\|_2^2\\
&+\frac{1-\sqrt{\sigma\tau}\|\boldsymbol{A}\|}{2\tau}\sum_{n=n_k}^{M-1}\|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|_2^2\\
&+\frac{1}{2\tau}(\|\boldsymbol{x}^M - \boldsymbol{x}^{M-1}\|_2^2 - \sqrt{\sigma\tau}\|\boldsymbol{A}\|\,\|\boldsymbol{x}^{n_k-1} - \boldsymbol{x}^{n_k-1}\|_2^2)\\
&-\mathrm{Re}\langle \boldsymbol{A}(\boldsymbol{x}^M - \boldsymbol{x}^{M-1}), \boldsymbol{\xi}^\circ - \boldsymbol{\xi}^M\rangle + \mathrm{Re}\langle \boldsymbol{A}(\boldsymbol{x}^{n_k-1} - \boldsymbol{x}^{n_k-2}), \boldsymbol{\xi}^\circ - \boldsymbol{\xi}^{n_k}\rangle\\
&\leqslant\frac{1}{2\sigma}\|\boldsymbol{\xi}^\circ - \boldsymbol{\xi}^{n_k}\|_2^2 + \frac{1}{2\tau}\|\boldsymbol{x}^\circ - \boldsymbol{x}^{n_k}\|_2^2.
\end{aligned}$$

We already know that $\|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|_2 = 0$ as $n \to \infty$, $\|\boldsymbol{x}^\circ - \boldsymbol{x}^{n_k}\|_2 = 0$ as $k \to \infty$, and the same for $\boldsymbol{\xi}$. We now see that $\|\boldsymbol{x}^\circ - \boldsymbol{x}^M\|_2 = \|\boldsymbol{\xi}^\circ - \boldsymbol{\xi}^M\|_2 = 0$ as $M \to \infty$, which means that $(\boldsymbol{x}^n)$ in fact converges to a minimizer of (5.2) and completing the proof.

□

This version of the primal-dual algorithm was proposed by Chambolle and Pock in [1], for problems with known saddle-point structure. For general convex functions F^* and G, Nesterov proved in [6] that $\mathcal{O}(1/k)$ is the optimal convergence rate. For a special case where we have strong convexity with known convexity parameter, γ, for either F^* or G, the convergence can reach a rate of $\mathcal{O}(1/k^2)$. Moreover, if both functions are strongly convex, the convergence rate $\mathcal{O}(1/e^k)$ can be achieved.

CHAPTER 6

Nesterov's Algorithm

6.1 Minimization of Non-Smooth Functions

In [5], Nesterov formulated an algorithm for minimizing smooth convex functions with convergence rate of $\mathcal{O}(1/k^2)$, where k is the number of iterations. Nesterov looked at problems of the form,

$$\min_{\boldsymbol{x}\in\mathcal{Q}_p} f(\boldsymbol{x}).$$

Where $f(\boldsymbol{x})$ is a smooth function and $\mathcal{Q}_p$ is a convex set, called the *primal* feasible set. Later, he further developed his algorithm to non-smooth convex functions in [6]. This is the algorithm we refer to as NESTA. The algorithm assumes $f(\boldsymbol{x})$ to be on the general form

$$f(\boldsymbol{x}) = \max_{\boldsymbol{u}\in\mathcal{Q}_d}\langle \boldsymbol{u}, \boldsymbol{W}\boldsymbol{x}\rangle,$$

where $\boldsymbol{x} \in \mathbb{R}^N$, $\boldsymbol{u} \in \mathbb{R}^m$ and $\boldsymbol{W} \in \mathbb{R}^{m\times N}$. $\mathcal{Q}_d$ is the *dual* feasible set, also a convex set. He introduced the prox-function, $p_d(\boldsymbol{u})$, and a smoothing parameter, μ, we find the smoothed approximation to the general problem that Nesterov introduced in [6],

$$f_\mu(\boldsymbol{x}) = \max_{\boldsymbol{u}\in\mathcal{Q}_d}\langle \boldsymbol{u}, \boldsymbol{W}\boldsymbol{x}\rangle - \mu p_d(\boldsymbol{u}).$$

We denote $u_\mu(\boldsymbol{x})$ as the optimal value of f_μ. It is precisely this general form that is the advantage of the NESTA algorithm. Minimizing the function on its general form is formulated as the saddle point problem,

$$\min_{\boldsymbol{x}\in\mathcal{Q}_p}\max_{\boldsymbol{u}\in\mathcal{Q}_d}\langle \boldsymbol{u}, \boldsymbol{W}\boldsymbol{x}\rangle - \mu p_d(\boldsymbol{u}).$$

From this, we will show that ℓ_1-minimization and total variation minimization (TV-minimization) are particular cases of this form. For these problems, the primal feasible set is the same and is based on the constraints of the problem. Depending on whether or not there is noise in the data, which there usually is in real-world problems, the constraints and their feasible sets read

$$\mathcal{Q}_p = \{\boldsymbol{x} \mid \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\},$$
$$\mathcal{Q}_p = \{\boldsymbol{x} \mid \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 \leqslant \eta\}.$$

We will now explore the two types of problems, first ℓ_1-minimization and then TV-minimization.

ℓ_1-Minimization

The ℓ_1-norm is on the form,

$$\|\boldsymbol{x}\|_1 = \max_{\boldsymbol{u}\in\mathcal{Q}_d}\langle \boldsymbol{u}, \boldsymbol{x}\rangle,$$

where, for this problem, the dual feasible set, $\mathcal{Q}_d$ is the ℓ_∞ ball and defined as,

$$\mathcal{Q}_d = \{\boldsymbol{u} \mid \|\boldsymbol{u}\|_\infty \leqslant 1\}.$$

In [9] Candès, Becker and Bobin found $p_d(\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{u}\|_2^2$ to be a good choice for a prox-function for this problem and our function now becomes the Huber function. We wish to find an expression for the gradient. We do so by first considering $f_\mu(\boldsymbol{x})$ to be of the form $f_\mu(\boldsymbol{x}) = \max_{\boldsymbol{u}\in\mathcal{Q}_d} g(\boldsymbol{u})$ and finding the maximum of $g(\boldsymbol{u})$.

$$\begin{aligned}
g(\boldsymbol{u}) &= \langle \boldsymbol{u}, \boldsymbol{x}\rangle - \frac{\mu}{2}\|\boldsymbol{u}\|_2^2 \\
&= \sum_i \left(u_i x_i - \frac{\mu}{2}u_i^2\right) \\
g_i(\boldsymbol{u}) &= u_i x_i - \frac{\mu}{2}u_i^2 \\
g_i'(\boldsymbol{u}) &= x_i - \mu u_i = 0 \\
\Rightarrow u_i &= \frac{x_i}{\mu}.
\end{aligned}$$

g_i is a quadratic function with a global maximum in x_i/μ. Since u_i must be in the interval $[-1, 1]$, we find that the function is split into two cases, whether the maximum is contained in the interval or not. If the maximum is not contained in the interval, the larges value u_i can reach is ± 1, based on where the maximum is, so the maximum becomes $\operatorname{sgn}(x_i)$. If the maximum is contained in the interval, it is the maximum value of u_i.

$$\max_{u_i\in[-1,1]} u_i = \begin{cases} \frac{x_i}{\mu}, & |x_i| < \mu, \\ \operatorname{sgn}(x_i), & |x_i| \geqslant \mu. \end{cases}$$

From this we find an expression for f_μ with a maximizing $\boldsymbol{u}$ and can compute the gradient.

$$f_\mu(\boldsymbol{x}) = \begin{cases} \sum_i \frac{x_i^2}{2\mu}, & |x_i| < \mu, \\ \sum_i |x_i| - \frac{\mu}{2}, & |x_i| \geqslant \mu. \end{cases}$$

$$[\nabla f_\mu(\boldsymbol{x})]_i = \begin{cases} \mu^{-1}x_i, & |x_i| < \mu, \\ \operatorname{sgn}(x_i), & |x_i| \geqslant \mu. \end{cases} \tag{6.1}$$

In Figure 6.1 below we can see the smoothness of f_μ and that it converges to the ℓ_1-norm as μ tends to zero.

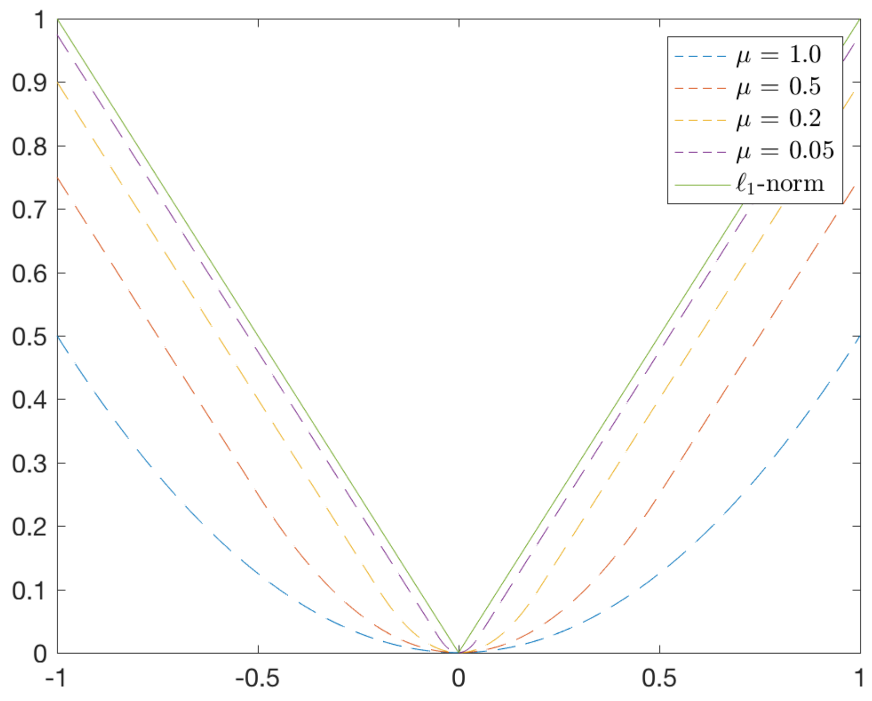

Figure 6.1: Plot of f_μ with decreasing μ's and the ℓ_1-norm.

Total Variation Minimization

Total variation minimization, or total variation *denoising*, was first introduced in 1992 in [4], as a method to remove noise from images. Today it is still mostly used for recovering noisy or under-sampled data. The TV-norm is defined as,

$$\|\boldsymbol{x}\|_{TV} := \sum_{i,j} \|\nabla x[i,j]\|_1, \quad \nabla x[i,j] = \begin{bmatrix} (D_1 x)[i,j] \\ (D_2 x)[i,j] \end{bmatrix}.$$

Where $\boldsymbol{D} = [D_1, D_2]^*$ is a linear and often a very sparse matrix of the horizontal and vertical differences,

$$(D_1 x)[i,j] = x[i+1,j] - x[i,j],$$
$$(D_2 x)[i,j] = x[i,j+1] - x[i,j].$$

Thanks to the framework of NESTA, we can solve these minimizations as well, but we have to rewrite the TV-norm to the general function NESTA works with,

$$\begin{aligned}\|\boldsymbol{x}\|_{TV} &= \sum_{i,j} \|\nabla x[i,j]\|_1 \\ &= \sum_{i,j} \left\| \begin{bmatrix} (D_1 x)[i,j] \\ (D_2 x)[i,j] \end{bmatrix} \right\|_1 \\ &= \|\boldsymbol{Dx}\|_1 \\ &= \max_{\boldsymbol{u}\in\mathcal{Q}_d} \langle \boldsymbol{u}, \boldsymbol{Dx} \rangle.\end{aligned}$$

From this general form, the dual feasible set is the same as for ℓ_1-minimization,

$$\mathcal{Q}_d = \{\boldsymbol{u} \mid \|\boldsymbol{u}\|_\infty \leqslant 1\}.$$

The smoothed version will take the general form,

$$\max_{\boldsymbol{u}\in\mathcal{Q}_d} \langle \boldsymbol{u}, \boldsymbol{Dx} \rangle - \mu p_d(\boldsymbol{u}).$$

As for ℓ_1-minimization, the prox-function, $p_d(\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{u}\|_2^2$, is a good choice. The smoothed function f_μ for this problem becomes,

$$f_\mu(\boldsymbol{x}) = \max_{\boldsymbol{u}\in\mathcal{Q}_d} \langle \boldsymbol{u}, \boldsymbol{Dx} \rangle - \frac{\mu}{2}\|\boldsymbol{u}\|_2^2.$$

This problem is no known function like the function we found for ℓ_1-minimization, but the gradient is computed similarly and takes the form,

$$\nabla f_\mu(\boldsymbol{x}) = \boldsymbol{D}^* u_\mu(\boldsymbol{x}).$$

6.2 Smoothness of f_μ and Lipschitz Constant

The following theorem is from [6], where Nesterov proves smoothness and the Lipschitz constant of the general smoothed function f_μ, where $u_\mu(\boldsymbol{x})$ is the optimal value of f_μ, as we recall. The operator norm we are using is defined in general as,

$$\|\boldsymbol{W}\|_{p,q} := \sup_{\|\boldsymbol{x}\|_p = 1} \|\boldsymbol{Wx}\|_q. \tag{6.2}$$

Theorem 6.1. *$f_\mu(\boldsymbol{x})$ is well defined and continuously differentiable at any $\boldsymbol{x} \in \mathbb{R}^N$. Moreover, this function is convex and its gradient*

$$\nabla f_\mu(\boldsymbol{x}) = \boldsymbol{W}^* u_\mu(\boldsymbol{x})$$

is Lipschitz continuous with constant

$$L_\mu = \frac{1}{\mu\sigma_d}\|\boldsymbol{W}\|_{1,2}^2.$$

Proof. We start by proving $\nabla f_\mu(\boldsymbol{x}) = \boldsymbol{W}^* u_\mu(\boldsymbol{x})$. First, we need to see that f_μ is in fact convex and continuously differentiable. $f_\mu(\boldsymbol{x})$ is defined as:

$$f_\mu(\boldsymbol{x}) = \max_{\boldsymbol{u}\in\mathcal{Q}_d} \langle \boldsymbol{u}, \boldsymbol{W}\boldsymbol{x}\rangle - \mu p_d(\boldsymbol{u}).$$

From Proposition 2.10, we know that the maximum of convex functions are convex, so we need to show that the functions defining $f_\mu(\boldsymbol{x})$ are convex. $p_d(\boldsymbol{u})$ is the prox-function and we know from Definition 2.15 that all prox-functions are strongly convex and $\langle \boldsymbol{u}, \boldsymbol{W}\boldsymbol{x}\rangle$ is the ℓ_1-norm we know to be convex. With this we can conclude that $f_\mu(\boldsymbol{x})$ is convex. The function is continuously differentiable since u_μ is unique. Note that u_μ must satisfy

$$\boldsymbol{W}\boldsymbol{x} - \mu \nabla p_d(\boldsymbol{u}) = 0,$$

as it is the optimal value of the maximum in f_μ. We can now calculate the gradient of $f_\mu(\boldsymbol{x})$:

$$\begin{aligned}
\nabla f_\mu(\boldsymbol{x}) &= \nabla \langle \boldsymbol{W}\boldsymbol{x}, u_\mu(\boldsymbol{x})\rangle - \nabla(\mu p_d(u_\mu(\boldsymbol{x}))) \\
&= \boldsymbol{W}^* u_\mu(\boldsymbol{x}) + \boldsymbol{W}\boldsymbol{x}\nabla u_\mu(\boldsymbol{x}) - \mu \nabla p_d(u_\mu(\boldsymbol{x}))\nabla u_\mu(\boldsymbol{x}) \\
&= \boldsymbol{W}^* u_\mu(\boldsymbol{x}) + \nabla u_\mu(\boldsymbol{x})\big(\boldsymbol{W}\boldsymbol{x} - \mu\nabla p_d(u_\mu(\boldsymbol{x}))\big) \\
&= \boldsymbol{W}^* u_\mu(\boldsymbol{x}).
\end{aligned}$$

This concludes the first part of the theorem.

As we recall from Definition 2.6, the Lipschitz constant of a function f, is the constant L such that:

$$\|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)\| \leqslant L\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|.$$

Before we find an expression for L, we do some calculations that will help us on the way. From the first-order optimality conditions (first-order term of the Taylor series of f_μ), we have:

$$\begin{aligned}
\langle \boldsymbol{W}\boldsymbol{x}_1 - \mu\nabla p_d(u_\mu(\boldsymbol{x}_1)), u_\mu(\boldsymbol{x}_2) - u_\mu(\boldsymbol{x}_1)\rangle \leqslant 0, \\
\langle \boldsymbol{W}\boldsymbol{x}_2 - \mu\nabla p_d(u_\mu(\boldsymbol{x}_2)), u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\rangle \leqslant 0.
\end{aligned}$$

We want to rewrite this by adding the two equations. For simpler notation, let $\boldsymbol{x}_1 = \boldsymbol{W}\boldsymbol{x}_1$, $\boldsymbol{y}_1 = \mu\nabla p_d(u_\mu(\boldsymbol{x}_1))$ and $\boldsymbol{z} = u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)$, equivalent for $\boldsymbol{x}_2$ and $\boldsymbol{y}_2$.

$$\begin{aligned}
\langle \boldsymbol{x}_1 - \boldsymbol{y}_1, -\boldsymbol{z}\rangle + \langle \boldsymbol{x}_2 - \boldsymbol{y}_2, \boldsymbol{z}\rangle &\leqslant 0 \\
\langle -\boldsymbol{x}_1 + \boldsymbol{y}_1, \boldsymbol{z}\rangle + \langle \boldsymbol{x}_2 - \boldsymbol{y}_2, \boldsymbol{z}\rangle &\leqslant 0 \\
\langle (-\boldsymbol{x}_1 + \boldsymbol{x}_2) + (\boldsymbol{y}_1 - \boldsymbol{y}_2), \boldsymbol{z}\rangle &\leqslant 0 \\
-\langle \boldsymbol{x}_1 - \boldsymbol{x}_2, \boldsymbol{z}\rangle + \langle \boldsymbol{y}_1 - \boldsymbol{y}_2, \boldsymbol{z}\rangle &\leqslant 0 \\
\langle \boldsymbol{x}_1 - \boldsymbol{x}_2, \boldsymbol{z}\rangle &\geqslant \langle \boldsymbol{y}_1 - \boldsymbol{y}_2, \boldsymbol{z}\rangle.
\end{aligned}$$

Using Theorem 2.9, we have

$$\begin{aligned}\langle \boldsymbol{W}(\boldsymbol{x}_1 - \boldsymbol{x}_2), u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\rangle \geqslant & \langle \mu(\nabla p_d(u_\mu(\boldsymbol{x}_1)) - \nabla p_d(u_\mu(\boldsymbol{x}_2))), \\ & u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\rangle \\ \geqslant & \mu\langle \nabla p_d(u_\mu(\boldsymbol{x}_1)) - \nabla p_d(u_\mu(\boldsymbol{x}_2)), \\ & u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\rangle.\end{aligned}$$

The prox-function and its gradient is bounded by,

$$\begin{aligned}p_d(\boldsymbol{u}) \geqslant & \frac{\sigma_d}{2}\|\boldsymbol{u} - \boldsymbol{u}_0\|_2^2, \\ \nabla p_d(\boldsymbol{u}) \geqslant & \sigma_d(\boldsymbol{u} - \boldsymbol{u}_0).\end{aligned}$$

Thus,

$$\begin{aligned}&\mu\langle \nabla p_d(u_\mu(\boldsymbol{x}_1)) - \nabla p_d(u_\mu(\boldsymbol{x}_2)), \\ &\qquad u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\rangle \geqslant \mu\langle \sigma_d(u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)), u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\rangle \\ &\qquad = \mu\sigma_d\|u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\|_2^2.\end{aligned}$$

These minor calculations will help us through the last step of the proof,

$$\begin{aligned}\|\boldsymbol{W}^*\boldsymbol{u}\|_1^* &= \max_{\boldsymbol{x}}\{\langle \boldsymbol{W}^*\boldsymbol{u}, \boldsymbol{x}\rangle \mid \|\boldsymbol{x}\|_1 = 1\} \\ &\leqslant \max_{\boldsymbol{x},\boldsymbol{u}}\{\langle \boldsymbol{W}\boldsymbol{x}, \boldsymbol{u}\rangle \mid \|\boldsymbol{x}\|_1 = 1, \|\boldsymbol{u}\|_2 = 1\} \\ &\leqslant \|\boldsymbol{W}\|_{1,2} \cdot \|\boldsymbol{u}\|_2. \qquad (6.3)\end{aligned}$$

Finally, by (6.3), $\langle \boldsymbol{u}, \boldsymbol{W}\boldsymbol{x}\rangle = \langle \boldsymbol{x}, \boldsymbol{W}^*\boldsymbol{u}\rangle_1$, Cauchy-Schwarz inequality and what we just calculated,

$$\begin{aligned}(\|\boldsymbol{W}^*u_\mu(\boldsymbol{x}_1) - \boldsymbol{W}^*u_\mu(\boldsymbol{x}_2)\|_1^*)^2 &\leqslant \|\boldsymbol{W}\|_{1,2}^2 \cdot \|u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)\|_2^2 \\ &\leqslant \frac{1}{\mu\sigma_d}\|\boldsymbol{W}\|_{1,2}^2\langle \boldsymbol{W}^*(u_\mu(\boldsymbol{x}_1) - u_\mu(\boldsymbol{x}_2)), \boldsymbol{x}_1 - \boldsymbol{x}_2\rangle \\ &\leqslant \frac{1}{\mu\sigma_d}\|\boldsymbol{W}\|_{1,2}^2 \cdot \|\boldsymbol{W}^*u_\mu(\boldsymbol{x}_1) - \boldsymbol{W}^*u_\mu(\boldsymbol{x}_2)\|_1^* \\ &\qquad\qquad \cdot \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_1,\end{aligned}$$

concluding the proof altogether. □

6.3 The Steps in NESTA for Compressive Sensing Problems

As mentioned in the beginning of the chapter, Nesterov introduced an algorithm for minimizing any smooth convex function in [5], which was later improved to handle non-smooth functions as well. The algorithm estimates two sequences, $\{\boldsymbol{y}_k\}$ and $\{\boldsymbol{z}_k\}$, and combines them in a weighted average to update the will-be solution, $\{\boldsymbol{x}_k\}$. The algorithm makes use of two scalar sequences as well, $\{\alpha_k\}$ and $\{\tau_k\}$, that we will discuss when exploring the algorithm.

Algorithm 3 NESTA as formulated in [9]

```
Initialize $\boldsymbol{x}_0$. For $k \geqslant 0$
    1. Compute $\nabla f_\mu(\boldsymbol{x}_k)$
    2. Compute $\boldsymbol{y}_k$
       $\boldsymbol{y}_k = \operatorname{argmin}_{\boldsymbol{x}\in\mathcal{Q}_p} \frac{L_\mu}{2}\|\boldsymbol{x}-\boldsymbol{x}_k\|_2^2 + \langle\nabla f_\mu(\boldsymbol{x}_k), \boldsymbol{x}-\boldsymbol{x}_k\rangle$
    3. Compute $\boldsymbol{z}_k$
       $\boldsymbol{z}_k = \operatorname{argmin}_{\boldsymbol{x}\in\mathcal{Q}_p} \frac{L_\mu}{\sigma_p} p_p(\boldsymbol{x}) + \sum_{i=0}^{k} \alpha_i\langle\nabla f_\mu(\boldsymbol{x}_i), \boldsymbol{x}-\boldsymbol{x}_i\rangle$
    4. Update $\boldsymbol{x}_k$
       $\boldsymbol{x}_k = \tau_k\boldsymbol{z}_k + (1-\tau_k)\boldsymbol{y}_k$
Stop when a given criterion is met.
```

We start by choosing an initial guess of the optimal solution, $\boldsymbol{x}_0$. When nothing is known, a good choice can be $\boldsymbol{x}_0 = \boldsymbol{A}^*\boldsymbol{b}$.

Step 1. Compute $\nabla f_\mu(\boldsymbol{x}_k)$

Compute $\nabla f_\mu(\boldsymbol{x}_k)$, the gradient of the smoothed function at $\boldsymbol{x}_k$. From [9], we know that a convenient choice of the dual prox-function is $p_d(\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{u}\|_2^2$. This applies for both ℓ_1-minimization and TV-minimization. With this prox-function, the gradient of f_μ, as we recall from (6.1),

$$[\nabla f_\mu(\boldsymbol{x})]_i = \begin{cases} \mu^{-1}x_i, & |x_i| < \mu, \\ \operatorname{sgn}(x_i), & |x_i| \geqslant \mu. \end{cases}$$

Step 2. Update $\boldsymbol{y}_k$

Update $\boldsymbol{y}_k$,

$$\boldsymbol{y}_k = \operatorname*{argmin}_{\boldsymbol{x}\in\mathcal{Q}_p} \frac{L_\mu}{2}\|\boldsymbol{x}-\boldsymbol{x}_k\|_2^2 + \langle\nabla f_\mu(\boldsymbol{x}_k), \boldsymbol{x}-\boldsymbol{x}_k\rangle.$$

In order to compute $\boldsymbol{y}_k$, we can use the Lagrangian of the problem to help us find the minimum. We recall that Lagrangian is defined as,

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}g(\boldsymbol{x}),$$

and for our problem it becomes,

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{L_\mu}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|_2^2 + \langle \nabla f_\mu(\boldsymbol{x}_k), \boldsymbol{x} - \boldsymbol{x}_k\rangle + \frac{\boldsymbol{\lambda}}{2}\big(\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 - \eta^2\big).$$

From our problem, we see that the KKT-conditions can be written like this, at the primal-dual solution $(\boldsymbol{y}_k, \boldsymbol{\lambda}_\eta)$

$$\begin{aligned}
\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 &\leqslant \eta^2 && \text{(primal feasibility)},\\
\boldsymbol{\lambda}_\eta\big(\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 - \eta^2\big) &= \boldsymbol{0} && \text{(complementary slackness)},\\
\boldsymbol{\lambda}_\eta &\geqslant \boldsymbol{0} && \text{(dual feasibility)},\\
L_\mu(\boldsymbol{y}_k - \boldsymbol{x}_k) + \boldsymbol{\lambda}_\eta \boldsymbol{A}^*(\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{b}) + \nabla f_\mu(\boldsymbol{x}_k) &= \boldsymbol{0} && \text{(stationary)}.
\end{aligned}$$

We rewrite the stationary condition and solve for $\boldsymbol{y}_k$,

$$\begin{aligned}
&L_\mu \boldsymbol{y}_k - L_\mu \boldsymbol{x}_k + \boldsymbol{\lambda}_\eta \boldsymbol{A}^*\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{\lambda}_\eta \boldsymbol{A}^*\boldsymbol{b} + \nabla f_\mu(\boldsymbol{x}_k) = \boldsymbol{0}\\
&L_\mu \boldsymbol{y}_k + \boldsymbol{\lambda}_\eta \boldsymbol{A}^*\boldsymbol{A}\boldsymbol{y}_k = \boldsymbol{\lambda}_\eta \boldsymbol{A}^*\boldsymbol{b} - \nabla f_\mu(\boldsymbol{x}_k) + L_\mu \boldsymbol{x}_k \qquad \Big|\cdot L_\mu^{-1}\\
&\Big(\boldsymbol{I} + \frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{A}\Big)\boldsymbol{y}_k = \frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{b} + \boldsymbol{x}_k - \frac{1}{L_\mu}\nabla f_\mu(\boldsymbol{x}_k)\\
&\boldsymbol{y}_k = \Big(\boldsymbol{I} + \frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{A}\Big)^{-1}\Big(\frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{b} + \boldsymbol{x}_k - \frac{1}{L_\mu}\nabla f_\mu(\boldsymbol{x}_k)\Big).
\end{aligned}$$

Where we can rewrite again the inverse term,

$$\begin{aligned}
\Big(\boldsymbol{I} + \frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{A}\Big)^{-1} &= \Big(-\frac{L_\mu}{\boldsymbol{\lambda}_\eta}\Big)\bigg(\Big(-\frac{L_\mu}{\boldsymbol{\lambda}_\eta}\Big)\boldsymbol{I} + \frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{A}\bigg)^{-1}\\
&= \Big(-\frac{L_\mu}{\boldsymbol{\lambda}_\eta}\Big)\bigg(\Big(-\frac{\boldsymbol{\lambda}_\eta}{L_\mu}\Big)\boldsymbol{I} + \frac{1}{(-\frac{L_\mu}{\boldsymbol{\lambda}_\eta})(-\frac{L_\mu}{\boldsymbol{\lambda}_\eta} - 1)}\boldsymbol{A}^*\boldsymbol{A}\bigg)\\
&= \boldsymbol{I} - \frac{\boldsymbol{\lambda}_\eta}{L_\mu + \boldsymbol{\lambda}_\eta}\boldsymbol{A}^*\boldsymbol{A}.
\end{aligned}$$

So, our final expression for $\boldsymbol{y}_k$ is,

$$\boldsymbol{y}_k = \Big(\boldsymbol{I} - \frac{\boldsymbol{\lambda}_\eta}{L_\mu + \boldsymbol{\lambda}_\eta}\boldsymbol{A}^*\boldsymbol{A}\Big)\Big(\frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{b} + \boldsymbol{x}_k - \frac{1}{L_\mu}\nabla f_\mu(\boldsymbol{x}_k)\Big).$$

From the stationary conditions, we can find an explicit expression for $\boldsymbol{\lambda}_\eta$ as well,

$$\begin{aligned}
&L_\mu \boldsymbol{y}_k - L_\mu \boldsymbol{x}_k + \boldsymbol{\lambda}_\eta \boldsymbol{A}^*\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{\lambda}_\eta \boldsymbol{A}^*\boldsymbol{b} + \nabla f_\mu(\boldsymbol{x}_k) = \boldsymbol{0} \qquad \Big|\cdot L_\mu^{-1}\\
&\boldsymbol{y}_k - \boldsymbol{x}_k + L_\mu^{-1}\boldsymbol{\lambda}_\eta \boldsymbol{A}^*(\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{b}) + L_\mu^{-1} = \boldsymbol{0}\\
&L_\mu^{-1}\boldsymbol{\lambda}_\eta \boldsymbol{A}^*(\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{b}) + \boldsymbol{y}_k = \underbrace{\boldsymbol{x}_k + L_\mu^{-1}\nabla f_\mu(\boldsymbol{x}_k)}_{=\boldsymbol{q}} \qquad \Big|\cdot \boldsymbol{A}\\
&L_\mu^{-1}\boldsymbol{\lambda}_\eta(\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{b}) + \boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{b} = \boldsymbol{A}\boldsymbol{q} - \boldsymbol{b} \qquad \Big|\cdot L_\mu\\
&(\boldsymbol{\lambda}_\eta + L_\mu)(\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{b}) = L_\mu(\boldsymbol{A}\boldsymbol{q} - \boldsymbol{b})\\
&(\boldsymbol{\lambda}_\eta + L_\mu)\eta = L_\mu\|\boldsymbol{A}\boldsymbol{q} - \boldsymbol{b}\|_2\\
&\boldsymbol{\lambda}_\eta = L_\mu(\eta^{-1}\|\boldsymbol{A}\boldsymbol{q} - \boldsymbol{b}\|_2 - 1).
\end{aligned}$$

Step 3. Update z_k

Update $\boldsymbol{z}_k$,

$$\boldsymbol{z}_k = \operatorname*{argmin}_{\boldsymbol{x}\in\mathcal{Q}_p} \frac{L_\mu}{\sigma_p} p_p(\boldsymbol{x}) + \langle \sum_{i=0}^{k} \alpha_i \nabla f_\mu(\boldsymbol{x}_i), \boldsymbol{x} - \boldsymbol{x}_i \rangle.$$

The prox-function is chosen based on $\mathcal{Q}_p$, it is strongly convex and smooth, as we recall. In the NESTA article, [9], the following function has been found to work well, which we know to be strongly convex by Proposition 2.11:

$$p_p(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}_0\|_2^2.$$

Making similar calculations as in step 2, we find that $\boldsymbol{z}_k$ can be expressed as,

$$\boldsymbol{z}_k = \Big(\boldsymbol{I} - \frac{\boldsymbol{\lambda}_\eta}{L_\mu + \boldsymbol{\lambda}_\eta}\boldsymbol{A}^*\boldsymbol{A}\Big)\Big(\frac{\boldsymbol{\lambda}_\eta}{L_\mu}\boldsymbol{A}^*\boldsymbol{b} + \boldsymbol{x}_0 - \frac{1}{L_\mu}\sum_{i\leqslant k}\nabla\alpha_i f_\mu(\boldsymbol{x}_i)\Big).$$

For the Lagrange multiplier, we find the expression, as in step 2,

$$\boldsymbol{\lambda}_\eta = L_\mu(\eta^{-1}\|\boldsymbol{A}\boldsymbol{q} - \boldsymbol{b}\|_2 - 1), \qquad \boldsymbol{q} = \boldsymbol{x}_0 - L_\mu^{-1}\sum_{i\leqslant k}\alpha_i\nabla f_\mu(\boldsymbol{x}_i).$$

Step 4. Update x_k

Update $\boldsymbol{x}_k$,

$$\boldsymbol{x}_k = \tau_k\boldsymbol{z}_k + (1 - \tau_k)\boldsymbol{y}_k.$$

$\boldsymbol{x}_k$ becomes the weighted average of $\boldsymbol{z}$ and $\boldsymbol{y}_k$. Nesterov showed in [6, thm. 2] that with $\alpha_k = {}^1\!/_{2(k+1)}$ and $\tau_k = {}^2\!/_{(k+3)}$, the algorithm converges with the rate $\frac{L}{k^2}$.

6.4 Continuation

Continuation is a way to speed up the convergence. The idea is quite simple, instead of running the algorithm with only the chosen μ, which we will denote μ_f, we make a sequence of μ's. Algorithm 4 computes the sequence of μ's, T is the number of continuation steps, t is the iterator from 1 to T, γ is the step factor based on T, μ_0 and μ_f and is defined as $\gamma = ({}^{\mu_f}\!/_{\mu_0})^{1/T}$. So why does this speed up the convergence? The idea of computing a sequence of decreasing μ's is based on a technique from [2], for solving least squares problems where λ is in the range $0 < \lambda < \|\boldsymbol{A}^*\boldsymbol{b}\|_\infty$. It was shown to be faster to compute the algorithm several times while using the intermediate solution as the initial guess of the optimal solution until the chosen μ_f is reached, and the desired accuracy is acquired. Continuation has been observed to have a particularly good effect on large problems. In [9], it was observed that more steps are not necessarily better. Below we see two graphs computed using the NESTA algorithm in MATLAB from [8], and a

Algorithm 4 Continuation algorithm from [9]

```
Initialize $\mu_0$, $x_0$ and number of continuation steps $T$.
For $1 \leqslant t \leqslant T$
    1. Apply NESTA with $\mu = \mu^{(t)}$ and $x_0 = x_{\mu^{(t-1)}}$
    2. Decrease $\mu$: $\mu^{(t+1)} = \gamma\mu^{(t)}$
Stop when $\mu_f$ is reached.
```

bar chart. The left graph shows how the algorithm converges to the desired solution with five continuation steps and without any continuation steps. We see that continuation have a huge impact on the number of iterations to find the optimal solution. The left graph shows the convergence with different number of continuation steps. The bar chart shows the number of iterations for different number of continuation steps. As we would expect, no continuation steps requires the most iterations. However, for this case, 2 loops, or 3 steps, require the least amount of iterations. The MATLAB program has been run on an image with dynamic range of 60 dB.

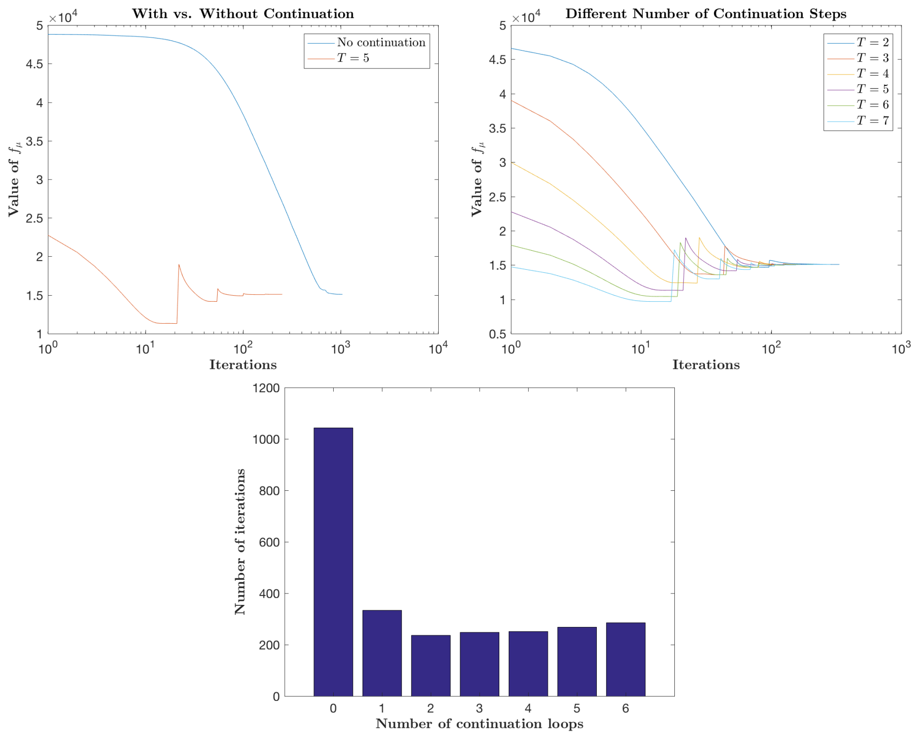

Figure 6.2: Top left: NESTA without continuation and with five continuation steps. Top right: NESTA with different number of continuation steps. Bottom: Number of iterations for varying number of continuation loops.

Convergence of Continuation

Theorem 6.2 from [9, thm. 3.1], states that each continuation step converges to the local optimal solution, which is denoted by $\boldsymbol{x}^*_{\mu^{(t)}}$. The global optimal solution is found by setting $t = T$.

Theorem 6.2. *At each continuation step,* $\lim_{k\to\infty} \boldsymbol{y}_k = \boldsymbol{x}^*_{\mu^{(t)}}$*, and*

$$f_{\mu^{(t)}}(\boldsymbol{y}_k) - f_{\mu^{(t)}}(\boldsymbol{x}^*_{\mu^{(t)}}) \leqslant \frac{2L_{\mu^{(t)}}\|\boldsymbol{x}^*_{\mu^{(t)}} - \boldsymbol{x}_{\mu^{(t-1)}}\|_2^2}{k^2}.$$

Proof. By [6, thm. 2] □

From this theorem, we can find the number of iterations necessary to achieve the desired accuracy, $\gamma^t\delta_0$. We want,

$$f_{\mu^{(t)}}(\boldsymbol{y}_k) - f_{\mu^{(t)}}(\boldsymbol{x}^*_{\mu^{(t)}}) \leqslant \gamma^t\delta_0,$$

which we can rewrite in the following way,

$$\gamma^t\delta_0 \leqslant \frac{2L_{\mu^{(t)}}\|\boldsymbol{x}^*_{\mu^{(t)}} - \boldsymbol{x}_{\mu^{(t-1)}}\|_2^2}{k^2}$$
$$k \leqslant \sqrt{\frac{2L_\mu}{\gamma^t\delta_0}}\|\boldsymbol{x}^*_{\mu^{(t)}} - \boldsymbol{x}_{\mu^{(t-1)}}\|_2.$$

By rewriting with the following: L_μ with $1/\mu_f$ and μ_f with $\gamma^t\mu_0$, and summing over the continuation steps, we find

$$\mathcal{N}_c = \sqrt{\frac{2}{\mu_0\delta_0}}\sum_{t=1}^{T}\gamma^{-t}\|\boldsymbol{x}^*_{\mu^{(t)}} - \boldsymbol{x}_{\mu^{(t-1)}}\|_2.$$

We have denoted $\mathcal{N}_c$, as the number of iterations with continuation and $\mathcal{N}$, as the number of iterations without continuation. When we do not use continuation, the number of iterations is found from,

$$\mathcal{N} = \sqrt{\frac{2}{\mu_0\delta_0}}\gamma^{-T}\|\boldsymbol{x}^*_f - \boldsymbol{x}_0\|_2.$$

A significant result we can derive from the number of iterations is whether the solution path is smooth or not, which again will tell us if continuation is profitable. Meaning that if all intermediate solutions are part of a segment $[\boldsymbol{x}_0, \boldsymbol{x}_{\mu_f}]$ in order, then

$$\sum_{t=1}^{T}\|\boldsymbol{x}^*_{\mu^{(t)}} - \boldsymbol{x}_{\mu^{(t-1)}}\|_2 = \|\boldsymbol{x}^*_{\mu_f} - \boldsymbol{x}_0\|_2,$$

and continuation requires fewer iterations than the regular algorithm. If the intermediate solutions do not lie on the segment, we can think of it as being a "detour" in search of the optimal solution.

CHAPTER 7

Summary

In this thesis, we started by introducing the field of compressive sensing in Chapter 1 and proving NP-hardness of ℓ_0-minimization in Chapter 2. We continued to find that the convex relaxation, ℓ_1-minimization, is a good substitute and in fact the only minimizer that both guarantees a solution and is not NP-hard to compute. In Chapter 3, we defined the recovery guarantee, null space property, and the quality measures, coherence and restricted isometry property. These measures are essential when working with the algorithms associated with compressive sensing, for which we gave a brief overview of in Chapter 4.

In Chapter 5, we derived the well-established primal-dual algorithm proposed by Chambolle and Pock in [1] and proved that convergence is guaranteed by Theorem 5.3. The primal-dual algorithm may achieve a convergence rate of $\mathcal{O}(1/k^2)$, for a special case of the convex functions F^* and G, and a slight modification of the algorithm. Whereas Nesterov proved in [6], that the convergence rate $\mathcal{O}(1/k)$, is optimal for this algorithm with general convex functions.

In Chapter 6, we followed Nesterov's development of the algorithm for minimizing non-smooth convex functions in [6], which achieves the same convergence rate of $\mathcal{O}(1/k^2)$, as his algorithm for smooth convex functions introduced in [5]. We formulated the general function both for ℓ_1-minimization and for total variation minimization. In Section 6.4, we introduced the concept of continuation, which, for some problems, speeds up the convergence considerably. It is difficult to give some exact rate of convergence with continuation, as it depends heavily on the problem at hand. For the example we observed in Figure 6.2, we see that continuation can improve the convergence by a factor of about 4.4, for this dynamic range.

Bibliography

[1] T. Pock A. Chambolle. “A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging”. In: (2010).

[2] W. Yin E. T. Hale and Y. Zhang. “A Fixed-Point Continuation Method for ℓ_1-Regularized Minimization with Applications to Compressed Sensing”. In: (2007).

[3] M. Friedlander E. van den Berg. “Probing the Pareto Frontier for Basis Pursuit Solutions”. In: (2008).

[4] S. Osher L. Rudin and E. Fatemi. “Nonlinear Total Variation Based Noise Removal Algorithms”. In: (1992).

[5] Yu. Nesterov. “A Method of Solving a Convex Programming Problem with Convergence Rate $(1/k^2)$”. In: (1983), pp. 372–376.

[6] Yu. Nesterov. “Smooth Minimization of Non-Smooth Functions”. In: (2005), pp. 127–152.

[7] Ø. Ryan. *Linear algebra, Signal Processing, and Wavelets. A Unified Approach*. 2017.

[8] E.J. Candès S. Becker J. Bobin. *NESTA: A Fast and Accurate First-Order Method for Sparse Recovery*. 2017-30-08. URL: `https://statweb.stanford.edu/~candes/nesta/`.

[9] E.J. Candès S. Becker J. Bobin. “NESTA: A Fast and Accurate First-Order Method for Sparse Recovery”. In: (2011), pp. 1–37.

[10] L. Vandenberghe S. Boyd. *Convex Optimization*. Cambridge University Press, 2004.

[11] H. Rauhut S. Foucart. *A Mathematical Introduction to Compressive Sensing*. Springer - Birkhäuser, 2013. ISBN: 978-0-8176-4947-0.

[12] R.J. Vanderbei. *Linear Programming*. Third. Foundations and extensions. Springer, New York, 2014. ISBN: 978-1-4614-7629-0; 978-1-4614-7630-6.